Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων

Εργαστηριο Υπολογιστικων Συστηματων

Σύστημα Αυθεντικοποίησης Φοιτητών και Έκδοσης Πιστοποιητικών πάνω στο Δίκτυο Cardano

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΔΡΕΑΣ ΜΑΝΤΖΟΥΤΑΣ

Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2024

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών
Εργαστήριο Υπολογιστικών Συστημάτων

Σύστημα Αυθεντικοποίησης Φοιτητών και Έκδοσης Πιστοποιητικών πάνω στο Δίκτυο Cardano

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΔΡΕΑΣ ΜΑΝΤΖΟΥΤΑΣ

Επιβλέπων: Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15η Ιουλίου, 2024.

.....	.....	.....
Νεκτάριος Κοζύρης	Άγγελος Κιαγιάς	Αριστείδης Παγουρτζής
Καθηγητής Ε.Μ.Π.	Καθηγητής Πανεπιστήμιο του Εδιμβούργου	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2024

.....

Μαντζουτας Ανδρεας
Διπλωματούχος Ηλεκτρολόγος Μηχανικός
και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Τα τελευταία χρόνια, η τεχνολογία blockchain έχει αναδειχθεί σε μετασχηματιστική δύναμη σε διάφορους τομείς, προσφέροντας ένα αποκεντρωμένο, απαραβίαστο αρχείο που ενισχύει τη διαφάνεια, την ασφάλεια και την αποτελεσματικότητα. Ξεκινώντας με την εισαγωγή του Bitcoin από τον Satoshi Nakamoto το 2008, η τεχνολογία blockchain αμφισβήτησε τα παραδοσιακά χρηματοπιστωτικά συστήματα και τώρα υπόσχεται να φέρει επανάσταση και σε άλλους κλάδους, συμπεριλαμβανομένου του ακαδημαϊκού χώρου. Οι παραδοσιακές μέθοδοι διαχείρισης των ακαδημαϊκών αρχείων μαστίζονται από προκλήσεις όπως ο σημαντικός διοικητικός φόρτος, η ευαισθησία σε ανθρώπινα σφάλματα και οι κίνδυνοι πλαστογράφησης εγγράφων.

Η παρούσα έρευνα μελετά την εφαρμογή της τεχνολογίας blockchain στη διαχείριση των ακαδημαϊκών αρχείων, προτείνοντας ένα σύστημα που αξιοποιεί την εγγενή ασφάλεια και τη διαφάνειά της για να ανατρέψει τις υπάρχουσες πρακτικές. Με την ενσωμάτωση των ακαδημαϊκών πιστοποιητικών στο blockchain, τα εκπαιδευτικά ιδρύματα μπορούν να μειώσουν δραστικά τα διοικητικά έξοδα, να εξαλείψουν τα ανθρώπινα λάθη και να διασφαλίσουν το αναλλοίωτο των αρχείων. Επιπλέον, ο αμετάβλητος και δημόσιος χαρακτήρας της εν λόγω τεχνολογίας εξασφαλίζει ότι η επαλήθευση των βαθμών, των πιστοποιητικών και των πτυχίων είναι αξιόπιστη, επιτρέποντας την άμεση επικύρωση από εργοδότες, άλλα ιδρύματα και οργανισμούς, χωρίς μεσάζοντες.

Εστιάζοντας στην πρακτική εφαρμογή, η παρούσα μελέτη περιγράφει λεπτομερώς την ανάπτυξη ενός συστήματος βασισμένου στο Cardano Blockchain για τη διαχείριση της πιστοποίησης ταυτότητας των φοιτητών και την έκδοση πιστοποιητικών. Το προτεινόμενο σύστημα χρησιμοποιεί έξυπνα συμβόλαια για την αυτοματοποίηση της έκδοσης και επαλήθευσης των ακαδημαϊκών πιστοποιητικών, διασφαλίζοντας ότι αυτά είναι επαληθεύσιμα και αμετάβλητα. Επιπλέον, η αρχιτεκτονική του συστήματος έχει σχεδιαστεί για να εξυπηρετεί τις ειδικές ανάγκες των εκπαιδευτικών ιδρυμάτων, από την καταχώρηση των βαθμών των φοιτητών έως την έκδοση πτυχίων, ενισχύοντας την ακεραιότητα και την αποτελεσματικότητα των εκπαιδευτικών διοικητικών διαδικασιών. Με την υιοθέτηση της τεχνολογίας blockchain, τα εκπαιδευτικά ιδρύματα μπορούν να στρέψουν την προσοχή τους στην παροχή ποιοτικής εκπαίδευσης, διασφαλίζοντας παράλληλα ότι τα ακαδημαϊκά επιτεύγματα καταγράφονται με ασφάλεια και επαληθεύονται αβίαστα, προαναγγέλλοντας μια νέα εποχή στη διαχείριση των ακαδημαϊκών αρχείων.

Λέξεις-κλειδιά — Blockchain, Έξυπνα Συμβόλαια, Cardano, Αποκεντρωμένες Εφαρμογές, Αποκεντρωμένα Αναγνωριστικά, ProofSpace

Abstract

In recent years, blockchain technology has emerged as a transformative force in various sectors, offering a decentralized, tamper-proof ledger that enhances transparency, security, and efficiency. Originating with Satoshi Nakamoto's introduction of Bitcoin in 2008, blockchain has challenged traditional financial systems and now promises to revolutionize other industries, including academia. Traditional methods of managing academic records are plagued with challenges such as significant administrative burdens, error susceptibility due to human involvement, and risks of document forgery.

This thesis explores the application of blockchain technology to the management of academic records, proposing a system that leverages its inherent security and transparency to overhaul current practices. By embedding academic credentials on the blockchain, educational institutions can drastically reduce administrative overhead, eliminate manual errors, and secure records against tampering. Moreover, the immutable and public nature of blockchain ensures that verification of grades, certificates, and degrees is reliable, allowing instant validation by employers, other institutions, and agencies without intermediaries.

Focusing on a practical implementation, this study details the deployment of a blockchain-based system on the Cardano network to manage student authentication and certificate issuance. The proposed system utilizes smart contracts to automate the issuing and verification of academic credentials, ensuring that they are verifiable and immutable. Additionally, the system's architecture is designed to accommodate the specific needs of educational institutions, from registering student grades to issuing degrees, enhancing the integrity and efficiency of educational administrative processes. By adopting blockchain technology, educational institutions can shift their focus towards delivering quality education while ensuring that academic achievements are securely recorded and effortlessly verifiable, heralding a new era in the management of academic records.

Keywords — Blockchain, Smart Contracts, Cardano, Aiken, Decentralized Applications, Decentralized Identifiers, ProofSpace

Ευχαριστίες

Με την παράδοση της παρούσας εργασίας, το ταξίδι μου ως φοιτητής στη σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Ηλεκτρονικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου φτάνει στο τέλος του. Σε όλο αυτό το ταξίδι, πολλοί μου συμπαραστάθηκαν, με ενέπνευσαν και με υποστήριξαν.

Θα ήθελα ιδιαίτερα να ευχαριστήσω τον Καθηγητή κο Νεκτάριο Κοζύρη, που μου έδωσε την ευκαιρία να εκπονήσω την διπλωματική μου εργασία στο Εργαστήριο Υπολογιστικών Συστημάτων, καθώς και την Διδάκτορα κα Κατερίνα Δόκα, τον Υποψήφιο Διδάκτορα κο Τάσο Κατσιγιάννη και τον κο Χρήστο Παλάσκα για όλη την υποστήριξη και καθοδήγηση που μου παρείχαν κατά τη διάρκεια αυτής της έρευνας και πορείας.

Είμαι ευγνώμων για την υποτροφία που έλαβα από την εταιρεία IOG, χάρη στην οποία κατέστη δυνατή η χρήση του λογισμικού ProofSpace, καθώς και για την άμεση βοήθεια που μου παρείχε η ομάδα του ProofSpace όποτε χρειάστηκε.

Δεν θα μπορούσα να μην ευχαριστήσω τους γονείς μου, Σπύρο και Αγγελική, καθώς και τα υπόλοιπα μέλη της οικογένειας μου, Νίκη, Δημήτρη και Κατερίνα, που με αγαπούν και με στηρίζουν πάντοτε.

Είμαι τυχερός που σε αυτό το ταξίδι είχα συνοδοιπόρους τους συμφοιτητές και καλούς μου φίλους Γιάννη, Αντώνη και Θάνο, τους οποίους ευχαριστώ θερμά, για όλη τη συμπαράσταση και τις στιγμές που ζήσαμε κατά τη διάρκεια αυτού του ταξιδιού.

Μαντζούτας Ανδρέας, Ιούλιος 2024

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Τα τελευταία χρόνια, η τεχνολογία blockchain έχει αναδειχθεί ως μια πρωτοποριακή καινοτομία, η οποία μεταμορφώνει ένα ευρύ φάσμα βιομηχανιών προσφέροντας ένα αποκεντρωμένο, απαραβίαστο σύστημα καταλόγου. Αρχικά παρουσιάστηκε από τον Satoshi Nakamoto το 2008, μέσω της δημοσίευσης "Bitcoin: A Peer-to-Peer Electronic Cash System"[19]. Η τεχνολογία blockchain σχεδιάστηκε παράλληλα με το Bitcoin ως ένα αποκεντρωμένο σύστημα για τη διενέργεια συναλλαγών απευθείας μεταξύ των μερών, χωρίς την ανάγκη διαμεσολαβητών. Η εξέλιξη αυτή όχι μόνο παρουσίασε μια νέα μορφή ψηφιακού νομίσματος, αλλά και πρωτοστάτησε σε μια νέα μέθοδο διαχείρισης δεδομένων που υπόσχεται αυξημένη διαφάνεια, ασφάλεια και αποτελεσματικότητα σε λειτουργίες που κυμαίνονται από τη χρηματοδότηση και τη διαχείριση της εφοδιαστικής αλυσίδας έως την υγειονομική περίθαλψη.

Η εφαρμογή της τεχνολογίας Blockchain επεκτείνεται σημαντικά στον ακαδημαϊκό τομέα, αντιμετωπίζοντας επίμονες προκλήσεις όπως η διαχείριση αρχείων και η έκδοση πιστοποιητικών και πτυχίων. Οι παραδοσιακές διαδικασίες στα εκπαιδευτικά ιδρύματα είναι συχνά επιβαρυμένες με μεγάλο διοικητικό φόρτο, επιρρεπείς σε ανθρώπινα λάθη και πλαστογραφίες. Τα εγγενή χαρακτηριστικά της τεχνολογίας blockchain -όπως η αμετάβλητη και δημόσια επαληθευσιμότητα- προσφέρουν μια ισχυρή λύση σε αυτά τα ζητήματα. Με την ενσωμάτωση των ακαδημαϊκών πιστοποιητικών στο blockchain, τα ιδρύματα μπορούν να εκσυγχρονίσουν δραματικά τις διοικητικές διαδικασίες, να μειώσουν τα σφάλματα και να ενισχύσουν την ασφάλεια των αρχείων, καθιστώντας έτσι την επαλήθευση των ακαδημαϊκών επιτευγμάτων απλή και αξιόπιστη για τους εργοδότες, άλλα ιδρύματα και οργανισμούς επαλήθευσης.

Η παρούσα διατριβή επιχειρεί την εφαρμογή της τεχνολογίας blockchain στον ακαδημαϊκό χώρο, μέσω του δικτύου Cardano, για να φέρει επανάσταση στα συστήματα πιστοποίησης ταυτότητας και έκδοσης πιστοποιητικών των φοιτητών. Προτείνουμε ένα σύστημα βασισμένο σε Blockchain που αξιοποιεί τις δυνατότητες του δικτύου Cardano για την έκδοση επαληθεύσιμων και αμετάβλητων ακαδημαϊκών πιστοποιητικών, ενισχύοντας έτσι την ακεραιότητα και την αποτελεσματικότητα των διαδικασιών πιστοποίησης. Η μελέτη εμβαθύνει στους τεχνικούς μηχανισμούς, τα οφέλη και τις προκλήσεις της ανάπτυξης τέτοιων λύσεων, με στόχο να εκσυγχρονίσει σημαντικά τις εκπαιδευτικές διοικητικές διαδικασίες.

1.1 Περίγραμμα της εργασίας

Στο κεφάλαιο 2 της παρούσας εργασίας, διερευνούμε το θεωρητικό υπόβαθρο της τεχνολογίας blockchain, αναλύοντας λεπτομερώς τα βασικά συστατικά της, καθώς και την έννοια των αποκεντρωμένων αναγνωριστικών. Τέλος, παρέχουμε μια εστιασμένη εξέταση του δικτύου Cardano, το οποίο στηρίζει την εφαρμογή μας.

Το κεφάλαιο 3 αναλύσει την αρχιτεκτονική του συστήματος της προτεινόμενης εφαρμογής μας, εξηγώντας τα συστατικά της και τις σχετικές περιπτώσεις χρήσης.

Στη συνέχεια, το Κεφάλαιο 4 εμβαθύνει στη λεπτομερή υλοποίηση του συστήματος, καλύπτοντας τα πάντα, από τα έξυπνα συμβόλαια που χρησιμοποιήθηκαν έως την ενσωμάτωση των front-end και back-end συστημάτων.

Τέλος, το Κεφάλαιο 5 αποτυπώνει τις προκλήσεις που αντιμετωπίστηκαν κατά την ανάπτυξη και περιγράφει πιθανές μελλοντικές βελτιώσεις.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

Αυτό το κεφάλαιο διερευνά τα θεωρητικά θεμέλια της τεχνολογίας blockchain, των αποκεντρωμένων ταυτοτήτων (DID) και του Cardano, της πλατφόρμας blockchain που χρησιμοποιήθηκε για τη μελέτη μας. Συζητά τις αρχές, τους μηχανισμούς και τα θεωρητικά θεμέλια αυτών των τεχνολογιών, δίνοντας έμφαση στον ρόλο τους στην παροχή ασφαλών και επαληθεύσιμων ψηφιακών ταυτοτήτων. Η θεωρητική θεμελίωση θα βοηθήσει τους αναγνώστες να κατανοήσουν τις δυνατότητες ενός συστήματος που βασίζεται στην τεχνολογία blockchain για τη διαχείριση ακαδημαϊκών αρχείων και θα παράσχει γνώσεις για την αποτελεσματική εφαρμογή ενός τέτοιου συστήματος.

2.1 Τεχνολογία Blockchain

2.1.1 Κύρια Στοιχεία

2.1.1.1 Συναλλαγή

Μια συναλλαγή blockchain είναι ένα ψηφιακό αρχείο μεταφοράς περιουσιακών στοιχείων ή δεδομένων, το οποίο υπογράφεται ψηφιακά για αυθεντικότητα και ακεραιότητα και στη συνέχεια επικυρώνεται από εξορύκτες ή επικυρωτές δικτύου για να συμπεριληφθεί σε ένα μπλοκ.

2.1.1.2 Μπλοκ

Ένα μπλοκ σε μια αλυσίδα blockchain είναι μια δομή δεδομένων που περιέχει επικυρωμένες συναλλαγές και μια επικεφαλίδα μπλοκ με σχετικά μεταδεδομένα [28]. Κάθε μπλοκ συνδέεται κρυπτογραφικά με τον προκάτοχό του, δημιουργώντας μια αμετάβλητη αλυσίδα. Αυτή η σύνδεση διασφαλίζει την ακεραιότητα και τη συνέχεια του ιστορικού των συναλλαγών, καθιστώντας κάθε μπλοκ κρίσιμο μέρος της δομής και της ασφάλειας της αλυσίδας μπλοκ.

2.1.1.3 Κόμβος

Ένας κόμβος είναι ένας υπολογιστής συνδεδεμένος στο δίκτυο blockchain, υπεύθυνος για την επαλήθευση των συναλλαγών και των μπλοκ, ενώ οι πλήρεις κόμβοι επιβάλλουν τους κανόνες του δικτύου και διατηρούν ένα πλήρες αντίγραφο του blockchain ledger.

2.1.1.4 Εξορύκτης

Ένας εξορύκτης είναι ένας κόμβος που εκτελεί υπολογιστική εργασία για την προσθήκη νέων μπλοκ στην αλυσίδα blockchain, την επικύρωση συναλλαγών και την επίλυση κρυπτογραφικών γρίφων στην εξόρυξη. Αυτό διασφαλίζει το δίκτυο, τους αποφέρει αμοιβές συναλλαγών και τους ανταμείβει με νέα νομίσματα.

2.1.1.5 Blockchain

Το Blockchain είναι ένα αποκεντρωμένο δίκτυο ομότιμων κόμβων που χρησιμοποιεί μια κατανεμημένη δομή δεδομένων, η οποία επιτρέπει μόνο προσθήκες, για τη διευκόλυνση των συναλλαγών και των αλληλεπιδράσεων χωρίς κεντρική αρχή. Αποτελείται από υπογεγραμμένες συναλλαγές μεταξύ ομοτίμων, οι οποίες ομαδοποιούνται σε μπλοκ και επαληθεύονται από τους κόμβους για να διασφαλιστεί η ακεραιότητα της δομής και να αποτραπούν δόλιες δραστηριότητες. Η συναίνεση είναι η θεμελιώδης αρχή που επιτρέπει σε όλους τους συμμετέχοντες να συμφωνούν στο αρχείο συναλλαγών, επιτρέποντας την ασφαλή, διαφανή και απαραβίαστη τήρηση αρχείων. Αυτή η αρχιτεκτονική υποστηρίζει ένα ευρύ φάσμα εφαρμογών πέραν των απλών συναλλαγών[8].

2.1.2 Τύποι Δικτύων Blockchain

Η τεχνολογία blockchain έχει εξελιχθεί σε τρεις κύριες κατηγορίες: δημόσια, ιδιωτικά και υβριδικά. Τα δημόσια Blockchain είναι ανοικτά και χωρίς άδεια, επιτρέποντας σε οποιονδήποτε να συμμετέχει στην εξόρυξη και την επαλήθευση των συναλλαγών. Τα ιδιωτικά περιορίζουν την πρόσβαση σε μια επιλεγμένη ομάδα, ενισχύοντας την ιδιωτικότητα και την αποτελεσματικότητα. Τέλος, τα υβριδικά blockchain συνδυάζουν αυτά τα μοντέλα, με μια κοινοπραξία να επιλέγει κόμβους για τη διακυβέρνηση. Αυτή η ταξινόμηση αναδεικνύει την ευελιξία της τεχνολογίας blockchain, από την προώθηση της διαφάνειας σε δημόσια δίκτυα έως τη διασφάλιση ευαίσθητων δεδομένων σε ιδιωτικά συστήματα[8].

2.1.2.1 Δημόσιο

Ένα δημόσιο blockchain είναι ένα αποκεντρωμένο σύστημα όπου οι συναλλαγές μπορούν να αποστέλλονται και να επαληθεύονται παγκοσμίως, επιτρέποντας σε οποιονδήποτε να συμμετέχει στη διαδικασία συναίνεσης. Εξασφαλίζεται από την κρυπτοοικονομία, έναν συνδυασμό οικονομικών κινήτρων και κρυπτογραφικών μηχανισμών επαλήθευσης, όπως η απόδειξη εργασίας ή η απόδειξη συμμετοχής. Ο βαθμός επιρροής στη διαδικασία συναίνεσης είναι ανάλογος με την ποσότητα των οικονομικών πόρων που μπορεί να διαθέσει ένα εμπλεκόμενο μέρος. Αυτό καθιστά τα δημόσια blockchain μια πλήρως αποκεντρωμένη εναλλακτική λύση στην κεντρική εμπιστοσύνη[5].

2.1.2.2 Υβριδικό

Ένα υβριδικό blockchain είναι ένας τύπος αλυσίδας μπλοκ όπου η διαδικασία συναίνεσης ελέγχεται από ένα προεπιλεγμένο σύνολο κόμβων, όπως χρηματοπιστωτικά ιδρύματα. Κάθε μέρος λειτουργεί έναν κόμβο και πρέπει να υπογράφουν κάθε μπλοκ για να είναι έγκυρο. Το δικαίωμα ανάγνωσης του blockchain μπορεί να είναι δημόσιο ή να περιορίζεται στους συμμετέχοντες[5].

2.1.2.3 Ιδιωτικό

Ένα πλήρως ιδιωτικό blockchain είναι ένα σύστημα όπου τα δικαιώματα εγγραφής είναι συγκεντρωμένα σε έναν οργανισμό, ενώ τα δικαιώματα ανάγνωσης μπορούν να είναι δημόσια ή περιορισμένα. Αυτό είναι σύνηθες σε εσωτερικές εφαρμογές όπως η διαχείριση βάσεων δεδομένων και ο έλεγχος, όπου η δυνατότητα δημόσιας ανάγνωσης μπορεί να μην είναι απαραίτητη σε ορισμένες περιπτώσεις[5].

2.1.3 Μοντέλα Blockchain

Η τεχνολογία blockchain έχει δύο μοντέλα για τη διαχείριση των συναλλαγών και της κατάστασης: το μοντέλο Λογαριασμού (Account Model) και το μοντέλο Unspent Transaction Output (UTXO). Το Ethereum χρησιμοποιεί το Μοντέλο Λογαριασμού, το οποίο ενημερώνει το υπόλοιπο κάθε λογαριασμού, ενώ το Bitcoin χρησιμοποιεί το Μοντέλο UTXO, το οποίο παρακολουθεί την ιδιοκτησία μέσω των μη δαπανημένων ποσών από προηγούμενες συναλλαγές.

2.1.3.1 Μοντέλο Λογαριασμού

Το Μοντέλο Λογαριασμού είναι μια μέθοδος διαχείρισης ψηφιακών περιουσιακών στοιχείων, που μοιάζει με τους παραδοσιακούς τραπεζικούς λογαριασμούς. Περιλαμβάνει μεταβάσεις καταστάσεων, οι οποίες περιλαμβάνουν άμεσες μεταφορές αξίας και πληροφοριών μεταξύ λογαριασμών. Αυτές οι μεταβάσεις κατηγοριοποιούνται σε δύο τύπους: αυτές που ελέγχονται από τους χρήστες μέσω ιδιωτικών κλειδιών και αυτές που διέπονται από τον κώδικα έξυπνων συμβολαίων. Το σύστημα ενημερώνει την παγκόσμια κατάσταση του blockchain, η οποία χρησιμεύει ως λογιστικό βιβλίο για τα υπόλοιπα, την αποθήκευση και τον εκτελέσιμο κώδικα κάθε λογαριασμού[4].

Το μοντέλο λογαριασμού χρησιμοποιεί συναλλαγές για την προσαρμογή των υπολοίπων των λογαριασμών, επιτρέποντας την εύκολη παρακολούθηση της ιδιοκτησίας των περιουσιακών στοιχείων και της μετακίνησής τους σε διάφορα δίκτυα. Η παγκόσμια καταγραφή της κατάστασης των υπολοίπων και των χαρακτηριστικών του λογαριασμού το καθιστά κατάλληλο για διάφορες εφαρμογές, από απλές μεταφορές έως σύνθετες αλληλεπιδράσεις έξυπνων συμβολαίων[17].

2.1.3.2 Μοντέλο Unspent Transaction Outputs

Το μοντέλο Unspent Transaction Outputs (UTXO) είναι ένα σύστημα που παρακολουθεί τις μη αναλωθείσες εξόδους από παλιότερες συναλλαγές και όχι τα υπόλοιπα λογαριασμών. Κάθε UTXO αντιπροσωπεύει ένα κομμάτι ψηφιακού νομίσματος που δεν έχει δαπανηθεί και μπορεί να χρησιμοποιηθεί ως είσοδος για νέες συναλλαγές[12]. Όταν ένας χρήστης ξεκινά μια συναλλαγή, το δίκτυο επαληθεύει και καταναλώνει αυτά τα UTXO για να δημιουργήσει νέες εξόδους, εξασφαλίζοντας την ανιχνευσιμότητα κάθε ψηφιακού νομίσματος.

Η αρχιτεκτονική του μοντέλου UTXO προάγει την ανωνυμία και την ιδιωτικότητα, καθώς αναλύει το ιστορικό συναλλαγών σε μεμονωμένες εξόδους, αντί να τις συγκεντρώνει σε λογαριασμούς χρηστών. Αυτή η δομή επιτρέπει επίσης την παράλληλη επεξεργασία, βελτιώνοντας δυνητικά την επεκτασιμότητα και την αποδοτικότητα του δικτύου, καθώς κάθε UTXO είναι ανεξάρτητο.

2.1.3.3 Μοντέλο Αντικειμένου

Το Sui Blockchain χρησιμοποιεί το Object Model , μια υβριδική προσέγγιση που συνδυάζει στοιχεία τόσο από το μοντέλο UTXO όσο και από το μοντέλο Λογαριασμού. Αυτό το μοντέλο δομεί τις συναλλαγές και τις καταστάσεις, αξιοποιώντας τα πλεονεκτήματα της παράλληλης επεξεργασίας των UTXO, διατηρώντας παράλληλα τη συνοχή της κατάστασης και την ευκολία προγραμματισμού. Το Sui Object Model αντιμετωπίζει τα ψηφιακά περιουσιακά στοιχεία ως μοναδικά αντικείμενα, επιτρέποντας την άμεση αλληλεπίδραση και τον χειρισμό. Αυτή η μέθοδος βελτιώνει τις συναλλαγές και τις εκτελέσεις έξυπνων συμβολαίων, προωθώντας μια πιο αποτελεσματική και κλιμακώσιμη υποδομή[18].

2.1.4 Αλγόριθμοι Συναίνεσης

Η συναίνεση είναι ένας κρίσιμος μηχανισμός σε κατανεμημένα συστήματα όπως τα blockchain, όπου οι κόμβοι συμφωνούν σε μια ενιαία κατάσταση της βάσης δεδομένων ή του λογιστικού βιβλίου, εξασφαλίζοντας μια συνεπή άποψη των πληροφοριών. Αυτός ο μηχανισμός αποτρέπει τις αποκλίσεις και διατηρεί την ακεραιότητα και την αξιοπιστία του δικτύου. Χωρίς συναίνεση, το δίκτυο θα μπορούσε να είναι ευάλωτο σε τρωτά σημεία ασφαλείας, όπως η διπλή δαπάνη και οι διακλαδώσεις, οδηγώντας σε έλλειψη εμπιστοσύνης και πιθανή αποτυχία του συστήματος[20].

Η Βυζαντινή Συμφωνία (Byzantine Agreement - BA) αντιμετωπίζει την πρόκληση της συναίνεσης σε κατανεμημένα δίκτυα με ελαττωματικούς ή κακόβουλους κόμβους, δίνοντας έμφαση στην ανάγκη για αξιόπιστες λύσεις που μπορούν να λειτουργούν αποτελεσματικά σε ανώνυμες και σύγχρονες ρυθμίσεις. Η BA περιγράφει την πολυπλοκότητα της επίτευξης συμφωνίας σε κατανεμημένα δίκτυα όπου οι συμμετέχοντες μπορεί να μην εμπιστεύονται ο ένας τον άλλον ή να ενεργούν κακόβουλα[15]. Οι τεχνολογίες blockchain χρησιμοποιούν μηχανισμούς συναίνεσης όπως η απόδειξη εργασίας (Proof of Work - PoW) και η απόδειξη συμμετοχής (Proof of Stake - PoS), οι οποίοι εξασφαλίζουν τη συναίνεση μέσω υπολογιστικά εντατικών εργασιών και ανταμείβουν τους επικυρωτές με βάση τη συμμετοχή τους στο δίκτυο. Αυτές οι μέθοδοι ενισχύουν την ασφάλεια του δικτύου έναντι επιθέσεων και χειραγώγησης[8][10][20].

2.1.4.1 Proof of Work

Το Proof of Work (PoW), ένα σύστημα που εισήχθη από το Bitcoin και αργότερα υιοθετήθηκε από το Ethereum, περιλαμβάνει εξορύκτες που λύνουν κρυπτογραφικούς γρίφους για να επικυρώνουν συναλλαγές και να δημιουργούν νέα μπλοκ. Η πολυπλοκότητα ρυθμίζεται δυναμικά για να διατηρείται η συνέπεια στους χρόνους παραγωγής μπλοκ. Ο πρώτος εξορύκτης που λύνει τον γρίφο κερδίζει κρυπτονόμισμα, προωθώντας τη συμμετοχή και την ασφάλεια στο δίκτυο, αποτρέποντας τη διπλή δαπάνη και εξασφαλίζοντας αμετάβλητες συναλλαγές[20][22].

Η ανθεκτικότητα του PoW οφείλεται στην υπολογιστική προσπάθεια που απαιτείται για την επίλυση των γρίφων, καθιστώντας τεχνικά δύσκολη και οικονομικά ανέφικτη την αλλοίωση του blockchain. Αυτό οφείλεται στο γεγονός ότι η αλλοίωση της αλυσίδας θα απαιτούσε από έναν κακόβουλο παράγοντα να ξεπεράσει το τίμιο δίκτυο λύνοντας σταθερά πρώτος τους γρίφους. Απαιτείται έλεγχος πάνω από το 50% της υπολογιστικής ισχύος του δικτύου, γεγονός που απαιτεί σημαντικούς οικονομικούς και ενεργειακούς πόρους. Η συναίνεση μεταξύ των συμμετεχόντων στο δίκτυο εξασφαλίζει μια ενιαία, αξιόπιστη έκδοση του ledger, διατηρώντας την ακεραιότητα και την αξιοπιστία του blockchain[26].

2.1.4.2 Proof of Stake

Το Proof of Stake (PoS) είναι ένας μηχανισμός συναίνεσης που διασφαλίζει τα δίκτυα blockchain απαιτώντας από τους επικυρωτές να ποντάρουν ένα συγκεκριμένο ποσό κρυπτονομίσματος ως εγγύηση. Αυτό τους επιτρέπει να συμμετέχουν στις διαδικασίες επικύρωσης και δημιουργίας μπλοκ, με την πιθανότητα επικύρωσης να είναι ανάλογη με το ποσό του νομίσματος που έχει στοιχηματιστεί. Οι επικυρωτές επιλέγονται τυχαία και η εγκυρότητα του μπλοκ διασφαλίζεται από άλλους επικυρωτές που πιστοποιούν την ακρίβειά του. Η διαδικασία αυτή ελαχιστοποιεί την κατανάλωση ενέργειας και εισάγει ένα κρυπτο-οικονομικό επίπεδο ασφάλειας, καθώς οι επικυρωτές έχουν οικονομικό κίνητρο να ενεργούν προς το συμφέρον του δικτύου[20][21].

Το PoS αντιμετωπίζει το πρόβλημα της διπλής δαπάνης διασφαλίζοντας ότι οι επικυρωτές έχουν ένα μερίδιο στον κίνδυνο, λειτουργώντας αποτρεπτικά έναντι της ανέντιμης συμπεριφοράς. Η αποκεντρωμένη φύση του PoS καθιστά τις επιθέσεις απαγορευτικά δαπανηρές, καθώς ένας επιτιθέμενος θα πρέπει να κατέχει ένα σημαντικό μέρος του νομίσματος που έχει στοιχηματίσει. Σε αντίθεση με το PoW, το PoS βασίζεται στην εσωτερική οικονομική αξία εντός του blockchain, ευθυγραμμίζοντας τα συμφέροντα των επικυρωτών με την ασφάλεια και την ακεραιότητα του δικτύου. Αυτή η μετατόπιση από την εξάντληση πόρων σε ένα μοντέλο με βάση το διακύβευμα προσφέρει ένα πιο ενεργειακά αποδοτικό πλαίσιο για την επίτευξη συναίνεσης στις τεχνολογίες κατανεμημένων καταλόγων[3].

2.1.4.3 Delegated Proof of Stake

Το Delegated Proof of Stake (DPoS) είναι μια εξέλιξη του μηχανισμού συναίνεσης Proof of Stake. Περιλαμβάνει τους συμμετέχοντες στο δίκτυο που ψηφίζουν για να εκλέξουν μια μικρή ομάδα αντιπροσώπων ή παραγωγών

μπλοκ, οι οποίοι είναι υπεύθυνοι για την επικύρωση των συναλλαγών και τη δημιουργία νέων μπλοκ. Αυτή η διαδικασία ψηφοφορίας εκδημοκρατίζει την εξουσία και επιτρέπει στους κατόχους νομισμάτων να έχουν λόγο στη συντήρηση του δικτύου. Οι αντιπρόσωποι έχουν κίνητρο να ενεργούν προς το συμφέρον του δικτύου μέσω ανταμοιβών και μπορούν να καταψηφιστούν εάν αποτύχουν ή ενεργήσουν κακόβουλα. Το DPoS μπορεί να μειώσει σημαντικά το κόστος λειτουργίας και συντήρησης του δικτύου, διατηρώντας παράλληλα την αποκέντρωση[14].

2.1.4.4 Proof of History

Η απόδειξη ιστορικού (Proof of History - PoH) είναι ένας μηχανισμός συναίνεσης blockchain που χρησιμοποιεί το χρόνο για να επαληθεύσει την ακολουθία και το χρονοδιάγραμμα των συναλλαγών. Αποδίδει μια μοναδική χρονοσφραγίδα σε κάθε συναλλαγή, επιτρέποντας την ανεξάρτητη επαλήθευση χωρίς συνεχή επικοινωνία μεταξύ των κόμβων. Αυτή η προσέγγιση συμβάλλει στην επίλυση του προβλήματος της διπλής δαπάνης, αποτρέποντας το ίδιο ψηφιακό περιουσιακό στοιχείο να δαπανηθεί περισσότερες από μία φορές[30].

2.1.5 Έξυπνα Συμβόλαια

Τα έξυπνα συμβόλαια, τα οποία εισήγαγε ο Nick Szabo το 1994[24], αυτοματοποιούν τις συμβατικές συμφωνίες μετατρέποντας τις παραδοσιακές ρήτρες σε προγραμματιζόμενο κώδικα ενσωματωμένο σε ψηφιακές ή φυσικές ιδιότητες. Στόχος τους είναι να επιβάλλουν αυτόνομα τους όρους των συμβάσεων. Στο πλαίσιο του οικοσυστήματος blockchain, τα έξυπνα συμβόλαια λειτουργούν ως σενάρια ή εφαρμογές, που εκτελούνται ανεξάρτητα σε κάθε κόμβο του δικτύου, όταν ενεργοποιούνται από συναλλαγές[10]. Αυτή η αυτόνομη εκτέλεση διασφαλίζει τη διαφανή και αποτελεσματική τήρηση των συμφωνιών χωρίς την ανάγκη κεντρικής αρχής ή μεσαζόντων.

Ένα έξυπνο συμβόλαιο σε ένα blockchain αποκτά μια μοναδική διεύθυνση για την ενεργοποίηση των συναλλαγών, επιτρέποντάς του να διαχειρίζεται αλληλεπιδράσεις με βάση τα δεδομένα μεταξύ των μερών. Μπορεί να διευκολύνει τις συναλλαγές με αυτόνομη επαλήθευση και εκτέλεση συναλλαγών βάσει προκαθορισμένων συνθηκών, όπως η ανταλλαγή ψηφιακών περιουσιακών στοιχείων υπό συγκεκριμένους όρους[10]. Αυτή η διαδικασία καταδεικνύει την ικανότητα του έξυπνου συμβολαίου να επιβάλλει συμφωνίες και να διαχειρίζεται με ασφάλεια περιουσιακά στοιχεία και καταστάσεις εντός του οικοσυστήματος blockchain.

2.1.5.1 Αποκεντρωμένες Εφαρμογές

Μια αποκεντρωμένη εφαρμογή (DApp) είναι λογισμικό που αντί να εκτελείται σε έναν κεντρικό υπολογιστή, τρέχει πάνω σε ένα κατανεμημένο σύστημα, συνήθως ένα δίκτυο blockchain. Χρησιμοποιώντας έξυπνα συμβόλαια, τα DApps μπορούν να λειτουργούν αυτόνομα, εξαλείφοντας την ανάγκη για κεντρική αρχή. Οι αποκεντρωμένες εφαρμογές επιτρέπουν άμεσες, χωρίς εμπιστοσύνη αλληλεπιδράσεις μεταξύ των χρηστών, υποστηρίζοντας ποικίλες λειτουργίες, από χρηματοοικονομικές υπηρεσίες έως τυχερά παιχνίδια, ενώ παράλληλα διασφαλίζουν την ακεραιότητα των δεδομένων και την αξιοπιστία του συστήματος.

2.2 Αποκεντρωμένα Συστήματα Ταυτοποίησης

Η ψηφιακή εποχή έχει οδηγήσει σε αύξηση της αποθήκευσης και της επεξεργασίας προσωπικών δεδομένων, αναδεικνύοντας τις αδυναμίες των παραδοσιακών συστημάτων διαχείρισης ταυτότητας (IDMS). Τα συγκεντρωτικά συστήματα, όπου μια και μόνη οντότητα ελέγχει τεράστιες ποσότητες δεδομένων, εγείρουν ανησυχίες για την προστασία της ιδιωτικής ζωής. Οι παραβιάσεις δεδομένων έχουν αναδείξει τα εγγενή τρωτά σημεία των συγκεντρωτικών IDMS, όπου οι χρήστες συχνά δεν έχουν τον έλεγχο των δεδομένων τους και η κοινή χρήση από τρίτους γίνεται χωρίς ρητή συγκατάθεση[1].

Οι αδυναμίες των σημερινών IDMS έχουν προκαλέσει το ενδιαφέρον για αποκεντρωμένες λύσεις ταυτότητας, όπως η διαχείριση ταυτότητας με βάση τα Blockchains. Οι αποκεντρωμένες ταυτότητες (DID) προσφέρουν μια προσέγγιση με επίκεντρο τον χρήστη, επιτρέποντας στα άτομα να ελέγχουν τα δεδομένα τους χωρίς κεντρικές αρχές ή μεσάζοντες[1][2][13]. Αυτή η στροφή προς την αυτοκυριαρχούμενη ταυτότητα (SSI) χρησιμοποιεί την αποκεντρωμένη αρχιτεκτονική του διαδικτύου για να αποκαταστήσει τον έλεγχο, την ασφάλεια και την ιδιωτικότητα των ατόμων. Τα DID προσφέρουν ενισχυμένη ιδιωτικότητα, αυξημένη ασφάλεια και μεγαλύτερο έλεγχο των προσωπικών δεδομένων, συμβαδίζοντας με την αυξανόμενη ζήτηση για ασφαλείς λύσεις ψηφιακής ταυτότητας.

2.2.1 Τεχνολογικό Υπόβαθρο

2.2.1.1 Αποκεντρωμένα Αναγνωριστικά

Τα αποκεντρωμένα αναγνωριστικά (DID) είναι μοναδικά και μόνιμα αναγνωριστικά που χρησιμοποιούνται σε αποκεντρωμένα συστήματα για επαληθευμένες ψηφιακές ταυτότητες[29]. Επιτρέπουν την αυτοκυριαρχική ταυτότητα και μπορούν να χρησιμοποιηθούν χωρίς κεντρικό μητρώο, πάροχο ταυτότητας ή αρχή έκδοσης πιστοποιητικών. Τα DIDs συνδέουν ένα υποκείμενο DID με ένα έγγραφο DID, επιτρέποντας σε οντότητες όπως άτομα, οργανισμούς, συσκευές και εφαρμογές να πιστοποιούν ψηφιακά την ταυτότητά τους χωρίς κεντρική εποπτεία. Αυτή η καινοτομία ενδυναμώνει τους χρήστες, επιτρέποντάς τους να έχουν τον έλεγχο της ταυτότητάς τους, προωθώντας την ψηφιακή αυτονομία μακριά από ένα παράδειγμα κεντρικού ελέγχου[2].

2.2.1.2 Επαληθεύσιμα Πιστοποιητικά

Τα επαληθεύσιμα πιστοποιητικά (VC) είναι ψηφιακές εκδόσεις φυσικών εγγράφων ταυτοποίησης, όπως τα διαβατήρια ή οι άδειες οδήγησης. Χρησιμοποιούν κρυπτογραφική ασφάλεια για την αποτροπή μη εξουσιοδοτημένης πρόσβασης και τη διατήρηση της ακεραιότητας των κοινών πιστοποιητικών. Αυτή η ασφαλής παρουσίαση ελαχιστοποιεί την έκθεση προσωπικών πληροφοριών, επιτρέποντας μεγαλύτερο έλεγχο στη διάδοση των δεδομένων[11][13].

2.2.1.3 Μοντέλο Αυτοκυριαρχούμενης Ταυτότητας

Το μοντέλο της αυτοκυριαρχούμενης ταυτότητας (SSI), το οποίο δίνει τη δυνατότητα στα άτομα να ελέγχουν πλήρως τα δεδομένα της ταυτότητάς τους, αποτελεί βασικό θεμέλιο για τις εξελίξεις στη διαχείριση ψηφιακών ταυτοτήτων, ανοίγοντας το δρόμο για ένα μέλλον όπου οι ψηφιακές ταυτότητες θα είναι τόσο φορητές όσο και επικεντρωμένες στην προστασία της ιδιωτικής ζωής[2][11].

2.2.1.4 Σκοπός του Αποκεντρωμένου Συστήματος Ταυτοποίησης

Το αποκεντρωμένο σύστημα ταυτοποίησης αποσκοπεί στην αντιμετώπιση των ζητημάτων προστασίας της ιδιωτικής ζωής και των παραβιάσεων δεδομένων στα παραδοσιακά συστήματα ταυτοποίησης, εξαλείφοντας την εξάρτηση από κεντρικές τεχνολογίες. Αυτή η καινοτόμος προσέγγιση ενισχύει τον έλεγχο και την ιδιωτικότητα των χρηστών, ανοίγοντας το δρόμο για ένα ασφαλές ψηφιακό οικοσύστημα. Καθώς οι τεχνολογίες εξελίσσονται, θα επαναπροσδιορίσουν τις ψηφιακές αλληλεπιδράσεις, προωθώντας την εμπιστοσύνη και την ασφάλεια[2][11][13].

2.2.2 Περιπτώσεις Χρήσης

Τα αποκεντρωμένα αναγνωριστικά (DID) μεταμορφώνουν τις λύσεις σε διάφορους τομείς, ενισχύοντας την ασφάλεια των συναλλαγών, την ιδιωτικότητα και την εμπιστευτικότητα. Βελτιώνουν το ηλεκτρονικό εμπόριο, την υγειονομική περίθαλψη, την εκπαίδευση και τις χρηματοπιστωτικές υπηρεσίες. Τα DIDs υποστηρίζουν επίσης ασφαλή ψηφιακά συστήματα ψηφοφορίας, προωθώντας την ακεραιότητα και την προσβασιμότητα στην εμπλοκή των πολιτών. Συνολικά, τα DIDs είναι απαραίτητα για ένα πιο ασφαλές, αποτελεσματικό και επικεντρωμένο στον χρήστη ψηφιακό οικοσύστημα[2][11][13].

2.2.3 Σύνδεση με την τεχνολογία Blockchain

Η τεχνολογία blockchain βελτιώνει θεμελιωδώς την εφαρμογή των αποκεντρωμένων αναγνωριστικών (DID) παρέχοντας μια ασφαλή, αμετάβλητη και αποκεντρωμένη υποδομή απαραίτητη για τη διαχείριση των ψηφιακών ταυτοτήτων[13]. Η ικανότητα της τεχνολογίας να λειτουργεί χωρίς κεντρική αρχή διασφαλίζει ότι οι χρήστες μπορούν να ελέγχουν τις δικές τους ταυτότητες, συμβαδίζοντας με την αυτοκυριαρχούμενη φύση των DIDs. Με την καταγραφή των DIDs σε ένα blockchain, κάθε ταυτότητα καταγράφεται μόνιμα με τρόπο που δεν επιδέχεται αλλοίωση, εγγυώντας την ακεραιότητα και την αυθεντικότητα των ψηφιακών ταυτοτήτων.

Η τεχνολογία blockchain αποθηκεύει με ασφάλεια τα DID και διευκολύνει τη συμβολαιογραφική επικύρωση και την επαλήθευση των πιστοποιητικών μέσω έξυπνων συμβάσεων, ενισχύοντας την ασφάλεια, την ιδιωτικότητα και την αυτονομία των χρηστών. Αυτή η απρόσκοπτη ενσωμάτωση των DIDs με την τεχνολογία blockchain δημιουργεί ένα ισχυρό πλαίσιο για τη διαχείριση ψηφιακών ταυτοτήτων, ενισχύοντας τις ψηφιακές αλληλεπιδράσεις.

2.3 Cardano

Το Cardano[7] είναι μια πλατφόρμα blockchain τρίτης γενιάς. Αναπτύχθηκε από μηχανικούς και ακαδημαϊκούς για την αντιμετώπιση ζητημάτων επεκτασιμότητας, ασφάλειας και αποκέντρωσης. Χρησιμοποιεί τον αλγόριθμο ομοφωνίας Ouroboros, ένα πρωτόκολλο PoS με υψηλότερη ενεργειακή απόδοση, και εισάγει το μοντέλο extended Unspent Transaction Output (eUTXO), μια εξέλιξη των παραδοσιακών συστημάτων UTXO. Αυτό το μοντέλο ενισχύει την επεξεργασία συναλλαγών και τα έξυπνα συμβόλαια, καθιστώντας το Cardano μια ισχυρή πλατφόρμα για αποκεντρωμένες εφαρμογές[6].

2.3.1 Extended UTXO

Το μοντέλο UTXO είναι ένα μοντέλο κρυπτονομισμάτων που αναπαριστά όλες τις μη δαπανημένες εξόδους ως ξεχωριστά UTXO, καθένα από τα οποία αντιστοιχεί σε ένα συγκεκριμένο ποσό κρυπτονομισμάτων. Διασφαλίζει την ακεραιότητα των συναλλαγών αποτρέποντας τη διπλή δαπάνη και διευκολύνοντας διαφανείς διαδρομές ελέγχου μεταφοράς περιουσιακών στοιχείων. Ωστόσο, η βασική του μορφή είναι περιορισμένη στην υποστήριξη σύνθετων έξυπνων συμβολαίων και κρατικών εφαρμογών που απαιτούν διαφοροποιημένη επεξεργασία συναλλαγών.

Το μοντέλο Extended UTXO (EUTXO) βελτιώνει το παραδοσιακό πλαίσιο UTXO ενσωματώνοντας αυθαίρετα δεδομένα και εξασφαλίζοντας τη συνέχεια της σύμβασης σε όλες τις συναλλαγές. Αυτό επιτρέπει πιο εξελιγμένη λογική έξυπνων συμβολαίων και λειτουργίες με κατάσταση, επιτρέποντας την ανάπτυξη σύνθετων αποκεντρωμένων εφαρμογών, διατηρώντας παράλληλα τα πλεονεκτήματα διαφάνειας και ασφάλειας του UTXO[9].

2.3.1.1 Κύρια Στοιχεία

Το μοντέλο EUTXO αποτελείται από τρία βασικά στοιχεία: Datum, Redeemer και Πλαίσιο(Context).

Datum Το Datum είναι ένα στοιχείο δεδομένων που επισυνάπτεται στις εξόδους συναλλαγών και χρησιμεύει ως παράμετρος κατά την επικύρωση. Επιτρέπει στα συμβόλαια να διατηρούν την κατάσταση χωρίς να αλλάζουν τον επικυρωτή τους, καθιστώντας το χρήσιμο για την κατάσταση των μηχανών κατάστασης εντός των blockchains, επιτρέποντας την πολύπλοκη και διαδραστική συμπεριφορά του συμβολαίου, διατηρώντας παράλληλα την αμετάβλητη λογική του συμβολαίου[9].

Redeemer Ο Redeemer είναι ένα κρίσιμο τμήμα στις συναλλαγές UTXO, παρέχοντας μία παράμετρο από τον καταναλωτή στο σενάριο επικύρωσης. Επιτρέπει στο σενάριο να λαμβάνει αποφάσεις με βάση τις παρεχόμενες πληροφορίες, επιτρέποντας δυναμικές αλληλεπιδράσεις με έξυπνα συμβόλαια, όπου η συμπεριφορά του συμβολαίου μπορεί να αλλάξει με βάση τις εισόδους του χρήστη[9].

Πλαίσιο Το Πλαίσιο είναι ένα κρίσιμο στοιχείο για την επικύρωση της συναλλαγής, το οποίο παρέχει λεπτομερείς πληροφορίες σχετικά με τη συναλλαγή, συμπεριλαμβανομένων των εξόδων, των εισόδων και ενός δείκτη. Αυτές οι πληροφορίες επιτρέπουν στον επικυρωτή να επιβάλλει αυστηρότερες συνθήκες από το βασικό μοντέλο UTXO. Βοηθά επίσης στην επιθεώρηση των τρεχόντων εξόδων της συναλλαγής, εξασφαλίζοντας τη συνέχεια της σύμβασης. Το Πλαίσιο εξασφαλίζει ντετερμινιστική εκτέλεση , επιτρέποντας στους χρήστες να προβλέψουν το αποτέλεσμα και την κατανάλωση πόρων πριν από την υποβολή μιας συναλλαγής[9].

2.3.1.2 Κύρια Πλεονεκτήματα

Το μοντέλο Extended UTXO βελτιώνει τις εφαρμογές blockchain επιτρέποντας στις συναλλαγές να μεταφέρουν πρόσθετα δεδομένα, επιτρέποντας τη δημιουργία πιο σύνθετων έξυπνων συμβολαίων ικανών να διαχειρίζονται και να χειρίζονται πληροφορίες με δεδομένα. Επιπλέον, το μοντέλο eUTXO ενισχύει τους κανόνες επικύρωσης συναλλαγών με τη χρήση ενός "Redeemer", επιτρέποντας τη δημιουργία λογικής έξυπνων συμβολαίων πέρα από τις απλές μεταφορές περιουσιακών στοιχείων, επιτρέποντας έτσι πιο περίπλοκες συναλλαγές στο οικοσύστημα blockchain. Τέλος, διασφαλίζει τη συνέχεια των συμβάσεων, εφαρμόζοντας με συνέπεια τους κανόνες ενός συμβολαίου σε πολλαπλές συναλλαγές μέσω του Πλαισίου.

Πτυχή	UTXO	eUTXO
Διαχείριση Δεδομένων	Δεν υποστηρίζει πρόσθετα δεδομένα με συναλλαγές.	Υποστηρίζει πρόσθετα δεδομένα μέσω του Datum, επιτρέποντας έξυπνα συμβόλαια με κατάσταση.
Κανόνες Συναλλαγών	Βασικοί κανόνες επικύρωσης σχετικοί με την ιδιοκτησία.	Προηγμένη επικύρωση με τον Redeemer, επιτρέποντας σύνθετη και υπό όρους λογική συναλλαγών.

Table 2.1: Σύγκριση των μοντέλων UTXO and eUTXO

2.3.2 Έξυπνα Συμβόλαια στο δίκτυο Cardano

Η εγγενής πλατφόρμα έξυπνων συμβολαίων της Cardano, Plutus, είναι ένα ολοκληρωμένο σύστημα Turing που εξασφαλίζει ασφαλή και αποδοτικά έξυπνα συμβόλαια. Οι ρίζες του στη Haskell παρέχουν υψηλή αξιοπιστία και ορθότητα στην εκτέλεση έξυπνων συμβολαίων. Τα συμβόλαια Plutus αποτελούνται τόσο από τον on-chain κώδικα, που εκτελείται εντός του blockchain, όσο και από τον off-chain κώδικα, στη συσκευή του χρήστη[6]. Έχουν πλέον εξελιχθεί ώστε να υποστηρίζουν ένα ευρύτερο φάσμα επιλογών προγραμματισμού, συμπεριλαμβανομένου της γλώσσας Aiken[27] για κώδικα on-chain και του Lucid[23] framework για αλληλεπιδράσεις off-chain.

Τα σενάρια Plutus μπορούν να χωριστούν σε δύο κύριες κατηγορίες:

- Σενάρια επικύρωσης: Το σενάριο Validator του Plutus είναι ένα συστατικό που θέτει συνθήκες σχετικά με το πότε μπορεί να καταναλωθεί ένα UTXO, ενισχύοντας την προγραμματισιμότητα. Χρησιμοποιεί τρία ορίσματα: Datum, Redeemer και Πλαίσιο. Το Datum αντιπροσωπεύει πληροφορίες κατάστασης, το Redeemer είναι η είδοσος του καταναλωτή και το Πλαίσιο είναι πληροφορίες συναλλαγής. Αυτά τα στοιχεία επιτρέπουν στους προγραμματιστές να ορίζουν σύνθετες συνθήκες κατανάλωσης για UTXOs, επιτρέποντας έξυπνα συμβόλαια στο Cardano Blockchain.
- Minting Policies: Τα minting policies (συναλλαγές νομισματοκοπείας) είναι ένας τύπος σεναρίου Plutus που ορίζει τους κανόνες για τη δημιουργία ή την απόσυρση tokens. Απαιτούν μόνο δύο ορίσματα: τον Redeemer και το Πλαίσιο. Αυτός ο μηχανισμός επιτρέπει στους προγραμματιστές να εισάγουν καινούρια Tokens στο δίκτυο Cardano.

2.3.3 Αποκεντρωμένα Αναγνωριστικά στο δίκτυο Cardano

2.3.3.1 Atala PRISM

Το Atala PRISM είναι μια αυτοκυριαρχούμενη πλατφόρμα ταυτότητας με βάση το Cardano που παρέχει μια σειρά υπηρεσιών για την έκδοση αποκεντρωμένων αναγνωριστικών (DID) και επαληθεύσιμων πιστοποιητικών, ενισχύοντας την ανάπτυξη και την ασφάλεια του οικοσυστήματος[16].

2.3.3.2 ProofSpace

Η ProofSpace είναι μια πλατφόρμα που απλοποιεί την επαλήθευση των ψηφιακών ταυτοτήτων στο διαδίκτυο, προσφέροντας ασφαλείς λύσεις που διαχειρίζεται ο χρήστης[25]. Χρησιμοποιεί το Atala PRISM για να ενεργοποιήσει τα αποκεντρωμένα αναγνωριστικά (DID) και τα επαληθεύσιμα πιστοποιητικά εντός του Cardano Blockchain, καθιστώντας το βασικό συστατικό της παρούσας διατριβής.

Κεφάλαιο 3

Αρχιτεκτονική του Συστήματος

3.1 Επισκόπηση της Εφαρμογής

3.1.1 Σχεδιασμός

Η εφαρμογή χρησιμοποιεί μια αρχιτεκτονική στην οποία κάθε ακαδημαϊκό μάθημα διαχειρίζεται μέσω ενός συγκεκριμένου έξυπνου συμβολαίου. Οι βαθμοί σε αυτό το σύστημα είναι tokenized, πράγμα που σημαίνει ότι σε κάθε μάθημα ανατίθεται ένα token βαθμού. Το ποσό των tokens που κατέχει ένας φοιτητής υποδεικνύει τον βαθμό του για το συγκεκριμένο μάθημα. Τα προφίλ των φοιτητών είναι δομημένα ως έξυπνα συμβόλαια για να παρέχουν την ασφαλή διαχείριση αυτών των token, απαγορεύοντας έτσι τις παράνομες μεταφορές.

Τα πτυχία αποστέλλονται με τη μορφή Non-Fungible Tokens (NFTs), τα οποία περιλαμβάνουν συγκεκριμένες πληροφορίες σχετικά με τους βαθμούς ενός φοιτητή. Το σύστημα ενσωματώνει το ProofSpace για την αποκεντρωμένη επαλήθευση της ταυτότητας, ενισχύοντας την ασφάλεια και τη νομιμότητα των ακαδημαϊκών πιστοποιητικών που χορηγούνται.

3.1.2 Ενδιαφερόμενοι Χρήστες

Το σύστημα υποστηρίζει τρεις τύπους χρηστών:

- Οι φοιτητές χρησιμοποιούν τους ακαδημαϊκούς κωδικούς τους για να αποκτήσουν πρόσβαση στο σύστημα, όπου μπορούν να ελέγξουν τους βαθμούς τους, να εγγραφούν σε μαθήματα και εξάμηνα και να παραλάβουν το πτυχίο τους μετά την ολοκλήρωση των σπουδών τους.
- Οι καθηγητές πρέπει να διαθέτουν λογαριασμό Cardano συνδεδεμένο με πορτοφόλι. Αυτό τους επιτρέπει να υπογράφουν συναλλαγές προκειμένου να βαθμολογούν τους φοιτητές. Μεταφέρουν τα απαραίτητα tokens βαθμολογίας στις διευθύνσεις των φοιτητών κατά τη διάρκεια της περιόδου βαθμολόγησης.
- Η γραμματεία κόβει tokens εγγραφής, διαχειρίζεται τα δεδομένα πολιτικής και ξεκινά και τερματίζει τις περιόδους εγγραφής, καθώς και ενημερώνει τα μαθήματα που απαιτούνται για την ολοκλήρωση των σπουδών.

3.1.3 Στοιχεία του Συστήματος

1. Έξυπνα Συμβόλαια: Η πρωταρχική λειτουργία της εφαρμογής υποστηρίζεται από έξυπνα συμβόλαια, γραμμένα σε Aiken, μια γλώσσα προγραμματισμού παρόμοια με τη Rust, η οποία χρησιμοποιείται για την κατασκευή έξυπνων συμβολαίων Plutus στο Cardano. Υπάρχουν επτά βασικά έξυπνα συμβόλαια:
   - Τέσσερα συμβόλαια Minting Policy για την έκδοση token βαθμού, NFT πτυχίου, token εγγραφής και ένα token εγκυρότητας.
   - Τρία συμβόλαια επικύρωσης που διαχειρίζονται προφίλ φοιτητών, λειτουργίες μαθημάτων και δεδομένα tokens βαθμών αντίστοιχα. Αυτά τα συμβόλαια διασφαλίζουν ότι όλες οι αλληλεπιδράσεις εντός του συστήματος διέπονται από αμετάβλητους, προκαθορισμένους κανόνες, εξασφαλίζοντας τη συνεπή εφαρμογή των πρωτοκόλλων του συστήματος.
2. Frontend: Η μηχανή template Pug χρησιμοποιείται για την παροχή διεπαφής για τους φοιτητές, τους καθηγητές και τη γραμματεία. Αυτή η διαμόρφωση επιτρέπει την παρουσίαση των δεδομένων, αλλά και την επίβλεψη των διαδικασιών βαθμολόγησης, στις οποίες οι καθηγητές υποβάλλουν τους βαθμούς των φοιτητών.
3. Backend: Λειτουργεί ως ένα ενιαίο API που έχει δημιουργηθεί με τη χρήση του Express framework. Αποτελείται από διάφορα μέρη: το πρώτο μέρος χρησιμοποιεί τη βιβλιοθήκη lucid-cardano για την εκτέλεση εργασιών που αφορούν το Cardano, όπως η διαχείριση συναλλαγών, ο εντοπισμός UTXO και η επικοινωνία με το δίκτυο. Το δεύτερο μέρος είναι υπεύθυνο για την κλήση του ProofSpace για την έκδοση διαπιστευτηρίων. Επιπλέον, διαχειρίζεται μια βάση δεδομένων SQLite3 για την εκπλήρωση τυχόν πρόσθετων αναγκών σε δεδομένα και αποθηκεύει το διοικητικό κλειδί που χρησιμοποιείται για την αποθήκευση πρόσθετων δεδομένων που δεν μπορούν ή δε χρειάζεται να βρίσκονται στην αλυσίδα. Εκτός από αυτά τα συγκεκριμένα μέρη, το API περιλαμβάνει επίσης μια λειτουργικότητα γενικού σκοπού για την εκπλήρωση άλλων εργασιών και τη διευκόλυνση της επικοινωνίας μεταξύ διαφορετικών στοιχείων.
4. ProofSpace: Το ProofSpace είναι υπεύθυνο για την έκδοση επαληθεύσιμων πιστοποιητικών και αποκεντρωμένων αναγνωριστικών. Επιπλέον, είναι συμβατό με την εφαρμογή ProofSpace για κινητά, η οποία επιτρέπει στους φοιτητές να λαμβάνουν ειδοποιήσεις σε πραγματικό χρόνο σχετικά με νέους βαθμούς και άλλες σημαντικές ενημερώσεις. Αυτό το χαρακτηριστικό ενισχύει τη διαδραστική εμπειρία και διασφαλίζει ότι οι φοιτητές παραμένουν άμεσα ενημερωμένοι.

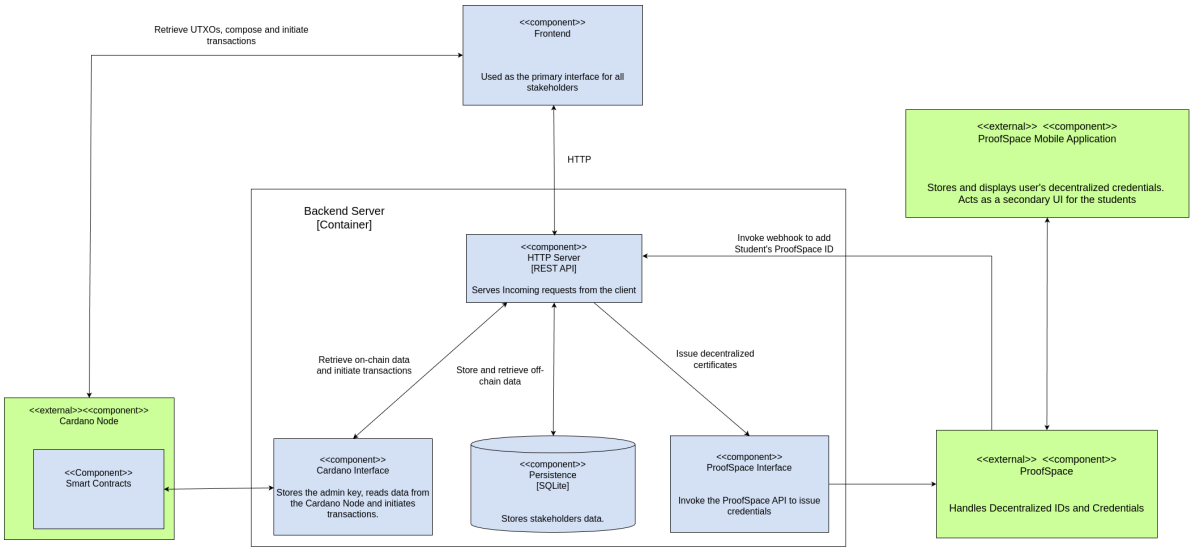


Figure 3.1.1: Διάγραμμα συνιστωσών συστήματος

3.2 Περιπτώσεις Χρήσης

3.2.1 Περίπτωση Χρήσης 1: Εγγραφή στο Πανεπιστήμιο

Η γραμματεία του πανεπιστημίου εγγράφει έναν φοιτητή, δημιουργώντας μια εγγραφή στη βάση δεδομένων με τα στοιχεία του. Το τμήμα blockchain της διαδικασίας αρχίζει όταν ο φοιτητής συνδεθεί στο σύστημα χρησιμοποιώντας τα ακαδημαϊκά του πιστοποιητικά για πρώτη φορά. Αρχικά, καλείται να σαρώσει έναν κωδικό QR στη διεπαφή, συνδέοντας την ταυτότητά του, ProofSpace ID, με το ακαδημαϊκό του προφίλ. Η σύνδεση ολοκληρώνεται όταν το ProofSpace ειδοποιεί το backend της εφαρμογής μέσω ενός webhook, αποθηκεύοντας το ID του φοιτητή για μελλοντική έκδοση πιστοποιητικών. Το backend δημιουργεί μια μοναδική διεύθυνση Cardano για τον φοιτητή, με παράμετρο το ID του και εξαργυρώνει ένα token εγγραφής στο πανεπιστήμιο με όνομα τον αριθμό μητρώου του φοιτητή και το μεταφέρει στη διεύθυνση του. Τέλος, το backend καλεί το API του ProofSpace για να εκδώσει ένα πιστοποιητικό εγγραφής στο πανεπιστήμιο.

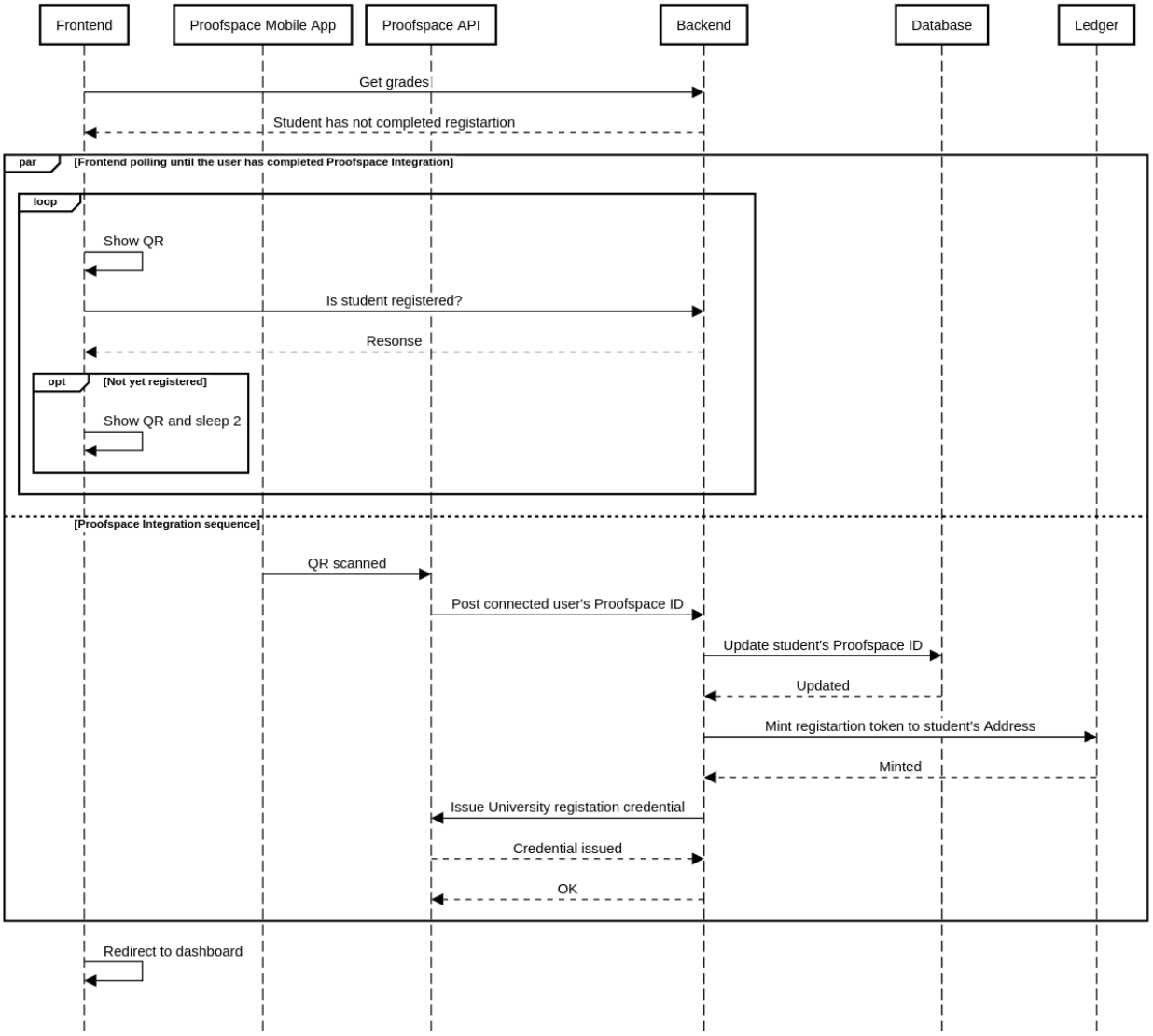


Figure 3.2.1: Διάγραμμα ακολουθίας περίπτωσης εγγραφής στο Πανεπιστήμιο

3.2.2 Περίπτωση Χρήσης 2: Προβολή βαθμών

Ο πίνακας βαθμολογίας είναι προσβάσιμος από έναν φοιτητή, όπου εκτελείται μια λειτουργία backend για την ανάκτηση των βαθμών του από το Cardano Ledger. Το σύστημα ανακτά όλα τα Unspent Transaction Outputs (UTXOs) που σχετίζονται με τη διεύθυνση του έξυπνου συμβολαίου του φοιτητή. Στη συνέχεια, το backend αλληλεπιδρά με το συμβόλαιο πολιτικής για να ανακτήσει έναν κατάλογο έγκυρων tokens βαθμού. Τέλος, με βάση αυτόν τον κατάλογο εντοπίζει ποια UTXO του φοιτητή περιέχουν έγκυρα tokens βαθμών και υπολογίζει τον βαθμό στο εκάστοτε μάθημα. Εάν υπάρχουν πολλοί βαθμοί για το ίδιο μάθημα, επιλέγεται ο υψηλότερος.

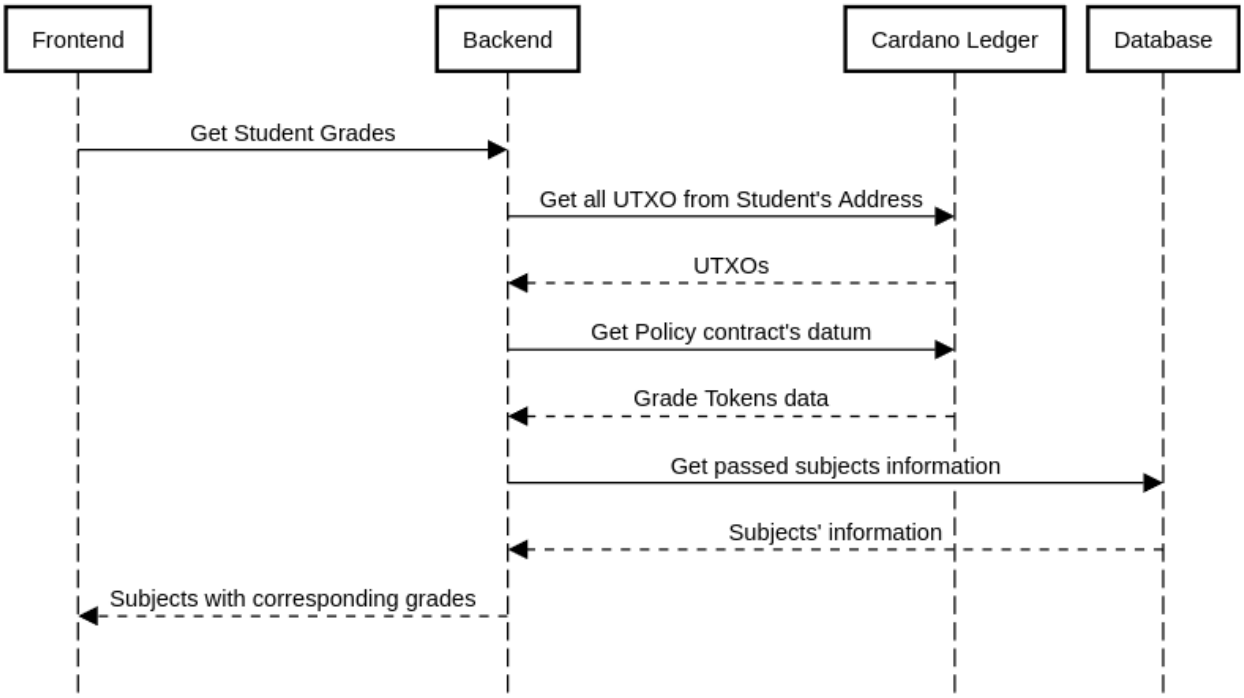


Figure 3.2.2: Διάγραμμα ακολουθίας περίπτωσης προβολής βαθμών

3.2.3 Περίπτωση Χρήσης 3: Εγγραφή σε εξάμηνο και Επιλογή Μαθημάτων

Η δήλωση μαθημάτων περιλαμβάνει δύο μέρη: την ανάκτηση των διαθέσιμων μαθημάτων και την εγγραφή. Όταν ένας φοιτητής συνδέεται, το frontend καλεί ένα endpoint για την ανάκτηση μιας λίστας με τα διαθέσιμα μαθήματα για εγγραφή. Το backend καθορίζει αρχικά τα έγκυρα θέματα με βάση τον τύπο του εξαμήνου και διασφαλίζει ότι είναι μικρότερα ή ίσα με το τρέχον εξάμηνο του φοιτητή. Τέλος, ελέγχει όλα τα UTXOs από τη διεύθυνση Cardano του φοιτητή σε σχέση με τις έγκυρες πολιτικές tokens βαθμολογίας για να εξακριβώσει τα μαθήματα που έχουν ήδη περαστεί και τα αφαιρεί από τη λίστα. Έπειτα, ο φοιτητής επιλέγει από τα διαθέσιμα μαθήματα και υποβάλλει τις επιλογές του. Καλείται το αντίστοιχο endpoint του backend για να προσθέσει τον αριθμό μητρώου του φοιτητή στον πίνακα "registrations" στο datum κάθε επιλεγμένου μαθήματος. Έπειτα, ένα token εγγραφής με τον αριθμό του τρέχοντος εξαμήνου του φοιτητή στέλνεται στη διεύθυνσή του - για την επικύρωση της εγγραφής και την αποφυγή διπλοεγγραφών - και καλείται το API του ProofSpace για την έκδοση πιστοποιητικών εγγραφής εξαμήνου και δήλωσης μαθημάτων.

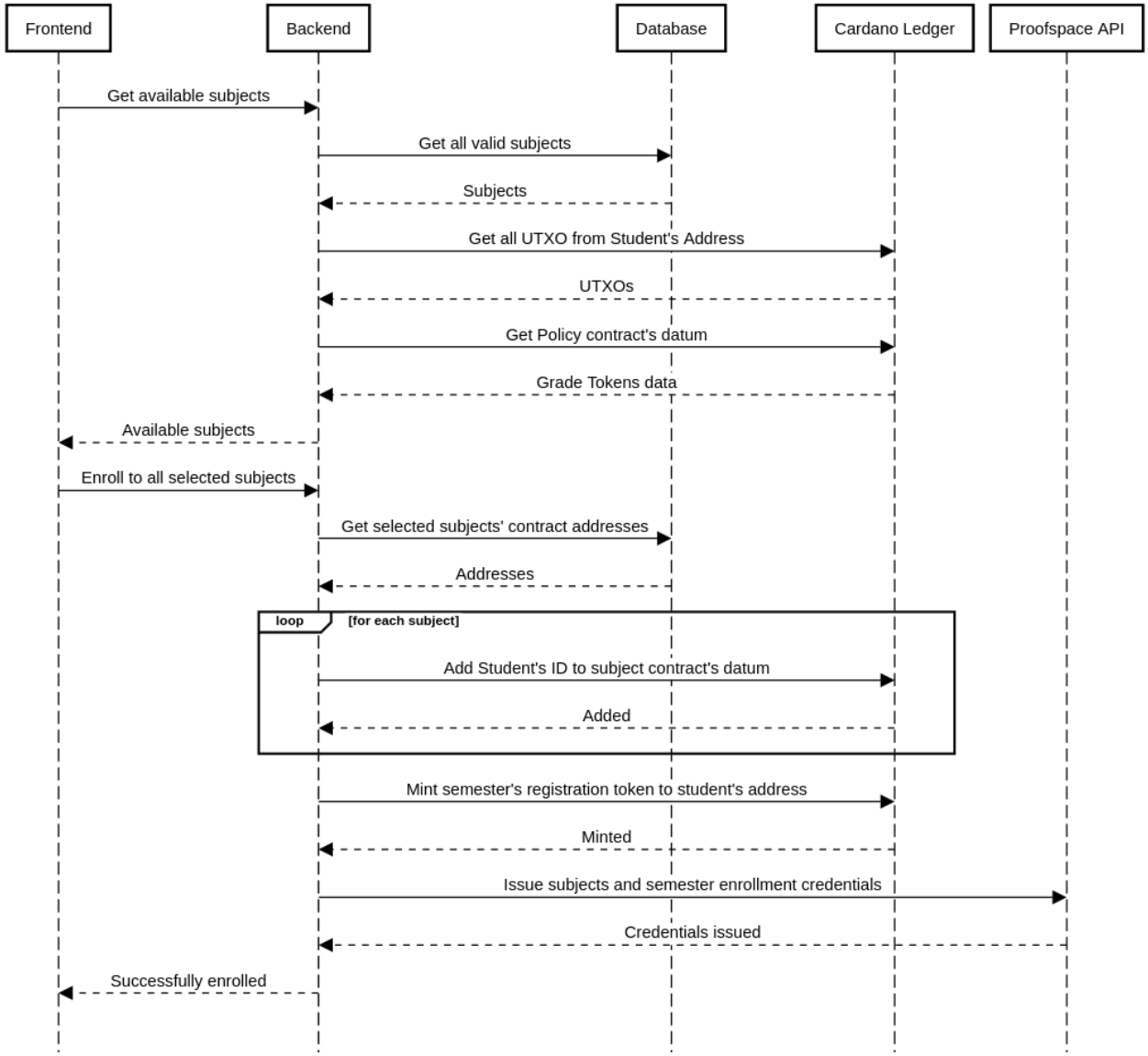


Figure 3.2.3: Διάγραμμα ακολουθίας περίπτωσης εγγραφής στο εξάμηνο

3.2.4 Περίπτωση Χρήσης 4: Απονομή Πτυχίου

Το σύστημα δρομολογεί τη διαδικασία για να παραλάβει ο φοιτητής το πτυχίο του μόλις κριθεί κατάλληλος, το οποίο επαληθεύεται τόσο εκτός όσο και εντός της αλυσίδας για να διασφαλιστεί η εγκυρότητα. Μόλις ένας φοιτητής ανακτήσει τους βαθμούς του, το σύστημα ελέγχει αυτόματα την ακαδημαϊκή του κατάσταση. Εάν έχει περάσει όλα τα απαιτούμενα μαθήματα, ένα κουμπί τον προτρέπει να διεκδικήσει το πτυχίο του. Ο φοιτητής συνδέει το πορτοφόλι του Nami και η ενεργή διεύθυνση αποστέλλεται στο backend για την κοπή του πτυχίου. Το backend υπολογίζει τον μέσο όρο του βαθμού σε όλα τα μαθήματα που πέρασε για να υπολογίσει τον τελικό βαθμό πτυχίου και ξεκινά μια συναλλαγή, δημιουργώντας το πτυχίο ως Non-Fungible Token (NFT) και μεταφέροντάς το στην πραγματική διεύθυνση του φοιτητή. Στη συνέχεια, το backend καλεί το API του ProofSpace για να εκδόσει το ψηφιακό πιστοποιητικό πτυχίου για τον φοιτητή.

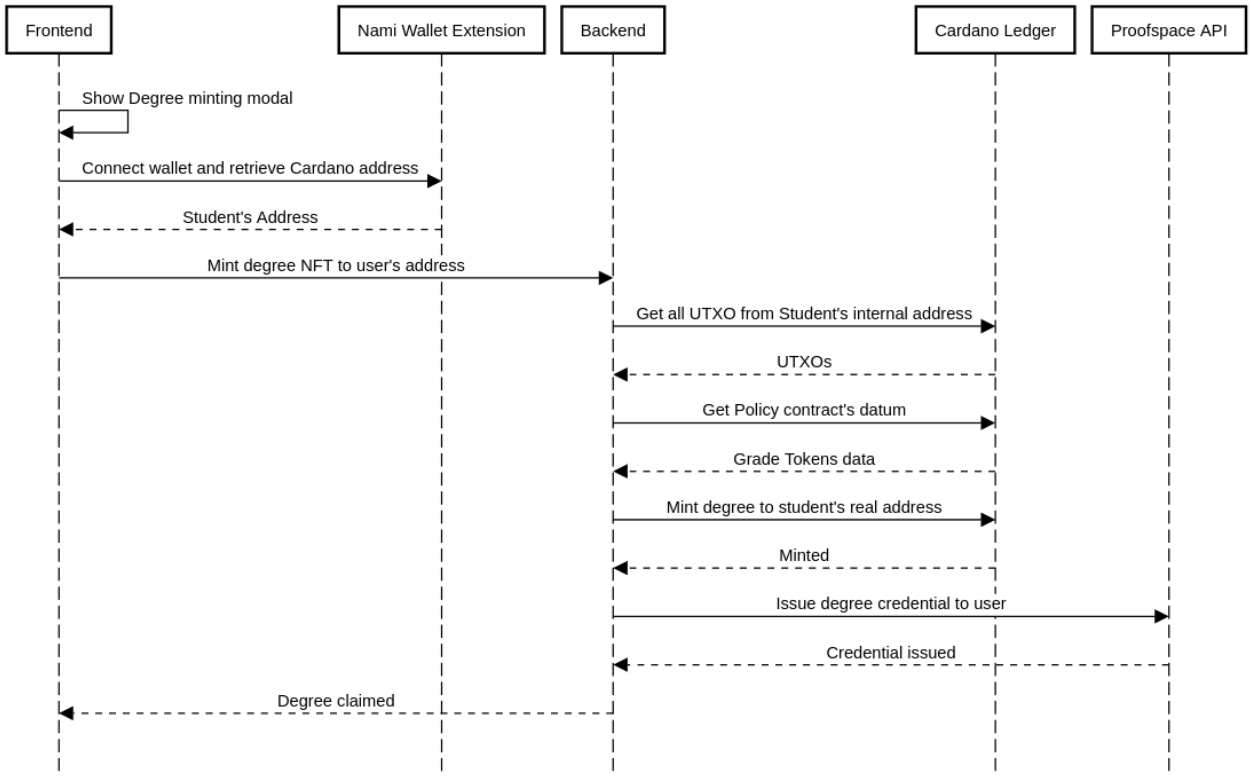


Figure 3.2.4: Διάγραμμα ακολουθίας περίπτωσης απονομής πτυχίου

3.2.5 Περίπτωση Χρήσης 5: Βαθμολόγηση

Η διαδικασία βαθμολόγησης σε ένα μάθημα ξεκινάει με την επιλογή ενός μαθήματος από τον καθηγητή και την αναζήτηση του έξυπνου συμβολαίου του μαθήματος για την ανάκτηση του πίνακα εγγραφών. Στη συνέχεια, το backend λαμβάνει το όνομα και το επώνυμο κάθε εγγεγραμμένου φοιτητή από τη βάση δεδομένων και το επιστρέφει στο frontend. Ο καθηγητής μπορεί να εισάγει βαθμούς για κάθε φοιτητή απευθείας στην διεπαφή και να τους υποβάλει πατώντας το κουμπί υποβολής. Αυτό ενεργοποιεί μια απευθείας κλήση στο έξυπνο συμβόλαιο του μαθήματος, το οποίο μεταφέρει τα κατάλληλα tokens βαθμού στη διεύθυνση κάθε φοιτητή με βάση τους εισαχθέντες βαθμούς. Ο καθηγητής καλείται να υπογράψει τη συναλλαγή για να πιστοποιήσει και να οριστικοποιήσει τη βαθμολόγηση. Μετά από μια επιτυχημένη συναλλαγή, το backend ξεκινά μια κλήση προς το API του ProofSpace για την έκδοση πιστοποιητικών για κάθε φοιτητή, που περιέχουν το όνομα του μαθήματος και τον βαθμό του φοιτητή.

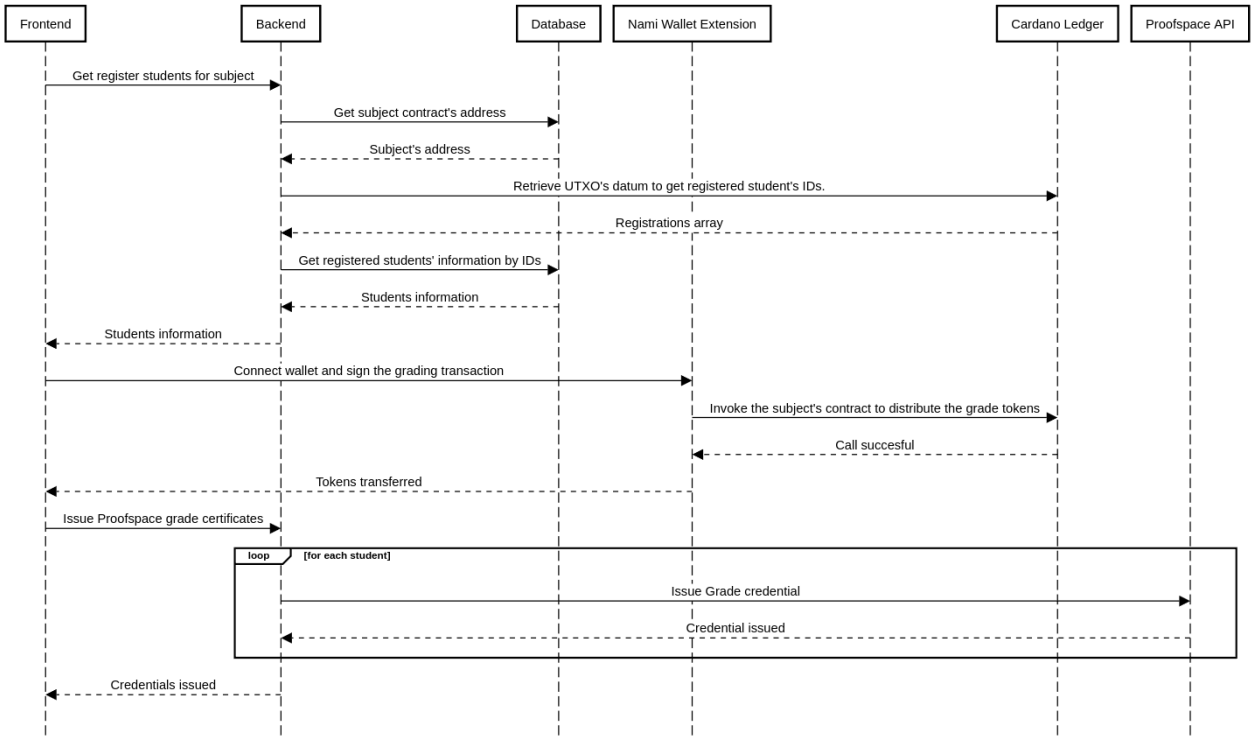


Figure 3.2.5: Διάγραμμα ακολουθίας περίπτωσης βαθμολόγησης

3.2.6 Περίπτωση Χρήσης 6: Εκκίνηση Περιόδου Εγγραφών

Η γραμματεία ξεκινά την περίοδο εγγραφών για την ακαδημαϊκή περίοδο εισάγοντας μια ημερομηνία λήξης μέσω του frontend. Η περίοδος εγγραφών αρχίζει από αυτή την ημερομηνία και ώρα. Το frontend αλληλεπιδρά με το backend καλώντας το σχετικό endpoint και το δεύτερο καλεί τα έξυπνα συμβόλαια για κάθε μάθημα για να ενημερώσει τα δεδομένα του με τη νέα ημερομηνία λήξης.

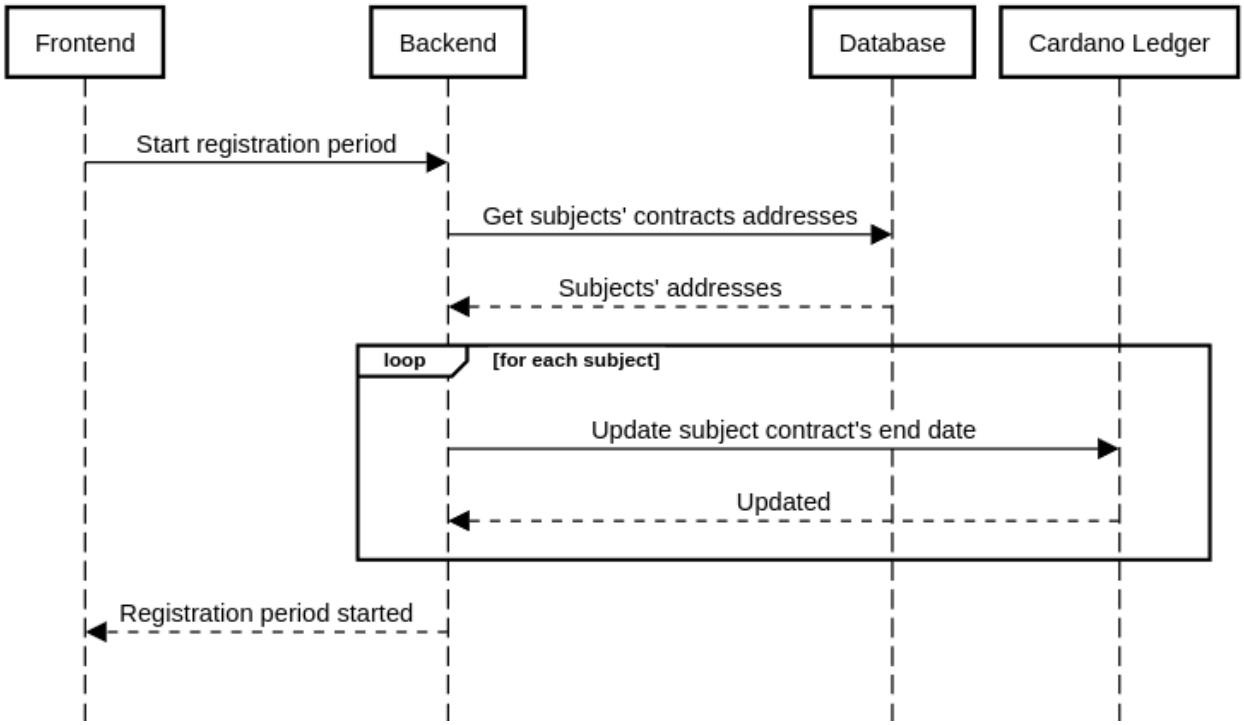


Figure 3.2.6: Διάγραμμα ακολουθίας περίπτωσης εκκίνησης περιόδου εγγραφών

3.2.7 Περίπτωση Χρήσης 7: Αλλαγή Προγράμματος Σπουδών

Σε περίπτωση αλλαγής στα υποχρεωτικά μαθήματα, και στο πρόγραμμα σπουδών, η γραμματεία έχει πρόσβαση σε μια λίστα όλων των μαθημάτων και καλείται να επιλέξει τα απαιτούμενα. Αυτή η επιλογή υποβάλλεται στη συνέχεια σε ένα backend endpoint που διαχειρίζεται τις ενημερώσεις στις απαιτήσεις του πτυχίου. Το backend ανακτά τα tokens βαθμού για τα επιλεγμένα θέματα, τα αναγνωριστικά minting policy ID τους και τα ονόματά τους. Στη συνέχεια, καλεί το συμβόλαιο πολιτικής που είναι υπεύθυνο για τον καθορισμό των απαιτήσεων πτυχίου, ενημερώνοντας τον κατάλογο των απαιτούμενων θεμάτων. Αυτή η τροποποίηση διασφαλίζει ότι οι μελλοντικές επαληθεύσεις πτυχίου βασίζονται στη νέα αλλαγή του προγράμματος σπουδών.

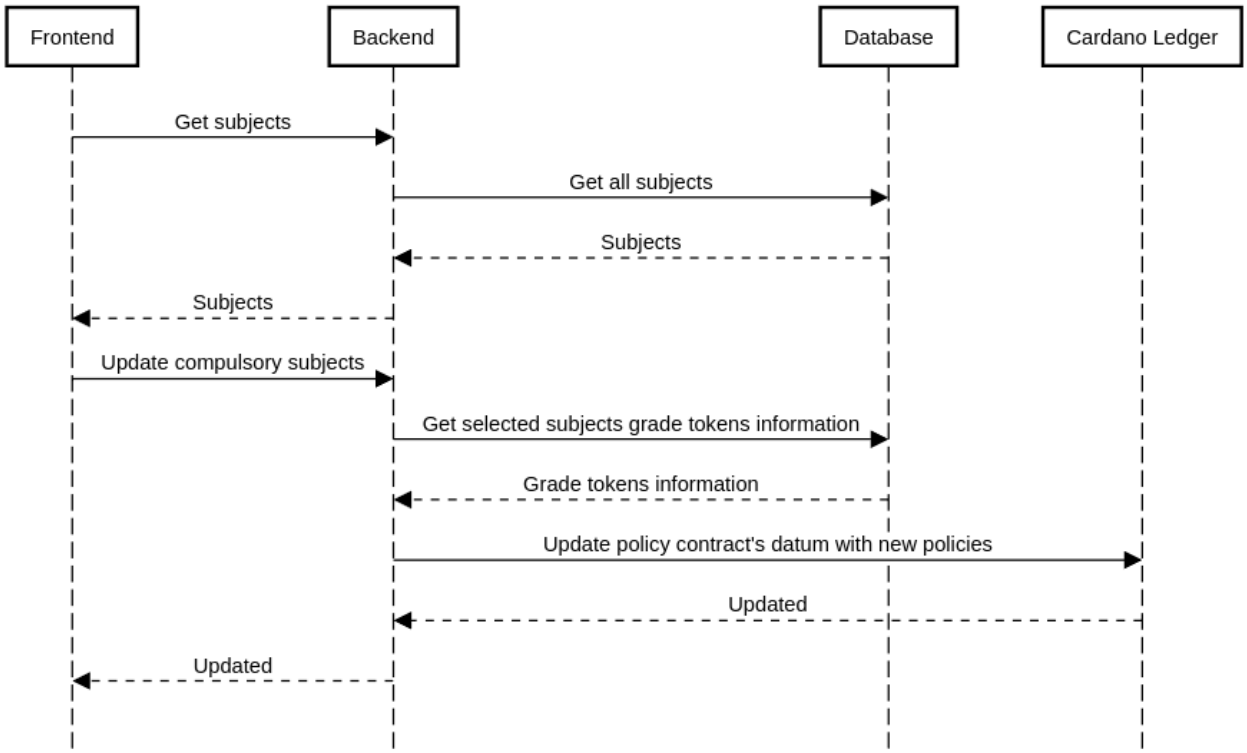


Figure 3.2.7: Διάγραμμα ακολουθίας περίπτωσης αλλαγής προγράμματος σπουδών

Κεφάλαιο 4

Υλοποίηση

4.1 Διεπαφή Χρήστη

Η διεπαφή χρήστη έχει σχεδιαστεί για να εξυπηρετεί τις ανάγκες των ενδιαφερόμενων φορέων, όπως οι φοιτητές, οι καθηγητές και τη γραμματεία. Χρησιμοποιεί το Pug templating enging και προσφέρει προηγμένα χαρακτηριστικά για μια ομαλή εμπειρία. Το frontend, επίσης, χρησιμοποιεί τη βιβλιοθήκη lucid-cardano για άμεσες αλληλεπιδράσεις με το blockchain, επιτρέποντας συναλλαγές και ανάκτηση δεδομένων. Η διεπαφή επιτρέπει στους χρήστες να αλληλεπιδρούν με το πορτοφόλι Nami, επιτρέποντας ασφαλείς συναλλαγές μέσα από το browser extension. Τέλος, το Blockfrost χρησιμεύει ως πάροχος υπηρεσιών blockchain, διασφαλίζοντας την απρόσκοπτη αλληλεπίδραση μεταξύ της DApp και του δικτύου Cardano.

4.1.1 Φοιτητές

4.1.1.1 Frontend

Η διεπαφή του φοιτητή έχει σχεδιαστεί για να είναι φιλική προς το χρήστη και απλή. Μιμείται τις παραδοσιακές αλληλεπιδράσεις εφαρμογών ιστού, αποκρύπτοντας τις αλληλεπιδράσεις με το Cardano blockchain στο μεγαλύτερο δυνατό βαθμό.

Όταν οι φοιτητές συνδέονται για πρώτη φορά, τους παρουσιάζεται ένας κωδικός QR που πρέπει να σαρώσουν χρησιμοποιώντας την εφαρμογή ProofSpace. Αυτό το βήμα είναι απαραίτητο για την ολοκλήρωση της εγγραφής τους στο πανεπιστήμιο και τη σύνδεση του ακαδημαϊκού τους προφίλ με την αποκεντρωμένη ταυτότητά τους.

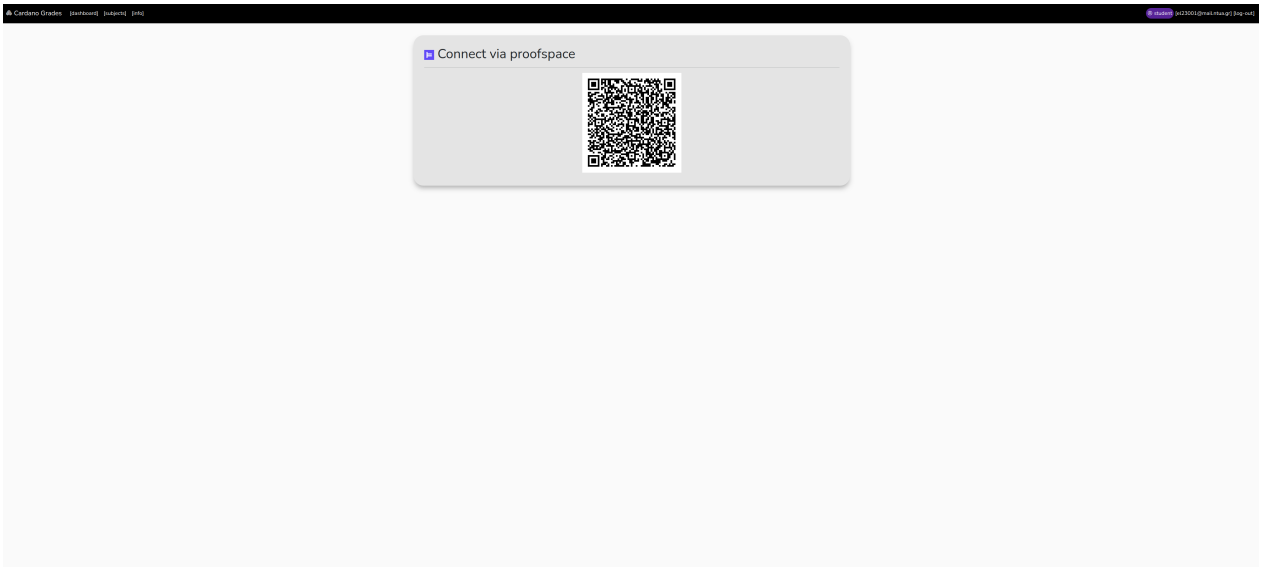


Figure 4.1.1: Σελίδα σάρωσης QR μετά την πρώτη σύνδεση

Για όλες τις επόμενες συνδέσεις, οι φοιτητές οδηγούνται στην αρχική σελίδα. Αυτή η διεπαφή εμφανίζει τους βαθμούς των μαθημάτων που έχουν περάσει, παρέχοντας μια γρήγορη επισκόπηση της προόδου τους.

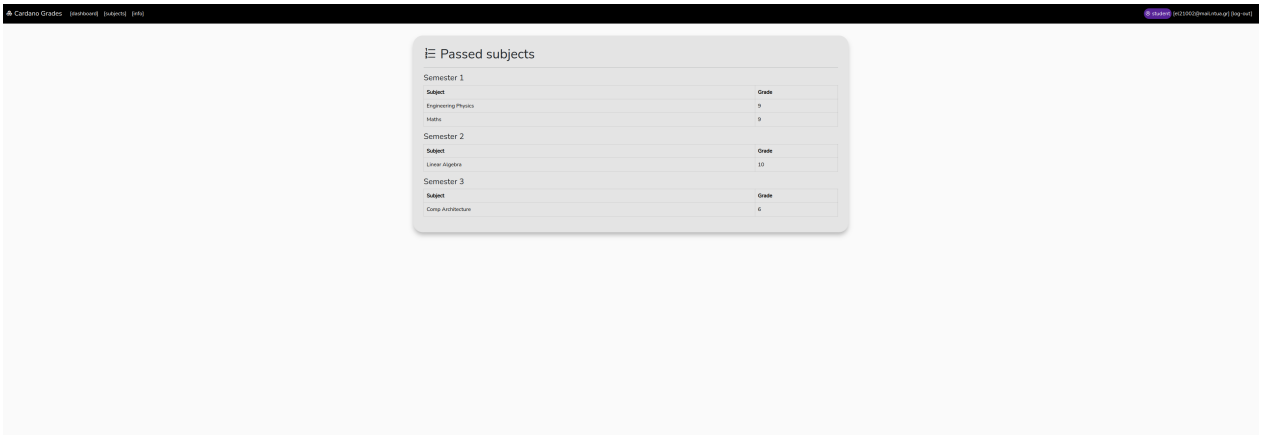


Figure 4.1.2: Αρχική σελίδα φοιτητή

Μια ακόμα σελίδα είναι η διεπαφή εγγραφής για το εξάμηνο, όπου οι φοιτητές μπορούν να δουν όλα τα διαθέσιμα μαθήματα για το τρέχον εξάμηνο. Μπορούν να επιλέξουν τα μαθήματα στα οποία επιθυμούν να εγγραφούν τσεκάροντας τα κουτάκια δίπλα σε κάθε μάθημα και στη συνέχεια να υποβάλουν τις επιλογές τους.

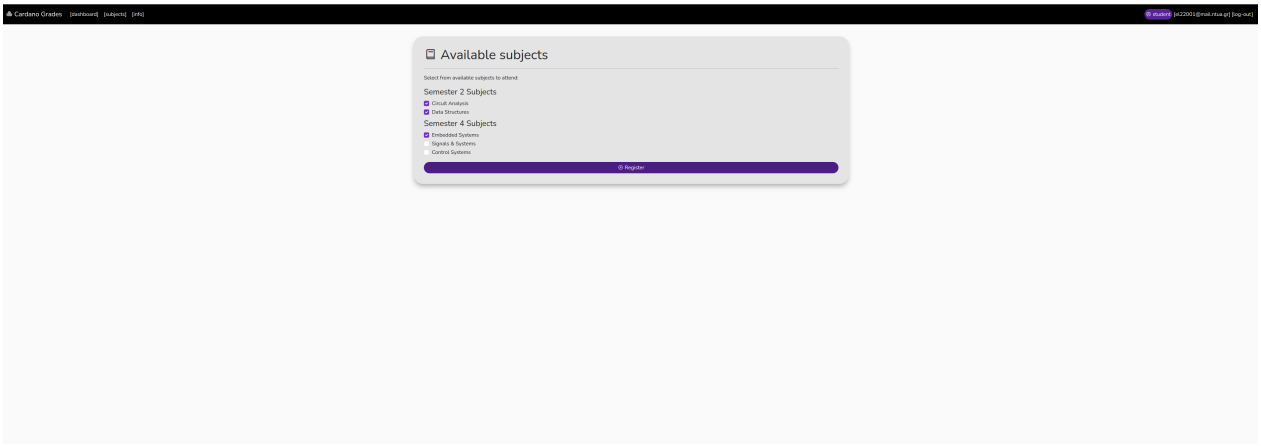


Figure 4.1.3: Σελίδα δήλωσης μαθημάτων νέου εξαμήνου

Επίσης, υπάρχει μια σελίδα όπου μπορούν να βλέπουν και να ενημερώνουν τα προσωπικά τους στοιχεία, όπως το email ή τον κωδικό πρόσβασης.

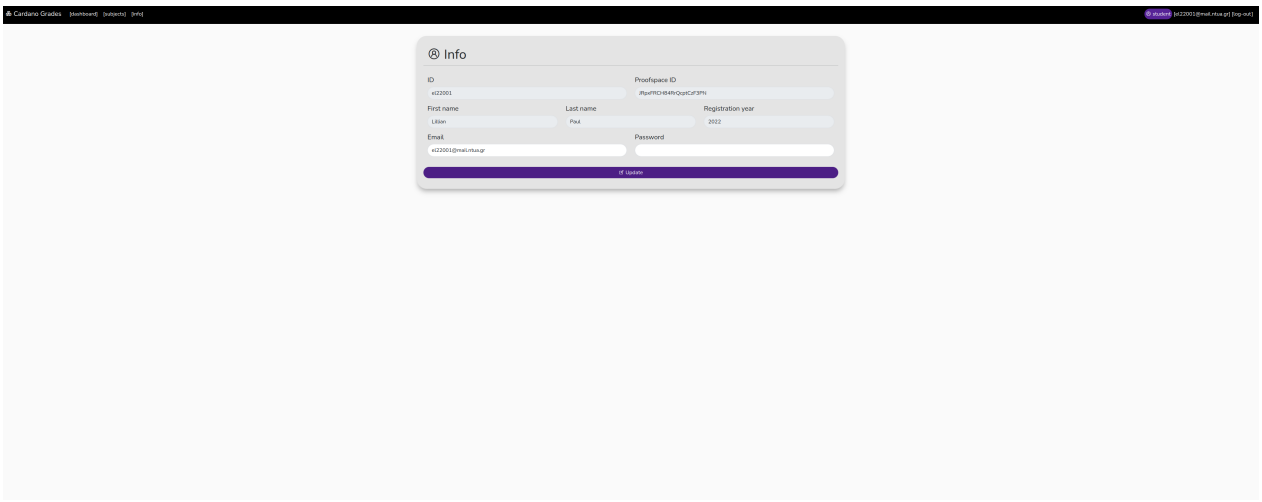


Figure 4.1.4: Προφίλ χρήστη φοιτητή

Τέλος, αφού οι φοιτητές ολοκληρώσουν όλα τα υποχρεωτικά μαθήματα, μια ειδοποίηση τους προτρέπει να παραλάβουν το πτυχίο τους. Αυτή η διαδικασία περιλαμβάνει ένα modal όπου οι φοιτητές καλούνται να συνδέσουν το πορτοφόλι τους. Αυτή είναι η μόνη φορά που οι φοιτητές καλούνται να αλληλεπιδράσουν απευθείας με το Cardano χρησιμοποιώντας ένα πορτοφόλι, το οποίο είναι απαραίτητο για την παραλαβή του πτυχίου τους ως NFT.

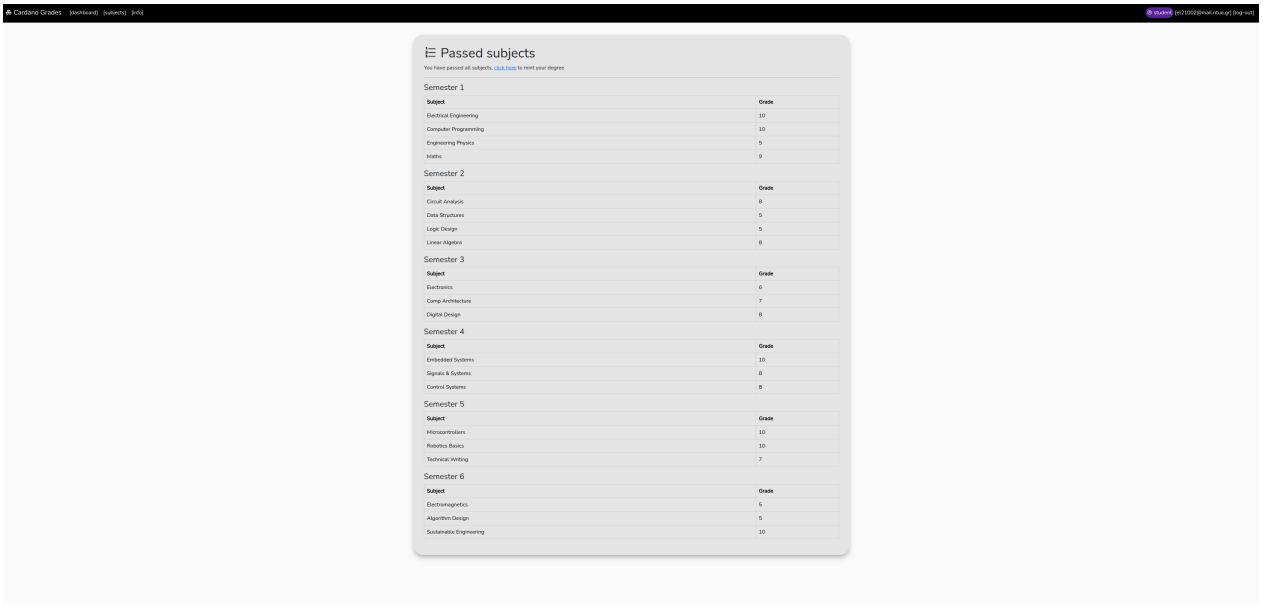


Figure 4.1.5: Μήνυμα παραλαβής πτυχίου στην αρχική σελίδα

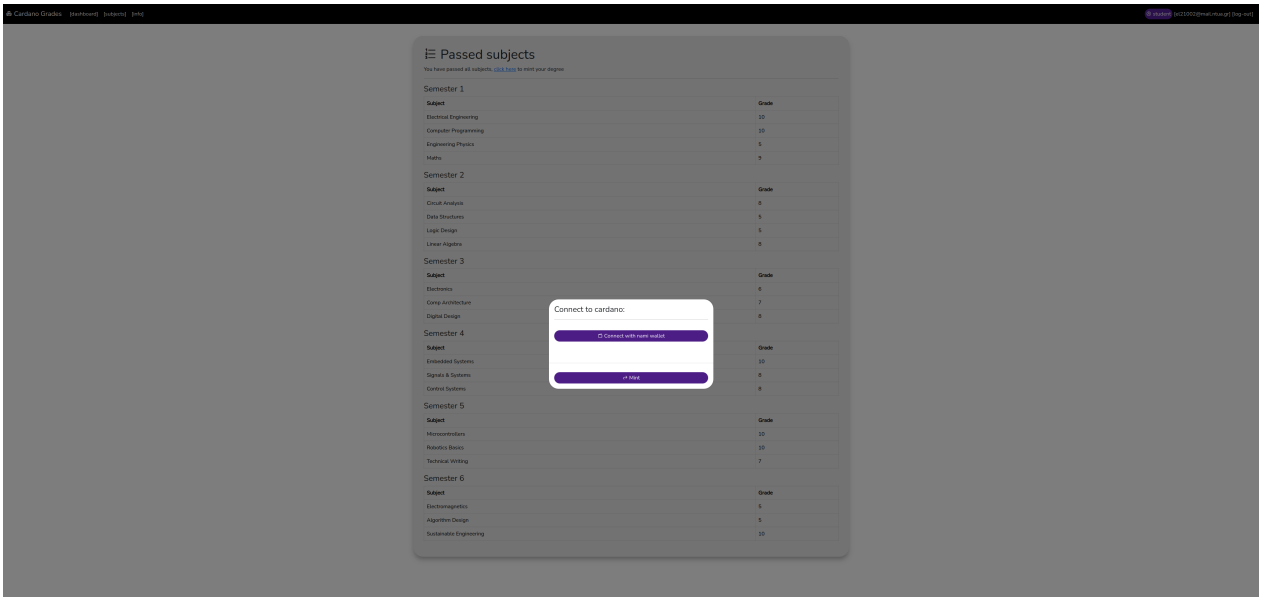


Figure 4.1.6: Modal παραλαβής πτυχίου

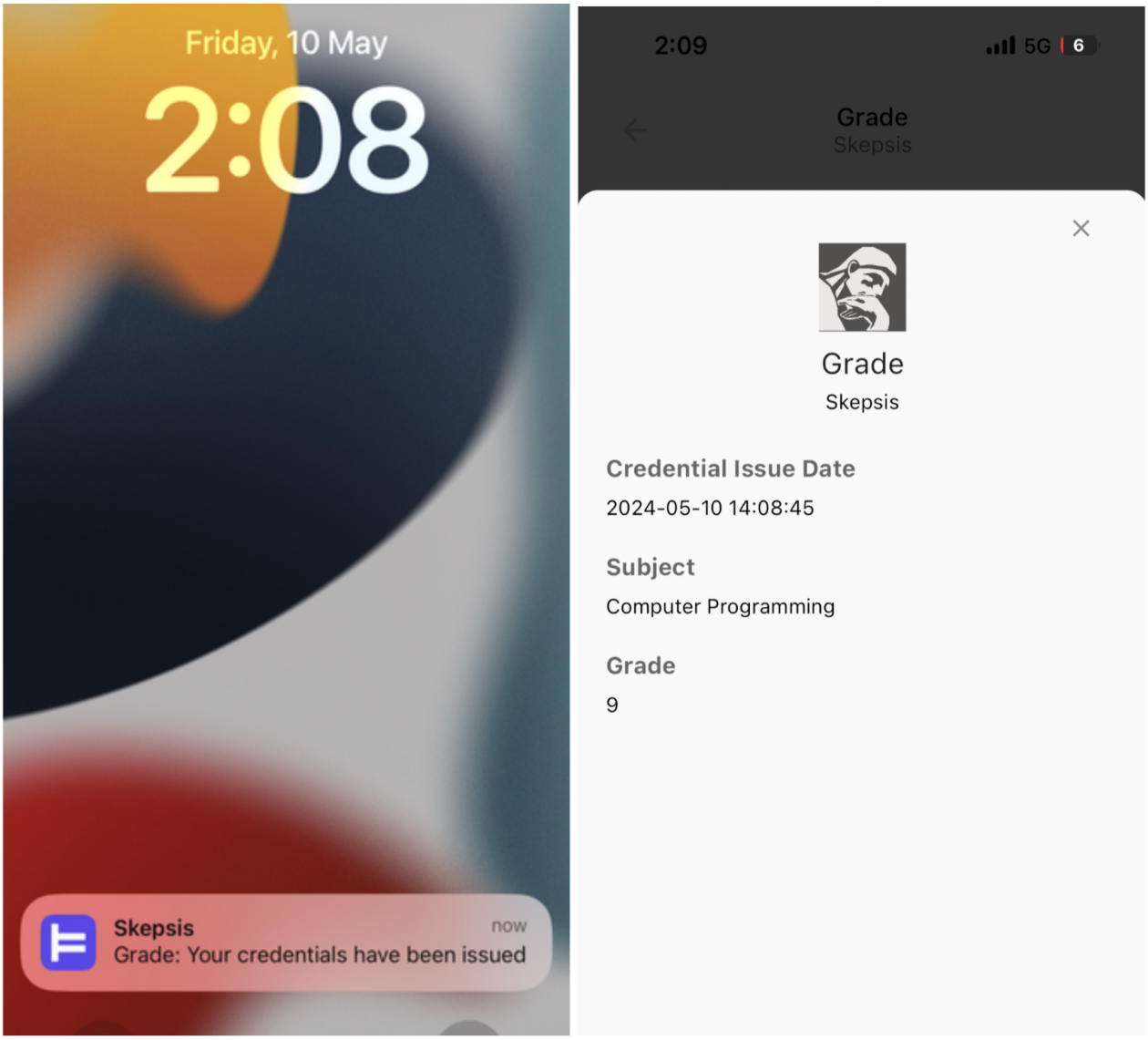
4.1.1.2 Εφαρμογή ProofSpace

Η εφαρμογή ProofSpace λειτουργεί ως δευτερεύουσα διεπαφή χρήστη για τους φοιτητές, συμπληρώνοντας την κύρια διεπαφή, προσφέροντας ένα ολοκληρωμένο ψηφιακό πορτοφόλι για τα ακαδημαϊκά πιστοποιητικά. Μέσω του ProofSpace, οι φοιτητές λαμβάνουν και αποθηκεύουν ψηφιακά πιστοποιητικά που σχετίζονται με διάφορες πτυχές της ακαδημαϊκής τους διαδρομής:

- Εγγραφή στο Πανεπιστήμιο: Μετά την ολοκλήρωση της αρχικής διαδικασίας εγγραφής, οι φοιτητές λαμβάνουν ένα ψηφιακό πιστοποιητικό που επιβεβαιώνει την εγγραφή τους στο πανεπιστήμιο.
- Εγγραφή ανά εξάμηνο: Κάθε φορά που ένας φοιτητής εγγράφεται σε ένα νέο εξάμηνο, λαμβάνει ένα πιστοποιητικό που αποδεικνύει την εγγραφή του.

- Βαθμός Μαθήματος: Αφού βαθμολογήσουν οι καθηγητές κάποιου μαθήματος, οι φοιτητές λαμβάνουν πιστοποιητικά για κάθε μάθημα, συμπεριλαμβανομένων εκείνων στα οποία δεν πέρασαν, επιτρέποντάς τους να διατηρούν ένα πλήρες αρχείο των επιδόσεων τους.
- Πτυχίο: Μετά την επιτυχή ολοκλήρωση όλων των απαιτούμενων μαθημάτων και τη λήψη του πτυχίου τους, εκδίδεται ένα τελικό πιστοποιητικό που χρησιμεύει ως ψηφιακή έκδοση του πτυχίου.

Η εφαρμογή ProofSpace λειτουργεί ως αρχείο της ακαδημαϊκής κατάστασης ενός φοιτητή, αποθηκεύοντας τα πιστοποιητικά και καθιστώντας τα προσβάσιμα ανά πάσα στιγμή. Αυτή η ρύθμιση είναι επωφελής, καθώς επιτρέπει στους φοιτητές να βλέπουν όλους τους βαθμούς τους, συμπεριλαμβανομένων των μη επιτυχόντων, τους οποίους δεν εμφανίζει η κύρια διεπαφή χρήστη. Επιπλέον, η εφαρμογή βελτιώνει την εμπειρία των φοιτητών στέλνοντας ειδοποιήσεις κάθε φορά που εκδίδεται ένα νέο πιστοποιητικό, διασφαλίζοντας ότι οι φοιτητές ενημερώνονται άμεσα για ενημερώσεις όπως οι νέοι βαθμοί.



(a) Proofspace new grade notification.

(b) Proofspace new grade credential.

Figure 4.1.7: Εφαρμογή ProofSpace

4.1.2 Καθηγητές

Η διεπαφή των εκπαιδευτικών έχει σχεδιαστεί για να διευκολύνει το έργο της βαθμολόγησης, ευθυγραμμιζόμενη με τον στόχο της εφαρμογής να απλοποιήσει την αλληλεπίδραση με την τεχνολογία blockchain. Ωστόσο, η διαδικασία προσθήκης βαθμών περιλαμβάνει άμεση αλληλεπίδραση με το δίκτυο, καθώς οι εκπαιδευτικοί είναι υπεύθυνοι για την αποστολή της συναλλαγής που απονέμει τους βαθμούς στους φοιτητές.

Η αρχική σελίδα των καθηγητών περιλαμβάνει έναν κατάλογο όλων των μαθημάτων που διδάσκουν.

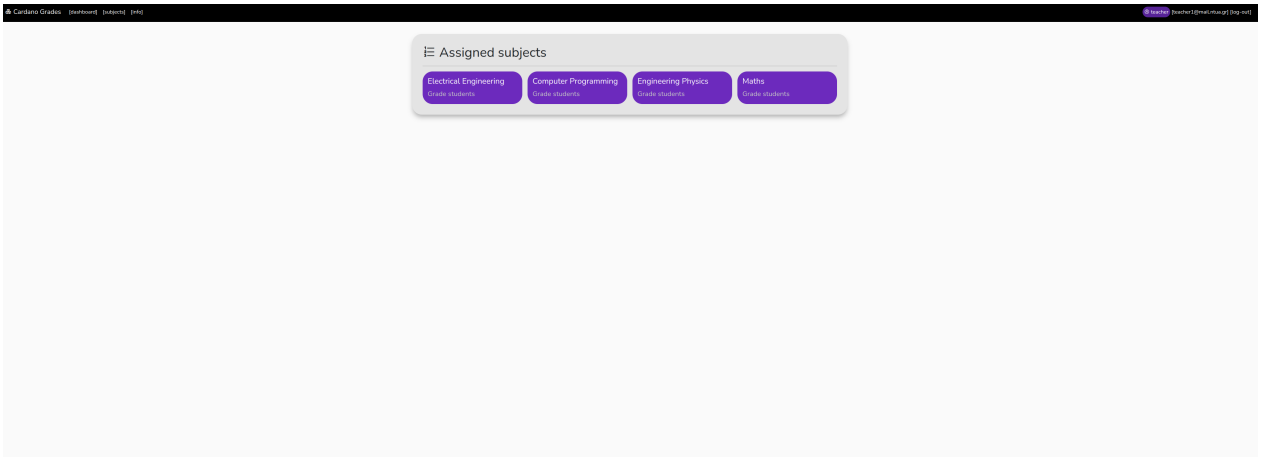


Figure 4.1.8: Αρχική σελίδα καθηγητών

Έπειτα, μπορούν να επιλέξουν οποιοδήποτε μάθημα, να δουν μια λίστα με όλους τους εγγεγραμμένους φοιτητές και να εισάγουν βαθμούς. Η διαδικασία βαθμολόγησης διευκολύνεται από τη βιβλιοθήκη cardano-lucid, η οποία χειρίζεται τις λεπτομέρειες της αλληλεπίδρασης με το Cardano. Η σχετική λειτουργία ανακτά πρώτα τη διεύθυνση του έξυπνου συμβολαίου και τα σχετικά UTXOs που απαιτούνται για τη συναλλαγή. Στη συνέχεια διαμορφώνει τις εξόδους της συναλλαγής για να διασφαλίσει ότι η διανομή των tokens βαθμού στις διευθύνσεις των φοιτητών εκτελείται σωστά, διαφορετικά ο Validator του μαθήματος θα αποτύχει. Είναι απαραίτητο για όλους τους καθηγητές να έχουν εγκατεστημένο ένα πορτοφόλι Nami στο πρόγραμμα περιήγησής τους, με μια διεύθυνση Cardano που περιέχει κάποια ADA για την πληρωμή των transaction fees, προκειμένου να εκτελεστεί αυτή η ενέργεια.

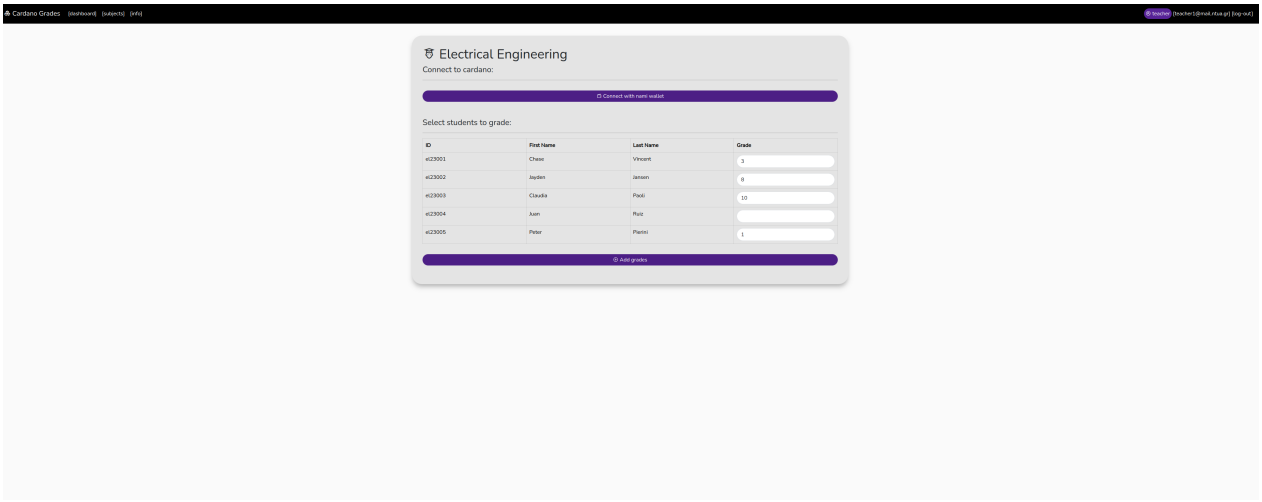


Figure 4.1.9: Σελίδα βαθμολόγησης

Τέλος, όπως και οι φοιτητές, οι καθηγητές έχουν τη δυνατότητα να βλέπουν και να τροποποιούν τα στοιχεία

τους μέσω της καθορισμένης σελίδας.

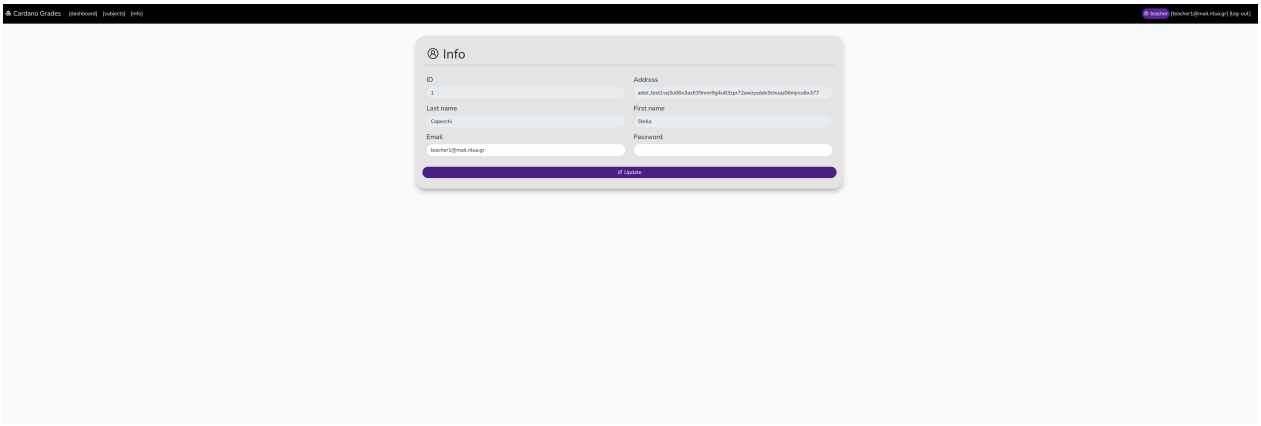


Figure 4.1.10: Προφίλ Καθηγητών

4.1.3 Γραμματεία

Η διεπαφή της γραμματείας αποτελείται από μία σελίδα, η οποία χωρίζεται σε 2 μέρη:

- Διαχείριση περιόδου εγγραφών: Στο πάνω μισό υπάρχει ένα πεδίο για εισαγωγή ημερομηνίας, όπου ο γραμματέας μπορεί να ορίσει την ημερομηνία λήξης της εγγραφής για την ακαδημαϊκή περίοδο. Με την εισαγωγή μιας ημερομηνίας ξεκινά ουσιαστικά η περίοδος εγγραφών του εξαμήνου, η οποία αρχίζει αμέσως και ολοκληρώνεται την καθορισμένη ημερομηνία λήξης.
- Διαχείρηση προγράμματος σπουδών: Στο κάτω μέρος της αρχικής σελίδας παρουσιάζεται ένας πλήρης κατάλογος όλων των μαθημάτων του πανεπιστημίου, τα οποία ανακτώνται από τη βάση δεδομένων. Όταν υπάρχει ανάγκη επικαιροποίησης των απαιτήσεων πτυχίου - λόγω αλλαγών στο πρόγραμμα σπουδών - η γραμματεία μπορεί να επιλέξει από αυτά τα μαθήματα για να τα ορίσει ως υποχρεωτικά για την απόκτηση πτυχίου.

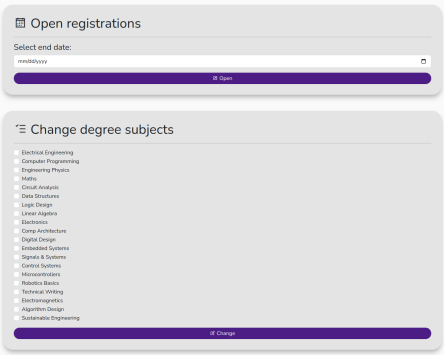


Figure 4.1.11: Σελίδα γραμματείας

4.2 Έξυπνα Συμβόλαια

Τα έξυπνα συμβόλαια λειτουργούν ως ο βασικός μηχανισμός για την επιβολή και τη διαχείριση της λογικής της εφαρμογής στο Cardano Blockchain. Είναι γραμμένα σε Aiken[27], μια γλώσσα προγραμματισμού που αναπτύχθηκε από την TxPipe ειδικά για τη δημιουργία συμβολαίων Cardano. Τα έξυπνα συμβόλαια της εφαρμογής θεσπίζουν τους αμετάβλητους κανόνες κάτω από τους οποίους πρέπει να λειτουργεί το σύστημα. Αυτοί οι κανόνες καθορίζουν τις συνθήκες υπό τις οποίες τα διάφορα tokens - βαθμού, πτυχίου και εγγραφής - μπορούν να κοπούν ή να καούν. Επιπλέον, διαχειρίζονται τις διαδικασίες εγγραφής, περιγράφοντας λεπτομερώς υπό ποιες προϋποθέσεις ένας φοιτητής μπορεί να εγγραφεί σε ένα μάθημα ή να λάβει βαθμούς και επιτρέπουν επίσης την ενημέρωση των απαιτήσεων του προγράμματος σπουδών.

4.2.1 Student Validator

Ο Student Validator χρησιμεύει ως η εσωτερική διεύθυνση Cardano για κάθε φοιτητή. Ο σκοπός του είναι να λαμβάνει και να αποθηκεύει τα tokens βαθμολογίας, μη επιτρέποντας στους φοιτητές να αλληλεπιδράσουν με αυτά. Η επιβολή αυτού του περιορισμού είναι σημαντική για τη διατήρηση της ακεραιότητας και της μονιμότητας των ακαδημαϊκών αρχείων. Ο επικυρωτής έχει μια άμεση λειτουργία: αποτρέπει κάθε συναλλαγή που προσπαθεί να χρησιμοποιήσει τα UTXO της διεύθυνσης αυτής. Κατά συνέπεια, μόλις τοποθετηθούν tokens στη διεύθυνση ενός φοιτητή, παραμένουν εκεί για πάντα, λειτουργώντας ως αμετάβλητο αρχείο των ακαδημαϊκών επιτευγμάτων του. Δεδομένου ότι ο επικυρωτής δεν επιτυγχλανει ποτέ όταν επιχειρεί να καταναλώσει ένα UTXO, δεν απαιτεί ούτε Datum ούτε Redeemer.

4.2.1.1 Παράμετροι

Ο Student Validator δέχεται μία παράμετρο τύπου "bytes", τον αριθμό μητρώου του εκάστοτε φοιτητή, επιτρέποντας τη δημιουργία μοναδικών διευθύνσεων.

4.2.2 Subject Validator

Ο Subject Validator χρησιμεύει ως "αποθήκη" για τα tokens βαθμού του εκάστοτε μαθήματος, διασφαλίζοντας ότι είναι κλειδωμένα και μπορούν να μεταφερθούν μόνο υπό συγκεκριμένες προϋποθέσεις. Οι λειτουργίες αυτού του συμβολαίου είναι τρεις: επιτρέπει την εγγραφή και την παρακολούθηση των φοιτητών που εγγράφονται στο μάθημα, επιτρέπει στη γραμματεία να ενημερώνει παραμέτρους, όπως τον καθηγητή ή την ημερομηνία λήξης της περιόδου εγγραφών και επιτρέπει στον καθηγητή να αποδίδει βαθμούς.

4.2.2.1 Παράμετροι

Ο Subject Validator δέχεται με τέσσερις παραμέτρους:

- Μία τύπου "PolicyID" για το αναγνωριστικό πολιτικής του token βαθμού του μαθήματος. Αυτό βοηθά στην αναγνώριση και διαχείριση των συγκεκριμένων tokens βαθμού.
- Ένα "AssetName" που αντιπροσωπεύει το όνομα του token βαθμού.
- Μία τύπου "PubKeyHash", που αποθηκεύει το δημόσιο κλειδί Cardano της γραμματείας. Χρησιμοποιείται ως Access control για τις ενέργειες που απαιτούν την εξουσιοδώτηση της γραμματείας.
- Άλλο ένα "PolicyID" για το αναγνωριστικό της πολιτικής του token εγγραφής, το οποίο χρησιμοποιείται σε ελέγχους επαλήθευσης για την επιβεβαίωση ότι ο εκάστοτε φοιτητής είναι πράγματι εγγεγραμμένος στο μάθημα.

4.2.2.2 Datum

Το Datum του έξυπνου συμβολαίου αποτελείται από τέσσερα πεδία:

- Δημόσιο κλειδί του καθηγητή: Ένα "pubKeyHash" που αποθηκεύει το δημόσιο κλειδί Cardano του καθηγητή που είναι υπεύθυνος για το μάθημα.

- Πίνακας εγγραφών: Ένας πίνακας από bytes που χρησιμοποιείται για την αποθήκευση των αριθμών μητρώων των φοιτητών που έχουν εγγραφεί στο μάθημα. Αυτό επιτρέπει στο σύστημα να διατηρεί ένα δυναμικό αρχείο της συμμετοχής των φοιτητών και της επιλεξιμότητας για τη λήψη βαθμών.
- Registration Closure Time: Ένα "POSIXTime", που δηλώνει την ακριβή ώρα σε δευτερόλεπτα που κλείνουν οι εγγραφές για το μάθημα.
- Reset Flag: Ένα πεδίο "Bool" με το όνομα reset δηλώνει αν ο πίνακας εγγραφών πρέπει να διαγραφεί μετά τον επόμενο κύκλο βαθμολόγησης.

4.2.2.3 Επιλογές Redeemer

Register Ο redeemer "Register" έχει σχεδιαστεί για να προσθέτει τους αριθμούς μητρώου των φοιτητών στον πίνακα εγγραφών. Για να πετύχει η συναλλαγή, πρέπει:

- Η κλήση να γίνει από τη γραμματεία.
- Να δημιουργείται μία έξοδος στην ίδια διεύθυνση, με τα τα ίδια ακριβώς Datum, εκτός από την προσθήκη του νέου αριθμού μητρώου στον πίνακα εγγραφών.
- Ο αριθμός των tokens βαθμών να παραμένει αμετάβλητος, επιβεβαιώνοντας ότι δεν υπάρχει κάποια μη εξουσιοδοτημένη μεταφορά.
- Τέλος, πρέπει να μην έχει παρέλθει η περίοδος εγγραφών.

Update Ο redeemer "Update" χρησιμοποιείται για την αλλαγή του καθηγητή του μαθήματος και για την ενημέρωση της καταληκτικής ημερομηνίας της περιόδου εγγραφών. Για να πετύχει, πρέπει:

- Η κλήση να γίνει από τη γραμματεία.
- Να δημιουργείται μία έξοδος στην ίδια διεύθυνση, με τα τα ίδια ακριβώς Datum, εκτός από το δημόσιο κλειδί του καθηγητή και την ημερομηνία εγγραφών, τα οποία μπορεί να έχουν ενημερωθεί.
- Ο αριθμός των tokens βαθμών να παραμένει αμετάβλητος, επιβεβαιώνοντας ότι δεν υπάρχει κάποια μη εξουσιοδοτημένη μεταφορά.

Grade Ο redeemer "Grade" χρησιμοποιείται για την αποστολή των tokens βαθμών στους φοιτητές. Οι συνθήκες για την επίτευξη της συναλλαγής είναι οι ακόλουθες:

1. Η κλήση να γίνει από τον καθηγητή του μαθήματος.
2. Η συναλλαγή δαπάνης πρέπει να περιλαμβάνει ως εισόδους αναφοράς όλα τα UTXOs που περιέχουν το token εγγραφής στο πανεπιστήμιο των φοιτητών που βαθμολογούνται. Αυτό επιτρέπει στο έξυπνο συμβόλαιο να ανακτήσει το αναγνωριστικό κάθε φοιτητή (κωδικοποιημένο ως όνομα του token εγγραφής του φοιτητής) και να επαληθεύσει ότι εμφανίζεται στον πίνακα εγγραφών του θέματος.
3. Για κάθε token βαθμού που αποστέλλεται σε μια εξωτερική διεύθυνση, πρέπει να υπάρχει ένα αντίστοιχο UTXO στην εν λόγω διεύθυνση που να περιέχει το κουπόνι εγγραφής του φοιτητή, διασφαλίζοντας την επιλεξιμότητα του για τη λήψη βαθμών. Η ποσότητα των tokens βαθμού σε αυτά τα outputs πρέπει να είναι μεταξύ 5 και 10. Όλα τα υπόλοιπα tokens πρέπει να επιστρέφονται σε μία έξοδο στη διεύθυνση του έξυπνου συμβολαίου του μαθήματος, διατηρώντας το υπόλοιπο μέρος της ρευστότητας κλειδωμένο. Τέλος, το άθροισμα όλων των tokens βαθμού που υπάρχουν στις εξόδους πρέπει να είναι ίσο με την ποσότητα των μαρκών βαθμού στο συμβόλαιο, διασφαλίζοντας ότι δεν θα καούν κατά τη διαδικασία.
4. Το "reset" flag χρησιμοποιείται για τη διαχείριση της διαδικασίας βαθμολόγησης σε κανονικές και επαναληπτικές εξετάσεις. Αρχικά θέτεται σε false και πρέπει να οριστεί σε true αφού ο καθηγητής βαθμολογήσει για πρώτη φορά. Η επακόλουθη βαθμολόγηση (για την επαναληπτική εξεταστική) με τη σημαία reset true απαιτεί την επαναφορά της σημαίας σε false και την εκκαθάριση του πίνακα εγγραφών, ετοιμάζοντας το σύστημα για τις εγγραφές του επόμενου εξαμήνου.

Τέλος, εμβαθύνουμε στη λογική υλοποίησης του αλγορίθμου επικύρωσης βαθμών, από το βήμα 3, ο οποίος αποτελεί τον πυρήνα του συστήματος.

- Για κάθε έξοδο που περιέχει tokens βαθμού:
  - Εάν η διεύθυνση εξόδου είναι η ίδια διεύθυνση του υποκειμένου συμβολαίου, η διαδικασία συνεχίζεται χωρίς περαιτέρω ελέγχους.
  - Εάν η διεύθυνση εξόδου είναι εξωτερική:
    * Η ποσότητα των tokens βαθμού πρέπει να εμπίπτει στο εύρος [5,10].
    * Μια είσοδος αναφοράς στην ίδια διεύθυνση πρέπει να περιέχει το token εγγραφής του φοιτητή, το οποίο προσδιορίζεται με την παράμετρο PolicyID εγγραφής. Το όνομα αυτού του token, το οποίο είναι το αναγνωριστικό του φοιτητή, πρέπει να περιέχεται στον πίνακα εγγραφών- διαφορετικά, η συναλλαγή θα αποτύχει, μπλοκάροντας τις προσπάθειες βαθμολόγησης μη εγγεγραμμένων φοιτητών.

4.2.3 Policy Data Validator

Το συμβόλαιο πολιτικής χρησιμεύει ως μητρώο για όλα τα έγκυρα tokens βαθμού του συστήματος. Είναι απαραίτητο για την προστασία από πιθανές απειλές ασφαλείας, όπου κακόβουλοι φορείς ενδέχεται να επιχειρήσουν να δημιουργήσουν και να χρησιμοποιήσουν πλαστά tokens βαθμού. Διατηρώντας ένα προστατευμένο αρχείο των αποδεκτών αναγνωριστικών πολιτικής και των ονομάτων των tokens βαθμών, το συμβόλαιο πολιτικής διασφαλίζει ότι μόνο τα εξουσιοδοτημένα διακριτικά αναγνωρίζονται και χρησιμοποιούνται εντός του συστήματος. Τέλος, το έξυπνο συμβόλαιο επιτρέπει στον γραμματέα να επικαιροποιεί τις πολιτικές, προκειμένου να προσαρμόζεται σε τυχόν αλλαγές στο πρόγραμμα σπουδών.

4.2.3.1 Παραμέτροι

Δέχεται μόνο μία παράμετρο, το δημόσιο κλειδί της γραμματείας.

4.2.3.2 Datum

Οι έγκυρες πολιτικές και ονόματα tokens αποθηκεύονται στο datum ως ένας ενιαίος πίνακας, διευκολύνοντας τις αναβαθμίσεις. Αυτός ο πίνακας αποτελείται από τούπλες, κάθε μία από τις οποίες είναι τύπου "(PolicyID, AssetName)". Αυτή η δομή επιτρέπει την εύκολη αποθήκευση και ανάκτηση των tokens βαθμού που αναγνωρίζονται στο σύστημα.

4.2.3.3 Redeemer Options

Το συμβόλαιο πολιτικών περιλαμβάνει μόνο ένα redeemer, σχεδιασμένο για την ενημέρωση του datum. Για να πετύχει η συναλλαγή, πρέπει να κληθεί από τη γραμματεία και να μην αλλάξει ο αριθμός των tokens που περιέχονται στην έξοδο.

4.2.4 Grade Token Minting Policy

Το token βαθμού αντιπροσωπεύει την ακαδημαϊκή επίδοση ενός φοιτητή σε κάθε μάθημα, με κάθε μάθημα να έχει το δικό του μοναδικό κουπόνι με απεριόριστη προμήθεια. Το Asset Name του token ταιριάζει με το όνομα του μαθήματος και όλα τα token είναι ασφαλώς κλειδωμένα μέσα στο αντίστοιχο έξυπνο συμβόλαιο.

4.2.4.1 Παράμετροι

Το έξυπνο συμβόλαιο Grade Token παραμετροποιείται με δύο τύπους:

- Μια αναφορά UTXO
- Ένα AssetName, το οποίο προσδιορίζει το όνομα που θα έχει το token και αντιστοιχεί στο όνομα του υποκειμένου.

4.2.4.2 Συνθήκες νομισματοκοπίας

Οι προϋποθέσεις υπό τις οποίες μπορούν να κοπούν μάρκες βαθμού είναι οι εξής:

- Το Asset Name πρέπει να ταιριάζει με την αντίστοιχη παράμετρο που καθορίζεται στο συμβόλαιο. Αυτή η ευθυγράμμιση διασφαλίζει ότι κάθε token συσχετίζεται σωστά με το προβλεπόμενο μάθημα της.
- Η συναλλαγή κοπής πρέπει να καταναλώνει το συγκεκριμένο UTXO με το οποίο παραμετροποιήθηκε η πολιτική. Απαιτώντας την κατανάλωση ενός καθορισμένου UTXO, το συμβόλαιο διασφαλίζει ότι μόλις το συγκεκριμένο UTXO καταναλωθεί, δεν μπορεί να χρησιμοποιηθεί ποτέ ξανά, εξαλείφοντας έτσι την πιθανότητα επαναχρησιμοποίησης των συνθηκών κοπής.

4.2.5 Registration Token Minting Policy

Η πολιτική κοπής tokens εγγραφής διευκολύνει την παρακολούθηση των εγγραφών στο πανεπιστήμιο και σε εξάμηνα. Συγκεκριμένα:

- Πανεπιστημιακές εγγραφές: Όταν ένας φοιτητής εγγράφεται στο πανεπιστήμιο, ένα token κόβεται και αποστέλλεται στην εσωτερική διεύθυνση του φοιτητή. Το asset name αυτού του κουπονιού αντιστοιχεί στον αριθμό μητρώου, συνδέοντας μοναδικά το κουπόνι με την εγγραφή του φοιτητή στο πανεπιστήμιο.
- Εγγραφή σε εξάμηνο: Για εγγραφές σε εξάμηνο, το όνομα περιουσιακού στοιχείου του κουπονιού δηλώνει τον αριθμό του εξαμήνου στο οποίο εγγράφεται ο φοιτητής, όπως 1 για το πρώτο εξάμηνο, 2 για το δεύτερο, κ.λπ. Αυτή η μέθοδος παρέχει έναν απλό και αποτελεσματικό τρόπο για να παρακολουθείται ποιοι φοιτητές είναι εγγεγραμμένοι σε ποια εξάμηνα, συμβάλλοντας στην αποφυγή διπλών εγγραφών και διασφαλίζοντας την ακριβή τήρηση αρχείων.

4.2.5.1 Παράμετροι

Δέχεται μόνο μία παράμετρο, το δημόσιο κλειδί της γραμματείας.

4.2.5.2 Συνθήκες νομισματοκοπίας

Μοναδική προϋπόθεση για την κοπή ενός token εγγραφής είναι η υπογραφή της συναλλαγής από την γραμματεία.

4.2.6 Validity Token Minting Policy

Το Validity Token παίζει ρόλο στην ασφάλεια και την ακεραιότητα της εφαρμογής, διασφαλίζοντας ότι το σωστό συμβόλαιο πολιτικής UTXO αναγνωρίζεται μοναδικά. Το Validity Token είναι ένα σημαντικό στοιχείο, επειδή αποτρέπει τους κακόβουλους φορείς από το να δημιουργούν ψεύτικα UTXO στη διεύθυνση του συμβολαίου πολιτικής, τα οποία διαφορετικά θα μπορούσαν να θεωρηθούν λανθασμένα ως έγκυρα, οδηγώντας ενδεχομένως σε εσφαλμένη έκδοση πτυχίων. Εξασφαλίζοντας ότι το Validity Token είναι παρόν στις τιμές του έγκυρου συμβολαίου πολιτικής UTXO, το σύστημα μπορεί να επιβεβαιώσει αξιόπιστα τη γνησιότητα των δεδομένων πολιτικής.

4.2.6.1 Παράμετροι

Έχει μία μόνο παράμετρο, μια αναφορά UTXO, η οποία χρησιμοποιείται για την προστασία από διπλή κοπή, διασφαλίζοντας ότι κάθε γεγονός νομισματοκοπίας συνδέεται με μια μοναδική έξοδο συναλλαγής.

4.2.6.2 Συνθήκες νομισματοκοπίας

Οι συνθήκες κοπής του Validity Token επιβάλλουν ότι το όνομα του token είναι "Validity", η ποσότητα που κόβεται είναι ακριβώς 1 και η καθορισμένη αναφορά εξόδου στην παράμετρο καταναλώνεται κατά τη διάρκεια της συναλλαγής κοπής. Αυτές οι συνθήκες είναι ζωτικής σημασίας για τη διατήρηση της ακεραιότητας του συστήματος, καθώς εγγυώνται ότι θα υπάρχει πάντα μόνο ένα Validity Token.

4.2.7 Degree NFT Minting Policy

Το Degree Token αντιπροσωπεύει το πτυχίο NFT του πανεπιστημίου. Αυτό το κουπόνι αποτελεί απόδειξη της ολοκλήρωσης των ακαδημαϊκών σπουδών ενός φοιτητή, όπου ο βαθμός ενσωματώνεται ως μεταδεδομένα εντός της συναλλαγής κοπής,. Σε αντίθεση με άλλα κουπόνια του συστήματος, το Degree Token δεν αποστέλλεται στην εσωτερική διεύθυνση του φοιτητή που διαχειρίζεται το έξυπνο συμβόλαιο, αλλά απευθείας στον προσωπικό λογαριασμό του φοιτητή. Το Asset name του Degree NFT είναι ο αριθμός μητρώου του φοιτητή.

4.2.7.1 Παράμετροι

Το Degree Token δέχεται τρεις παραμέτρους που διασφαλίζουν τη νομιμότητα και την ακεραιότητα της διαδικασίας έκδοσης πτυχίου:

- Το PolicyID του token εγγραφής: Αυτή η παράμετρος χρησιμοποιείται για να επαληθεύσει ότι ο φοιτητής που λαμβάνει το πτυχίο είναι επίσημα εγγεγραμμένος στο πανεπιστήμιο.
- Το PolicyID του Token εγκυρότητας: Απαραίτητη για τη διασφάλιση ότι η συναλλαγή κοπής πτυχίου αναφέρεται στη σωστή έξοδο στην διεύθυνση του συμβολαίου πολιτικών. Αυτή η παράμετρος επιβεβαιώνει ότι το συμβόλαιο πολιτικών UTXO που περιέχει τις απαιτήσεις του πτυχίου περιέχει και το σύμβολο εγκυρότητας, επαληθεύοντας έτσι την αυθεντικότητα και την επικαιρότητα των κριτηρίων απόκτησης του πτυχίου.
- Δημόσιο κλειδί της γραμματείας: Η παράμετρος αυτή περιλαμβάνεται επειδή η γραμματεία είναι το μόνο άτομο που έχει την εξουσιοδότηση να κόβει πτυχία.

4.2.7.2 Συνθήκες νομισματοκοπίας

Οι απαραίτητες προϋποθέσεις για την κοπή ενός NFT πτυχίου είναι:

- Η συναλλαγή κοπής πρέπει να υπογράφεται από τον γραμματέα.
- Η συναλλαγή κοπής πρέπει να περιλαμβάνει ως εισόδους αναφοράς το UTXO του συμβολαίου πολιτικών, το οποίο περιέχει τον πλήρη κατάλογο των token βαθμού που πρέπει να διαθέτει ο φοιτητής για να διεκδικήσει το πτυχίο του, το UTXO που περιέχει το token εγγραφής του στο πανεπιστήμιο από την εσωτερική διεύθυνση του φοιτητή, και πρόσθετα UTXO για κάθε μάθημα που έχει περάσει, όπως αποδεικνύεται από την παρουσία των αντίστοιχων κουπονιών βαθμού στην εκάστοτε έξοδο.
- Το UTXO του συμβολαίου πολιτικών που αναφέρεται πρέπει να περιέχει ως τιμή το token εγκυρότητας, ώστε να διασφαλίζεται η αυθεντικότητά του.
- Το asset name του Degree NFT που κόβεται πρέπει να ταιριάζει με το όνομα του κουπονιού εγγραφής που βρίσκεται στο UTXO του φοιτητή. Αυτή η συνθήκη διασφαλίζει ότι το πτυχίο αποδίδεται σωστά στον εγγεγραμμένο φοιτητή, επιβεβαιώνοντας την ταυτότητα και την κατάσταση εγγραφής του.
- Για όλα τα κουπόνια βαθμού που απαριθμούνται στον πίνακα δεδομένων του συμβολαίου πολιτικών, πρέπει να υπάρχει μια αντίστοιχη είσοδος αναφοράς που να δείχνει ότι ο φοιτητής κατέχει το συγκεκριμένο κουπόνι σε ποσότητα πέντε ή περισσότερων. Η απαίτηση αυτή επαληθεύει ότι ο φοιτητής έχει περάσει επιτυχώς όλα τα απαιτούμενα μαθήματα με ικανοποιητικούς βαθμούς.

4.2.7.3 Έκδοση Πτυχίου ως NFT

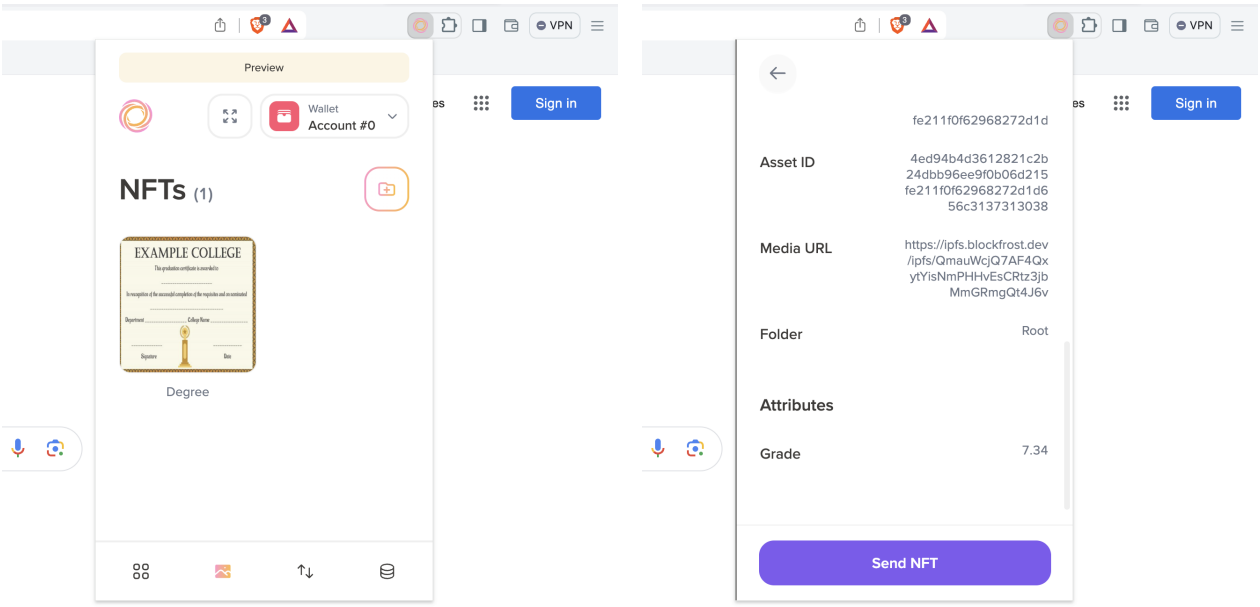
Για την αναπαράσταση του πτυχίου ως NFT στο Cardano, αξιοποιούμε τη δυνατότητα προσάρτησης μεταδεδομένων στις συναλλαγές νομισματοκοπίας. Σύμφωνα με το Cardano Improvement Proposal 26 (CIP-26), τα μεταδεδομένα ενός NFT πρέπει να ακολουθούν ένα συγκεκριμένο πρότυπο, που έχει ως εξής:

```
{
  "721": {
    "<policy_id>": {
      "<asset_name>": {
        "name": <string>,
        "image": <uri | array>,
        "mediaType": image/<mime_sub_type>,
        "description": <string | array>,
```

```
        "files": [{
          "name": <string>,
          "mediaType": <mime_type>,
          "src": <uri | array>,
          <other_properties>
        }],
        <other properties>
      }
    },
    "version": <version_id>
  }
}
```

Στην υλοποίησή μας, εκτός από το όνομα και την περιγραφή του token, περιλαμβάνουμε μια εικόνα, η οποία προς το παρόν είναι η ίδια για όλα τα πτυχία. Επιπλέον, ενσωματώνουμε τον βαθμό του φοιτητή στα μεταδεδομένα. Τα μεταδεδομένα συνδέονται με τη συναλλαγή χρησιμοποιώντας τη βιβλιοθήκη Lucid. Παρακάτω παρατίθεται ένα απλοποιημένο παράδειγμα νομισματοκοπίας που περιλαμβάνει αυτά τα μεταδεδομένα και μεταφέρει το token στον φοιτητή:

```
async function mintDegreeNFT(name, degreeGrade) {
    const unit = degreePolicyId + fromText(name);

    const metadata = {
        [degreePolicyId]: {
            [name]: {
                image: "ipfs://QmauWcjQ7AF4QxytYisNmPHHvEsCRtz3jbMmGRmgQt4J6v",
                name: "Degree",
                grade: degreeGrade,
                description: "University Degree NFT"
            }
        },
        version: "1.0"
    };

    const tx = await lucid
        .newTx()
        .mintAssets({ [unit]: 1n })
        .validTo(Date.now() + 100000)
        .attachMintingPolicy(mintingPolicy)
        .payToAddress(addself, { [unit]: 1n })
        .attachMetadata(721, metadata)
        .complete();

    const signedTx = await tx.sign().complete();
    const txHash = await signedTx.submit();

    return txHash;
}

await mintDegreeNFT("el17108", "7.34");
```



(a) Πτυχίο NFT

(b) Μεταδεδομένα ενός NFT Πτυχίου

Figure 4.2.1: Το πτυχίο όπως εμφανίζεται στο πορτοφόλι Lace

4.3 Backend

Το backend έχει αναπτυχθεί χρησιμοποιώντας το framework Express για τη διαχείριση των web requests, καθώς και τη βιβλιοθήκη lucid-cardano για την υλοποίηση των λειτουργιών που σχετίζονται με το δίκτυο Cardano, όπως αναζήτηση και ανάγνωση UTXOs, αποστολή συναλλαγών και άλλα. Ακολουθεί μία λεπτομερής περιγραφή των υλοποιημένων endpoints.

4.3.1 Προδιαγραφές API φοιτητών

4.3.1.1 Προβολή βαθμών

Endpoint /student/dashboard

Μέθοδος GET

Σκοπός Προβολή των βαθμών του φοιτητή

Λειτουργία Αρχικά ανακτά όλα τα UTXOs από την διεύθυνση του φοιτητή χρησιμοποιώντας τη βιβλιοθήκη lucid-cardano, μαζί με τον πίνακα που περιέχει τα έγκυρα tokens βαθμών από το datum του συμβολαίου πολιτικής. Στη συνέχεια, αντιστοιχίζει τις συμβολικές πολιτικές βαθμού από αυτά τα UTXOs με τα αντίστοιχα μαθήματα.

4.3.1.2 Προβολή διαθέσιμων μαθημάτων για εγγραφή

Endpoint /student/subjects

Μέθοδος GET

Σκοπός Προβολή των μαθημάτων στα οποία ο φοιτητής έχει δικαίωμα εγγραφής.

Λειτουργία Αυτό το endpoint λειτουργεί ανακτώντας πρώτα όλα τα μαθήματα από τη βάση δεδομένων που ανταποκρίνονται στα κριτήρια εγγραφής: πρέπει να διδάσκονται στο εξάμηνο που φοιτά ο φοιτητής ή σε προηγούμενα από αυτό και να ταιριάζουν με τον τρέχοντα τύπο εξαμήνου(μονό/ζυγό). Στη συνέχεια ανακτά τις έγκυρες πολιτικές από το datum του συμβολαίου πολιτικής και φιλτράρει τα θέματα που δεν απαιτούνται για το πτυχίο. Τέλος, ανακτά τους βαθμούς του χρήστη και αποκλείει τυχόν μαθήματα που ο φοιτητής έχει ήδη περάσει.

4.3.1.3 Δήλωση μαθημάτων

Endpoint /student/subjects

Μέθοδος POST

Σκοπός Εγγραφή στο τρέχον εξάμηνο και στα επιλεγμένα μαθήματα.

Λειτουργία Το endpoint επικυρώνει πρώτα ότι τα μαθήματα που έχει επιλέξει ο φοιτητής είναι έγκυρα για εγγραφή, διασφαλίζοντας τη συμμόρφωση με τους κανόνες δήλωσης μαθημάτων. Στη συνέχεια, αλληλεπιδρά με το έξυπνο συμβόλαιο κάθε επιλεγμένου μαθήματος χρησιμοποιώντας τον redeemer "Register" για να προσθέσει τον φοιτητή στους πίνακες εγγραφών. Στη συνέχεια, ένα κουπόνι εγγραφής για το τρέχον εξάμηνο στέλνεται στη διεύθυνση του φοιτητή. Τέλος, καλεί το ProofSpace API για να εκδώσει ένα πιστοποιητικό εγγραφής εξαμήνου και ένα πιστοποιητικό εγγραφής για κάθε επιλεγμένο μάθημα.

4.3.1.4 Προβολή στοιχείων φοιτητή

Endpoint /student/info

Μέθοδος GET

Σκοπός Επιστροφή των στοιχείων του φοιτητή.

Λειτουργία Ανακτά τον αριθμό μητρώου, το όνομα, το επώνυμο, το email, το ProofSpace ID και το έτος εγγραφής του φοιτητή, όπως εμφανίζονται στη βάση δεδομένων.

4.3.1.5 Ενημέρωση στοιχείων φοιτητή

Endpoint /student/info

Μέθοδος POST

Σκοπός Ενημέρωση των στοιχείων του φοιτητή.

Λειτουργία Επιτρέπει την ενημέρωση του email και/ή του κωδικού του φοιτητή.

4.3.1.6 Παραλαβή πτυχίου

Endpoint POST /student/mint

Μέθοδος POST

Σκοπός Δημιουργία και παραλαβή του πτυχίου NFT.

Λειτουργία Ξεκινά με την ανάκτηση των UTXO του φοιτητή μαζί με τα έγκυρα tokens βαθμών από το συμβόλαιο έγκυρων πολιτικών για να επαληθεύσει ότι ο φοιτητής έχει ολοκληρώσει τις σπουδές του. Υπολογίζει τον βαθμό πτυχίου ως τον μέσο όρο των βαθμών που αντιπροσωπεύουν τα UTXOs. Στη συνέχεια, το backend δημιουργεί τη συναλλαγή που στέλνει το NFT στον φοιτητή, χρησιμοποιώντας όλα τα απαραίτητα UTXOs του φοιτητή ως εισόδους αναφοράς. Τέλος, καλεί το API του ProofSpace για να εκδώσει το πιστοποιητικό πτυχίου.

4.3.2 Προδιαγραφές API καθηγητών

4.3.2.1 Προβολή διδασκόμενων μαθημάτων καθηγητή

Endpoint /teacher/subjects

Μέθοδος GET

Σκοπός Προβολή των μαθημάτων που διδάσκει και βαθμολογεί ο συγκεκριμένος καθηγητής.

Λειτουργία Επιστρέφει τον κατάλογο των μαθημάτων που έχουν ανατεθεί στον καθηγητή από τη βάση δεδομένων.

4.3.2.2 Προβολή εγγεγραμμένων φοιτητών σε μάθημα

Endpoint /teacher/subject/:subjectId/addGrade

Μέθοδος GET

Σκοπός Προβολή των φοιτητών που έχουν εγγραφεί στο εν λόγω μάθημα.

Λειτουργία Το endpoint ανακτά αρχικά τη διεύθυνση του έξυπνου συμβολαίου για το μάθημα που καθορίζεται από το 'subjectId' από τη βάση δεδομένων. Στη συνέχεια, αποκτά πρόσβαση στον πίνακα εγγραφών από το datum του UTXO σε αυτή τη διεύθυνση, ο οποίος περιέχει μόνο τους αριθμούς μητρώου των φοιτητών. Για να παρέχει μια ολοκληρωμένη εικόνα, αντλούνται επίσης από τη βάση δεδομένων τα ονόματα και επώνυμα που σχετίζονται με αυτά τα ID. Το αποτέλεσμα είναι ένας ολοκληρωμένος κατάλογος που περιέχει αριθμούς μητρώου, ονόματα και επώνυμα.

4.3.2.3 Βαθμολόγηση

Endpoint /teacher/subject/:subjectId/addGrade

Μέθοδος POST

Σκοπός Έκδοση των αποκεντρομένων πιστοποιητικών του βαθμού μέσω του ProofSpace.

Λειτουργία Την κλήση Cardano για την έκδοση βαθμών μέσω του έξυπνου συμβολαίου του μαθήματος διαχειρίζεται απευθείας το frontend. Αφού υποβληθούν οι βαθμοί στο blockchain, αυτό το τελικό σημείο διασφαλίζει ότι κάθε φοιτητής λαμβάνει ένα ψηφιακό πιστοποιητικό βαθμού από το ProofSpace.

4.3.2.4 Προβολή πληροφοριών καθηγητή

Endpoint /teacher/info

Μέθοδος GET

Σκοπός Επιστροφή των στοιχείων του καθηγητή.

Λειτουργία Ανακτά το ID, το όνομα, το επώνυμο, το email και τη διεύθυνση Cardano του εκπαιδευτικού, όπως εμφανίζονται στη βάση δεδομένων.

4.3.2.5 Ενημέρωση πληροφοριών καθηγητή

Endpoint /teacher/info

Μέθοδος POST

Σκοπός Ενημέρωση των στοιχείων του καθηγητή.

Λειτουργία Ενημερώνει το email ή/και τον κωδικού πρόσβασης του εκπαιδευτικού στη βάση δεδομένων.

4.3.3 Προδιαγραφές API γραμματείας

4.3.3.1 Προσθήκη νέου φοιτητή

Endpoint /admin/addStudent

Μέθοδος POST

Σκοπός Προσθήκη ενός νέου φοιτητή στο σύστημα.

Λειτουργία Αυτό το endpoint προσθέτει έναν νέο φοιτητή με προσωρινό κωδικό πρόσβασης στη βάση δεδομένων.

4.3.3.2 Προσθήκη νέου καθηγητή

Endpoint /admin/addTeacher

Μέθοδος POST

Σκοπός Προσθήκη νέου καθηγητή.

Λειτουργία Προσθέτει έναν νέο καθηγητή στη βάση δεδομένων με βάση τα υποβληθέντα δεδομένα.

4.3.3.3 Προσθήκη νέου μαθήματος

Endpoint /admin/addSubject

Μέθοδος POST

Σκοπός Προσθήκη ενός νέου μαθήματος στο πρόγραμμα σπουδών.

Λειτουργία Μετά τη λήψη νέων δεδομένων για το μάθημα,προσθέτει τα στοιχεία του στη βάση δεδομένων και δημιουργεί ένα νέο token βαθμού για αυτό. Στη συνέχεια, τα token βαθμού που κόπηκαν κλειδώνονται σε ένα νέο έξυπνο συμβόλαιο, κλειδώνοντας τη ρευστότητα και συσχετίζοντάς την απευθείας με το μάθημα.

4.3.3.4 Ενημέρωση υπάρχοντος μαθήματος

Endpoint /admin/updateSubject

Μέθοδος POST

Σκοπός Ενημέρωση των στοιχείων ενός μαθήματος.

Λειτουργία Μπορεί να ενημερώσει διάφορες λεπτομέρειες ενός μαθήματος, όπως ο διδάσκων καθηγητής ή το εξάμηνο. Οι ενημερώσεις πραγματοποιούνται είτε στην βάση δεδομένων είτε μέσω κλήσεων στο έξυπνο συμβόλαιο του μαθήματος, ανάλογα με τις παραμέτρους που μεταβάλλονται.

4.3.3.5 Αλλαγή προγράμματος σπουδών

Endpoint /admin/changeSubjects

Μέθοδος POST

Σκοπός Ενημέρωση των μαθημάτων που απαιτούνται για απόκτηση πτυχίου.

Λειτουργία Αυτό το endpoint αλληλεπιδρά με το έξυπνο συμβόλαιο που περιέχει τα έγκυρα tokens για την ενημέρωση της συστοιχίας των πολιτικών, και συγκεκριμένα των tokens βαθμού που απαιτούνται για την ολοκλήρωση του πτυχίου.

4.3.3.6 Εκκίνηση περιόδου εγγραφών

Endpoint /admin/startRegistration

Μέθοδος POST

Σκοπός Εκκίνηση της περιόδου εγγραφών για το τρέχον εξάμηνο.

Λειτουργία Ανακτά πληροφορίες για τα μαθήματα του τρέχοντος εξαμήνου, υπολογίζει τις διευθύνσεις των συμβολαίων τους και καλεί ξεχωριστά κάθε ένα χρησιμοποιώντας τον redeemer 'Update' για να ορίσει το νέο "end_date".

4.3.4 Προδιαγραφές API γενικού σκοπού

4.3.4.1 Σύνδεση

Endpoint /login

Μέθοδος POST

Σκοπός Σύνδεση

Λειτουργία Επιτρέπει στους χρήστες να συνδεθούν στο σύστημα χρησιμοποιώντας τα διαπιστευτήρια που είναι αποθηκευμένα στη βάση δεδομένων. Ταυτοποιεί τους χρήστες με βάση το όνομα χρήστη, τον κωδικό πρόσβασης και τον ρόλο που έχουν δηλώσει.

4.3.4.2 Αποσύνδεση

Endpoint /logout

Μέθοδος GET

Σκοπός Αποσύνδεση

Λειτουργία Χειρίζεται την αποσύνδεση του χρήστη, εξασφαλίζοντας την ανάκληση των προνομίων πρόσβασης.

4.3.4.3 Προβολή πληροφοριών υποχρεωτικών μαθημάτων

Endpoint /policies

Μέθοδος GET

Σκοπός Επιστροφή πληροφοριών σχετικά με τα υποχρεωτικά μαθήματα μαζί με τα στοιχεία των tokens βαθμών.

Λειτουργία Αυτό το τελικό σημείο ανακτά τον κατάλογο των υποχρεωτικών μαθημάτων και των tokens βαθμών. Στη συνέχεια, αντλεί πρόσθετες πληροφορίες από τη βάση δεδομένων, συντάσσοντας ένα ολοκληρωμένο σύνολο δεδομένων.

4.3.4.4 Πληροφορίες μαθήματος

Endpoint /subjects/:id

Μέθοδος GET

Σκοπός Επιστροφή λεπτομερειών για το επιλεγμένο μάθημα.

Λειτουργία Το endpoint ανακτά πρώτα τα δεδομένα σχετικά με το μάθημα από τη βάση δεδομένων. Στη συνέχεια υπολογίζει το token βαθμού και τη διεύθυνση του έξυπνου συμβολαίου. Τέλος, αντλεί πρόσθετες πληροφορίες, όπως η περίοδος λήξης της εγγραφής, η διεύθυνση του καθηγητή και άλλα σχετικά δεδομένα από τα δεδομένα του συμβολαίου του μαθήματος, παρέχοντας μια λεπτομερή εικόνα.

4.4 ProofSpace

Το ProofSpace είναι μια πλατφόρμα λογισμικού ως υπηρεσία (SaaS) που ειδικεύεται στο χειρισμό αποκεντρωμένων αναγνωριστικών και πιστοποιητικών στο Cardano Blockchain. Η υπηρεσία αυτή απλοποιεί τη διαδικασία διαχείρισης αποκεντρωμένων πιστοποιητικών με την αφαίρεση των υποκείμενων αλληλεπιδράσεων σε μια λύση που δεν απαιτεί συγγραφή κώδικα, επιτρέποντας τη δημιουργία λειτουργιών που βασίζονται σε πιστοποιητικά εντός εφαρμογών. Το ProofSpace παρέχει ένα API που επιτρέπει σε εφαρμογές όπως η δική μας να εκδίδουν πιστοποιητικά.

Με την ενσωμάτωση του ProofSpace, η εφαρμογή μας επιτρέπει στους φοιτητές να διεκδικούν και να επαληθεύουν δημόσια τα ακαδημαϊκά τους πιστοποιητικά μέσω της εφαρμογής ProofSpace. Αυτή η λειτουργία είναι σημαντική για την επαλήθευση των εγγραφών των φοιτητών σε διάφορα ακαδημαϊκά περιβάλλοντα, όπως η παρακολούθηση μαθημάτων και η πρόσβαση σε εργαστήρια, όπου μπορεί να ισχύουν προϋποθέσεις εγγραφής. Επιπλέον, το ProofSpace βελτιώνει την εμπειρία των χρηστών στέλνοντας ειδοποιήσεις push απευθείας στα κινητά των φοιτητών κάθε φορά που καταγράφονται νέοι βαθμοί ή εκδίδονται πιστοποιητικά, παρέχοντας ενημερώσεις σε πραγματικό χρόνο χωρίς να χρειάζεται οι φοιτητές να ελέγχουν επανειλημμένα την διαδικτυακή πλατφόρμα.

Στην εφαρμογή μας, το ProofSpace χρησιμοποιείται για τη διαχείριση πέντε διαφορετικών τύπων ακαδημαϊκών πιστοποιητικών(Credentials), μαζί με έξι διαφορετικούς τύπους αλληλεπιδράσεων(Interactions). Κάθε πιστοποιητικό εξυπηρετεί έναν συγκεκριμένο σκοπό:

- Εγγραφή στο πανεπιστήμιο(University Registration): Εκδίδεται μία φορά σε κάθε φοιτητή και σηματοδοτεί την επίσημη εγγραφή του στο πανεπιστήμιο. Περιλαμβάνει δεδομένα όπως το όνομα του πανεπιστημίου και τον αριθμό μητρώου του φοιτητή.
- Εγγραφή σε εξάμηνο(Semester Registration): Οι φοιτητές λαμβάνουν αυτό το πιστοποιητικό για κάθε εξάμηνο που εγγράφονται, σηματοδοτώντας την εγγραφή τους στο συγκεκριμένο εξάμηνο.
- Εγγραφή σε μάθημα(Subject Registration): Χορηγείται σε κάθε εξάμηνο που ο φοιτητής εγγράφεται σε ένα μάθημα και περιέχει στοιχεία για το μάθημα και το έγκυρο εξάμηνο.
- Πιστοποιητικό βαθμολογίας(Grade Credential): Χορηγείται στους φοιτητές μετά τη λήψη βαθμού σε ένα μάθημα, αναφέροντας λεπτομερώς το όνομα του μαθήματος και τον βαθμό του φοιτητή.
- Πιστοποιητικό πτυχίου(Degree Credential): Η τελική πιστοποίηση, που σηματοδοτεί την ολοκλήρωση του ακαδημαϊκού προγράμματος ενός φοιτητή και περιλαμβάνει το βαθμό πτυχίου μαζί με άλλα βασικά στοιχεία.

Οι αλληλεπιδράσεις εντός της εφαρμογής κατηγοριοποιούνται σε δύο κύριους τύπους:

- Αλληλεπίδραση με κωδικό QR: Αυτή η αλληλεπίδραση περιλαμβάνει έναν κωδικό QR που εμφανίζεται στον φοιτητή κατά την αρχική του είσοδο στην ιστοσελίδα. Σκοπός αυτού του κωδικού QR είναι να συνδέσει το ProofSpace ID του φοιτητή με την υπηρεσία μας.
- Απλές αλληλεπιδράσεις «Webhook»: Αυτές περιλαμβάνουν ενεργοποιήσεις από κλήσεις API σε συγκεκριμένα endpoints που έχουν σχεδιαστεί για την έκδοση πιστοποιητικών απευθείας στους χρήστες. Αυτός ο μηχανισμός διασφαλίζει ότι τα πιστοποιητικά εκδίδονται άμεσα και με ακρίβεια ως απάντηση σε διάφορες ακαδημαϊκές δραστηριότητες, όπως η εγγραφή σε μαθήματα ή η επίτευξη βαθμών.

4.4.1 Πιστοποιητικά

Σε αυτή την ενότητα, θα εξερευνήσουμε πρώτα πώς να δημιουργήσουμε έναν νέο πιστοποιητικό στο ProofSpace CRM, εξετάζοντας όλες τις διαθέσιμες επιλογές. Αργότερα, θα εμβαθύνουμε στους συγκεκριμένους ορισμούς κάθε πιστοποιητικού που χρησιμοποιείται στην εφαρμογή μας.

4.4.1.1 Προσθήκη νέου πιστοποιητικού

Για να δημιουργήσουμε έναν νέο πιστοποιητικό στο ProofSpace CRM, πρέπει πρώτα να δημιουργήσουμε ένα σχήμα και στη συνέχεια να δημιουργήσουμε τον ορισμό με βάση αυτό. Ξεκινήστε μεταβαίνοντας στη σελίδα 'Schemas'. Μόλις φτάσετε εκεί, κάντε κλικ στο κουμπί 'Add Schema' που βρίσκεται στην επάνω δεξιά γωνία της διεπαφής. Αυτή η ενέργεια θα ανοίξει μια νέα σελίδα που θα σας ζητά να εισαγάγετε τις λεπτομέρειες του σχήματος. Εισάγετε τις απαιτούμενες πληροφορίες στο modal για να προχωρήσετε στη δημιουργία του νέου σχήματος πιστοποιητικών.

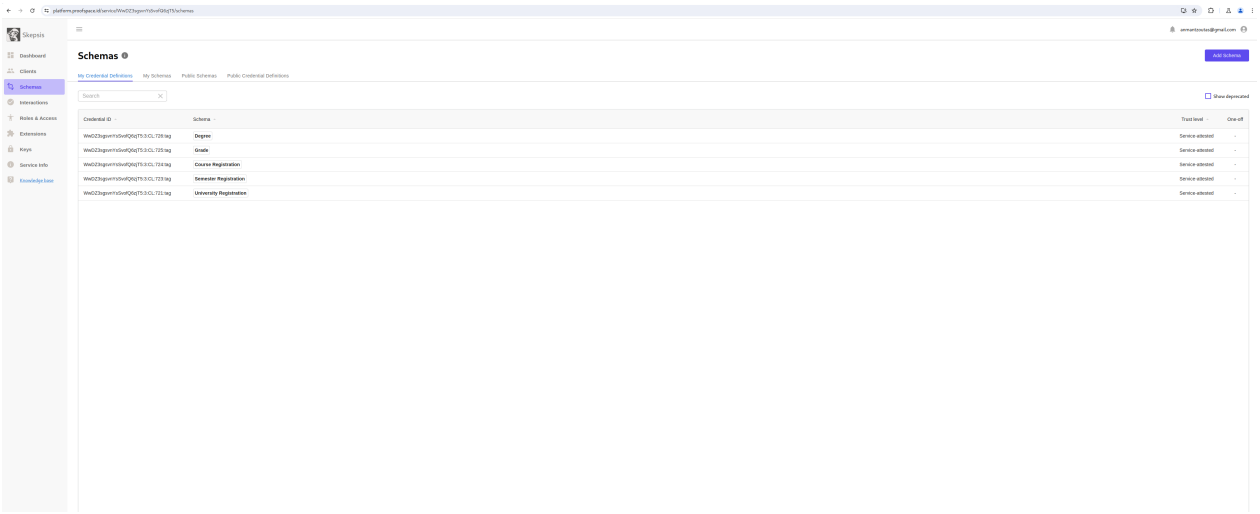


Figure 4.4.1: Σελίδα Σχημάτων Πιστοποιητικών

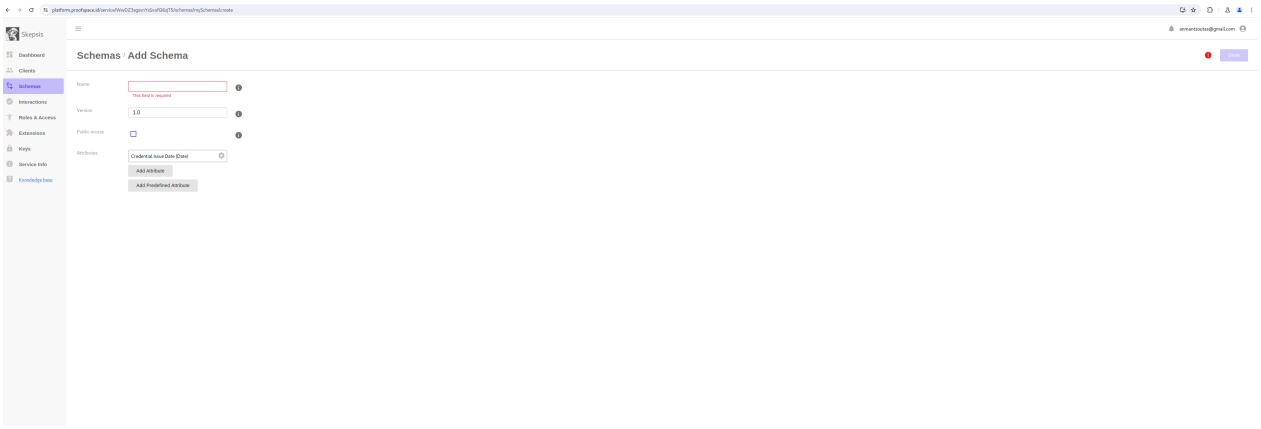


Figure 4.4.2: Προσθήκη νέου σχήματος

Για να συνεχίσετε τη διαδικασία, εισαγάγετε το όνομα του σχήματος και τυχόν επιθυμητά χαρακτηριστικά. Για να προσθέσετε ένα νέο χαρακτηριστικό(attribute), κάντε κλικ στο κουμπί «Add Attribute». Θα σας ζητηθεί να δώσετε ένα όνομα για το χαρακτηριστικό, να επιλέξετε τον αναμενόμενο τύπο (όπως κείμενο, αριθμός, ημερομηνία κ.λπ.) και μπορείτε επίσης να προσθέσετε μια προαιρετική περιγραφή για περαιτέρω διευκρινίσεις.

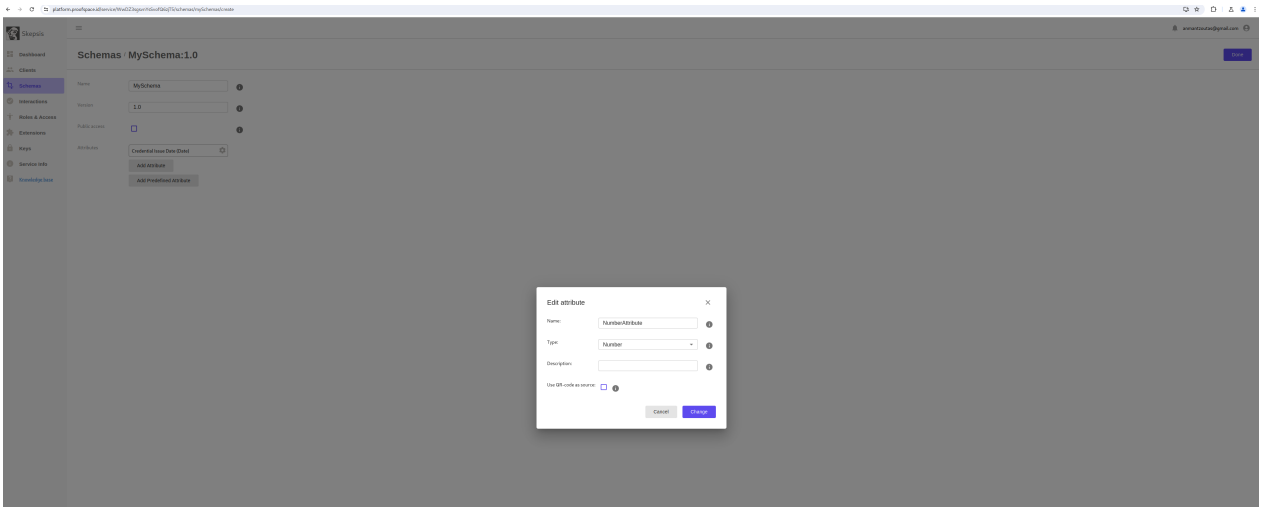


Figure 4.4.3: Προσθήκη χαρακτηριστικού

Μόλις προστεθούν όλα τα απαιτούμενα χαρακτηριστικά, κάντε κλικ στο κουμπί «Done» στην επάνω δεξιά γωνία, ακολουθούμενο από το «Save & Publish». Επιβεβαιώστε την ενέργεια για να οριστικοποιήσετε και να καταστήσετε διαθέσιμο προς χρήση το νέο σχήμα πιστοποιητικών σας. Μετά την αποθήκευση, το νεοδημιουργηθέν σχήμα θα εμφανιστεί στην κεντρική σελίδα του σχήματος. Για να προχωρήσετε στη δημιουργία του ορισμού, κάντε κλικ στο «Create» στη γραμμή του αντίστοιχου σχήματος.

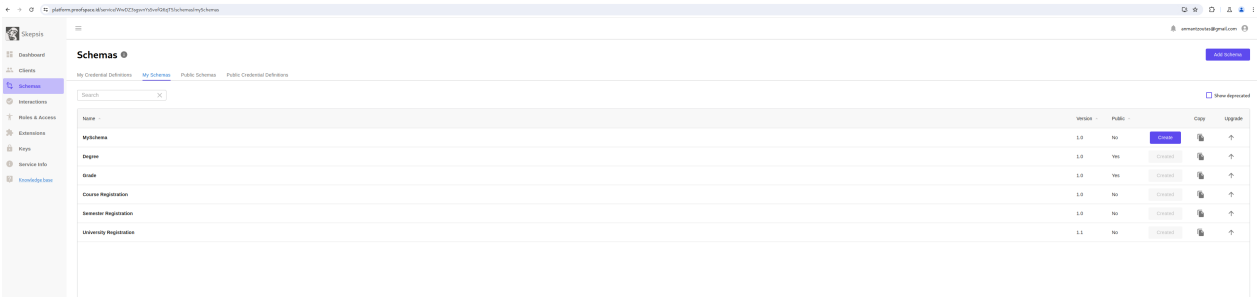


Figure 4.4.4: Το καινούριο Σχήμα

Μόλις πλοηγηθείτε στην καθορισμένη σελίδα για τη δημιουργία του ορισμού, θα δείτε μία προειδοποίηση ότι η δημιουργία ενός ορισμού είναι μια ενέργεια μιας χρήσης- αφού οριστεί, δεν μπορεί να τροποποιηθεί. Η διεπαφή χωρίζεται σε δύο τμήματα: στο άνω μισό απαιτείται η εισαγωγή των ρυθμίσεων, ενώ στο κάτω μισό εμφανίζονται πληροφορίες για το σχετικό σχήμα.

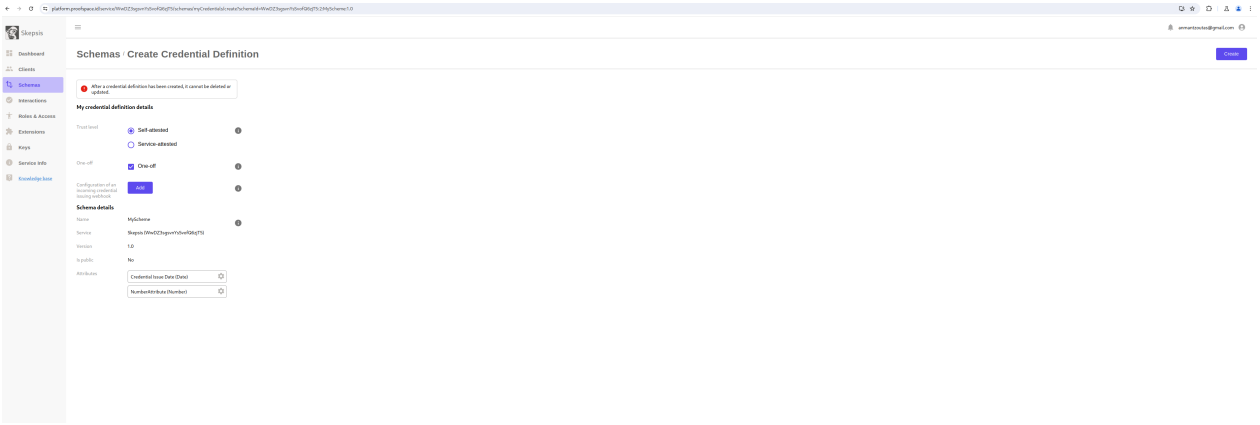


Figure 4.4.5: Δημιουργία ορισμού πιστοποιητικών

Κατ' αρχάς, πρέπει να αποφασίσετε μεταξύ «self-attested» και «service-attested» πιστοποιητικών. Τα «self-attested» (αυτοεπιβεβαιούμενα) επιτρέπουν στους χρήστες να εισάγουν οι ίδιοι χειροκίνητα τις τιμές των χαρακτηριστικών του πιστοποιητικού. Από την άλλη πλευρά, τα «service-attested» διαχειρίζονται εξ ολοκλήρου από την υπηρεσία, ενώ οι χρήστες απλώς λαμβάνουν τα ολοκληρωμένα πιστοποιητικά. Για τα αυτοεπιβεβαιούμενα πιστοποιητικά, έχετε επίσης τη δυνατότητα να ενεργοποιήσετε την επιλογή «one-off». Αυτή η επιλογή απαιτεί από τον χρήστη να συμπληρώνει εκ νέου τα χαρακτηριστικά κάθε φορά που συμμετέχει σε μια αλληλεπίδραση που εκδίδει αυτό το πιστοποιητικό.

Επιπλέον, μπορείτε να ρυθμίσετε ένα εξερχόμενο webhook, καθορίζοντας μια μέθοδο ελέγχου ταυτότητας και, προαιρετικά, ένα πρόθεμα IP για την αίτηση. Αυτή η δυνατότητα επιτρέπει σε εξωτερικές εφαρμογές να εκδίδουν αυτό το συγκεκριμένο πιστοποιητικό. Για να ενεργοποιηθεί αυτό το webhook, πρέπει να υπάρχει κατάλληλη αλληλεπίδραση.

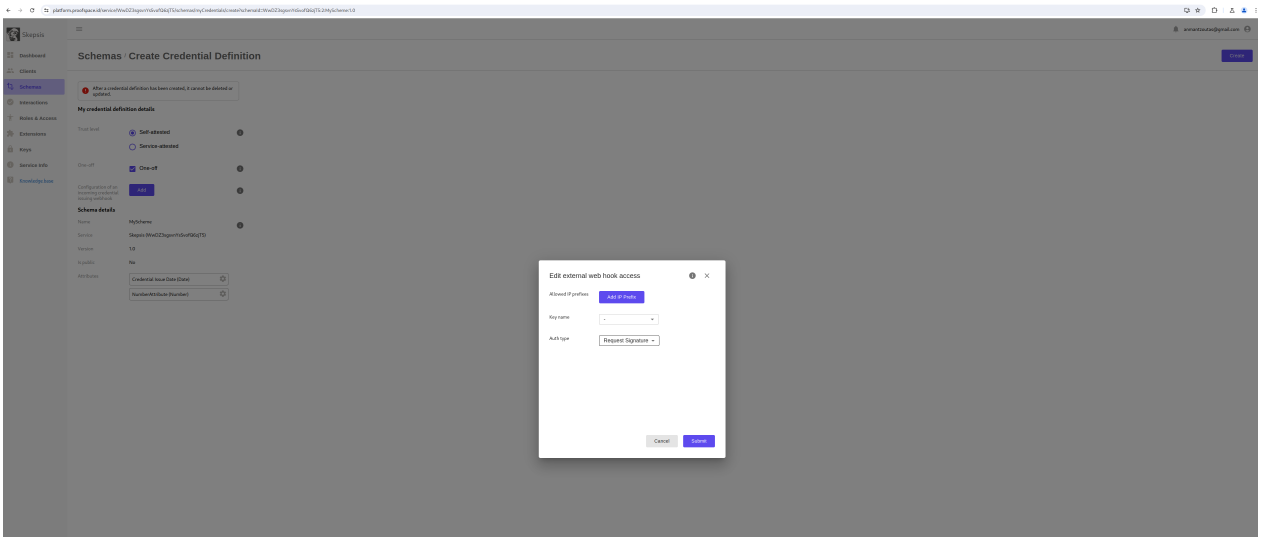


Figure 4.4.6: Ρύθμιση Webhook

Αφού ρυθμίσετε όλες τις επιλογές σύμφωνα με τις ανάγκες σας, ολοκληρώστε τη διαδικασία κάνοντας κλικ στην επιλογή «Δημιουργία» που βρίσκεται στην επάνω δεξιά γωνία της σελίδας.

4.4.1.2 Πιστοποιητικά της Εφαρμογής

Πριν εμβαθύνουμε στους συγκεκριμένους ορισμούς πιστοποιητικών που χρησιμοποιούνται στην εφαρμογή μας, είναι σημαντικό να σημειώσουμε ότι, ενώ τα σχήματα διαφέρουν, οι διαμορφώσεις για όλους τους ορισμούς είναι ίδιες και για τα πέντε πιστοποιητικά. Όλα έχουν οριστεί ως «service-attested», αντικατοπτρίζοντας τη φύση των δεδομένων που διαχειρίζονται -όπως οι βαθμοί και οι εγγραφές- τα οποία απαιτούν υψηλό βαθμό ελέγχου για την αποφυγή οποιασδήποτε χειραγώγησης από τους φοιτητές. Επιπλέον, η διαμόρφωση για το εισερχόμενο webhook έκδοσης πιστοποιητικών είναι τυποποιημένη- απαιτεί όλα τα αιτήματα να πιστοποιούνται με ένα συγκεκριμένο κλειδί RSA, το οποίο αποθηκεύεται με ασφάλεια στον backend της εφαρμογής.

University Registration Η πιστοποίηση εγγραφής στο Πανεπιστήμιο σηματοδοτεί την έναρξη της ακαδημαϊκής διαδρομής ενός φοιτητή στο πανεπιστήμιο. Κάθε φοιτητής λαμβάνει αυτό το πιστοποιητικό μόνο μία φορά, το οποίο ενεργοποιείται από την ενέργειά του να σαρώσει τον κωδικό QR που εμφανίζεται στη διαδικτυακή διεπαφή της εφαρμογής. Αυτό το πιστοποιητικό χρησιμεύει ως επίσημο αρχείο εγγραφής και περιλαμβάνει δύο βασικά χαρακτηριστικά:

- School: Ένα χαρακτηριστικό τύπου κειμένου που καταγράφει το όνομα του πανεπιστημίου στο οποίο εγγράφεται ο φοιτητής.
- StudentID: Επίσης ένα χαρακτηριστικό τύπου κειμένου, το οποίο καταγράφει τον αριθμό μητρώου του φοιτητή.

Εκτός από αυτά τα συγκεκριμένα χαρακτηριστικά, κάθε πιστοποιητικό εγγραφής στο Πανεπιστήμιο περιλαμβάνει επίσης ένα τυπικό χαρακτηριστικό για την ημερομηνία έκδοσης, το οποίο καταγράφει την ακριβή στιγμή που δημιουργήθηκε και εκδόθηκε το πιστοποιητικό. Αυτό το χαρακτηριστικό είναι κοινό για όλα τα διαφορετικά πιστοποιητικά.

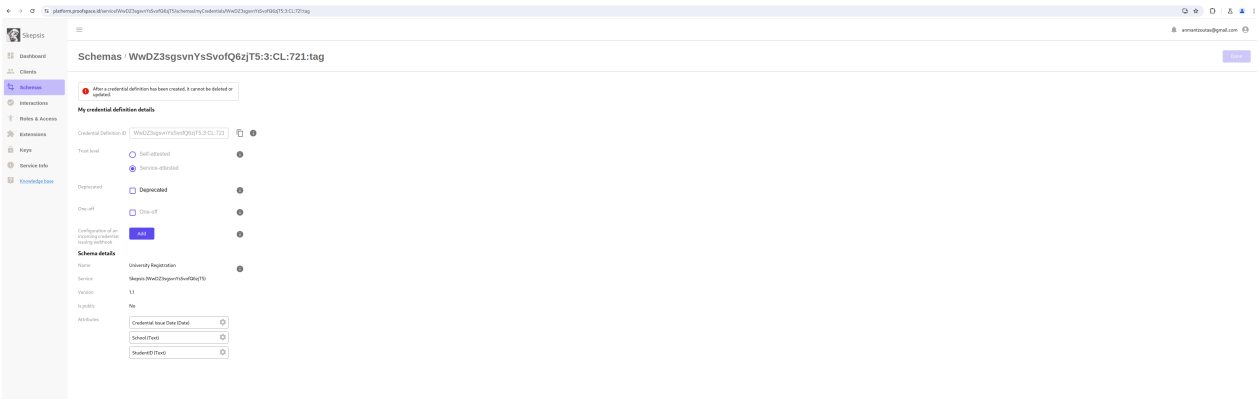


Figure 4.4.7: Ορισμός Πιστοποιητικού University Registration

Semester Registration Η εξαμηνιαία εγγραφή έχει σχεδιαστεί για να παρακολουθεί και να επικυρώνει τη συνεχή ακαδημαϊκή πρόοδο ενός φοιτητή. Κάθε φοιτητής λαμβάνει ξεχωριστή πιστοποίηση για κάθε εξάμηνο στο οποίο εγγράφεται καθ' όλη τη διάρκεια της πανεπιστημιακής του θητείας.

Εκτός από την τυπική ιδιότητα της ημερομηνίας έκδοσης που υπάρχει σε κάθε πιστοποιητικό, το πιστοποιητικό εγγραφής εξαμήνου περιλαμβάνει μια ειδική ιδιότητα:

- Semester: Αυτό το χαρακτηριστικό είναι τύπου αριθμού και δηλώνει την ακολουθία του εξαμήνου για το οποίο έχει εγγραφεί ο φοιτητής. Για παράδειγμα, είναι «1» για το πρώτο εξάμηνο, «2» για το δεύτερο, κ.ο.κ.

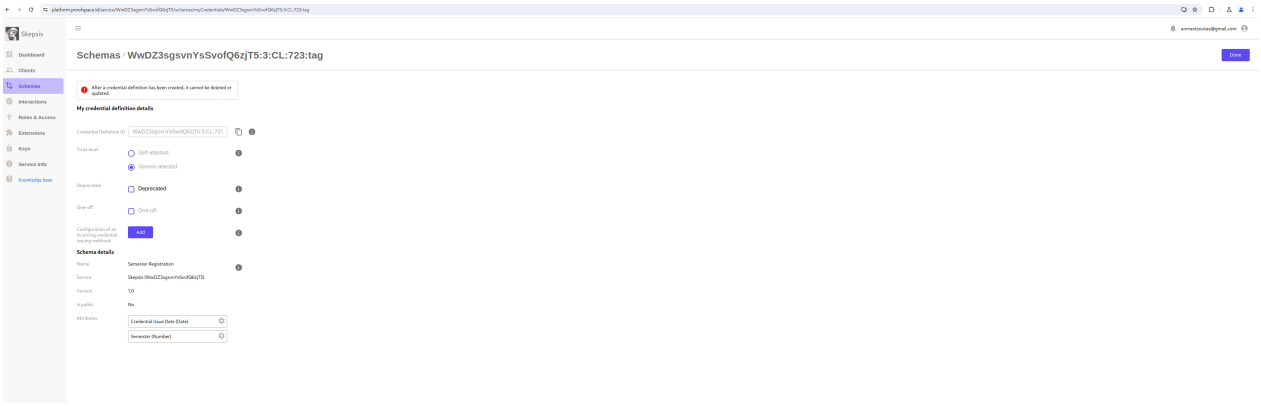


Figure 4.4.8: Ορισμός Πιστοποιητικού Semester Registration

Course Registration Η πιστοποίηση εγγραφής μαθημάτων χορηγείται στους φοιτητές για κάθε μάθημα που εγγράφονται για να παρακολουθήσουν μαθήματα και να δώσουν εξετάσεις. Αυτό το πιστοποιητικό είναι ζωτικής σημασίας για τη διαχείριση των εγγραφών μαθημάτων και τη διασφάλιση ακριβών ακαδημαϊκών αρχείων. Περιλαμβάνει δύο συγκεκριμένα χαρακτηριστικά:

- Subject: Μια ιδιότητα κειμένου που καταγράφει το όνομα του μαθήματος στο οποίο έχει εγγραφεί ο φοιτητής.
- Semester: Μια αριθμητική ιδιότητα που δηλώνει το εξάμηνο κατά το οποίο ο φοιτητής εγγράφηκε στο μάθημα. Αυτό βοηθά στην παρακολούθηση περιπτώσεων όπου ένας φοιτητής μπορεί να εγγραφεί στο ίδιο μάθημα πολλές φορές, σε διαφορετικά εξάμηνα.

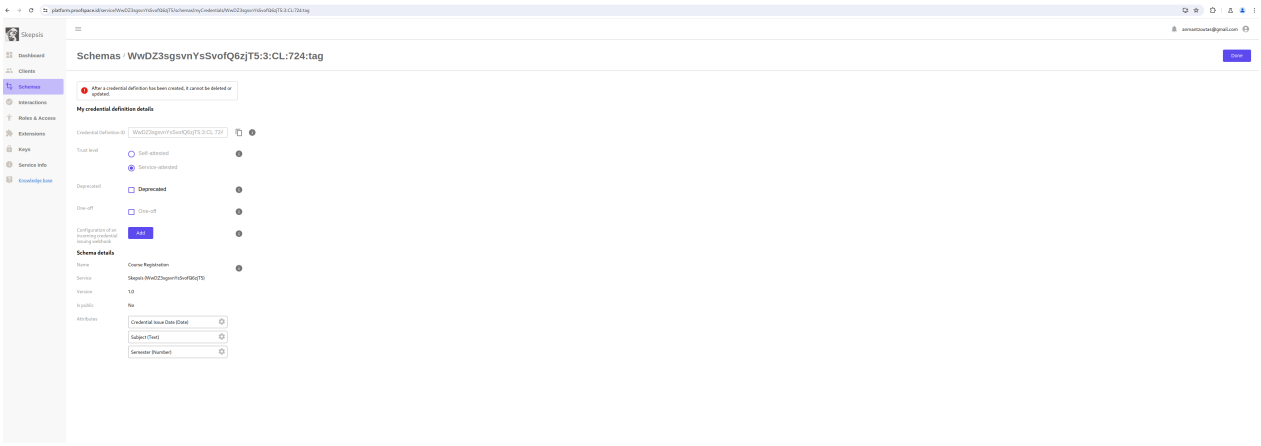


Figure 4.4.9: Ορισμός Πιστοποιητικού Course Registration

Grade Credential Το Πιστοποιητικό Βαθμολογίας είναι ζωτικής σημασίας για την τεκμηρίωση των ακαδημαϊκών επιδόσεων των φοιτητών σε συγκεκριμένα μαθήματα. Αυτό το πιστοποιητικό εκδίδεται στους φοιτητές μετά τη λήψη βαθμού για ένα μάθημα και περιλαμβάνει δύο βασικά πεδία:

- Subject: Ένα πεδίο κειμένου που καθορίζει το όνομα του μαθήματος για το οποίο αποδίδεται ο βαθμός.
- Grade: Ένα αριθμητικό πεδίο που καταγράφει το βαθμό του φοιτητή για το μάθημα, με αναμενόμενες τιμές που κυμαίνονται από 0 έως 10.

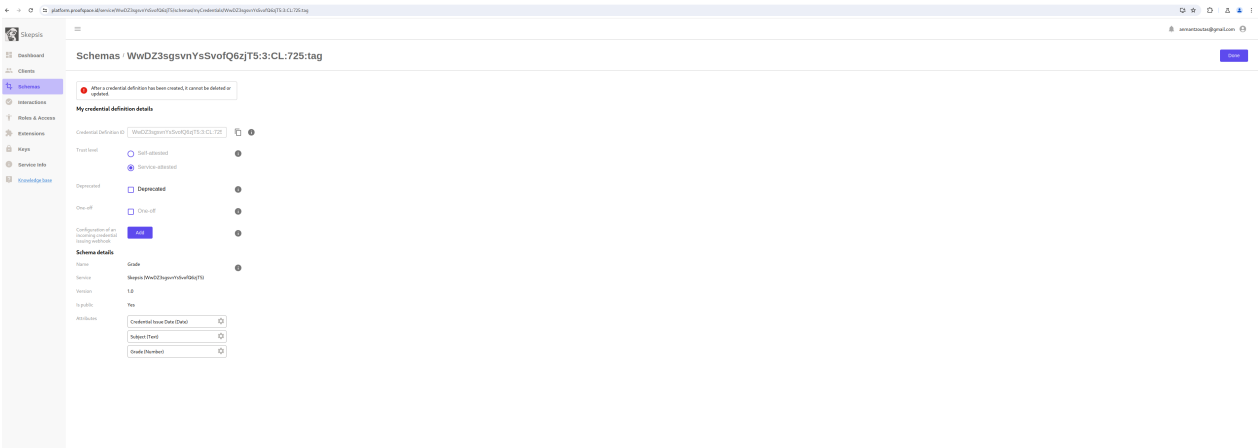


Figure 4.4.10: Ορισμός Πιστοποιητικού Grade

Degree Credential Το Πιστοποιητικό Πτυχίου χρησιμεύει ως το οριστικό αρχείο που σηματοδοτεί την ολοκλήρωση της ακαδημαϊκής διαδρομής ενός φοιτητή. Εκδίδεται μόλις ο φοιτητής ολοκληρώσει τις σπουδές του και είναι έτοιμος να διεκδικήσει το πτυχίο του. Το εν λόγω πιστοποιητικό περικλείει το αποκορύφωμα των ακαδημαϊκών προσπαθειών του φοιτητή και περιλαμβάνει δύο σημαντικά πεδία:

- School: Ένα πεδίο κειμένου που υποδεικνύει το όνομα του πανεπιστημίου από το οποίο αποφοιτά ο φοιτητής. Αυτό επιβεβαιώνει το ίδρυμα που απονέμει το πτυχίο.
- Degree: Ένα πεδίο κειμένου που καταγράφει τον τελικό αθροιστικό βαθμό του φοιτητή, εκφρασμένο ως δεκαδική τιμή στο εύρος 5 έως 10. Ο βαθμός αυτός αντικατοπτρίζει τη συνολική ακαδημαϊκή επίδοση του φοιτητή κατά τη διάρκεια των σπουδών του.

Αυτό το πιστοποιητικό αποτελεί σημαντικό ακαδημαϊκό ορόσημο, παρέχοντας μια επίσημη και επαληθεύσιμη καταγραφή του πτυχίου και της ακαδημαϊκής θέσης του φοιτητή κατά την αποφοίτησή του.

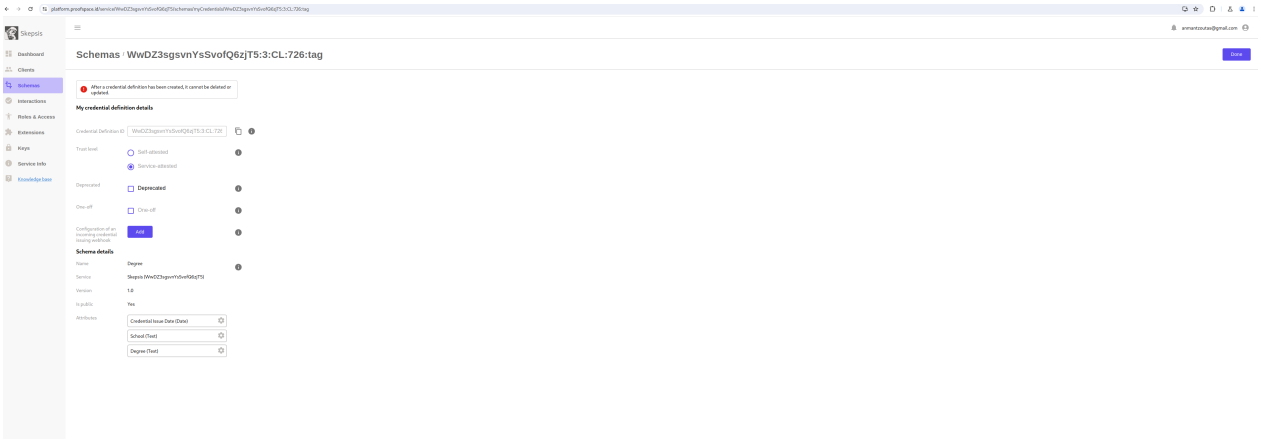


Figure 4.4.11: Ορισμός Πιστοποιητικού Degree

4.4.2 Αλληλεπιδράσεις

Εκτός από την προσθήκη των ορισμών των πιστοποιητικών, η διαμόρφωση του ProofSpace απαιτεί τον ορισμό συγκεκριμένων αλληλεπιδράσεων. Οι αλληλεπιδράσεις είναι ροές που καθορίζουν τις συνθήκες υπό τις οποίες μπορεί να εκδοθεί ένα πιστοποιητικό. Αυτές οι προκαθορισμένες ροές είναι απαραίτητες, καθώς διασφαλίζουν ότι τα πιστοποιητικά εκδίδονται μόνο όταν πληρούνται ορισμένα κριτήρια, διατηρώντας την ακεραιότητα και τη συνάφεια των δεδομένων εντός των πιστοποιητικών. Αυτό το βήμα είναι απαραίτητο για την αυτοματοποίηση της διαδικασίας έκδοσης πιστοποιητικών.

4.4.2.1 Προσθήκη νέας αλληλεπίδρασης

Για να δημιουργήσετε μια νέα αλληλεπίδραση, μεταβείτε στη σελίδα «Interactions» της πλατφόρμας. Εκεί, κάντε κλικ στο κουμπί «New Interaction» που βρίσκεται στην επάνω δεξιά γωνία της διεπαφής. Αυτή η ενέργεια θα ξεκινήσει τη διαδικασία δημιουργίας μιας νέας αλληλεπίδρασης, επιτρέποντάς σας να ορίσετε τις συγκεκριμένες συνθήκες και τα εναύσματα για την έκδοση πιστοποιητικών με βάση τις απαιτήσεις του συστήματός σας.

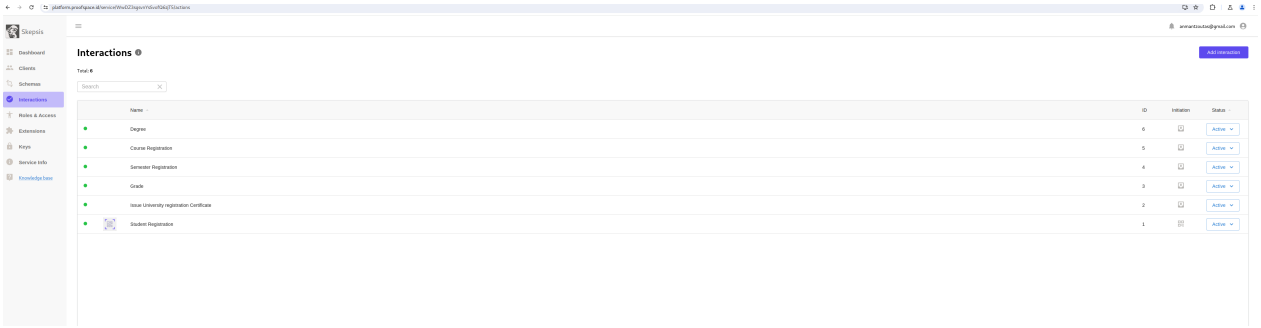


Figure 4.4.12: Σελίδα Αλληλεπιδράσεων

Μόλις κάνετε κλικ στο κουμπί, θα μεταφερθείτε στη σελίδα δημιουργίας αλληλεπίδρασης. Αυτή η σελίδα είναι οργανωμένη σε τέσσερις καρτέλες: Interaction Info(Πληροφορίες αλληλεπίδρασης), Details(Λεπτομέρειες), Instances(Περιπτώσεις) και Extensions(Επεκτάσεις). Για τη δημιουργία μιας βασικής αλληλεπίδρασης, θα χρειαστεί να εργαστείτε κυρίως με τις δύο πρώτες καρτέλες: Πληροφορίες αλληλεπίδρασης και Λεπτομέρειες.

Στην καρτέλα Πληροφορίες αλληλεπίδρασης, καλείστε να δώσετε κάποιες πληροφορίες:

- Name: Εισάγετε ένα μοναδικό όνομα για την αλληλεπίδραση ώστε να την αναγνωρίζετε εύκολα στο σύστημα.

- Description: Παρέχετε μια συνοπτική περιγραφή του τι κάνει η αλληλεπίδραση και τυχόν σημαντικές λεπτομέρειες που τη διαφοροποιούν από άλλες αλληλεπιδράσεις.
- Notification Messages: Διαμορφώστε τρεις τύπους μηνυμάτων ειδοποίησης που θα εμφανίζονται ως αναδυόμενα παράθυρα στην εφαρμογή ProofSpace για κινητά:
  - Success Message: Αυτό το μήνυμα εμφανίζεται όταν η αλληλεπίδραση ολοκληρωθεί επιτυχώς, ενημερώνοντας τον χρήστη για την επιτυχή έκβαση.
  - Pending Message: Αυτό το μήνυμα εμφανίζεται όταν η αλληλεπίδραση βρίσκεται ακόμη σε εξέλιξη ή εκκρεμεί η ολοκλήρωσή της, ενημερώνοντας τον χρήστη για την κατάσταση.
  - Failure Message: Σε περίπτωση αποτυχίας της αλληλεπίδρασης, αυτό το μήνυμα εμφανίζεται, παρέχοντας ανατροφοδότηση σχετικά με την ανεπιτυχή προσπάθεια.

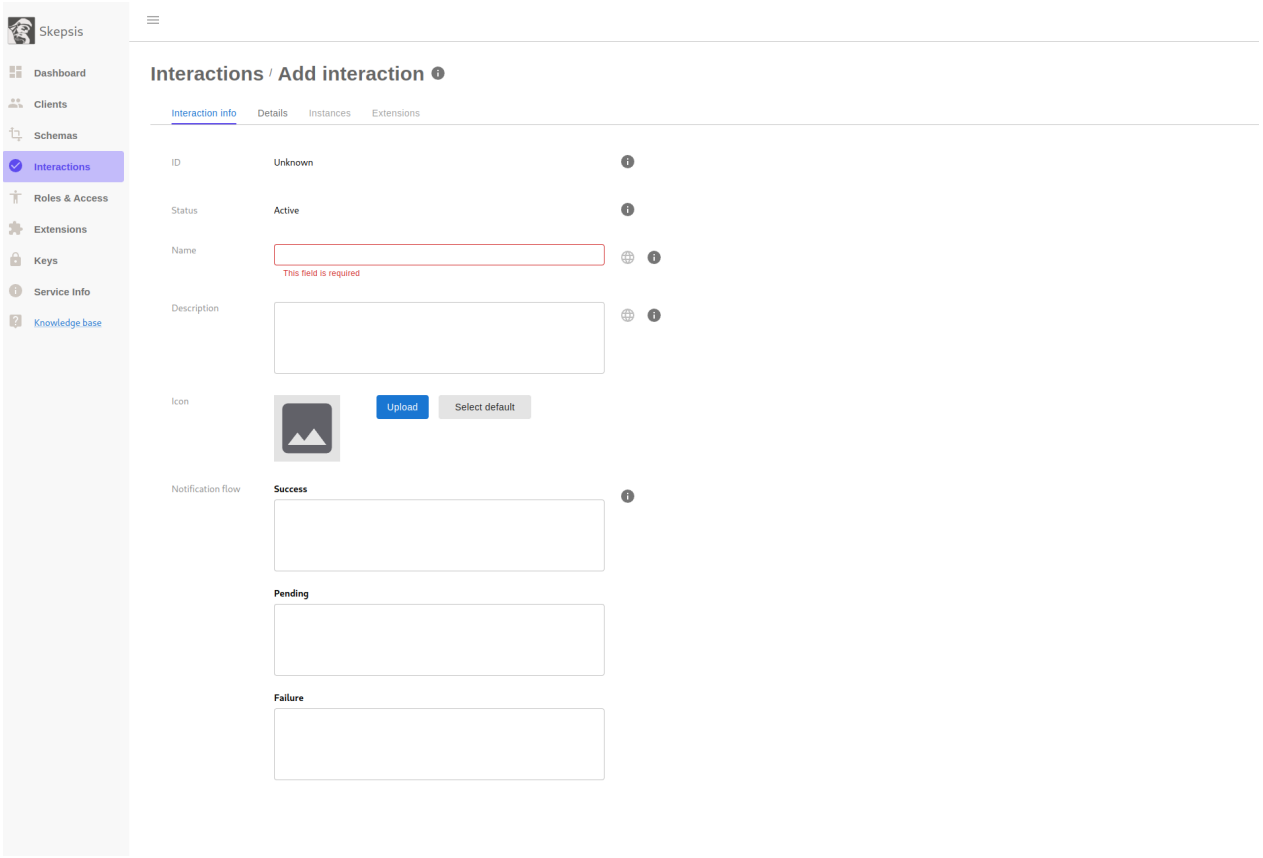


Figure 4.4.13: Πληροφορίες Αλληλεπίδρασης

Έπειτα, πλοηγηθείτε στην καρτέλα Details, όπου θα ρυθμίσετε τις πιο τεχνικές λεπτομέρειες της αλληλεπίδρασης. Αυτό περιλαμβάνει τον καθορισμό του τρόπου έναρξης της αλληλεπίδρασης, τη διαχείριση της ταυτόχρονης λειτουργίας και τη διαμόρφωση πρόσθετων ρυθμίσεων και αποτελεσμάτων.

Initiation Media Αρχικά, ορίστε τα μέσα έναρξης για να καθορίσετε τον τρόπο με τον οποίο θα ενεργοποιηθεί η αλληλεπίδραση. Υπάρχουν τρεις επιλογές:

- QR Code: Ενεργοποιείται όταν ο καταναλωτής σαρώνει έναν κωδικό QR χρησιμοποιώντας την εφαρμογή ProofSpace απ' το κινητό του.
- Button: Ενεργοποιείται από το πάτημα του κουμπιού «Proceed» από τον χρήστη στην οθόνη αλληλεπίδρασης εντός της εφαρμογής.

- Incoming Push from Service: Ενεργοποιείται από μια κλήση API από μια εξωτερική υπηρεσία προς το API του ProofSpace.

Concurrency Type Στη συνέχεια, επιλέξτε τον τύπο ταυτόχρονης εκτέλεσης, ο οποίος υπαγορεύει τον τρόπο με τον οποίο μπορεί να εκτελεστεί η αλληλεπίδραση:

- Once: Η αλληλεπίδραση μπορεί να ενεργοποιηθεί μόνο μία φορά ανά πελάτη.
- Sequenced: Η αλληλεπίδραση μπορεί να πραγματοποιηθεί πολλές φορές από τον ίδιο πελάτη, αλλά μόνο διαδοχικά- μια νέα αλληλεπίδραση μπορεί να ξεκινήσει μόνο αφού ολοκληρωθεί η τρέχουσα.
- Concurrent: Επιτρέπει την ταυτόχρονη έναρξη της αλληλεπίδρασης από τον ίδιο πελάτη.

Store Interaction Event Εξετάστε αν θα πρέπει να επιλέξετε την επιλογή «Store interaction event». Με την επιλογή αυτού του πλαισίου θα αποθηκευτούν όλες οι πληροφορίες που σχετίζονται με το συμβάν. Εάν δεν επιλεγεί, μόνο περιορισμένα δεδομένα, όπως το συμβάν της αλληλεπίδρασης, τα ονόματα των κοινών πιστοποιητικών και ορισμένα απαιτούμενα πιστοποιητικά με επίπεδο εμπιστοσύνης «Service attested» (Βεβαιωμένη υπηρεσία), θα είναι ορατά στον πίνακα οργάνων σας για 24 ώρες πριν διαγραφούν οριστικά. Οι πληροφορίες σχετικά με τα εκδοθέντα πιστοποιητικά θα διαγραφούν αμέσως μετά την ολοκλήρωση της αλληλεπίδρασης.

Required and Issued Credentials

- Required Credential Definitions: Καθορίστε ποιους ορισμούς πιστοποιητικών πρέπει να διαθέτει ένας χρήστης για να ξεκινήσει και να ολοκληρώσει την αλληλεπίδραση.
- Issued Credentials: Καθορίστε ποια πιστοποιητικά υπηρεσίας θα εκδοθούν στο χρήστη εάν η αλληλεπίδραση είναι επιτυχής.

Webhook Configuration Τέλος, ρυθμίστε ένα webhook που θα κληθεί μετά την ολοκλήρωση της αλληλεπίδρασης. Αυτό το webhook μπορεί να χρησιμοποιηθεί για την αυτοματοποίηση περαιτέρω διεργασιών ή την ειδοποίηση άλλων συστημάτων για το αποτέλεσμα της αλληλεπίδρασης.

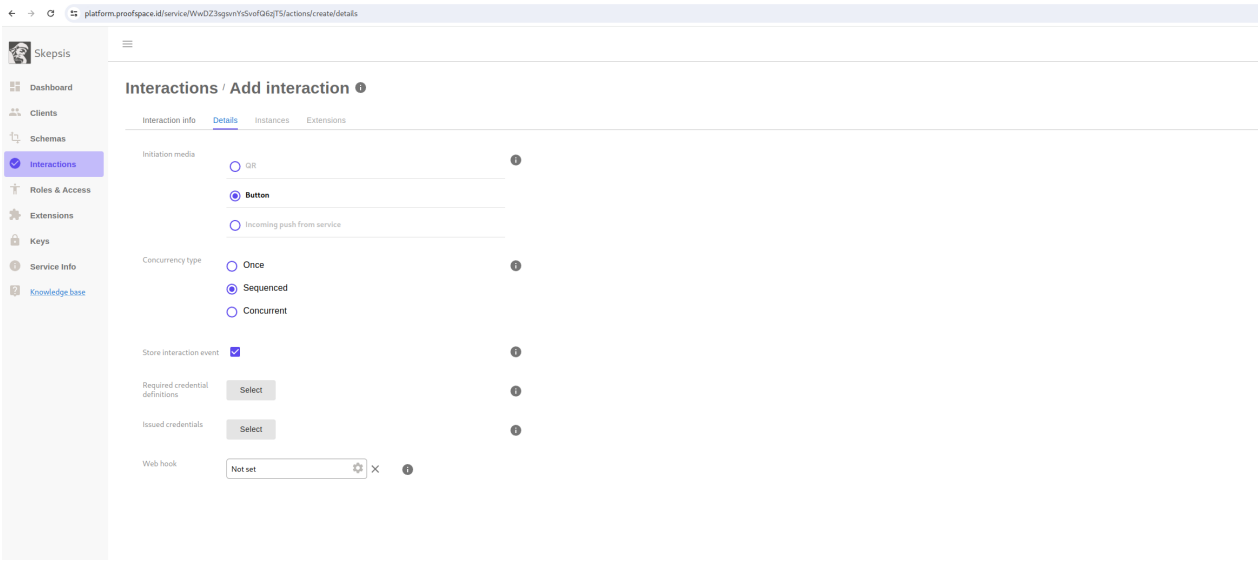


Figure 4.4.14: Λεπτομέρειες Αλληλεπίδρασης

Αφού διαμορφωθούν όλα τα πεδία, απλώς μεταβείτε στη σελίδα «Interactions» κάνοντας κλικ σε αυτήν στην αριστερή γραμμή πλοήγησης. Θα πρέπει να μπορείτε να δείτε τη νεοδιαμορφωμένη αλληλεπίδραση να εμφανίζεται εκεί. Η αλληλεπίδραση αποθηκεύεται αυτόματα μόλις εισαγάγετε όλες τις απαραίτητες λεπτομέρειες.

4.4.2.2 Αλληλεπιδράσεις της Εφαρμογής

Η εφαρμογή μας χρησιμοποιεί έξι διαφορετικές αλληλεπιδράσεις για τη διαχείριση των πιστοποιητικών των χρηστών:

- Αλληλεπίδραση QR: Η αρχική αλληλεπίδραση είναι ένας μηχανισμός βασισμένος σε κώδικα QR, ο οποίος έχει σχεδιαστεί για να καταγράφει το αποκεντρωμένο αναγνωριστικό του συνδεδεμένου χρήστη (DiD) και το αναγνωριστικό ειδοποίησής του. Αυτές οι πληροφορίες είναι ζωτικής σημασίας, καθώς δημιουργούν μια σύνδεση με τον λογαριασμό ProofSpace του χρήστη, επιτρέποντας στο σύστημα να στέλνει μελλοντικά πιστοποιητικά άμεσα και με ασφάλεια. Το αναγνωριστικό ειδοποίησης χρησιμεύει ως συγκατάθεση του χρήστη, επιτρέποντας στο σύστημα να μεταφέρει πιστοποιητικά στο λογαριασμό του και να λαμβάνει ειδοποιήσεις push.
- Αλληλεπιδράσεις Incoming Push from Service : Οι υπόλοιπες πέντε αλληλεπιδράσεις είναι δομημένες με τον ίδιο τρόπο και κατηγοριοποιούνται ως αλληλεπιδράσεις «Incoming Push from Service». Καθεμία από αυτές είναι υπεύθυνη για την έκδοση ενός από τα πέντε διαφορετικά πιστοποιητικά που περιγράφονται στο σύστημα (εγγραφή στο Πανεπιστήμιο, εγγραφή σε εξάμηνο, εγγραφή σε γνωστικό αντικείμενο, πιστοποιητικό βαθμού και πιστοποιητικό πτυχίου). Δεδομένου ότι η ρύθμιση για αυτές τις αλληλεπιδράσεις είναι η ίδια εκτός από το συγκεκριμένο πιστοποιητικό που εκδίδεται, θα περιγράψουμε λεπτομερώς μόνο μία από αυτές ως αντιπροσωπευτικό παράδειγμα.

Αλληλεπίδραση University Registration Ο στόχος της αλληλεπίδρασης University Registration είναι η σύνδεση της εφαρμογής ProofSpace και του DiD του χρήστη με την υπηρεσία μας, εξασφαλίζοντας τη συγκατάθεση του χρήστη για την λήψη ειδοποιήσεων. Μας επιτρέπει επίσης να καταγράψουμε το DiD του φοιτητή και να το αποθηκεύσουμε στη βάση δεδομένων του backend μας, το οποίο είναι απαραίτητο για την έκδοση νέων πιστοποιητικών. Η αλληλεπίδραση διαμορφώνεται ως εξής:

- Initiation Media: Η αλληλεπίδραση ξεκινά μέσω σάρωσης ενός κωδικού QR. Αυτός ο κωδικός QR εμφανίζεται στο web UI την πρώτη φορά που ο φοιτητής συνδέεται, μια διαδικασία που περιγράφεται λεπτομερώς στην ενότητα frontend.
- Concurrency Type: Ο τύπος ταυτόχρονης λειτουργίας για αυτή την αλληλεπίδραση έχει οριστεί σε «Once», που σημαίνει ότι μπορεί να ενεργοποιηθεί μόνο μία φορά για κάθε χρήστη, γεγονός που ευθυγραμμίζεται με τη χρήση της για σκοπούς αρχικής εγγραφής.
- Store Interaction Event: Επιλέγουμε να αποθηκεύσουμε το συμβάν αλληλεπίδρασης για να διασφαλίσουμε τη διατήρηση ενός αρχείου αυτής της θεμελιώδους δραστηριότητας, το οποίο βοηθά στις διαδικασίες ελέγχου.
- Required Credential Definitions: Το «Notification ID» είναι το απαιτούμενο πιστοποιητικό για αυτή την αλληλεπίδραση. Είναι απαραίτητο καθώς αντιπροσωπεύει τη συγκατάθεση του χρήστη για μελλοντικές ειδοποιήσεις.
- Certificates Issued: Η συγκεκριμένη αλληλεπίδραση δεν εκδίδει άμεσα πιστοποιητικά. Αντίθετα, η επιτυχής ολοκλήρωσή της πυροδοτεί μια ακολουθία γεγονότων που διευκολύνεται από το backend μας.
- Διαμόρφωση Webhook: Με την ολοκλήρωση αυτής της αλληλεπίδρασης, ενεργοποιείται ένα από τα endpoint της εφαρμογής μας. Αυτή η ενέργεια ανακτά το DiD του φοιτητή από τα δεδομένα της αλληλεπίδρασης και το αποθηκεύει στη βάση δεδομένων του backend μας. Μετά από αυτή την ανάκτηση δεδομένων, το ίδιο endpoint πραγματοποιεί κλήση στο ProofSpace API για την έκδοση του πιστοποιητικού εγγραφής στο Πανεπιστήμιο.

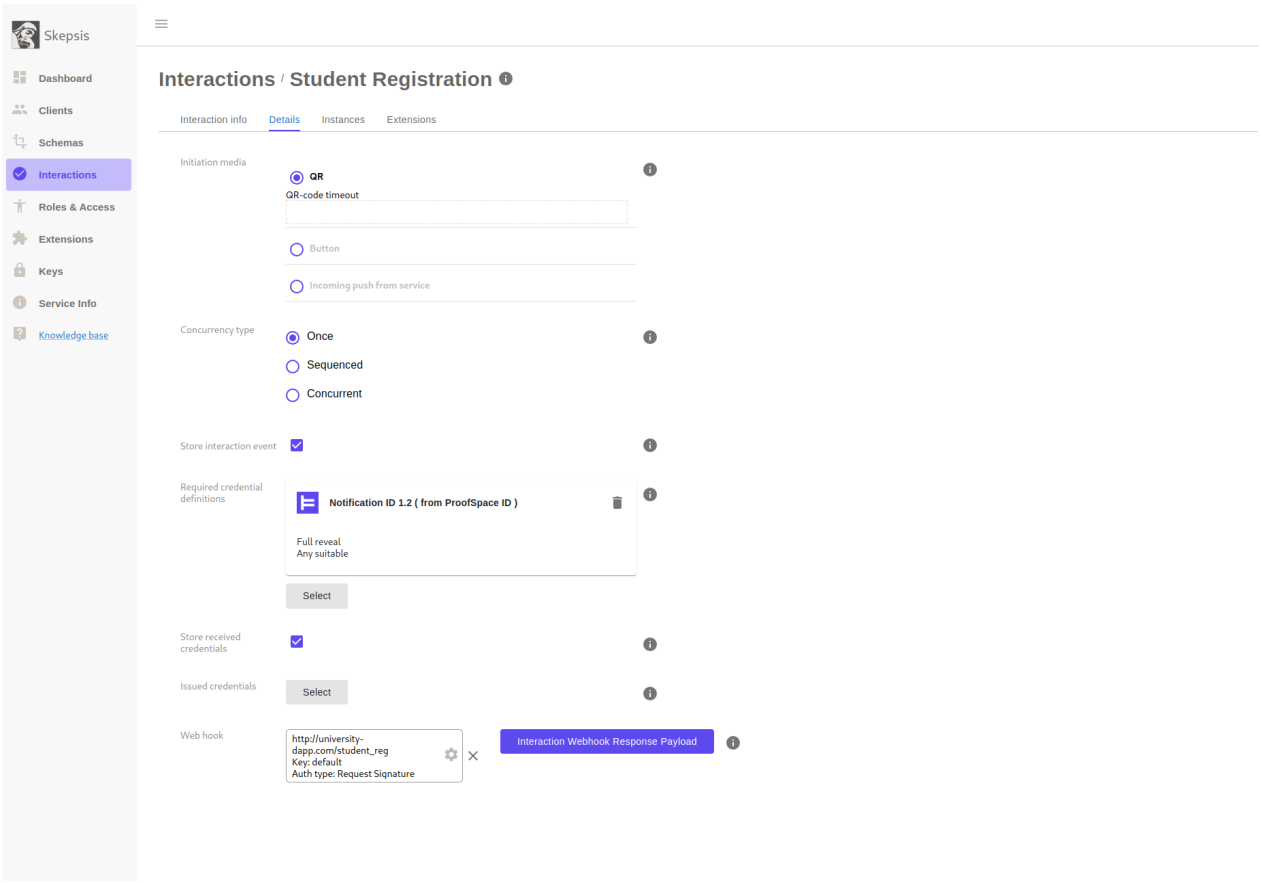


Figure 4.4.15: Λεπτομέρειες Αλληλεπίδρασης Student Registration

Αλληλεπίδραση Course Registration Η Αλληλεπίδραση εγγραφής μαθημάτων έχει σχεδιαστεί ειδικά για την έκδοση πιστοποιητικών εγγραφής μαθημάτων σε φοιτητές που εγγράφονται σε μαθήματα. Αυτή η αλληλεπίδραση λειτουργεί πανομοιότυπα με τις άλλες τέσσερις, διαφέροντας μόνο ως προς τα συγκεκριμένα πιστοποιητικά που εκδίδονται. Διαμορφώνεται ως εξής:

- Initiation Media: Η έναρξη για αυτή την αλληλεπίδραση είναι «Incoming Push from Service». Το backend σύστημά μας ενεργοποιεί αυτή την έκδοση πιστοποιητικών όταν ένας φοιτητής εγγράφεται επιτυχώς σε ένα μάθημα, με βάση τις επικυρώσεις που πραγματοποιούνται εντός της εφαρμογής.
- Concurrency Type: Δεδομένου ότι το μέσο εκκίνησης είναι «Incoming Push from Service», δεν υπάρχει ανάγκη για τύπο ταυτόχρονης πρόσβασης.
- Store Interaction Event: Επιλέγουμε να αποθηκεύσουμε και τις λεπτομέρειες του συμβάντος αυτής της αλληλεπίδρασης.
- Required Credential Definitions: Δεν υπάρχει απαιτούμενο πιστοποιητικό για αυτή την αλληλεπίδραση.
- Certificates Issued: Για αυτή την αλληλεπίδραση, εκδίδεται το πιστοποιητικό «Εγγραφή σε μάθημα». Είναι σημαντικό να σημειωθεί ότι, ενώ ο μηχανισμός είναι ο ίδιος, το συγκεκριμένο πιστοποιητικό που εκδίδεται διαφέρει μεταξύ των διαφόρων αλληλεπιδράσεων - η κάθε μία είναι προσαρμοσμένη στην αντίστοιχη ακαδημαϊκή δραστηριότητα (π.χ. εγγραφή σε εξάμηνο, έκδοση βαθμών).
- Διαμόρφωση Webhook: Δεν απαιτείται webhook. Η διαδικασία διαχειρίζεται εξ ολοκλήρου εσωτερικά από το backend, το οποίο χειρίζεται απευθείας την έκδοση των πιστοποιητικών με βάση την επιτυχή εγγραφή και επικύρωση του μαθήματος.

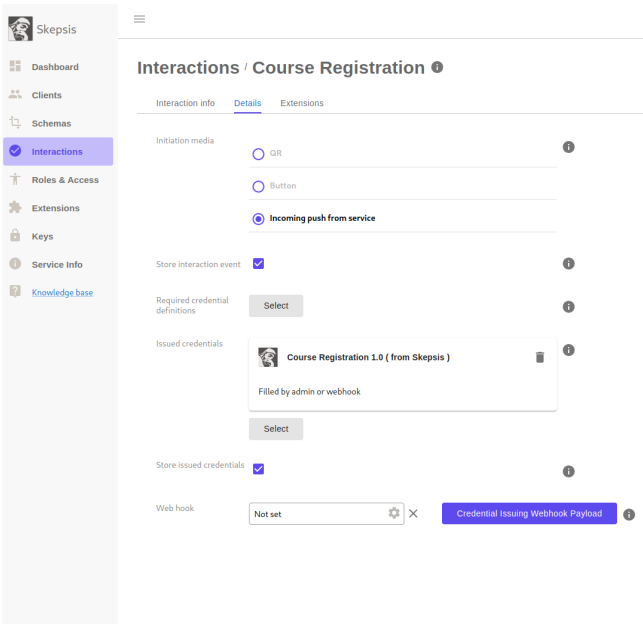


Figure 4.4.16: Λεπτομέρειες Αλληλεπίδρασης Course Registration

4.4.3 Έκδοση πιστοποιητικών μέσω του ProofSpace API

Σε αυτή την ενότητα, θα παρουσιάσουμε πώς να εκδίδετε πιστοποιητικά ProofSpace χρησιμοποιώντας το API του ProofSpace.

4.4.3.1 Ρύθμιση κλειδιών

Για να διασφαλιστεί ότι μόνο εξουσιοδοτημένες οντότητες μπορούν να εκδώσουν πιστοποιητικά από την υπηρεσία μας, το ProofSpace χρησιμοποιεί έναν μηχανισμό ελέγχου ταυτότητας που βασίζεται στην ασύμμετρη κρυπτογραφία, συγκεκριμένα την RSA. Αυτό το χαρακτηριστικό ασφαλείας απαιτεί τα αιτήματα προς το API του ProofSpace να υπογράφονται με το ιδιωτικό μας κλειδί. Για να ρυθμίσετε τα κατάλληλα κλειδιά, αποκτήστε πρόσβαση στην ενότητα «Keys» στο ταμπλό του ProofSpace και κάντε κλικ στο «Add» για να ξεκινήσετε τη διαμόρφωση ενός νέου κλειδιού. Στη συνέχεια, ρυθμίστε τις ακόλουθες επιλογές:

- Όνομα κλειδιού εισόδου: Ορίστε ένα όνομα στο κλειδί σας για εύκολη αναγνώριση.
- Επιλέξτε τύπο κλειδιού: Έχετε δύο επιλογές για τη ρύθμιση του κλειδιού σας:
  - RSA Public Key: Εάν διαθέτετε ήδη ένα ζεύγος κλειδιών RSA, επιλέξτε αυτή την επιλογή και επικολλήστε το δημόσιο κλειδί σας στο παρεχόμενο πλαίσιο κειμένου.
  - RSA Keypair: Εάν δεν διαθέτετε ζεύγος κλειδιών RSA, επιλέξτε αυτή την επιλογή και κάντε κλικ στο εικονίδιο «generate». Το σύστημα θα δημιουργήσει αυτόματα ένα νέο ζεύγος κλειδιών RSA για εσάς. Είναι σημαντικό να αντιγράψετε και να αποθηκεύσετε με ασφάλεια το ιδιωτικό κλειδί RSA που εμφανίζεται στο σχετικό πλαίσιο, καθώς θα είναι απαραίτητο για την έκδοση πιστοποιητικών. Σημειώστε ότι, αν και αυτή η επιλογή είναι πιο απλή, ενδέχεται να έχει ορισμένες επιπτώσεις στην ασφάλεια.
- Set as Default for Webhook Requests: Εάν επιθυμείτε αυτό το ζεύγος κλειδιών να είναι η προεπιλεγμένη μέθοδος ελέγχου ταυτότητας για τα εισερχόμενα αιτήματα webhook, επιλέξτε την επιλογή «Set as default for incoming webhook requests» (Ορισμός ως προεπιλογή για εισερχόμενα αιτήματα webhook). Εάν η περίπτωση χρήσης σας απαιτεί διαφορετικά ζεύγη κλειδιών για διάφορα πιστοποιητικά, μπορείτε να επιλέξετε να καταργήσετε την επιλογή αυτή. Ωστόσο, για λόγους απλότητας και συνέπειας, είναι σκόπιμο να την κρατήσετε επιλεγμένη για την εφαρμογή μας.

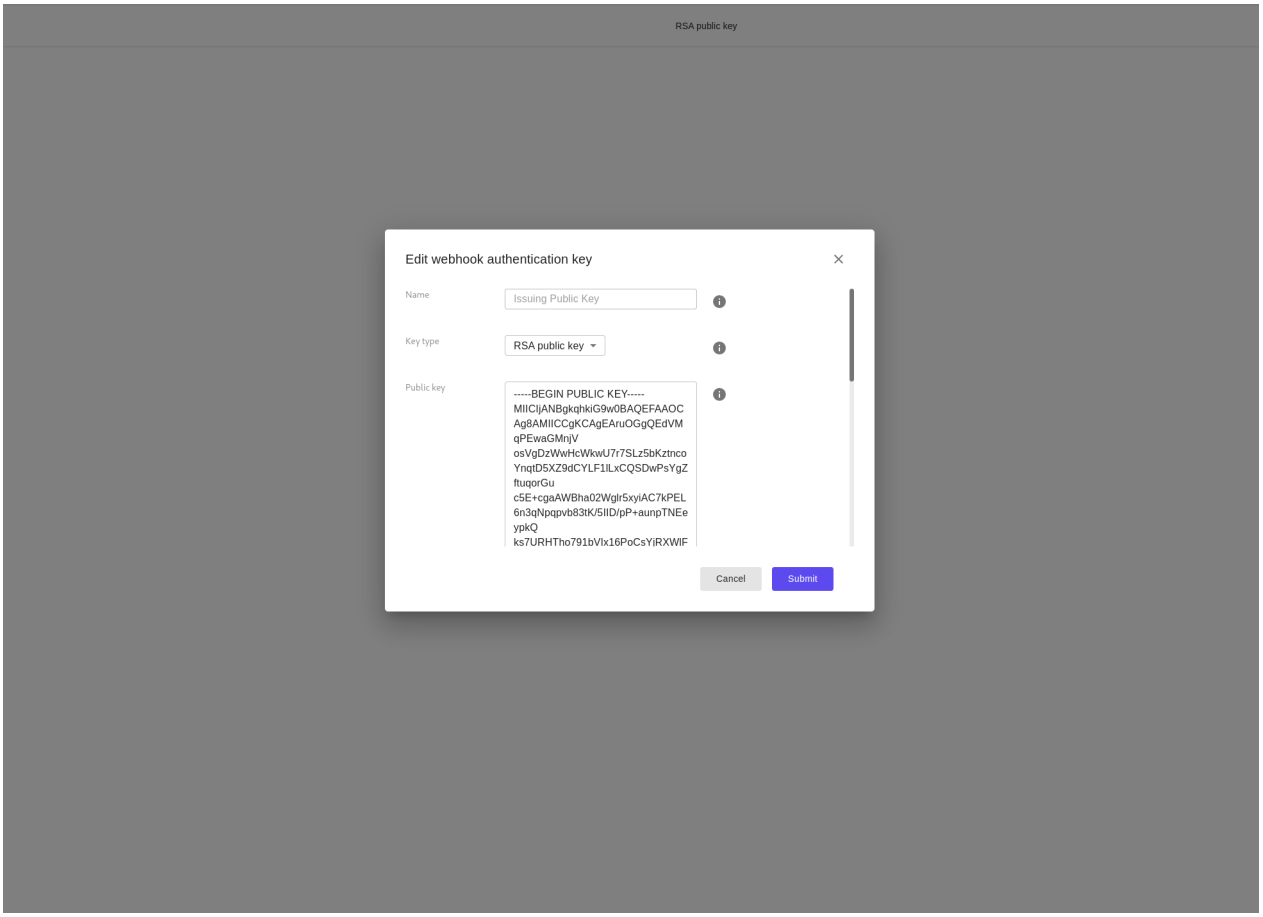


Figure 4.4.17: Προσθήκη Κλειδιού

4.4.3.2 Κλήση του ProofSpace API

Σε αυτή την παράγραφο, θα εξερευνήσουμε τον τρόπο κλήσης του API ProofSpace για την έκδοση ενός πιστοποιητικού University Registration από ένα περιβάλλον Node.js. Αυτή η μέθοδος είναι εφαρμόσιμη για την έκδοση οποιουδήποτε τύπου πιστοποιητικών χρησιμοποιώντας την ίδια προσέγγιση.

Για να εκδώσετε ένα πιστοποιητικό μέσω του ProofSpace API, πρέπει να στείλετε ένα αίτημα POST στη συγκεκριμένη διεύθυνση URL που παρέχεται από το ProofSpace. Αρχικά, λάβετε το «serviceDid» σας από τη σελίδα Service Info στο ταμπλό της υπηρεσίας ProofSpace. Στην περίπτωσή μας, το «serviceDid» είναι WwDZ3sgsvnYsSvofQ6zjT5, οπότε η διεύθυνση URL για την έκδοση πιστοποιητικών θα είναι: 'https://platform.proofspace.id/service-backend/v1/service/WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued'

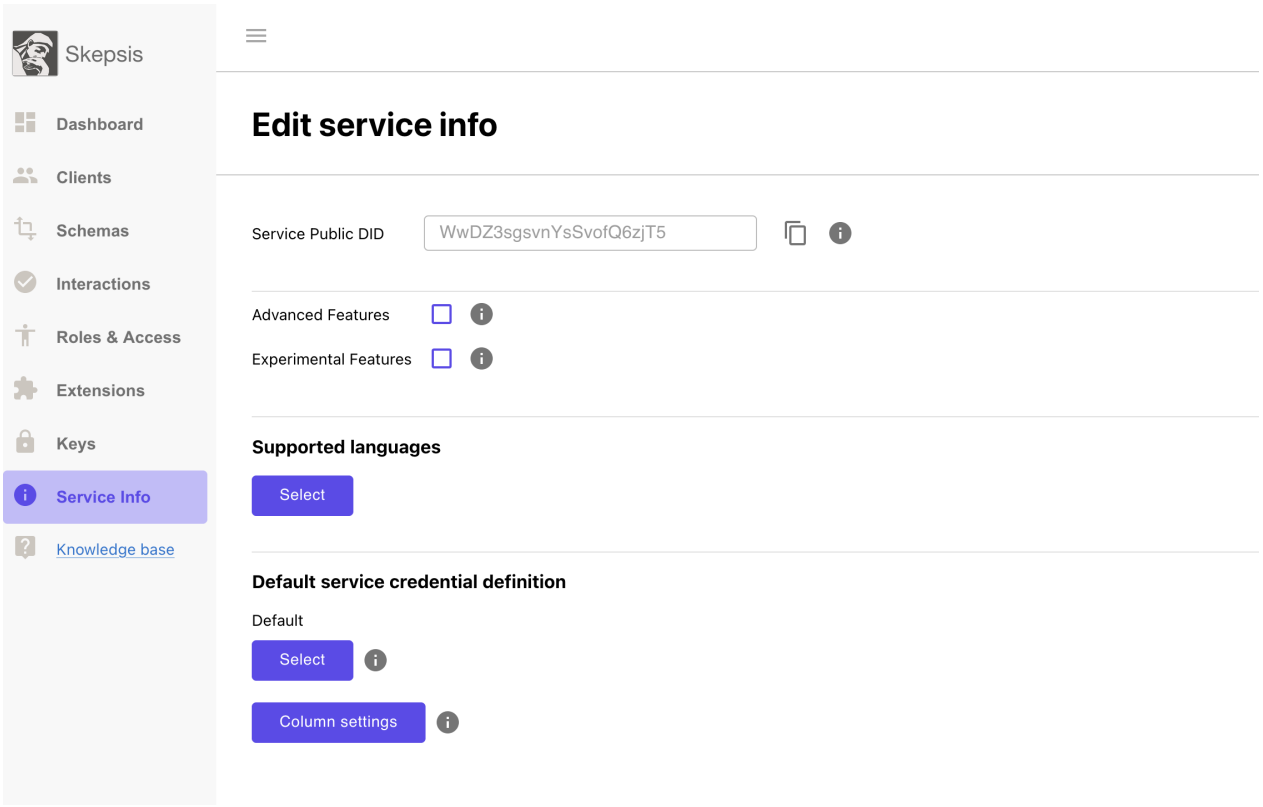


Figure 4.4.18: Ανάκτηση Service ID

Κατά τη διαμόρφωση του σώματος της αίτησης POST για την έκδοση ενός πιστοποιητικού μέσω του API ProofSpace, η δομή πρέπει να είναι η ακόλουθη:

```
{
  serviceDid: string,
  subscriberConnectDid: string,
  credentials: WebhookCredentialValuesDTO[],
  issuedAt: number,
  subscriberEventId: string,
  actionTemplate?: string,
  actionParams?: Array<NameValueDTO>,
}
```

Ακολουθεί μια ανάλυση των πιο σημαντικών πεδίων:

- serviceDid: Το DiD της υπηρεσίας, το οποίο ανακτήσατε στο προηγούμενο βήμα από το ταμπλό της υπηρεσίας ProofSpace.
- subscriberConnectDid: Αυτό το πεδίο πρέπει να περιέχει το DiD του χρήστη που θα λάβει το εκδοθέν πιστοποιητικό.
- credentials: Εδώ, καθορίζετε έναν πίνακα WebhookCredentialValuesDTO, ο οποίος ορίζει κάθε πιστοποιητικό που εκδίδεται σε αυτή την αίτηση.
- issuedAt: Πρόκειται για μια ημερομηνία που δείχνει πότε εκδόθηκαν τα πιστοποιητικά.

WebHookCredentialValuesDTO Το WebHookCredentialValuesDTO είναι ένα αντικείμενο που περιλαμβάνει όλες τις απαραίτητες λεπτομέρειες σχετικά με ένα πιστοποιητικό που εκδίδεται. Ακολουθεί μια λεπτομερής εξήγηση για κάθε συστατικό μέσα σε αυτό το αντικείμενο:

```
{
  credentialId: string,
```

```
  schemaId: string,
  fields: { name: string, value: string }[],
  utcIssuedAt: number,
  revoked: boolean,
  utcRevokedAt?: number,
}
```

- credentialId: Πρόκειται για το αναγνωριστικό του ορισμού του πιστοποιητικού που εκδίδεται.
- schemaId: Αντιπροσωπεύει το αναγνωριστικό του σχήματος με το οποίο συμμορφώνεται αυτό το πιστοποιητικό.
- fields: Πρόκειται για έναν πίνακα ζευγών κλειδιών-τιμών, όπου κάθε ζεύγος αντιστοιχεί σε ένα χαρακτηριστικό που ορίζεται στο σχήμα του πιστοποιητικού. Κάθε όνομα στο ζεύγος αντιστοιχεί σε ένα χαρακτηριστικό του σχήματος και η τιμή είναι τα δεδομένα για το εν λόγω χαρακτηριστικό.
- utcIssuedAt: Η ημερομηνία που σηματοδοτεί την ημερομηνία έκδοσης του πιστοποιητικού.

Για να ανακτήσετε το 'credentialId', μεταβείτε στη σελίδα 'Schemas' του ταμπλό του ProofSpace και αντιγράψτε το επιθυμητό από τη στήλη 'Credential ID'.

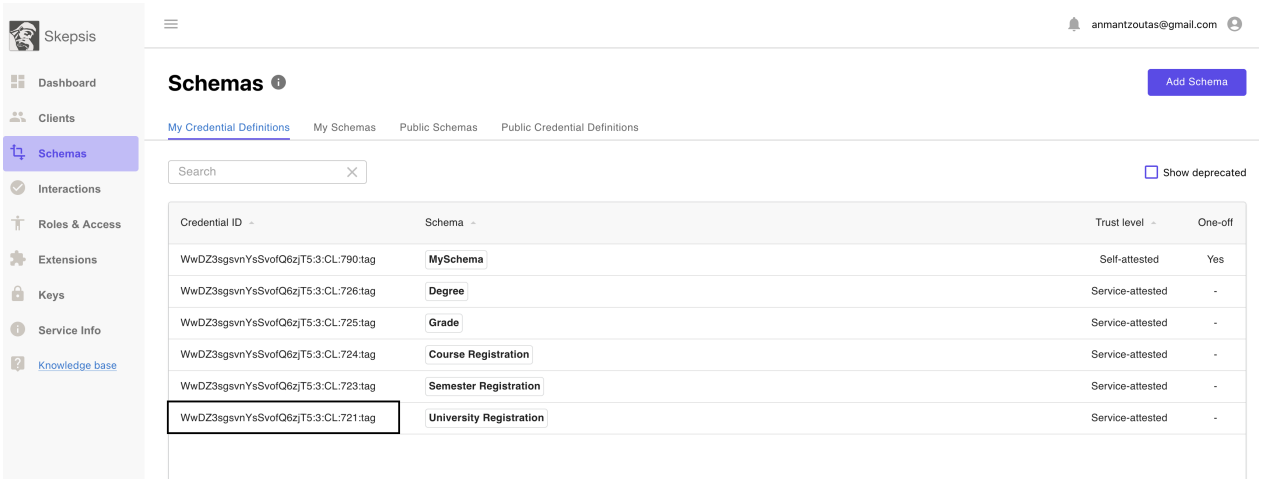


Figure 4.4.19: University Registration Credential ID

Για να ανακτήσετε το «schemaId», μεταβείτε στην καρτέλα «Τα σχήματά μου» στη σελίδα «Σχήματα», κάντε κλικ στο σχήμα που θέλετε να χρησιμοποιήσετε και, στη συνέχεια, αντιγράψτε το από τη σελίδα πληροφοριών.

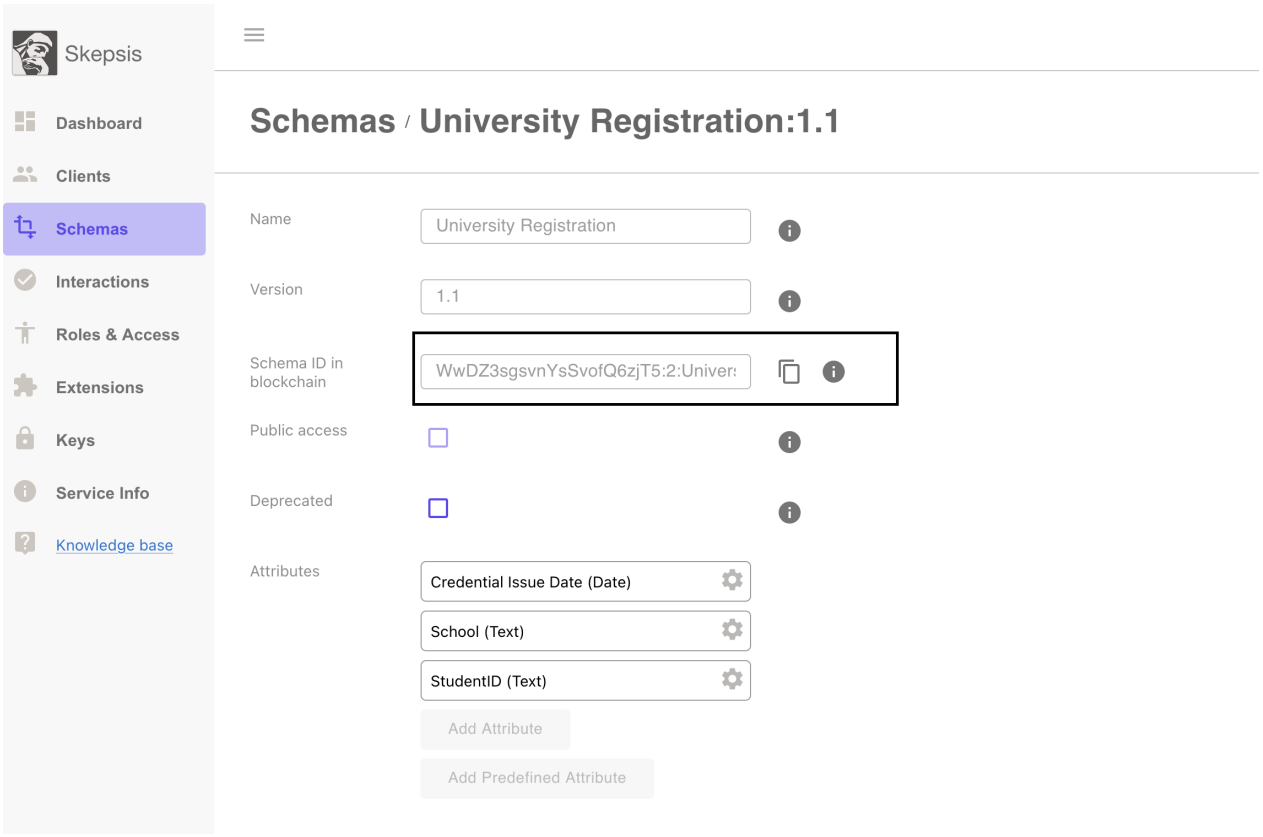


Figure 4.4.20: University Registration Schema ID

Για αυτή την επίδειξη, θα χρησιμοποιήσουμε συγκεκριμένα χαρακτηριστικά για τη διαδικασία έκδοσης πιστοποιητικών. Τα χαρακτηριστικά περιλαμβάνουν το όνομα της σχολής «ECE NTUA» και τον αριθμό μητρώου «el17108», ο οποίος είναι ο προσωπικός μου αριθμός μητρώου. Επιπλέον, θα χρησιμοποιήσουμε το αναγνωριστικό μου ProofSpace ως «subscriberConnectDid». Με βάση αυτή τη ρύθμιση, το παρακάτω είναι το απόσπασμα κώδικα JavaScript που έχουμε αναπτύξει μέχρι στιγμής:

```
const url = "https://platform.proofspace.id/service-backend/v1/service/
    WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued";
const serviceDid = "WwDZ3sgsvnYsSvofQ6zjT5";
const credentialId = "WwDZ3sgsvnYsSvofQ6zjT5:3:CL:721:tag";
const schemaId = "WwDZ3sgsvnYsSvofQ6zjT5:2:University Registration:1.1";

const credentialValues = {
    credentialId,
    schemaId,
    fields: [
        { name: "School", value: "ECE NTUA" },
        { name: "StudentID", value: "el17108" },
    ],
    utcIssuedAt: new Date().getTime(),
    revoked: false
};

const request = {
    serviceDid,
    subscriberConnectDid: userDid,
    subscriberEventId: "",
    credentials: [credentialValues],
    issuedAt: new Date().getTime(),
};
```

Μετά από αυτά τα βήματα, πρέπει να ρυθμίσουμε τις επικεφαλίδες της αίτησης. Πρώτον, ορίζουμε το

«Content-Type» σε «application/json». Στη συνέχεια, ρυθμίζουμε την επικεφαλίδα «X-Body-Signature» για την εξουσιοδότηση της αίτησης. Αυτό περιλαμβάνει την ασύμμετρη κρυπτογραφία που συζητήσαμε νωρίτερα. Σειριοποιούμε το σώμα της αίτησης σε μια συμβολοσειρά, την υπογράφουμε χρησιμοποιώντας το ιδιωτικό κλειδί RSA που δημιουργήσαμε στην προηγούμενη υποενότητα και κωδικοποιούμε την υπογραφή σε base64. Τέλος, προσθέτουμε την κωδικοποιημένη υπογραφή στην επικεφαλίδα «X-Body-Signature». Σημειώστε ότι για χάρη αυτής της επίδειξης, θα συμπεριλάβουμε το ιδιωτικό κλειδί στον κώδικα σε απλό κείμενο. Ωστόσο, αυτό δεν αποτελεί καλή πρακτική και καλό είναι να αποφεύγεται σε πραγματικές εφαρμογές. Ο κώδικας έχει ως εξής, χρησιμοποιώντας την ενσωματωμένη βιβλιοθήκη «crypto» για την υπογραφή του αντικειμένου:

```
const headers = {
    'Content-Type': 'application/json'
};

const binaryBody = Buffer.from(JSON.stringify(request), "utf-8");

const private_key = `-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAtsEaCBfGaRhWfkLuiKvK8InKgZyl0od1DcNAlVi6YaEmEruS
e8lPJfewXR3uxIg3SYOAGI6LYFFIJxAMcvHYEm0axW5lo4ov1hYRMA76tl/eo1Kp
3Cp7Ai8W9GshcQzbFwPX9UjbijHkRkwCIpW+iK/aipj/nBg1LJsOR4b9hpbB92Hz
+b4nviGBD5CA0vzbyDhLq/4oQbyOvAsAFAqKZEqBHq4s+WcKDHXFkvsmxnzexAlH
JtGjhFU+B3myXPMoMEHUInoKW0cSez5/WmkU0rPcNu4jmVmpjnVek+7s9q8F+blA
RaI4WKUUM+MA3sBGkMt3QeyKOm0L9JfA5C3q0ZmJ4xkJvaK0zCdml8DbgKasIy/L
JMXkQZyOZGlg2YsjcLK3lsElefAQ8d4/J4wSF+I/0uVKV/vxzVBx9kKgpUGJQu9l
0pGsxaQyZ94wUBiJybZqx8EUQmLIH7ONjSAGFHEISPLOlqRihJRz3zVdIKn04k1I
ftqk477mz5YJD5zVq9w9qx77oCzfH34N8pU/vTUaFmoWHhdnrb8slXaATOMejksA
kCK/wqUhYnGol8cNq92uTC/IdcEbBkHyHyQ0HV88R1myoqY9bg4CbZzJDrtCzAUH
mN5fVCUVh7WL9Pt5pskD3zPPv77gxzJYLI+/9kOgwCXHrk9kGnFwgjsJHcUCAwEA
AQKCAgABsPbLNWEeLt36DCeF90UJBdVsbqUwNhW0u2ZZG3maFrIyAS9hWuMTIGAu
9Gx1wCb21LM2RvduPEJVLjZXR/qubPkO868YHNrFee3Nyo2ya1+1ljnBMllyEzp8
p5MnJy3N4Di66eIyPLT2m5R3enhJboswtirDaIvxbjvrGIo3GSURY4Qs/wACJ/A4
UyeGDFYCW9cDb/YyaYrOnbFx/gj61Z+tR4JeCoCSzfI2ooFg29355XYVm2IOgvVw
f0VRy+RG0g0jBosa/iTjJBhsn0VUj9qNXZXRwP0rNtXZu5/QxpzhC0BXhKFN3LU7
p7Gl3hSpGWDwqL48vS6bpF3iHrSqGN+d+WtNGCMWSNcq9b8Xp0ft7kx3XHMKYU/s
2+HtMO3jp0fxr4i7tG/DZpaahLA/HEIWG4AK5CEKOL0UDYtqwyoNedoPjhkr5nxf
Kx4BX40b2DG83hU5imsp2zP7XAkmiIpNYNdXosFrKoWcBU3vCKus9/MGS34ZNOR4
9LhRA1I478qfsUXcmx1PcmIRyyobanZsyzxad9n/yxDxCZ50nnxCMvJUWSRU9n8l
3dwafbgMV/ispxQXc9oTof61LCFVlrukjNYIcz+03V31q2/tp+dL2UNZWObT72ED
Aw8SPy6yB5BYpNU1BdC+jPmfDuDlnUPNSY8hBqV6otlXNIDCnwKCAQEA9GctrVnB
pZHRMwPZfL9VK7AiWkLdoWThtmuNHRZm0XhepYH8xaxlx6myQtkx3QYdJc1gpORC
BEoP8DtDdfOAZ4LGjic1GbXyHpI8bdgGf4H11EXZPr+PlAGLSyW8Ie+CYHfrKpYL
uHLaOIrSx4k+Fbj4B4obNEg/s4QSzQichI+UiwGmMuYx5KBlBXHnBWdN0+ICwbJq
Q3XRffVi0Xc0wDJjESXyVdqzMVnb18bHRkjwxTx2tgFZEBVYLeWtX3YTeQkzP/Yx
xIvzzXVtTrZa/DyD/bXs9ea9oJ0q80ABmyWUESTFrzrbYnAQG2UWYyPb6gv2Lw1D
1w5zTb5soZL0xwKCAQEAv20Pu8TfUABm1Imo/IsDA0koNB/0BpIEyfAeFHQ0UPR6
3rzli90Dn96PBfmlyOTPQir1RnSxpcxQW6eyU1ZGKasLFoS41mrnFTvbdOof/2Vd
kbNcwFb3wnLjm1YP4ImP36ugM4z4n++y03Uqbxz5wAsawq+5U+M0Z4miUrNRsPjz
OPBCl25VLXo3wFvP8qhITоDHnO5GNUBPCWZttlgi7Lwu8C/BRLGFSfIBAU7HFLIh
3foeLh6u+I4F1/ew8KMkrHxnQxDzMI8HaVtjwGtNdZmyXwf4M4ekJfGMowbPLh+k
3Ud68i2KPAmR2k50uS+gt/RfBYlqxqGCs22urK51EwKCAQAhaQVugCmft6pQfgM6
MXG0MjTP//oKOR8977/vrQNLOVg+6eEjKKkVPVZ4GvbdDxvm8gd3Ixj/Oxyoq4FS
W4I/JDp+bk+s/xllInqOTYy924c+iHnlQQygoAFpuXPZc8Nv6pTEUpSxl0nLlvav
fkx1bb7CURfc8s96e+CSFM5ItNQKmKQeqydA6MhGkUaSj4etImVcE0bBiBamUGg8
2D3m1DyEuJpDME10vqL2AV92dTwsylI0GUUSlMhkJdSoixyoXCCcasxs4PDoMlhJ
/VW8y40Dnt4Q04UysqmPbT2K9v0hxtQbWOFoWIdrJ0uDLADBIvd5gSe3+2q5wwFY
qR07AoIBAQCQ9qh5i8eJMCSvQ2iDh0pKzLXzc1c7Oo0B2CMPXkjYHayGzQoQdtjt
On8BCWNw1UXXV+zS5EA7zJP2NPKohacwmuUUsoG/bsl8nQ2+ZibBJCRN2ESfK7OI
tjSDSBNR+RrJSZFSOKl3hPApjosKBilkM4SadrSkQGVJG0Of2Ug9+ulF7zyerOE4
RVu/Akhh/Op0+LUxtIfaf87a0YS4F9qq6H+rSJ7VxJGpw+ja8qt0kslu7qC4UGg0
tmjaeiq2sPVu4e7ZLRbdaRM1yA2shBXuvuwDZ2egrlGyok/cjaWLyk7xe9YxXCdY
++5GIi/mpZjSde8XpE8GCP2Pn7ot7xedAoIBAQDEnlbgWPS2tldLoVFG4ega9x09
G/SVen5kkmLRaBNLV6rGRnCQoYkosUdbKMUsTb8WJGJhIllu5u0SgbaYyxrG63ob
FdDgVjIPJneL1xnX3s6qP9084k2cBqwE3LtNfVK6eWYEwXvaeHQN/Ur3KbJQ3tDi
LkNpzd4ysqRmy01UOIQxWG0nnM+LDRjhmong8u+HAeiYmZ9WZ2D06nd5dbq19FeT
F0dJYc4jhBZQoW7oaaNAMJ4JY/ViM10ISzjwaWcanQ0B1pH0yk0oSReWd82BFQfb
sLC/A+iBdA4dNfB9fnB+zrF0Php+2mpMZOnNyo5Nvm2N08wVOYk5lho0psEj
-----END RSA PRIVATE KEY-----
`;

```

```
60 const key = crypto.createPrivateKey({ key: private_key });
61 const signature = crypto.sign('sha3-256', binaryBody, key);
62
63 headers['X-Body-Signature'] = signature.toString('base64');
```

Τέλος, μπορούμε να προχωρήσουμε στην αποστολή του αιτήματος στο ProofSpace API χρησιμοποιώντας το «axios»:

```
1  axios.post(
2      url,
3      binaryBody,
4      {
5          headers: headers
6      }
7  ).then((reply) => {
8      return reply.data;
9  }).catch((err) => {
10     throw err;
11 });
```

Ολόκληρος ο κώδικας βρίσκεται στο Παράρτημα Β.

Κεφάλαιο 5

Επίλογος

Στην παρούσα εργασία, επιδείξαμε την υλοποίηση μιας αποκεντρωμένης εφαρμογής που σχεδιάστηκε για τη διαχείριση ακαδημαϊκών διαδικασιών όπως η βαθμολόγηση, η εγγραφή μαθημάτων και η έκδοση πιστοποιητικών σε ένα πανεπιστημιακό περιβάλλον. Αξιοποιώντας την τεχνολογία blockchain και τα έξυπνα συμβόλαια στο δίκτυο Cardano, μειώσαμε σημαντικά το διοικητικό φόρτο που παραδοσιακά επιβαρύνει το προσωπικό του πανεπιστημίου, ιδίως τη γραμματεία. Το σύστημα αυτοματοποιεί πολλές από τις εργασίες ρουτίνας που κάποτε ήταν χειροκίνητες, μειώνοντας την πιθανότητα ανθρώπινου λάθους και αυξάνοντας την αποτελεσματικότητα των διαδικασιών διαχείρισης αρχείων και επαλήθευσης.

Ωστόσο, τα ακαδημαϊκά ιδρύματα χρήζουν εκ φύσεως κάποιο επίπεδο κεντρικής εξουσίας, θέτοντας περιορισμούς στον βαθμό στον οποίο το σύστημά μας μπορεί να αποκεντρωθεί πλήρως. Κρίσιμες ενέργειες εντός της εφαρμογής, όπως η κοπή NFT πτυχίου και η ενημέρωση σημαντικών ακαδημαϊκών πολιτικών, εξακολουθούν να απαιτούν την εποπτεία και την έγκριση της γραμματείας. Αυτός ο συγκεντρωτισμός εκδηλώνεται στον έλεγχο του κλειδιού διαχείρισης, το οποίο είναι απαραίτητο για την υπογραφή όλων των σημαντικών συναλλαγών. Κατά συνέπεια, ενώ έχουμε αποκεντρώσει πολλά στοιχεία της ακαδημαϊκής διοικητικής διαδικασίας, το σύστημα εξακολουθεί να βασίζεται σε κεντρικές μορφές εξουσίας για τη λήψη κρίσιμων αποφάσεων και την εποπτεία.

Επιπλέον, αρκετές πτυχές παραμένουν πλήρως συγκεντρωτικές, όπως η προσθήκη νέων φοιτητών στο σύστημα, η εισαγωγή νέων μαθημάτων και η ανάθεση μαθημάτων σε καθηγητές. Αυτές οι εργασίες απαιτούν άμεση εισαγωγή και επικύρωση από το διοικητικό προσωπικό, ενισχύοντας την παρουσία κεντρικού ελέγχου σε τομείς, ενώ τα σχετικά δεδομένα αποθηκεύονται σε μια παραδοσιακή βάση δεδομένων εκτός αλυσίδας. Παρά τους περιορισμούς αυτούς, η εφαρμογή αντιπροσωπεύει ένα ουσιαστικό βήμα προς τα εμπρός στον εκσυγχρονισμό της εκπαιδευτικής διοίκησης, παρέχοντας ένα ισχυρό πλαίσιο που ενισχύει σημαντικά τη διαφάνεια, την ασφάλεια και την αποτελεσματικότητα της διαχείρισης των ακαδημαϊκών αρχείων.

5.1 Τεχνολογικές προκλήσεις

5.1.1 Εκμάθηση Plutus

Η μετάβαση από το Ethereum στο Cardano παρουσίασε προκλήσεις σε αυτό το έργο, κυρίως λόγω της μη εξοικείωσής μου με το μοντέλο UTXO και τη Haskell. Η διαδικασία διευκολύνθηκε με τη χρήση εργαλείων όπως το Aiken για την ανάπτυξη των έξυπνων συμβολαίων και το Lucid για τις αλληλεπιδράσεις εκτός της αλυσίδας, παράλληλα με την εκπαιδευτική υποστήριξη από το "Plutus Pioneer Program" της Input Output Global (IOG).

5.1.2 Σκέψεις για το απόρρητο

Μια ακόμα σημαντική πρόκληση που αντιμετωπίστηκε ήταν η διατήρηση του απορρήτου των ακαδημαϊκών αρχείων και των πληροφοριών των φοιτητών. Ο κύριος μηχανισμός προστασίας της ιδιωτικότητας βασίζεται στην απόκρυψη των διευθύνσεων συμβολαίων και των token που χρησιμοποιούνται - για παράδειγμα, οι φοιτητές δεν γνωρίζουν τις διευθύνσεις blockchain των άλλων ή ακόμη και τις δικές τους στο πλαίσιο της εφαρμογής. Παρόλα

αυτά, η εγγενής διαφάνεια ενός δημόσιου blockchain σημαίνει ότι όλα τα δεδομένα συναλλαγών είναι ανοιχτά προσβάσιμα. Ένας χρήστης, με αρκετό χρόνο και πόρους, θα μπορούσε ενδεχομένως να αποκρυπτογραφήσει τη δομή του συστήματος αναλύοντας τα μοτίβα συναλλαγών σε έναν εξερευνητή blockchain. Αυτός ο τομέας αποτελεί το σημαντικότερο πεδίο βελτίωσης. Οι πιθανές βελτιώσεις θα μπορούσαν να περιλαμβάνουν την εφαρμογή πιο προηγμένων τεχνικών κρυπτογράφησης για την ενίσχυση της ιδιωτικότητας των δεδομένων ή τη χρήση μιας ιδιωτικής ή παράπλευρης λύσης αλυσίδας που είναι ειδικά προσαρμοσμένη για τη διαχείριση ευαίσθητων ακαδημαϊκών δεδομένων.

5.1.3 Σκέψεις περί Συγκεντρωτισμού

Δεδομένης της συγκεντρωτικής φύσης των πανεπιστημίων, η οποία περιλαμβάνει πολλαπλά επίπεδα εξουσίας, όπως πρυτάνεις, κοσμήτορες και καθηγητές, ένα πλήρως αποκεντρωμένο μοντέλο μπορεί να μην είναι κατάλληλο. Ορισμένοι ενδιαφερόμενοι φορείς απαιτούν έλεγχο του συστήματος για την αποτελεσματική διαχείριση και διακυβέρνηση. Σε απάντηση, επιλέξαμε την αποκέντρωση μόνο των στοιχείων που σχετίζονται με τα ακαδημαϊκά αρχεία των φοιτητών, τα οποία επωφελούνται από την αμετάβλητη φύση της τεχνολογίας Blockchain για την αποτροπή απώλειας και αλλοίωσης. Ωστόσο, άλλες πτυχές, ιδίως εκείνες που αφορούν τον διοικητικό έλεγχο, όπως ο ρόλος της γραμματείας, παρέμειναν συγκεντρωτικές.

5.2 Περιορισμοί

Κατά την ανάπτυξη του πρωτότυπου, έπρεπε να αποδεχτούμε κάποιους περιορισμούς που αντικατοπτρίζουν το πεδίο εφαρμογής και τις σχεδιαστικές επιλογές του συστήματος:

Ένας καθηγητής ανά μάθημα Το σύστημα είναι σχεδιασμένο να αντιστοιχίζει μόνο έναν καθηγητή ανά μάθημα. Αν και θεωρητικά είναι δυνατή η συμμετοχή πολλών καθηγητών, μόνο ο καθηγητής του οποίου το κλειδί είναι καταχωρημένο στο σχετικό έξυπνο συμβόλαιο μπορεί να προσθέσει βαθμούς.

Μοναδικό πρόγραμμα σπουδών Σε αντίθεση με τα σύγχρονα πανεπιστήμια που μπορεί να προσφέρουν πολλαπλά προγράμματα σπουδών προσαρμοσμένα σε διαφορετικές κατευθύνσεις ή έτη εγγραφής, το σύστημά μας υποστηρίζει μόνο ένα ενεργό πρόγραμμα σπουδών ανά πάσα στιγμή. Κατά συνέπεια, όλοι οι φοιτητές πρέπει να τηρούν το ίδιο πρόγραμμα σπουδών και τυχόν αλλαγές στο πρόγραμμα σπουδών τους επηρεάζουν όλους.

Μόνο τα υποχρεωτικά μαθήματα Η τρέχουσα υποδομή έχει σχεδιαστεί για τη διαχείριση ενός καθορισμένου προγράμματος σπουδών, όπου όλα τα μαθήματα είναι υποχρεωτικά για την απόκτηση πτυχίου. Αν και το σύστημα έχει τη δυνατότητα να περιέχει προαιρετικά μαθήματα και διάφορες κατευθύνσεις, αυτό απαιτεί πρόσθετες τροποποιήσεις στον κώδικα.

Αποθήκευση μόνο περασμένων μαθημάτων Για την ελαχιστοποίηση των φόρων συναλλαγών, το σύστημα αποθηκεύει στο blockchain μόνο τους βαθμούς των μαθημάτων που έχουν περάσει οι φοιτητές. Ωστόσο, ένα πλήρες αρχείο όλων των ακαδημαϊκών δραστηριοτήτων, συμπεριλαμβανομένων των μαθημάτων που δεν έχουν περαστεί, διατηρείται στην εφαρμογή ProofSpace, διασφαλίζοντας την ολοκληρωμένη τήρηση αρχείων εκτός αλυσίδας.

5.3 Μελλοντικές κατευθύνσεις

Κλείνοντας την παρούσα εργασία, προτείνουμε διάφορες κατευθύνσεις για πιθανές βελτιώσεις και επέκτασης της εφαρμογής, οι οποίες θα μπορούσαν να εμπνεύσουν μελλοντική έρευνα.

Βελτιωμένες ρυθμίσεις απορρήτου Ένας κρίσιμος τομέας για μελλοντική ανάπτυξη είναι η ενίσχυση των μέτρων προστασίας απορρήτου. Με την ενσωμάτωση προηγμένων κρυπτογραφικών τεχνικών, όπως zero knowledge proofs ή πιο εξελιγμένων μεθόδων κρυπτογράφησης, το σύστημα μπορεί να προστατεύσει καλύτερα την εμπιστευτικότητα των αρχείων των φοιτητών.

Αντιμετώπιση των περιορισμών Ένας ακόμα σημαντικός τομέας βελτίωσης αφορά την αντιμετώπιση των περιορισμών που περιγράφηκαν στην προηγούμενη ενότητα. Οι βελτιώσεις αυτές όχι μόνο θα ευθυγραμμίσουν το σύστημα με τις σύνθετες ανάγκες των σύγχρονων εκπαιδευτικών ιδρυμάτων, αλλά και θα διευρύνουν τις δυνατότητές του για ευρύτερη υιοθέτηση.

Έξυπνη βελτιστοποίηση συμβολαίων για επεκτασιμότητα Η τρέχουσα υλοποίηση αντιμετωπίζει προκλήσεις όσον αφορά την επεκτασιμότητα, ιδίως στη διαχείριση μεγάλου αριθμού φοιτητών λόγω των υπολογιστικών ορίων του Cardano. Η μελλοντική εργασία θα μπορούσε να επικεντρωθεί στη βελτιστοποίηση και την αναδιάρθρωση των έξυπνων συμβολέων, ώστε να μειωθεί η ανάγκη διαχωρισμού των συναλλαγών σε μικρότερα τμήματα.

Χρήση production-grade software Η περαιτέρω επαγγελματοποίηση της εφαρμογής, θα απαιτούσε μετάβαση από SQLite3 σε κάποιο άλλο σύστημα διαχείρισης βάσεων δεδομένων (DBMS) και αναβάθμιση από το Pug σε ένα πιο επεκτάσιμο frontend framework. Τέτοιες αναβαθμίσεις θα υποστήριζαν μεγαλύτερες βάσεις χρηστών και πιο σύνθετες αλληλεπιδράσεις δεδομένων, διευκολύνοντας την ομαλότερη και πιο αξιόπιστη απόδοση της εφαρμογής.

Χρήση εξατομικευμένων εικόνων στα NFT πτυχίου Στην υπάρχουσα έκδοση, όλα τα NFT Πτυχίου που κόβονται περιλαμβάνουν την ίδια εικόνα. Μια μελλοντική βελτίωση θα ήταν η ενσωμάτωση προηγμένων μεθόδων επεξεργασίας εικόνας για την αυτόματη εξατομίκευση αυτών των εικόνων με βάση τις προσωπικές πληροφορίες του κάθε φοιτητή. Επιπλέον, η χρήση μιας υπηρεσίας για το ανέβασμα των εικόνων στο IPFS και η συμπερίληψη του αντίστοιχου συνδέσμου στα μεταδεδομένα θα ενίσχυε περαιτέρω τη μοναδικότητα κάθε πτυχίου NFT.

Αυξημένη αποκέντρωση Τέλος, η αύξηση της αποκέντρωσης του συστήματος αποτελεί σημαντικό στόχο. Αυτό θα μπορούσε να επιτευχθεί με την αντικατάσταση του κλειδιού διαχειριστή με ένα σύστημα πολλαπλών υπογραφών, κατανέμοντας έτσι τον έλεγχο σε περισσότερα άτομα. Εναλλακτικά, ορισμένες διοικητικές λειτουργίες θα μπορούσαν να αυτοματοποιηθούν και να διαχειρίζονται από ένα έξυπνο συμβόλαιο, μειώνοντας περαιτέρω την ανάγκη για κεντρικό έλεγχο και ενισχύοντας τη διαφάνεια και την αμεροληψία κρίσιμων διαδικασιών, όπως η ανάθεση μαθημάτων και η έκδοση πτυχίων.

Κεφάλαιο 6

Βιβλιογραφία

[1] Agarkar, A. A. et al. "Blockchain Aware Decentralized Identity Management and Access Control System". In: *Measurement: Sensors* 31 (2024), p. 101032. DOI: https://doi.org/10.1016/j.measen.2024.101032.

[2] Alizadeh, M., Andersson, K., and Schelén, O. "Comparative Analysis of Decentralized Identity Approaches". In: *IEEE Access* 10 (2022), pp. 92273–92283. DOI: 10.1109/ACCESS.2022.3202553.

[3] Bentov, I., Gabizon, A., and Mizrahi, A. *Cryptocurrencies without Proof of Work*. 2017. arXiv: 1406.5694 [cs.CR].

[4] Buterin, V. *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. Accessed: 2024-03-04. 2013.

[5] Buterin, V. *On Public and Private Blockchains*. Research & Development. Aug. 2015.

[6] Cardano Foundation. *Cardano Documentation*. 2024.

[7] Cardano Foundation. *Cardano Website*. 2024.

[8] Casino, F., Dasaklis, T., and Patsakis, C. "A Systematic Literature Review of Blockchain-Based Applications: Current Status, Classification and Open Issues". In: *Telematics and Informatics* 36 (Nov. 2018). DOI: 10.1016/j.tele.2018.11.006.

[9] Chakravarty, M. M. T. et al. "The Extended UTXO Model". In: *Financial Cryptography and Data Security: FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2020, pp. 525–539. ISBN: 978-3-030-54454-6. DOI: 10.1007/978-3-030-54455-3_37.

[10] Christidis, K. and Devetsikiotis, M. "Blockchains and Smart Contracts for the Internet of Things". In: *IEEE Access* 4 (2016), pp. 2292–2303. DOI: 10.1109/ACCESS.2016.2566339.

[11] Čučko, Š. and Turkanović, M. "Decentralized and Self-Sovereign Identity: Systematic Mapping Study". In: *IEEE Access* 9 (2021), pp. 139009–139027. DOI: 10.1109/ACCESS.2021.3117588.

[12] Delgado-Segura, S. et al. "Analysis of the Bitcoin UTXO Set". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2019), pp. 78–91. ISSN: 0302-9743. DOI: 10.1007/978-3-662-58820-8_6.

[13] Dib, O. and Toumi, K. "Decentralized Identity Systems: Architecture, Challenges, Solutions and Future Directions". In: *Annals of Emerging Technologies in Computing (AETiC)* 4.5 (Dec. 2020), pp. 19–40. ISSN: 2516-0281. DOI: 10.33166/AETiC.2020.05.002.

[14] Do, T., Nguyen, T., and Pham, H. *Delegated Proof of Reputation: A Novel Blockchain Consensus*. 2019. arXiv: 1912.04065 [cs.CR].

[15] Garay, J., Kiayias, A., and Leonardos, N. *The Bitcoin Backbone Protocol: Analysis and Applications*. Cryptology ePrint Archive, Paper 2014/765. 2014.

[16] Global, I. O. *Atala PRISM: A Self-Sovereign Identity Platform*. 2024.

[17] Horizen Academy. *UTXO vs. Account Model*. 2023.

[18] Mysten Labs Team. *The Sui Smart Contracts Platform*. 2021.

[19] Nakamoto, S. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: *metzdowd.com* (2008).

[20] nhsz. *Consensus Mechanisms*. 2023.

[21] nhsz. *Proof-of-Stake (PoS)*. 2023.

[22] nhsz. *Proof-of-Work (PoW)*. 2023.
[23] SpaceBudz. *Lucid: Cardano Off-Chain Framework*. 2024.
[24] Szabo, N. *Smart Contracts*. 1994.
[25] Team, P. *ProofSpace: Decentralized Identity*. 2024.
[26] Tschorsch, F. and Scheuermann, B. "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies". In: *IEEE Communications Surveys & Tutorials* 18.3 (2016), pp. 2084–2123. DOI: 10.1109/COMST.2016.2535718.
[27] TxPipe. *Aiken: A Modern Smart Contract Platform for Cardano*. 2024.
[28] Wood, G. et al. "Ethereum: A Secure Decentralised Generalised Transaction Ledger". In: *Ethereum Project Yellow Paper* 151 (2014), pp. 1–32.
[29] World Wide Web Consortium (W3C). *Decentralized Identifiers (DIDs) V1.0, Core Architecture, Data Model, and Representations*. 2022.
[30] Yakovenko, A. *Solana: A New Architecture for a High Performance Blockchain*. 2021.

Κεφάλαιο 7

Παράρτημα

A Κώδικας Έξυπνων Συμβολαίων

Σε αυτή την ενότητα, παρέχουμε τον κώδικα για τα on-chain στοιχεία της εφαρμογής, γραμμένο στη γλώσσα προγραμματισμού Aiken.

A.1 Student Validator

```
use aiken/transaction.{ScriptContext}

validator(_student: ByteArray) {
  fn spend(_d: Void, _r: Void, _ctx: ScriptContext) {
    False
  }
}
```

A.2 Subject Validator

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/interval.{Finite}
use aiken/list
use aiken/time.{PosixTime}
use aiken/transaction.{
  InlineDatum, Input, Output, ScriptContext, Spend, Transaction, ValidityRange,
}
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId, Value}

type Datum {
  teacher: VerificationKeyHash,
  registrations: List<StudentId>,
  register_until: PosixTime,
  reset: Bool,
}

type Action {
  Register
  Update
  Grade
}

type StudentId =
  ByteArray

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>
```

```
30
31 validator(
32   policy: PolicyId,
33   asset_name: AssetName,
34   secreta: VerificationKeyHash,
35   registration_policy: PolicyId,
36 ) {
37   fn spend(datum: Datum, redeemer: Action, ctx: ScriptContext) {
38     expect Spend(my_output_reference) = ctx.purpose
39
40     let Transaction { inputs, outputs, reference_inputs, .. } = ctx.transaction
41     expect Some(self_input) =
42       inputs
43         |> transaction.find_input(my_output_reference)
44
45     let self_output = get_self_output(outputs, self_input.output.address)
46     expect InlineDatum(d) = self_output.datum
47     expect Datum { teacher, registrations, register_until, reset }: Datum = d
48     when redeemer is {
49       // Can only register a new student. Everything has to remain unchanged, except for the
      registrations list
50       Register -> and {
51           datum.reset == False,
52           is_period_valid(ctx.transaction.validity_range, datum.register_until),
53           signed_by(ctx.transaction, secreta),
54           teacher == datum.teacher,
55           reset == datum.reset,
56           register_until == datum.register_until,
57           are_equal_except_last(datum.registrations, registrations),
58           same_values(self_input.output.value, self_output.value),
59         }
60       // Can only update teacher or extend registration period
61       Update -> and {
62           signed_by(ctx.transaction, secreta),
63           reset == datum.reset,
64           datum.registrations == registrations,
65           same_values(self_input.output.value, self_output.value),
66         }
67       Grade -> and {
68           signed_by(ctx.transaction, datum.teacher),
69           if datum.reset {
70             and {
71               list.is_empty(registrations),
72               reset == False,
73               register_until == datum.register_until,
74             }
75           } else {
76             and {
77               registrations == datum.registrations,
78               register_until == datum.register_until,
79               reset == True,
80             }
81           },
82           validate_grade_outputs(
83             outputs,
84             reference_inputs,
85             self_input.output,
86             self_output,
87             datum.registrations,
88             policy,
89             asset_name,
90             registration_policy,
91           ),
92         }
93     }
94   }
95 }
96
97 /// Validate outputs
98 fn validate_grade_outputs(
```

```
  outputs: List<Output>,
  ref_inputs: List<Input>,
  self_input: Output,
  self_output: Output,
  registrations: List<ByteArray>,
  grade_policy: PolicyId,
  grade_asset: AssetName,
  registration_policy: PolicyId,
) -> Bool {
  let grades =
    list.filter_map(
      outputs,
      fn(o) {
        let grade = value.quantity_of(o.value, grade_policy, grade_asset)
        if and {
          grade >= 5,
          grade <= 10,
          o != self_output,
        } {
          expect Some(student_ref_input) =
            list.find(ref_inputs, fn(i) { i.output.address == o.address })
          expect [(student, _)] =
            student_ref_input.output.value
              |> value.tokens(registration_policy)
              |> dict.to_list()

          expect list.has(registrations, student)
          Some(grade)
        } else if grade != 0 {
          // if the grade is not 0, but neither in [5,10] range, then it has to be the
    self_output
          expect o == self_output
          Some(grade)
        } else {
          None
        }
      },
    )

  list.foldr(grades, 0, fn(sum, total) { sum + total }) == value.quantity_of(
    self_input.value,
    grade_policy,
    grade_asset,
  )
}

/// Get the output that has the same address as this validator
fn get_self_output(outputs: List<Output>, self: Address) -> Output {
  expect [self_output] = list.filter(outputs, fn(o) { o.address == self })
  self_output
}

/// Check if 2 Value objects are identical, except for the lovelace amount
fn same_values(input_values: Value, output_values: Value) -> Bool {
  list.is_empty(
    list.difference(
      value.flatten(value.without_lovelace(input_values)),
      value.flatten(value.without_lovelace(output_values)),
    ),
  )
}

/// Check if signer[0] == vk
fn signed_by(transaction: Transaction, vk: VerificationKeyHash) -> Bool {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}

/// Checks if 2 arrays are equal except for the last element of the second array
fn are_equal_except_last(input_list: List<a>, output_list: List<a>) -> Bool {
  expect Some(output_) = list.init(output_list)
```

```
  input_list == output_
}

/// Check if registration period has not expired
fn is_period_valid(range: ValidityRange, target_time: PosixTime) -> Bool {
  when range.lower_bound.bound_type is {
    Finite(tx_earliest_time) -> tx_earliest_time <= target_time
    _ -> False
  }
}
```

A.3 Policy Contract Validator

```
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{Output, ScriptContext, Spend, Transaction}
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId, Value}

type Datum {
  subjects: List<GradeToken>,
}

type GradeToken {
  policy: PolicyId,
  name: AssetName,
}

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

validator(secreta: VerificationKeyHash) {
  fn change_subjects(_datum: Datum, _redeemer: Void, ctx: ScriptContext) -> Bool {
    expect Spend(my_output_reference) = ctx.purpose

    let Transaction { inputs, outputs, .. } = ctx.transaction
    expect Some(self_input) =
      inputs
        |> transaction.find_input(my_output_reference)

    let self_output = get_self_output(outputs, self_input.output.address)
    and {
      signed_by(ctx.transaction, secreta),
      same_values(self_input.output.value, self_output.value),
    }
  }
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) -> Bool {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}

fn get_self_output(outputs: List<Output>, self: Address) -> Output {
  expect [self_output] = list.filter(outputs, fn(o) { o.address == self })
  self_output
}

fn same_values(input_values: Value, output_values: Value) -> Bool {
  list.is_empty(
    list.difference(
      value.flatten(value.without_lovelace(input_values)),
      value.flatten(value.without_lovelace(output_values)),
    ),
  )
}
```

A.4 Grade Token Minting Policy

```
use aiken/dict
use aiken/list
use aiken/transaction.{OutputReference, ScriptContext, Transaction} as tx
use aiken/transaction/value

validator(token_name: ByteArray, utxo_ref: OutputReference) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { inputs, mint, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    and {
      asset_name == token_name,
      if amount > 0 {
        list.any(inputs, fn(input) { input.output_reference == utxo_ref })
      } else {
        True
      },
    }
  }
}
```

A.5 Registration Token Minting Policy

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{ScriptContext, Transaction} as tx
use aiken/transaction/credential.{VerificationKey}
use aiken/transaction/value

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

validator(secreta: VerificationKeyHash) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { mint, .. } = transaction
    expect [(_, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    if amount > 0 {
      signed_by(ctx.transaction, secreta)
    } else {
      True
    }
  }
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}
```

A.6 Validity Token Minting Policy

```
use aiken/dict
use aiken/list
use aiken/transaction.{OutputReference, ScriptContext, Transaction} as tx
use aiken/transaction/value
```

```
validator(utxo_ref: OutputReference) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { inputs, mint, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    and {
      list.any(inputs, fn(input) { input.output_reference == utxo_ref }),
      asset_name == #"56616c6964697479",
      amount == 1,
    }
  }
}
```

A.7 Degree NFT Minting Policy

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{InlineDatum, Input, ScriptContext, Transaction} as tx
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId}

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

type RefDatum {
  subjects: List<GradeToken>,
}

type GradeToken {
  policy: PolicyId,
  name: AssetName,
}

validator(
  register_policy: PolicyId,
  validity_policy: PolicyId,
  secreta: VerificationKeyHash,
) {
  fn policy(_: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { mint, reference_inputs, outputs, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    expect Some(student_input) =
      list.find(
        reference_inputs,
        fn(i) {
          value.quantity_of(i.output.value, register_policy, asset_name) == 1
        },
      )

    let student_address = student_input.output.address
    expect Some(ref_input) =
      list.find(
        reference_inputs,
        fn(i) {
```

```
          value.quantity_of(
            i.output.value,
            validity_policy,
            #"56616c6964697479",
          ) == 1
        },
      )

    expect InlineDatum(ref_data) = ref_input.output.datum
    expect RefDatum { subjects }: RefDatum = ref_data

    and {
      signed_by(ctx.transaction, secreta),
      amount == 2,
      validate_grades(subjects, reference_inputs, student_address),
      list.any(
        outputs,
        fn(o) {
          o.address == student_address && value.quantity_of(
            o.value,
            policy_id,
            asset_name,
          ) == 1
        },
      ),
    }
  }
}

fn validate_grades(
  policies: List<GradeToken>,
  grade_inputs: List<Input>,
  student_address: Address,
) -> Bool {
  let values =
    list.foldr(
      grade_inputs,
      value.zero(),
      fn(i, v) {
        if i.output.address == student_address {
          value.merge(v, i.output.value)
        } else {
          v
        }
      },
    )

  list.all(policies, fn(p) { value.quantity_of(values, p.policy, p.name) >= 5 })
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}
```

B Κώδικας έκδοσης πιστοποιητικών

Σε αυτή την ενότητα, παρέχουμε τον πλήρη κώδικα που χρησιμοποιείται για την έκδοση επαληθεύσιμων πιστοποιητικών με κλήση του ProofSpace API από ένα περιβάλλον Node.js.

```
const crypto = require('crypto');
const axios = require('axios');

const url = "https://platform.proofspace.id/service-backend/v1/service/
    WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued";
const serviceDid = "WwDZ3sgsvnYsSvofQ6zjT5";
const credentialId = "WwDZ3sgsvnYsSvofQ6zjT5:3:CL:721:tag";
const schemaId = "WwDZ3sgsvnYsSvofQ6zjT5:2:University Registration:1.1";
const userDid = "JRpxFRCH84RrQcptCzF3PN";

const private_key = `-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAtsEaCBfGaRhWfkLuiKvK8InKgZyl0od1DcNAlVi6YaEmEruS
e8lPJfewXR3uxIg3SYOAGI6LYFFIJxAMcvHYEm0axW5lo4ov1hYRMA76tl/eo1Kp
3Cp7Ai8W9GshcQzbFwPX9UjbijHkRkwCIpW+iK/aipj/nBg1LJsOR4b9hpbB92Hz
+b4nviGBD5CA0vzbyDhLq/4oQbyOvAsAFAqKZEqBHq4s+WcKDHXFkvsmxnzexAlH
JtGjhFU+B3myXPMoMEHUInoKW0cSez5/WmkUOrPcNu4jmVmpjnVek+7s9q8F+blA
RaI4WKUUM+MA3sBGkMt3QeyKOm0L9JfA5C3q0ZmJ4xkJvaK0zCdml8DbgKasIy/L
JMXkQZyOZGlg2YsjcLK3lsElefAQ8d4/J4wSF+I/0uVKV/vxzVBx9kKgpUGJQu9l
0pGsxaQyZ94wUBiJybZqx8EUQmLIH7ONjSAGFHEISPLOlqRihJRz3zVdIKn04k1I
ftqk477mz5YJD5zVq9w9qx77oCzfH34N8pU/vTUaFmoWHhdnrb8slXaATOMejksA
kCK/wqUhYnGol8cNq92uTC/IdcEbBkHyHyQOHV88R1myoqY9bg4CbZzJDrtCzAUH
mN5fVCUVh7WL9Pt5pskD3zPPv77gxzJYLI+/9kOgwCXHrk9kGnFwgjsJHcUCAwEA
AQKCAgABsPbLNWEeLt36DCeF90UJBdVsbqUwNhWOu2ZZG3maFrIyAS9hWuMTIGAu
9Gx1wCb21LM2RvduPEJVLjZXR/qubPkO868YHNrFee3Nyo2ya1+1ljnBMllyEzp8
p5MnJy3N4Di66eIyPLT2m5R3enhJboswtirDaIvxbjvrGIo3GSURY4Qs/wACJ/A4
UyeGDFYCW9cDb/YyaYrOnbFx/gj61Z+tR4JeCoCSzfI2ooFg29355XYVm2IOgvVw
fOVRy+RG0g0jBosa/iTjJBhsn0VUj9qNXZXRwP0rNtXZu5/QxpzhC0BXhKFN3LU7
p7Gl3hSpGWDwqL48vS6bpF3iHrSqGN+d+WtNGCMWSNcq9b8Xp0ft7kx3XHMKYU/s
2+HtMO3jp0fxr4i7tG/DZpaahLA/HEIWG4AK5CEKOL0UDYtqwyoNedoPjhkr5nxf
Kx4BX40b2DG83hU5imsp2zP7XAkmiIpNYNdXosFrKoWcBU3vCKus9/MGS34ZNOR4
9LhRA1I478qfsUXcmx1PcmIRyyobanZsyzxad9n/yxDxCZ50nnxCMvJUWSRU9n8l
3dwafbgMV/ispxQXc9oTof61LCFVlrukjNYIcz+03V31q2/tp+dL2UNZWObT72ED
Aw8SPy6yB5BYpNU1BdC+jPmfDuDlnUPNSY8hBqV6otlXNIDCnwKCAQEA9GctrVnB
pZHRMwPZfL9VK7AiWkLdoWThtmuNHRZm0XhepYH8xaxlx6myQtkx3QYdJc1gpORC
BEoP8DtDdfOAZ4LGjic1GbXyHpI8bdgGf4H11EXZPr+PlAGLSyW8Ie+CYHfrKpYL
uHLaOIrSx4k+Fbj4B4obNEg/s4QSzQichI+UiwGmMuYx5KBlBXHnBWdN0+ICwbJq
Q3XRffVi0Xc0wDJjESXyVdqzMVnb18bHRkjwxTx2tgFZEBVYLeWtX3YTeQkzP/Yx
xIvzzXVtTrZa/DyD/bXs9ea9oJ0q80ABmyWUESTFrzrbYnAQG2UWYyPb6gv2Lw1D
1w5zTb5soZL0xwKCAQEAv20Pu8TfUABm1Imo/IsDA0koNB/0BpIEyfAeFHQ0UPR6
3rzli90Dn96PBfmlyOTPQir1RnSxpcxQW6eyU1ZGKasLFoS41mrnFTvbdOof/2Vd
kbNcwFb3wnLjm1YP4ImP36ugM4z4n++y03Uqbxz5wAsawq+5U+M0Z4miUrNRsPjz
OPBCl25VLXo3wFvP8qhIToDHnO5GNUBPCWZttlgi7Lwu8C/BRLGFSfIBAU7HFLIh
3foeLh6u+I4F1/ew8KMkrHxnQxDzMI8HaVtjwGtNdZmyXwf4M4ekJfGMowbPLh+k
3Ud68i2KPAmR2k50uS+gt/RfBYlqxqGCs22urK51EwKCAQAhaQVugCmft6pQfgM6
MXG0MjTP//oKOR8977/vrQNLOVg+6eEjKKkVPVZ4GvbdDxvm8gd3Ixj/Oxyoq4FS
W4I/JDp+bk+s/xllInqOTYy924c+iHnlQQygoAFpuXPZc8Nv6pTEUpSxl0nLlvav
fkx1bb7CURfc8s96e+CSFM5ItNQKmKQeqydA6MhGkUaSj4etImVcE0bBiBamUGg8
2D3m1DyEuJpDME10vqL2AV92dTwsylI0GUUSlMhkJdSoixyoXCCcasxs4PDoMlhJ
/VW8y40Dnt4Q04UysqmPbT2K9v0hxtQbW0FoWIdrJ0uDLADBIvd5gSe3+2q5wwFY
qR07AoIBAQCQ9qh5i8eJMCSvQ2iDh0pKzLXzc1c7Oo0B2CMPXkjYHayGzQoQdtjt
0n8BCWNw1UXXV+zS5EA7zJP2NPKohacwmuUUsoG/bsl8nQ2+ZibBJCRN2ESfK70I
tjSDSBNR+RrJSZFSOKl3hPApjosKBilkM4SadrSkQGVJG0Of2Ug9+ulF7zyerOE4
RVu/Akhh/Op0+LUxtIfaf87a0YS4F9qq6H+rSJ7VxJGpw+ja8qt0kslu7qC4UGg0
tmjaeiq2sPVu4e7ZLRbdaRM1yA2shBXuvuwDZ2egrlGyok/cjaWLyk7xe9YxXCdY
++5GIi/mpZjSde8XpE8GCP2Pn7ot7xedAoIBAQDEnlbgWPS2tldLoVFG4ega9x09
G/SVen5kkmLRaBNLV6rGRnCQoYkosUdbKMUsTb8WJGJhIllu5u0SgbaYyxrG63ob
FdDgVjIPJneL1xnX3s6qP9084k2cBqwE3LtNfVK6eWYEwXvaeHQN/Ur3KbJQ3tDi
LkNpzd4ysqRmy01UOIQxWG0nnM+LDRjhmong8u+HAeiYmZ9WZ2D06nd5dbq19FeT
F0dJYc4jhBZQoW7oaaNAMJ4JY/ViM10ISzjwaWcanQ0B1pH0yk0oSReWd82BFQfb
sLC/A+iBdA4dNfB9fnB+zrF0Php+2mpMZOnNyo5Nvm2N08wVOYk5lho0psEj
-----END RSA PRIVATE KEY-----
`;

const credentialValues = {
    credentialId,
```

```
    schemaId,
    fields: [
        { name: "School", value: "ECE NTUA" },
        { name: "StudentID", value: "el17108" },
    ],
    utcIssuedAt: new Date().getTime(),
    revoked: false
};

const request = {
    serviceDid,
    subscriberConnectDid: userDid,
    subscriberEventId: "",
    credentials: [credentialValues],
    issuedAt: new Date().getTime(),
};

const headers = {
    'Content-Type': 'application/json'
};

const binaryBody = Buffer.from(JSON.stringify(request), "utf-8");

const key = crypto.createPrivateKey({ key: private_key });
const signature = crypto.sign('sha3-256', binaryBody, key);

headers['X-Body-Signature'] = signature.toString('base64');

axios.post(
    url,
    binaryBody,
    {
        headers: headers
    }
).then((reply) => {
    return reply.data;
}).catch((err) => {
    throw err;
});
```

National Technical University of Athens
School of Electrical and Computer Engineering
Computer Science Laboratory

Student Authentication and Certificate Issuing System on the Cardano Blockchain

Diploma Thesis

ANDREAS MANTZOUTAS

Supervisor: Nectarios Koziris
Professor NTUA

Athens, July 2024

National Technical University of Athens
School of Electrical and Computer Engineering
Division of Computer Science
Computer Science Laboratory

Student Authentication and Certificate Issuing System on the Cardano Blockchain

Diploma Thesis

ANDREAS MANTZOUTAS

Supervisor: Nectarios Koziris
Professor NTUA

Approved by the examination committee on July 15th, 2024.

.....	.....	.....
Nectarios Koziris	Aggelos Kiayias	Aris Pagourtzis
Professor NTUA	Professor University of Edinburgh	Professor NTUA

Athens, July 2024

.....
Mantzoutas Andreas
Graduate in Electrical and Computer Engineering

Abstract

In recent years, blockchain technology has emerged as a transformative force in various sectors, offering a decentralized, tamper-proof ledger that enhances transparency, security, and efficiency. Originating with Satoshi Nakamoto's introduction of Bitcoin in 2008, blockchain has challenged traditional financial systems and now promises to revolutionize other industries, including academia. Traditional methods of managing academic records are plagued with challenges such as significant administrative burdens, error susceptibility due to human involvement, and risks of document forgery.

This thesis explores the application of blockchain technology to the management of academic records, proposing a system that leverages its inherent security and transparency to overhaul current practices. By embedding academic credentials on the blockchain, educational institutions can drastically reduce administrative overhead, eliminate manual errors, and secure records against tampering. Moreover, the immutable and public nature of blockchain ensures that verification of grades, certificates, and degrees is reliable, allowing instant validation by employers, other institutions, and agencies without intermediaries.

Focusing on a practical implementation, this study details the deployment of a blockchain-based system on the Cardano network to manage student authentication and certificate issuance. The proposed system utilizes smart contracts to automate the issuing and verification of academic credentials, ensuring that they are verifiable and immutable. Additionally, the system's architecture is designed to accommodate the specific needs of educational institutions, from registering student grades to issuing degrees, enhancing the integrity and efficiency of educational administrative processes. By adopting blockchain technology, educational institutions can shift their focus towards delivering quality education while ensuring that academic achievements are securely recorded and effortlessly verifiable, heralding a new era in the management of academic records.

Keywords — Blockchain, Smart Contracts, Cardano, Aiken, Decentralized Applications, Decentralized Identifiers, ProofSpace

Acknowledgements

With the delivery of this thesis, my journey as a student at the School of Electrical and Computer Engineering of the National Technical University of Athens comes to an end. Throughout this journey, many have stood by me, inspired and supported me.

I would especially like to thank Professor Nectarios Koziris, who gave me the opportunity to work on my thesis at the Computer Systems Laboratory, as well as PhD Ms. Katerina Doka, PhD Candidate Mr. Tasos Katsigiannis and Mr. Christos Palaskas for all the support and guidance they provided me throughout this research and journey.

I am grateful for the scholarship I received from IOG, which made it possible to use the ProofSpace software, and for the immediate help I received from the ProofSpace team whenever I needed it.

I couldn't help but thank my parents, Spyros and Angeliki, as well as the rest of my family members, Niki, Dimitris and Katerina, for their love and support.

I am lucky to have had my fellow students and good friends Giannis, Antonis and Thanos as my companions on this journey, whom I thank warmly for all the support and the moments we experienced during this trip.

Mantzoutas Andreas, July 2024

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, blockchain technology has surged in popularity, heralding a new era in digital innovation. This revolutionary technology offers a decentralized, tamper-proof ledger, solving problems that have long plagued various industries. By ensuring transparency, security, and efficiency, blockchain technology is reshaping sectors such as finance, supply chain management, and healthcare, facilitating seamless transactions, authenticating the provenance of goods, and securely managing digital identities.

The inception of blockchain can be traced back to the pseudonymous figure Satoshi Nakamoto, who in 2008 introduced Bitcoin, the first cryptocurrency, through the seminal whitepaper "Bitcoin: A Peer-to-Peer Electronic Cash System."[19]. Beyond presenting a new digital currency, Nakamoto unveiled the blockchain technology, a decentralized system enabling direct transactions without the need for intermediaries. This marked the beginning of a technological revolution, with Bitcoin and its foundational blockchain technology challenging traditional notions of currency and financial systems.

This thesis delves into the application of blockchain technology within the realm of academia, focusing on its potential to revolutionize student authentication and certificate issuing systems, and exploring the implementation of these systems on the Cardano network. By harnessing the power of the Cardano network, educational institutions can issue verifiable and immutable academic credentials, significantly enhancing the integrity and efficiency of the certification process. This thesis examines the technical mechanisms, benefits, and challenges of deploying a blockchain-based solution for student authentication and certificate issuance, marking a significant step towards the modernization of educational administrative processes.

1.1 Motivation

The management of academic records, from issuing grades to certificates and degrees, is a cornerstone of educational institutions' operations. However, this process is traditionally fraught with challenges, including the significant workload on administrative staff, the risk of errors due to human involvement, and the ever-present threat of document forgery.

By leveraging blockchain for the issuance and management of academic records, we can dramatically reduce the administrative burden, effectively eliminating errors attributed to manual processes. More importantly, blockchain's inherent security features ensure that once a record is added, it cannot be altered or tampered with, thus significantly lowering the risk of forgery. Furthermore, blockchain technology introduces an unprecedented level of transparency and ease of verification for academic credentials. Since records on the blockchain are public and immutable, verifying the authenticity of grades, certificates, and degrees becomes a straightforward process. Employers, other educational institutions, and verification agencies can instantly confirm the validity of an academic record without the need for intermediaries or time-consuming verification processes.

Our study delves into how implementing a blockchain-based system for academic record management can revolutionize the sector. It proposes a system where the security, transparency, and efficiency of blockchain

technology are harnessed to streamline the issuance, management, and verification of academic credentials. This approach promises a future where educational institutions can focus more on delivering quality education and less on administrative logistics, where the authenticity of academic achievements is easily verifiable, and where the value of academic credentials is unequivocally upheld.

1.2 Thesis Structure

In Chapter 2 of this thesis, we explore the theoretical background of blockchain technology, detailing its basic components, and the concept of decentralized identifiers. Last but not least, we provide a focused examination of the Cardano network, which underpins our application.

Chapter 3 shifts to discuss the system architecture of our proposed implementation, explaining its components and relevant use cases.

Following this, Chapter 4 delves into the detailed implementation of the system, covering everything from the smart contracts on the Cardano blockchain to the integration of the front-end and back-end systems.

Finally, Chapter 5 reflects on the challenges encountered during development, discusses the lessons learned, and outlines potential future enhancements.

Chapter 2

Theoretical Part

In this section, we delve into the essential theoretical foundations necessary for understanding our study's focus. Our discussion is fundamentally anchored in blockchain technology, focusing on its principles, mechanics, and theoretical underpinnings. Additionally, the concept of decentralized IDs (DIDs) will be discussed, highlighting their role in providing secure and verifiable digital identities. We will also provide an overview of Cardano, the blockchain platform powering our implementation, detailing our reasons for selecting it. Understanding these technologies and their interplay is vital for appreciating the potential of a blockchain-based system for managing academic records. This theoretical groundwork will equip readers with the knowledge needed to navigate the intricacies of implementing such a system effectively.

2.1 Blockchain Fundamentals

2.1.1 Main Components

2.1.1.1 Transaction

A transaction in blockchain terminology is a digital record representing the transfer of assets or data between parties. Transactions are signed digitally to ensure authenticity and integrity, awaiting validation and inclusion in a block by miners or validators within the network.

2.1.1.2 Block

A block within a blockchain is a fundamental data structure that encapsulates a collection of transactions, which have been validated by the network's participants. Alongside the transaction data, it includes a block header, containing relevant metadata [29]. Each block is cryptographically linked to its predecessor through this header, creating an immutable chain. This linkage ensures the integrity and continuity of the transaction history, making each block a crucial component of the blockchain's structure and security.

2.1.1.3 Node

A node refers to any computer that connects to the blockchain network. Nodes participate in the blockchain protocol by verifying transactions and blocks, with full nodes additionally enforcing the network's rules and maintaining a complete copy of the blockchain ledger.

2.1.1.4 Miner

A miner is a type of node that performs the computational work required to add new blocks to the blockchain. Miners validate transactions and compete to solve cryptographic puzzles in a process known as mining, which secures the network and earns them transaction fees and new coins as rewards.

2.1.1.5 Blockchain

A blockchain is a distributed, append-only timestamped data structure that facilitates a decentralized peer-to-peer network. This innovative technology allows participants, who may not necessarily trust one another, to carry out transactions and interactions without the need for a centralized authority. At the heart of a blockchain are signed transactions between peers, which represent agreements or the exchange of assets. These transactions are grouped into blocks and verified by nodes. This ensures the integrity of the blockchain and prevents fraudulent activities, such as double-spending. The fundamental principle that allows all participants to agree on the transaction record is consensus, a process that is essential to the blockchain's operation. Through this architecture, blockchains enable secure, transparent, and tamper-proof record-keeping, supporting a wide range of applications beyond simple transactions[8].

2.1.2 Blockchain Network Types

Blockchain technology has given rise to diverse network types, each designed to cater to specific needs regarding governance, privacy, and participation. Scholarly work identifies three main categories: public, private, and federated blockchains. Public blockchains are open and permissionless, allowing anyone to participate in mining and transaction verification. Private blockchains restrict access to a select group of users, enhancing privacy and efficiency by limiting participation. Federated blockchains merge these models, operated by a consortium that selects nodes for governance, combining the scalability of private systems with a level of decentralized control. This streamlined classification underscores the versatility of blockchain technology, from promoting transparency in public networks to safeguarding sensitive data in private and federated systems[8].

2.1.2.1 Public

A public blockchain is a blockchain that anyone in the world can read, anyone in the world can send transactions to and expect to see them included if they are valid, and anyone in the world can participate in the consensus process - the process for determining what blocks get added to the chain and what the current state is. As a substitute for centralized or quasi-centralized trust, public blockchains are secured by cryptoeconomics - the combination of economic incentives and cryptographic verification using mechanisms such as proof of work or proof of stake, following a general principle that the degree to which someone can have an influence in the consensus process is proportional to the quantity of economic resources that they can bring to bear. These blockchains are generally considered to be "fully decentralized"[5].

2.1.2.2 Consortium

A consortium blockchain is a blockchain where the consensus process is controlled by a pre-selected set of nodes; for example, one might imagine a consortium of 15 financial institutions, each of which operates a node and of which 10 must sign every block in order for the block to be valid. The right to read the blockchain may be public, or restricted to the participants, and there are also hybrid routes such as the root hashes of the blocks being public together with an API that allows members of the public to make a limited number of queries and get back cryptographic proofs of some parts of the blockchain state. These blockchains may be considered "partially decentralized"[5].

2.1.2.3 Private

A fully private blockchain is a blockchain where write permissions are kept centralized to one organization. Read permissions may be public or restricted to an arbitrary extent. Likely applications include database management, auditing, etc internal to a single company, and so public readability may not be necessary in many cases at all, though in other cases public auditability is desired[5].

2.1.3 Blockchain Models

In the landscape of blockchain technology, two models emerge for managing transactions and state on distributed ledgers: the Account Model and the Unspent Transaction Output (UTXO) Model. These models represent fundamentally different approaches to recording and validating transactions within a blockchain network. The Account Model, employed by Ethereum, operates on a state-based system where each account's balance is directly updated. In contrast, the UTXO Model, utilized by Bitcoin, relies on tracking ownership through unspent outputs from previous transactions.

2.1.3.1 Account Based Transaction Model

The Account Model represents a method of managing digital assets within a blockchain network by conceptualizing them as balances in accounts, akin to how traditional bank accounts operate. In this model, state transitions are being direct transfers of value and information between accounts. Those are categorized into two primary types: those controlled by users through private keys and those governed by contract code, which are associated with smart contracts. When transactions occur, the system updates the global state of the blockchain. This global state serves as a ledger, recording the balances, storage, and executable code associated with each account[4].

In this model, transactions directly adjust the balances of the involved accounts. This approach allows for a straightforward mechanism for tracking asset ownership and movement across the network. The Account Model's reliance on a global state that records account balances and their attributes highlights its efficiency and suitability for a wide array of applications, from simple transfers to complex interactions facilitated by smart contracts[17].

2.1.3.2 Unspent Transaction Output Model

Unlike account-based systems, the Unspent Transaction Outputs (UTXO) Model does not track balances within accounts. Instead, it maintains a comprehensive record of all unspent outputs from past transactions. Each UTXO represents a slice of digital currency that has not been spent and can be used as an input for new transactions[12]. When a user initiates a transaction, they utilize one or more UTXOs as inputs, which the network verifies and then consumes to generate new UTXOs as outputs. This method ensures that each piece of digital currency can be traced back to its creation point.

The UTXO model's architecture inherently supports anonymity and privacy, as it complicates the tracing of funds by breaking down transaction histories into individual outputs rather than aggregating them under user accounts. Additionally, this structure allows for parallel processing of transactions, potentially improving the scalability and efficiency of the network. Each UTXO is independent, enabling multiple transactions to be processed simultaneously without interference.

2.1.3.3 Object Model

The Object Model, employed in the Sui Blockchain, is a distinctive approach within blockchain technology that merges elements from both the UTXO (Unspent Transaction Output) and account-based models. This hybrid model structures blockchain transactions and states in a way that aims to leverage the parallel processing advantages of UTXOs while maintaining the state coherence and ease of programming found in account-based systems. In practice, the Sui Object Model treats digital assets as unique objects that can be owned and managed directly by users or smart contracts, allowing for direct interaction and manipulation of these objects. This method streamlines transactions and smart contract executions, facilitating a more efficient and scalable blockchain infrastructure [18].

2.1.4 Consensus

Consensus, in the context of distributed systems such as blockchains, is the process through which a network of nodes arrives at a mutual agreement on a single state of the database or ledger, ensuring that all participants have a consistent view of the information. This mechanism is fundamental in preventing discrepancies and maintaining the integrity and reliability of the network. Simply put, consensus ensures that all nodes in a network agree on the validity and order of transactions, similar to reaching a unanimous decision on a movie to watch among a group of friends. Without consensus, the network would be susceptible to double-spending, forks, and other security vulnerabilities, leading to a lack of trust and potential failure of the system [20].

The Byzantine Agreement (BA) specifically addresses the issue of achieving consensus in the presence of faulty or malicious nodes, highlighting the importance of robust solutions that can operate effectively even in anonymous and synchronous settings without trusted setups. This problem underscores the complexity of reaching agreement in a distributed network where participants may not necessarily trust each other or may even act adversarially [15]. The consensus mechanism encompasses a variety of protocols, rewards, and punitive measures designed to encourage honest participation and deter malicious activities. In blockchain technologies, consensus mechanisms such as Proof of Work (PoW) and Proof of Stake (PoS) play pivotal roles. PoW, for instance, secures consensus through computationally intensive tasks, while PoS offers a more energy-efficient alternative, rewarding validators based on their stake in the network. These methods both facilitate agreement on the state of the ledger and enhance the network's security against attacks and manipulation [8][10][20].

2.1.4.1 Proof of Work

Proof of Work (PoW), originally implemented by Bitcoin and used by Ethereum in its initial phase, requires network participants, known as miners, to solve cryptographic puzzles to validate transactions and create new blocks. The complexity of these puzzles is adjusted dynamically to ensure consistency in the block creation rate. The first miner to solve the puzzle is rewarded with cryptocurrency, incentivizing participation and security within the network. This method effectively prevents double-spending and ensures that each transaction is confirmed and immutable once added to the blockchain [20][22].

The robustness of PoW comes from the computational effort - or "work" - required to solve the puzzles, making any attempt to alter the blockchain both technically challenging and economically unfeasible. This is because altering the blockchain to, for instance, spend the same coins twice, would require a malicious actor to outpace the honest network by consistently solving the puzzles first. Achieving this would necessitate control over more than 50% of the network's computational power, a feat that requires significant financial and energy resources, thereby securing the network against fraud and attacks. The consensus among network participants on the validity of transactions and the chronological order of blocks ensures a single, trusted version of the ledger, maintaining the blockchain's integrity and trustworthiness [27].

2.1.4.2 Proof of Stake

Proof of Stake (PoS) represents a consensus mechanism that aims to secure blockchain networks through a process where validators stake a certain amount of cryptocurrency as collateral to be allowed to participate in the block validation and creation process. In PoS, the chance to validate transactions and create new blocks is proportional to the amount of currency a validator has staked, rather than the computational work they can perform. Validators are chosen randomly to propose a block, and the validity of blocks is ensured by other validators who attest to the block's accuracy. This process not only minimizes energy consumption but also introduces a crypto-economic layer of security: validators have a financial incentive to act in the network's best interest, as malicious behavior could lead to the loss of their staked assets [20][21].

PoS addresses the double-spending problem by ensuring that validators have something of value at risk while they participate in the network governance. This stake acts as a deterrent against dishonest behavior, such as attempting to validate fraudulent transactions or creating forks in the chain. The decentralized nature of this system means that to compromise the network, an attacker would need to own a significant portion of the staked currency, making attacks prohibitively expensive. Unlike PoW, where security relies on external resources (electricity, mining hardware), PoS relies on the internal economic value within the blockchain, aligning validators' interests with the network's security and integrity. By shifting from resource depletion (in PoW) to a stake-based model, PoS offers a more energy-efficient framework for achieving consensus in distributed ledger technologies [3].

2.1.4.3 Delegated Proof of Stake

Delegated Proof of Stake (DPoS) is an evolution of the Proof of Stake (PoS) consensus mechanism. In DPoS, network participants vote to elect a small group of delegates or block producers, who are then responsible for validating transactions and creating new blocks. This voting process ensures that the power to influence the network's direction and security is democratized, allowing coin holders to have a say in who maintains the network. The delegates are incentivized to act in the best interest of the network through rewards distributed for block production and can be voted out if they fail to perform their duties adequately or act maliciously, introducing a layer of accountability not present in traditional PoS models [23]. By delegating the responsibility of transaction validation to a selected few, DPoS can significantly reduce the cost of network operation and maintenance, while still maintaining a degree of decentralization[14].

2.1.4.4 Proof of History

Proof of History (PoH), employed by the Solana blockchain, is a consensus mechanism that incorporates the passage of time into the blockchain's data structure, enabling a unique way to verify the sequence and timing of transactions. By assigning a timestamp to each transaction, PoH allows for the independent verification of the order of events without necessitating continuous communication between nodes. This approach significantly aids in solving the double-spending problem, as each transaction's unique timestamp prevents the same digital asset from being spent more than once[31].

2.1.5 Smart Contracts

Smart contracts, a concept initially introduced by Nick Szabo in 1994 [25], embody a significant leap in automating contractual agreements and minimizing the need for trusted intermediaries [10]. At their core, smart contracts are transaction protocols that execute the terms of a contract, converting traditional contractual clauses into programmable code embedded within digital or physical properties. This innovation aims to enforce contract terms autonomously, reducing the occurrence of exceptions, both malicious and accidental. Within the blockchain ecosystem, smart contracts function as scripts or applications stored on a blockchain, executing independently across every node in the network when triggered by transactions [10]. This autonomous execution ensures that contractual agreements are upheld transparently and efficiently, without the requirement for a central authority or intermediaries.

When a smart contract is deployed on a blockchain, it gains a unique address through which it can be triggered by transactions. This decentralized execution model enables smart contracts to manage data-driven interactions between parties on a network. For instance, a smart contract can facilitate a trade between parties by autonomously verifying and executing transactions based on predefined conditions, such

as exchanging digital assets under specific terms [10]. This process not only exemplifies the smart contract's ability to enforce agreements but also demonstrates its capacity to securely manage assets and state, acting as an autonomous agent within the blockchain ecosystem.

By automating enforcement and execution, smart contracts eliminate the need for intermediaries, reducing potential for fraud and lowering transaction costs. Furthermore, their immutable and transparent nature ensures that all parties have confidence in the execution and outcomes of contracts. This innovation extends beyond simple transactions, enabling complex interactions and fostering the development of decentralized autonomous organizations (DAOs) and new forms of digital interactions. As smart contracts continue to evolve, they promise to redefine the landscape of contractual agreements in the digital age, offering a secure, efficient, and transparent mechanism for executing and enforcing contracts [10][32].

2.1.5.1 Decentralized Applications

A decentralized application (DApp) is a software application that runs on a distributed system, typically a blockchain network, rather than on a single centralized server. Built on top of blockchain technology and utilizing smart contracts, DApps operate autonomously without the need for a central authority. By leveraging smart contracts, DApps enable direct, trustless interactions between users, supporting a variety of functions from financial services to gaming, while ensuring data integrity and system reliability.

2.2 Decentrialized IDs

In the digital age, our lives have become increasingly intertwined with the internet, leading to an explosion in the amount of personal data stored and processed online. This surge in digital activity has highlighted significant shortcomings in traditional Identity Management Systems (IDMS). Centralized systems, where a single entity controls vast amounts of data, have been the norm. Such arrangements raises privacy and security concerns, as evidenced by numerous data breaches targeting centralized servers responsible for storing users' identities and managing federated identity systems[1]. These incidents underline the inherent vulnerabilities of centralized IDMS, where users often lack control over their personal data and third-party sharing occurs without explicit consent.

The inadequacies of current IDMS have catalyzed interest in decentralized identity solutions, such as blockchain-based identity management. Decentralized Identifiers (DIDs) emerge as a promising alternative, offering a user-centric approach that grants individuals control over their own data, bypassing the need for central authorities or third-party intermediaries[1][2][13]. This paradigm shift towards Self-Sovereign Identity (SSI) leverages the core principles of the internet's decentralized architecture, aiming to restore control, security, and privacy back to the individual. The benefits of DIDs include enhanced privacy, increased security, and greater control over personal data, which align with the growing demand for more secure and user-empowered digital identity solutions.

2.2.1 Technical Foundations

The transition towards decentralized identity systems signifies a pivotal evolution from traditional models, emphasizing the urgent need for enhanced security, privacy, and user sovereignty. This section delves into the foundational elements and operational frameworks of decentralized identity systems.

2.2.1.1 Decentrialized IDentifiers

Decentralized Identifiers (DIDs) are identifiers for decentralized systems where users can have verified digital identities [30]. They are introduced to the concept of self-sovereign identity. A DID identifies any entity. These identifiers allow a DID controller to demonstrate control over it. They may be used without a centralized registry, identity provider, or certificate authority. DIDs are Uniform Resource Identifiers (URIs) that link a DID subject to a DID document. An example of a DID is did:example:123456abcdef[2]. Unique and persistent, DIDs allow entities - including individuals, organizations, devices, and applications - to authenticate digitally without centralized oversight. This innovation hands control of identity back to the users, facilitating digital autonomy away from the centralized control paradigm[2].

2.2.1.2 Verifiable Credentials

Verifiable Credentials (VCs), serving as the digital counterparts to physical identification documents such as passports or driving licenses, leverage cryptographic security to prevent unauthorized access and ensure the integrity of the credentials shared [13]. They enable the presentation of credentials in a secure, verifiable manner, with minimal exposure of personal information, thus allowing individuals greater control over the dissemination of their data [11][13].

2.2.1.3 Self-Sovereign Identity

Underpinning these advancements is the Self-Sovereign Identity (SSI) model, which places the individual at the heart of their identity management[2][11]. In this model, individuals have complete control over their identity data, from creation to storage and sharing. SSI, supported by the mechanisms of DIDs and VCs, promises a future where digital identities are both portable and privacy-centric[11].

2.2.1.4 Purpose

The decentralized identity framework aims to address and mitigate the challenges associated with traditional identity management systems, such as privacy concerns and the risk of data breaches, by eliminating unnecessary reliance on centralized or singular technologies. This innovative approach not only enhances user

control and privacy but also paves the way for a more secure digital ecosystem. As these technologies evolve, they are set to redefine digital interactions, fostering an environment of enhanced trust and security[2][11][13].

2.2.2 Use Cases

Decentralized Identifiers (DIDs) offer transformative solutions across multiple sectors by enabling secure, self-sovereign digital identities. In e-commerce, DIDs enhance transaction security and user privacy, while in healthcare, they ensure the confidential sharing of medical records. The education sector benefits from secure verification of academic credentials, and financial services see streamlined Know Your Customer (KYC) processes. Additionally, DIDs can support secure digital voting systems, promoting integrity and accessibility in civic engagement. Overall, DIDs are pivotal in advancing a more secure, efficient, and user-centric digital ecosystem[2][11][13].

2.2.3 DIDs and Blockchains

Blockchain technology fundamentally enhances the implementation of Decentralized Identifiers (DIDs) by providing a secure, immutable, and decentralized infrastructure essential for managing digital identities[13]. The technology's ability to operate without a central authority ensures that users can control their own identities, aligning with the self-sovereign nature of DIDs. By recording DIDs on a blockchain, each identity is permanently registered in a tamper-evident manner, guaranteeing the integrity and authenticity of digital identities.

Furthermore, blockchain not only securely stores DIDs but also facilitates the notarization and verification of associated credentials through smart contracts. These automated contracts streamline identity verification processes, making digital interactions more efficient and trustworthy[13]. The seamless integration of DIDs with blockchain technology thus establishes a robust framework for digital identity management, where security, privacy, and user autonomy are significantly enhanced.

2.3 Cardano

Cardano[7] stands as a pioneering third-generation proof-of-stake (PoS) blockchain platform. Born from a research-first approach, it is the brainchild of leading engineers and academic experts, crafted to address the trilemma of scalability, security, and decentralization[6]. At its core, Cardano employs the Ouroboros consensus algorithm, a PoS protocol designed to offer security guarantees on par with proof-of-work systems but with significantly higher energy efficiency[6]. The network also introduces the extended Unspent Transaction Output (eUTXO) model, an evolution of the traditional UTXO system used in blockchains like Bitcoin. This model enhances the ability to process transactions and smart contracts, positioning Cardano as a formidable platform for developing enterprise-level decentralized applications (dApps).

2.3.1 Extended UTXO

The Unspent Transaction Output (UTXO) model represents all unspent outputs from transactions as distinct UTXOs, each corresponding to a specific amount of cryptocurrency. It ensures transaction integrity by allowing each UTXO to be spent only once, thereby preventing double-spending and facilitating a transparent audit trail of asset transfers. However, while the UTXO model excels in tracking asset ownership and ensuring security, its basic form is limited in supporting more complex smart contracts and stateful applications that require nuanced transaction processing.

Addressing these limitations, the Extended UTXO (EUTXO) model enhances the traditional UTXO framework by introducing two significant improvements: the incorporation of arbitrary data (datum) within UTXOs and the capability for validators to ensure contract continuity across transactions. This adaptation allows for more sophisticated contract logic and stateful operations to be executed within the UTXO framework, expanding the potential for developing complex decentralized applications without forsaking the inherent benefits of UTXO's transparency and security [9].

2.3.1.1 Basic Components

In the Enhanced UTXO (EUTXO) model, three fundamental components play critical roles in the operation and validation of transactions: Datum, Redeemer, and Context. These components are key to understanding how stateful smart contracts and complex transaction logic can be implemented on a blockchain utilizing the EUTXO model.

Datum The Datum is a piece of data attached to a transaction output, serving as an additional argument during the validation process. It essentially allows a contract to maintain some state without altering its validator. This is particularly useful for carrying the state of state machines within the blockchain, facilitating more complex and interactive contract behavior while preserving the contract's immutable logic[9].

Redeemer The Redeemer is another critical component, representing an argument that is supplied by the spender of a UTXO. It acts as an input to the validation script, enabling the script to make decisions based on the provided information during the execution of a transaction. This allows for dynamic interactions with smart contracts, where the behavior of the contract can change based on the inputs provided by the users attempting to interact with it[9].

Context The Context provides detailed information about the transaction being validated, also passed as an additional argument to the validator. It includes data such as the transaction outputs, input being validated, and an index indicating the specific input under validation. This set of information empowers the validator to enforce more stringent conditions than possible under the basic UTXO model. Importantly, it facilitates the inspection of the current transaction's outputs, crucial for ensuring contract continuity. The Context ensures that the execution of scripts during validation is deterministic, allowing users to predict the outcome of script execution and resource consumption before submitting a transaction[9].

2.3.1.2 Key Benefits

The Extended UTXO (eUTXO) model introduces several advancements over the traditional UTXO model, making blockchain applications more versatile and powerful. Firstly, the eUTXO model allows transactions

to carry additional data or state (Datum), enabling the development of richer and more complex smart contracts that can manage and manipulate stateful information.

Secondly, the eUTXO model expands transaction validation rules through the use of a "Redeemer", providing flexibility to define complex conditions for transactions. This allows for the creation of smart contract logic that goes beyond simple asset transfers, opening up possibilities for more sophisticated transactional interactions within the blockchain ecosystem.

Lastly, the eUTXO model ensures contract continuity by making sure that a contract's rules are consistently applied across a series of transactions. This is achieved by incorporating the "Context" component, which provides detailed information about the transaction being validated.

Aspect	**UTXO**	**eUTXO**
Data Handling	Does not support additional data with transactions.	Supports additional data through Datum, enabling stateful smart contracts.
Transaction Rules	Basic validation rules focused on ownership.	Advanced validation with Redeemer, allowing complex and conditional transaction logic.

Table 2.1: Comparison of UTXO and eUTXO Models

2.3.2 Cardano Smart Contracts

Plutus is Cardano's native smart contract platform. It stands as a Turing-complete system, ensuring that smart contracts developed with Plutus are both secure and efficient. With its roots in Haskell, Plutus brings a high degree of reliability and correctness to smart contract execution. Plutus contracts are divided into two main parts: on-chain code that executes on the blockchain and off-chain code that runs on the user's device. The integration of these components allows for the development of sophisticated applications on the Cardano blockchain, leveraging the Plutus Application Framework for off-chain code and the Plutus compiler for converting on-chain code into Plutus Core[6].

As already discussed, smart contracts in Plutus are designed to operate with both on-chain and off-chain components. Initially, Plutus smart contracts were primarily written in Haskell, allowing for a unified approach to both segments of a dApp. However, the ecosystem has evolved to support a broader range of programming options. For on-chain code, languages like Aiken[28] have emerged, while off-chain interactions can now leverage JavaScript libraries such as Lucid[24].

Plutus scripts can be divided into two main categories: Validator scripts and Minting policies.

2.3.2.1 Validator Scripts

In Plutus, a Validator script is a component that determines the conditions under which a specific Unspent Transaction Output (UTXO) can be spent, adding programmability to the UTXO model. The script takes three arguments: the Datum, the Redeemer, and the Context. The Datum represents state information attached to the UTXO, the Redeemer is provided by the spender to satisfy the conditions of the script, and the Context contains information about the transaction attempting the spend. Together, these components enable developers to define complex spending conditions for UTXOs, allowing for the creation of smart contracts on the Cardano blockchain.

2.3.2.2 Minting Policies

Minting policies represent the second category of Plutus scripts on the Cardano blockchain, setting the rules for the creation (minting) or destruction (burning) of tokens. Unlike Validator scripts, minting policies require two arguments to operate: the Redeemer and the Context. The Redeemer provides specific conditions or actions requested by the user, such as minting or burning tokens, while the Context supplies information about the transaction in which the operation is taking place. Through these arguments, minting policies define the circumstances under which tokens can be introduced or removed from circulation, ensuring that token

management is carried out securely and in accordance with the predefined rules established by the script. This mechanism grants developers the ability to implement controlled and programmable token economies on Cardano's blockchain platform.

2.3.3 DIDs on Cardano

2.3.3.1 Atala PRISM

Atala PRISM is a self-sovereign identity (SSI) platform and service suite for verifiable data and digital identity. Built on Cardano, it offers core infrastructure for issuing DIDs (decentralized identifiers) and verifiable credentials, alongside tools and frameworks to help expand your ecosystem [16].

2.3.3.2 ProofSpace

ProofSpace is a platform designed to revolutionize the verification of digital identities on the internet[26]. It simplifies the adoption of secure, user-managed identity solutions through a no-code platform that supports verifiable credentials. Among other technologies, ProofSpace utilizes IOG's Atala PRISM to enable Decentralized Identifiers (DIDs) and Verifiable Credentials on the Cardano blockchain, making it a core component of this thesis.

Chapter 3

Methodology and System Architecture

3.1 Overview of the dApp

3.1.1 Design

The DApp utilizes a system architecture in which each academic subject is administered through a specific smart contract. Grades in this system are tokenized, meaning that each subject is assigned a grade token. The amount of tokens a student possesses indicates their grade for that particular subject. Student profiles are structured as smart contracts to provide the secure administration of these tokens, hence prohibiting unlawful transfers.

Degrees are sent in the form of Non-Fungible Tokens (NFTs), which include specific information about a student's grades in various disciplines. The system integrates ProofSpace for decentralized identity verification, enhancing the security and legitimacy of the academic credentials granted.

3.1.2 Stakeholders

The system supports three kinds of actors:

- Students use their university credentials to access the system, where they can check their grades, enroll in courses and semesters, and mint their degree upon finishing their studies.
- Teachers must have a Cardano account connected to a wallet. This allows them to sign transactions in order to grade students. They transfer the necessary grade tokens to the student addresses throughout the grading period.
- Secratary mints registration tokens, manages policy data, and starts and ends registration periods as well as updates the subjects needed to complete a degree.

3.1.3 System Components

Components of the system:

1. Smart Contracts: The primary function of the DApp is powered by smart contracts written in Aiken, a programming language similar to Rust, which is used for constructing Plutus smart contracts on the Cardano blockchain. There exist seven key smart contracts:
   - Four minting policy contracts for the issuance of grade tokens, degree NFTs, registration tokens, and a validity token.
   - Three validator contracts that manage student profiles, subject operations, and policy data respectively. These contracts ensure that all interactions within the system are governed by immutable, predefined rules, ensuring consistent enforcement of the system's protocols.
2. Frontend: The Pug templating engine is used to provide unique views for students, professors, and the secretaries in the user interface. This configuration not only enables the presentation of data but also oversees the offchain grading procedures in which teachers submit student grades.
3. Backend: This component serves as the central hub for all off-chain code operations, functioning as a single API built using the Express framework. This API consists of several parts: the first part incorporates the lucid-cardano library to perform Cardano-specific tasks such as managing transactions, locating UTXOs, and communicating with the network. The second part is responsible for invoking ProofSpace to issue Credentials. The persistence part manages a SQLite3 database to fulfill any additional data needs. Additionally, the backend renders the frontend views for stakeholders using the Pug templating engine. Apart from these specific parts, the API also includes a general-purpose functionality to handle various other tasks and facilitate communication between different components.
4. ProofSpace: ProofSpace is an essential external component that plays a critical role in issuing verifiable credentials and decentralized IDs. In addition, it is compatible with the ProofSpace mobile application, which enables students to receive real-time notifications regarding new grades and other significant updates. This feature enhances the interactive experience and ensures that students stay updated promptly.

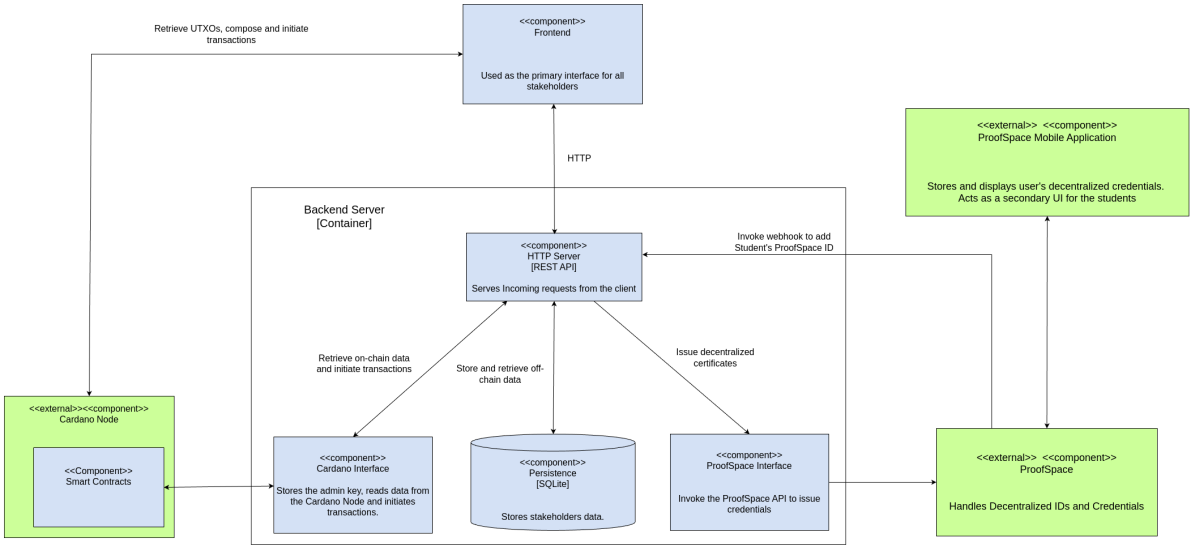


Figure 3.1.1: Component diagram

3.2 Use Cases

In this section, we will explore the use cases that guided the design of our application, detailing the expected functionalities and how they are envisioned to operate. Accompanying UML sequence diagrams are included to visually represent these interactions and workflows, providing a clearer understanding of the system's architecture and the processes it supports.

3.2.1 Use Case 1: University Registration

3.2.1.1 Description

The process begins after a student has been registered off-chain by the university secretary, who creates an entry in the database with the student's information, such as name, ID, email, and a temporary password. The blockchain component of the registration process starts when the student first logs into the system using their academic credentials. Upon login, the student is presented with a QR code on the DApp interface, which they need to scan with their mobile device using the ProofSpace app. This action connects their ProofSpace ID with their academic profile. The connection is completed when ProofSpace notifies the app's backend via a webhook, resulting in the student's ProofSpace ID being stored in the database for future credential issuance. The backend then proceeds to generate a unique blockchain address for the student. This address is crafted using the student validator script, parameterized with the student's ID, resulting in a unique address. The backend mints a university registration token, named after the student's ID, using the registration token minting policy, and transfers it to the student's address. Lastly, the backend calls the ProofSpace API to issue a university registration certificate for the student.

3.2.1.2 UML Sequence Diagram

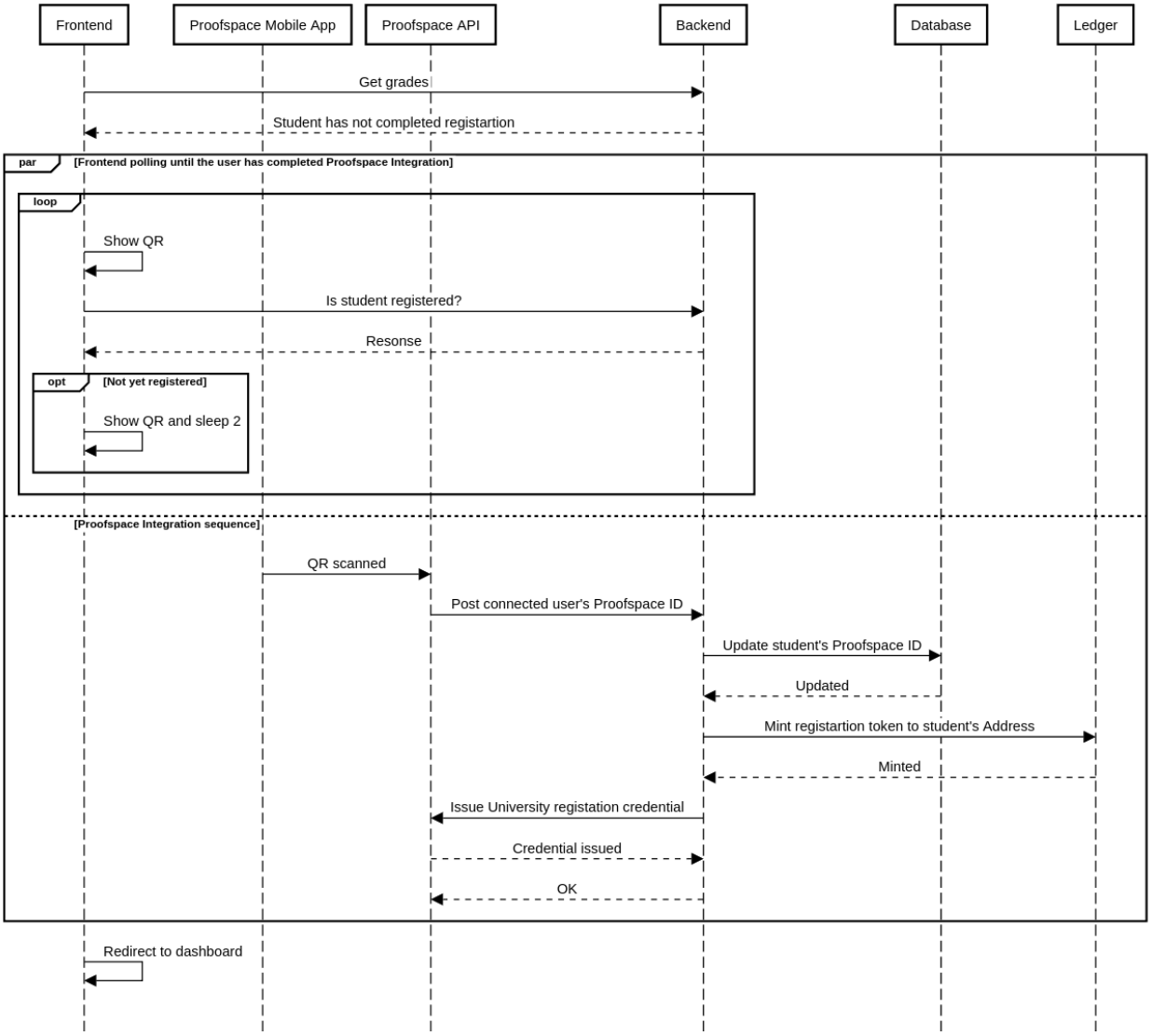


Figure 3.2.1: University registration sequence diagram

3.2.2 Use Case 2: View Grades

3.2.2.1 Description

Upon logging in, the student is directed to the grade dashboard on the frontend interface of the DApp. This triggers a backend operation that retrieves the student's grades from the Cardano Ledger. It first queries the blockchain to retrieve all Unspent Transaction Outputs (UTXOs) associated with this student script's address. These UTXOs hold the tokens that represent the student's grades. Next, the backend interacts with the policy contract to fetch a list of all valid grade tokens that the system recognizes. With this list, it identifies which of the UTXOs contain grade tokens and determines the corresponding subjects for each token. If there are multiple grades for the same subject, the system selects the highest. Once the relevant data is compiled, it is sent back to the frontend, where it is displayed on the student's grade dashboard.

3.2.2.2 UML Sequence Diagram

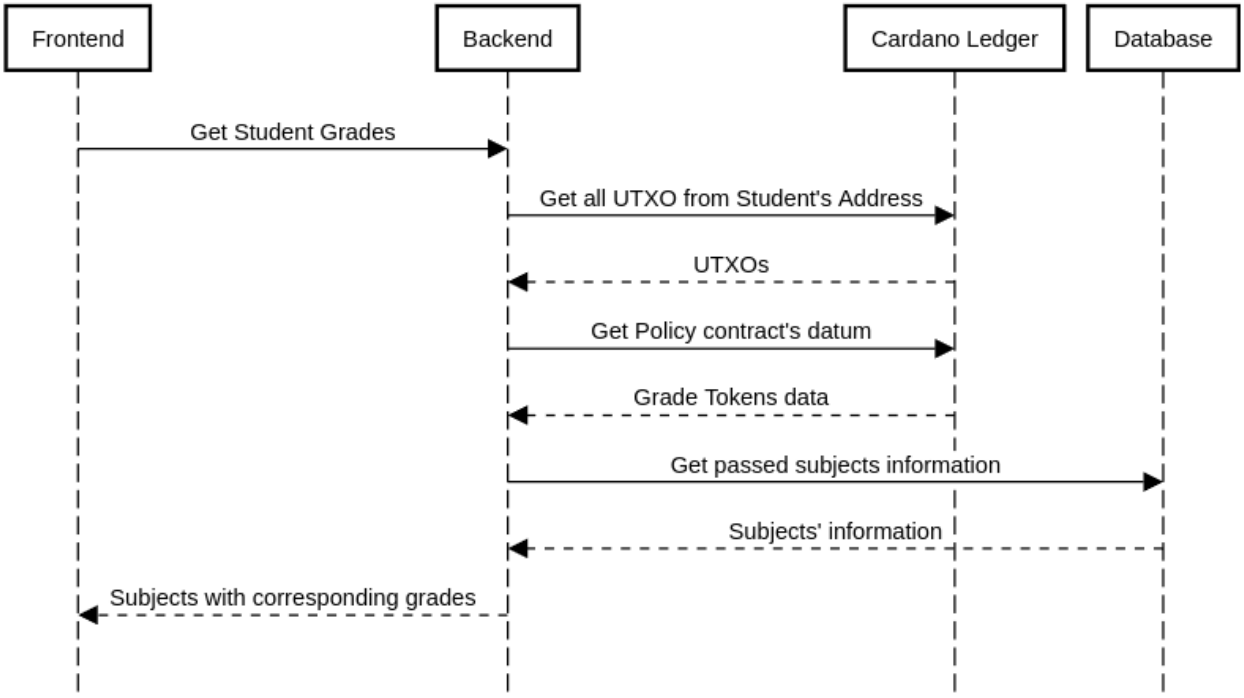


Figure 3.2.2: View grades sequence diagram

3.2.3 Use Case 3: Semester Enrollment

3.2.3.1 Description

The semester enrollment consists of two parts, the retrieval of the available subjects and the subject enrollment. When a student logs into the dApp and navigates to the "register" page, the frontend initiates a backend request to retrieve a list of available subjects for enrollment. The backend determines the valid subjects based on the semester type (odd or even) corresponding with the student's enrollment semester, and ensures that the subjects are of a semester number less than or equal to the student's current semester. For example, a student enrolling in the 6th semester can only enroll in subjects offered during the 2nd, 4th, and 6th semesters. It then checks all UTXOs from the student's Cardano address against valid grade token policies from the policy contract to ascertain previously passed subjects. The backend filters out these passed subjects from the list of valid ones and returns the remaining subjects to the frontend, providing the student with the subjects they are eligible to enroll in. After viewing the available subjects, the student selects the subjects they wish to enroll in and submits their choices. This triggers the backend to process each selection by making a smart contract call to add the student's ID to the "registration" array in the datum for each selected subject. Concurrently, a registration token indicative of the student's current semester number is minted to the student's address to validate registration and prevent duplicate registrations. Additionally, the backend interfaces with the ProofSpace API to issue a semester registration certificate and individual subject registration certificates for each subject the student has enrolled in.

3.2.3.2 UML Sequence Diagram

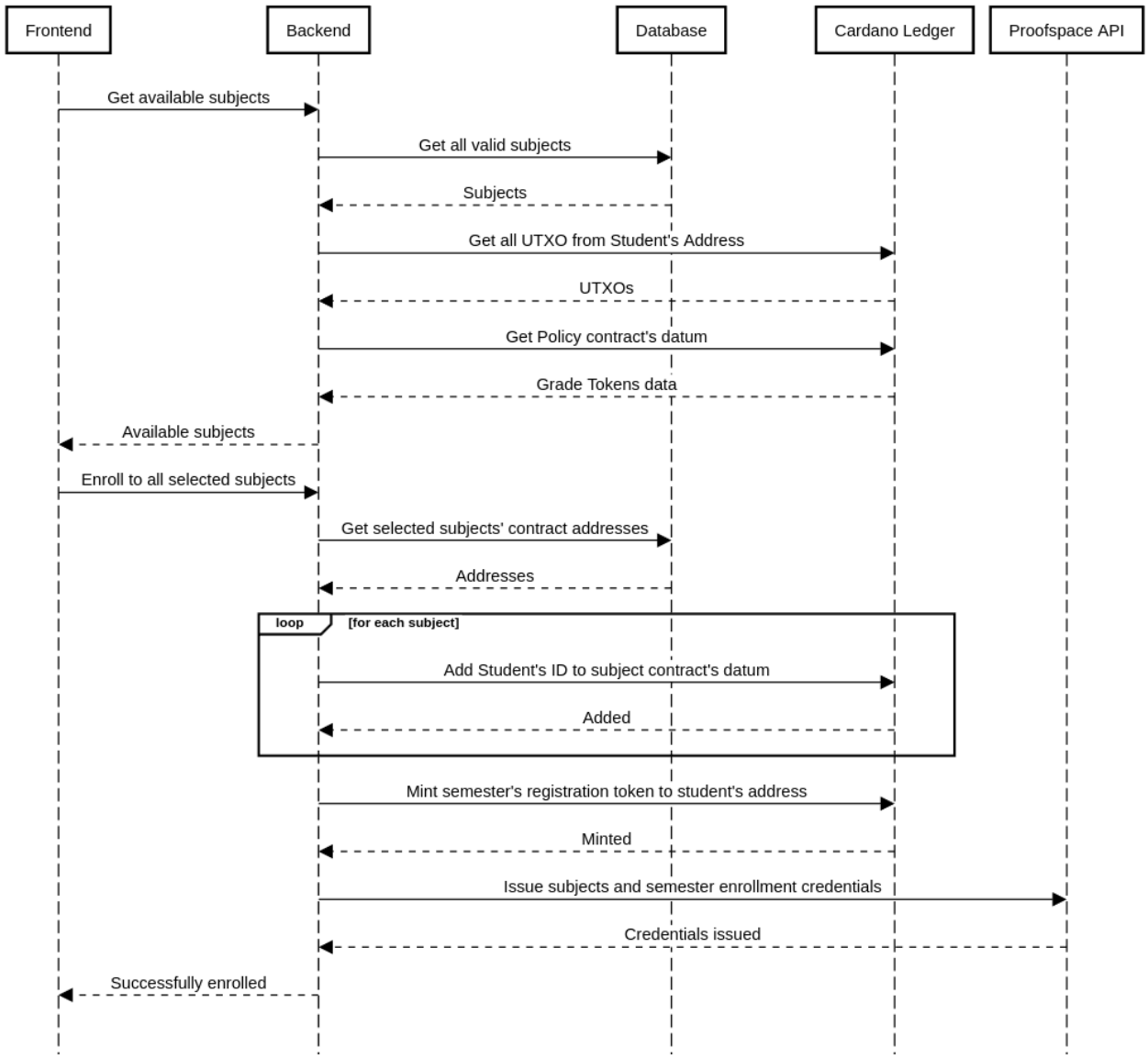


Figure 3.2.3: Semester enrollment sequence diagram

3.2.4 Use Case 4: Degree Minting

3.2.4.1 Description

The process for a student to claim their degree is initiated once they are deemed eligible, which is determined by the system confirming that all required subjects have been passed. This eligibility is verified both off-chain and on-chain to ensure compliance with academic requirements. When a student accesses the main dashboard and their grades are retrieved, the system automatically checks their academic status. If the student has successfully passed all necessary subjects, a button appears prompting them to claim their degree. Clicking this button opens a modal asking the student to connect their Nami wallet. Once the student connects their Nami wallet, the active address is sent to the backend through the relevant endpoint designed to handle the degree minting process. The backend then calculates the average grade across all passed subjects to compute the final degree grade. A minting transaction is initiated, which creates the degree as a Non-Fungible Token (NFT) and transfers it to the real student's address provided by the Nami wallet -not the internal address used by the system - and the degree grade is added as metadata. Following the successful minting and transfer of the degree NFT, the backend makes a call to the ProofSpace API to mint the digital degree certificate for the student.

3.2.4.2 UML Sequence Diagram

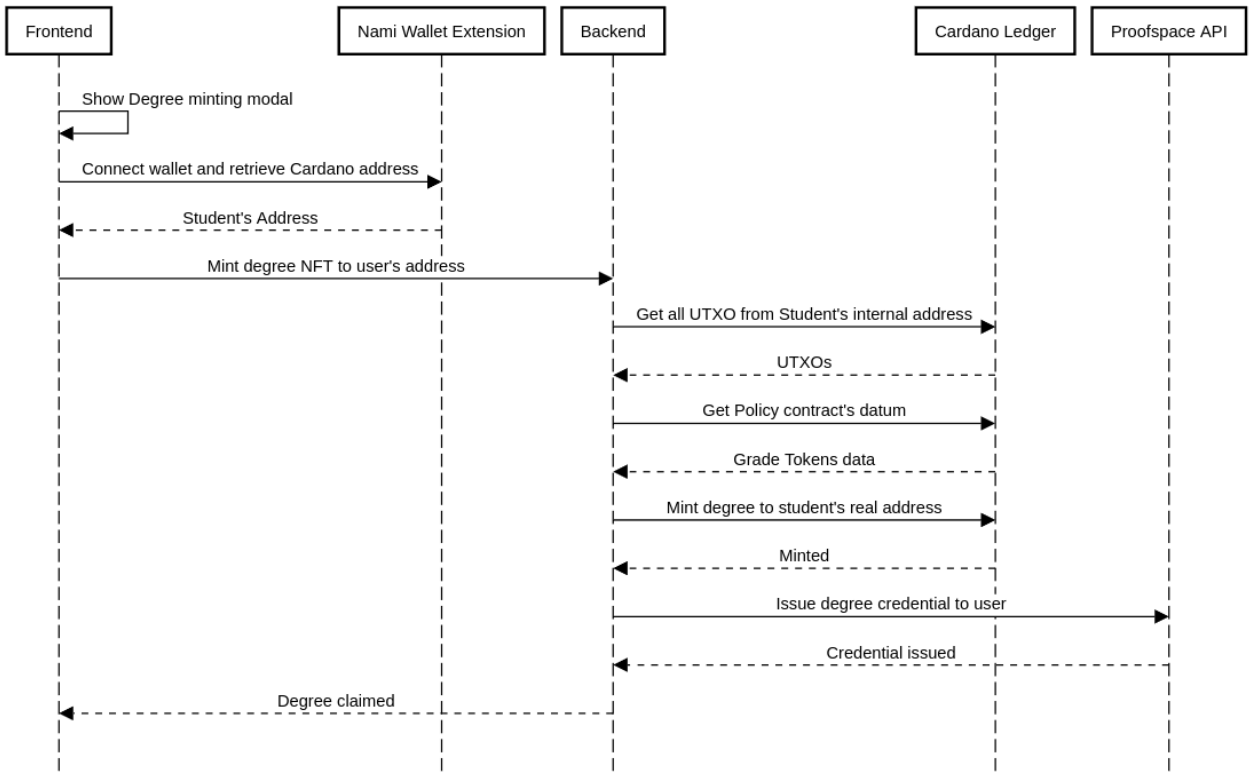


Figure 3.2.4: Degree minting sequence diagram

3.2.5 Use Case 5: Add Grades

3.2.5.1 Description

The process begins when a teacher selects the subject for which they want to add grades. Once a subject is selected, the backend queries the relevant subject smart contract to retrieve the registrations array from its datum. This array contains the IDs of students registered for the subject. Using these IDs, the backend then accesses the off-chain database to fetch the first name and last name of each registered student. The backend compiles and returns this data (ID, first name, last name) to the teacher, providing a clear list of students eligible to receive grades. The teacher can then input grades for each student directly in the interface provided. Once the grades are entered, the teacher submits them by pressing the submit button on the frontend. This action triggers a direct smart contract call from the frontend to the subject's smart contract, which transfers the appropriate grade tokens to each student's address based on the submissioned data. The teacher is prompted to sign the transaction to authenticate and finalize the grading. Following a successful transaction, the backend initiates a call to the ProofSpace API to issue credentials for the subject to each student. These credentials, containing the subject's name and the student's grade, are issued to all students regardless of their grade score, ensuring that every student receives a record of their participation and performance. The issuance of these credentials serves to notify students of their new grades and provides them with verifiable documentation of their academic achievement.

3.2.5.2 UML Sequence Diagram

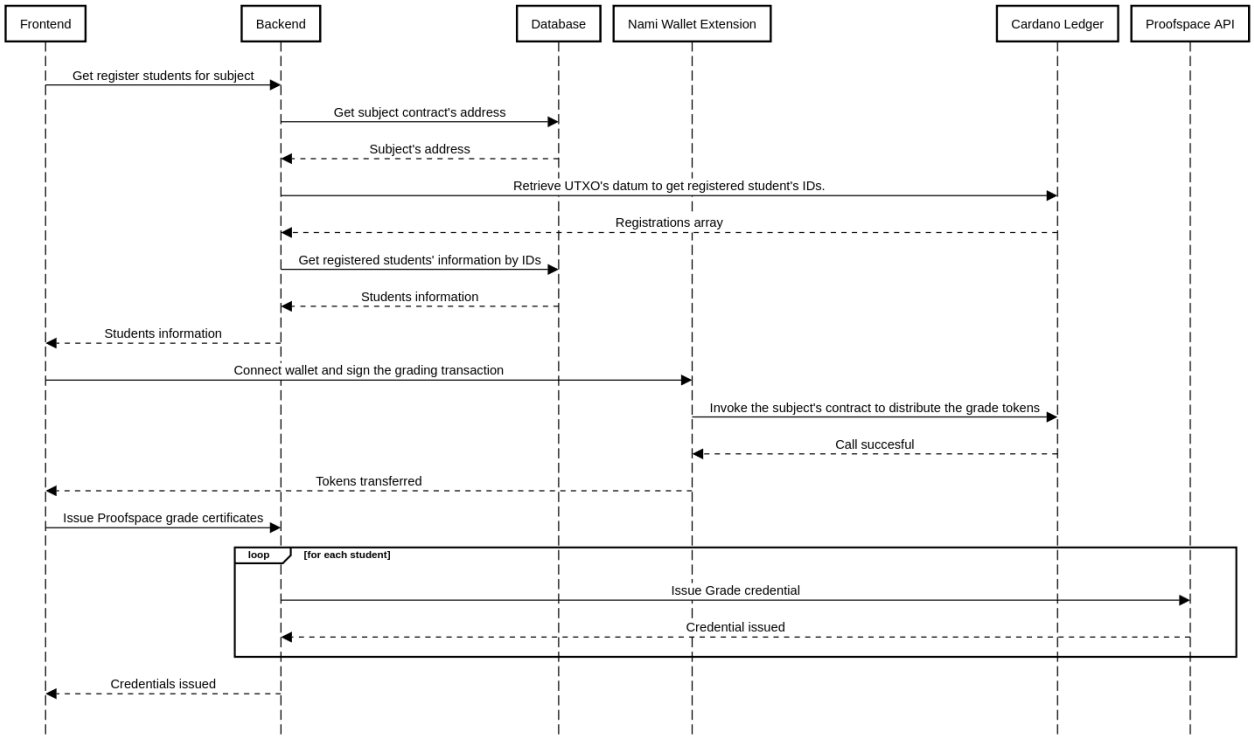


Figure 3.2.5: Adding grades sequence diagram

3.2.6 Use Case 6: Start Registration Period

3.2.6.1 Description

The process begins when the secretary decides to open registration for the upcoming academic period. The secretary enters an end date for the registration period through the frontend. The registration period commences at the exact date and time that the secretary provides this end date. Once the end date is entered, the frontend interacts with the backend by invoking the relevant endpoint. The backend then calls the smart contracts (script) associated with each subject to update their datum with the new end date. With this update, students can now begin enrolling for these subjects and the system has been modified to consider the time period between the present date and the stated end date as active for registration.

3.2.6.2 UML Sequence Diagram

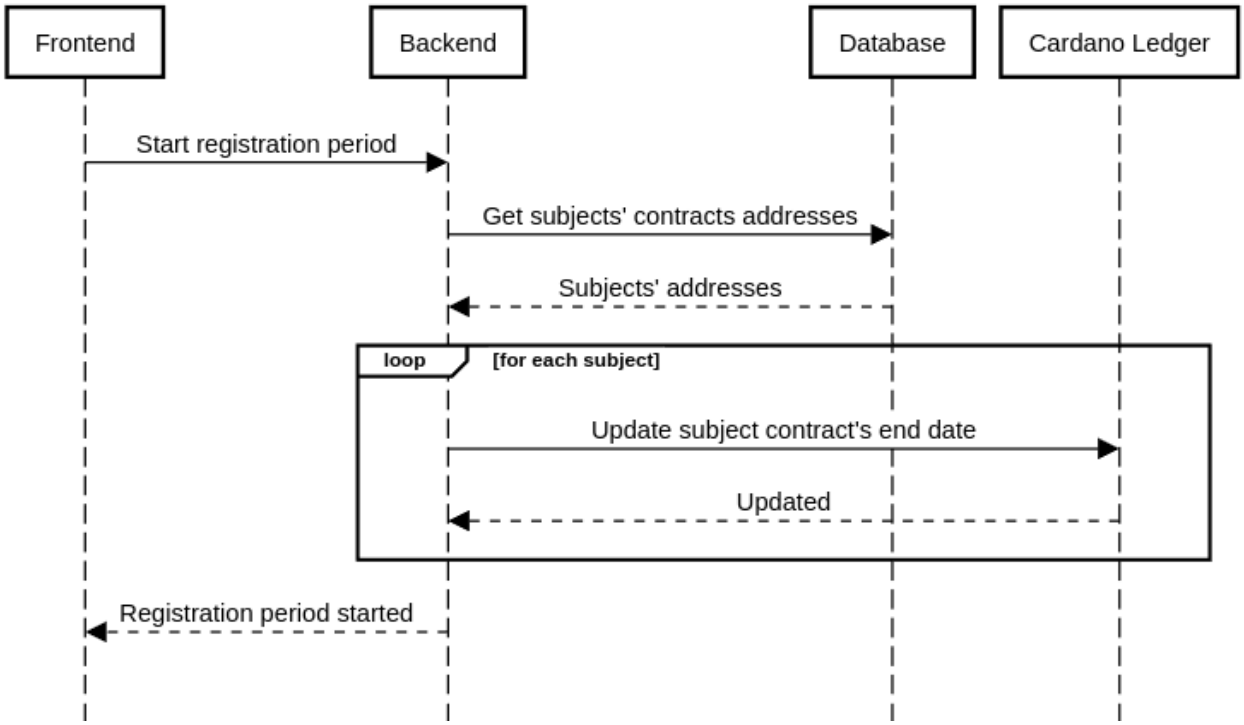


Figure 3.2.6: Start registration period sequence diagram

3.2.7 Use Case 7: Curriculum Change

3.2.7.1 Description

This case starts when a curriculum change necessitates that the secretary modify the list of subjects that are required for students to earn their degrees. From the administrative interface of the DApp, the secretary has access to a list of all subjects that are currently registered in the off-chain database. The secretary selects which ones will be required going forward and submits this selection to a backend endpoint dedicated to handling changes in degree requirements. Upon receiving the selection, the backend function first retrieves the information about the grade tokens for the selected subjects, namely their minting policy IDs and names. Once the data about the grade tokens is collected, the backend makes a call to the policy contract responsible for defining degree requirements. This contract call involves updating the datum within the policy contract with the new policies that now include the updated list of required subjects. The modification of this datum ensures that all future degree verifications will be based on the new set of requirements, reflecting the curriculum change.

3.2.7.2 UML Sequence Diagram

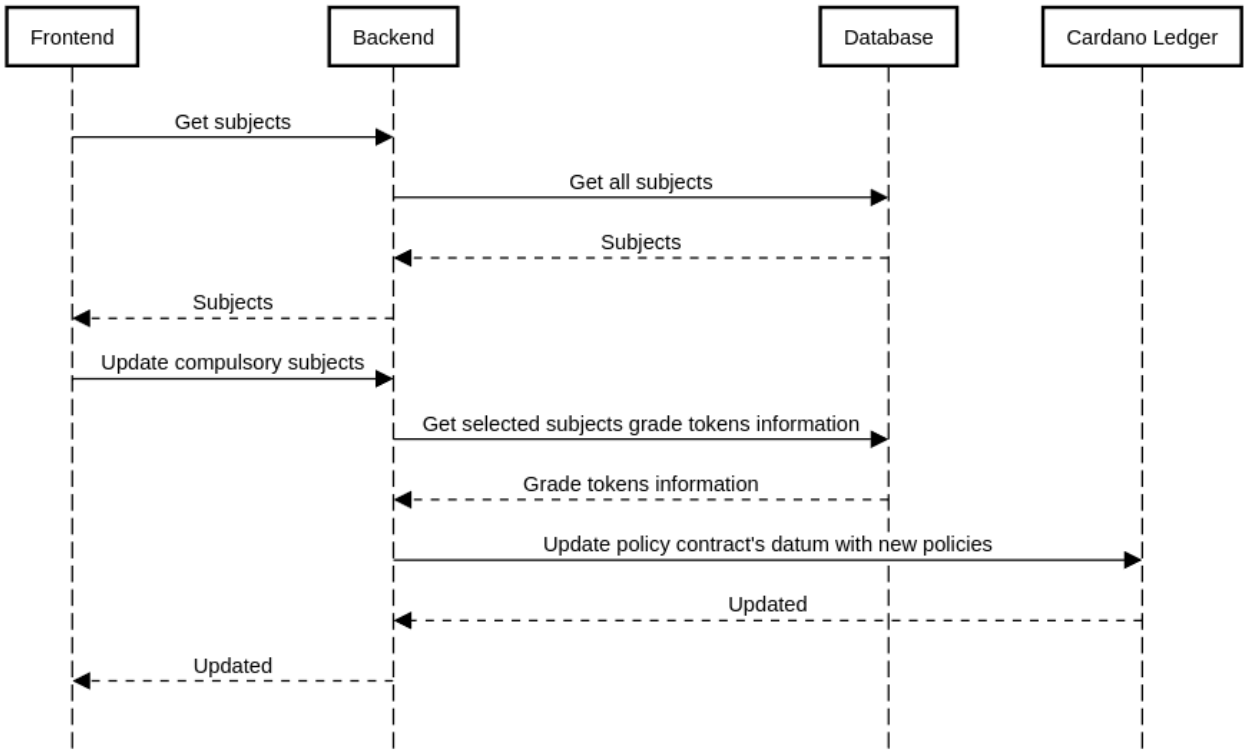


Figure 3.2.7: Curriculum change sequence diagram

Chapter 4

DApp Implementation

4.1 User Interface

The user interface is designed to meet the different needs of its stakeholders, namely students, teachers, and the secretary. It is built using the Pug templating engine. It not only provides a smooth and user-friendly experience but also includes advanced features that are necessary for a decentralized application running on the Cardano blockchain. The frontend utilizes the lucid-cardano library to provide direct interactions with the blockchain, including the ability to initiate transactions and retrieve blockchain data. In addition, the interface allows interaction with the Nami wallet, enabling users to securely perform blockchain actions directly within their web browser. Blockfrost serves as the blockchain service provider, facilitating smooth interaction between the DApp and the Cardano network. The frontend has a single login gateway for all individuals involved, allowing users to input their login information, choose their role, and then access a customized dashboard that corresponds to their specific requirements.

4.1.1 Students

4.1.1.1 Frontend

The student interface is designed to be user-friendly and uncomplicated, reducing the complexity commonly found in applications based on blockchain technology. The goal is to offer a smooth user experience that mimics the interaction with a traditional web application, therefore simplifying the underlying web3 processes. The DApp ensures that students can prioritize their educational objectives by minimizing their engagement with blockchain technology to only the essential degree minting procedure.

When students log in for the first time, they are presented with a QR code that they must scan using the ProofSpace mobile app. This step is essential for completing their university registration and linking their academic credentials with their decentralized ID.

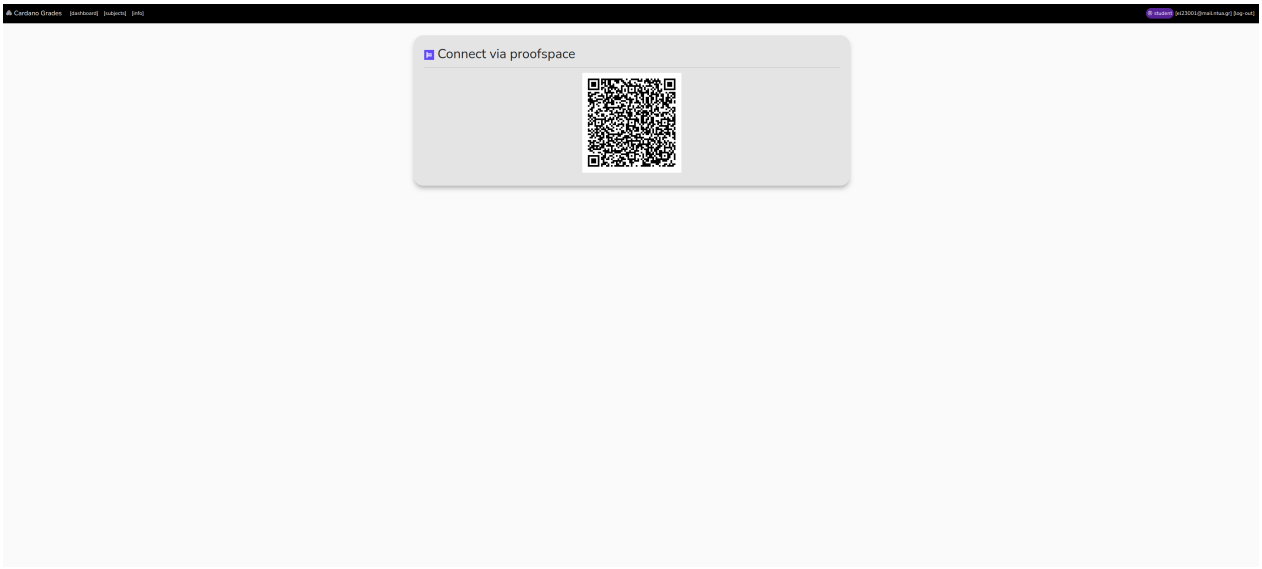


Figure 4.1.1: Student's first login view

After the initial setup and for all subsequent logins, students are directed to the dashboard. This interface displays the grades of the subjects they have passed, providing a quick overview of their academic achievements.

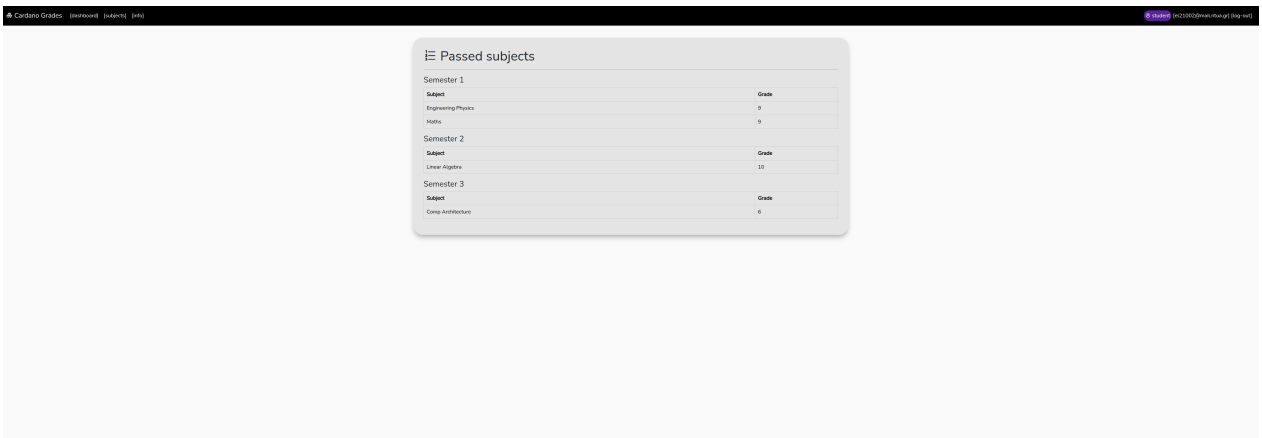


Figure 4.1.2: Student's dashboard

Another critical page is the semester enrollment interface, where students can view all available subjects for the upcoming semester. They can select the subjects they wish to enroll in by ticking boxes next to each subject and then submit their choices.

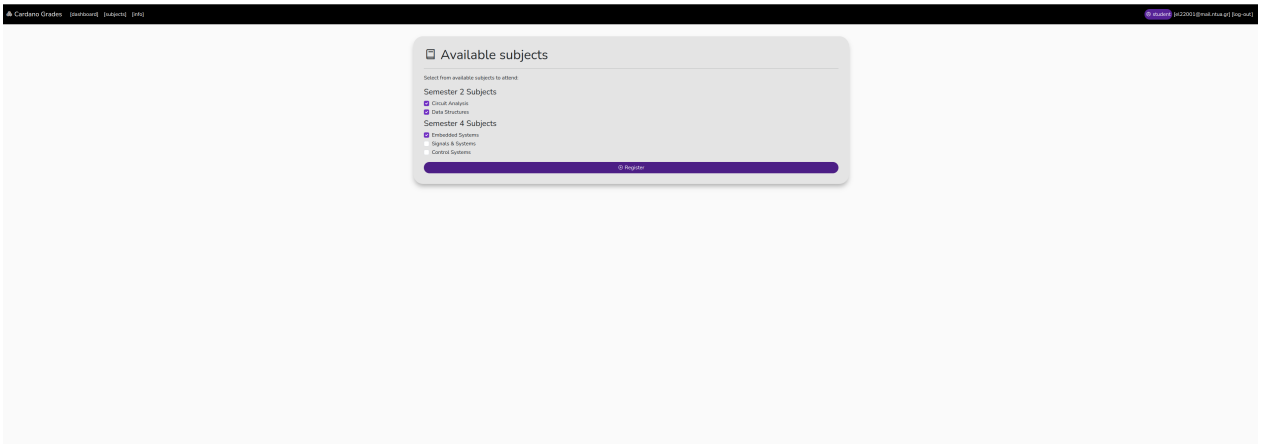


Figure 4.1.3: Student's semester and subject enrollment page

Students also have access to a page where they can view and update their personal information, such as email or password, ensuring their profile remains current and secure.

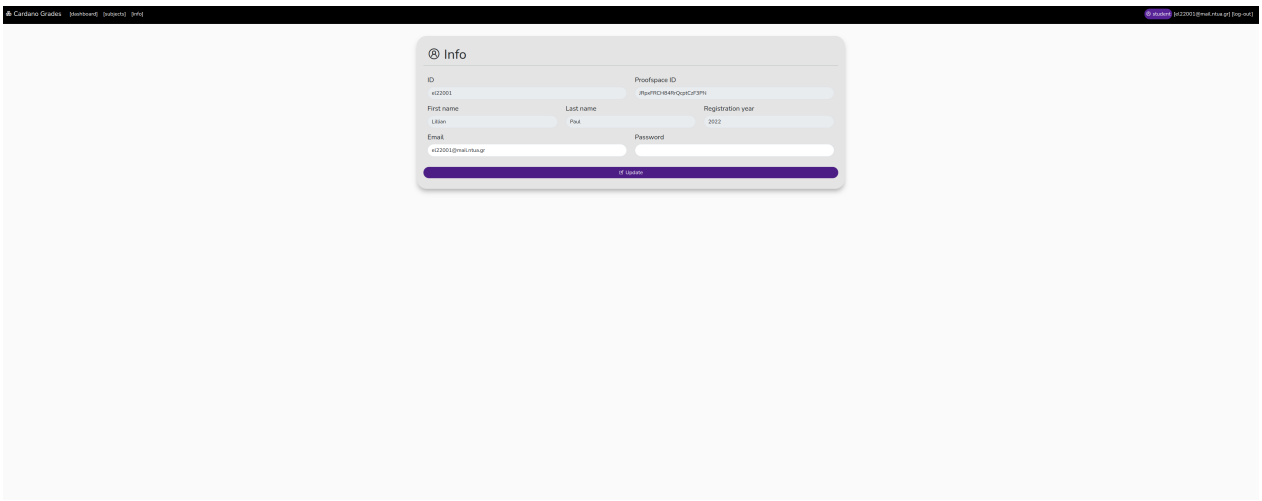


Figure 4.1.4: Student's information page

Lastly, after students have completed all compulsory subjects, a notification prompts them on the dashboard to claim their degree. This process involves a modal where students are asked to connect their Nami wallet. This is the only time students are required to interact directly with the Cardano blockchain by using a wallet, which is necessary for minting their degree as an NFT.

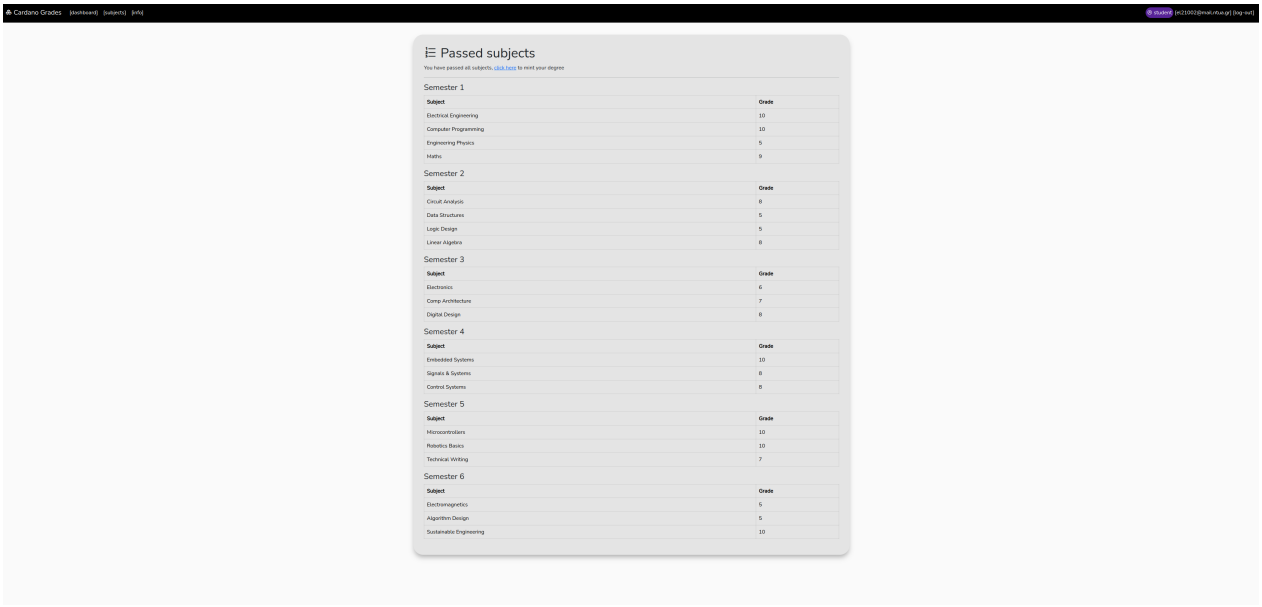


Figure 4.1.5: Student's degree minting message

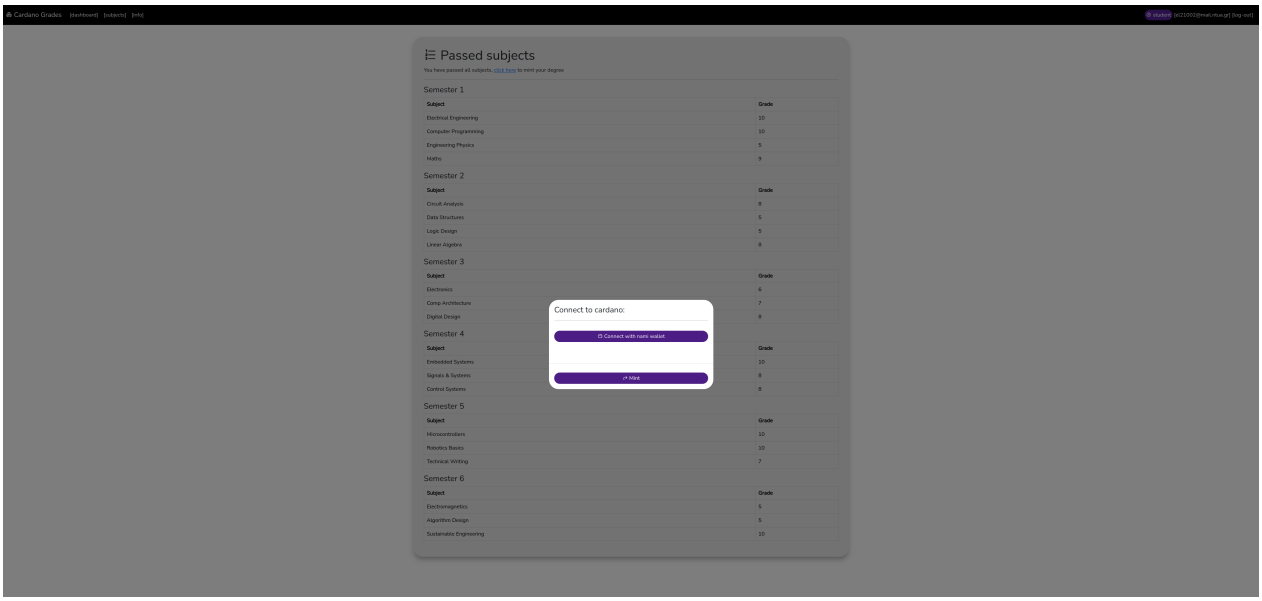


Figure 4.1.6: Student's degree minting modal

4.1.1.2 ProofSpace Mobile Application

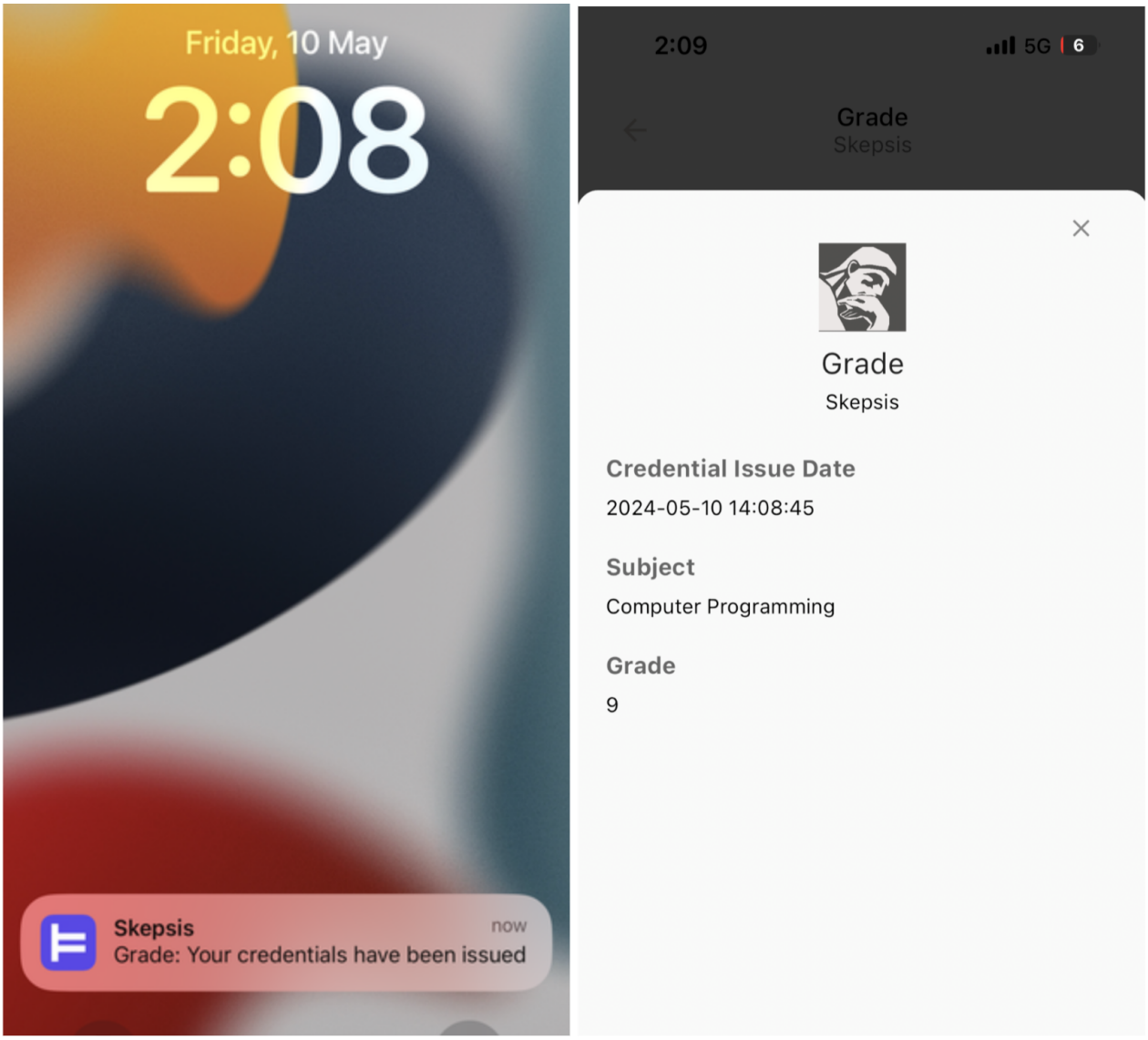
The ProofSpace mobile application acts as a secondary user interface for students, complementing the main web interface by offering a comprehensive digital wallet for their academic credentials. Through ProofSpace, students receive and store digital credentials related to various aspects of their academic journey:

- University Enrollment: Upon completing their initial registration process, students receive a digital credential that confirms their enrollment at the university.
- Semester Enrollment: Each time a student enrolls in a new semester, they receive a credential that details the subjects they have registered for that semester.
- Subject Grade: After grades are issued by teachers, students receive credentials for each subject,

including those where they did not pass, allowing them to keep a complete record of their academic performance.

- Degree: Upon successful completion of all required courses and receiving their degree, a final credential is issued that serves as a digital version of their degree certificate.

The ProofSpace app acts as a record keeper of a student's academic status, storing credentials and making them accessible at any time. This setup is particularly beneficial as it allows students to view all their grades, including non-passing ones, which the primary user interface does not display. Furthermore, the ProofSpace app enhances the student experience by sending push notifications whenever a new credential is issued, ensuring that students are promptly informed about updates such as new grades.



(a) Proofspace new grade notification.

(b) Proofspace new grade credential.

Figure 4.1.7: ProofSpace Mobile Application

4.1.2 Teachers

The teacher interface is designed to facilitate the task of grading, aligning with the application's goal of simplifying interaction with blockchain technology. However, due to the nature of blockchain transactions,

the process of adding grades involves direct interaction with the web3 elements, as teachers are responsible for initiating and signing transactions that alter students' academic records.

Upon logging in, teachers are directed to their personalized dashboard where they can easily access a list of all the subjects they are assigned to teach.

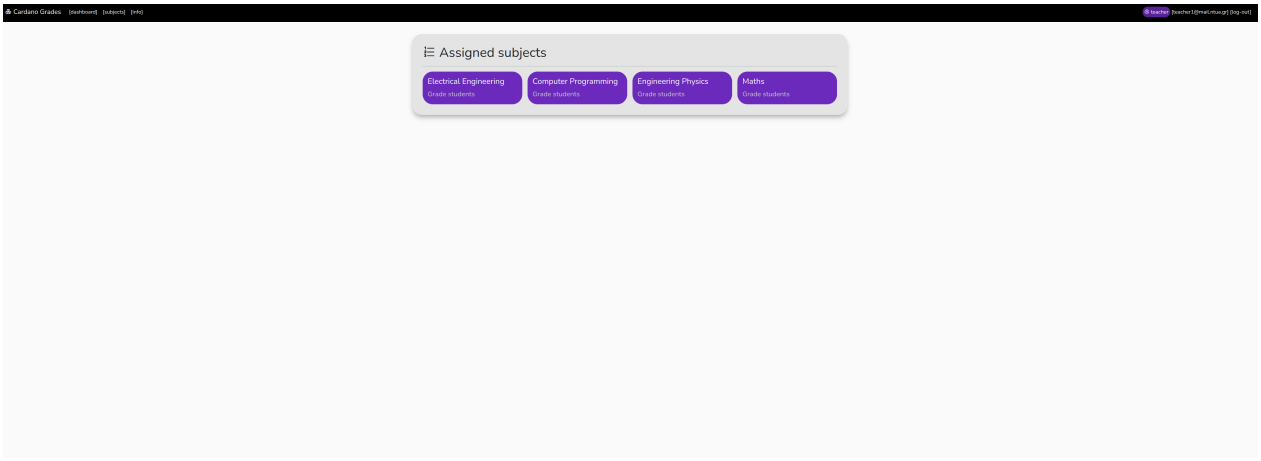


Figure 4.1.8: Teacher's dashboard

This presentation allows teachers to select any subject and view a list of all registered students for that course. From here, teachers can input grades for each. The grading process is facilitated by the cardano-lucid library, which handles the details of interacting with the Cardano blockchain. The relevant function first retrieves the smart contract's address and the relevant UTXOs needed for the transaction. It then crafts the transaction outputs to ensure that the distribution of grade tokens to the students' addresses is executed correctly, otherwise the subject's validator would cause the transaction to fail. It is crucial for all teachers to have a Nami wallet installed on their browser, with a funded Cardano address, in order to execute this action.

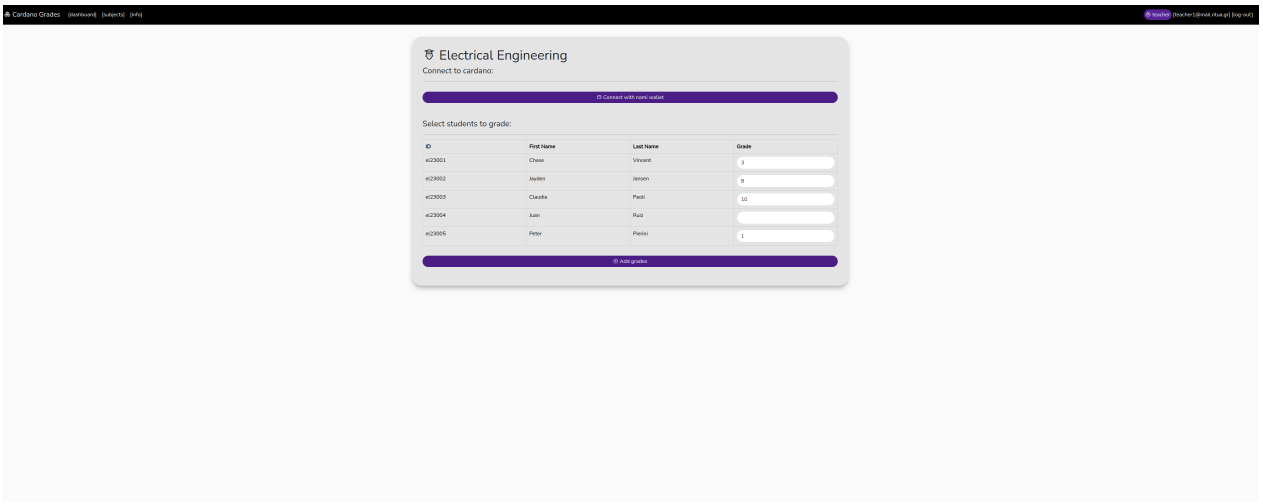


Figure 4.1.9: Teacher's grading interface

Lastly, akin to students, teachers have the ability to view and modify their personal information through the designated page.

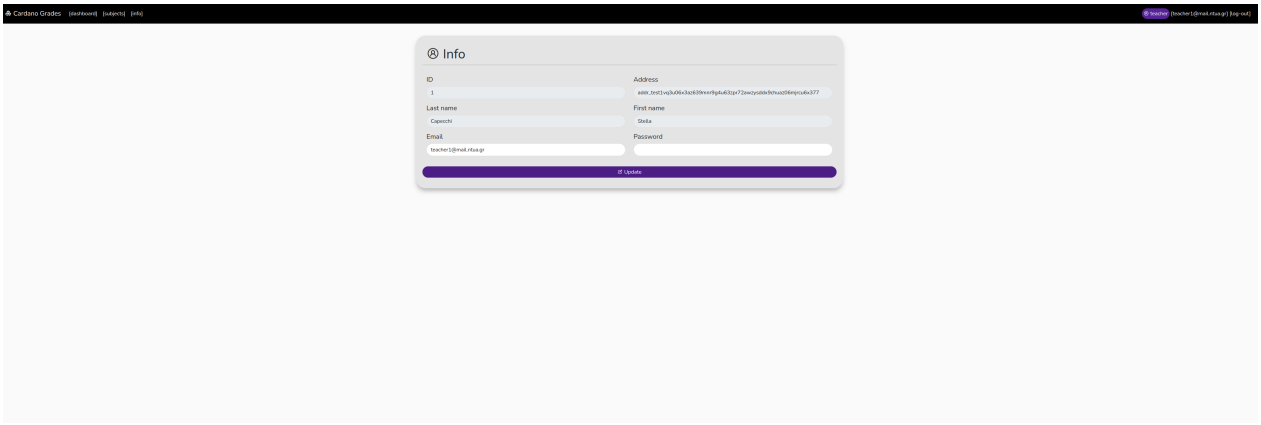


Figure 4.1.10: Teacher's information page

4.1.3 Secretary

The secretary's interface is streamlined into a single-dashboard page designed for management of administrative tasks. This interface is divided into two main sections:

- Registration Period Management: The upper half of the dashboard features a date input field where the secretary can set the end registration date for the academic period. By entering a date, the secretary effectively starts the semester enrollment period, which begins immediately and concludes on the specified end date.
- Curriculum Management: The lower half of the dashboard presents a comprehensive list of all the university's registered subjects, which are retrieved from the off-chain database. This section enables the secretary to manage the curriculum effectively. When there is a need to update the degree requirements - often due to curriculum changes - the secretary can select subjects from this list to designate them as compulsory for obtaining a degree.

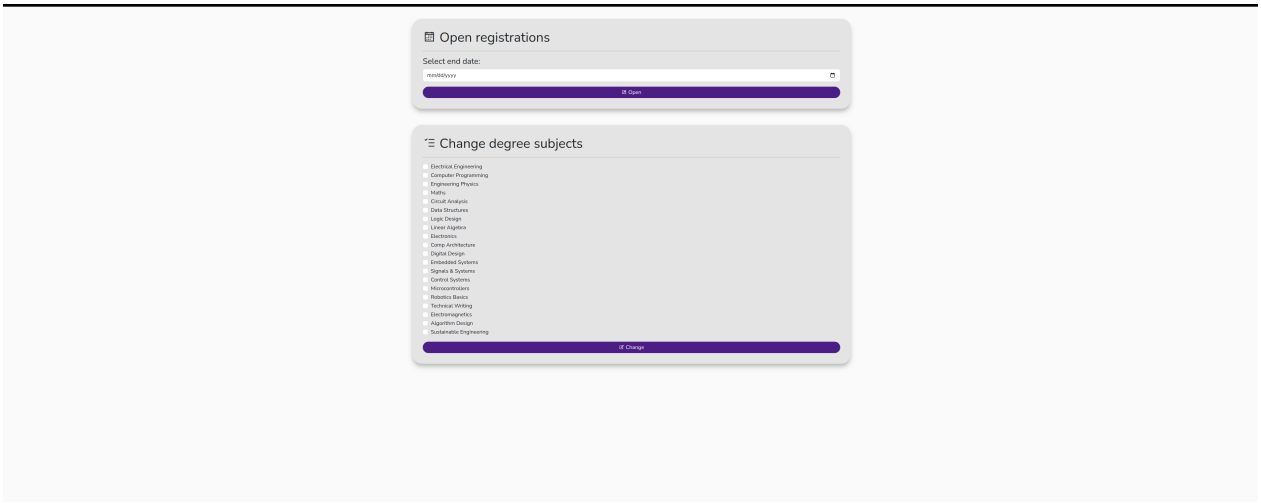


Figure 4.1.11: Secretary's dashboard

4.2 Smart Contracts

The smart contracts serve as the fundamental framework of the app, operating as the core mechanism for enforcing and managing the application's logic on the Cardano blockchain. These contracts are written in Aiken[28], a programming language developed by TxPipe specifically for writing Cardano contracts. They

are designed to establish well-defined, immutable rules that govern all the critical aspects of the university management system. These rules specify the conditions under which the different tokens - namely grade tokens, degree tokens, and registration tokens - can be minted or burned. Additionally, the smart contracts manage the registration processes, detailing under which conditions a student can register for a subject or receive grades and they also allow for the dynamic updating of curriculum requirements. The complete code is provided in the Appendix for reference.

4.2.1 Student Validator

The Student Validator serves as the internal blockchain address for each student. The purpose of this system is to receive and securely store the grade tokens of the students. The Student Validator is specifically designed to prevent students from transferring or burning their grade tokens, in contrast to a typical wallet. Enforcing this limitation is crucial to sustain the integrity and permanence of academic records. The validator has a direct function: it prevents any transactions that try to use the unspent transaction outputs (UTXOs) stored at that particular address. Consequently, once tokens are placed into a student's address, they persist there permamently, functioning as an immutable record of the student's academic accomplishments. Since the validator always triggers an error when attempting to consume a UTXO, it requires neither Datum nor Redeemer, thus it has no configurable options.

4.2.1.1 Parameters

The Student Validator is a parametrized script with a single parameter of type "bytes", utilizing each student's ID as the parameter to generate unique addresses for individual students.

4.2.2 Subject Validator

The Subject Validator smart contract serves as a secure repository for the subject's grade tokens, ensuring they are locked and can only be transferred under specific conditions outlined in the validator. The functionalities of this contract are threefold: it allows for the registration and tracking of students enrolled in the subject, permits the secretary to update the subject's teacher and the registration closing date, and enables the teacher to assign grades.

4.2.2.1 Parameters

The Subject Validator smart contract is parametrized with four key parameters:

- A "PolicyID" type for the policy ID of the subject's grade token, which helps in identifying and managing the specific grade tokens associated with the subject.
- An "AssetName", which is essentially an alias for bytes, representing the name of the grade token. This ensures that the correct grade tokens are tracked and managed.
- A "PubKeyHash" type, which stores the secretary's Cardano public key. This is critical for maintaining the immutability of the contract, ensuring that only authorized changes can be made by the secretary.
- Another "PolicyID" for the registration policy's ID, which is used in verification checks to confirm that all students receiving grade tokens are officially registered in the university.

4.2.2.2 Datum

The Datum structure for the Subject Validator smart contract consists of four fields:

- Teacher's Public Key: A "pubKeyHash" that stores the Cardano public key of the teacher responsible for the subject.
- Registrations Array: An array of bytes used to track the IDs of all students registered for the subject. This allows the system to keep a dynamic record of student participation and eligibility for receiving grades.

- Registration Closure Time: A "POSIXTime", represented as an integer, denotes the exact time in seconds when registrations for the subject close.
- Reset Flag: A "Bool" field named reset indicates whether the registrations array needs to be cleared after the next grading cycle.

4.2.2.3 Redeemer Options

Register The Register redeemer is designed to add a new student IDs to the registrations array. In order for the transaction to succeed, it has to be invoked by the secretary and operate under strict conditions to maintain the integrity of the contract's data. Specifically, it requires the creation of a new UTXO at the same address, with the datum unaltered except for the addition of the new student's ID in the registrations array. Furthermore, the values within the UTXO must remain unchanged to prevent any unauthorized transfer of grade tokens, and, lastly, the action must occur within the designated registration period, as specified by the closure time in the datum.

Update The Update redeemer is designed to handle updates to the teacher's public key in the event of a change in teaching personnel, as well as modifications to the registration closure date. For the call to be successful, the transaction must have been signed by the secretary. Additionally, to ensure the integrity of the contract, the Update redeemer requires that all elements of the datum remain unchanged except for the teacher's public key and the registration closure date. Lastly, the values within the transaction must also remain unchanged to prevent any manipulation of token balances.

Grade The Grade redeemer y designed to distribute grade tokens, ensuring that only eligible students receive grades.

1. Authority to Call: Only the subject's teacher is authorized to invoke the Grade redeemer.
2. Verification of Student Registration: The spending transaction must include reference inputs of all UTXOs containing the university registration token from the students being graded. This requirement allows the smart contract to retrieve each student's ID (encoded as the asset's name in the token) and verify that it appears in the subject's registrations array, confirming that the student is legitimately registered for the subject.
3. Distribution of Grade Tokens: For each grade token sent to an external address, there must be a corresponding UTXO at that address containing the student's registration token, ensuring the student's eligibility for receiving grades. The quantity of grade tokens in these UTXOs must be between 5 and 10, indicative of passing grades. All remaining grade tokens must be returned to a UTXO at the subject contract's address, keeping the bulk of the liquidity securely locked within the subject contract, except for those tokens distributed to students. Lastly, the sum of all grade tokens present in the outputs must be equal to the quantity of grade tokens in the contract, ensuring no tokens are burned during the process.
4. Handling the reset Flag: The reset flag is used to manage the grading process across standard and repeated exams. Initially set to false, the flag must be set to true after a teacher grades for the first time. Subsequent grading (for repeats) with the reset flag true necessitates resetting the flag to false and clearing the registrations array, preparing the system for the next semester's registrations.

Lastly, we will delve into the implementation logic of the grade validation algorithm, from step 3, which forms the core of the entire system. Implementation Logic:

- For each transaction output that contains grade tokens:
  - If the output address is the subject contract's self address, the process continues without further checks.
  - If the output address is external:
    - The quantity of grade tokens must fall within the range [5,10].

* A reference input at the same address must contain the student's registration token, identified using the registration PolicyID parameter. This token's asset name, which is the student's ID, must be listed in the registrations array; otherwise, the transaction will fail, blocking attempts to grade unregistered students.

4.2.3 Policy Data Validator

The Policy Contract serves as a registry for all valid system grade tokens, sensible for maintaining the integrity and authenticity of academic credentials. This contract is required for protecting against potential security threats where malicious actors might attempt to create and use fake grade tokens. By holding a protected record of the accepted policy IDs and asset names for grade tokens, the Policy Contract ensures that only authorized tokens are recognized and utilized within the system. This registry is referenced by the degree minting policy to verify that all conditions for minting a degree - such as the possession of necessary grade tokens - are met. Additionally, the contract allows the secretary to update the policies, in order to accommodate any changes in the curriculum.

4.2.3.1 Parameters

The Policy Contract is parametrized with a single parameter, the secretary's public key hash.

4.2.3.2 Datum

The valid token policies and asset names are stored within the datum as a single array, facilitating upgrades. This array consists of tuples, each of which is of type "(PolicyID, AssetName)". This structure enables convenient storage and retrieval of the grade tokens recognized within the system.

4.2.3.3 Redeemer Options

The Policy Contract includes a single redeemer option specifically designed for updating the datum, which details the grade tokens required for degree issuance. For the transaction to be successful, it must be initiated by the secretary. Additionally, a new UTXO must be created at the same address with the exact same values as the previous one.

4.2.4 Grade Token Minting Policy

The Grade Token represents a student's academic performance in each subject, with each subject having its own unique token with unlimited supply. For example, owning six tokens indicates a grade of six in that subject. The token's Asset Name matches the subject's name, and all tokens are securely locked within the corresponding subject's smart contract.

4.2.4.1 Parameters

The Grade Token smart contract is parameterized with two types:

- A UTXO Reference
- An AssetName, which specifies the name that the minted token will have, corresponding to the subject's name.

Even if two subjects share the same names, they will still have distinct policies due to the UTXO reference parameter, which is unique for each subject's contract.

4.2.4.2 Minting Conditions

The conditions under which grade tokens can be minted are:

- The asset name of the tokens to be minted must match the AssetName parameter specified in the contract. This alignment ensures that each token is correctly associated with its intended subject.

- The minting transaction must consume the specific UTXO with which the policy was parameterized. This step is important as it prevents double minting of tokens. By requiring the consumption of a designated UTXO, the contract ensures that once that UTXO has been consumed, it cannot be used ever again, thereby eliminating the possibility of reusing the minting conditions.

4.2.5 Registration Token Minting Policy

The Registration Token Minting Policy facilitates tracking both university admissions and semester enrollments. Specifically:

- University Registration: When a student registers at the university, a token is minted and sent to the student's registered address. The asset name of this token corresponds to the student's ID, uniquely linking the token to that student's university registration.
- Semester Enrollment: For semester enrollments, the asset name of the token denotes the semester number the student is enrolling in, such as 1 for the first semester, 2 for the second, etc. This method provides a straightforward and effective way to track which students are enrolled in which semesters, helping to prevent double-registrations and ensure accurate record-keeping.

4.2.5.1 Parameters

The Registration Token's minting policy is parametrized with a single parameter, the secretary's public key hash.

4.2.5.2 Minting Conditions

The sole condition for minting a registration token is that the transaction must be signed by the secretary.

4.2.6 Validity Token Minting Policy

The Validity Token plays a crucial role in the security and integrity of the application, ensuring that the correct policy contract UTXO is uniquely identified. This token's Unit is hardcoded within both the off-chain and on-chain components of the system. The Validity Token is a sensible component because it prevents malicious actors from creating fake UTXOs at the policy contract's address that could otherwise be mistaken as valid, potentially leading to the incorrect issuance of degrees. By ensuring that the Validity Token is present in the valid policy UTXO's values, the system can reliably confirm the authenticity of the policy data.

4.2.6.1 Parameters

It has a single parameter, a UTXO Reference, which is used to guard against double-minting by ensuring each minting event is linked to a unique transaction output.

4.2.6.2 Minting Conditions

The minting conditions for the Validity Token enforce that the asset name is "Validity", the quantity minted is exactly one, and the specified output reference in the parameter is consumed during the minting transaction. These conditions are crucial for maintaining the integrity of the system, guaranteeing that there will ever be only one Validity Token in existence.

4.2.7 Degree NFT Minting Policy

The Degree Token represents the university's degree NFT. This token is a testament to the completion of a student's academic studies. The student's degree grade is embedded as metadata within the minting transaction, providing a detailed and permanent record of their academic achievement. Unlike other tokens in the system, the Degree Token is not sent to the internal student's address managed by the smart contract; instead, it is directly transferred to the student's personal account. The Asset name of the degree NFT is the student's ID.

4.2.7.1 Parameters

The Degree Token in the university management DApp is parameterized with three crucial elements that ensure the legitimacy and integrity of the degree issuance process:

- University's Registration Token's Minting PolicyID: This parameter is used to verify that the student receiving the degree is officially registered at the university.
- Validity Token's Policy ID: Essential for ensuring the degree minting transaction references the correct policy contract's datum. This parameter confirms that the policy UTXO holding the degree requirements contains the validity token, thereby verifying the authenticity and currentness of the degree eligibility criteria.
- Secretary's Public Key Hash: This parameter is included because the secretary is the only individual authorized to mint degrees.

4.2.7.2 Minting Conditions

The minting conditions defined to ensure the integrity and appropriateness of the degree issuance process. Here are the necessary conditions:

- The minting transaction must include as reference inputs the policy contract's UTXO, which contains the complete list of grade tokens a student must possess to claim their degree, the student's UTXO from their internal address, which holds their university registration token, and additional UTXOs for each subject the student has passed, as evidenced by the presence of the corresponding grade tokens.
- The policy contract UTXO referenced must contain the validity token as its value to ensure and prove its authenticity.
- The asset name of the degree token being minted must match the registration token's asset name found in the student's UTXO. This condition ensures that the degree is correctly attributed to the registered student, confirming their identity and registration status.
- For all grade tokens listed in the policy contract's datum array, there must be a corresponding reference input showing that the student possesses that token in a quantity of five or more. This requirement verifies that the student has successfully passed all required subjects with satisfactory grades.
- The minting transaction must be signed by the secretary.

4.2.7.3 Minting Degree as NFT

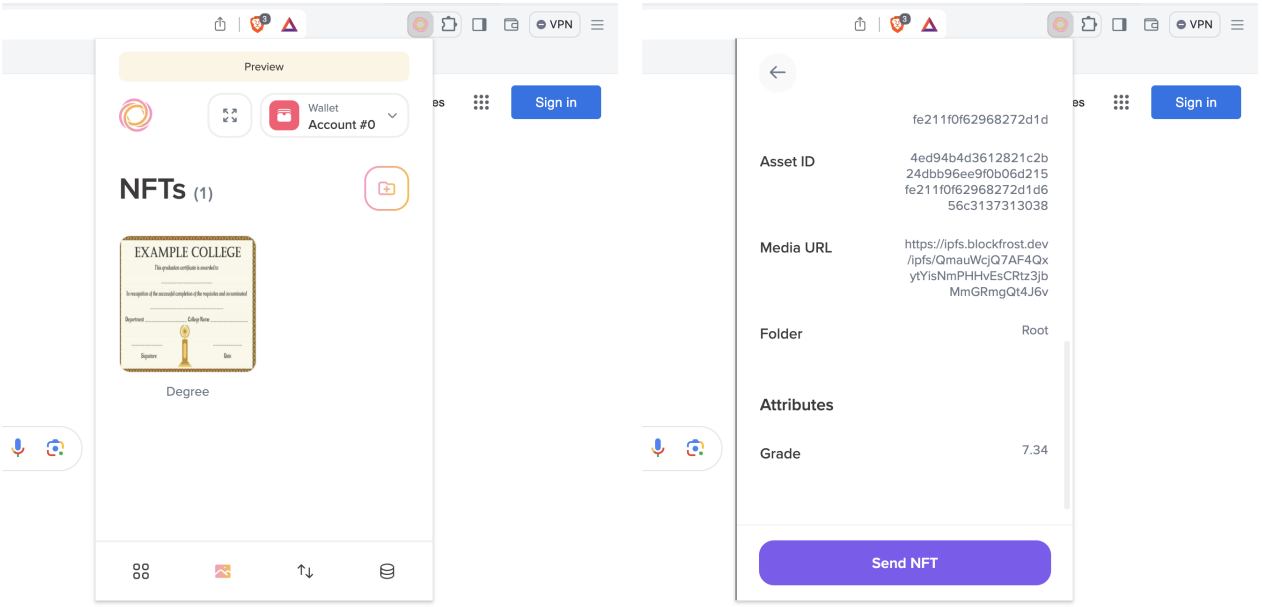
To represent the degree as an NFT on the Cardano blockchain, we leverage the capability to attach metadata to minting transactions. According to Cardano Improvement Proposal 26 (CIP-26), NFT metadata must adhere to a specific standard that identifies the native token as an NFT. The structure includes mandatory fields and optional fields for custom properties. The standard format is as follows:

```
{
  "721": {
    "<policy_id>": {
      "<asset_name>": {
        "name": <string>,
        "image": <uri | array>,
        "mediaType": image/<mime_sub_type>,
        "description": <string | array>,
        "files": [{
          "name": <string>,
          "mediaType": <mime_type>,
          "src": <uri | array>,
          <other_properties>
        }],
        <other properties>
      }
    },
    "version": <version_id>
  }
```

```
}
```

In our implementation, besides the token's name and description, we include an image for display purposes, which currently is uniform across all Degree Tokens. In the future, we plan to provide personalized images that reflect each student's degree specifically. Additionally, we incorporate the student's grade within the metadata. The metadata is attached to the transaction using the Lucid library. Below is a simplified example of a minting snippet that includes these metadata and transfers the token to the student:

```
async function mintDegreeNFT(name, degreeGrade) {
    const unit = degreePolicyId + fromText(name);

    const metadata = {
        [degreePolicyId]: {
            [name]: {
                image: "ipfs://QmauWcjQ7AF4QxytYisNmPHHvEsCRtz3jbMmGRmgQt4J6v",
                name: "Degree",
                grade: degreeGrade,
                description: "University Degree NFT"
            }
        },
        version: "1.0"
    };

    const tx = await lucid
        .newTx()
        .mintAssets({ [unit]: 1n })
        .validTo(Date.now() + 100000)
        .attachMintingPolicy(mintingPolicy)
        .payToAddress(addself, { [unit]: 1n })
        .attachMetadata(721, metadata)
        .complete();

    const signedTx = await tx.sign().complete();
    const txHash = await signedTx.submit();

    return txHash;
}

await mintDegreeNFT("el17108", "7.34");
```



(a) Degree NFT

(b) Degree NFT's metadata

Figure 4.2.1: Degree NFT as displayed on the Lace wallet

4.3 Backend

The backend, implemented in JavaScript using the Express framework, is integral to its operation, handling web server tasks and API requests. It leverages the lucid-cardano library to manage all blockchain interactions, ensuring robust integration with the Cardano network. Key functions include maintaining a connection with the off-chain database and handling most of the off-chain Cardano logic. Moreover, except for grading transactions, which are initiated by teachers, all other transactions - such as student registrations and degree minting - are securely signed using the admin key stored on the backend server. Lastly, the backend ensures system integrity through request validation and employs authorization middleware to restrict user access to relevant resources.

4.3.1 Student API Specification

4.3.1.1 View grades

Endpoint /student/dashboard

Method GET

Purpose To retrieve the student's passed subjects' grades.

Functionality It first retrieves all UTXOs from the student's internal address using the cardano-lucid library, along with the valid policies array from the policy contract's datum. It then matches grade token policies from these UTXOs to their respective subjects. The data is formatted and returned.

4.3.1.2 View subjects available for registration

Endpoint /student/subjects

Method GET

Purpose To return all available subjects that the student can register in.

Functionality This endpoint operates by first retrieving all subjects from the database that match the registration criteria: subjects must be from lower semesters than the student and match the student's current semester type (odd/even). It then retrieves the valid policies from the policy contract's datum and filters out any subject that is not required for the degree. Additionally, it retrieves the user's grades and excludes any subjects that the student has already passed. This ensures that only relevant and necessary subjects are presented for registration.

4.3.1.3 Enroll in subjects

Endpoint /student/subjects

Method POST

Purpose To enroll in the current semester and in the chosen semester subjects.

Functionality This endpoint first validates that the semesters selected by the student are valid for registration, ensuring compliance with the semester structure (e.g., odd/even alignment). It then interacts with each selected subject's smart contract using the 'Register' redeemer option to add the student to the registrations arrays. A registration token for the current semester is subsequently minted to the student's address. Finally, the endpoint calls the ProofSpace API to issue a semester registration certificate and a subject registration certificate for each enrolled subject, providing official documentation of the student's registration.

4.3.1.4 View student's information

Endpoint /student/info

Method GET

Purpose To return the student's personal information.

Functionality This endpoint retrieves the student's ID, first name, last name, email, ProofSpace ID, and registration year as they appear in the database.

4.3.1.5 Update student's information

Endpoint /student/info

Method POST

Purpose To update the student's personal information.

Functionality This endpoint allows for the updating of the student's email and/or password in the off-chain database.

4.3.1.6 Mint Degree

Endpoint POST /student/mint

Method POST

Purpose To mint the student's degree.

Functionality This endpoint begins by retrieving the student's UTXOs along with the valid policies from the policy contract to verify that the student has completed their studies and is eligible to claim their degree. It calculates the degree grade as the average of the grades represented by the UTXOs. The backend then crafts the minting transaction, using all the necessary student UTXOs as reference inputs, and mints the degree NFT to the provided address. Lastly, it invokes the ProofSpace API to issue the degree credential, officially certifying the student's academic achievement.

4.3.2 Teacher API Specification

4.3.2.1 View teaching subjects

Endpoint /teacher/subjects

Method GET

Purpose To retrieve the subjects this teacher is assigned to.

Functionality This endpoint fetches and returns the list of subjects assigned to the teacher directly from the off-chain database.

4.3.2.2 View registered students for Subject

Endpoint /teacher/subject/:subjectId/addGrade

Method GET

Purpose To retrieve the list of students registered in the specified subject.

Functionality This endpoint initially retrieves the smart contract address for the subject specified by ‘subjectId‘ from the off-chain database. It then accesses the registrations array from the UTXO’s datum at this address, which contains only the student IDs. To provide a more detailed view, the endpoint further queries the off-chain database to fetch the first and last names associated with these IDs. The result is a comprehensive list of student IDs, first names, and last names.

4.3.2.3 Add grades

Endpoint /teacher/subject/:subjectId/addGrade

Method POST

Purpose To issue the ProofSpace grade certificate to the students in the specified subject.

Functionality While the Cardano call to issue grades via the subject’s smart contract is managed directly by the frontend, this endpoint is solely responsible for invoking the ProofSpace API. After grades are submitted and processed on the blockchain, this endpoint ensures that each student receives a digital grade certificate from ProofSpace, formally documenting their academic achievement in the subject.

4.3.2.4 View teacher’s information

Endpoint /teacher/info

Method GET

Purpose To return the teacher’s personal information.

Functionality This endpoint retrieves the teacher’s ID, first name, last name, email and Cardano Address as they appear in the database.

4.3.2.5 Update teacher’s information

Endpoint /teacher/info

Method POST

Purpose To update the teacher’s personal information.

Functionality This endpoint allows for the updating of the teacher’s email and/or password in the off-chain database.

4.3.3 Secretary API Specification

4.3.3.1 Add a new Student

Endpoint /admin/addStudent

Method POST

Purpose To register a new student to the system.

Functionality This endpoint adds a new student entry to the database based on the data provided in the POST request.

4.3.3.2 Add a new Teacher

Endpoint /admin/addTeacher

Method POST

Purpose To register a new teacher to the system.

Functionality This endpoint creates a new teacher entry in the database based on the submitted data.

4.3.3.3 Add a new Subject

Endpoint /admin/addSubject

Method POST

Purpose To add a new subject to the university curriculum.

Functionality Upon receiving new subject data, this endpoint adds the subject details to the database and mints a new grade token for the subject. The minted grade tokens are then locked in the newly created subject smart contract, securing the liquidity and associating it directly with the subject.

4.3.3.4 Update a Subject

Endpoint /admin/updateSubject

Method POST

Purpose To update information on an existing subject.

Functionality This endpoint can update various details of a subject such as the teacher or semester. Updates are made either in the off-chain database or via calls to the subject's smart contract, such as updating the teacher's public key if there has been a change in teaching staff.

4.3.3.5 Change compulsory subjects

Endpoint /admin/changeSubjects

Method POST

Purpose To update the list of compulsory subjects required for degree completion.

Functionality This endpoint interacts with the policy contract to update the array of policies, specifically the grade tokens required for degree completion.

4.3.3.6 Start registration period

Endpoint /admin/startRegistration

Method POST

Purpose To start the registration period for the current semester.

Functionality This endpoint retrieves current semester subject information, calculates their contract addresses, and individually calls each subject's contract using the 'Update' redeemer to set the 'end_date' for registrations, thus opening the registration window for students.

4.3.4 General API Specification

4.3.4.1 Login

Endpoint /login

Method POST

Purpose Login

Functionality This endpoint allows users to log in to the system using credentials stored in the database. It authenticates users based on their provided username, password and role.

4.3.4.2 Logout

Endpoint /logout

Method GET

Purpose Logout

Functionality This endpoint handles user logout, effectively ending the user session and ensuring that access privileges are revoked.

4.3.4.3 Get compulsory subjects information

Endpoint /policies

Method GET

Purpose To return information about compulsory subjects along with their grade token policies and asset names.

Functionality This endpoint retrieves the list of compulsory subjects and their associated grade token policies directly from the policy contract. It then pulls additional necessary information from the database about these subjects, compiling a comprehensive dataset to return to the requester.

4.3.4.4 Get subject information by ID

Endpoint /subjects/:id

Method GET

Purpose To return detailed information about a specific subject.

Functionality This endpoint first retrieves necessary data about the subject from the database. It then calculates the grade token's unit and the subject's address based on this data. Finally, it fetches additional information such as the registration close period, the teacher's address, and other relevant data from the subject contract's datum, providing a detailed view of the subject.

4.4 ProofSpace

ProofSpace is a Software as a Service (SaaS) platform that specializes in handling decentralized IDs and credentials within the Cardano blockchain, among other platforms. This service simplifies the process of managing decentralized credentials by abstracting the underlying interactions into a no-code solution, enabling the creation of certificate-based functionalities within applications. ProofSpace provides an API that allows applications like ours to trigger certificate issuing actions seamlessly, facilitating efficient and secure credential management.

By integrating ProofSpace, our application enables students to claim and verify their academic certificates publicly through the ProofSpace app. This functionality is crucial for verifying student registrations in various academic settings, such as classroom attendance and laboratory access, where enrollment prerequisites may apply. Additionally, ProofSpace enhances user experience by sending push notifications directly to students' mobile devices whenever new grades are recorded or certificates are issued, providing real-time updates without the need for students to repeatedly check the web platform. Opting for ProofSpace over a custom-built solution allows us to leverage a well-established system that provides a user-friendly interface and reliable service, avoiding the complexities and potential risks associated with developing such a system by ourselves.

In our application, ProofSpace is utilized to manage five distinct types of academic credentials, alongside six different types of interactions. Each credential serves a specific purpose within the academic lifecycle of a student:

- University Registration: Issued once to each student, this credential signifies their official registration with the university. It includes data such as the university name and the student's ID.
- Semester Registration: Students receive this credential for each semester they register, marking their enrollment in that specific term.
- Subject Registration: Issued each semester a student enrolls in a subject, this credential contains details about the subject and the valid semester.
- Grade Certificate: This is provided to students upon receiving a grade in a subject, detailing the subject name and the student's grade.
- Degree Certificate: The final credential, it signifies the completion of a student's academic program and includes the degree grade along with other essential details.

The interactions within the application are categorized into two main types:

- QR Code Interaction: This singular interaction involves a QR code displayed on the student's frontend during their initial login to the web interface. The purpose of this QR code is to link the student's ProofSpace DiD with our service.
- Simple "Webhook" Interactions: These involve triggers from API calls to specific endpoints designed to issue credentials directly to users. This mechanism ensures that credentials are issued promptly and accurately in response to various academic activities such as registering for subjects or achieving grades.

4.4.1 Credential Definitions

In this section, we will first explore how to create a new Credential Definition in the ProofSpace CRM, examining all available options. Later, we will delve into the specific definitions of each credential used within our application.

4.4.1.1 Adding a new Credential

To create a new credential definition in the ProofSpace CRM, we have to first create a schema, and later create the definition based on it. Begin by navigating to the 'Schemas' page. Once there, click the 'Add Schema' button located in the top right corner of the interface. This action will open a new page that prompts you to enter the details of the schema. Input the required information into the modal to proceed with the creation of the new credential schema.

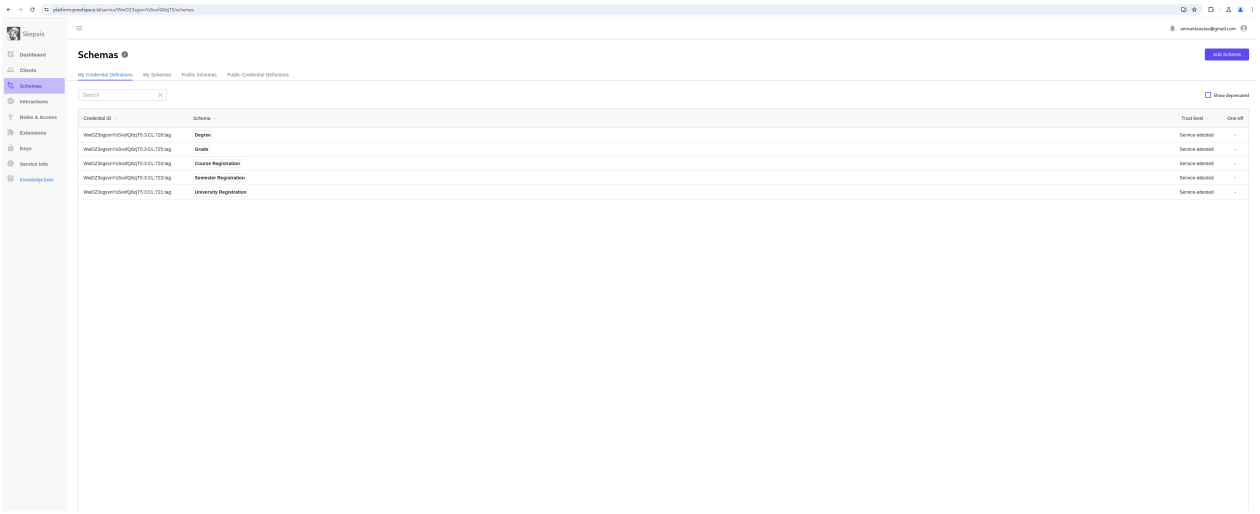


Figure 4.4.1: Credential Schemas Main page

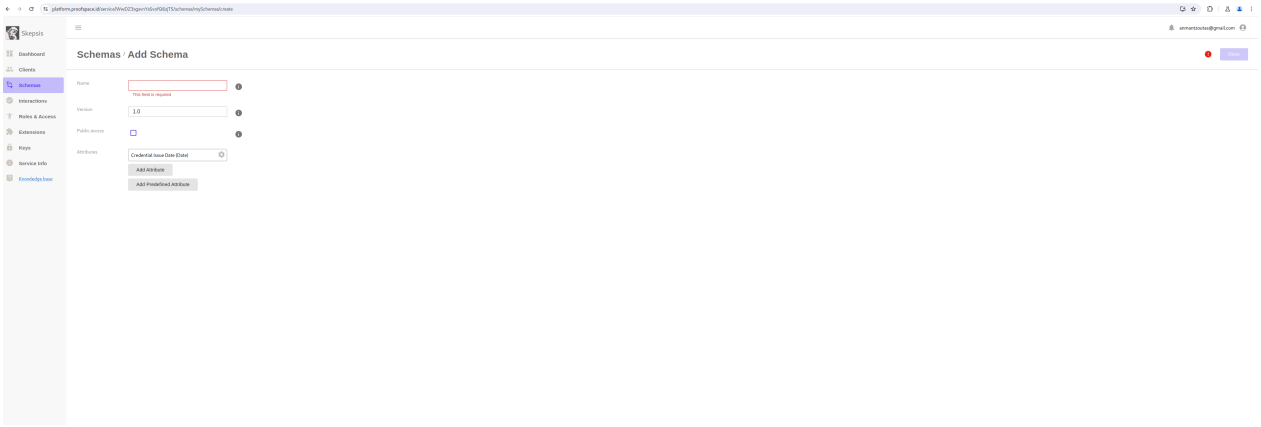


Figure 4.4.2: Adding a new Credential Schema

To continue with the process of creating a new credential schema, enter the name of the schema and any desired attributes. To add a new attribute, click the "Add Attribute" button. You will be prompted to provide a name for the attribute, select the expected type (such as text, number, date, etc.), and you can also add an optional description for further clarification.

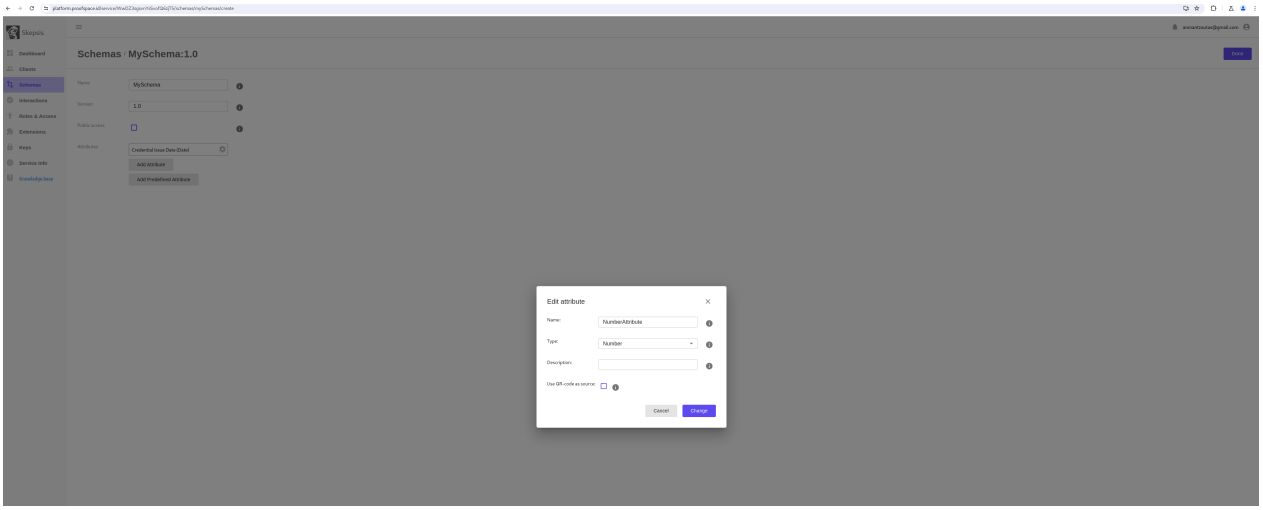


Figure 4.4.3: Adding a new attribute

Once all required attributes are added, click the "Done" button in the top right corner, followed by "Save & Publish." Confirm the action to finalize and make your new credential schema available for use. After saving, the newly created schema will appear on the Schema's main page. To proceed with creating the definition, click on "Create" in the corresponding schema's row.

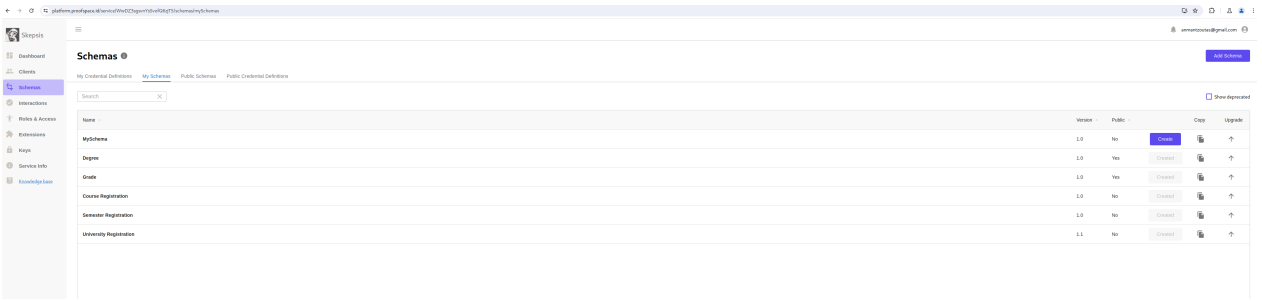


Figure 4.4.4: The newly created Schema

Once you navigate to the designated page to create the definition, you'll be reminded that creating a definition is a one-time operation; once set, it cannot be modified. The interface is split into two sections: the upper half requires you to input the definition's settings, while the lower half displays information about the relevant schema.

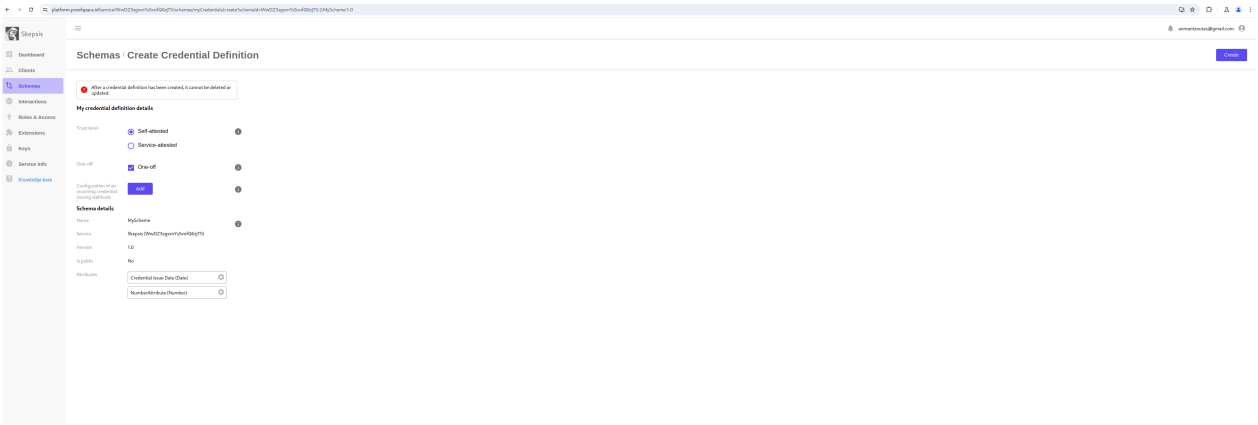


Figure 4.4.5: Creating a new Credential Definition

First, you must decide between 'self-attested' and 'service-attested' credentials. Self-attested credentials allow users to manually input the attributes themselves. Service-attested credentials, on the other hand, are entirely managed by the service, with users simply receiving the completed credentials. For self-attested credentials, you also have the option to enable 'one-off' credentials, which require the user to fill in the attributes again each time they engage in an interaction that requires this credential.

Additionally, you can set up an incoming credential issuing webhook by specifying an authentication method and, optionally, an IP prefix for the request. This feature enables external applications to issue this specific credential. In order to trigger this webhook, an appropriate interaction must exist.

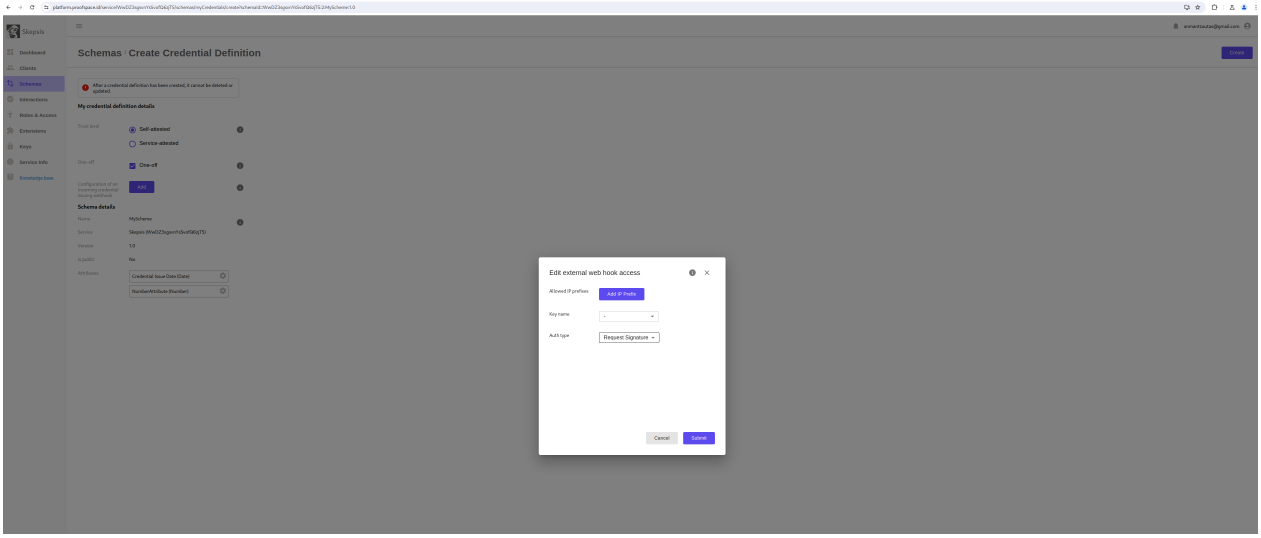


Figure 4.4.6: Configuring an external webhook

Once all options are set according to your needs, finalize the process by clicking "Create" located in the top right corner of the page.

4.4.1.2 Application's Credential Definitions

Before delving into the specific credential definitions utilized in our application, it's important to note that while the schemas vary, the configurations for all definitions remain consistent across all five credentials. All are set as service-attested, reflecting the nature of the data they handle -such as grades and registrations- which requires a high degree of control to prevent any manipulation by students.

Additionally, the configuration for the incoming credential issuing webhook is standardized; it requires that all requests be authenticated with a specific RSA key, which is securely stored on the application's backend server. This security measure ensures that only authorized actions trigger the issuance of credentials. Details on how these RSA keys are configured, and the process for issuing credentials via these webhooks, will be explained in subsequent sections.

University Registration The University Registration credential marks the commencement of a student's academic journey at the university. Each student receives this credential only once, triggered by their action of scanning a QR code displayed on the web interface of the application. This credential serves as an official record of enrollment and includes two key attributes:

- School: A text-type attribute that records the name of the university the student is enrolling in.
- StudentID: Also a text-type attribute, this records the unique student ID assigned upon enrollment.

In addition to these specific attributes, each University Registration credential also includes a standard attribute for the issuance date, which records the exact moment the credential was created and issued. This attribute is common across all different credentials.

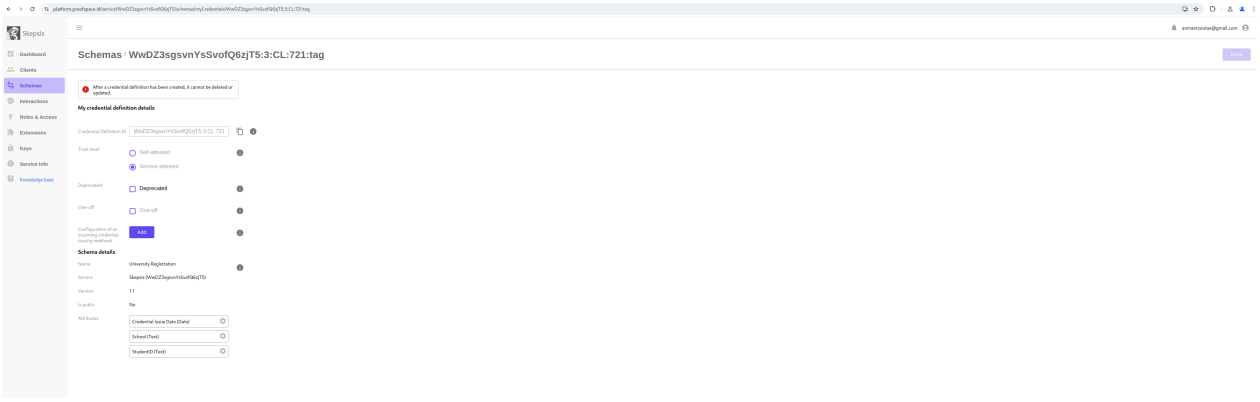


Figure 4.4.7: University Registration Credential Definition

Semester Registration The Semester Registration credential is designed to track and validate a student's continuous academic progress. Each student receives a separate credential for every semester they register for throughout their university tenure

Apart from the standard issuance date attribute found in every credential, the Semester Registration credential includes one specific attribute:

- Semester: This attribute is of type number and denotes the sequence of the semester for which the student has registered. For instance, it would be '1' for the first semester, '2' for the second, and so forth.

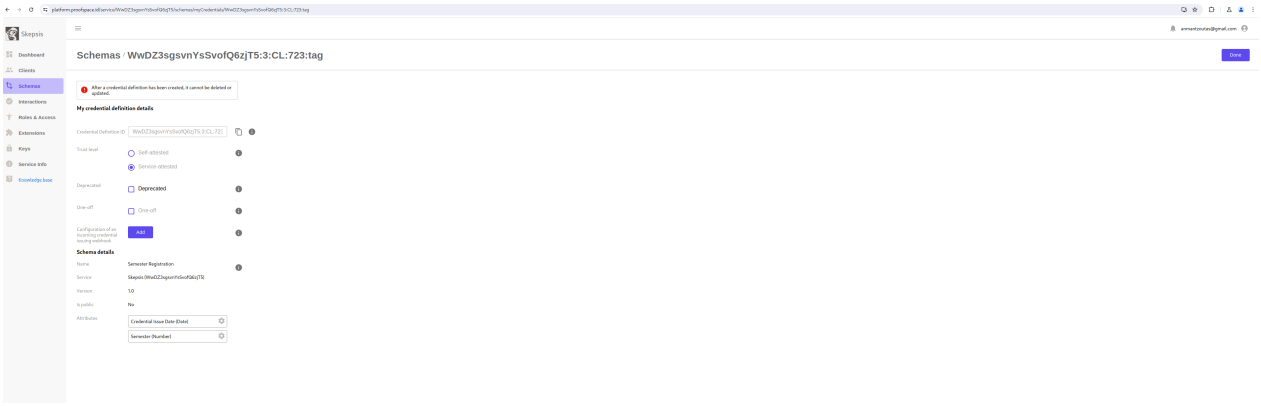


Figure 4.4.8: Semester Registration Credential Definition

Course Registration The Course Registration credential is issued to students for each subject they register to attend classes and take exams. This credential is crucial for managing course enrollments and ensuring accurate academic records. It includes two specific attributes:

- Subject: A text attribute that records the name of the subject the student has registered for.
- Semester: A numeric attribute indicating the semester during which the student registered for the subject. This helps track instances where a student may register for the same subject in different semesters.

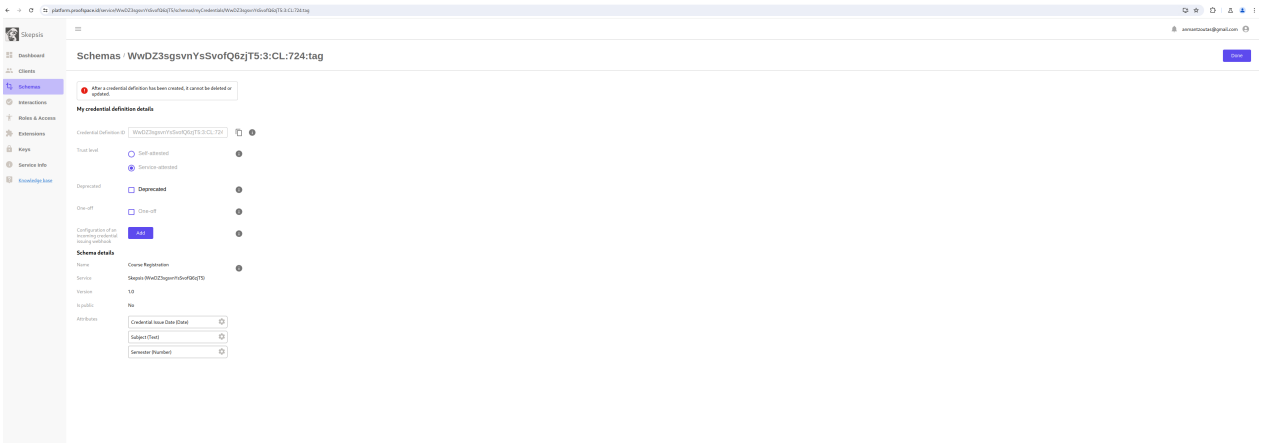


Figure 4.4.9: Course Registration Credential Definition

Grade Credential The Grade Credential is crucial for documenting the academic performance of students in specific subjects. This credential is issued to students upon receiving a grade for a subject and includes two key fields:

- Subject: A text field that specifies the name of the subject for which the grade is assigned.
- Grade: A numeric field that records the student's grade for the subject, with expected values ranging from 0 to 10.

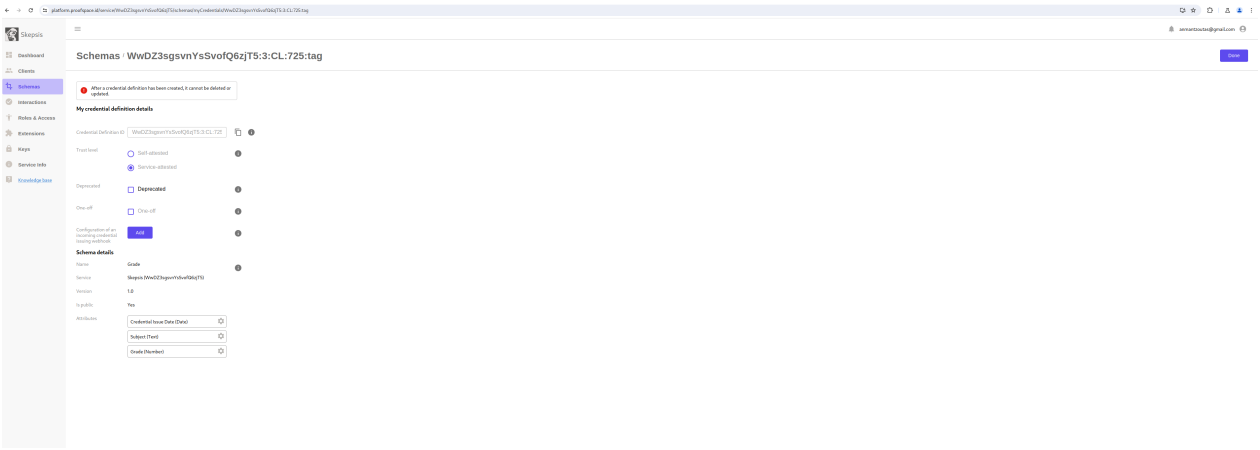


Figure 4.4.10: Grade Credential Definition

Degree Credential The Degree Credential serves as the definitive record marking the completion of a student's academic journey. It is issued once a student finishes their studies and is ready to claim their degree. This credential encapsulates the culmination of the student's academic efforts and includes two important fields:

- School: A text field indicating the name of the university from which the student is graduating. This confirms the institution conferring the degree.
- Degree Grade: A text field that records the final cumulative grade of the student, expressed as a decimal value within the range of 5 to 10. This grade reflects the student's overall academic performance during their course of study.

This credential is a significant academic milestone, providing a formal and verifiable record of the student's degree and academic standing upon graduation.

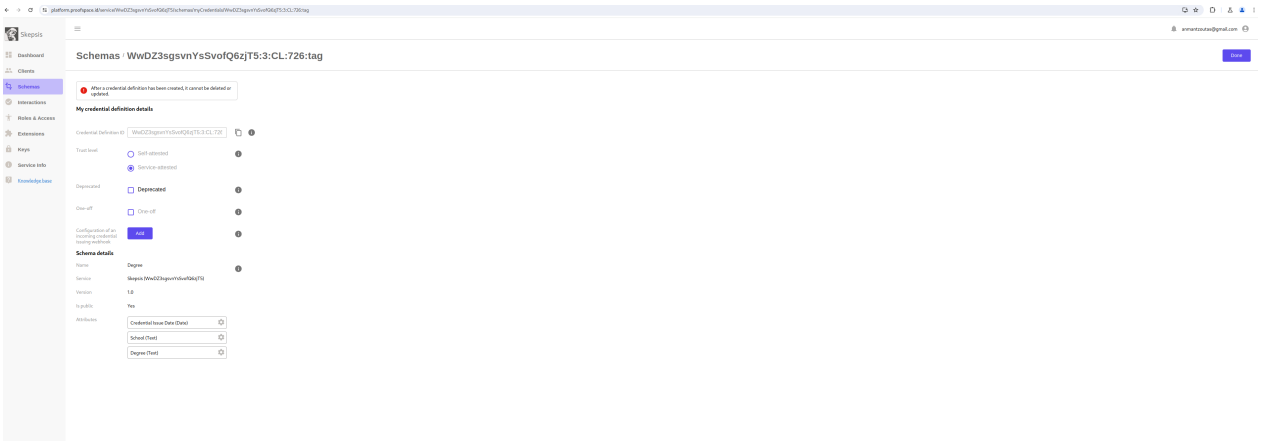


Figure 4.4.11: Degree Credential Definition

4.4.2 Interactions

In addition to adding the credential definitions, establishing a functional ProofSpace setup requires defining specific interactions. Interactions are workflows that specify the conditions under which a credential can be issued. These predefined flows are essential as they ensure that credentials are only issued when certain criteria are met, maintaining the integrity and relevance of the data within the credentials. This step is vital for automating the credential issuance process.

4.4.2.1 Adding a new Interaction

To create a new interaction in ProofSpace, navigate to the "Interactions" page within the platform. Once there, click the "New Interaction" button located in the top right corner of the interface. This action will initiate the process of setting up a new interaction, allowing you to define the specific conditions and triggers for issuing credentials based on your system's requirements.

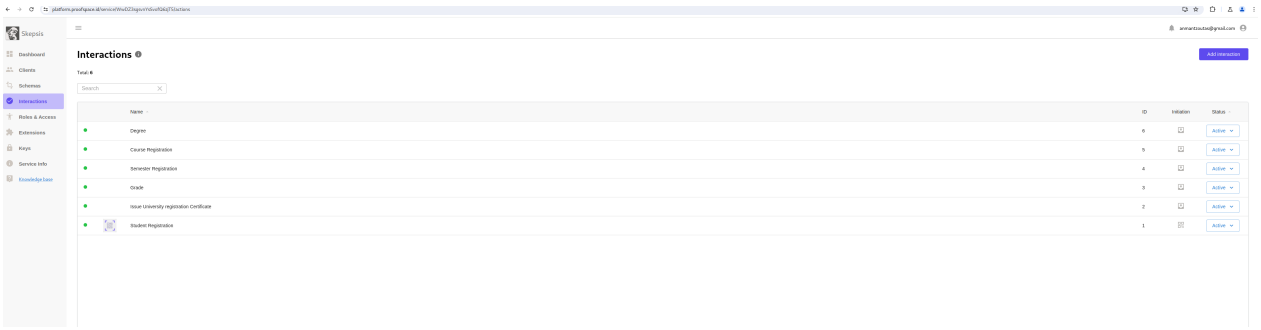


Figure 4.4.12: Interactions main page

Once you click the "New Interaction" button, you will be redirected to the interaction creation page. This page is organized into four tabs: Interaction Info, Details, Instances, and Extensions. For setting up a basic interaction, you will primarily need to work with the first two tabs: Interaction Info and Details.

In the Interaction Info tab, you are required to provide several pieces of information:

- Name: Enter a unique name for the interaction to easily identify it in the system.
- Description: Provide a concise description of what the interaction does and any important details that differentiate it from other interactions.
- Notification Messages: Configure three types of notification messages that will appear as pop-ups in the mobile ProofSpace application:
  - Success Message: This message is displayed when the interaction is successfully completed, informing the user of the successful outcome.
  - Pending Message: This message appears when the interaction is still in progress or pending completion, keeping the user informed about the status.
  - Failure Message: In case the interaction fails, this message will be displayed, providing feedback on the unsuccessful attempt.

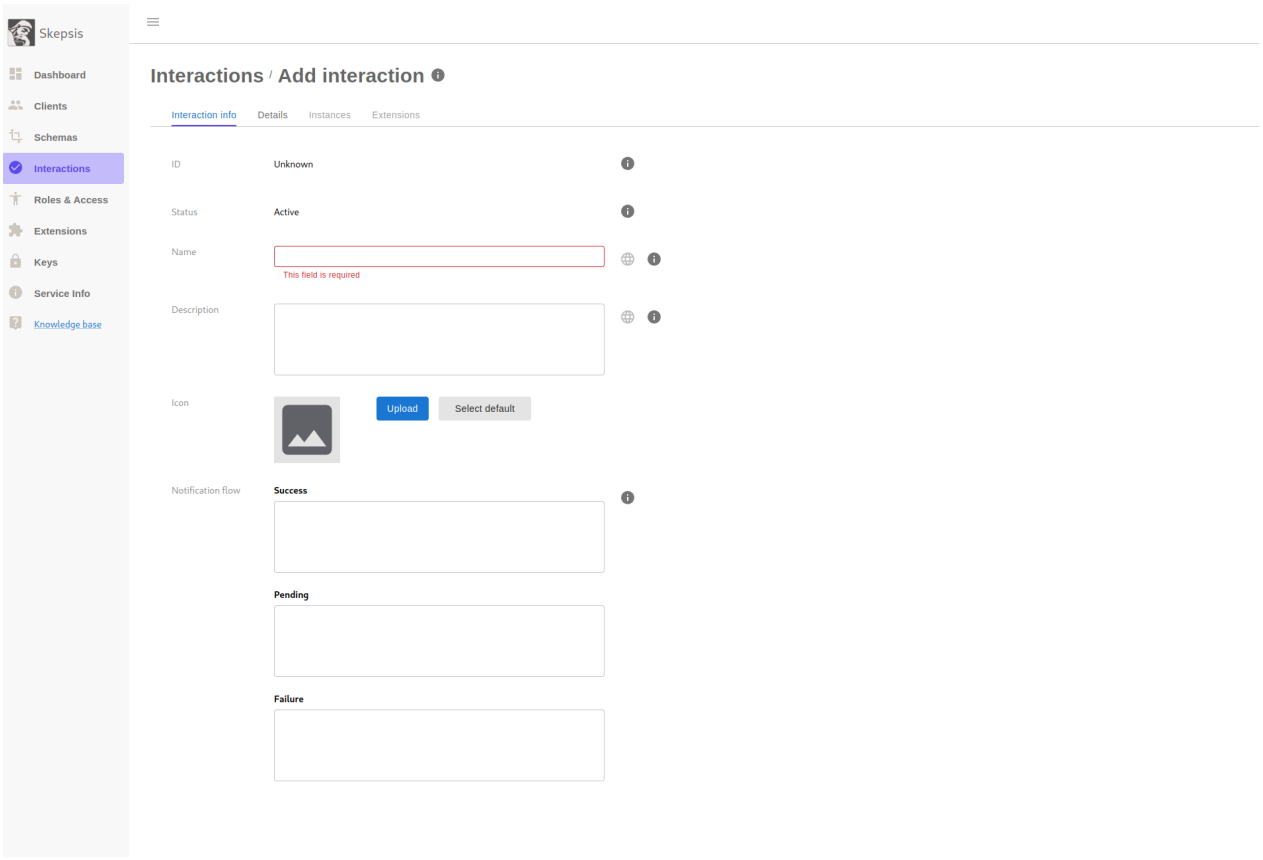


Figure 4.4.13: Interaction Information

Upon navigating to the Details tab, you will set up the more technical aspects of the interaction. This includes specifying how the interaction will be initiated, managing concurrency, and configuring additional settings and outcomes.

Initiation Media First, define the initiation media to determine how the interaction will be triggered. There are three options:

- QR Code: Triggered when a consumer scans a QR code using the ProofSpace mobile app.
- Button: Initiated by the user pressing the "Proceed" button on the interaction screen within the mobile app.
- Incoming Push from Service: Activated by an API call from an external service to the ProofSpace API.

Concurrency Type Next, select the concurrency type, which dictates how the interaction can be executed:

- Once: The interaction can be triggered only once per client.
- Sequenced: The interaction can occur multiple times by the same client but only sequentially; a new interaction can start only after the current one is completed and marked as "Complete."
- Concurrent: Allows the interaction to be initiated simultaneously by the same client.

Store Interaction Event Consider whether to check the "Store interaction event" option. Checking this box will save all information associated with the event. If unchecked, only limited data such as the occurrence of the interaction, the names of shared credentials, and certain required credentials with trust level "Service attested" will be visible on your dashboard for 24 hours before being permanently deleted. Information regarding issued credentials will be immediately deleted after the interaction completes.

Required and Issued Credentials

- Required Credential Definitions: Specify which credential definitions a user must possess to initiate and complete the interaction.
- Issued Credentials: Define what service credentials will be issued to the user if the interaction is successful.

Webhook Configuration Lastly, configure a webhook to be invoked upon the completion of the interaction. This webhook can be used to automate further processes or notify other systems of the interaction's outcome.

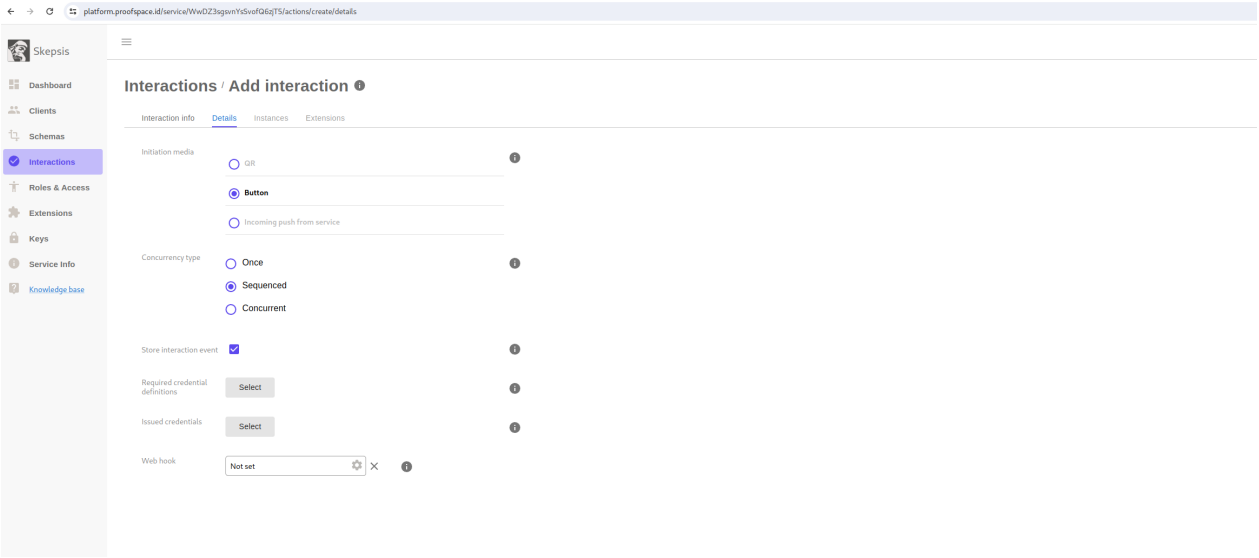


Figure 4.4.14: Interaction Details

Once all the fields are configured, simply navigate to the "Interactions" page by clicking on it in the left navbar. You should be able to see the newly configured interaction listed there. The interaction is saved automatically once you have entered all the necessary details, ensuring that your settings are preserved without needing to manually save before exiting.

4.4.2.2 Application's Interactions

Our application utilizes six different interactions to manage user credentials effectively:

- QR Interaction: The initial interaction is a QR code-based mechanism designed to capture the connected user's Decentralized Identifier (DiD) and their notification ID. This information is crucial as it establishes a connection with the user's ProofSpace account, enabling the system to send future credentials directly and securely. The notification ID serves as the user's consent, allowing the system to transfer credentials to their account and them receiving push notifications.
- Push by External Service Interactions: The remaining five interactions are structured identically and categorized as "Push by External Service" interactions. Each of these is responsible for issuing one of the five different credentials outlined in the system (University Registration, Semester Registration, Subject Registration, Grade Credential, and Degree Credential). Since the setup for these interactions is the same except for the specific credential issued, we will detail only one of them as a representative example.

University Registration Interaction The primary goal of the University Registration interaction is to link the user's ProofSpace app and DiD with our service, securing the user's consent to retrieve notifications. It also allows us to capture the student's DiD and store it in our backend's database, which is crucial for issuing

new credentials as the student progresses through their academic journey. The interaction is configured as follows:

- Initiation Media: The interaction is initiated through a QR code, which the student scans using the ProofSpace app. This QR code is prominently displayed on the web UI the first time the student logs in, a process detailed in the frontend section.
- Concurrency Type: The concurrency type for this interaction is set to "Once," meaning it can only be triggered a single time for each user, which aligns with its use for initial registration purposes.
- Event Storage: We opt to store the interaction event to ensure a record of this foundational activity is maintained, which aids in auditing processes.
- Required Credential: The "Notification ID" is the required credential for this interaction. It is essential as it represents the user's consent for future notifications.
- Certificates Issued: This particular interaction does not issue any certificates directly. Instead, its successful completion triggers a sequence of events facilitated by our backend.
- Webhook Configuration: Upon completion of this interaction, one of our application's endpoints is triggered. This action retrieves the student's DiD from the interaction data and stores it in our backend database. Following this data retrieval, the same endpoint makes a call to the ProofSpace API to issue the University Registration certificate.

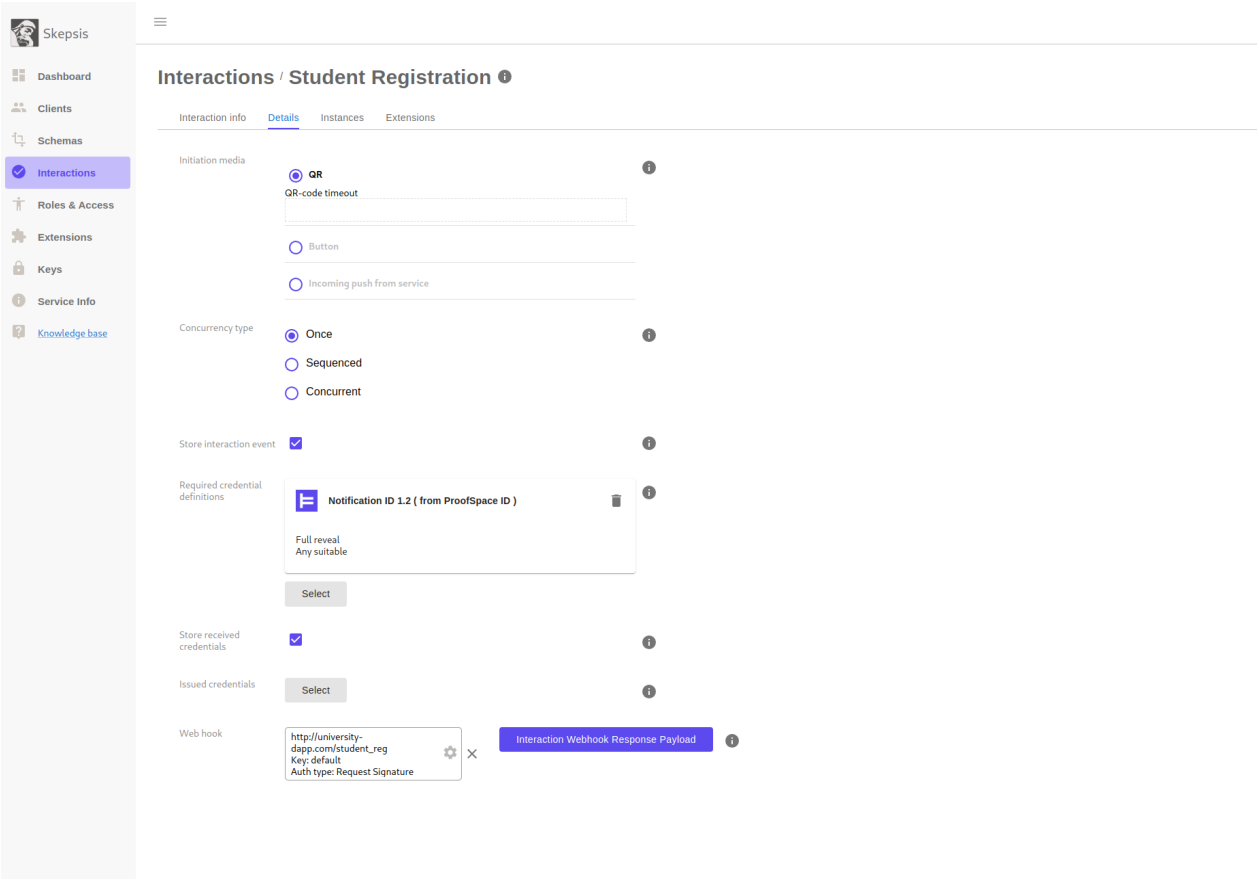


Figure 4.4.15: Student Registration Interaction Details

Course Registration Interaction The Course Registration Interaction is designed specifically to issue Course Registration credentials to students who enroll in subjects. This interaction functions identically to the other four, differing only in the specific credentials issued. It is configured as follows:

- Initiation Media: The initiation for this interaction is an "Incoming push from service." Our backend system triggers this credential issuance when a student successfully enrolls in a course, based on validations performed within the application.
- Concurrency Type: Given that the initiation media is "Incoming push by service," there is no need for a concurrency type. Each trigger is treated as a separate event, appropriate for actions directly initiated by backend processes.
- Event Storage: We opt to store the event details of this interaction too.
- Required Credential: There is no required credential for this interaction because our backend performs comprehensive checks to confirm a user's eligibility. This includes verifying conditions through the application's on-chain components and smart contracts. If any prerequisite smart contract calls fail, indicating the user does not meet all requirements, the process to issue the credential is halted.
- Certificates Issued: For this interaction, the "Course Registration" certificate is issued. It is important to note that while the mechanism is the same, the specific credential issued varies among the different interactions—each tailored to the respective academic activity (e.g., semester registration, grade issuance).
- Webhook Configuration: No webhook is required for this interaction. The process is entirely managed internally by the backend, which directly handles the issuance of credentials based on successful course registration and validation.

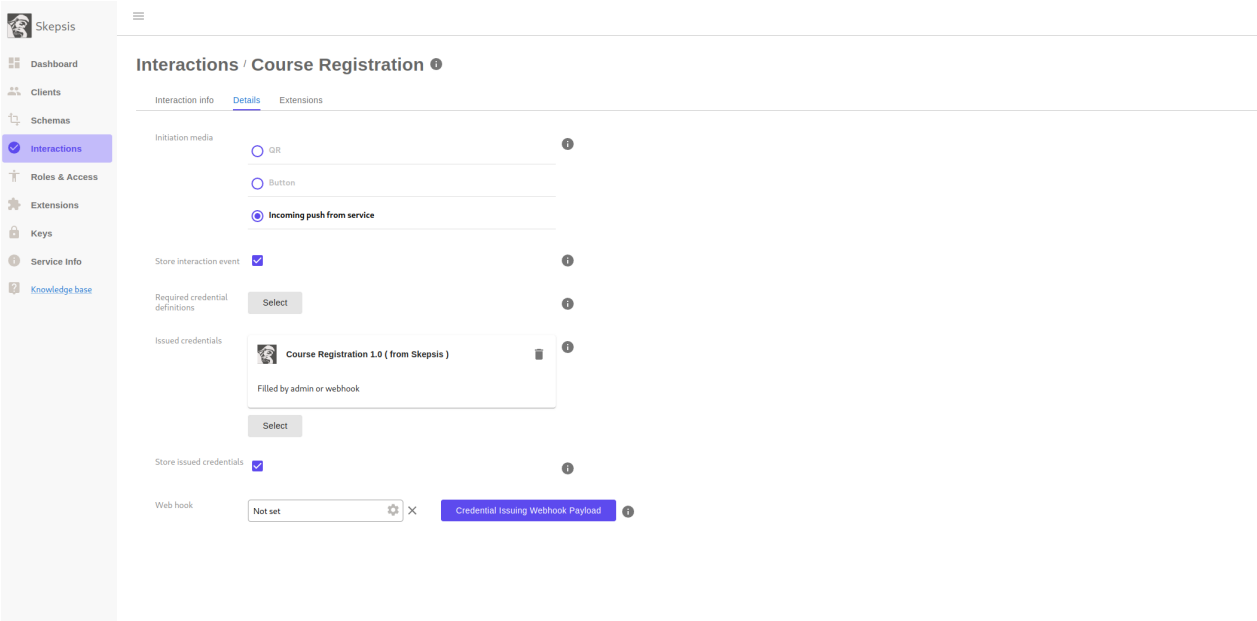


Figure 4.4.16: Course Registration Interaction Details

4.4.3 Issuing a Credential via the API

In this section, we will demonstrate how to issue ProofSpace credentials using the ProofSpace API.

4.4.3.1 Configuring Webhook Keys

To ensure that only authorized entities can issue credentials from our service, ProofSpace utilizes an authentication mechanism based on asymmetric cryptography, specifically RSA. This security feature requires that requests to the ProofSpace API be signed with our private key. To set up the appropriate keys, access the "Keys" section on the ProofSpace dashboard and click on "Add" to begin configuring a new key. Then, configure the following options:

- Input Key Name: Assign a descriptive name to your key for easy identification.
- Select Key Type: You have two options for setting up your key:
  - RSA Public Key: If you already possess an RSA keypair, select this option and paste your public key into the provided text box.
  - RSA Keypair: If you do not have an RSA keypair, select this option and click the "generate" icon. The system will automatically generate a new RSA keypair for you. It is crucial to securely copy and store the RSA private key displayed in the relevant box, as it will be necessary for issuing credentials. Note that while this option is more straightforward, it may carry certain security implications.
- Set as Default for Webhook Requests: If you wish for this keypair to be the default authentication method for incoming webhook requests, select the "Set as default for incoming webhook requests" option. If your use-case requires different keypairs for various credentials, you may choose to uncheck this option. However, for simplicity and consistency, keeping it checked is advisable for our application.

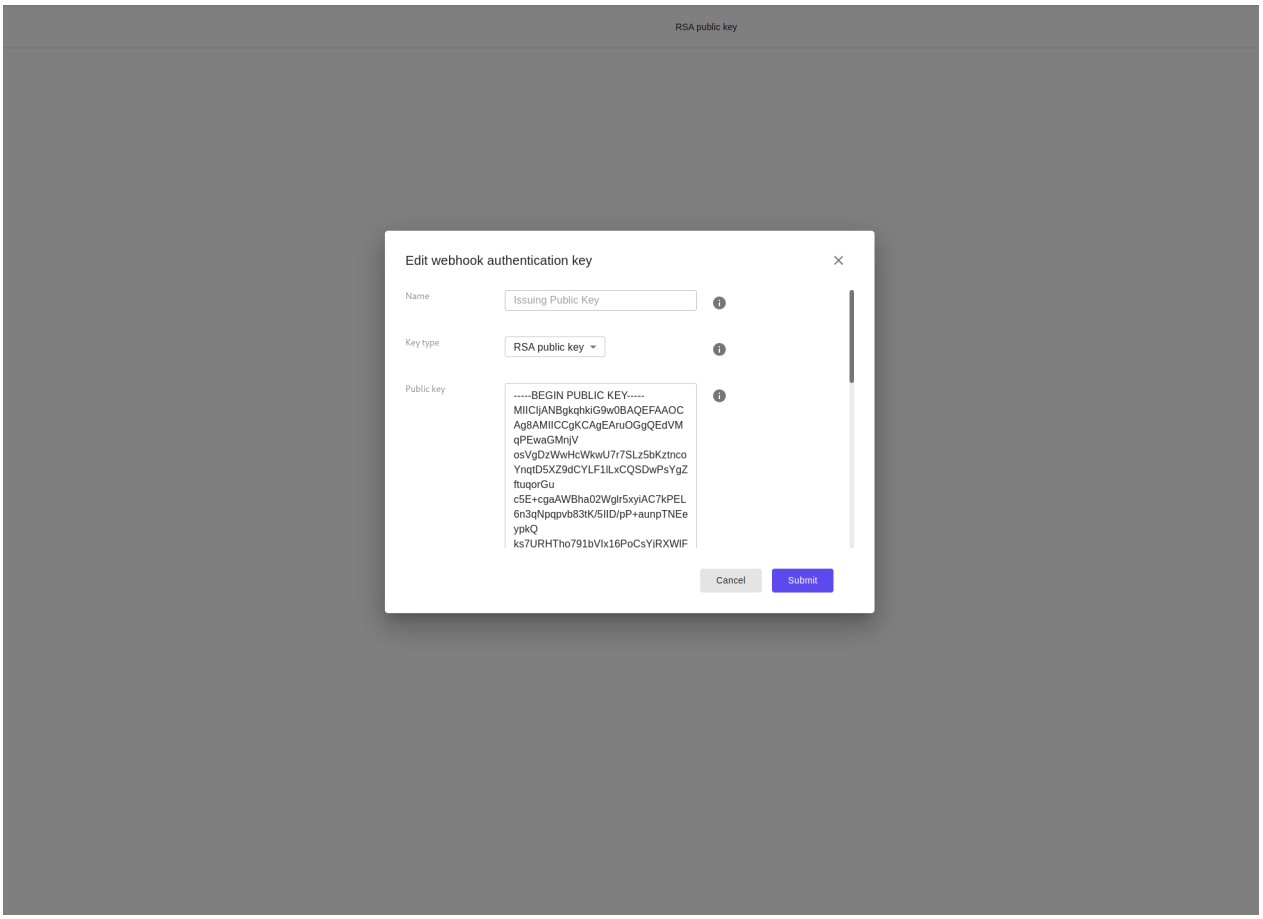


Figure 4.4.17: Adding a new Key

4.4.3.2 Invoking the ProofSpace API

In this paragraph, we'll explore how to invoke the ProofSpace API to issue a University Registration certificate from a Node.js environment. This method is applicable to issuing any type of credential using the same approach.

To issue a credential via the ProofSpace API, you must send a POST request to the specific URL provided by ProofSpace. First, obtain your "serviceDid" from the Service Info page on

your ProofSpace Service's dashboard. In this case, the "serviceDid" is WwDZ3sgsvnYsSvofQ6zjT5, so the target URL for issuing credentials will be: 'https://platform.proofspace.id/service-backend/v1/service/WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued'

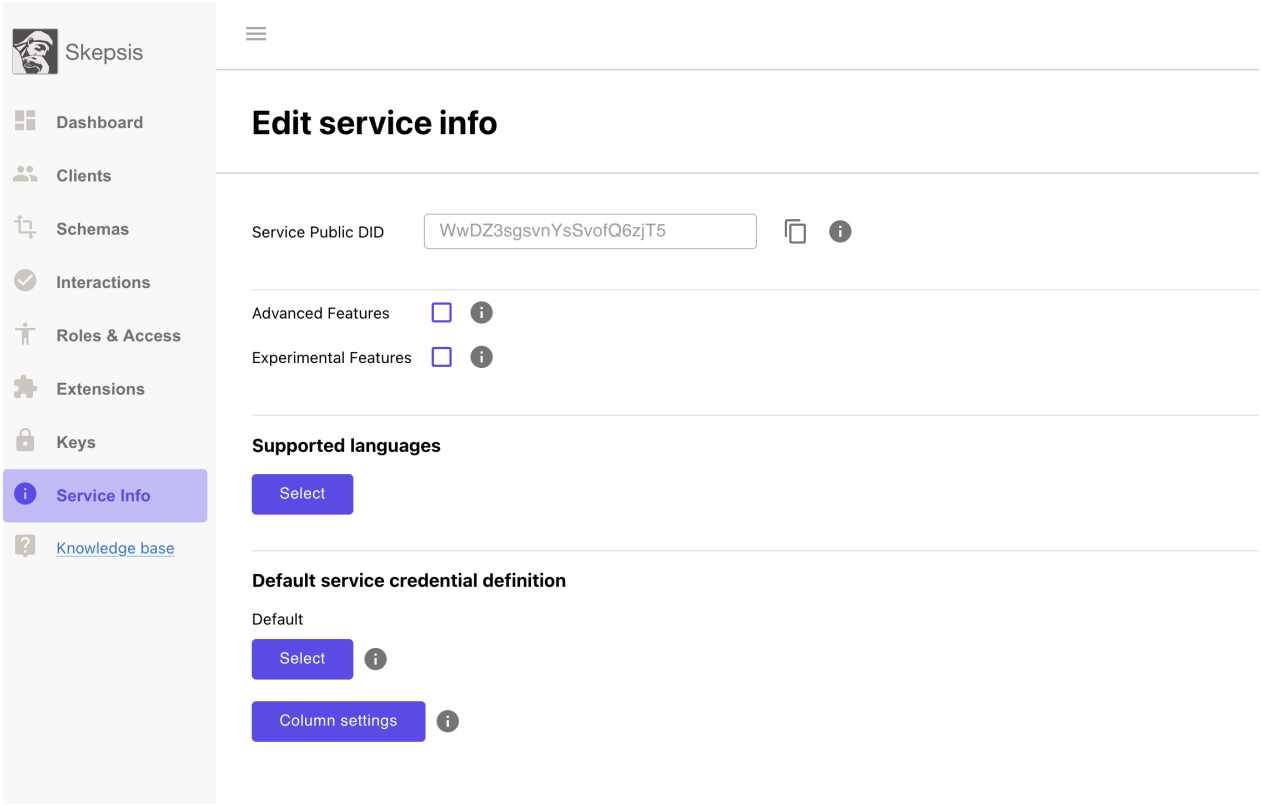


Figure 4.4.18: Service ID retrieval

When crafting the body of the POST request to issue a credential via the ProofSpace API, the structure should be the following:

```
{
  serviceDid: string;
  subscriberConnectDid: string;
  credentials: WebhookCredentialValuesDTO[];
  issuedAt:
  subscriberEventId: string;
  actionTemplate?: string;
  actionParams?: Array<NameValueDTO>;
}
```

Here's a breakdown of the most important fields:

- serviceDid: This is the Service's DiD, which you have retrieved in the previous step from your ProofSpace Service's dashboard.
- subscriberConnectDid: This field must contain the DiD of the user who will receive the issued certificate.
- credentials: Here, you specify an array of WebhookCredentialValuesDTO, which defines each credential being issued in this request.
- issuedAt: This is a timestamp indicating when the credentials were issued.

WebHookCredentialValuesDTO The WebHookCredentialValuesDTO is an object that includes all the necessary details about a credential being issued. Here's a detailed explanation of each component within

this object:

```
{
  credentialId: string;
  schemaId: string;
  fields: { name: string, value: string }[];
  utcIssuedAt: number;
  revoked: boolean;
  utcRevokedAt?: number;
}
```

- credentialId: This is the unique identifier for the credential being issued.
- schemaId: Represents the ID of the schema that this credential conforms to.
- fields: This is an array of key-value pairs where each pair corresponds to an attribute defined in the credential's schema. Each name in the pair matches a schema attribute, and value is the data for that attribute.
- utcIssuedAt: The UTC timestamp marking when the credential is issued.

In order to retrieve the 'credentialId', navigate to the 'Schemas' page of the ProofSpace dashboard and copy the desired one from the "Credential ID" column.

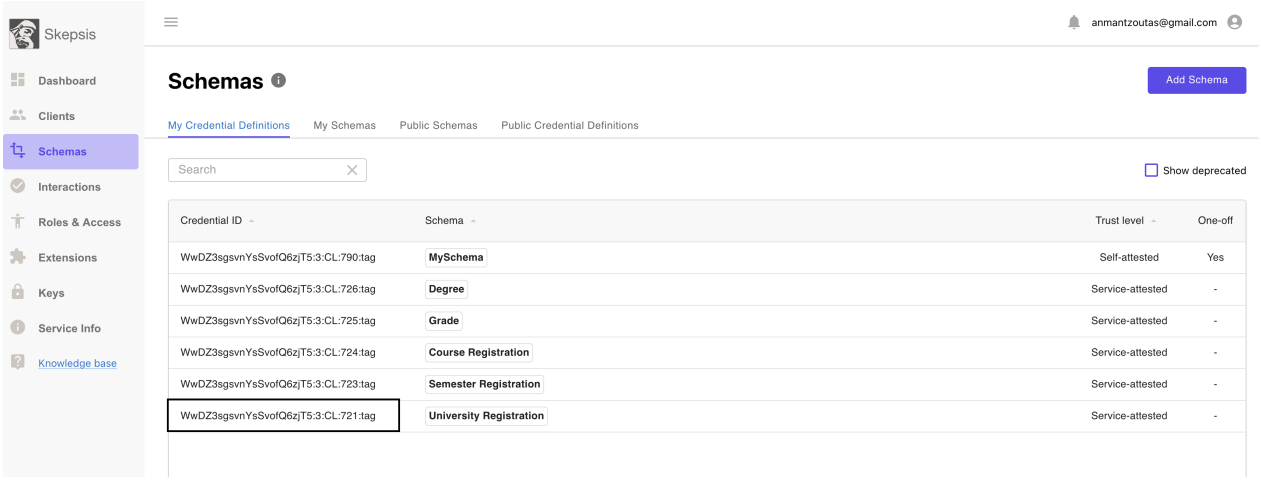


Figure 4.4.19: University Registration Credential ID

To retrieve the 'schemaId', navigate to the 'My Schemas' tab on the 'Schemas' page, click on the schema you wish to use, and then copy it from the information page.

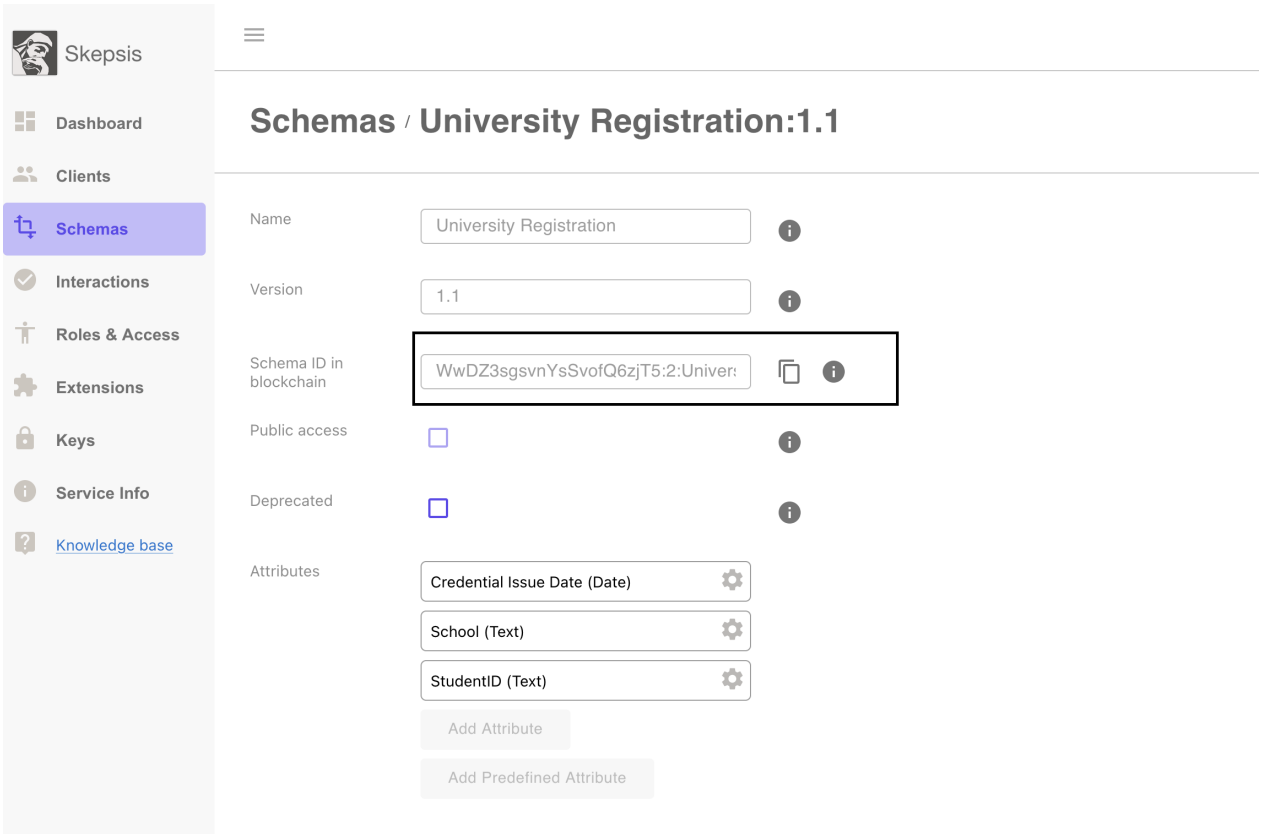


Figure 4.4.20: University Registration Schema ID

For this demonstration, we'll use specific attributes for the credential issuance process. The attributes include the school name "ECE NTUA" and a student ID, "el17108", which is my personal identifier. Additionally, we will use my ProofSpace ID as the "subscriberConnectDid". Based on this setup, the following is the JavaScript code snippet we have developed so far:

```
const url = "https://platform.proofspace.id/service-backend/v1/service/
    WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued";
const serviceDid = "WwDZ3sgsvnYsSvofQ6zjT5";
const credentialId = "WwDZ3sgsvnYsSvofQ6zjT5:3:CL:721:tag";
const schemaId = "WwDZ3sgsvnYsSvofQ6zjT5:2:University Registration:1.1";

const credentialValues = {
    credentialId,
    schemaId,
    fields: [
        { name: "School", value: "ECE NTUA" },
        { name: "StudentID", value: "el17108" },
    ],
    utcIssuedAt: new Date().getTime(),
    revoked: false
};

const request = {
    serviceDid,
    subscriberConnectDid: userDid,
    subscriberEventId: "",
    credentials: [credentialValues],
    issuedAt: new Date().getTime(),
};
```

Following these steps, we need to set up the request's headers. First, we set the "Content-Type" to "application/json". Next, we configure the "X-Body-Signature" header to authorize the request. This

involves the asymmetric cryptography we discussed earlier. We serialize the request body into a string, sign it using the RSA private key we created in the previous subsection, and encode the signature into base64. Finally, we add the encoded signature to the "X-Body-Signature" header. Please note that for the sake of this demonstration, we'll include the private key in the code in plaintext. However, this is not a good practice, and it's advisable to avoid doing so in real-world applications. The code is as follows, using the "crypto" built-in library to sign the object:

```
const headers = {
    'Content-Type': 'application/json'
};

const binaryBody = Buffer.from(JSON.stringify(request), "utf-8");

const private_key = `-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAtsEaCBfGaRhWfkLuiKvK8InKgZyl0od1DcNAlVi6YaEmEruS
e8lPJfewXR3uxIg3SYOAGI6LYFFIJxAMcvHYEm0axW5lo4ov1hYRMA76tl/eo1Kp
3Cp7Ai8W9GshcQzbFwPX9UjbijHkRkwCIpW+iK/aipj/nBg1LJsOR4b9hpbB92Hz
+b4nviGBD5CA0vzbyDhLq/4oQbyOvAsAFAqKZEqBHq4s+WcKDHXFkvsmxnzexAlH
JtGjhFU+B3myXPMoMEHUInoKW0cSez5/WmkUOrPcNu4jmVmpjnVek+7s9q8F+blA
RaI4WKUUM+MA3sBGkMt3QeyKOm0L9JfA5C3q0ZmJ4xkJvaK0zCdml8DbgKasIy/L
JMXkQZyOZGlg2YsjcLK3lsElefAQ8d4/J4wSF+I/0uVKV/vxzVBx9kGpUGJQu9l
0pGsxaQyZ94wUBiJybZqx8EUQmLIH7ONjSAGFHEISPLOlqRihJRz3zVdIKn04k1I
ftqk477mz5YJD5zVq9w9qx77oCzfH34N8pU/vTUaFmoWHhdnrb8slXaATOMejksA
kCK/wqUhYnGol8cNq92uTC/IdcEbBkHyHyQ0HV88R1myoqY9bg4CbZzJDrtCzAUH
mN5fVCUVh7WL9Pt5pskD3zPPv77gxzJYLI+/9kOgwCXHrk9kGnFwgjsJHcUCAwEA
AQKCAgABsPbLNWEeLt36DCeF90UJBdVsbqUwNhW0u2ZZG3maFrIyAS9hWuMTIGAu
9Gx1wCb21LM2RvduPEJVLjZXR/qubPkO868YHNrFee3Nyo2ya1+1ljnBMllyEzp8
p5MnJy3N4Di66eIyPLT2m5R3enhJboswtirDaIvxbjvrGIo3GSURY4Qs/wACJ/A4
UyeGDFYCW9cDb/YyaYrOnbFx/gj61Z+tR4JeCoCSzfI2ooFg29355XYVm2IOgvVw
f0VRy+RG0g0jBosa/iTjJBhsn0VUj9qNXZXRwP0rNtXZu5/QxpzhC0BXhKFN3LU7
p7Gl3hSpGWDwqL48vS6bpF3iHrSqGN+d+WtNGCMWSNcq9b8Xp0ft7kx3XHMKYU/s
2+HtMO3jp0fxr4i7tG/DZpaahLA/HEIWG4AK5CEKOL0UDYtqwyoNedoPjhkr5nxf
Kx4BX40b2DG83hU5imsp2zP7XAkmiIpNYNdXosFrKoWcBU3vCKus9/MGS34ZNOR4
9LhRA1I478qfsUXcmx1PcmIRyyobanZsyzxad9n/yxDxCZ50nnxCMvJUWSRU9n8l
3dwafbgMV/ispxQXc9oTof61LCFVlrukjNYIcz+03V31q2/tp+dL2UNZWObT72ED
Aw8SPy6yB5BYpNU1BdC+jPmfDuDlnUPNSY8hBqV6otlXNIDCnwKCAQEA9GctrVnB
pZHRMwPZfL9VK7AiWkLdoWThtmuNHRZm0XhepYH8xaxlx6myQtkx3QYdJc1gpORC
BEoP8DtDdfOAZ4LGjic1GbXyHpI8bdgGf4H11EXZPr+PlAGLSyW8Ie+CYHfrKpYL
uHLaOIrSx4k+Fbj4B4obNEg/s4QSzQichI+UiwGmMuYx5KBlBXHnBWdN0+ICwbJq
Q3XRffVi0Xc0wDJjESXyVdqzMVnb18bHRkjwxTx2tgFZEBVYLeWtX3YTeQkzP/Yx
xIvzzXVtTrZa/DyD/bXs9ea9oJ0q80ABmyWUESTFrzrbYnAQG2UWYyPb6gv2Lw1D
1w5zTb5soZL0xwKCAQEAv20Pu8TfUABm1Imo/IsDA0koNB/0BpIEyfAeFHQ0UPR6
3rzli90Dn96PBfmlyOTPQir1RnSxpcxQW6eyU1ZGKasLFoS41mrnFTvbdOof/2Vd
kbNcwFb3wnLjm1YP4ImP36ugM4z4n++y03Uqbxz5wAsawq+5U+M0Z4miUrNRsPjz
OPBCl25VLXo3wFvP8qhIToDHnO5GNUBPCWZttlgi7Lwu8C/BRLGFSfIBAU7HFLIh
3foeLh6u+I4F1/ew8KMkrHxnQxDzMI8HaVtjwGtNdZmyXwf4M4ekJfGMowbPLh+k
3Ud68i2KPAmR2k50uS+gt/RfBYlqxqGCs22urK51EwKCAQAhaQVugCmft6pQfgM6
MXG0MjTP//oKOR8977/vrQNLOVg+6eEjKKkVPVZ4GvbdDxvm8gd3Ixj/Oxyoq4FS
W4I/JDp+bk+s/xllInqOTYy924c+iHnlQQygoAFpuXPZc8Nv6pTEUpSxl0nLlvav
fkx1bb7CURfc8s96e+CSFM5ItNQKmKQeqydA6MhGkUaSj4etImVcE0bBiBamUGg8
2D3m1DyEuJpDME10vqL2AV92dTwsylI0GUUSlMhkJdSoixyoXCCcasxs4PDoMlhJ
/VW8y40Dnt4Q04UysqmPbT2K9v0hxtQbW0FoWIdrJ0uDLADBIvd5gSe3+2q5wwFY
qR07AoIBAQCQ9qh5i8eJMCSvQ2iDh0pKzLXzc1c7Oo0B2CMPXkjYHayGzQoQdtjt
On8BCWNw1UXXV+zS5EA7zJP2NPKohacwmuUUsoG/bsl8nQ2+ZibBJCRN2ESfK70I
tjSDSBNR+RrJSZFSOKl3hPApjosKBilkM4SadrSkQGVJG0Of2Ug9+ulF7zyerOE4
RVu/Akhh/Op0+LUxtIfaf87a0YS4F9qq6H+rSJ7VxJGpw+ja8qt0kslu7qC4UGg0
tmjaeiq2sPVu4e7ZLRbdaRM1yA2shBXuvuwDZ2egrlGyok/cjaWLyk7xe9YxXCdY
++5GIi/mpZjSde8XpE8GCP2Pn7ot7xedAoIBAQDEnlbgWPS2tldLoVFG4ega9x09
G/SVen5kkmLRaBNLV6rGRnCQoYkosUdbKMUsTb8WJGJhIllu5u0SgbaYyxrG63ob
FdDgVjIPJneL1xnX3s6qP9084k2cBqwE3LtNfVK6eWYEwXvaeHQN/Ur3KbJQ3tDi
LkNpzd4ysqRmy01UOIQxWG0nnM+LDRjhmong8u+HAeiYmZ9WZ2D06nd5dbq19FeT
F0dJYc4jhBZQoW7oaaNAMJ4JY/ViM1OISzjwaWcanQ0B1pH0yk0oSReWd82BFQfb
sLC/A+iBdA4dNfB9fnB+zrF0Php+2mpMZOnNyo5Nvm2N08wVOYk5lho0psEj
-----END RSA PRIVATE KEY-----
`;

const key = crypto.createPrivateKey({ key: private_key });
const signature = crypto.sign('sha3-256', binaryBody, key);

```

```
63 headers['X-Body-Signature'] = signature.toString('base64');
```

Finally, we can proceed to post the request to the ProofSpace API using "axios":

```
axios.post(
    url,
    binaryBody,
    {
        headers: headers
    }
).then((reply) => {
    return reply.data;
}).catch((err) => {
    throw err;
});
```

The whole code for this demonstration can be found in Appendix B.

Chapter 5

Epilogue

In this thesis, we have successfully demonstrated the implementation of a decentralized application) designed to manage academic processes such as grading, subject registration, and the issuance of credentials within a university setting. By leveraging blockchain technology and smart contracts on the Cardano network, we achieved a significant reduction in the administrative burden traditionally placed on university staff, particularly the secretary. The system automates many of the routine tasks that were once manual, reducing the potential for human error and increasing the efficiency of record management and verification processes.

However, the nature of academic institutions, which inherently require some level of central authority, imposes limitations on the extent to which our system can be fully decentralized. Critical actions within the DApp, such as minting degree tokens and updating important academic policies, still require the oversight and approval of the secretary. This centralization is manifested in the control over the admin key, which is necessary for signing all significant transactions. Consequently, while we have decentralized many components of the academic administrative process, the system still relies on central authority figures for crucial decision-making and oversight.

Furthermore, several operational aspects remain fully centralized, including the addition of new students to the system, the introduction of new subjects, and the assignment of subjects to teachers. These tasks require direct input and validation from administrative personnel, reinforcing the presence of central control in areas where accuracy and authority are paramount, while the related data are stored in a traditional off-chain database. Despite these limitations, the DApp represents a substantial forward step in the modernization of educational administration, providing a robust framework that significantly enhances the transparency, security, and efficiency of managing academic records.

5.1 Technical challenges

5.1.1 Plutus Mastery

Transitioning from an Ethereum background to the Cardano platform presented notable challenges in this project, primarily due to my unfamiliarity with Cardano's UTXO model and Haskell. Fortunately, the learning curve was mitigated by the use of tools like Aiken for on-chain development and Lucid for off-chain interactions, alongside invaluable educational support from the "Plutus Pioneer Program" by Input Output Global (IOG). These resources were instrumental in familiarizing myself with Cardano's unique programming model and significantly aided the successful implementation of the application.

5.1.2 Privacy Considerations

A significant challenge encountered was maintaining the privacy of academic records and student information while utilizing a public blockchain. The primary privacy mechanism in the application relies on the obscurity of contract addresses and token units; for example, students do not know each other's blockchain addresses or even their own in the context of the application. Despite this, the inherent transparency of a public

blockchain means that all transaction data is openly accessible. A sophisticated user, with enough time and resources, could potentially decipher the system's structure by analyzing transaction patterns on a blockchain explorer. This area represents the most crucial opportunity for improvement. Potential enhancements could include implementing more advanced cryptographic techniques to enhance data privacy or exploring the use of a private or sidechain solution specifically tailored for managing sensitive academic data.

5.1.3 Centralization Considerations

Addressing centralization risks in developing a decentralized system for university management presented a unique set of challenges. Given the centralized nature of universities, which includes multiple layers of authority such as teachers, deans, and presidents, a completely decentralized model may not be wholly appropriate. Certain stakeholders naturally require significant control over the system to manage and govern effectively. In response, we strategically opted for decentralizing components related to students' academic records, which benefit from blockchain's immutability to prevent loss and tampering. However, other aspects, particularly those involving administrative control like the role of the secretary, remained centralized. This hybrid approach balances the need for traditional administrative control with the advantages of decentralization, reflecting a pragmatic adaptation of blockchain technology to the specific organizational structure and needs of educational institutions.

5.2 Limitations

In the development of this prototype model, we had to accept certain limitations that reflect the current scope and design choices of the system:

Single Teacher per Subject The system is currently designed to associate only one teacher per subject. While it is theoretically possible to involve multiple teachers for a single subject, only the teacher whose key is registered in the relevant smart contract can add grades.

One Active Curriculum Unlike modern universities that may offer multiple curricula tailored to different student cohorts or registration years, our system supports only a single active curriculum at any given time. Consequently, all students must adhere to the same curriculum, and any changes to the curriculum affect all students simultaneously.

Mandatory Subjects Only The current infrastructure is designed to manage a set curriculum where all subjects are mandatory for obtaining a degree. While the system has the potential to accommodate optional subjects and various academic tracks, such adaptations would require additional modifications to the existing codebase.

Storage of Passed Subjects Only To minimize transaction fees, the system stores only the records of subjects that students have passed on the blockchain. Although this approach reduces costs, it may limit the complete academic visibility on-chain. However, a full record of all academic activities, including subjects not passed, is maintained on the ProofSpace app, ensuring comprehensive record-keeping off-chain.

5.3 Future Directions

In closing this thesis, we propose several avenues for future directions and potential enhancements of the application discussed, which could inspire future research

Enhanced Privacy Settings A critical area for future development is the enhancement of privacy measures. By incorporating advanced cryptographic techniques, such as zero-knowledge proofs or more sophisticated encryption methods, the system can better protect the confidentiality of student records. This would not only increase trust among users but also comply with stringent data protection regulations.

Overcoming Existing Limitations Another significant area for improvement involves addressing the restrictions imposed by the limitations outlined in the previous section. These improvements would not only align the system more closely with the complex needs of modern educational institutions but also expand its potential for broader adoption and impact.

Smart Contract Optimization for Scalability The current implementation faces challenges with scalability, particularly in handling a large number of students due to Cardano's computational limits. Future work could focus on optimizing and restructuring the smart contracts to reduce the need for splitting transactions into smaller segments. This optimization would help decrease transaction fees and enhance the system's capacity to manage larger volumes of data efficiently.

Migration to Production-Grade frameworks To further professionalize the DApp, migrating from SQLite3 to a more robust database management system (DBMS) and upgrading from the Pug templating engine to a more scalable frontend framework would be beneficial. Such upgrades would support larger user bases and more complex data interactions, facilitating smoother and more reliable application performance.

Personalized Degree NFT images The current design includes the same image on all minted degree NFTs. A promising future enhancement would be to integrate advanced image processing methods to automatically personalize these images based on the student's personal information. This personalization could include details such as the exact university name, the student's name, the date, and the degree awarded. Additionally, utilizing a service to upload these personalized images to IPFS and including the corresponding link in the metadata would further enhance the uniqueness of each degree NFT.

Increased Decentralization Finally, increasing the decentralization of the system is a vital goal. This could be achieved by replacing the admin key with a multisignature scheme, thereby distributing control among multiple parties and reducing the reliance on a single authority figure such as the secretary. Alternatively, certain administrative functions could be automated and managed by a smart contract, further reducing the need for centralized control and enhancing the transparency and impartiality of critical processes such as subject assignment and degree issuance.

Chapter 6

Bibliography

[1] Agarkar, A. A. et al. "Blockchain Aware Decentralized Identity Management and Access Control System". In: *Measurement: Sensors* 31 (2024), p. 101032. DOI: https://doi.org/10.1016/j.measen.2024.101032.

[2] Alizadeh, M., Andersson, K., and Schelén, O. "Comparative Analysis of Decentralized Identity Approaches". In: *IEEE Access* 10 (2022), pp. 92273–92283. DOI: 10.1109/ACCESS.2022.3202553.

[3] Bentov, I., Gabizon, A., and Mizrahi, A. *Cryptocurrencies without Proof of Work*. 2017. arXiv: 1406.5694 [cs.CR].

[4] Buterin, V. *Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform*. Accessed: 2024-03-04. 2013.

[5] Buterin, V. *On Public and Private Blockchains*. Research & Development. Aug. 2015.

[6] Cardano Foundation. *Cardano Documentation*. 2024.

[7] Cardano Foundation. *Cardano Website*. 2024.

[8] Casino, F., Dasaklis, T., and Patsakis, C. "A Systematic Literature Review of Blockchain-Based Applications: Current Status, Classification and Open Issues". In: *Telematics and Informatics* 36 (Nov. 2018). DOI: 10.1016/j.tele.2018.11.006.

[9] Chakravarty, M. M. T. et al. "The Extended UTXO Model". In: *Financial Cryptography and Data Security: FC 2020 International Workshops, AsiaUSEC, CoDeFi, VOTING, and WTSC, Kota Kinabalu, Malaysia, February 14, 2020, Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2020, pp. 525–539. ISBN: 978-3-030-54454-6. DOI: 10.1007/978-3-030-54455-3_37.

[10] Christidis, K. and Devetsikiotis, M. "Blockchains and Smart Contracts for the Internet of Things". In: *IEEE Access* 4 (2016), pp. 2292–2303. DOI: 10.1109/ACCESS.2016.2566339.

[11] Čučko, Š. and Turkanović, M. "Decentralized and Self-Sovereign Identity: Systematic Mapping Study". In: *IEEE Access* 9 (2021), pp. 139009–139027. DOI: 10.1109/ACCESS.2021.3117588.

[12] Delgado-Segura, S. et al. "Analysis of the Bitcoin UTXO Set". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (2019), pp. 78–91. ISSN: 0302-9743. DOI: 10.1007/978-3-662-58820-8_6.

[13] Dib, O. and Toumi, K. "Decentralized Identity Systems: Architecture, Challenges, Solutions and Future Directions". In: *Annals of Emerging Technologies in Computing (AETiC)* 4.5 (Dec. 2020), pp. 19–40. ISSN: 2516-0281. DOI: 10.33166/AETiC.2020.05.002.

[14] Do, T., Nguyen, T., and Pham, H. *Delegated Proof of Reputation: A Novel Blockchain Consensus*. 2019. arXiv: 1912.04065 [cs.CR].

[15] Garay, J., Kiayias, A., and Leonardos, N. *The Bitcoin Backbone Protocol: Analysis and Applications*. Cryptology ePrint Archive, Paper 2014/765. 2014.

[16] Global, I. O. *Atala PRISM: A Self-Sovereign Identity Platform*. 2024.

[17] Horizen Academy. *UTXO vs. Account Model*. 2023.

[18] Mysten Labs Team. *The Sui Smart Contracts Platform*. 2021.

[19] Nakamoto, S. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: *metzdowd.com* (2008).

[20] nhsz. *Consensus Mechanisms*. 2023.

[21] nhsz. *Proof-of-Stake (PoS)*. 2023.

[22] nhsz. *Proof-of-Work (PoW)*. 2023.
[23] Saad, S. M. S., Radzi, R. Z. R. M., and Othman, S. H. "Comparative Analysis of the Blockchain Consensus Algorithm Between Proof of Stake and Delegated Proof of Stake". In: *2021 International Conference on Data Science and Its Applications (ICoDSA)*. 2021, pp. 175–180. DOI: 10.1109/ICoDSA53588.2021.9617549.
[24] SpaceBudz. *Lucid: Cardano Off-Chain Framework*. 2024.
[25] Szabo, N. *Smart Contracts*. 1994.
[26] Team, P. *ProofSpace: Decentralized Identity*. 2024.
[27] Tschorsch, F. and Scheuermann, B. "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies". In: *IEEE Communications Surveys & Tutorials* 18.3 (2016), pp. 2084–2123. DOI: 10.1109/COMST.2016.2535718.
[28] TxPipe. *Aiken: A Modern Smart Contract Platform for Cardano*. 2024.
[29] Wood, G. et al. "Ethereum: A Secure Decentralised Generalised Transaction Ledger". In: *Ethereum Project Yellow Paper* 151 (2014), pp. 1–32.
[30] World Wide Web Consortium (W3C). *Decentralized Identifiers (DIDs) V1.0, Core Architecture, Data Model, and Representations*. 2022.
[31] Yakovenko, A. *Solana: A New Architecture for a High Performance Blockchain*. 2021.
[32] Zheng, Z. et al. "An Overview on Smart Contracts: Challenges, Advances and Platforms". In: *Future Gener. Comput. Syst.* 105 (2020), pp. 475–491. ISSN: 0167-739X. DOI: 10.1016/j.future.2019.12.019.

Chapter 7

Appendix

A Smart Contracts Code

In this section, we provide the code for the application's on-chain components, written in the Aiken programming language

A.1 Student Validator

```
use aiken/transaction.{ScriptContext}

validator(_student: ByteArray) {
  fn spend(_d: Void, _r: Void, _ctx: ScriptContext) {
    False
  }
}
```

A.2 Subject Validator

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/interval.{Finite}
use aiken/list
use aiken/time.{PosixTime}
use aiken/transaction.{
  InlineDatum, Input, Output, ScriptContext, Spend, Transaction, ValidityRange,
}
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId, Value}

type Datum {
  teacher: VerificationKeyHash,
  registrations: List<StudentId>,
  register_until: PosixTime,
  reset: Bool,
}

type Action {
  Register
  Update
  Grade
}

type StudentId =
  ByteArray

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>
```

```
30
31 validator(
32   policy: PolicyId,
33   asset_name: AssetName,
34   secreta: VerificationKeyHash,
35   registration_policy: PolicyId,
36 ) {
37   fn spend(datum: Datum, redeemer: Action, ctx: ScriptContext) {
38     expect Spend(my_output_reference) = ctx.purpose
39
40     let Transaction { inputs, outputs, reference_inputs, .. } = ctx.transaction
41     expect Some(self_input) =
42       inputs
43         |> transaction.find_input(my_output_reference)
44
45     let self_output = get_self_output(outputs, self_input.output.address)
46     expect InlineDatum(d) = self_output.datum
47     expect Datum { teacher, registrations, register_until, reset }: Datum = d
48     when redeemer is {
49       // Can only register a new student. Everything has to remain unchanged, except for the
      registrations list
50       Register -> and {
51           datum.reset == False,
52           is_period_valid(ctx.transaction.validity_range, datum.register_until),
53           signed_by(ctx.transaction, secreta),
54           teacher == datum.teacher,
55           reset == datum.reset,
56           register_until == datum.register_until,
57           are_equal_except_last(datum.registrations, registrations),
58           same_values(self_input.output.value, self_output.value),
59         }
60       // Can only update teacher or extend registration period
61       Update -> and {
62           signed_by(ctx.transaction, secreta),
63           reset == datum.reset,
64           datum.registrations == registrations,
65           same_values(self_input.output.value, self_output.value),
66         }
67       Grade -> and {
68           signed_by(ctx.transaction, datum.teacher),
69           if datum.reset {
70             and {
71               list.is_empty(registrations),
72               reset == False,
73               register_until == datum.register_until,
74             }
75           } else {
76             and {
77               registrations == datum.registrations,
78               register_until == datum.register_until,
79               reset == True,
80             }
81           },
82           validate_grade_outputs(
83             outputs,
84             reference_inputs,
85             self_input.output,
86             self_output,
87             datum.registrations,
88             policy,
89             asset_name,
90             registration_policy,
91           ),
92         }
93     }
94   }
95 }
96
97 /// Validate outputs
98 fn validate_grade_outputs(
```

```
  outputs: List<Output>,
  ref_inputs: List<Input>,
  self_input: Output,
  self_output: Output,
  registrations: List<ByteArray>,
  grade_policy: PolicyId,
  grade_asset: AssetName,
  registration_policy: PolicyId,
) -> Bool {
  let grades =
    list.filter_map(
      outputs,
      fn(o) {
        let grade = value.quantity_of(o.value, grade_policy, grade_asset)
        if and {
          grade >= 5,
          grade <= 10,
          o != self_output,
        } {
          expect Some(student_ref_input) =
            list.find(ref_inputs, fn(i) { i.output.address == o.address })
          expect [(student, _)] =
            student_ref_input.output.value
              |> value.tokens(registration_policy)
              |> dict.to_list()

          expect list.has(registrations, student)
          Some(grade)
        } else if grade != 0 {
          // if the grade is not 0, but neither in [5,10] range, then it has to be the
    self_output
          expect o == self_output
          Some(grade)
        } else {
          None
        }
      },
    )

  list.foldr(grades, 0, fn(sum, total) { sum + total }) == value.quantity_of(
    self_input.value,
    grade_policy,
    grade_asset,
  )
}

/// Get the output that has the same address as this validator
fn get_self_output(outputs: List<Output>, self: Address) -> Output {
  expect [self_output] = list.filter(outputs, fn(o) { o.address == self })
  self_output
}

/// Check if 2 Value objects are identical, except for the lovelace amount
fn same_values(input_values: Value, output_values: Value) -> Bool {
  list.is_empty(
    list.difference(
      value.flatten(value.without_lovelace(input_values)),
      value.flatten(value.without_lovelace(output_values)),
    ),
  )
}

/// Check if signer[0] == vk
fn signed_by(transaction: Transaction, vk: VerificationKeyHash) -> Bool {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}

/// Checks if 2 arrays are equal except for the last element of the second array
fn are_equal_except_last(input_list: List<a>, output_list: List<a>) -> Bool {
  expect Some(output_) = list.init(output_list)
```

```
  input_list == output_
}

/// Check if registration period has not expired
fn is_period_valid(range: ValidityRange, target_time: PosixTime) -> Bool {
  when range.lower_bound.bound_type is {
    Finite(tx_earliest_time) -> tx_earliest_time <= target_time
    _ -> False
  }
}
```

A.3 Policy Contract Validator

```
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{Output, ScriptContext, Spend, Transaction}
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId, Value}

type Datum {
  subjects: List<GradeToken>,
}

type GradeToken {
  policy: PolicyId,
  name: AssetName,
}

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

validator(secreta: VerificationKeyHash) {
  fn change_subjects(_datum: Datum, _redeemer: Void, ctx: ScriptContext) -> Bool {
    expect Spend(my_output_reference) = ctx.purpose

    let Transaction { inputs, outputs, .. } = ctx.transaction
    expect Some(self_input) =
      inputs
        |> transaction.find_input(my_output_reference)

    let self_output = get_self_output(outputs, self_input.output.address)
    and {
      signed_by(ctx.transaction, secreta),
      same_values(self_input.output.value, self_output.value),
    }
  }
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) -> Bool {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}

fn get_self_output(outputs: List<Output>, self: Address) -> Output {
  expect [self_output] = list.filter(outputs, fn(o) { o.address == self })
  self_output
}

fn same_values(input_values: Value, output_values: Value) -> Bool {
  list.is_empty(
    list.difference(
      value.flatten(value.without_lovelace(input_values)),
      value.flatten(value.without_lovelace(output_values)),
    ),
  )
}
```

A.4 Grade Token Minting Policy

```
use aiken/dict
use aiken/list
use aiken/transaction.{OutputReference, ScriptContext, Transaction} as tx
use aiken/transaction/value

validator(token_name: ByteArray, utxo_ref: OutputReference) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { inputs, mint, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    and {
      asset_name == token_name,
      if amount > 0 {
        list.any(inputs, fn(input) { input.output_reference == utxo_ref })
      } else {
        True
      },
    }
  }
}
```

A.5 Registration Token Minting Policy

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{ScriptContext, Transaction} as tx
use aiken/transaction/credential.{VerificationKey}
use aiken/transaction/value

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

validator(secreta: VerificationKeyHash) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { mint, .. } = transaction
    expect [(_, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    if amount > 0 {
      signed_by(ctx.transaction, secreta)
    } else {
      True
    }
  }
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}
```

A.6 Validity Token Minting Policy

```
use aiken/dict
use aiken/list
use aiken/transaction.{OutputReference, ScriptContext, Transaction} as tx
use aiken/transaction/value
```

```
validator(utxo_ref: OutputReference) {
  fn policy(_datum: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { inputs, mint, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    and {
      list.any(inputs, fn(input) { input.output_reference == utxo_ref }),
      asset_name == #"56616c6964697479",
      amount == 1,
    }
  }
}
```

A.7 Degree NFT Minting Policy

```
use aiken/dict
use aiken/hash.{Blake2b_224, Hash}
use aiken/list
use aiken/transaction.{InlineDatum, Input, ScriptContext, Transaction} as tx
use aiken/transaction/credential.{Address, VerificationKey}
use aiken/transaction/value.{AssetName, PolicyId}

type VerificationKeyHash =
  Hash<Blake2b_224, VerificationKey>

type RefDatum {
  subjects: List<GradeToken>,
}

type GradeToken {
  policy: PolicyId,
  name: AssetName,
}

validator(
  register_policy: PolicyId,
  validity_policy: PolicyId,
  secreta: VerificationKeyHash,
) {
  fn policy(_: Void, ctx: ScriptContext) -> Bool {
    let ScriptContext { transaction, purpose } = ctx
    expect tx.Mint(policy_id) = purpose
    let Transaction { mint, reference_inputs, outputs, .. } = transaction
    expect [(asset_name, amount)] =
      mint
        |> value.from_minted_value
        |> value.tokens(policy_id)
        |> dict.to_list()

    expect Some(student_input) =
      list.find(
        reference_inputs,
        fn(i) {
          value.quantity_of(i.output.value, register_policy, asset_name) == 1
        },
      )

    let student_address = student_input.output.address
    expect Some(ref_input) =
      list.find(
        reference_inputs,
        fn(i) {
```

```
          value.quantity_of(
            i.output.value,
            validity_policy,
            #"56616c6964697479",
          ) == 1
        },
      )

    expect InlineDatum(ref_data) = ref_input.output.datum
    expect RefDatum { subjects }: RefDatum = ref_data

    and {
      signed_by(ctx.transaction, secreta),
      amount == 2,
      validate_grades(subjects, reference_inputs, student_address),
      list.any(
        outputs,
        fn(o) {
          o.address == student_address && value.quantity_of(
            o.value,
            policy_id,
            asset_name,
          ) == 1
        },
      ),
    }
  }
}

fn validate_grades(
  policies: List<GradeToken>,
  grade_inputs: List<Input>,
  student_address: Address,
) -> Bool {
  let values =
    list.foldr(
      grade_inputs,
      value.zero(),
      fn(i, v) {
        if i.output.address == student_address {
          value.merge(v, i.output.value)
        } else {
          v
        }
      },
    )

  list.all(policies, fn(p) { value.quantity_of(values, p.policy, p.name) >= 5 })
}

fn signed_by(transaction: Transaction, vk: VerificationKeyHash) {
  list.at(transaction.extra_signatories, 0) == Some(vk)
}
```

B Credential Issuance Code

In this section, we provide the complete code used to issue verifiable credentials by invoking the ProofSpace API from a Node.js environment.

```
const crypto = require('crypto');
const axios = require('axios');

const url = "https://platform.proofspace.id/service-backend/v1/service/
    WwDZ3sgsvnYsSvofQ6zjT5/webhook-accept/credentials-issued";
const serviceDid = "WwDZ3sgsvnYsSvofQ6zjT5";
const credentialId = "WwDZ3sgsvnYsSvofQ6zjT5:3:CL:721:tag";
const schemaId = "WwDZ3sgsvnYsSvofQ6zjT5:2:University Registration:1.1";
const userDid = "JRpxFRCH84RrQcptCzF3PN";

const private_key = `-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAtsEaCBfGaRhWfkLuiKvK8InKgZyl0od1DcNAlVi6YaEmEruS
e8lPJfewXR3uxIg3SYOAGI6LYFFIJxAMcvHYEm0axW5lo4ov1hYRMA76tl/eo1Kp
3Cp7Ai8W9GshcQzbFwPX9UjbijHkRkwCIpW+iK/aipj/nBg1LJsOR4b9hpbB92Hz
+b4nviGBD5CA0vzbyDhLq/4oQbyOvAsAFAqKZEqBHq4s+WcKDHXFkvsmxnzexAlH
JtGjhFU+B3myXPMoMEHUInoKW0cSez5/WmkUOrPcNu4jmVmpjnVek+7s9q8F+blA
RaI4WKUUM+MA3sBGkMt3QeyKOm0L9JfA5C3q0ZmJ4xkJvaK0zCdml8DbgKasIy/L
JMXkQZyOZGlg2YsjcLK3lsElefAQ8d4/J4wSF+I/0uVKV/vxzVBx9kKgpUGJQu9l
0pGsxaQyZ94wUBiJybZqx8EUQmLIH7ONjSAGFHEISPLOlqRihJRz3zVdIKn04k1I
ftqk477mz5YJD5zVq9w9qx77oCzfH34N8pU/vTUaFmoWHhdnrb8slXaATOMejksA
kCK/wqUhYnGol8cNq92uTC/IdcEbBkHyHyQOHV88R1myoqY9bg4CbZzJDrtCzAUH
mN5fVCUVh7WL9Pt5pskD3zPPv77gxzJYLI+/9kOgwCXHrk9kGnFwgjsJHcUCAwEA
AQKCAgABsPbLNWEeLt36DCeF90UJBdVsbqUwNhWOu2ZZG3maFrIyAS9hWuMTIGAu
9Gx1wCb21LM2RvduPEJVLjZXR/qubPkO868YHNrFee3Nyo2ya1+1ljnBMllyEzp8
p5MnJy3N4Di66eIyPLT2m5R3enhJboswtirDaIvxbjvrGIo3GSURY4Qs/wACJ/A4
UyeGDFYCW9cDb/YyaYrOnbFx/gj61Z+tR4JeCoCSzfI2ooFg29355XYVm2IOgvVw
f0VRy+RG0g0jBosa/iTjJBhsn0VUj9qNXZXRwP0rNtXZu5/QxpzhC0BXhKFN3LU7
p7Gl3hSpGWDwqL48vS6bpF3iHrSqGN+d+WtNGCMWSNcq9b8Xp0ft7kx3XHMKYU/s
2+HtMO3jp0fxr4i7tG/DZpaahLA/HEIWG4AK5CEKOL0UDYtqwyoNedoPjhkr5nxf
Kx4BX40b2DG83hU5imsp2zP7XAkmiIpNYNdXosFrKoWcBU3vCKus9/MGS34ZNOR4
9LhRA1I478qfsUXcmx1PcmIRyyobanZsyzxad9n/yxDxCZ50nnxCMvJUWSRU9n8l
3dwafbgMV/ispxQXc9oTof61LCFVlrukjNYIcz+03V31q2/tp+dL2UNZWObT72ED
Aw8SPy6yB5BYpNU1BdC+jPmfDuDlnUPNSY8hBqV6otlXNIDCnwKCAQEA9GctrVnB
pZHRMwPZfL9VK7AiWkLdoWThtmuNHRZm0XhepYH8xaxlx6myQtkx3QYdJc1gpORC
BEoP8DtDdfOAZ4LGjic1GbXyHpI8bdgGf4H11EXZPr+PlAGLSyW8Ie+CYHfrKpYL
uHLaOIrSx4k+Fbj4B4obNEg/s4QSzQichI+UiwGmMuYx5KBlBXHnBWdN0+ICwbJq
Q3XRffVi0Xc0wDJjESXyVdqzMVnb18bHRkjwxTx2tgFZEBVYLeWtX3YTeQkzP/Yx
xIvzzXVtTrZa/DyD/bXs9ea9oJ0q80ABmyWUESTFrzrbYnAQG2UWYyPb6gv2Lw1D
1w5zTb5soZL0xwKCAQEAv20Pu8TfUABm1Imo/IsDA0koNB/0BpIEyfAeFHQ0UPR6
3rzli90Dn96PBfmlyOTPQir1RnSxpcxQW6eyU1ZGKasLFoS41mrnFTvbdOof/2Vd
kbNcwFb3wnLjm1YP4ImP36ugM4z4n++y03Uqbxz5wAsawq+5U+M0Z4miUrNRsPjz
OPBCl25VLXo3wFvP8qhIToDHnO5GNUBPCWZttlgi7Lwu8C/BRLGFSfIBAU7HFLIh
3foeLh6u+I4F1/ew8KMkrHxnQxDzMI8HaVtjwGtNdZmyXwf4M4ekJfGMowbPLh+k
3Ud68i2KPAmR2k50uS+gt/RfBYlqxqGCs22urK51EwKCAQAhaQVugCmft6pQfgM6
MXG0MjTP//oKOR8977/vrQNLOVg+6eEjKKkVPVZ4GvbdDxvm8gd3Ixj/Oxyoq4FS
W4I/JDp+bk+s/xllInqOTYy924c+iHnlQQygoAFpuXPZc8Nv6pTEUpSxl0nLlvav
fkx1bb7CURfc8s96e+CSFM5ItNQKmKQeqydA6MhGkUaSj4etImVcE0bBiBamUGg8
2D3m1DyEuJpDME10vqL2AV92dTwsylI0GUUSlMhkJdSoixyoXCCcasxs4PDoMlhJ
/VW8y40Dnt4Q04UysqmPbT2K9v0hxtQbW0FoWIdrJ0uDLADBIvd5gSe3+2q5wwFY
qR07AoIBAQCQ9qh5i8eJMCSvQ2iDh0pKzLXzc1c7Oo0B2CMPXkjYHayGzQoQdtjt
0n8BCWNw1UXXV+zS5EA7zJP2NPKohacwmuUUsoG/bsl8nQ2+ZibBJCRN2ESfK70I
tjSDSBNR+RrJSZFSOKl3hPApjosKBilkM4SadrSkQGVJG0Of2Ug9+ulF7zyerOE4
RVu/Akhh/Op0+LUxtIfaf87a0YS4F9qq6H+rSJ7VxJGpw+ja8qt0kslu7qC4UGg0
tmjaeiq2sPVu4e7ZLRbdaRM1yA2shBXuvuwDZ2egrlGyok/cjaWLyk7xe9YxXCdY
++5GIi/mpZjSde8XpE8GCP2Pn7ot7xedAoIBAQDEnlbgWPS2tldLoVFG4ega9x09
G/SVen5kkmLRaBNLV6rGRnCQoYkosUdbKMUsTb8WJGJhIllu5u0SgbaYyxrG63ob
FdDgVjIPJneL1xnX3s6qP9084k2cBqwE3LtNfVK6eWYEwXvaeHQN/Ur3KbJQ3tDi
LkNpzd4ysqRmy01UOIQxWG0nnM+LDRjhmong8u+HAeiYmZ9WZ2D06nd5dbq19FeT
F0dJYc4jhBZQoW7oaaNAMJ4JY/ViM10ISzjwaWcanQ0B1pH0yk0oSReWd82BFQfb
sLC/A+iBdA4dNfB9fnB+zrF0Php+2mpMZOnNyo5Nvm2N08wVOYk5lho0psEj
-----END RSA PRIVATE KEY-----
`;

const credentialValues = {
    credentialId,
```

```
    schemaId,
    fields: [
        { name: "School", value: "ECE NTUA" },
        { name: "StudentID", value: "el17108" },
    ],
    utcIssuedAt: new Date().getTime(),
    revoked: false
};

const request = {
    serviceDid,
    subscriberConnectDid: userDid,
    subscriberEventId: "",
    credentials: [credentialValues],
    issuedAt: new Date().getTime(),
};

const headers = {
    'Content-Type': 'application/json'
};

const binaryBody = Buffer.from(JSON.stringify(request), "utf-8");

const key = crypto.createPrivateKey({ key: private_key });
const signature = crypto.sign('sha3-256', binaryBody, key);

headers['X-Body-Signature'] = signature.toString('base64');

axios.post(
    url,
    binaryBody,
    {
        headers: headers
    }
).then((reply) => {
    return reply.data;
}).catch((err) => {
    throw err;
});
```