ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Συγκριτική αξιολόγηση επίδοσης συστήματος διαχείρισης μαθημάτων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Θεόδωρου Γεωργόπουλου

Επιβλέπων: Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π

Αθήνα, Οκτώβριος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ, ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Συγκριτική αξιολόγηση επίδοσης συστήματος διαχείρισης μαθημάτων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

Θεόδωρου Γεωργόπουλου

Επιβλέπων: Ευστάθιος Συκάς
Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 8η Νοεμβρίου 2022.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	….....
Ευστάθιος Συκάς	Νικόλαος Μήτρου	Ιωάννα Ρουσσάκη
Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Αναπλ.Καθηγήτρια Ε.Μ.Π

Αθήνα, Οκτώβριος 2022

(Υπογραφή)

.....
Θεόδωρος Γεωργόπουλος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

Περίληψη

Η διαδικασία δοκιμής επίδοσης (performance testing) ενός συστήματος με σκοπό τη μελέτη της απόκρισής του αποτελεί ένα πολύ σημαντικό μέρος της σχεδίασης ενός συστήματος. Με τη δοκιμή επίδοσης προσομοιώνεται η λειτουργία ενός συστήματος και εξετάζεται η συμπεριφορά του όταν αυτό υπόκειται σε κάποιο φορτίο. Μια υποκατηγορία της δοκιμής επίδοσης, αποτελεί η δοκιμή φορτίου (load testing). Με τη διαδικασία της δοκιμής φορτίου προσομοιώνεται η αναμενόμενη συμπεριφορά ενός συστήματος προκειμένου να διαπιστωθεί εάν το σχεδιαζόμενο σύστημα πληροί τις προϋποθέσεις επιθυμητής λειτουργίας.
Στην παρούσα διπλωματική εργασία διεξάγεται μια σειρά από τεστ δοκιμής φορτίου στο σύστημα διαχείρισης μαθημάτων Moodle. Τα τεστ σχεδιάζονται με τέτοιο τρόπο, ώστε να προσομοιώνουν την πραγματική συμπεριφορά των χρηστών ενός τέτοιου συστήματος. Έτσι, δοκιμάζεται το σύστημα σε διάφορες αρχιτεκτονικές προκειμένου να μελετηθεί η επίδοσή του αναφορικά με τους χρόνους απόκρισης, αλλά και το μέγιστο πλήθος των ταυτόχρονων χρηστών που μπορεί να αντέξει. Οι αρχιτεκτονικές που μελετώνται είναι εκείνη όπου το Moodle αποτελεί έναν αυτοδύναμο εξυπηρετητή, στη συνέχεια διαχωρίζεται η βάση δεδομένων από το σύστημα για να αποτελέσει μέρος ενός αυτοδύναμου MySQL εξυπηρετητή και τέλος γίνεται η μετάβαση σε σύστημα μιας συστάδας αποτελούμενη από δύο Moodle εξυπηρετητές (Moodle cluster). Επιπλέον, διεξάγονται και οι αναγκαίες βελτιστοποιήσεις που αυξάνουν την επίδοση του συστήματος. Τέλος, συλλέγονται τα αποτελέσματα των δοκιμών και εξάγονται συμπεράσματα σχετικά με την απόκριση του συστήματος.

Λέξεις κλειδιά

Moodle, learning management system (LMS), performance testing, load testing, stress testing, Apache JMeter, APDEX score, Benchmark, Redis, NFS, Unison, MySQLTuner, PHP-FPM, Multi-Processing Module (MPM), Thinking time, cluster, reverse proxy, load balancer.

Abstract

The process of performance testing, in order to study a systems response, is a very important part of a system design process. Performance testing simulates the operation of a system and examines its behavior and responsiveness under a particular workload. Load testing is a specific type of performance testing. With load testing, the expected behavior of a system is simulated in order to determine whether the designed system meets the conditions of the desired operation.

In this diploma thesis, a series of load tests are carried out in the Moodle learning management system (LMS). The tests are designed in such a way, as to simulate the real behavior of the users of such a system. Thus, the system is tested in various system architectures, in order to study its performance in terms of response time and maximum number of simultaneous users. The studied system architectures are the ones where Moodle is a standalone server based on the LAMP software stack, then the database is separated from the system to become part of a MySQL standalone server and finally a Moodle cluster consisting of two Moodle servers. In addition, necessary system optimizations are done, in order to increase the performance. Finally, the test results are collected and conclusions are made about the system's response.

Keywords

Moodle, learning management system (LMS), performance testing, load testing, stress testing, Apache JMeter, APDEX score, Benchmark, Redis, NFS, Unison, MySQLTuner, PHP-FPM, Multi-Processing Module (MPM), Thinking time, cluster, reverse proxy, load balancer.

Ευχαριστίες

Με την ολοκλήρωση της παρούσας διπλωματικής εργασίας θα ήθελα να ευχαριστήσω ιδιαιτέρως τον επιβλέποντα καθηγητή κ. Ευστάθιο Συκά για την εμπιστοσύνη που μου έδειξε και την ευκαιρία που μου έδωσε ώστε να μελετήσω ένα τόσο ενδιαφέρον θέμα. Με τη συνεχή στήριξή του και τις συμβουλές του η συνεργασία που είχαμε κατά την εκπόνησή της ήταν άψογη.
Επιπλέον θα ήθελα να ευχαριστήσω όλους τους καθηγητές μου που κατά τη διάρκεια της φοίτησής μου στο Πολυτεχνείο μου μετέφεραν πολύτιμες και υψηλού επιπέδου γνώσεις.

Ακόμα θα ήθελα να ευχαριστήσω την οικογένεια και τους φίλους μου, για τη στήριξή τους κατά τη διάρκεια των σπουδών μου.
Ιδιαιτέρως οφείλω ένα μεγάλο ευχαριστώ στον αδερφό μου Αλέξανδρο, γιατί ήταν μαζί μου σε όλες τις δύσκολες στιγμές των φοιτητικών μου χρόνων, ενώ παράλληλα κάναμε αυτό το πανέμορφο ταξίδι.
Τέλος, θα ήθελα να ευχαριστήσω το φίλο και συνάδελφο Μάνο Ρωμανό για τη συνεργασία που είχαμε και τις υπέροχες στιγμές που ζήσαμε μαζί στη σχολή.

Περιεχόμενα

Κατάλογος Στιγμιότυπων

Κατάλογος Πινάκων

Μέρος Α

Κεφάλαιο 1: Εισαγωγή

Η παρούσα διπλωματική εργασία έχει ως αντικείμενο τη μελέτη και δοκιμή της επίδοσης (performance testing) συστημάτων διαχείρισης μαθημάτων όταν αυτά υπόκεινται σε διαφορετικό πλήθος χρηστών που αιτούνται τις υπηρεσίες. Συγκεκριμένα γίνεται η δοκιμή φορτίου (load testing) του λογισμικού Moodle σε διάφορες αρχιτεκτονικές και η σύγκριση μεταξύ τους. Με τη χρήση του λογισμικού Apache JMeter επιτυγχάνεται η μοντελοποίηση της δραστηριότητας των εικονικών χρηστών και κατ' επέκταση η δοκιμή φορτίου, μέσω της διεξαγωγής HTTP αιτημάτων προς τον εξυπηρετητή ιστού.
Παράλληλα αναζητείται η στένωση του συστήματος (bottleneck), στην οποία οφείλονται οι κακές επιδόσεις, καθώς και η εύρεση της βέλτιστης υλοποίησης.

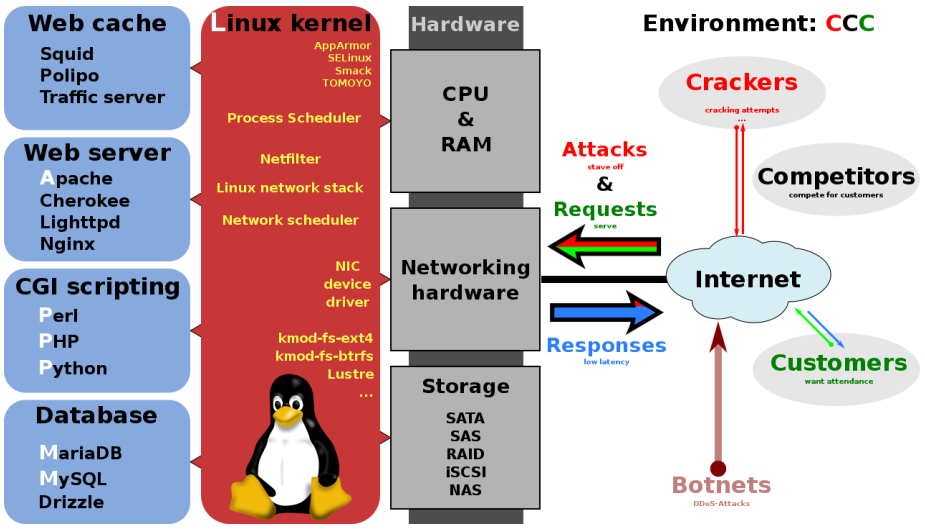
Σκοπός διπλωματικής εργασίας

Κύριος στόχος είναι η ορθή διεξαγωγή της δοκιμής φορτίου προς τον Moodle εξυπηρετητή, μέσω της οποίας παρατηρείται η απόκριση του συστήματος, καθώς και το μέγιστο πλήθος χρηστών υπό κανονικές συνθήκες λειτουργίας. Μέσω του Moodle προσομοιώνουμε ένα περιβάλλον ενός εκπαιδευτικού ιδρύματος όπως για παράδειγμα του Ε.Μ.Π, αλλά μικρότερης κλίμακας, το οποίο αποτελείται από μαθήματα και εγγεγραμμένους εικονικούς χρήστες, οι οποίοι χρησιμοποιούνται για τη δημιουργία HTTP αιτημάτων προς τον εξυπηρετητή.
Η δοκιμή φορτίου απαιτεί τη δημιουργία του τεστ, τη διαλογή των αποτελεσμάτων και τη διεξαγωγή συμπερασμάτων αναφορικά με τις μετρικές προς μελέτη. Η δοκιμή φορτίου συνίσταται κυρίως από ενέργειες που εκτελούν συνήθως οι φοιτητές και οι οποίες μοντελοποιούνται μέσω του Apache JMeter. Οι ενέργειες αυτές είναι το login, logout, η ανάγνωση των ανακοινώσεων , του forum κτλ.
Η σημαντικότερη μετρική είναι ο χρόνος απόκρισης στα αιτήματα των χρηστών. Έτσι ορίζονται οι χρόνοι απόκρισης εντός των επιθυμητών ορίων προκειμένου η εφαρμογή να αποδίδει τα επιθυμητά αποτελέσματα. Παράλληλα με την ανάλυση φορτίου διεξάγεται και η προσομοίωση ακραίων καταστάσεων ή καταπόνησης (stress testing) μέσω της οποίας προσδιορίζεται η απόκριση της πλατφόρμας υπό ακραίες καταστάσεις.

1.1: Μεθοδολογία και δομή υλοποίησης του συστήματος

Στην παρούσα διπλωματική εργασία σχεδιάστηκαν τεστ δοκιμής φορτίου με τρόπο, ώστε να ανταποκρίνονται στην πραγματική συμπεριφορά των χρηστών μιας τέτοιας εφαρμογής (μαθητών, φοιτητών και όχι μόνο). Στη συνέχεια διεξήχθη η δοκιμή φορτίου του λογισμικού Moodle σε ποικίλες αρχιτεκτονικές. Έτσι τα βήματα εκτέλεσης και οι αρχιτεκτονικές είναι οι ακόλουθες:

- Αρχικά, το Moodle εγκαταστάθηκε σε έναν αυτοδύναμο εξυπηρετητή, ο οποίος βασίζεται στη δομή LAMP (εικόνα 1.1), δηλαδή συγκεντρώνει όλες τις αναγκαίες υπηρεσίες και λογισμικά για τη λειτουργία της εφαρμογής, όπως τον εξυπηρετητή ιστού (Apache), τη βάση δεδομένων (MySQL) και τον ίδιο τον κώδικα της εφαρμογής (PHP).



Εικόνα 1.1: Δομή συστήματος LAMP [1]

- Στη συνέχεια, τροποποιήθηκε η παραπάνω αρχιτεκτονική αποσπώντας τη βάση δεδομένων από το αρχικό μηχάνημα και εγκαθιστώντας τη σε έναν αυτοδύναμο MySQL εξυπηρετητή.

- Έπειτα επιτυγχάνεται η μετάβαση στο HTTPS, με τον τερματισμό του ssl και με τη χρήση ενός Nginx reverse proxy δρομολογούνται όλα τα HTTP αιτήματα στον Moodle εξυπηρετητή.

- Η τελευταία αρχιτεκτονική, αποτελεί το δομικό στοιχείο για την κλιμάκωση του συστήματος, προκειμένου να μπορεί να επεκταθεί, ώστε να ικανοποιεί υψηλό αριθμό ταυτόχρονων χρηστών. Η αρχιτεκτονική αυτή, είναι μια συστάδα από δύο Moodle

εξυπηρετητές (Moodle cluster), οι οποίοι αποτελούν τους back-end εξυπηρετητές του συστήματος εκτελώντας τον κώδικα PHP, ενώ ένας Nginx load balancer αναλαμβάνει την προώθηση όλων των εισερχόμενων αιτημάτων σε αυτούς, αλλά και τον καταμερισμό του συνολικού φορτίου.

Η επιλογή των παραπάνω αρχιτεκτονικών γίνεται, καθώς ο αυτοδύναμος Moodle εξυπηρετητής παρουσιάζει ένα όριο αναφορικά με την εξυπηρέτηση μεγάλου αριθμού ταυτόχρονων χρηστών. Επομένως, είναι αναπόφευκτη η μετάβαση στην αρχιτεκτονική της συστάδας αποτελούμενη από Moodle εξυπηρετητές, σε περίπτωση που είναι επιθυμητή η αύξηση του πλήθους των ταυτόχρονων χρηστών που επισκέπτονται και χρησιμοποιούν την εφαρμογή.
Επομένως το πρώτο βήμα για τη βελτίωση της επίδοσης του αυτοδύναμου Moodle εξυπηρετητή αποτελεί ο διαχωρισμός της βάσης δεδομένων MySQL και η εγκατάστασή της σε έναν άλλον αυτοδύναμο MySQL εξυπηρετητή.
Έτσι η τελική αρχιτεκτονική που στοχεύει στην κλιμάκωση της εφαρμογής είναι η συστάδα των δυο Moodle εξυπηρετητών. Στη συστάδα αυτή είναι αναγκαία και η ύπαρξη ενός λογισμικού το οποίο συγχρονίζει τον κοινόχρηστο φάκελο της εφαρμογής με όνομα moodledata. Αυτό, επιτυγχάνεται με τη χρήση του NFS ή του Unison, τα οποία μελετώνται στο κεφάλαιο 4.3.
Επιπλέον χρησιμοποιείται και το λογισμικό Redis, με σκοπό την ύπαρξη κρυφής μνήμης (cache) των συνόδων (sessions) του Moodle, όπως εξηγείται σε επόμενο κεφάλαιο.

1.2: Moodle

Το Moodle είναι λογισμικό ανοιχτού κώδικα το οποίο δημιουργήθηκε το 1999 από τον Αυστραλό Martin Dougiamas και προσφέρει ολοκληρωμένες υπηρεσίες ασύγχρονης τηλεκπαίδευσης.

Εικόνα 1.2.1: Λογότυπο Moodle

Επιπρόσθετα παρέχει τη δυνατότητα δημιουργίας, διαχείρισης μαθημάτων και εκπαιδευτικών πόρων, καθώς και υποστηρίζει ποικίλους τρόπους αξιολόγησης.
Το πλεονέκτημα χρήσης του Moodle έγκειται στο γεγονός ότι αποτελεί μία πλατφόρμα η οποία συγκεντρώνει όλες τις δυνατότητες διαχείρισης μάθησης με εύκολο τρόπο. Επιπλέον προσφέρει πλήρη ελευθερία στην προσαρμογή της εφαρμογής στις ανάγκες του κάθε

εκπαιδευτικού ιδρύματος ή οργανισμού, μέσω της τροποποίησης του πηγαίου κώδικα ή και των ρυθμίσεών της. Τέλος, από τη σκοπιά των υπολογιστικών συστημάτων, το Moodle καθίσταται μία πλατφόρμα η οποία μπορεί με μεγάλη ευελιξία να επεκταθεί από εκατοντάδες χρήστες έως και εκατομμύριους για εφαρμογές σε μεγάλους οργανισμούς, διατηρώντας τις υψηλές επιδόσεις της [2]. Ο τρόπος εγκατάστασης και όλες οι κατάλληλες ρυθμίσεις παρουσιάζονται στο κεφάλαιο 8.

Εργαλεία του Moodle

Εργαλείο Moodle για τη δημιουργία Test courses

Το Moodle, παρέχει τη δυνατότητα δημιουργίας μαθημάτων που μπορούν να χρησιμοποιηθούν για δοκιμή επίδοσης [3]. Διατίθενται πολλές επιλογές δημιουργίας ενός τέτοιου μαθήματος, αναφορικά με το πλήθος των εγγεγραμμένων χρηστών ή το μέγεθος του μαθήματος.
Το μέγεθος ενός μαθήματος εξαρτάται από το πλήθος των ανακοινώσεων, των πεδίων, των εργασιών, αλλά και των εγγεγραμμένων χρηστών. Τα μεγέθη των μαθημάτων που μπορούν να δημιουργηθούν απεικονίζονται στον παρακάτω πίνακα.

Size of course	Approx. size	No. of assignments	No. of pages	No. of small files	No. of big files	No. of sections	No. of users	No. of forum posts	Approx. time to create
XS	10 kB	1	1	1	1	1	1	2	1 sec
S	10 MB	10	50	64	2	10	100	20	30 sec
M	100 MB	100	200	128	5	100	1,000	500	2 min
L	1 GB	500	1,000	1,024	10	500	10,000	5,000	30 min
XL	10 GB	1,000	5,000	16,384	10	1,000	50,000	10,000	2 hours
XXL	20 GB	2,000	10,000	32,768	10	2,000	100,000	20,000	4 hours

Πίνακας 1.2: Μεγέθη των Test courses [3]

Στα πλαίσια της παρούσας διπλωματικής εργασίας, θα αρκεστούμε στα μαθήματα μεγέθους **S** και **M**. Επομένως, για τη δημιουργία ενός μαθήματος μετατρέπουμε το Moodle σε DEVELOPER MODE μεταβαίνοντας στο πεδίο Site administration → Development → Debugging → Debug messages και επιλογή DEVELOPER: extra Moodle debug messages for developers.
Θα πρέπει να σημειωθεί, ότι κατά τη δημιουργία ενός μαθήματος δημιουργούνται, αλλά και εγγράφονται χρήστες στο μάθημα. Επομένως, θα πρέπει προηγουμένως να έχει οριστεί ένας κωδικός πρόσβασης για όλους τους δημιουργούμενους χρήστες. Έτσι εισάγουμε την παρακάτω ρύθμιση στο αρχείο παραμετροποίησης του Moodle **/var/www/html/moodle/config.php** :

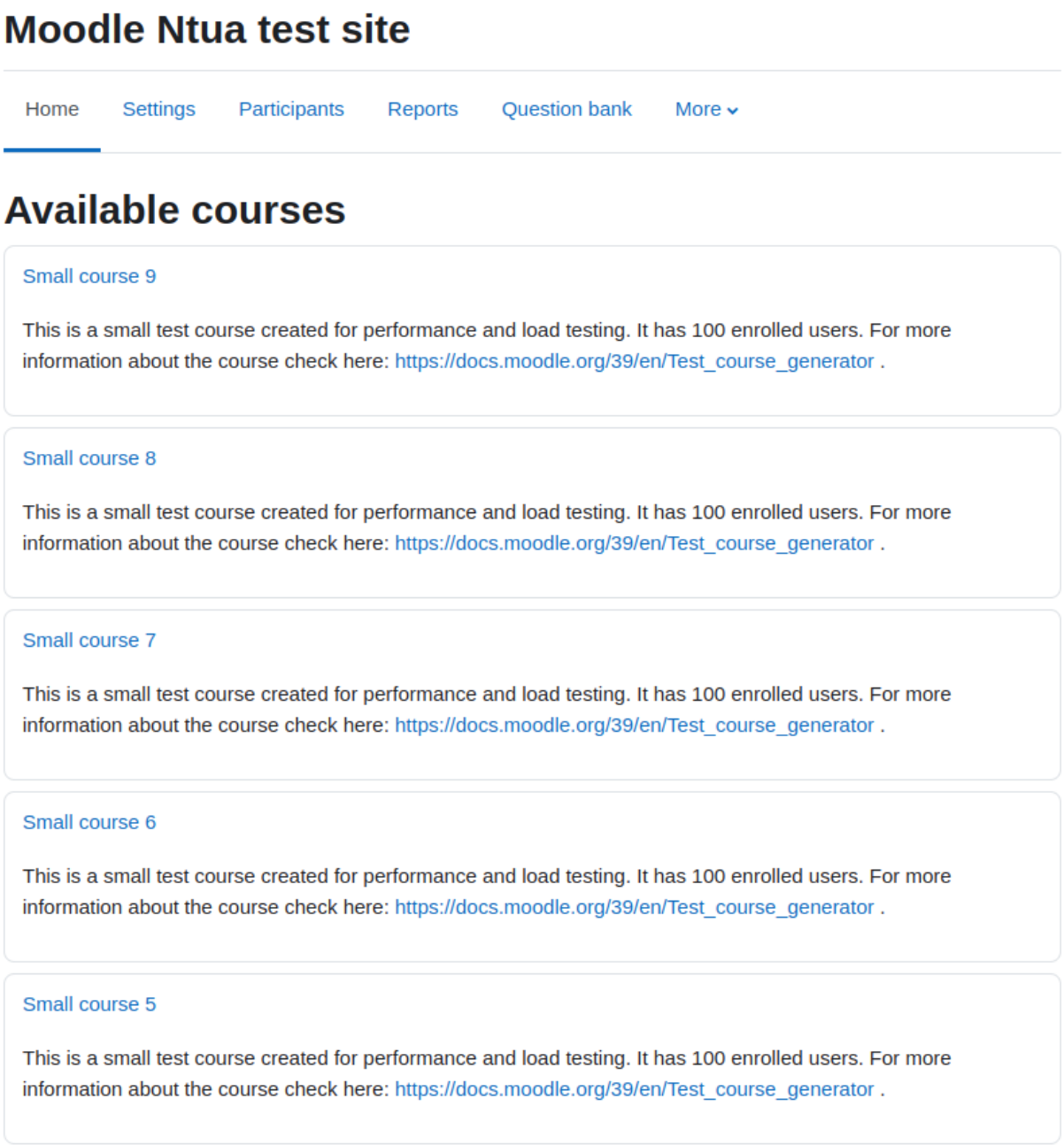
```
$CFG->tool_generator_users_password = 'YOURSECRET';
```

όπου ως *YOURSECRET* επιλέγουμε το επιθυμητό password για τους χρήστες.

Για να χρησιμοποιηθεί το εργαλείο δημιουργίας μαθημάτων θα πρέπει να επιλεχθεί:
Site administration⟶Development⟶Make test course και στη συνέχεια διαλέγουμε το επιθυμητό μέγεθος, όνομα του μαθήματος και περιγραφή για το μάθημα.

Αρχική ιδέα της διπλωματικής εργασίας αποτελούσε η μοντελοποίηση ενός περιβάλλοντος όπως του Ε.Μ.Π και γενικότερα ενός ακαδημαϊκού περιβάλλοντος μικρότερης κλίμακας. Για το λόγο αυτό δημιουργήθηκαν 9 Small courses και 3 Medium.
Η λογική πίσω από αυτήν την επιλογή είναι πως κάθε φοιτητής παρακολουθεί και είναι εγγεγραμμένος κατά μέσο όρο σε **9 μαθήματα** ανά εξάμηνο. Τα μαθήματα μεγέθους Small είναι περίπου του ίδιου μεγέθους με τα πραγματικά μαθήματα στις υποδομές του Ε.Μ.Π. Τα 3 Medium μαθήματα δημιουργήθηκαν για τον εμπλουτισμό του συνολικού συστήματος.

Στα σενάρια των test scripts για τη δοκιμή φορτίου μέσω του JMeter μοντελοποιούμε φοιτητές, οι οποίοι επισκέπτονται τα μαθήματα στα οποία είναι εγγεγραμμένοι και εκτελούν συγκεκριμένες ενέργειες ανά μάθημα. Οι λεπτομέρειες του σεναρίου αναλύονται στη συνέχεια στο κεφάλαιο 2.1.
Στα μαθήματα αυτά έχουν εγγραφεί **10000** χρήστες στο καθένα και μέρος αυτών χρησιμοποιείται για τη διεξαγωγή του τεστ. Στην παρακάτω εικόνα απεικονίζεται ένα στιγμιότυπο από τα μαθήματα που δημιουργήθηκαν (Small course 1 έως 9).



Εικόνα 1.2.2: Δημιουργημένα μαθήματα μεγέθους S

Εργαλείο Moodle για τη δημιουργία JMeter Test

Εκτός από τη δημιουργία ενός μαθήματος, υπάρχει και εργαλείο δημιουργίας ενός JMeter test script ή Test Plan όπως ονομάζεται από το Moodle. Μέσω αυτού του εργαλείου, συντάσσεται αυτόματα ένα test script για το JMeter. Για τη δημιουργία του τεστ ακολουθούμε τα παρακάτω βήματα:

Site administration⟶Development⟶Make JMeter Test Plan και διαλέγουμε το κατάλληλο μέγεθος μαθήματος και συγκεκριμένο μάθημα.
Αφού δημιουργηθεί το τεστ, κατεβάζουμε το test script αρχείο .jmx και το .csv με τους χρήστες και τους κωδικούς τους.

Moodle Benchmark plugin

Το Moodle παρέχει ένα βασικό plugin, μέσω του οποίου μπορεί να εξαχθεί ένα απλό Benchmark Score, αλλά και συμπεράσματα για την επίδοση του συστήματος [4].
Να σημειωθεί, ότι όσο χαμηλότερο είναι το Benchmark Score, τόσο καλύτερη είναι η επίδοση του συστήματος, όπως διακρίνεται στο επόμενο στιγμιότυπο.
Η εγκατάσταση και ρύθμισή του παρουσιάζεται στο κεφάλαιο 8.

System Benchmark

Bechmark Score : 144 points

#	Description	Time in seconds	Acceptable limit	Critical limit
1	Moodle loading time Run the configuration file «config.php»	0.224	0.5	0.8
2	Function called many times A function is called in a loop to test the processor speed	0.383	0.5	0.8
3	Reading files Test the read speed in the Moodle's temporary folder	0.008	0.5	0.8
4	Creating files Test the write speed in the Moodle's temporary folder	0.046	1	1.25
5	Reading course Test the read speed to read a course	0.124	0.75	1
6	Writing course Test the database speed to write a course	0.247	1	1.25
7	Complex request (n°1) Test the database speed to execute a complex request	0.013	0.5	0.7
8	Complex request (n°2) Test the database speed to execute a complex request	0.014	0.3	0.5
9	Time to connect with the guest account Measuring the time to load the login page with the guest account	0.251	0.3	0.8
10	Time to connect with a fake user account Measuring the time to load the login page with a fake user account	0.130	0.3	0.8
	Total time	1.440 sec.		
	Score	144 points		

Εικόνα 1.2.3: Moodle Benchmark plugin [5]

1.3: Δοκιμή επίδοσης (performance testing)

Δοκιμή φορτίου (load testing)

Όπως προαναφέρθηκε, κύριος στόχος της παρούσας διπλωματικής εργασίας είναι η μελέτη της απόκρισης και αντοχής συστημάτων διαχείρισης μαθημάτων, και πιο συγκεκριμένα του Moodle υπό ορισμένο πλήθος χρηστών. Για να επιτευχθεί αυτό διεξάγεται η δοκιμή φορτίου μέσα από κατάλληλα εργαλεία.

Ο όρος δοκιμή φορτίου αναφέρεται στη διαδικασία υποβολής φορτίου σε ένα σύστημα ή κάποιον εξυπηρετητή με σκοπό τη μελέτη της απόκρισης του συστήματος. Πιο συγκεκριμένα, πρόκειται για τη μοντελοποίηση της **αναμενόμενης** και **επιθυμητής λειτουργίας** του συστήματος προς σχεδίαση. Αυτή επιτυγχάνεται προσομοιώνοντας πολλαπλούς εικονικούς χρήστες (virtual users), οι οποίοι επισκέπτονται ταυτόχρονα τον Moodle εξυπηρετητή [6].
Μέσα από τη διαδικασία αυτή, κύριο μέλημα αποτελεί η όσο το δυνατόν πιο **ρεαλιστική** μοντελοποίηση της συμπεριφοράς των χρηστών, η οποία βασίζεται σε πραγματικές ενέργειες που εκτελούνται κατά την επίσκεψη σε έναν Moodle εξυπηρετητή. Με αυτόν τον τρόπο, είναι δυνατή η ανάλυση της απόκρισης του συστήματος και κατ' επέκταση ο προσδιορισμός της ποιότητας των παρεχόμενων υπηρεσιών (Quality of Service), προτού λειτουργήσει επίσημα το σύστημα στο διαδίκτυο. Τα κύρια δεδομένα και οι μετρικές του συστήματος που συλλέγονται από τη διαδικασία αυτή, είναι η χρήση του επεξεργαστή, της μνήμης, οι χρόνοι απόκρισης στα διαφορετικά αιτήματα, καθώς και ο μέγιστος ρυθμός απόδοσης (throughput) του συστήματος σε αιτήματα ανά δευτερόλεπτο. Κατά την ανάλυση των μετρικών γίνεται εμφανής και η στένωση του συστήματος (bottleneck) στην οποία οφείλονται τυχόν κακές επιδόσεις.
Από τη συλλογή των μετρικών αναδεικνύεται και ο μέγιστος αριθμός ταυτόχρονων χρηστών που μπορεί να δεχθεί το σύστημα. Εκτός τούτου, με αυτό το είδος τεστ μοντελοποιούνται και οι ώρες αιχμής, οι οποίες δεν είναι παρά τα διαστήματα όπου η ζήτηση των πόρων παρουσιάζει υψηλή αύξηση σε πολύ μικρό χρονικό διάστημα.

Δοκιμή καταπόνησης (stress testing)

Με τον όρο δοκιμή καταπόνησης υποδηλώνεται η δοκιμή στην οποία υπόκειται ένα σύστημα, υπό ασυνήθιστα υψηλό ή υπερβολικό φορτίο και προκαλεί τη χρήση των υπολογιστικών πόρων στο μέγιστο βαθμό τους. Ο στόχος ενός τέτοιου τεστ διαφέρει από τη δοκιμή φορτίου, καθώς στοχεύει στον προσδιορισμό του σημείου κατάρρευσης του συστήματος, αλλά και στον τρόπο κατάρρευσής του. Με τη μέθοδο αυτή, σημειώνονται τα όρια και τα σημεία λειτουργίας που οδηγούν σε αυτά τα αποτελέσματα [7].
Στην παρούσα διπλωματική εργασία, η δοκιμή καταπόνησης διεξάγεται μέσω της αύξησης του πλήθους των ταυτόχρονων εικονικών χρηστών σε έναν υπερβολικά υψηλό αριθμό.

Εργαλεία δοκιμής φορτίου (load testing tools)

Για την επιβολή φορτίου με σκοπό τη μέτρηση της επίδοσης ενός υπολογιστικού συστήματος, χρησιμοποιείται ένα λογισμικό διεξαγωγής δοκιμής φορτίου. Το λογισμικό αυτό έχει τη δυνατότητα να προσομοιώνει εκατοντάδες χιλιάδες εικονικούς χρήστες, οι οποίοι επισκέπτονται το σύστημα ταυτόχρονα. Επιπλέον, δύναται να αναλύσει τα συλλεγόμενα αποτελέσματα και να δημιουργήσει στατιστικά με βάση τους χρόνους απόκρισης, το πλήθος των επιτυχημένων αιτημάτων ή και το ρυθμό απόδοσης (throughput) του συστήματος.
Η προσομοίωση των εικονικών χρηστών επιτυγχάνεται με τη δημιουργία κατάλληλων αιτημάτων προς τον εξυπηρετητή, και αποτελούν αιτήματα διαφόρων πρωτοκόλλων. Τα αιτήματα αυτά συλλέγονται σε ένα test script, το οποίο εν τέλει χρησιμοποιείται και για τη διεξαγωγή του τεστ. Με τη χρήση των εργαλείων δοκιμής φορτίου , μπορούν να ελεγχθούν πρωτόκολλα όπως το HTTP, JDBC για συνδέσεις με τη βάση δεδομένων, το FTP, το LDAP και γενικότερα web services.
Υπάρχουν διάφορα λογισμικά που επιτυγχάνουν αυτόν το σκοπό, με τα πιο γνωστά να είναι: το Apache JMeter, το BlazeMeter , το LoadRunner, το WebLOAD, αλλά και το Tsung [8], [9], [10], [11].
Στην παρούσα διπλωματική εργασία επιλέγεται η χρήση του Apache JMeter

Εργαλείο δοκιμής φορτίου Apache JMeter

Εικόνα 1.3: Λογότυπο Apache JMeter

Το Apache JMeter αποτελεί ένα από τα πιο γνωστά και υψηλής απόδοσης λογισμικά ανοικτού κώδικα (open source), πλήρως γραμμένο σε Java και χρησιμοποιείται για τη διεξαγωγή δοκιμής φορτίου [12].
Δύναται να υποστηρίξει τη δοκιμή φορτίου για όλα τα βασικά πρωτόκολλα και υπηρεσίες, όπως είναι τα:

- HTTP(S)
- SOAP / REST web services
- TCP
- LDAP
- FTP
- Mail όπως SMTP(S), POP3(S) και IMAP(S)
- JDBC για βάσεις δεδομένων
- Message-oriented middleware (MOM) μέσω JMS
- Java αντικειμένων.

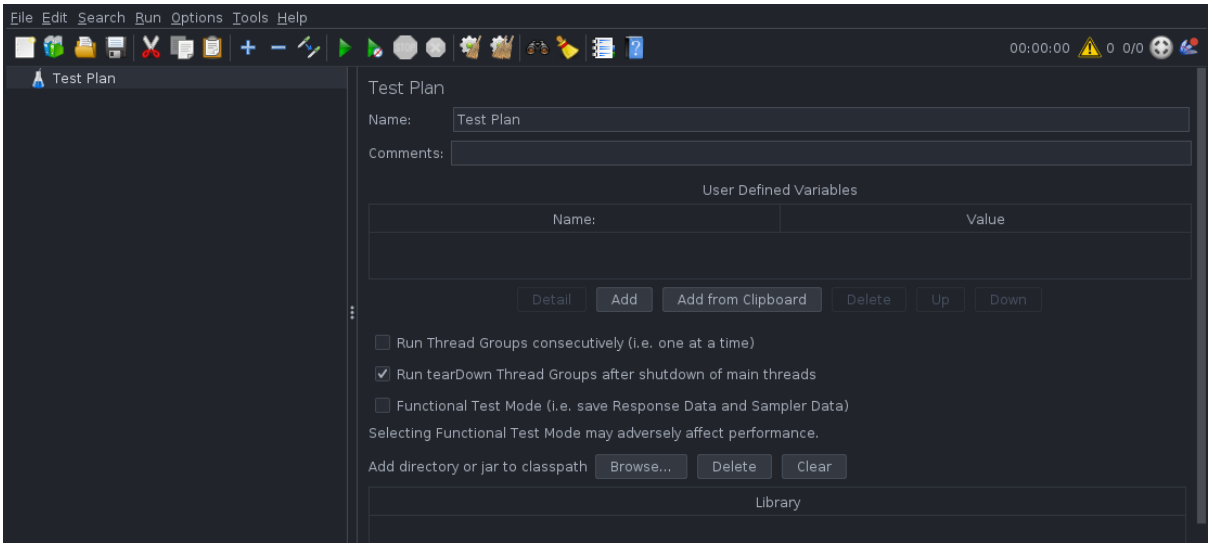
Το πλεονέκτημά του έγκειται στο γεγονός, ότι μπορεί να χρησιμοποιηθεί σε οποιαδήποτε πλατφόρμα, καθώς είναι γραμμένο σε Java, αρκεί η Java να είναι εγκατεστημένη.
Επιπλέον, για τη διεξαγωγή ενός τεστ με πολύ υψηλό αριθμό χρηστών επιτρέπει την δυνατότητα κατανεμημένης δοκιμής (distributed load testing), δημιουργώντας μια αρχιτεκτονική master-slave, προκειμένου να μοιραστεί ο φόρτος σε περισσότερα JMeter μηχανήματα.
Ακόμη, παρέχει πληθώρα από plugins, τα οποία μπορούν να εγκατασταθούν, ώστε να εμπλουτίσουν και να διευκολύνουν τη δημιουργία ενός test script.

Για τη διευκόλυνση δημιουργίας ενός τεστ παρέχεται η δυνατότητα να οριστεί σε έναν browser το JMeter να λειτουργεί ως proxy. Με αυτόν τον τρόπο, είναι δυνατή η καταγραφή των κινήσεων και των αιτημάτων κατά την περιήγηση σε μια ιστοσελίδα, με αποτέλεσμα να δημιουργείται με αυτοματοποιημένο τρόπο ένα test script, χωρίς να χρειάζεται να δημιουργηθεί χειροκίνητα.

Τέλος, το σημαντικότερο χαρακτηριστικό είναι ότι, λόγω της υποστήριξης πολλαπλών νημάτων (multithreading) μπορούν να προσομοιωθούν διαφορετικά Thread Groups με διαφορετική λειτουργία για το καθένα, με απώτερο σκοπό την μοντελοποίηση υψηλού αριθμού ταυτόχρονων εικονικών χρηστών.

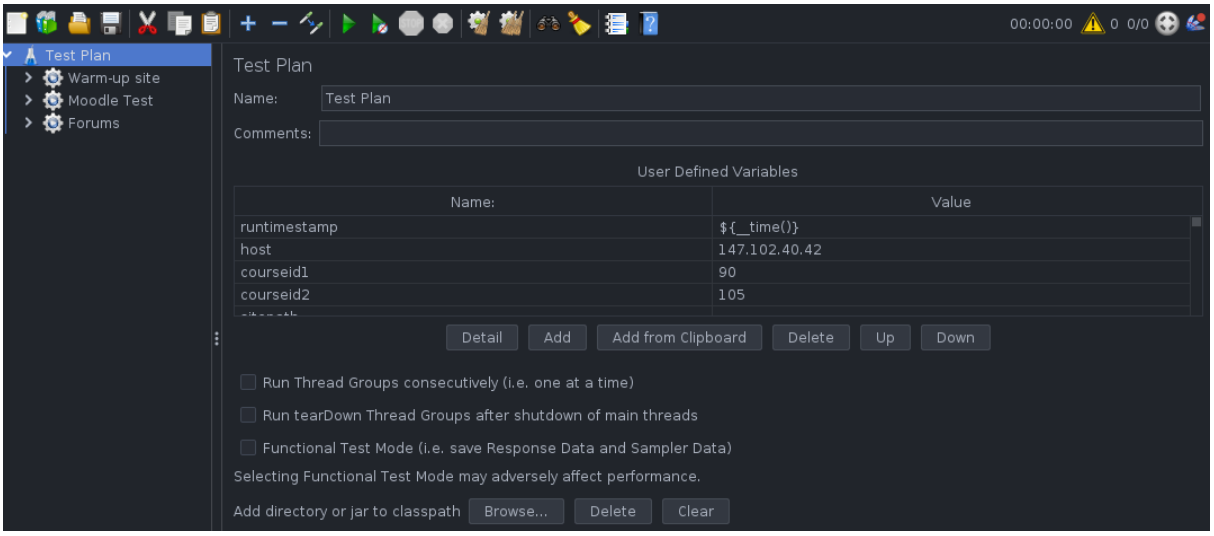
Κεφάλαιο 2: Apache JMeter

Γραφικό περιβάλλον χρήστη JMeter (GUI)



Εικόνα 2.a: Γραφικό περιβάλλον χρήστη JMeter (GUI)

Για να ανοίξουμε ένα JMeter Test Plan μεταβαίνουμε στο πεδίο File→Open και στη συνέχεια διαλέγουμε το επιθυμητό .jmx αρχείο. Το τελικό τεστ διακρίνεται παρακάτω:



Εικόνα 2.b: JMeter Test Plan

2.1: Δομή και ορθή σύνταξη ενός JMeter test

Το βασικό και δομικό στοιχείο ενός JMeter τεστ είναι το Test Plan. Αυτό, περιγράφει τη σειρά όλων των βημάτων που θα εκτελεστούν. Αποτελείται από Thread Groups, logic controllers, sample generating controllers, listeners, timers, assertions, και configuration elements, τα οποία επεξηγούνται στη συνέχεια [13].

Για τη διεξαγωγή σωστής δοκιμής φορτίου πρέπει να ληφθούν υπόψη αρκετές παράμετροι, οι οποίες είναι κρίσιμες για τη μοντελοποίηση των εικονικών χρηστών, προκειμένου να ανταποκρίνεται σε ένα ρεαλιστικό σενάριο. Έτσι, στη συνέχεια παρουσιάζεται και επεξηγείται το τελικό JMeter Test Plan, το οποίο προέκυψε από τροποποίηση του αυτοματοποιημένου τεστ που δημιουργήθηκε μέσω του Moodle, προκειμένου να αποτελεί ένα ρεαλιστικό σενάριο.

Test Plan

Το Test Plan είναι το πρώτο και κύριο στοιχείο ενός JMeter test script. Εδώ, καθορίζονται οι κύριες μεταβλητές με ισχύ και εμβέλεια σε όλο το τεστ καθώς και ο τρόπος εκτέλεσης των Thread Groups. Οι μεταβλητές καθορίζονται στον πίνακα 2.1, όπου καθορίζεται το όνομα και η τιμή (value) κάθε μεταβλητής.
Το τεστ που έχει δημιουργηθεί αποτελείται από τρία Thread Groups, τα οποία θα παρουσιαστούν στη συνέχεια αναλυτικά. Προς το παρόν, αναφέρουμε ότι επιλέγουμε να μην εκτελεστούν διαδοχικά, αλλά να εκκινήσουν ταυτόχρονα (με μια καθυστέρηση που εισάγεται σε κάποια Thread Groups).

User Defined Variables

Στο Test Plan υπάρχει η δυνατότητα να ορισθούν μεταβλητές στον πίνακα User Defined Variables όπως φαίνεται και από την εικόνα 2.b. Έτσι, ορίζουμε τις παρακάτω μεταβλητές, οι οποίες θα χρησιμοποιηθούν και από τα Thread Groups στη συνέχεια.

Name	Value
runtimestamp	${__time()}: Συνάρτηση που επιστρέφει την τρέχουσα ώρα σε ms (milliseconds) [14].
host	Η IP του Moodle, εν προκειμένω η 147.102.40.42
courseidX (X από 1-9)	Απαιτείται για το url κάθε course της μορφής: 147.102.40.42/course/view.php?id=**courseidX**
forumactivityidX (X από 1-9)	Απαιτείται για το url του forum thread κάθε μαθήματος της μορφής: 147.102.40.42/mod/forum/view.php?id=**forumactivityidX**

forumdiscussionidX (X από 1-9)	Απαιτείται για το url του Discussion 1 του forum κάθε μαθήματος της μορφής: 147.102.40.42/mod/forum/discuss.php?d=**forumdiscussionidX**
sendforumdiscussionidX (X από 1-9)	Απαιτείται για το url του Discussion 2 του forum κάθε μαθήματος της μορφής: 147.102.40.42/mod/forum/discuss.php?d=**sendforumdiscussionidX**
forumreplyidX (X από 1-9)	Απαιτείται για το url του πεδίου reply για το Discussion 2 κάθε μαθήματος της μορφής: 147.102.40.42/mod/forum/post.php?reply=**forumreplyidX**
allannouncementsX (X από 1-9)	Απαιτείται για το url ολόκληρου του announcements thread κάθε μαθήματος της μορφής: 147.102.40.42/mod/forum/view.php?id=**allannouncementsX**
announcementX (X από 1-9)	Απαιτείται για το url του Announcement 1 κάθε μαθήματος της μορφής:147.102.40.42/mod/forum/discuss.php?d=**announcementX**
assignmentidX (X από 1-9)	Απαιτείται για το url του assignment 1 κάθε μαθήματος της μορφής: 147.102.40.42/mod/assign/view.php?id=**assignmentidX**

Πίνακας 2.1: Περιεχόμενα πίνακα User Defined Variables

Παρατηρεί κανείς, ότι οι παραπάνω μεταβλητές αφορούν τα **9 courses** που έχουν δημιουργηθεί και έχουν σημειωθεί με X από 1-9 για εξοικονόμηση χώρου.

2.2: Thread Groups

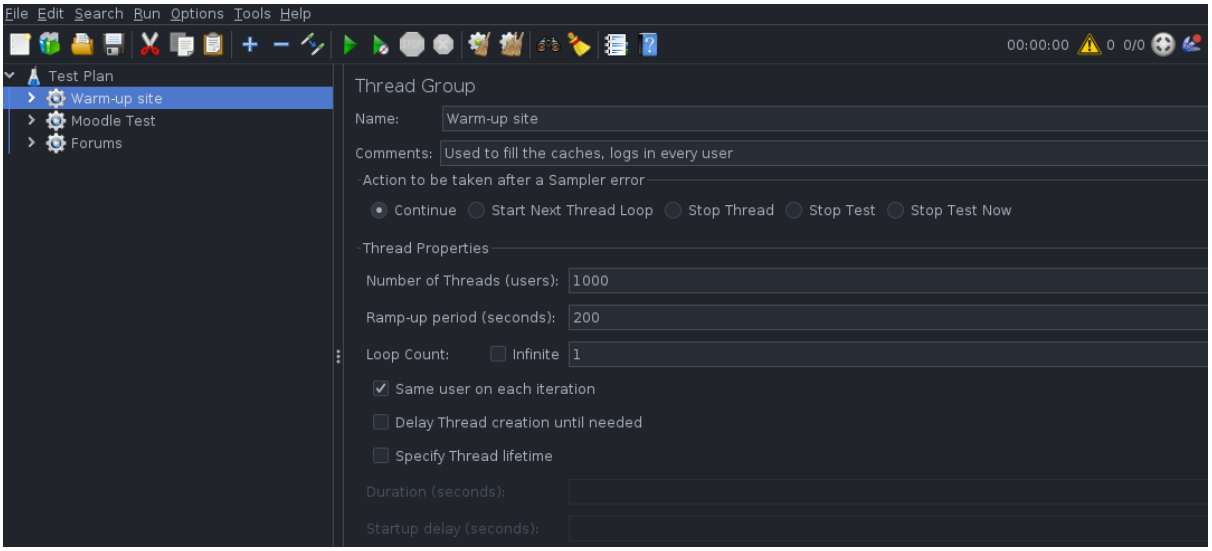
Τα Thread Groups αποτελούν τα αρχικά στοιχεία ενός JMeter Test Plan. Όλοι οι υπόλοιποι controllers και samplers πρέπει να ανήκουν κάτω από το Thread Group. Το Thread Group ελέγχει το πλήθος των νημάτων, δηλαδή των εικονικών χρηστών που θα προσομοιωθούν [15].
Όπως διακρίνεται παρακάτω, οι ρυθμίσεις που αφορούν το Thread Group είναι:

- **Number of Threads (users)**: καθορίζει το πλήθος των εικονικών χρηστών για την προσομοίωση.
- **Ramp-up period:** Ο χρόνος που χρειάζεται να παρέλθει, έως ότου το πλήθος των ενεργών χρηστών (threads) να είναι το επιλεγμένο. Για παράδειγμα, αν έχουν επιλεγεί 10 threads και η περίοδος ramp-up είναι 100 sec, τότε μετά από 100 δευτερόλεπτα θα είναι ενεργό το πλήθος των επιλεγμένων χρηστών. Με άλλα λόγια, κάθε 100/10=10 δευτερόλεπτα θα εκκινείται ένας νέος χρήστης.
  Η ορθή επιλογή του ramp-up period είναι πολύ σημαντική, καθώς ο σκοπός είναι να μην ταλαιπωρηθεί κατά την εκκίνηση του τεστ το σύστημα. Έτσι, θα πρέπει η

περίοδος να είναι μεγάλη, ώστε να αποφευχθεί ο μεγάλος φόρτος κατά την εκκίνηση, αλλά παράλληλα αντίστοιχα μικρή, ώστε το τελευταίο νήμα να εκκινήσει πριν τελειώσει το πρώτο.

- **Loop count:** Το πλήθος των επαναλήψεων που αφορούν την εκτέλεση του συγκεκριμένου Thread Group.

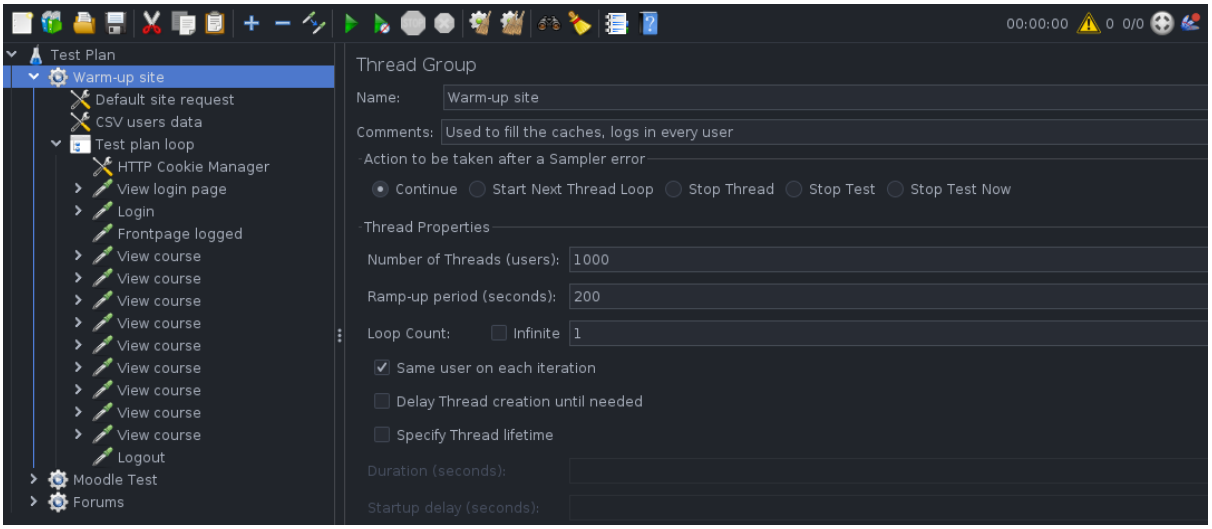
Επιπλέον, μπορεί να καθοριστεί και ο χρόνος καθυστέρησης εκκίνησης του Thread Group, αλλά και η διάρκεια ζωής του.



Εικόνα 2.2.1: Thread Group

2.3: Thread Group Warm-up site

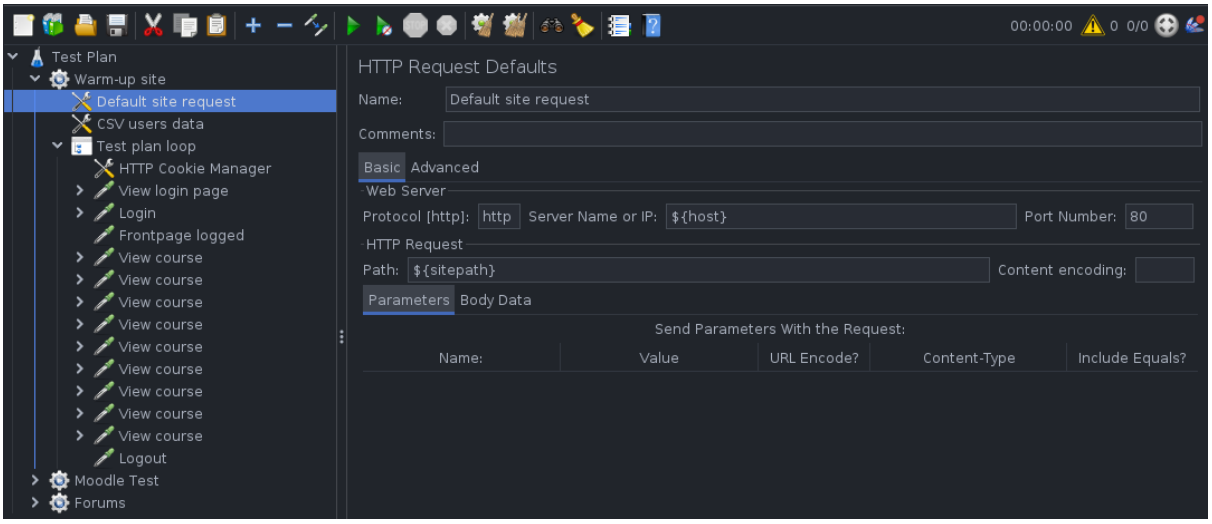
Το τελικό τεστ, περιλαμβάνει τρία Thread Groups. Το πρώτο Thread Group ονομάζεται Warm-up site. Ο ρόλος του είναι να προσομοιώσει το επιθυμητό πλήθος των χρηστών με σκοπό να αποθηκευτούν στην κρυφή μνήμη οι σύνοδοι των χρηστών. Η ενέργεια κάθε χρήστη είναι η σύνδεση (login) και στο τέλος η αποσύνδεση (logout).
Τα στοιχεία (elements) αυτού του Thread Group είναι εμφανή στην παρακάτω εικόνα.



Εικόνα 2.3.1: Thread Group Warm-up site

HTTP Request Defaults

Μέσω του στοιχείου HTTP Request Defaults, μπορούμε να ορίσουμε τις προκαθορισμένες μεταβλητές και τιμές για όλους τους HTTP Request controllers που θα χρησιμοποιηθούν στη συνέχεια (Login,View course κτλ βλ. εικόνα 2.3.1). Έτσι, σε περίπτωση που χρησιμοποιούνται πολλοί HTTP Request controllers δεν απαιτείται να ορίσουμε σε κάθε έναν ξεχωριστά τα πεδία Protocol, Server Name or IP και Port Number, αφού έχουν οριστεί ήδη μία φορά στο HTTP Request Defaults.
Επομένως, για το τελικό τεστ το πρωτόκολλο είναι το HTTP με port 80 και server name με value την τιμή της μεταβλητής host (χρησιμοποιούμε ${} για να αποκτήσουμε το value της μεταβλητής) που έχει οριστεί στον πίνακα User Defined Variables, με τις μεταβλητές στο Test Plan.



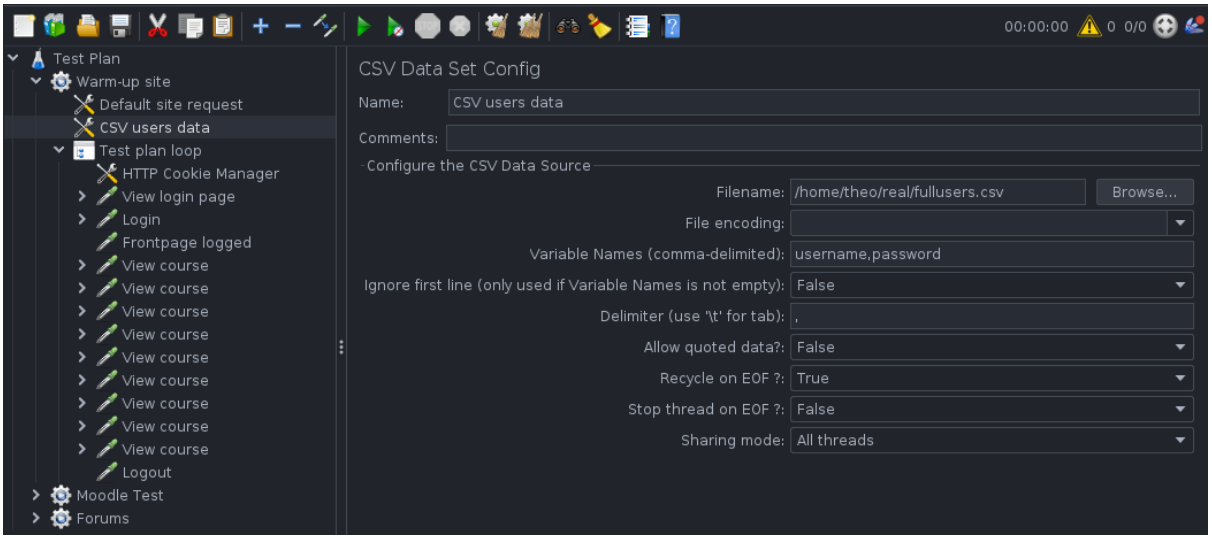
Εικόνα 2.3.2: HTTP Request Defaults

CSV Data Set Config

Με τη χρήση του CSV Data Set Config, είναι δυνατόν να αναγνωστούν δεδομένα από ένα αρχείο τύπου CSV. Πιο συγκεκριμένα, χρησιμοποιείται για την ανάγνωση των usernames και password κάθε χρήστη. Το CSV αρχείο αυτό δημιουργήθηκε από το Moodle κατά την αυτόματη δημιουργία ενός JMeter Test Plan στο κεφάλαιο 1.2. Το CSV έχει τη μορφή:

tool_generator_000001,YOURSECRET

όπου tool_generator_000001 είναι ο πρώτος χρήστης με password YOURSECRET κ.ο.κ.
Έτσι, για τη σωστή ανάγνωση του CSV αρχείου, επιλέγουμε την κατάλληλη διαδρομή του αρχείου στον εξυπηρετητή, στον οποίο έχει εγκατασταθεί το JMeter, εν προκειμένω το /home/theo/real/fullusers.csv, αλλά και τις υπόλοιπες ρυθμίσεις που φαίνονται στην παρακάτω εικόνα.



Εικόνα 2.3.3: CSV Data Set Config

Loop Controller

Το επόμενο στοιχείο που ακολουθεί είναι ένας loop controller, μέσω του οποίου μπορεί να καθοριστεί το πλήθος των επαναλήψεων που αφορούν τους samplers κάτω από αυτόν.

HTTP Cookie Manager

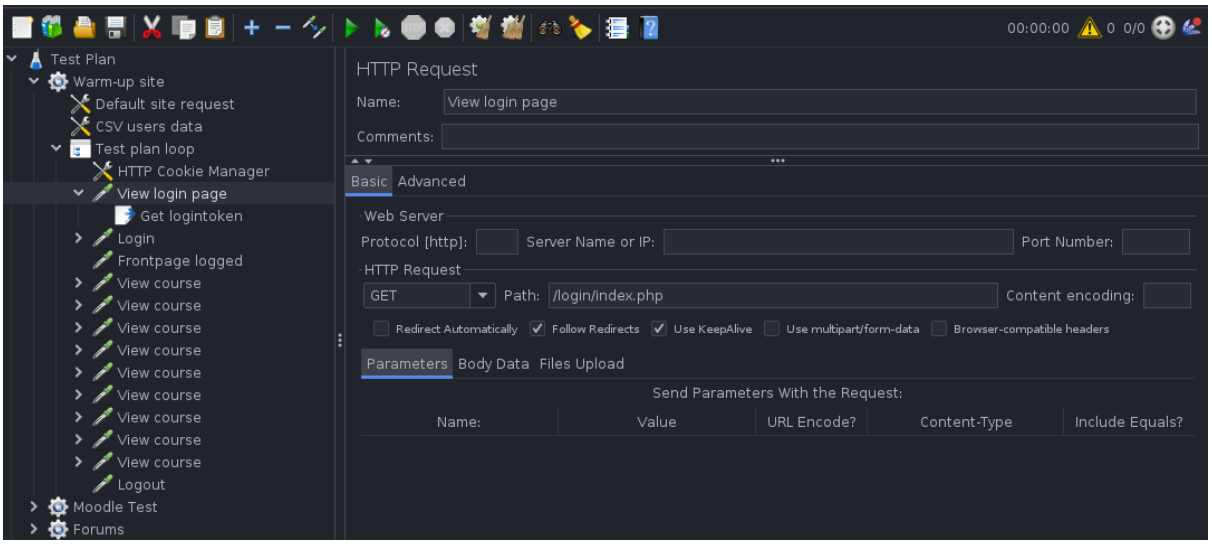
Το επόμενο στοιχείο που απεικονίζεται στην εικόνα 2.3.3 είναι το HTTP Cookie Manager, το οποίο χρησιμοποιείται για την αποθήκευση των cookies που συλλέγονται από μία απάντηση, αλλά και την αποστολή τους σε περίπτωση που αυτό κριθεί απαραίτητο. Επιπλέον, κάθε thread του JMeter τεστ έχει το δικό του χώρο αποθήκευσης των cookies [16].

HTTP Request

Αναφορικά με τον sampler HTTP Request, παρέχει τη δυνατότητα αποστολής HTTP/HTTPS αιτημάτων προς έναν εξυπηρετητή ιστού. Επιπλέον, μπορεί να καθορίσει τη μέθοδο αιτήματος GET ή POST, καθώς και επιπλέον μεταβλητές σε ένα αίτημα [17]. Τα HTTP αιτήματα που χρησιμοποιούνται, είναι εκείνα που φαίνονται στην εικόνα 2.3.3, δηλαδή τα View login page, Login, Frontpage logged, View course και logout. Στη συνέχεια, παρουσιάζεται αναλυτικά το περιεχόμενο των samplers αυτών. Η προεπιλεγμένη έκδοση του HTTP που χρησιμοποιείται από το JMeter είναι η HTTP 1.1 [18].

View login page

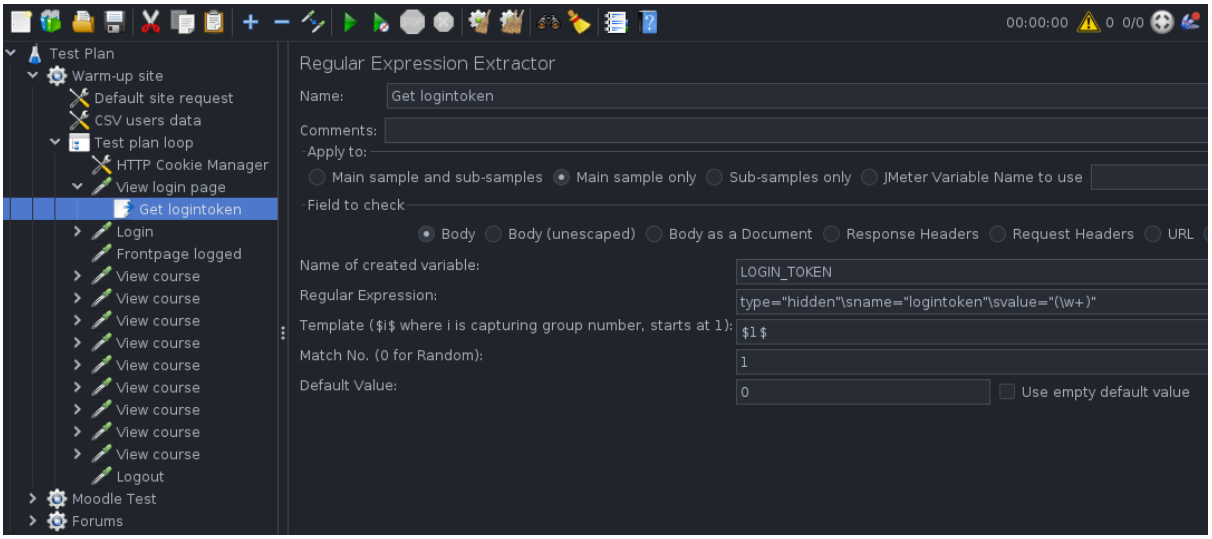
Όπως φαίνεται και στην παρακάτω εικόνα 2.3.4, ο συγκεκριμένος HTTP request sampler δημιουργεί ένα HTTP αίτημα προς τον εξυπηρετητή ιστού, προκειμένου να αποκτήσει την αρχική σελίδα που ορίζεται στη διαδρομή /login/index.php με τη μέθοδο GET. Εδώ, γίνεται εμφανής και η χρησιμότητα του HTTP Request Defaults, όπου δεν χρειάζεται να ορίσουμε εκ νέου τα πεδία Protocol, server name or IP και Port Number, εφόσον έχουν κληρονομηθεί από εκείνο.
Επίσης, με την επιλογή Use keepalive διατηρούνται ανοιχτές οι ήδη δημιουργημένες συνδέσεις με τον εξυπηρετητή, αρκεί στην καρτέλα advanced να επιλεγεί ως Implementation το HTTPClient4. Τέλος, με την επιλογή Follow redirects, είμαστε βέβαιοι ότι ελέγχεται αν υπάρχουν redirects και ακολουθούνται αν αυτά υπάρχουν.



Εικόνα 2.3.4: HTTP Request View login page

Regular Expression Extractor

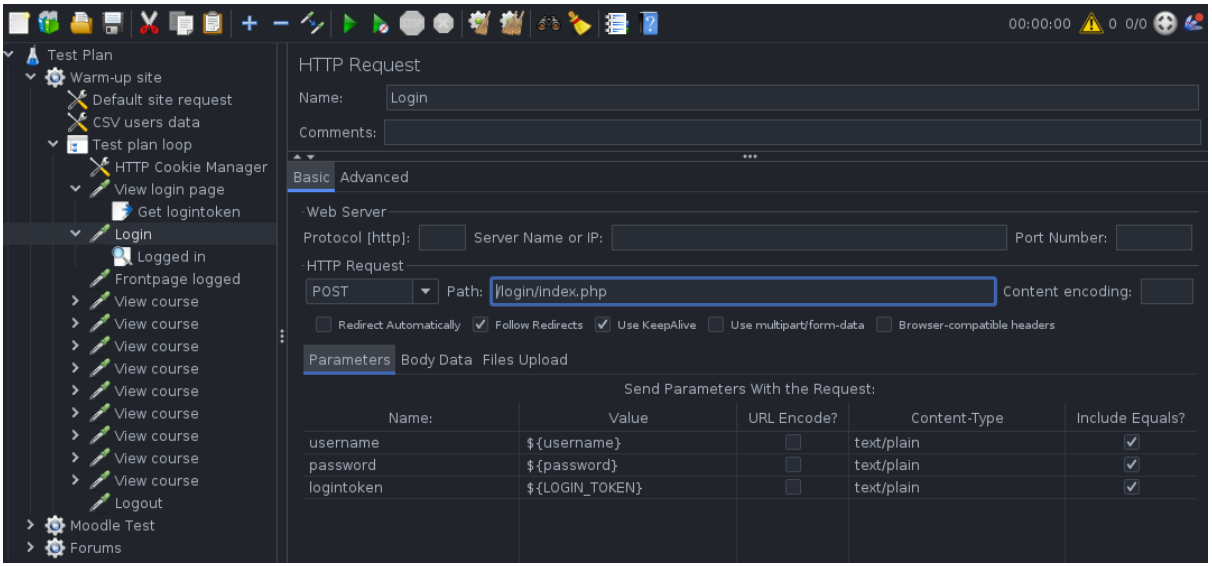
Ο Regular expression extractor χρησιμοποιείται για να εξαχθεί η τιμή του logintoken από τον εξυπηρετητή. Στο τεστ, έχει μετονομαστεί σε Get logintoken. Το logintoken, είναι ένα χαρακτηριστικό του Moodle που προσφέρει ασφάλεια από τρωτότητες που εκμεταλλεύονται τις συνόδους των χρηστών (users sessions) [19]. Οι λεπτομέρειες για τον τρόπο σύνταξης μπορούν να βρεθούν στην επίσημη ιστοσελίδα του Apache JMeter [20].



Εικόνα 2.3.5: Regular Expression Extractor για το logintoken

Login

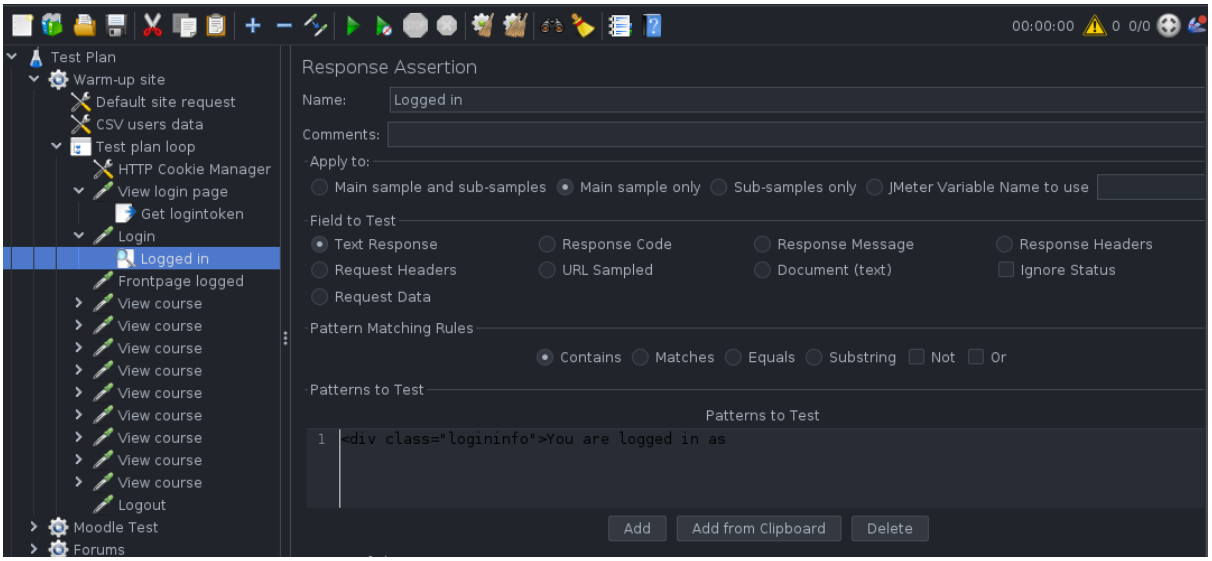
Το στοιχείο Login, αποτελεί τον HTTP sampler που χρησιμοποιείται για τη σύνδεση των χρηστών. Όπως διακρίνεται και από την παρακάτω εικόνα, πρόκειται για ένα HTTP αίτημα, όπου με τη μέθοδο POST αποστέλλονται στον εξυπηρετητή οι μεταβλητές username, password, logintoken και οι τιμές τους. Όπως προαναφέρθηκε, οι μεταβλητές username και password διαβάζονται από το CSV αρχείο, ενώ το logintoken αποκτάται από το προηγούμενο στοιχείο το Get logintoken όπως εξηγήθηκε προηγουμένως.



Εικόνα 2.3.6: HTTP Request login

Response Assertion

Παράλληλα με το παραπάνω HTTP αίτημα, προστίθεται και ένα στοιχείο του τεστ, το οποίο ελέγχει μέσα από το Text Response εάν οι συνδέσεις των χρηστών ήταν επιτυχημένες, εφόσον αυτά περιέχουν τη φράση: You are logged in as. Αυτός ο έλεγχος πραγματοποιείται με τη χρήση του Response assertion και με τον τρόπο που υποδεικνύεται στην παρακάτω εικόνα.



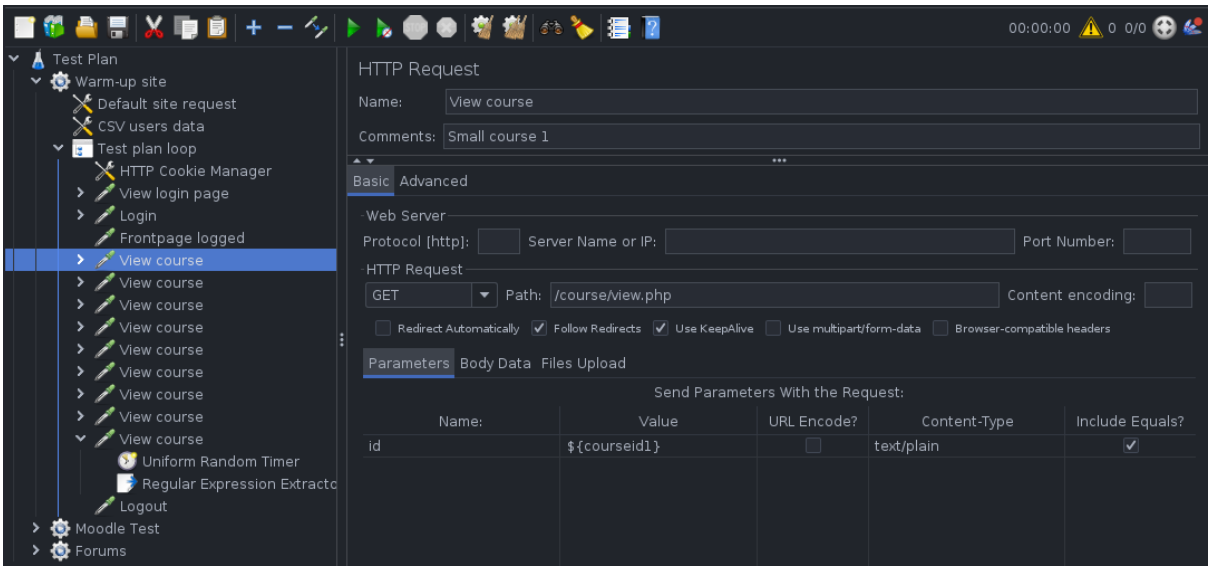
Εικόνα 2.3.7: Response Assertion και έλεγχος των συνδέσεων

Frontpage logged

Το στοιχείο Frontpage logged, χρησιμοποιείται για την αποθήκευση του χρόνου φόρτωσης της ιστοσελίδας μετά από κάθε σύνδεση.

View course

Το παρακάτω HTTP αίτημα, χρησιμοποιείται για την απόκτηση της ιστοσελίδας του course 1. Με τη μέθοδο GET, καθορίζονται οι μεταβλητές και οι τιμές που αφορούν στο course 1 εισάγοντας την τιμή της μεταβλητής courseid1 που ορίστηκε στην αρχή στο Test Plan.
Τα επόμενα οκτώ HTTP αιτήματα που ακολουθούν, αφορούν τα μαθήματα 2 έως 9 και απλώς εισάγεται το αντίστοιχο course id των μαθημάτων αυτών.



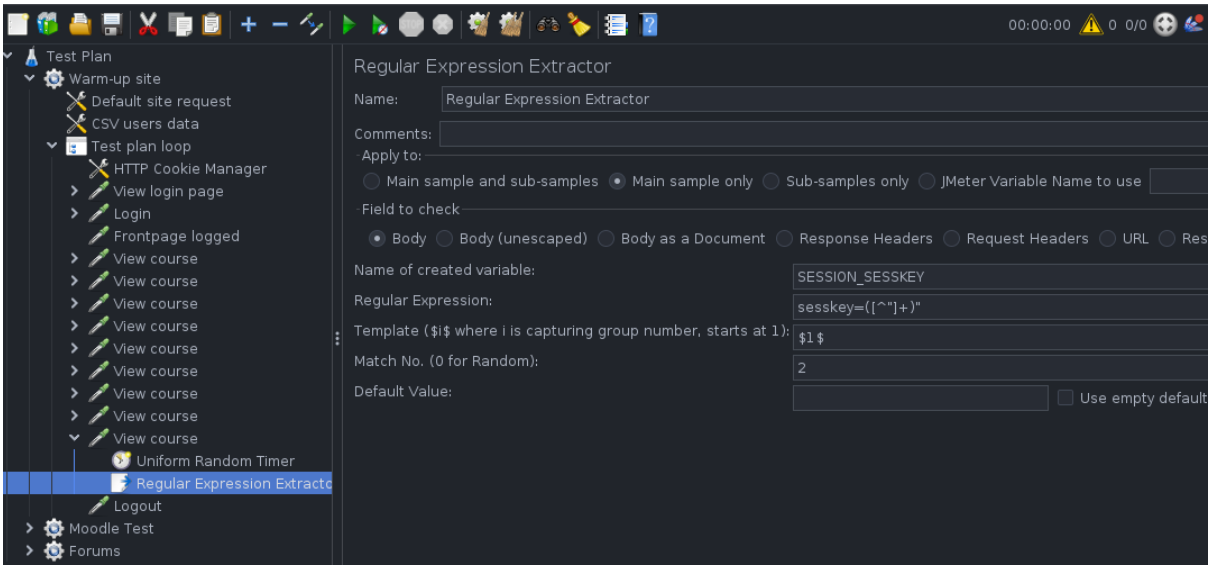
Εικόνα 2.3.8: HTTP Request View course

Session key

Το Moodle, χρησιμοποιεί πέρα από το logintoken και ένα session key που ονομάζεται και αλλιώς sesskey. Σε κάθε σύνδεση, το Moodle προσθέτει ένα τυχαίο string στη σύνοδο κάθε χρήστη.

Πριν από κάθε σημαντική ενέργεια όπως την υποβολή δεδομένων προστίθεται το sesskey στα δεδομένα προς αποστολή. Έτσι, πριν την εκτέλεση κάποιας ενέργειας επαληθεύεται η τιμή του sesskey με εκείνη στη σύνοδο και μόνο σε περίπτωση που ταιριάζουν πραγματοποιείται η ενέργεια [21].

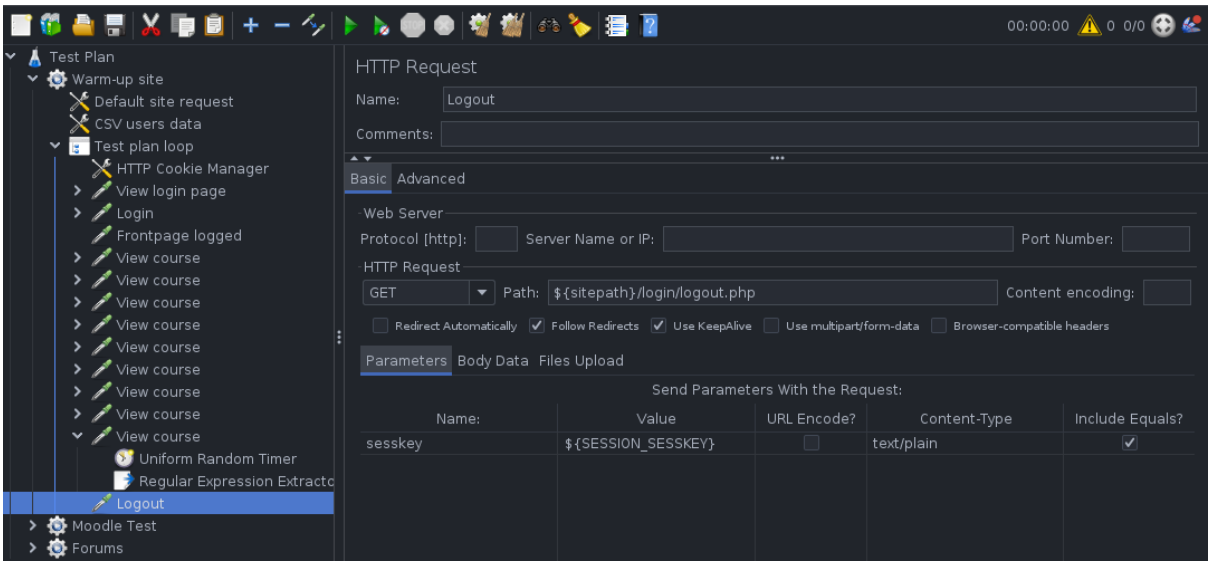
Ο τρόπος που αποκτάται το sesskey είναι παρόμοιος με το logintoken. Προσθέτουμε ένα Regular Expression extractor στο HTTP request για το View course με τα κατάλληλα expressions που απεικονίζονται στην παρακάτω εικόνα 2.3.9.



Εικόνα 2.3.9: Regular Expression Extractor για το session key

Logout

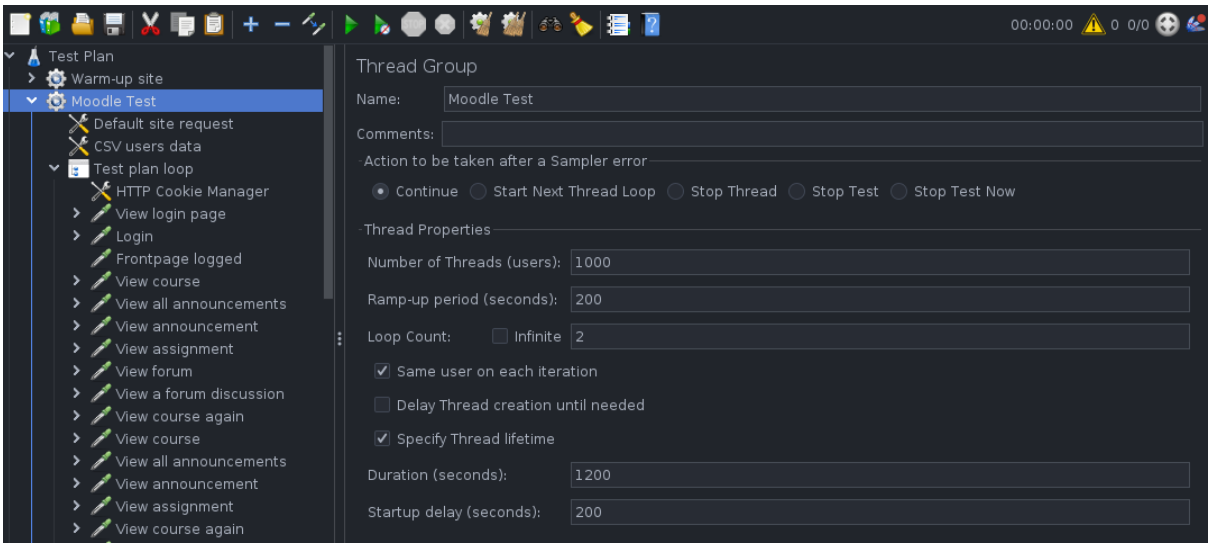
Το στοιχείο Logout, χρησιμοποιείται για την αποσύνδεση των χρηστών. Το μόνο που απαιτείται για την επιτυχή αποσύνδεση είναι η αποστολή του sesskey για την επαλήθευση της εγκυρότητας των χρηστών και αυτό πραγματοποιείται με τη μέθοδο GET και την αποστολή της μεταβλητής SESSION_KEY καθώς και της τιμής της.



Εικόνα 2.3.10: HTTP Request Logout

2.4: Thread Group Moodle Test

Το δεύτερο Thread Group το οποίο υπάρχει στο test script του JMeter ονομάζεται Moodle Test. Περιλαμβάνει τις κύριες και βασικές ενέργειες που εκτελεί ένας εικονικός χρήστης (π.χ ένας φοιτητής) κατά την περιήγησή του στο Moodle.
Οι ενέργειες αυτές προστέθηκαν στο τεστ έπειτα από μελέτη των ενεργειών που εκτελεί ένας φοιτητής στην πραγματικότητα. Έτσι, οι ενέργειες είναι αυτές που φαίνονται στην παρακάτω εικόνα.

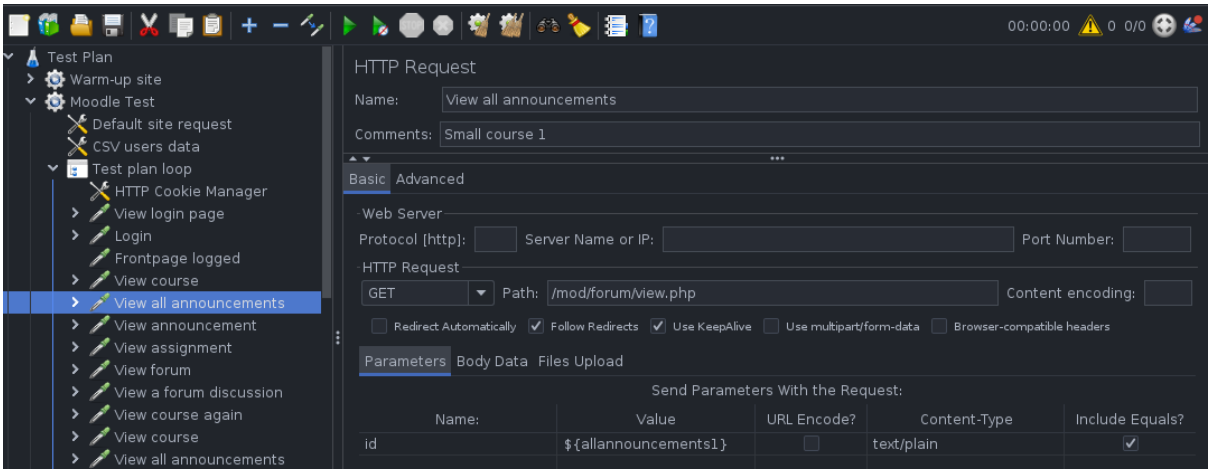


Εικόνα 2.4.1: Βασικές ενέργειες του Thread Group Moodle Test

Στην παραπάνω εικόνα απεικονίζονται οι ενέργειες που εκτελούν οι εικονικοί χρήστες σε αυτό το Thread Group. Όπως διακρίνεται στην εικόνα, κάποια HTTP αιτήματα είναι κοινά με αυτά στο Warm-up site (Login,View login page, Frontpage logged κτλ). Επομένως παρακάτω επεξηγούνται οι υπόλοιπες ενέργειες που διαφέρουν σε σχέση με πριν.

View all announcements

Όπως και στην περίπτωση των υπόλοιπων HTTP request samplers, έτσι και σε αυτό το HTTP request καθορίζουμε με κατάλληλο τρόπο τη διαδρομή που αφορά στο announcements thread του Moodle, καθώς και την τιμή (id) του στο πεδίο των παραμέτρων για το HTTP GET αίτημα, όπως διακρίνεται στην παρακάτω εικόνα.



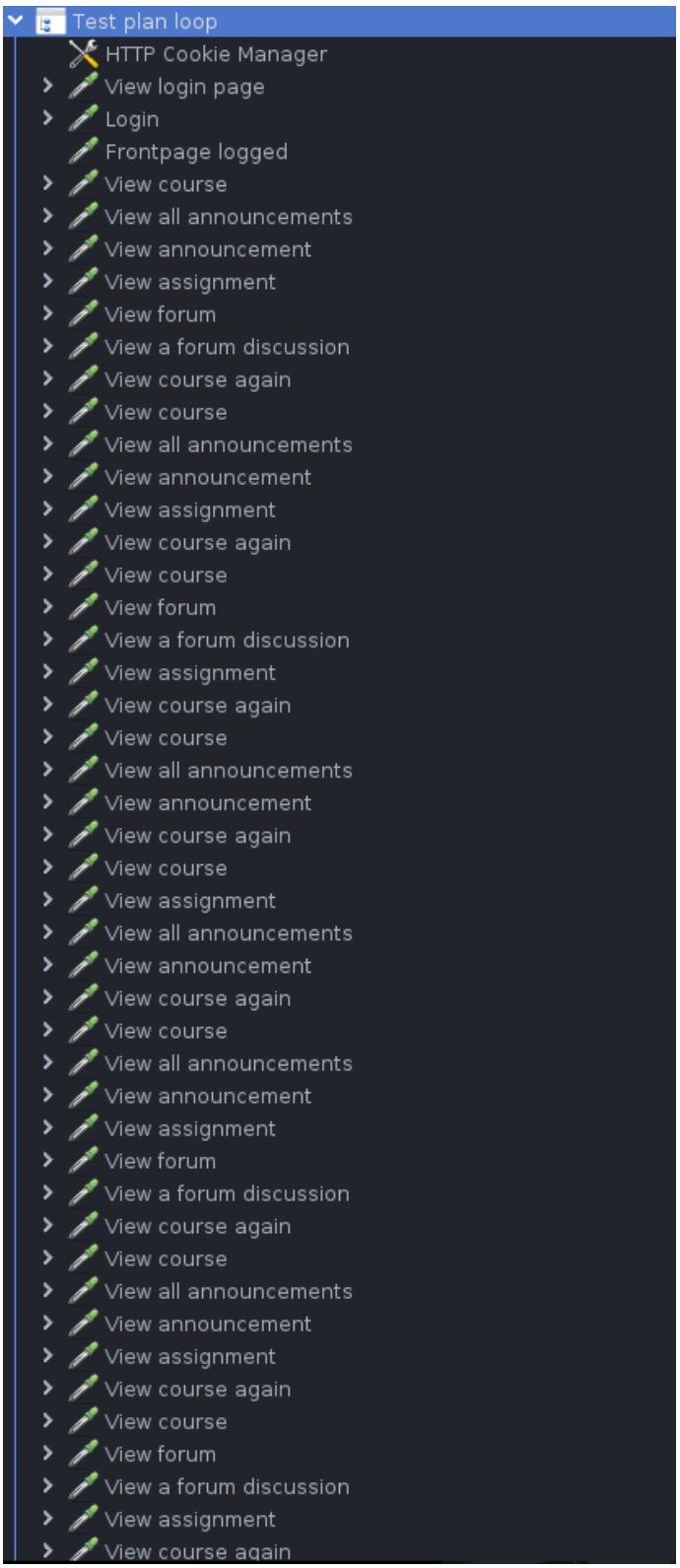
Εικόνα 2.4.2: HTTP Request View all announcements

Με εντελώς παρόμοιο τρόπο δημιουργούνται και τα υπόλοιπα HTTP αιτήματα ορίζοντας κατάλληλα τις μεταβλητές με τις τιμές τους στα πεδία GET και POST. Ανακεφαλαιώνοντας, οι κύριες ενέργειες που εκτελούνται είναι:

- **Login** (σύνδεση του χρήστη)
- **View course** (προβολή ενός μαθήματος)
- **View all announcements** (προβολή announcements thread με όλες τις ανακοινώσεις)
- **View announcement** (ανάγνωση μιας συγκεκριμένης ανακοίνωσης)
- **View assignment** (ανάγνωση εργασίας)
- **View forum** (προβολή forum thread με όλα τα post στο forum)
- **View a forum discussion** (ανάγνωση μιας συγκεκριμένης συζήτησης στο forum)
- **View course again** (ανανέωση του Moodle)
- **Logout** για την αποσύνδεση.

<u>Τρόπος εκτέλεσης των βασικών ενεργειών Moodle Test</u>

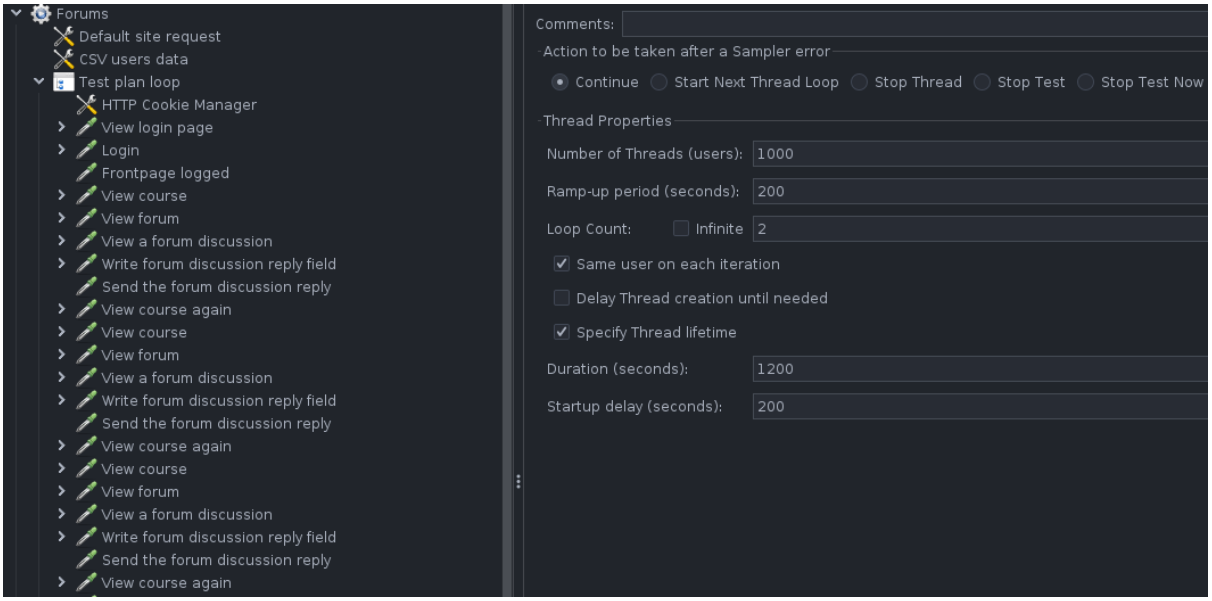
Προκειμένου να επιτευχθεί μεγαλύτερη τυχαιότητα των ενεργειών των χρηστών δε διενεργούνται τα ίδια HTTP αιτήματα σε κάθε course. Αντιθέτως, σε κάποια γίνεται ανάγνωση του forum και σε άλλα των ανακοινώσεων, όπως υποδεικνύεται στην παρακάτω εικόνα. Έτσι διενεργούνται διαφορετικές ενέργειες σε κάθε ένα από τα 9 μαθήματα που έχουν δημιουργηθεί.



Εικόνα 2.4.3: HTTP Requests του Moodle Test Thread Group

2.5: Thread Group Forums

Το τρίτο και τελευταίο Thread Group έχει όνομα Forums και συνίσταται κυρίως από ενέργειες που αφορούν το forum. Ο λόγος που έχει διαχωριστεί από το Moodle Test είναι για να μπορεί να επιλεγεί διαφορετικός αριθμός χρηστών, οι οποίοι γράφουν στο forum. Διαφορετικά, αυτό δε δύναται να πραγματοποιηθεί στο ίδιο Thread Group. Όπως θα εξηγηθεί σε επόμενο κεφάλαιο, ένα μικρό ποσοστό των χρηστών γράφει στο forum (όπως συμβαίνει και στα πραγματικά μαθήματα των ακαδημαϊκών ιδρυμάτων).
Οι ενέργειες που αφορούν στο forum απεικονίζονται στο παρακάτω στιγμιότυπο.

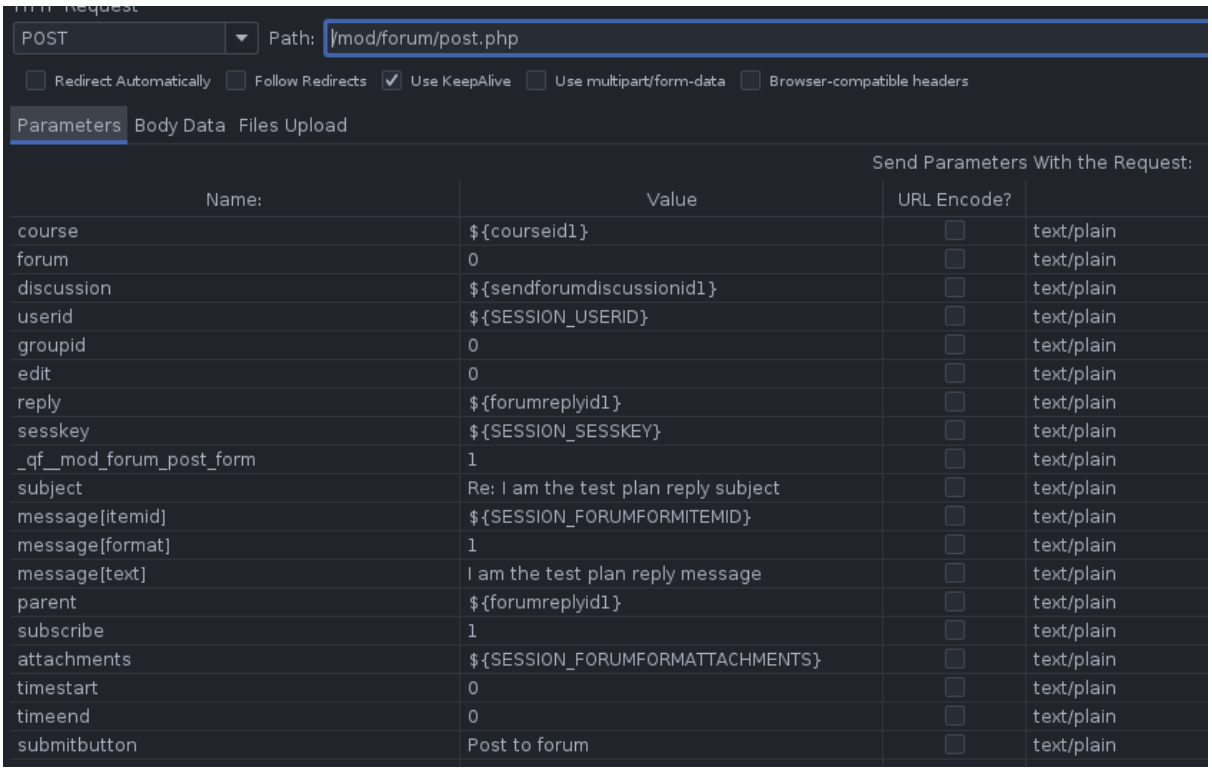


Εικόνα 2.5.1: Thread Group Forums

Οι ενέργειες που πραγματοποιούνται μέσω αυτού του Thread Group είναι οι εξής:

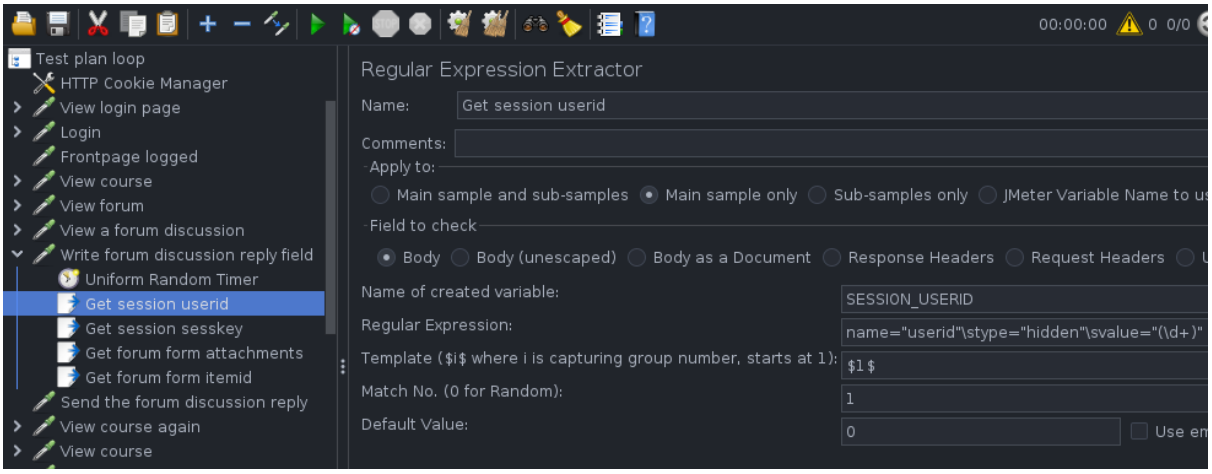
- **View forum** (προβολή forum thread με όλα τα post στο forum)
- **View a forum discussion** (ανάγνωση του Discussion 1 ενός course)
- **Write forum discussion reply filed** (σύνταξη κειμένου για την απάντηση στο Discussion 2 ενός course)
- **Send the forum discussion reply** (αποστολή του μηνύματος reply για το forum)
- **View course again** (ανανέωση του Moodle)
- **Logout** για την αποσύνδεση.

Ο τρόπος δημιουργίας των παραπάνω HTTP αιτημάτων δε διαφέρει καθόλου από τα προηγούμενα Thread Groups. Ωστόσο, απαιτείται προσοχή στη σύνταξη του **Send the forum discussion reply field,** καθώς απαιτούνται συγκεκριμένες παράμετροι προς αποστολή για την POST μέθοδο. Οι παράμετροι αυτές διακρίνονται στην παρακάτω εικόνα.



Εικόνα 2.5.2: Μεταβλητές για το HTTP Request Send the forum discussion reply

Έτσι, ο τρόπος απόκτησης των SESSION_USERID, SESSION_SESSKEY, SESSION_FORUMFORMATTACHMENTS και SESSION_FORUMFORMITEMID γίνεται με τη χρήση **Regular Expression Extractor** στον προηγούμενο HTTP request sampler.



Εικόνα 2.5.3: Απαιτούμενα sessions για την αποστολή του reply

το sesskey, όπως φαίνεται στην παραπάνω εικόνα έχει τη μορφή που παρουσιάστηκε προηγουμένως. Τα Get forum form attachments και Get forum form itemid έχουν την παρακάτω μορφή:

Name of created variable: SESSION_FORUMFORMATTACHMENTS
Regular Expression: value="(\d+)"\sname="attachments"\stype="hidden"

καθώς και αντίστοιχα:

Name of created variable: SESSION_FORUMFORMITEMID
Regular Expression: type="hidden"\sname="message\[itemid\]"\svalue="(\d+)"

Αναλογία πλήθους εικονικών χρηστών

Το JMeter τεστ αποτελείται από τα Thread Groups: Warm-up site, Moodle Test και Forums, τα οποία παρουσιάστηκαν αναλυτικά στα προηγούμενα κεφάλαια. Το πλήθος των νημάτων, δηλαδή των εικονικών χρηστών, είναι το ίδιο για το Warm-up site και το Moodle Test. Ωστόσο, για το Forums επιλέγουμε το πλήθος των χρηστών, οι οποίοι γράφουν κάποια απάντηση σε ένα forum ενός μαθήματος να είναι το **6%** των συνολικών χρηστών του Moodle Test.
Η επιλογή αυτής της αναλογίας, βασίζεται στην πραγματική συμπεριφορά ενός τέτοιου συστήματος σε ένα εκπαιδευτικό ίδρυμα ή οργανισμό. Συνήθως, είναι πολύ λίγοι οι χρήστες οι οποίοι διατυπώνουν ένα ερώτημα ή απαντούν στα forums.

Επομένως, τα τεστ εκτελέστηκαν για 500 ταυτόχρονους χρήστες (στα Moodle Test και Warm-up site) και 30 στο Forums. 600 χρήστες με 36 στο Forums κ.ο.κ.
Δηλαδή με κάθε αύξηση 100 χρήστων, αυξάνονται κατά 6 οι χρήστες στο Forums, διατηρώντας το ποσοστό του πλήθους των χρηστών στο Forums ίσο με το 6% των χρηστών του Moodle-test και Wam-up site. Το μέγιστο πλήθος ταυτόχρονων χρηστών που εξετάστηκε ήταν οι 2000 χρήστες (με 120 χρήστες στο Forums Thread Group).

2.6: Μοντέλο άφιξης χρηστών

Ο τρόπος με τον οποίο θα μπορούσαν να μοντελοποιηθούν οι αφίξεις των χρηστών στο σύστημα του Moodle είναι μέσω μιας διαδικασίας Poisson. Οι αφίξεις των χρηστών στην πραγματικότητα, συχνά μπορούν να θεωρηθούν ως αφίξεις Poisson.
Οι προϋποθέσεις για να υφίσταται μία διαδικασία Poisson είναι οι εξής:

- Να είναι γνωστός ο **σταθερός** ρυθμός των αφίξεων
- Οι αφίξεις των χρηστών να είναι ανεξάρτητες μεταξύ τους
- Η πιθανότητα εμφάνισης ενός χρήστη να είναι η ίδια σε οποιοδήποτε διάστημα, ανεξάρτητα από την θέση του χρονικού διαστήματος.

Έτσι, η διαδικασία Poisson για ένα πλήθος χρηστών που καταφθάνουν στο σύστημα σε ένα χρονικό διάστημα μεγαλύτερο από το χρόνο απόκρισης του συστήματος (ο οποίος ακολουθεί εκθετική κατανομή), αποτελεί μία διαδικασία Poisson.

Η διαδικασία Poisson είναι βολική για την περιγραφή των αφίξεων, καθώς αποτελεί μία ικανοποιητική προσέγγιση της πραγματικότητας και μπορεί να υλοποιηθεί μέσω του JMeter. Ο τρόπος προσέγγισης μιας διαδικασίας Poisson επιτυγχάνεται μέσω του **Precise throughput timer** και όχι μέσω του Poisson random timer όπως θα εξηγηθεί παρακάτω.
Ωστόσο στα τεστ που εκτελέστηκαν παρουσιάστηκαν πολλά σφάλματα στα HTTP αιτήματα στο συγκεκριμένο timer και επομένως εγκαταλείφθηκε το μοντέλο αυτό.

Χρόνος σκέψης (Thinking Time)

Η βασικότερη παράμετρος για την δημιουργία μιας ορθής δοκιμής φορτίου είναι η εισαγωγή χρόνων σκέψης πριν από την ενέργεια κάθε χρήστη. Όπως αναφέρθηκε και στην εισαγωγή, επειδή μια δοκιμή φορτίου αποτελεί μία προσομοίωση της πραγματικής συμπεριφοράς των χρηστών και της κανονικής λειτουργίας του συστήματος, είναι αναγκαία η ύπαρξη του χρόνου σκέψης.

Στην πραγματικότητα, ένας χρήστης που επισκέπτεται το Moodle δεν εκτελεί τη μια ενέργεια αμέσως μετά την προηγούμενή του. Για αυτόν το λόγο, λαμβάνεται υπόψη ο χρόνος σκέψης στη δημιουργία του τεστ. Αν δεν ληφθεί υπόψη, το τεστ δεν αποτελεί μια μοντελοποίηση της πραγματικής συμπεριφοράς των χρηστών, καθώς το φορτίο που εισάγεται στο σύστημα είναι πολύ μεγαλύτερο από το πραγματικό. Έτσι, δεν θα μπορεί να υπολογιστεί προσεγγιστικά το μέγιστο πλήθος ταυτόχρονων χρηστών (concurrent users) που αντέχει το σύστημα.
Πρέπει να τονιστεί πως η παράλειψη του χρόνου σκέψης δεν συνιστά δοκιμή καταπόνησης, παρόλο που μπορεί να εισαχθεί μεγάλο φορτίο με αυτή την τεχνική. Ο λόγος είναι ότι η απόκριση του συστήματος δεν μπορεί να προβλεφθεί σε αυτήν την περίπτωση.
Επομένως, πρέπει να δοθεί μεγάλη έμφαση στη μελέτη της συμπεριφοράς των χρηστών σε ένα σύστημα πριν από τη δημιουργία ενός τεστ δοκιμής φορτίου και κατ' επέκταση δοκιμής καταπόνησης.

JMeter Timers

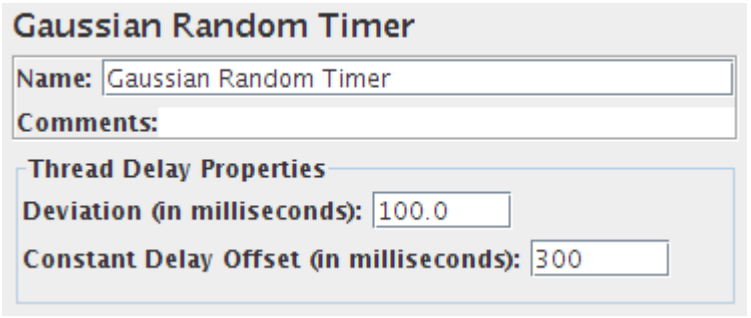
Υπάρχουν αρκετά είδη timers που δημιουργούν επιθυμητή, αλλά τυχαία σε κάποιο επιλεγμένο χρονικό διάστημα **χρονοκαθυστέρηση** [22]. Οι κύριοι είναι οι:

- Constant timer
- Gaussian random timer
- Poisson random timer
- Uniform random timer
- Precise throughput timer

Στην παρούσα εργασία χρησιμοποιείται ο Gaussian random timer, ο Uniform random timer και ο Precise throughput timer, οι οποίοι επεξηγούνται στη συνέχεια.

Gaussian random timer

Ο συγκεκριμένος timer προκαλεί καθυστέρηση, η οποία είναι μία απόκλιση (ορίζεται στο πεδίο deviation) από την τιμή του constant delay offset και έτσι η κατανομή των προκυπτουσών τιμών ακολουθεί την κανονική κατανομή.



Εικόνα 2.6.1: Gaussian random timer [22]

Uniform random timer

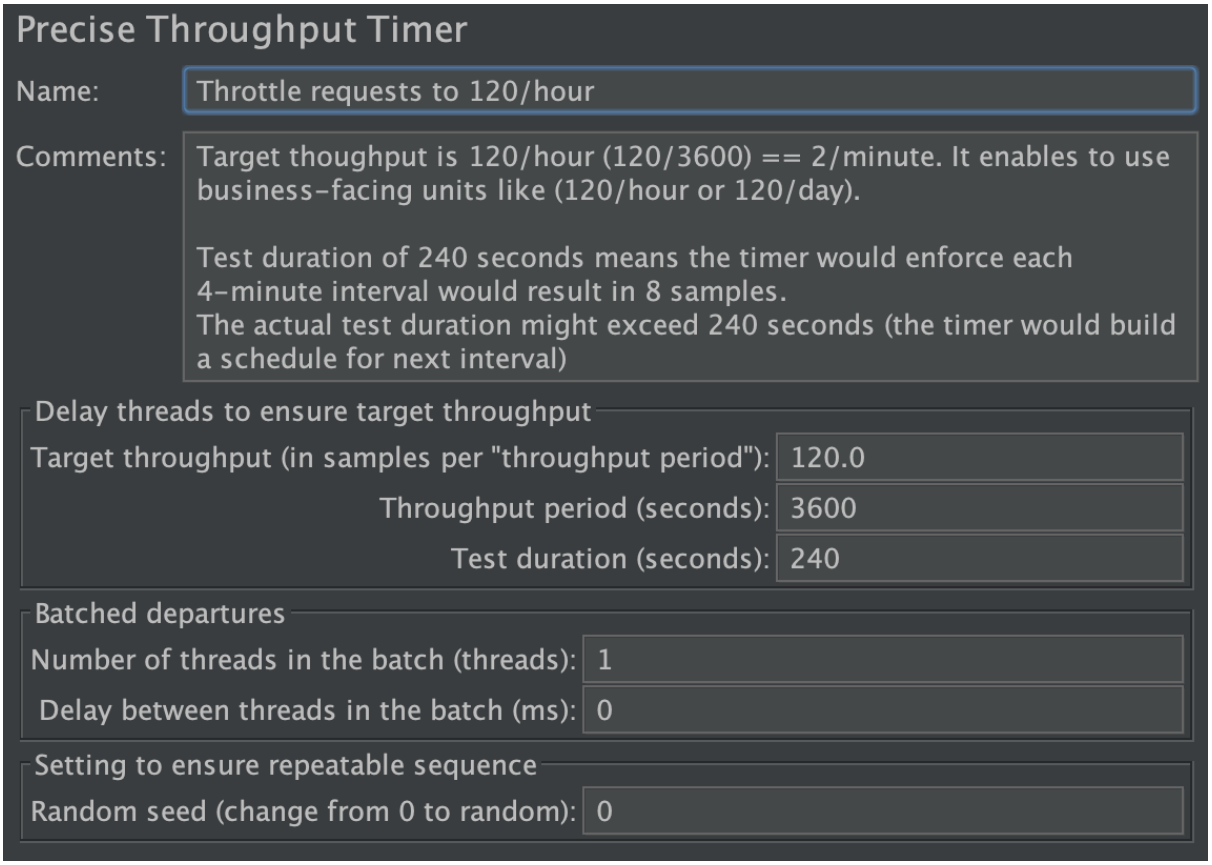
Ο Uniform random timer χρησιμοποιείται για την εισαγωγή τυχαίας καθυστέρησης στο ορισμένο διάστημα: [random delay maximum, random delay maximum+constant delay offset].
Η μεταβλητή random delay maximum παίρνει τιμές από 0 έως την τιμή που επιλέγουμε. Για παράδειγμα, για constant delay offset = 2000ms και random delay = 1000ms δημιουργούμε μια τυχαία καθυστέρηση στο διάστημα [2000ms, 3000ms].
Κάθε χρονική καθυστέρηση έχει την ίδια πιθανότητα να δημιουργηθεί με οποιαδήποτε άλλη στο ορισμένο διάστημα.

Εικόνα 2.6.2: Uniform random timer [22]

Precise throughput timer

Ο timer αυτός εισάγει κατάλληλη χρονική καθυστέρηση, προκειμένου να διατηρηθεί το επιθυμητό throughput (σε samples/min) όσο πιο κοντά γίνεται στην επιλεγμένη τιμή. Φυσικά, ο τελικός ρυθμός απόδοσης του συστήματος (throughput) μπορεί να είναι μικρότερος εάν το σύστημα δεν έχει επαρκή ισχύ για να αντέξει την επιλεγμένη τιμή, ή αν άλλοι timers εισάγουν περισσότερη καθυστέρηση μειώνοντας έτσι το συνολικό πλήθος των threads ανά λεπτό [23].

Ο precise throughput timer μπορεί να χρησιμοποιηθεί για τη μοντελοποίηση **αφίξεων Poisson.** Αυτό, επιτυγχάνεται με τον ορισμό του επιθυμητού ρυθμού των αφίξεων, με αποτέλεσμα ο συγκεκριμένος timer να δημιουργεί κατάλληλες καθυστερήσεις στα samples, με στόχο να διατηρηθεί ο συνολικός ρυθμός των αφίξεων σταθερός.
Παρόλο που χρησιμοποιήθηκε ο συγκεκριμένος timer για τη μοντελοποίηση τέτοιων αφίξεων, παρουσιάζει αρκετά σφάλματα κατά την εκτέλεση της δοκιμής φορτίου και έτσι εγκαταλείφθηκε.



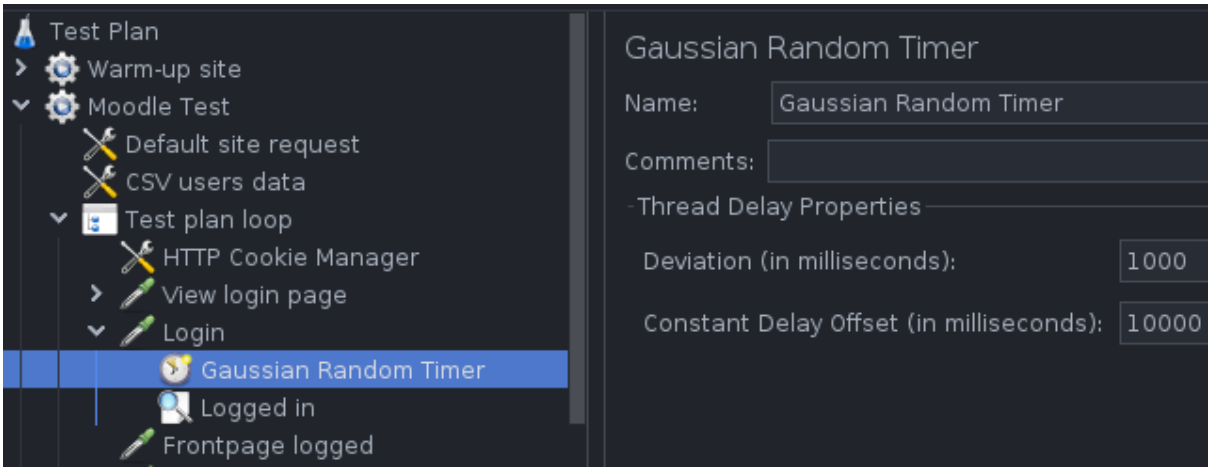
Εικόνα 2.6.3: Precise throughput timer [22]

Εφαρμογή χρόνων σκέψης στα Thread Groups

Οι χρόνοι σκέψης στα Thread Groups εισάγονται στους HTTP request samplers. Στο πρώτο Thread Group δεν έχουν προστεθεί χρόνοι σκέψης, καθώς ο σκοπός του είναι η αποθήκευση στην κρυφή μνήμη των διαπιστευτηρίων των χρηστών (user credentials) και συνόδων. Επομένως προστίθενται χρόνοι σκέψης στο κύριο Thread Group που είναι το Moodle Test αλλά και στο Forums.

<u>Χρόνοι σκέψης στο Moodle Test Thread Group</u>

Είναι προφανές, ότι οι κύριες ενέργειες που αφορούν το συγκεκριμένο Thread Group απαιτούν κάποιο χρόνο προετοιμασίας πριν εκτελεστούν. Ο τρόπος με τον οποίο προσθέτουμε το χρόνο σκέψης είναι μέσω καθυστέρησης που παρέχουν οι timers που προαναφέρθηκαν. Όπως διακρίνεται και στην παρακάτω εικόνα, προσθέτουμε κάτω από κάθε HTTP request sampler έναν timer, του οποίου η εμβέλεια (scope) είναι τοπική και αφορά μόνο στο συγκεκριμένο HTTP αίτημα.



Εικόνα 2.6.4: Gaussian random timer για το HTTP Request Login

Από την παραπάνω εικόνα διαφαίνεται επίσης, ότι ορίζεται ο χρόνος πληκτρολόγησης του username και του password για τη σύνδεση να είναι από 9 έως 10 δευτερόλεπτα.
Με παρόμοιο τρόπο, ορίζεται ο χρόνος σκέψης και στους υπόλοιπους HTTP request samplers, χρησιμοποιώντας ωστόσο τον uniform random timer κατάλληλα. Έτσι, παρακάτω παρουσιάζονται οι χρόνοι σκέψης για το συγκεκριμένο Thread Group.

- **Login:** Απαιτούνται κατά μέσο όρο για μία σύνδεση 10 δευτερόλεπτα (10000 ms). Οπότε με τον Gaussian random timer θέτουμε χρόνο σκέψης **9-11 δευτερολέπτων.**

- **Frontpage logged:** δε χρησιμοποιείται thinking time, αφού απλώς αποκαλύπτει, τον απαιτούμενο χρόνο για την εμφάνιση της ιστοσελίδας μετά από τη σύνδεση.

- **View course:** Απαιτούνται περίπου **4-6 δευτερόλεπτα** για την επιλογή ενός συγκεκριμένου μαθήματος προς προβολή. Για το σκοπό αυτό χρησιμοποιείται ο uniform random timer, όπως και στους επόμενους samplers που ακολουθούν.

- **View all announcements:** Απαιτούνται περίπου **3-5 δευτερόλεπτα** για να βρεθεί και να επιλεγεί το πεδίο με όλες τις ανακοινώσεις.

- **View announcement:** Απαιτούνται περίπου **10-15 δευτερόλεπτα** για να επιλεγεί και να ολοκληρωθεί η ανάγνωση μιας ανακοίνωσης.

- **View assignment:** Απαιτούνται περίπου **10-15 δευτερόλεπτα** για να διαβαστεί ένα assignment (εργασία).

- **View forum:** Απαιτούνται **3-4 δευτερόλεπτα** για να επιλεγεί το thread με τα forum posts.

- **View a forum discussion:** Απαιτούνται **10-15 δευτερόλεπτα** για την ανάγνωση ενός discussion του forum.

- **View course again:** Εισάγεται **10-15 δευτερόλεπτα** παύση, αφού ο χρήστης ανανεώσει τη σελίδα και στη συνέχεια μένει απλώς συνδεδεμένος στο σύστημα.

- **Logout:** Εισάγεται επιπλέον χρόνος αδράνειας διάρκειας **15-25** δευτερολέπτων πριν το logout κάθε χρήστη.

Οι χρόνοι σκέψης αυτοί προέκυψαν ύστερα από τη μελέτη του χρόνου εκτέλεσης αυτών των ενεργειών στην υποδομή του Ε.Μ.Π το οποίο βασίζεται στο Moodle.

<u>Χρόνοι σκέψης στο Forums Thread Group</u>

Όπως στην περίπτωση του Moodle Test, έτσι και εδώ είναι αναγκαία η χρήση χρόνων σκέψης, προκειμένου να μοντελοποιηθεί ένα ρεαλιστικό σενάριο δοκιμής φορτίου. Έτσι, παρουσιάζονται οι παρακάτω χρόνοι σκέψης που αφορούν τις ενέργειες του forum.

- **Login: 9-11 δευτερόλεπτα**, όπως στο Moodle Test.
- **View course: 4-6 δευτερόλεπτα**, όπως στο Moodle Test.
- **View forum: 3-4 δευτερόλεπτα**, όπως στο Moodle Test.
- **View a forum discussion:** Απαιτούνται **20-40 δευτερόλεπτα** για την ανάγνωση ενός discussion του forum.
- **Write forum discussion reply field:** Απαιτούνται περίπου **1-2 λεπτά** για να γραφεί το κείμενο για το reply ενός forum.
- **Send the forum discussion reply**: δε χρησιμοποιείται χρόνος σκέψης, καθώς ο συγκεκριμένος sampler απλώς χρησιμοποιείται για την αποστολή της απάντησης στο forum.

- **View course again:** Εισάγεται **20-30 δευτερόλεπτα** παύση, αφού ο χρήστης ανανεώσει την σελίδα και στη συνέχεια παραμένει απλώς συνδεδεμένος στο σύστημα.
- **Logout:** Εισάγεται επιπλέον χρόνος αδράνειας διάρκειας **10-20** δευτερολέπτων πριν το logout κάθε χρήστη.

Ανακεφαλαιώνοντας, στο τεστ που έχει δημιουργηθεί χρησιμοποιούνται τρία Thread Groups. Όπως θα φανεί και στη συνέχεια, πρώτα εκτελείται το Warm-up site και αφού τελειώσει εκκινούν ταυτόχρονα το Moodle Test και το Forums. Έτσι, το μεγαλύτερο ποσοστό των εικονικών χρηστών εκτελεί τις ενέργειες του Moodle Test, ενώ ένα μικρό ποσοστό διαβάζει και γράφει στα forums.

Εκτέλεση δοκιμής φορτίου μέσω CLI mode (NON GUI mode)

Για την διεξαγωγή της δοκιμής φορτίου με τη χρήση του JMeter, είναι αναγκαία η χρήση του εργαλείου σε CLI (Command Line Interface) mode και όχι μέσω του GUI. Ο λόγος είναι κυρίως η χαμηλότερη κατανάλωση μνήμης και του επεξεργαστή. Επιπλέον, μέσω του CLI mode, παρέχεται η δυνατότητα αποθήκευσης του τεστ σε CSV ή XML αρχείο, καθώς και η δημιουργία ενός HTML report με την ανάλυση των αποτελεσμάτων [24].
Επομένως, ο τρόπος με τον οποίο εκτελούμε ένα τεστ είναι μεταβαίνοντας στον φάκελο /bin όπου είναι εγκατεστημένο το JMeter και εκτελούμε την εντολή:

```
sudo ./jmeter -n -t /home/theo/loadtests/test.jmx -l /home/theo/loadtests/test.jtl -e -o /home/theo/loadtests/test
```

Όπου χρησιμοποιούμε τις επιλογές:

- **-n**: Με αυτόν τον τρόπο ορίζουμε το CLI mode.
- **-t**: Ορίζουμε το όνομα του JMX αρχείου που περιέχει το τεστ.
- **-l**: Ορίζουμε το αρχείο jtl, στο οποίο θα εγγραφούν τα αποτελέσματα.
- **-o**: Ορίζουμε το φάκελο, στον οποίο θα αποθηκευτεί το HTML report που δημιουργεί το JMeter.

Έτσι, με την παραπάνω εντολή εκτελούμε το πρόγραμμα σε CLI mode, όπου γίνεται η ανάγνωση του jmx αρχείου που περιέχει το τεστ, και αποθηκεύουμε τα αποτελέσματα σε ένα jtl αρχείο, ενώ σε έναν άδειο φάκελο αποθηκεύουμε το HTML report που δημιουργείται μετά το πέρας της εκτέλεσης του τεστ.
Το αρχείο jtl είναι είναι ένα text file, στο οποίο αποθηκεύονται τα αποτελέσματα και το οποίο μπορεί να χρησιμοποιηθεί για debugging αποκαλύπτοντας τα αποτελέσματα κάθε αιτήματος ξεχωριστά.

2.7: Τρόπος εμφάνισης των αποτελεσμάτων του JMeter test

Για να παρουσιαστεί ο τρόπος με τον οποίο προβάλλονται τα αποτελέσματα των τεστ, χρησιμοποιούνται τα αποτελέσματα της πρώτης δοκιμής στο σύστημα του Moodle ως αυτοδύναμου εξυπηρετητή. Το πρώτο τεστ, αποτελείται από 530 ταυτόχρονους χρήστες συνολικά (οι 30 χρήστες είναι στα Forums).

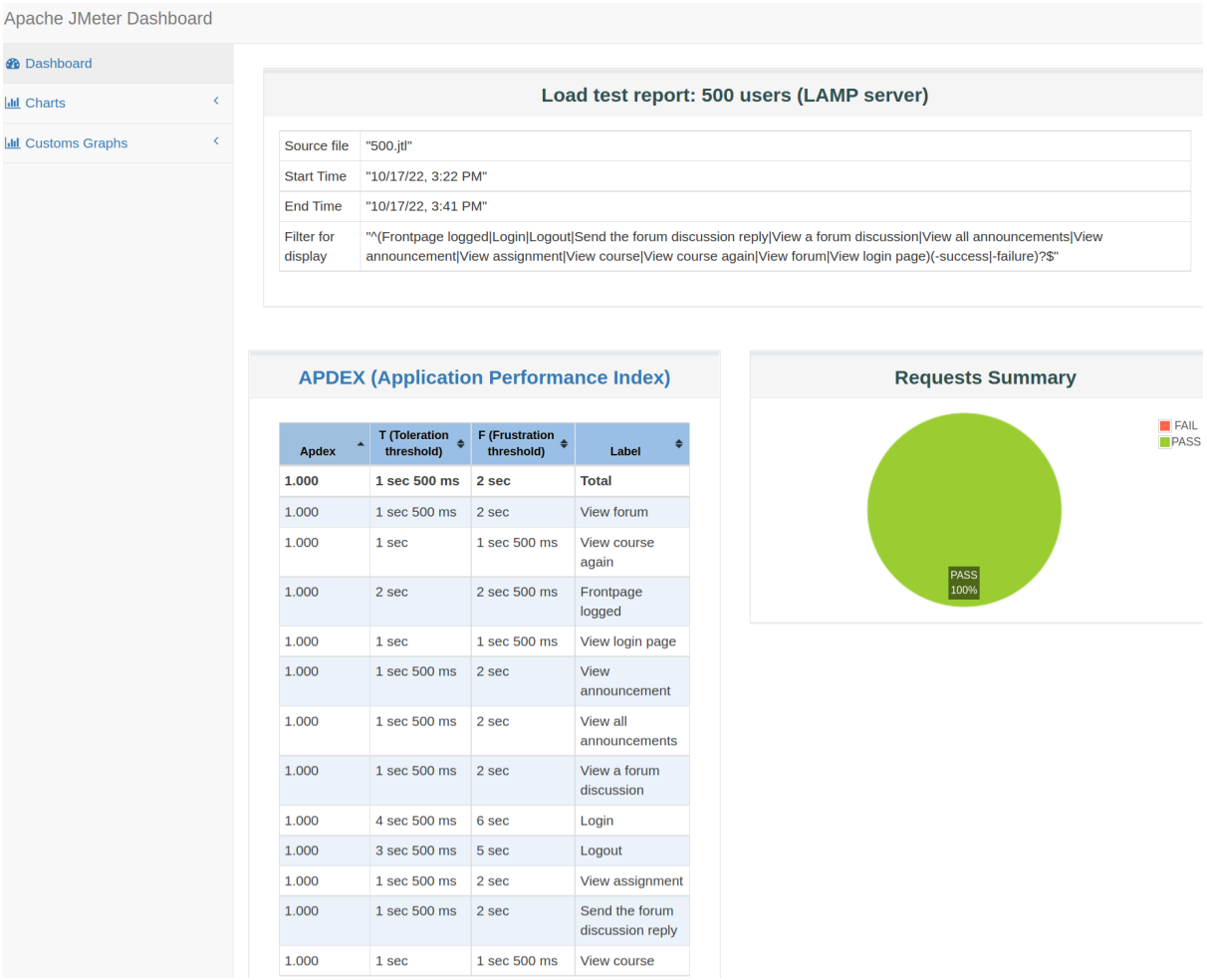
Για τη διεξαγωγή του συγκεκριμένου τεστ επιλέγουμε τις παρακάτω ρυθμίσεις στο jmx αρχείο:

- **ramp-up period = 130 δευτερόλεπτα** για τα Warm-up site και Moodle Test με 500 χρήστες και για το Forums (30 χρήστες) **10 δευτερόλεπτα**. Έτσι, δίνουμε χρόνο στο σύστημα για να μην υπερφορτωθεί κατά την εκκίνηση και παραγωγή των threads.

Θα πρέπει να τονιστεί ότι, τα Thread Groups Moodle Test και Forums εκκινούν ταυτόχρονα, αλλά επιλέγεται καθυστέρηση εκκίνησης **startup delay = 130 δευτερόλεπτα,** όση δηλαδή περίπου η διάρκεια εκτέλεσης του Warm-up site. Με αυτόν τον τρόπο, επιδιώκουμε να μην αυξηθεί το φορτίο στον εξυπηρετητή κατά την εκτέλεση του διαδικαστικού Thread Group Warm-up site.
Τέλος ορίζουμε και την επιθυμητή συνολική διάρκεια του τεστ στα **20 λεπτά** περίπου.

JMeter HTML report

Τα αποτελέσματα των τεστ, παρουσιάζονται στο HTML report, το οποίο έχει όνομα index.html και αποθηκεύτηκε στο φάκελο που επιλέχθηκε. Το HTML report δημιουργείται αυτόματα από το JMeter και συλλέγει τα αποτελέσματα όπως διακρίνεται στις παρακάτω εικόνες.



Εικόνα 2.7.1: Apache JMeter HTML report

Statistics

Requests	Executions			Response Times (ms)							Throughput	Network (KB/sec)	
Label	#Samples	FAIL	Error %	Average	Min	Max	Median	90th pct	95th pct	99th pct	Transactions/s	Received	Sent
Total	**67384**	**0**	**0.00%**	**148.66**	**5**	**3035**	**118.00**	**220.00**	**268.00**	**386.99**	**59.57**	**10127.60**	**13.84**
Login	1530	0	0.00%	510.55	303	3035	404.00	839.80	1099.70	1816.42	2.01	425.91	1.75
View assignment	7814	0	0.00%	215.93	120	866	188.00	322.50	392.00	529.85	8.14	1060.10	1.69
View forum	5139	0	0.00%	193.85	99	1013	159.00	306.00	400.00	636.80	5.24	1817.83	1.09
Logout	1192	0	0.00%	167.99	32	769	122.50	326.70	365.75	436.21	1.07	96.36	0.40
View all announcements	6000	0	0.00%	145.38	74	902	120.00	237.90	294.00	438.97	6.47	1726.78	1.34
Frontpage logged	1530	0	0.00%	145.31	75	765	114.00	252.70	330.00	534.94	2.01	421.80	0.74
View a forum discussion	5082	0	0.00%	139.74	70	763	115.00	223.00	287.85	437.51	5.30	1097.65	1.11
View announcement	6000	0	0.00%	122.49	60	714	101.00	197.00	248.00	369.99	6.46	883.08	1.35
View course	13647	0	0.00%	115.47	56	865	96.00	175.00	226.00	381.04	12.10	1976.33	2.46
View course again	8902	0	0.00%	111.41	55	752	91.00	180.00	230.00	351.97	9.54	1558.08	1.94
Send the forum discussion reply	184	0	0.00%	66.46	31	208	54.50	115.00	147.75	200.35	0.21	0.40	0.13
View login page	1530	0	0.00%	26.77	14	458	23.00	39.00	47.00	73.38	2.04	47.41	0.26

Errors

Type of error	Number of errors	% in errors	% in all samples

Top 5 Errors by sampler

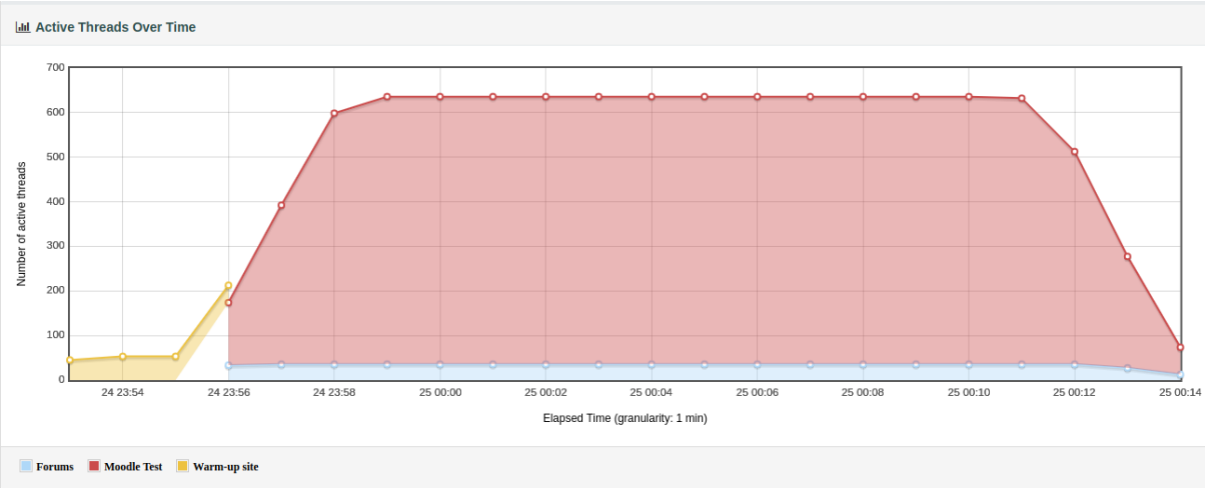
Sample	#Samples	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors	Error	#Errors
Total	67384	0										

Εικόνα 2.7.2: Apache JMeter HTML report

Στα παραπάνω στιγμιότυπα, απεικονίζεται το HTML report για το τεστ των 530 χρηστών. Για την επιλογή των περιεχομένων και των samplers που θα απεικονιστούν στο HTML report και για τους οποίους θα εμφανιστούν τα παραπάνω χαρακτηριστικά, είναι αναγκαία η επεξεργασία του αρχείου user.properties στον φάκελο /bin του JMeter.
Επομένως, εισάγουμε το παρακάτω φίλτρο για την απεικόνιση των επιθυμητών samplers στο παραπάνω αρχείο:

jmeter.reportgenerator.exporter.html.series_filter=^(Frontpage logged|Login|Logout|Send the forum discussion reply|View a forum discussion|View all announcements|View announcement|View assignment|View course|View course again|View forum|View login page)(-success|-failure)?$

Επιπλέον μία από τις πιο χρήσιμες πληροφορίες που προσφέρει το HTML report, είναι ο τρόπος εκτέλεσης των Thread Groups στη διάρκεια του χρόνου και μπορεί να βρεθεί στο πεδίο **Charts⟶Time⟶Active Threads Over Time**. Έτσι, παρακάτω παρουσιάζονται τα Thread Groups και ο τρόπος με τον οποίο εξελίσσονται στο χρόνο για το τεστ των 636 ταυτόχρονων χρηστών (36 χρήστες στο Forums).



Εικόνα 2.7.3: Active Threads Over Time

Από την παραπάνω εικόνα, είναι εμφανής και ο τρόπος λειτουργίας του τεστ. Όπως προαναφέρθηκε, δίνεται επαρκής χρόνος στο Warm-up site (κίτρινο χρώμα) και ύστερα εκκινούν τα Moodle Test (κόκκινο) και Forums (μπλέ) .

Application Performance Index (APDEX)

Το Application Performance Index (APDEX), πρόκειται για ένα ανοιχτό πρότυπο (open standard) που έχει αναπτυχθεί για τη μέτρηση της απόδοσης υπολογιστικών εφαρμογών. Ο κύριος στόχος του, είναι η μετατροπή των μετρήσεων σε μία ομοιόμορφη πληροφορία που αφορά την ικανοποίηση των χρηστών της εφαρμογής [25].
Η βαθμολογία υπολογίζεται με βάση το πλήθος των ευχαριστημένων (satisfied), ανεκτικών (tolerating) και απογοητευμένων (frustrated) χρηστών. Έτσι, ο μέγιστος χρόνος ενός ευχαριστημένου χρήστη είναι ίσος με “t”, ενός ανεκτικού χρήστη ίσος με “4t”, ενώ για χρόνους εξυπηρέτησης μεγαλύτερους από “4t”, ο χρήστης θεωρείται απογοητευμένος. Επιπλέον, η προκύπτουσα βαθμολογία αποτελεί ένα σταθμισμένο μέσο όρο με βάρη: 1, 0.5 και 0 αντίστοιχα.

Συνήθως, ο μέσος χρόνος απόκρισης μιας εφαρμογής αποκρύπτει πολύτιμη πληροφορία για την κατανομή των χρόνων απόκρισης, με αποτέλεσμα να μην είναι ξεκάθαρη η εμπειρία και η ικανοποίηση των χρηστών. Έτσι, το APDEX score μετατρέπει πολλές μετρήσεις σε έναν αριθμό από το 0 έως το 1 (0 = κανένας ευχαριστημένος, 1 = όλοι ευχαριστημένοι). Η σχέση που χρησιμοποιείται για τον υπολογισμό είναι η παρακάτω.

$$\text{Apdex}_t = \frac{SatisfiedCount + (ToleratingCount * 0.5) + (FrustratedCount * 0)}{TotalSamples}$$

Είναι προφανές, ότι κάθε εφαρμογή έχει άλλες απαιτήσεις σε χρόνους απόκρισης ανάλογα με τη σημαντικότητα του κάθε αιτήματος. Για το σύστημα του Moodle που μελετάμε, θέτουμε τους παρακάτω χρόνους με βάση τους οποίους θα υπολογιστούν και τα APDEX scores στα τεστ. Για να παραμετροποιήσουμε τους προεπιλεγμένος χρόνους του JMeter, τροποποιούμε το αρχείο **user.properties**, εισάγοντας τους παρακάτω επιθυμητούς χρόνους σε ms:

```
jmeter.reportgenerator.apdex_per_transaction=Login:4500|6000;\
    Logout:3500|5000;\
    Send the forum discussion reply:1500|2000;\
    View course again:1000|1500;\
    View course:1000|1500;\
    View login page:1000|1500;\
    Frontpage logged:2000|2500;\
    View assignment:1500|2000;\
    View forum:1500|2000;\
    View a forum discussion:1500|2000;\
    View all announcements:1500|2000;\
    View announcement:1500|2000
```

Με αυτόν τον τρόπο, έχουμε θέσει τους χρόνους για τα toleration threshold και frustration threshold που απεικονίζονται στην εικόνα 2.7.1. Έτσι, υπολογίζεται ξεχωριστά για κάθε HTTP αίτημα το APDEX score.

Τέλος, θα πρέπει να επισημανθεί, ότι μια εφαρμογή έχει εξαιρετική συμπεριφορά για APDEX score από 0.94 έως 1, πολύ καλή για 0.85 έως 0.93, καλή για 0.84 έως 0.70 και οτιδήποτε κάτω από 0.7 θεωρείται κακή επίδοση.

Κεφάλαιο 3: Βελτιστοποιήσεις συστήματος

3.1: Βελτιστοποίηση MySQL μέσω MySQLTuner

Ορισμένες βελτιστοποιήσεις του συστήματος είναι αναγκαίες πριν την εκτέλεση των τεστ. Πιο συγκεκριμένα, χρησιμοποιούμε το εργαλείο **MySQLTuner**, μέσω του οποίου προτείνονται αλλαγές που αφορούν την επίδοση της βάσης δεδομένων MySQL.

Για την εγκατάσταση εκτελούμε:

```
sudo apt-get install mysqltuner
```

και για να τρέξουμε το πρόγραμμα εκτελούμε την εντολή:

```
sudo mysqltuner
```

Στη συνέχεια, προστίθενται οι προτεινόμενες αλλαγές στο αρχείο παραμετροποίησης της MySQL, το οποίο είναι το /etc/mysql/mysql.conf.d/mysqld.cnf.

```
skip-name-resolve=1
innodb_buffer_pool_size=2G
innodb_log_file_size=512M
innodb_io_capacity_max=2500
innodb_io_capacity=500
innodb_buffer_pool_instances=2
key_buffer_size=0
```

3.2: Επιλογή κατάλληλου Apache Multi-Processing Module (MPM)

Μια εξαιρετικά κρίσιμη παράμετρος που πρέπει να ληφθεί υπόψη πριν από οποιαδήποτε δοκιμή φορτίου είναι η σωστή επιλογή του mpm module για τον Apache εξυπηρετητή. Το προεπιλεγμένο και χρησιμοποιούμενο mpm module μπορεί να βρεθεί με την εντολή:

```
sudo apachectl -M | grep 'mpm'
```

Στην περίπτωσή μας, το προεπιλεγμένο module είναι το mpm_prefork με τις παρακάτω ρυθμίσεις, οι οποίες μπορούν να βρεθούν στο αρχείο παραμετροποίησης **/etc/apache2/mods-available/ mpm_prefork.conf**, όπως διακρίνεται παρακάτω

```
<IfModule mpm_prefork_module>
    StartServers                5
    MinSpareServers         5
    MaxSpareServers         10
    MaxRequestWorkers       150
    MaxConnectionsPerChild  0
</IfModule>
```

Όπως διαφαίνεται παραπάνω, η τιμή MaxRequestWorkers είναι μικρή, καθώς ορίζει το μέγιστο πλήθος των ταυτόχρονων συνδέσεων που μπορεί να εξυπηρετήσει ο Apache ίσο με 150 [26]. Επιπλέον, συνίσταται η χρήση του event ή worker αντί του prefork, καθώς απαιτεί λιγότερους πόρους μνήμης.
Έτσι, επιλέχθηκε το mpm_worker module με τις παρακάτω ρυθμίσεις:

```
<IfModule mpm_worker_module>
ServerLimit 250
StartServers 100
MinSpareThreads 75
MaxSpareThreads 250
ThreadLimit 64
ThreadsPerChild 64
MaxRequestWorkers 8000
MaxConnectionsPerChild 10000
</IfModule>
```

Σύγκριση μεταξύ mpm_prefork και mpm_worker

Ο τρόπος λειτουργίας των δύο αυτών modules διαφέρει σημαντικά, με αποτέλεσμα να υπάρχουν έντονες διαφορές στην επίδοση του συστήματος ανάλογα με την εκάστοτε επιλογή.

Τρόπος λειτουργίας του mpm_prefork

Ο τρόπος διαχείρισης των αιτημάτων επιτυγχάνεται με τη δημιουργία διεργασιών (processes), όπου κάθε διεργασία αποτελείται από ένα και μόνο νήμα (thread), καθένα εκ των οποίων διαχειρίζεται τα αιτήματα. Όσο το πλήθος των αιτημάτων παραμένει μικρότερο από το πλήθος των διεργασιών, η επίδοση του συγκεκριμένου module είναι πολύ καλή. Ωστόσο δε συνίσταται η επιλογή του για συστήματα με πολύ υψηλό αριθμό χρηστών, καθώς κάθε διεργασία καταναλώνει αρκετούς πόρους μνήμης και επεξεργαστή. Επομένως δεν έχει ιδανική επίδοση σε συστήματα όπου επιθυμείται η κλιμάκωση τους [27].

Τρόπος λειτουργίας του mpm_worker

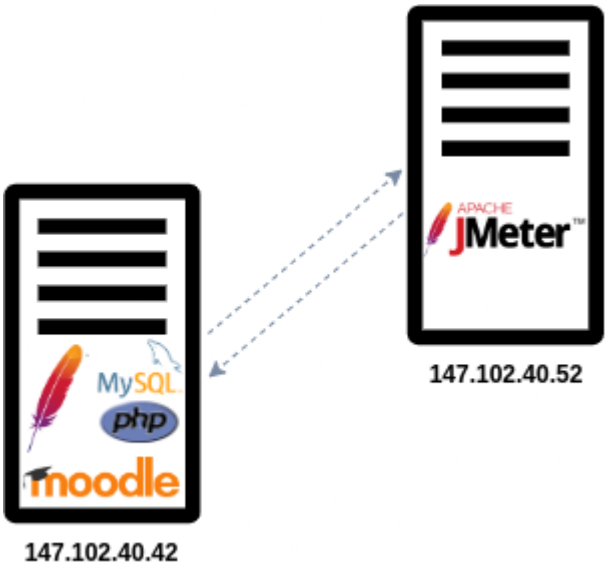
Η σημαντική διαφορά στον τρόπο λειτουργίας του συγκεκριμένου module έγκειται στο γεγονός ότι κάθε διεργασία μπορεί να διαχειρίζεται πολλαπλά νήματα, με κάθε νήμα να διαχειρίζεται έως και μία σύνδεση. Έτσι σε σύγκριση με το mpm_prefork το πλεονέκτημα αυτής της πολυνηματικής(multithreaded) υλοποίησης είναι εμφανές, καθώς τα νήματα απαιτούν πολύ λιγότερους πόρους επεξεργαστή και μνήμης. Ακόμα με αυτόν τον τρόπο το πλήθος των νημάτων είναι πολύ μεγαλύτερο από εκείνο των διεργασιών, επομένως κάθε νέα εισερχόμενη σύνδεση μπορεί να εκμεταλλευτεί αμέσως ένα απελευθερωμένο νήμα, χωρίς να χρειάζεται να αναμένει την απελευθέρωση μιας διεργασίας όπως στην περίπτωση του prefork.

Κεφάλαιο 4: Περιγραφή αρχιτεκτονικών για τη δοκιμή φορτίου

Η δοκιμή φορτίου εκτελέστηκε σε διάφορες αρχιτεκτονικές, με κύριο σκοπό τη σύγκριση μεταξύ τους, καθώς και την εύρεση της βέλτιστης αρχιτεκτονικής. Στη συνέχεια, παρουσιάζονται οι υλοποιημένες αρχιτεκτονικές και έπειτα παρουσιάζονται και αναλύονται όλα τα αποτελέσματα μεταξύ τους.

4.1: Moodle ως αυτοδύναμος εξυπηρετητής

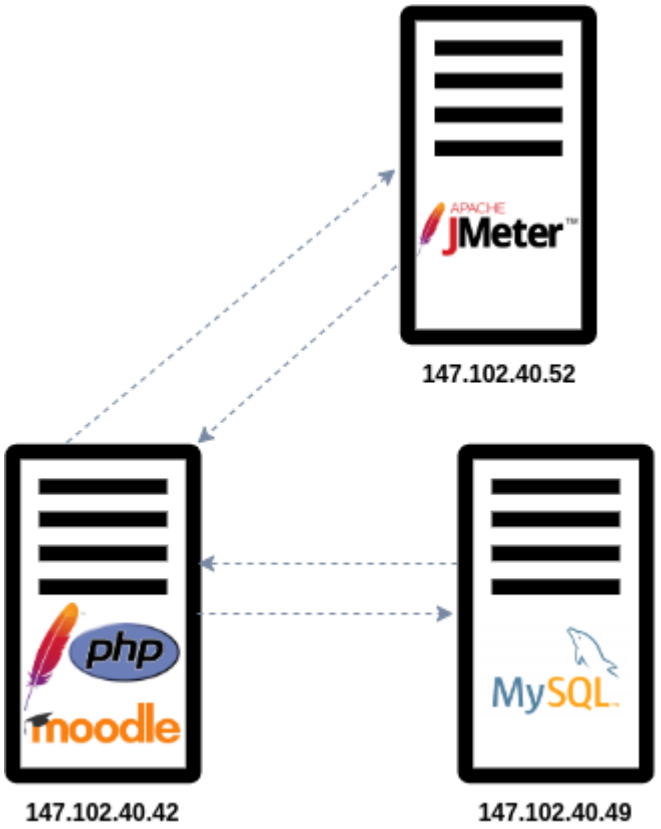
Όπως περιγράφηκε και στην εισαγωγή, στην παρούσα διπλωματική εργασία διεξάγεται η δοκιμή φορτίου στο Moodle, το οποίο υλοποιείται σε διάφορες αρχιτεκτονικές που θα παρουσιαστούν στη συνέχεια. Η αρχική υλοποίηση είναι με το Moodle εγκατεστημένο σε ένα μόνο εξυπηρετητή, μαζί με τη βάση δεδομένων.



Εικόνα 4.1: Moodle ως αυτοδύναμος εξυπηρετητής

Έτσι, το Linux μηχάνημα διαθέτει συγκεντρωμένες όλες τις υπηρεσίες όπως φαίνεται στο παραπάνω σχήμα. Αρχικά διαθέτει 6 πυρήνες επεξεργαστή και 16 GB αποθηκευτικού χώρου στη μνήμη RAM.
Στη συνέχεια, χρησιμοποιούμε την ίδια αρχιτεκτονική για το σύστημά μας με τη διαφορά ότι αυξάνουμε τους πυρήνες του επεξεργαστή προκειμένου να διακρίνουμε τυχόν βελτιώσεις.
Για τη χρήση του Apache JMeter, χρησιμοποιείται ένα άλλο Linux μηχάνημα, όπως φαίνεται παραπάνω.

4.2: Διαχωρισμός της βάσης δεδομένων από το Moodle και δημιουργία αυτοδύναμου MySQL εξυπηρετητή



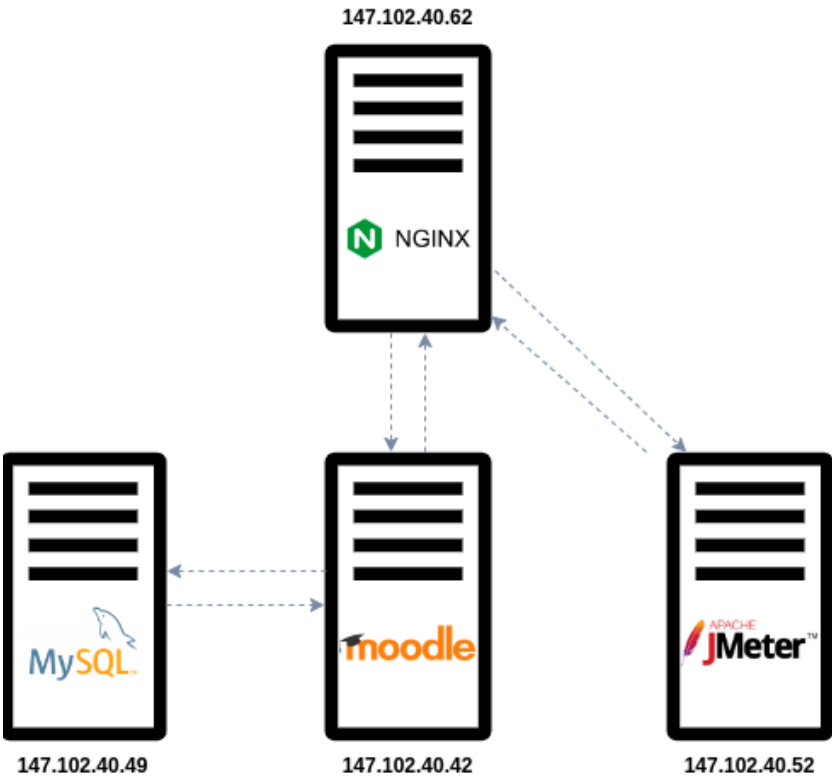
Εικόνα 4.2.1: Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή

Η δεύτερη αρχιτεκτονική που μελετάται είναι η επίδοση του συστήματος, αφού διαχωριστεί η βάση δεδομένων από το Moodle που τρέχει τον κώδικα PHP. Για το σκοπό αυτό, χρησιμοποιείται ένα ξεχωριστό μηχάνημα (ο εξυπηρετητής 147.102.40.49), όμοιο με εκείνο του Moodle, το οποίο θα έχει ως σκοπό να φιλοξενεί αποκλειστικά τη βάση δεδομένων MySQL του συστήματος.
Έτσι, η εγκατάσταση αυτής της αρχιτεκτονικής παρουσιάζεται στο κεφάλαιο 8.

Nginx reverse proxy και μετάβαση σε HTTPS

Εικόνα 4.2.2: Λογότυπο Nginx

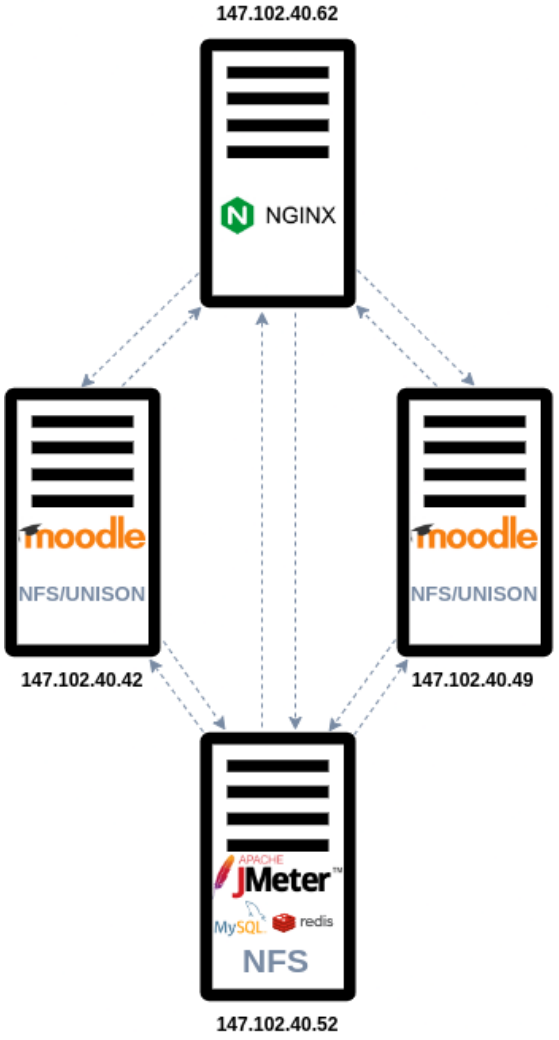
Η επόμενη προσθήκη στην αρχιτεκτονική, είναι η προσθήκη ενός reverse proxy. Υπάρχουν αρκετές επιλογές για τη χρήση ενός reverse proxy με τα πιο γνωστά λογισμικά να αποτελούν το Nginx, το HAProxy, το Traefik και το Squid. Η επιλογή που πραγματοποιείται είναι εκείνη της χρήσης του Nginx.
Με τον όρο reverse proxy, εννοούμε τον εξυπηρετητή, ο οποίος βρίσκεται "μπροστά" από back-end εφαρμογές και είναι εκείνος που αναλαμβάνει να προωθεί τα αιτήματα των χρηστών στις back-end εφαρμογές, εν προκειμένω στο Moodle εξυπηρετητή [28].
Οι χρήστες της εφαρμογής δεν γνωρίζουν την ύπαρξη του reverse proxy, καθώς οι απαντήσεις που λαμβάνουν φαίνονται σαν να προήλθαν απευθείας από τις back-end εφαρμογές.



Εικόνα 4.2.3: Λειτουργία του Nginx ως reverse proxy

Η εγκατάσταση του Nginx, του certbot και η ρύθμιση του HTTPS παρουσιάζονται στο κεφάλαιο 8.

4.3: Συστάδα Moodle (Moodle cluster)



Εικόνα 4.3.1: Nginx ως load balancer

Η τελευταία αρχιτεκτονική που δημιουργήθηκε, αποτελεί μέρος της κλιμάκωσης του συστήματος με την μετάβαση στη δομή μιας συστάδας αποτελούμενη από δύο Moodle εξυπηρετητές. Η αρχιτεκτονική αυτή, είναι η βάση όλων των σύγχρονων αρχιτεκτονικών μεγάλης κλίμακας, αφού εκμεταλλεύεται παράλληλα την υπολογιστική ισχύ περισσότερων υπολογιστών, προσφέροντας έτσι δυνατότητες επέκτασης του συστήματος.
Επιπλέον, τα οφέλη χρήσης ενός τέτοιου συστήματος πέρα από την αύξηση της επίδοσης, είναι η αύξηση της διαθεσιμότητας, καθώς σε περίπτωση αποτυχίας ενός εξυπηρετητή υπάρχει εναλλακτική εξυπηρέτηση από τους υπόλοιπους εξυπηρετητές.

Στην υλοποίηση που ακολουθείται γίνεται χρήση 2 πανομοιότυπων Moodle εξυπηρετητών, οι οποίοι τρέχουν τον ίδιο κώδικα PHP για το Moodle, ενώ χρησιμοποιούν και οι δύο μια βάση δεδομένων σε τρίτο μηχάνημα. Για τη σωστή λειτουργία του συστήματος, είναι απαραίτητη και η ύπαρξη ενός κοινόχρηστου φακέλου moodledata, ο οποίος είναι ο φάκελος στον οποίο αποθηκεύονται τα δεδομένα της εφαρμογής. Για το σκοπό αυτό, χρησιμοποιούνται αλλά και συγκρίνονται μεταξύ τους δύο πρωτόκολλα που επιτυγχάνουν τον συγχρονισμό των δίσκων και αυτά είναι το NFS και το Unison.

Επομένως, το συνολικό σύστημα αποτελείται από 2 Moodle εξυπηρετητές, έναν τρίτο που υλοποιεί το JMeter, τη βάση δεδομένων, το NFS και το Redis και ο Nginx που λειτουργεί πλέον και ως load balancer.

Στη συγκεκριμένη αρχιτεκτονική, είναι πολύτιμη και η ύπαρξη κρυφής μνήμης Redis, καθώς αυξάνει σημαντικά τις επιδόσεις του συστήματος.

Υλοποίηση με κρυφή μνήμη Redis

Εικόνα 4.3.2: Λογότυπο Redis

Για την αρχιτεκτονική της συστάδας των δύο Moodle εξυπηρετητών είναι απαραίτητη η χρήση μιας ενδιάμεσης κρυφής μνήμης για τις συνόδους του Moodle. Υπάρχουν αρκετές επιλογές για το σκοπό αυτό, με τα κυριότερα λογισμικά να είναι τα Memcached, MongoDB cache, APC user cache (APCu) και Redis [29]. Στην υλοποίηση της συστάδας του Moodle επιλέγεται η χρήση του Redis ως κρυφή μνήμη.

Το Redis (Remote Dictionary Server) χρησιμοποιείται στην αρχιτεκτονική της συστάδας των Moodle ως κρυφή μνήμη για κάποια δεδομένα της εφαρμογής αλλά και για τις συνόδους. Η χρήση της κρυφής μνήμης Redis είναι αναγκαία για την καλή επίδοση της συστάδας των Moodle εξυπηρετητών και όπως θα παρουσιαστεί στη συνέχεια στα αποτελέσματα των τεστ είναι προφανής η διαφορά στην αύξηση της επίδοσης με τη χρήση της.
Η εγκατάσταση και ρύθμιση του Redis παρουσιάζεται στο Κεφάλαιο 8.

Χρήση NFS (Network File System)

Για την επιτυχή λειτουργία της συστάδας των Moodle εξυπηρετητών είναι απαραίτητη η χρήση ενός πρωτοκόλλου μέσω του οποίου θα μπορεί ο φάκελος moodledata να είναι κοινόχρηστος. Αρχικά, παρουσιάζεται η επίδοση του NFS και στη συνέχεια του πρωτοκόλλου Unison.

Το NFS είναι ένα κατανεμημένο σύστημα αρχείων, που επιτρέπει σε ένα χρήστη σε ένα δίκτυο, την πρόσβαση σε αρχεία που βρίσκονται σε έναν άλλο υπολογιστή. Με αυτόν τον τρόπο, είναι προσβάσιμα αυτά τα αρχεία, όπως ακριβώς θα ήταν και στην περίπτωση που ήταν τοπικά. Για την εγκατάστασή του, υπάρχει ο NFS host και οι NFS clients.

Η εγκατάσταση και ρύθμισή του παρουσιάζεται στο Κεφάλαιο 8.

Unison

Ένα εναλλακτικό λογισμικό για το συγχρονισμό αρχείων είναι το Unison. Το Unison διαφέρει από το NFS, ως προς τον τρόπο λειτουργίας του. Χρησιμοποιήθηκε, προκειμένου να επιτραπεί σε δύο αντίγραφα του moodledata να αποθηκευτούν σε διαφορετικούς υπολογιστές (Moodle εξυπηρετητές), με αποτέλεσμα να τροποποιούνται ξεχωριστά και να ανακοινώνονται μόνο οι αλλαγές που πραγματοποιήθηκαν.

Το αποτέλεσμα είναι ο moodledata να είναι και σε αυτήν την περίπτωση κοινόχρηστος. Η εγκατάσταση και ρύθμισή του παρουσιάζεται στο Κεφάλαιο 8.

Κεφάλαιο 5: Σύγκριση αποτελεσμάτων δοκιμής φορτίου

Στο παρόν κεφάλαιο, θα παρουσιαστούν και θα συγκριθούν οι επιδόσεις των αρχιτεκτονικών που περιγράφηκαν παραπάνω. Συνοπτικά, επαναλαμβάνονται οι αρχιτεκτονικές στις οποίες διενεργήθηκαν τα τεστ:

- Moodle ως αυτοδύναμος εξυπηρετητής με 16GB αποθηκευτικού χώρου στη μνήμη RAM , καθώς και με 6 και 10 πυρήνες επεξεργαστή.

- Moodle (6 και 10 πυρήνες επεξεργαστή) ως αυτοδύναμος εξυπηρετητής με Nginx reverse proxy.

- Moodle (6 και 10 πυρήνες επεξεργαστή) με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (6 και 10 πυρήνες επεξεργαστή αντίστοιχα).

- Moodle (6 και 10 πυρήνες επεξεργαστή) με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (6 και 10 πυρήνες επεξεργαστή αντίστοιχα) και χρήση Nginx ως reverse proxy.

- Συστάδα Moodle (6 και 10 πυρήνες επεξεργαστή) με MySQL, κρυφή μνήμη Redis και NFS σε ξεχωριστό σύστημα (με 12 πυρήνες επεξεργαστή και 16GB αποθηκευτικού χώρου στη μνήμη RAM) και χρήση Nginx ως load balancer.

- Συστάδα Moodle με NFS χωρίς κρυφή μνήμη Redis και Nginx ως load balancer.

- Συστάδα Moodle (6 πυρήνες επεξεργαστή) με Unison, κρυφή μνήμη Redis και Nginx ως load balancer.

Σε όλες τις αρχιτεκτονικές, ο Nginx είχε σταθερά 4 πυρήνες επεξεργαστή και 8GB αποθηκευτικού χώρου στη μνήμη RAM.

Κριτήρια σύγκρισης επίδοσης μεταξύ αρχιτεκτονικών

Οι συγκρίσεις μεταξύ των αρχιτεκτονικών που υλοποιούνται, γίνονται λαμβάνοντας υπόψη τα ακόλουθα κριτήρια:

- Το μέσο (average) χρόνο απόκρισης στα αιτήματα των χρηστών.

- Το μέγιστο πλήθος των ταυτόχρονων χρηστών (concurrent users).

- Το συνολικό APDEX score καθώς και των επιμέρους HTTP αιτημάτων.

Ένα ακόμη κριτήριο που λαμβάνεται υπόψη σε τεστ δοκιμής φορτίου είναι και ο συνολικός ρυθμός απόδοσης (throughput) του συστήματος. Ωστόσο, ο συνολικός ρυθμός απόδοσης στα εκτελούμενα τεστ δεν αποτελεί χρήσιμη πληροφορία συγκριτικά με τους χρόνους απόκρισης ή το μέγιστο πλήθος των ταυτόχρονων χρηστών.
Αυτό, διότι σε αυτή τη περίπτωση υπολογίζεται σε αιτήματα (transactions) ανά δευτερόλεπτο, ωστόσο δεν έχουν όλα τα αιτήματα την ίδια βαρύτητα, καθώς άλλα απαιτούν μεγαλύτερους χρόνους απόκρισης και άλλα λιγότερους.
Επομένως, με τον τρόπο με τον οποίο έχουν δημιουργηθεί τα τεστ, είναι αρκετές και ενδεικτικές οι τρεις παραπάνω παράμετροι για το σύστημά μας.

Η βέλτιστη απόδοση του συστήματος επιτυγχάνεται με την αύξηση των ταυτόχρονων χρηστών που μπορεί να εξυπηρετήσει το σύστημα με το μικρότερο δυνατό μέσο χρόνο απόκρισης. Το χειρότερο APDEX score, το οποίο αντιστοιχεί στο μέγιστο πλήθος ταυτόχρονων χρηστών και το οποίο μπορούμε να ανεχτούμε είναι ίσο με 0.7 έως 0.75. Ωστόσο, για τη βέλτιστη απόδοση επιθυμούμε ένα APDEX score στο εύρος 0.85 με 1.

5.1: Σύγκριση αυτοδύναμου Moodle εξυπηρετητή-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (6 πυρήνες επεξεργαστή)

Αρχικά, συγκρίνεται η επίδοση του Moodle ως αυτοδύναμος εξυπηρετητής με το Moodle, όπου η βάση δεδομένων είναι σε ξεχωριστό σύστημα. Όλα τα μηχανήματα έχουν 6 πυρήνες επεξεργαστή και 16 GB αποθηκευτικού χώρου στη μνήμη RAM. Οι μέσοι χρόνοι απόκρισης που λαμβάνονται υπόψη είναι εκείνοι των HTTP αιτημάτων με το μεγαλύτερο χρόνο. Αυτοί, αφορούν στο **Login**, και στο **View forum**. Έτσι, παρακάτω παρουσιάζονται οι συγκρίσεις με τους χρόνους απόκρισης σε **milliseconds**, καθώς και το APDEX score.

Πλήθος ταυτόχρονων χρηστών	**Αυτοδύναμο Moodle**		**APDEX score**	**Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή**		**APDEX score**
	Login	View forum		Login	View forum	
530	510.55	193.85	1.0	410.85	173.28	1.0
636	1886.1	616.55	0.96	491.63	209.14	1.0
848	5707.73	3394.76	0.3	2227.59	751.25	0.938
1060	-	-	-	5163.33	2685.48	0.37

Πίνακας 5.1.1: Σύγκριση αυτοδύναμου Moodle εξυπηρετητή-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (6 πυρήνες επεξεργαστή)

Στον επόμενο πίνακα παρουσιάζονται τα ίδια αποτελέσματα, κατόπιν προσθήκης του Nginx ως reverse proxy.

Πλήθος ταυτόχρονων χρηστών	**Αυτοδύναμο Moodle + reverse proxy**		**APDEX score**	**Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή + reverse proxy**		**APDEX score**
	Login	View forum		Login	View forum	
530	568.73	217.12	1.0	424.20	175.92	1.0
636	1958.94	622.14	0.96	568.70	247.03	1.0
848	5808.82	3414.87	0.29	2407.40	836.37	0.92
1060	-	-	-	5640.35	3132.95	0.35

Πίνακας 5.1.2: Σύγκριση αυτοδύναμου Moodle εξυπηρετητή + reverse proxy-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή + reverse proxy (6 πυρήνες επεξεργαστή)

Σχολιασμός αποτελεσμάτων

Από τα παραπάνω αποτελέσματα, αναδεικνύεται το πλεονέκτημα διαχωρισμού της βάσης δεδομένων. Διαφαίνεται ότι με αυτόν τον τρόπο μειώνεται ο συνολικός φόρτος εργασίας, καθώς απελευθερώνονται πόροι επεξεργαστή και μνήμης από το Moodle εξυπηρετητή. Έτσι, το μέγιστο επιτρεπόμενο πλήθος ταυτόχρονων χρηστών είναι περίπου **650-700** χρήστες στην περίπτωση λειτουργίας του Moodle ως **αυτοδύναμου εξυπηρετητή** και ίσο με **850-900** χρήστες **στο σύστημα με διαχωρισμένη τη βάση δεδομένων**. Με άλλα λόγια, ο διαχωρισμός της βάσης αυξάνει το συνολικό πλήθος ταυτόχρονων χρηστών κατά 200.

Η απότομη χειροτέρευση στους χρόνους απόκρισης (από τους **636** στους **848** και από τους **848** στους **1060** αντίστοιχα) οφείλεται στο γεγονός ότι η χρήση του επεξεργαστή εκεί είναι η μέγιστη.

Αναφορικά με τη χρήση του Nginx ως reverse proxy, παρατηρείται μια ελάχιστη καθυστέρηση στους χρόνους απόκρισης. Αυτό οφείλεται στο γεγονός ότι ο Nginx αποτελεί έναν επιπλέον κόμβο στη συνολική διαδρομή, που θα πρέπει να ακολουθήσουν τα αιτήματα των χρηστών. Ακόμη, επειδή ο Nginx αναλαμβάνει τη διαδικασία του SSL τερματισμού, λόγω της μετατροπής από HTTP σε HTTPS, προστίθεται μια μικρή καθυστέρηση στο συνολικό χρόνο απόκρισης του συστήματος.

5.2: Σύγκριση αυτοδύναμου Moodle εξυπηρετητή-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (10 πυρήνες επεξεργαστή)

Σε αυτό το κεφάλαιο, πραγματοποιείται η σύγκριση που έγινε προηγουμένως με τη διαφορά ότι αυξάνονται οι πυρήνες του επεξεργαστή σε 10, τόσο του Moodle όσο και του μηχανήματος που φιλοξενεί τη βάση δεδομένων. Έτσι τα αποτελέσματα της δοκιμής φορτίου είναι τα παρακάτω.

Πλήθος ταυτόχρονων χρηστών	**Αυτοδύναμο Moodle**		**APDEX score**	**Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή**		**APDEX score**
	Login	View forum		Login	View forum	
848	442.70	170.00	1.0	428.36	191.96	1.0
1060	1067.50	385.22	0.99	563.47	234.86	1.0
1166	2041.27	837.95	0.92	572.06	247.08	1.0
1272	3064.01	1557.58	0.63	1071.77	433.42	0.99
1484	-	-	-	2677.64	1238.25	0.74

Πίνακας 5.2.1: Σύγκριση αυτοδύναμου Moodle εξυπηρετητή-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (10 πυρήνες επεξεργαστή)

Πλήθος ταυτόχρονων χρηστών	**Αυτοδύναμο Moodle + reverse proxy**		**APDEX score**	**Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή + reverse proxy**		**APDEX score**
	Login	View forum		Login	View forum	
848	560.91	205.12	0.99	458.42	217.90	1.0
1060	1219.93	429.94	0.98	602.53	245.87	0.99
1166	2209.93	944.02	0.89	873.80	341.34	0.98
1272	3155.85	1596.83	0.62	1454.31	563.86	0.9
1484	-	-	-	2713.00	1249.36	0.73

Πίνακας 5.2.2: Σύγκριση αυτοδύναμου Moodle + reverse proxy-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή + reverse proxy (10 πυρήνες επεξεργαστή)

Και σε αυτήν την περίπτωση παρατηρείται η ελάχιστη καθυστέρηση που εισάγει ο reverse proxy, για τους λόγους που αναφέρθηκαν προηγουμένως. Επίσης για τους ίδιους λόγους με πριν, είναι εμφανής η βελτίωση με το διαχωρισμό της βάσης δεδομένων. Έτσι, το μέγιστο επιτρεπόμενο πλήθος ταυτόχρονων χρηστών κυμαίνεται στο εύρος **1150-1200** χρήστες στην

περίπτωση λειτουργίας του Moodle ως αυτοδύναμου εξυπηρετητή και στο εύρος **1500-1550** χρήστες στο σύστημα με διαχωρισμένη τη βάση δεδομένων. Αν συγκριθεί το σύστημα των 10 πυρήνων με εκείνο των 6 πυρήνων επεξεργαστή, παρατηρείται μία συνολική αύξηση **500-600** ταυτόχρονων χρηστών.

5.3: Σύγκριση NFS με κρυφή μνήμη Redis-NFS χωρίς Redis στη συστάδα των Moodle εξυπηρετητών

Η επόμενη σύγκριση γίνεται για την ανάδειξη της χρησιμότητας της κρυφής μνήμης Redis. Έτσι, συγκρίνεται το σύστημα της συστάδας των Moodle στο οποίο οι Moodle εξυπηρετητές έχουν 6 πυρήνες επεξεργαστή ο καθένας. Η βάση δεδομένων και η κρυφή μνήμη Redis βρίσκονται σε μηχάνημα με 12 πυρήνες επεξεργαστή και 16GB μνήμη.

Πλήθος ταυτόχρονων χρηστών	**Συστάδα Moodle - NFS χωρίς Redis**		**APDEX score**	**Συστάδα Moodle - NFS με Redis**		**APDEX score**
	Login	View forum		Login	View forum	
530	3191.73	434.80	0.85	529.21	296.43	1.0
848	8168.72	970.34	0.5	603.31	360.49	1.0
1060	15469.18	2891.38	0.001	737.80	422.16	0.99
1590	40858.12	10268.22	0	2320.94	1190.29	0.83

Πίνακας 5.3: Σύγκριση NFS με κρυφή μνήμη Redis-NFS χωρίς Redis στη συστάδα των Moodle εξυπηρετητών

Από τον παραπάνω πίνακα, είναι εμφανείς οι απαγορευτικοί χρόνοι του συστήματος όταν δε χρησιμοποιείται η κρυφή μνήμη Redis. Το Redis, έχει επιλεχθεί για τη λειτουργία ως κρυφή μνήμη για την εφαρμογή και για τις συνόδους. Η χρήση της κρυφής μνήμης Redis λύνει το πρόβλημα που παρουσιάζει ο κοινόχρηστος δίσκος, το οποίο είναι οι μεγάλοι χρόνοι ανάγνωσης και εγγραφής σε αυτόν, καθώς ο δίσκος είναι κοινόχρηστος σε τρεις εξυπηρετητές (NFS host και Moodle-NFS clients).

5.4: Σύγκριση μεταξύ NFS και Unison στη συστάδα των Moodle εξυπηρετητών

Για τη λειτουργία της συστάδας των Moodle εξυπηρετητών, είναι αναγκαία η ύπαρξη του κοινόχρηστου φακέλου moodledata. Έτσι, είναι απαραίτητη και η σύγκριση του NFS και του Unison, προκειμένου να διαπιστωθεί ποιο εκ των δύο προσφέρει καλύτερη επίδοση στο σύστημα.
Επομένως, παρακάτω παρουσιάζονται τα αποτελέσματα της δοκιμής φορτίου στην αρχιτεκτονική της συστάδας των Moodle εξυπηρετητών (6 πυρήνες επεξεργαστή), με τη χρήση του NFS και του Unison αντίστοιχα.

Πλήθος ταυτόχρονων χρηστών	**Συστάδα Moodle - NFS και Redis**		**APDEX score**	**Συστάδα Moodle - Unison και Redis**		**APDEX score**
	Login	View forum		Login	View forum	
530	529.21	296.43	1.0	392.09	164.54	1.0
848	603.31	360.49	1.0	431.83	200.49	1.0
1060	737.80	422.16	0.99	492.66	228.32	1.0
1166	749.43	448.54	0.99	569.83	271.38	1.0
1272	1287.76	727.57	0.97	779.74	364.99	0.99
1378	1546.74	826.80	0.94	1710.07	830.96	0.9
1484	1706.46	957.32	0.85	1957.22	989.09	0.88
1590	2320.94	1190.29	0.83	2253.84	1107.62	0.86
1802	4466.74	2344.34	0.51	4251.11	2256.72	0.56

Πίνακας 5.4: Σύγκριση μεταξύ NFS και Unison στη συστάδα των Moodle εξυπηρετητών (6 πυρήνες επεξεργαστή)

Από τα παραπάνω εξάγεται το συμπέρασμα ότι το Unison έχει ελαφρώς καλύτερες επιδόσεις από το NFS.

5.5: Σύγκριση συστάδας Moodle εξυπηρετητών με το αυτοδύναμο Moodle

Η τελική και πιο σημαντική σύγκριση αποτελεί εκείνη μεταξύ της συστάδας των Moodle εξυπηρετητών και του αρχικού Moodle, όπου η βάση δεδομένων είναι σε αυτοδύναμο MySQL εξυπηρετητή και με τη χρήση reverse proxy. Παρακάτω γίνεται η σύγκριση όπου όλοι οι Moodle εξυπηρετητές έχουν 6 πυρήνες επεξεργαστή.

Πλήθος ταυτόχρονων χρηστών	**Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή + reverse proxy**		**APDEX score**	**Συστάδα Moodle - Unison και Redis**		**APDEX score**
	Login	View forum		Login	View forum	
530	424.20	175.92	1.0	392.09	164.54	1.0
636	568.70	247.03	1.0	-	-	-
848	2407.40	836.37	0.92	431.83	200.49	1.0
1060	5640.35	3132.95	0.35	492.66	228.32	1.0
1166	-	-	-	569.83	271.38	1.0
1272	-	-	-	779.74	364.99	0.99
1378	-	-	-	1710.07	830.96	0.9
1484	-	-	-	1957.22	989.09	0.88
1590	-	-	-	2253.84	1107.62	0.86
1802	-	-	-	4251.11	2256.72	0.56

Πίνακας 5.5.1: Σύγκριση συστάδας των Moodle εξυπηρετητών-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (6 πυρήνες επεξεργαστή)

Ομοίως, πραγματοποιείται η σύγκριση όπου όλοι οι Moodle εξυπηρετητές έχουν 10 πυρήνες επεξεργαστή, συγκρίνοντας τη συστάδα των Moodle εξυπηρετητών με τη χρήση NFS, με το αυτοδύναμο σύστημα.

Πλήθος ταυτόχρονων χρηστών	Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή+ reverse proxy		APDEX score	Συστάδα Moodle - NFS και Redis		APDEX score
	Login	View forum		Login	View forum	
848	458.42	217.90	1.0	468.49	275.48	1.0
1060	602.53	245.87	0.99	527.01	310.38	1.0
1166	873.80	341.34	0.98	-	-	-
1272	1454.31	563.86	0.9	-	-	-
1484	2713.00	1249.36	0.73	-	-	-
1696	-	-	-	1012.04	688.03	0.98
1802	-	-	-	1291.94	858.14	0.88
1911	-	-	-	2219.05	1494.81	0.6
2120	-	-	-	3287.14	2080.27	0.47

Πίνακας 5.5.2: Σύγκριση συστάδας των Moodle εξυπηρετητών-Moodle με τη βάση δεδομένων σε αυτοδύναμο MySQL εξυπηρετητή (10 πυρήνες επεξεργαστή)

Στους παραπάνω πίνακες, όπου δεν υπάρχουν χρόνοι απόκρισης αλλά πάυλες (-), είναι διότι οι χρόνοι είναι πολύ μεγάλοι (αυτοδύναμο Moodle) ή μικροί (συστάδα Moodle) και δεν υπήρχε ανάγκη εκτέλεσης των τεστ.

5.6: Συμπεράσματα σύγκρισης συστάδας Moodle με το αρχικό Moodle

Για την αύξηση των χρηστών και την κλιμάκωση του συστήματος είναι αναγκαία η μετάβαση στην αρχιτεκτονική της συστάδας των Moodle εξυπηρετητών. Ωστόσο, ενώ η υπολογιστική ισχύς διπλασιάζεται στην αρχιτεκτονική αυτή, δεν ισχύει το ίδιο και για τους μέγιστους χρήστες που μπορεί να αντέξει το σύστημα.
Πιο συγκεκριμένα αν παρατηρήσει κανείς τη σύγκριση του αρχικού συστήματος με εκείνο της συστάδας των Moodle εξυπηρετητών, θα ήταν αναμενόμενο οι χρόνοι απόκρισης της συστάδας των Moodle να είναι παρόμοιοι με εκείνοι του αρχικού συστήματος στους μισούς χρήστες από ότι της συστάδας. Το γεγονός αυτό, πράγματι συμβαίνει για τους 530 (αρχικό σύστημα) και τους 1060 (συστάδα) χρήστες, καθώς οι χρόνοι απόκρισης είναι πολύ κοντινοί μεταξύ τους. Παρόλα αυτά, όσο αυξάνεται το πλήθος των ταυτόχρονων χρηστών αυτό παύει να ισχύει.

Έτσι συμπεραίνεται λοιπόν ότι στην αρχιτεκτονική της συστάδας των Moodle εξυπηρετητών η στένωση του συστήματος είναι ο χρόνος εγγραφής στον κοινόχρηστο δίσκο. Αυτό, είναι και αναμενόμενο, στη συγκεκριμένη αρχιτεκτονική, καθώς το Moodle χρησιμοποιεί τον

κοινόχρηστο φάκελο moodledata για λειτουργίες απόκρυψης δεδομένων της εφαρμογής (caching).

Συγκεκριμένα, όσο αυξάνεται το πλήθος των χρηστών, αυξάνεται η συχνότητα εγγραφής και ανάγνωσης των δεδομένων αυτών στην κρυφή μνήμη του κοινόχρηστου δίσκου.
Επομένως οι επιδόσεις σε αυτήν την αρχιτεκτονική είναι χειρότερες, χωρίς να αποδίδεται αυτό στην επίδοση και ταχύτητα των NFS και Unison, αλλά στη χρονοβόρα διαδικασία εγγραφής και ανάγνωσης από τον κοινόχρηστο φάκελο moodledata.

Παρόλα αυτά, είναι αναπόφευκτη η μετάβαση στην αρχιτεκτονική της συστάδας των Moodle εξυπηρετητών αν το σύστημα έχει απαιτήσεις για υψηλό αριθμό χρηστών.
Επιπλέον, η αρχιτεκτονική της συστάδας προσφέρει υψηλή διαθεσιμότητα, καθώς σε περίπτωση αδυναμίας ενός εκ των δύο moodle εξυπηρετητών το σύστημα παραμένει λειτουργικό.

5.7: Τελικά συμπεράσματα για το πλήθος των χρηστών

Συμπερασματικά, συγκεντρώνοντας τα αποτελέσματα προκειμένου να διαπιστωθεί το μέγιστο πλήθος των χρηστών σε κάθε αρχιτεκτονική ισχύουν τα εξής:

- **650-700** χρήστες στην περίπτωση λειτουργίας του Moodle ως αυτοδύναμου εξυπηρετητή και **850-900** με τη βάση δεδομένων ξεχωριστά (6 πυρήνες επεξεργαστή).

- **1150-1200** χρήστες στην περίπτωση λειτουργίας του Moodle ως αυτοδύναμου εξυπηρετητή και **1500-1550** με τη βάση δεδομένων ξεχωριστά (10 πυρήνες επεξεργαστή).

- **1650-1700** χρήστες στη συστάδα των Moodle εξυπηρετητών με τη χρήση Unison (6 πυρήνες επεξεργαστή)

- **1800-1850** χρήστες στη συστάδα των Moodle εξυπηρετητών με τη χρήση NFS (10 πυρήνες επεξεργαστή).

Το μέγιστο πλήθος χρηστών το οποίο μπορεί να είναι ανεκτό από τις απαιτήσεις του συστήματος αντιστοιχεί σε APDEX score στο εύρος 0.7 με 0.75.
Ωστόσο, θα πρέπει να επισημανθεί ότι το παραπάνω μέγιστο πλήθος χρηστών αντιστοιχεί για τη συγκεκριμένο δοκιμή φορτίου που διενεργήθηκε. Το συγκεκριμένο σύστημα δύναται να αντέξει περισσότερους χρήστες, οι οποίοι δεν είναι τόσο απαιτητικοί (εκτελούν κυρίως browsing) όσο οι εικονικοί χρήστες που προσομοιώθηκαν, ή λιγότερους αν είναι μεγαλύτερο το φορτίο.

Κεφάλαιο 6: Δοκιμή καταπόνησης

Το τελευταίο τεστ που διεξήχθη είναι μια δοκιμή καταπόνησης. Στόχος, αποτελεί η μελέτη της απόκρισης του συστήματος όταν αυτό υποβάλλεται σε **υπερβολικό** αριθμό ταυτόχρονων χρηστών. Για το σκοπό αυτό, χρησιμοποιείται ένα τεστ, στο οποίο μεταβάλλεται μόνο το πλήθος των χρηστών, χωρίς να διαφέρει η δομή του από τα προηγούμενα. Επιπλέον το συγκεκριμένο τεστ τερματίζει απότομα, χωρίς να υπάρχει χρόνος ομαλού τερματισμού των χρηστών όπως συμβαίνει στην περίπτωση των τεστ της δοκιμής φορτίου. Έτσι, προσομοιώνουμε συνολικά **4240** χρήστες στην αρχιτεκτονική της συστάδας των Moodle εξυπηρετητών (με 10 πυρήνες επεξεργαστή), ενώ γίνεται χρήση του NFS. Τα αποτελέσματα απεικονίζονται παρακάτω.

Πλήθος ταυτόχρονων χρηστών	**Συστάδα Moodle - NFS με Redis**		**APDEX score**
	Login	View forum	
4240	26784.96	16042.09	0.1

Πίνακας 6: Δοκιμή καταπόνησης 4240 ταυτόχρονων χρηστών(10 πυρήνες επεξεργαστή)

Από τους παραπάνω χρόνους, διαπιστώνεται ότι το σύστημα οριακά δεν ανταποκρίνεται (unresponsive). Επιπλέον, το τεστ παρουσίασε και σφάλματα με κωδικό 504/Gateway Time-out. Ο κωδικός αυτού του σφάλματος, προέρχεται από τον Nginx και υποδηλώνει ότι ο χρόνος προκειμένου να λάβει απάντηση ο Nginx από κάποιον Moodle εξυπηρετητή ήταν υπερβολικά μεγάλος, με αποτέλεσμα την αποτυχία των αιτημάτων αυτών.
Το συμπέρασμα που λαμβάνεται ωστόσο από τη συγκεκριμένη δοκιμή είναι ότι το σύστημα δεν καταρρέει (crashing), αλλά ανταποκρίνεται με πολύ μεγάλους χρόνους απόκρισης.

Κεφάλαιο 7: Δοκιμή επίδοσης και πραγματική συμπεριφορά του συστήματος

Με τη διαδικασία της δοκιμής επίδοσης της εφαρμογής του Moodle ήταν δυνατή η προσομοίωση της λειτουργίας του συστήματος. Έτσι, η προσομοίωση επικεντρώνεται αποκλειστικά στον εντοπισμό αδυναμιών του συστήματος.
Ωστόσο, το περιβάλλον της προσομοίωσης μπορεί να διαφέρει από την πραγματική συμπεριφορά και τους χρόνους απόκρισης του συστήματος. Οι παράμετροι που δε λαμβάνονται υπόψη κατά τη διαδικασία της προσομοίωσης, αλλά εμφανίζονται στην πραγματικότητα είναι οι εξής:

- Καθυστέρηση στο δίκτυο: Επειδή η δοκιμή διεξάγεται από υπολογιστή που βρίσκεται στο ίδιο δίκτυο με τον Moodle εξυπηρετητή, δε λαμβάνεται υπόψη η καθυστέρηση

που εισάγεται από το δίκτυο και τη διαδρομή που ακολουθούν τα αιτήματα ενός πραγματικού χρήστη.

- Απρόβλεπτες αστοχίες που μπορεί να εμφανιστούν, όπως αστοχία ενός εξυπηρετητή.

- Αστοχίες λόγω σφαλμάτων (bugs) που μπορεί να υπάρχουν ή να εμφανιστούν σε μελλοντικές αναβαθμίσεις του moodle.

7.1: Δυσκολίες κατά τη διαδικασία της δοκιμής φορτίου

Η διαδικασία διεξαγωγής επίδοσης (δοκιμή φορτίου και δοκιμή καταπόνησης) απαιτεί προσεκτικό σχεδιασμό προκειμένου να ληφθούν υπόψη όλες οι παράμετροι ενός συστήματος. Οι ενέργειες που ακολουθήθηκαν ήταν οι εξής:

- Μελέτη και κατάστρωση ορθής δοκιμής φορτίου, λαμβάνοντας υπόψη την αναμενόμενη και επιθυμητή λειτουργία του συστήματος.

- Αξιολόγηση των αστοχιών του προγράμματος. Δεν ήταν λίγες οι φορές όπου το JMeter εμφάνισε κάποια σφάλματα κατά την εκτέλεσή του. Επομένως ήταν αναγκαία η διόρθωσή τους και ο προσδιορισμός της αιτίας τους.

- Αξιολόγηση όλων των δυνατών βελτιστοποιήσεων και μελέτη των σημαντικότερων εξ αυτών: Στο σύστημα που μελετήθηκε ήταν αναγκαία η μελέτη βελτιστοποίησης αναφορικά με τον Apache, την MySQL, τον Nginx αλλά και τη PHP, γεγονός που επιβεβαιώνεται από την σύγκριση μεταξύ των επιλογών.

- Προσδιορισμός της στένωσης του συστήματος. Ανάλογα με την αρχιτεκτονική που μελετάται, είναι διαφορετικές οι αιτίες που προκαλούν μεγαλύτερους χρόνους απόκρισης.

- Ο συνολικός χρόνος σχεδιασμού και εκτέλεσης της προσομοίωσης: Η δοκιμή μιας εφαρμογής, απαιτεί αρκετό χρόνο για τη σχεδίαση, αλλά και ανάλυση των αποτελεσμάτων. Έτσι, είναι σημαντική η σωστή διαχείριση του χρόνου, προκειμένου να μελετηθούν σωστά όλες οι επιθυμητές παράμετροι του συστήματος.

Μέρος Β

Κεφάλαιο 8: Εγκαταστάσεις λογισμικών και απαραίτητες παραμετροποιήσεις

8.1: Εγκατάσταση Apache-MySQL-PHP

Οι παρακάτω οδηγίες για την εγκατάσταση των απαιτούμενων λογισμικών προέρχονται από την επίσημη ιστοσελίδα του Moodle [30].

Για την εγκατάσταση των Apache, MySQL και PHP, εκτελούμε:

```
sudo apt-get update
sudo apt-get install apache2 mysql-client mysql-server php libapache2-mod-php
```

Στη συνέχεια εγκαθιστούμε επιπλέον απαιτούμενο λογισμικό.

```
sudo apt-get install graphviz aspell ghostscript clamav php7.4-pspell php7.4-curl php7.4-gd php7.4-intl php7.4-mysql php7.4-xml php7.4-xmlrpc php7.4-ldap php7.4-zip php7.4-soap php7.4-mbstring
```

Για να λάβουν χώρα οι αλλαγές, θα πρέπει να επανεκκινηθεί ο Apache ως εξής:

```
sudo service apache2 restart
```

8.2: Εγκατάσταση-παραμετροποίηση Moodle

Η εγκατάσταση του Moodle γίνεται με τη χρήση του εργαλείου Git, οπότε απαιτείται η εγκατάστασή του:

```
sudo apt-get install git

cd /opt
```

Στη συνέχεια, εγκαθιστούμε τον κώδικα του Moodle.

```
sudo git clone git://git.moodle.org/moodle.git

cd moodle
```

Για την απόκτηση των διαθέσιμων branches εκτελούμε:

```
sudo git branch -a
```

Έπειτα, επιλέγουμε τη νεότερη έκδοση του Moodle:

```
sudo git branch --track MOODLE_400_STABLE origin/MOODLE_400_STABLE
```

Τέλος, ολοκληρώνουμε την εγκατάσταση όπως διακρίνεται παρακάτω:

```
sudo git checkout MOODLE_400_STABLE
```

Για την αντιγραφή του Moodle στο /var/www/html/, εκτελούμε:

```
sudo cp -R /opt/moodle /var/www/html/
```

Ύστερα, δημιουργούμε το φάκελο moodledata που περιέχει όλα τα αρχεία του Moodle.

```
sudo mkdir /var/moodledata
```

Τέλος, ορίζουμε τα κατάλληλα δικαιώματα ως εξής:

```
sudo chown -R www-data /var/moodledata

sudo chmod -R 777 /var/moodledata

sudo chmod -R 0755 /var/www/html/moodle
```

Ο λόγος που αντιγράφουμε το Moodle στο /opt είναι διότι μας παρέχει την ευελιξία να πραγματοποιούμε οποιεσδήποτε αλλαγές επιθυμούμε εκεί, όπως είναι η εγκατάσταση νέων plugins, ή και updates μέσω του Git ευκολότερα, προτού γίνουν στο webroot (/var/www/html/moodle).

Παραμετροποίηση MySQL

Αναφορικά με τη MySQL, εισάγουμε τις παρακάτω ρυθμίσεις στο αρχείο παραμετροποίησης της MySQL:

```
sudo vi /etc/mysql/mysql.conf.d/mysqld.cnf

default_storage_engine = innodb

innodb_file_per_table = 1
```

Για να λάβουν χώρα οι αλλαγές, επανεκκινούμε την MySQL:

```
sudo service mysql restart
```

Για τη δημιουργία της βάσης δεδομένων του Moodle και ενός MySQL user για τη βάση αυτή, επιτελούμε τα εξής:

```
sudo mysql -u root -p
```

δημιουργία της βάσης με όνομα moodle:

```
CREATE DATABASE moodle DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
```

Αναφορικά με τη δημιουργία του MySQL user, απαιτούνται τα εξής:

```
create user 'moodle_user'@'localhost' IDENTIFIED BY 'passwordformoodle_user';
```

Στο σημείο αυτό, θα πρέπει να προστεθούν τα κατάλληλα δικαιώματα στο χρήστη για τη βάση moodle, όπου στο πεδίο passwordformoodle_user εισάγουμε τον επιθυμητό κωδικό για το χρήστη.

```
GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,CREATE TEMPORARY TABLES,DROP,INDEX,ALTER ON moodle.* TO 'moodle_user'@'localhost';
```

Στη συνέχεια, αντιγράφουμε το αρχείο config-dist.php στο config.php, εισάγοντας τις παρακάτω παραμετροποιήσεις:

```
$CFG->dbtype    = 'mysqli';
$CFG->dblibrary = 'native';
$CFG->dbhost    = 'localhost';
$CFG->dbname    = 'moodle';
$CFG->dbuser    = 'moodle_user';
$CFG->dbpass    = 'passwordformoodle_user;
$CFG->wwwroot = ‘http:147.102.40.42’
$CFG->dataroot = ‘/’var/moodledata
```

Παραμετροποίηση του εξυπηρετητή Apache

Ύστερα, ορίζουμε στο αρχείο παραμετροποίησης του Apache, το οποίο είναι το /etc/apache2/sites-available/000-default.conf τα ακόλουθα:

```
DocumentRoot /var/www/html/moodle
```

Με αυτόν τον τρόπο, θα προσκομίζονται σωστά τα ζητούμενα αρχεία του Moodle όταν επισκεπτόμαστε την ιστοσελίδα με το url http://147.102.40.42.

Ακόμα, επισκεπτόμενοι την παραπάνω διεύθυνση, ακολουθούμε τα τελικά βήματα της εγκατάστασης ορίζοντας το όνομα και τον κωδικό πρόσβασης για το διαχειριστή της ιστοσελίδας.

Αφού έχει ολοκληρωθεί η εγκατάσταση, συνίσταται να οριστούν και τα παρακάτω “system paths” για καλύτερη επίδοση του Moodle:

- **Path to du: /usr/bin/du**
- **Path to apsell: /usr/bin/aspell**
- **Path to dot: /usr/bin/dot**

Αναφορικά με την απόκρυψη των εσωτερικών διαδρομών (internal paths), απαιτούνται οι εξής παραμετροποιήσεις στο αρχείο παραμετροποίησης του Apache [31] :

```
RewriteEngine On
    RewriteRule "(\/vendor\/)" - [F]
    RewriteRule "(\/node_modules\/)" - [F]
    RewriteRule "(\/lib/classes\/)" - [F]
```

```
RewriteRule "(^|/)\.(?!well-known\/)" - [F]
RewriteRule "(composer\.json)" - [F]
RewriteRule "(\.lock)" - [F]
RewriteRule "(\/environment.xml)" - [F]
Options -Indexes
RewriteRule "(\/install.xml)" - [F]
RewriteRule "(\/README)" - [F]
RewriteRule "(\/readme)" - [F]
RewriteRule "(\/moodle_readme)" - [F]
RewriteRule "(\/upgrade\.txt)" - [F]
RewriteRule "(phpunit\.xml\.dist)" - [F]
RewriteRule "(\/tests\/behat\/)" - [F]
RewriteRule "(\/fixtures\/)" - [F]
```

Τέλος, για να μην είναι εγγράψιμος ο φάκελος που περιέχει τον κώδικα του Moodle εκτελούμε:

```
sudo chmod -R 0755 /var/www/html/moodle
```

8.3: Εγκατάσταση php7.4-fpm

Η έκδοση PHP που εγκαταστάθηκε προηγουμένως (libapache2-mod-php), είναι επαρκής για τη λειτουργία του Apache ως εξυπηρετητής ιστού, προκειμένου να προσκομίζει σε χρήστες αρχεία τύπου .php, δημιουργώντας έτσι δυναμικές ιστοσελίδες. Ωστόσο, μια από τις αρχικές προτεινόμενες βελτιστοποιήσεις είναι η αλλαγή της χρησιμοποιούμενης έκδοσης PHP και η μετάβαση στην έκδοση FPM (php7.4-fpm). Αυτό, διότι η έκδοση php-fpm (FastCGI Process Manager) υπερέχει σε επιδόσεις από τη mod-php. Συγκριτικά με την έκδοση mod-php, προσφέρει περισσότερη ασφάλεια, ταχύτητα μιας και κλιμακώνεται εύκολα για ανάγκες που απαιτούν περισσότερους χρήστες, καθώς και υποστηρίζει αρκετές τροποποιήσεις και ρυθμίσεις [32].

Για την εγκατάσταση της έκδοσης php7.4-fpm, εγκαθιστούμε αρχικά τα προαπαιτούμενα λογισμικά:

```
sudo apt install software-properties-common
sudo add-apt-repository ppa:ondrej/php
```

Στη συνέχεια, εγκαθιστούμε την έκδοση 7.4 της PHP, η οποία είναι συμβατή με το JMeter, καθώς η νεότερη (8.1) δεν υποστηρίζεται την τρέχουσα στιγμή από το πρόγραμμα.

```
sudo apt update
```

```
sudo apt install php7.4 php7.4-fpm
```

Για την ενεργοποίηση των Apache modules επιτελούμε:

```
sudo a2enmod actions fcgid alias proxy_fcgi
```

Έπειτα, ορίζουμε τον Apache να λειτουργεί χρησιμοποιώντας το FPM/FastCGI, εισάγοντας τις κατάλληλες ρυθμίσεις στο αρχείο παραμετροποίησής του.

```
sudo vim /etc/apache2/sites-available/000-default.conf
```

Για την επιλογή της PHP-FPM θα πρέπει να προστεθούν τα παρακάτω:

```
<FilesMatch \.php$>
    # 2.4.10+ can proxy to unix socket
    SetHandler "proxy:unix:/var/run/php/php7.4-fpm.sock|fcgi://localhost"
</FilesMatch>
```

Ενεργοποίηση Zend OPcache

Μια ακόμη βελτιστοποίηση που προτείνεται κατά τα πρώτα στάδια της εγκατάστασης του Moodle, είναι η ενεργοποίηση της επέκτασης OPcache. Η επέκταση αυτή, προσφέρει αυξημένη ταχύτητα και λιγότερη χρήση μνήμης [33]. Αν και από την έκδοση **PHP 5.5** και αργότερα είναι ενεργοποιημένη από προεπιλογή, εισάγουμε τις παρακάτω μεταβλητές στο php.ini αρχείο της PHP:

```
sudo vi /etc/php/7.4/fpm/php.ini
```

```
[opcache]
opcache.enable = 1
opcache.memory_consumption = 1024
opcache.max_accelerated_files = 10000
opcache.revalidate_freq = 60

; Required for Moodle
opcache.use_cwd = 1
opcache.validate_timestamps = 1
opcache.save_comments = 1
opcache.enable_file_override = 0
```

Ενεργοποίηση του Cron

Για τη σωστή και ομαλή λειτουργία του Moodle, είναι αναγκαία η ενεργοποίηση του Cron, μέσω του οποίου εκτελούνται ασύγχρονα σημαντικές διεργασίες του Moodle [34].
Για να τρέξει το Cron script του Moodle αρκεί να εκτελέσουμε την εντολή :

```
sudo php /var/www/html/moodle/admin/cli/cron.php
```

και για να οριστεί ένα crontab, το οποίο θα τρέχει το συγκεκριμένο Cron script κάθε λεπτό, θα πρέπει να εκτελεστεί η παρακάτω εντολή:

```
sudo crontab -u www-data -e
* * * * * /usr/bin/php  /var/www/html/moodle/admin/cli/cron.php >/dev/null
```

Ολοκλήρωση παραμετροποιήσεων Moodle

Προκειμένου να ολοκληρωθούν όλες οι απαραίτητες ρυθμίσεις του Moodle, ελέγχουμε τυχόν ειδοποιήσεις για προτεινόμενες αλλαγές. Έτσι, στην καρτέλα του Moodle στο πεδίο Dashboard⟶Site administration⟶Server⟶Environment εμφανίζεται η σύσταση αλλαγής της μεταβλητής **max_input_vars = 5000** [35]**.** Επομένως, για να πραγματοποιήσουμε την αλλαγή αλλάζουμε την μεταβλητή στην επιθυμητή τιμή στο αρχείο **/etc/php/7.4/fpm/php.ini**. Μια επιπλέον συνιστώμενη ρύθμιση αποτελεί η αλλαγή από HTTP σε HTTPS η οποία πραγματοποιείται στο κεφάλαιο 4.2 εισάγοντας και τον Nginx σε ρόλο reverse proxy.

Εγκατάσταση Moodle Benchmark plugin

Έτσι για την εγκατάσταση και ενεργοποίησή του μετατρέπουμε τον φάκελο /var/www/html/moodle σε εγγράψιμο με την εντολή:

```
sudo chmod -R 777 /var/www/html/moodle
```

Στη συνέχεια, ανεβάζουμε το .zip αρχείο μέσω του πεδίου Dashboard⟶Site administration⟶Plugins⟶Install plugins και έπειτα Install plugin from ZIP file.

Για να τεθεί σε λειτουργία το plugin, μεταβαίνουμε στο πεδίο Dashboard→Site administration→Reports→Benchmark. Τα προτεινόμενα αποτελέσματα αφορούν στις επιδόσεις και τους χρόνους επεξεργασίας αιτημάτων που υποβάλλονται από το εκτελούμενο Benchmark Test.

8.4: Εγκατάσταση και ρύθμιση Apache JMeter

Όπως προαναφέρθηκε, το JMeter είναι αποκλειστικά γραμμένο σε Java και γι' αυτό απαιτείται η εγκατάσταση της Java στο Linux μηχάνημα που το φιλοξενεί. Για την εγκατάστασή της ακολουθούμε τα παρακάτω βήματα:

```
sudo apt-get update
```

Προκειμένου να μπορούν να τρέξουν εφαρμογές Java είναι αρκετή η εγκατάσταση του περιβάλλοντος εκτέλεσης προγραμμάτων Java, Java Runtime Environment (JRE):

```
sudo apt install default-jre
```

Για την ανάπτυξη ωστόσο και εκτέλεση προγραμμάτων Java εγκαθίσταται το Java Development Kit (JDK).

```
sudo apt-get install openjdk-16-jdk
```

Για να επιβεβαιώσουμε την έκδοση ,αλλά και εγκατάστασή της, εκτελούμε την εντολή:

```
sudo java -version
openjdk version "16.0.1" 2021-04-20
OpenJDK Runtime Environment (build 16.0.1+9-Ubuntu-120.04)
OpenJDK 64-Bit Server VM (build 16.0.1+9-Ubuntu-120.04, mixed mode, sharing)
```

Για την επιβεβαίωση εγκατάστασης του compiler εκτελούμε:

```
javac -version
javac 16.0.1
```

Αναφορικά με το Apache JMeter, επισκεπτόμαστε την επίσημη ιστοσελίδα του και κατεβάζουμε το αρχείο **apache-jmeter-5.4.3.tgz** [36].
Στη συνέχεια, κάνουμε unzip το αρχείο:

```
tar zxvf apache-jmeter-5.4.3.tgz
```

Ρύθμιση JVM για το JMeter

Πριν την εκτέλεση των τεστ του JMeter είναι αναγκαία η ρύθμιση του JMeter. Επειδή τα τεστ απαιτούν χώρο από τη μνήμη του μηχανήματος στο οποίο είναι εγκατεστημένο το πρόγραμμα, οφείλουμε να παραμετροποιήσουμε κάποιες προεπιλεγμένες ρυθμίσεις. Κατά την εκτέλεση των πρώτων τεστ παρατηρήθηκε το σφάλμα ***java.lang.OutOfMemoryError: Java heap space***. Αυτό το σφάλμα εμφανίζεται, διότι από προεπιλογή είναι ρυθμισμένο το πρόγραμμα να χρησιμοποιεί 1024 MB για το Java heap space. Έτσι, επειδή τα τεστ απαιτούσαν περισσότερο χώρο από το Java heap εμφανιζόταν το σφάλμα αυτό. Προκειμένου να αυξήσουμε το χώρο του Java heap που χρησιμοποιεί το πρόγραμμα, δημιουργούμε ένα αρχείο στον φάκελο /apache-jmeter-5.4.3/bin με όνομα setenv.sh [37].
Στη συνέχεια, προσθέτουμε τις επιθυμητές αλλαγές:

```
# This is the file bin/setenv.sh,
# it will be sourced in by bin/jmeter

# Use a bigger heap and metaspace, than the default
export HEAP="-Xms3G -Xmx6G -XX:MaxMetaspaceSize=256m"

# Try to guess the locale from the OS. The space as value is on purpose!
export JMETER_LANGUAGE=" "

#GC_ALGO Java runtime options to specify JVM garbage collection algorithm.
export GC_ALGO="-XX:+UseG1GC -XX:MaxGCPauseMillis=250
-XX:G1ReservePercent=20"
```

Με αυτόν τον τρόπο, μεταβάλλουμε το μέγιστο διαθέσιμο μέγεθος του Java heap στα 6GB και το αρχικό διαθέσιμο μέγεθος του JVM στα 3GB. Επιπλέον, ορίζουμε τον προεπιλεγμένο JVM garbage collector με τις ρυθμίσεις του.

8.5: Εγκατάσταση αυτοδύναμου MySQL εξυπηρετητή

Εκτελούμε τα παρακάτω βήματα.

Αρχικά, πρέπει να εγκατασταθεί η MySQL στο νέο μηχάνημα και αυτό επιτυγχάνεται με τα βήματα που περιγράφονται στην εγκατάσταση του Moodle στο Κεφάλαιο 8.1.
Στη συνέχεια, θα πρέπει να μεταφερθεί η βάση δεδομένων από το προηγούμενο μηχάνημα σε αυτό.
Επομένως, θα πρέπει πρώτα να εξαγάγουμε τη βάση δεδομένων (dump) και στη συνέχεια να την αντιγράψουμε στο νέο μηχάνημα [38]. Για την εξαγωγή της βάσης, εκτελούμε την εντολή:

```
sudo mysqldump -u root -p  moodle> /home/theo/moodle.sql
```

Ση συνέχεια, μεταφέρουμε τη βάση αυτή στο νέο μηχάνημα μέσω SCP (Secure Copy):

```
sudo scp theo@147.102.40.49:/home/theo/moodle.sql /home/theo
```

Έπειτα κάνουμε επαναφορά της βάση στο νέο μηχάνημα με την εντολή:

```
sudo mysql -u root -p moodle < /home/theo/moodle.sql
```

Αφού έχει δημιουργηθεί η βάση του Moodle, δημιουργούμε ένα νέο χρήστη, ο οποίος θα έχει πρόσβαση στη βάση δεδομένων από το μηχάνημα που φιλοξενεί το Moodle (147.102.40.42).

```
create user 'moodle_user_mysql'@'147.102.40.42' IDENTIFIED BY
'passwordformoodle_user_mysql';
```

Ύστερα, προσδίδουμε τα κατάλληλα δικαιώματα στο χρήστη αυτό με την εντολή:

```
GRANT SELECT,INSERT,UPDATE,DELETE,CREATE,CREATE TEMPORARY
TABLES,DROP,INDEX,ALTER ON moodle.* TO 'moodle_user_mysql'@'147.102.40.42';
```

Τέλος, εκτελούμε:

```
FLUSH PRIVILEGES;
```

Επιπλέον, είναι αναγκαία και η ρύθμιση του αρχείου παραμετροποίησης **/etc/mysql/mysql.conf.d/mysql.cnf** της MySQL προκειμένου να γίνονται δεκτές οι συνδέσεις στη βάση δεδομένων από άλλα μηχανήματα:

```
bind-address  = 0.0.0.0
```

Ακόμη, προσθέτουμε κανόνα στο τείχος προστασίας UFW, ώστε να επιτρέπονται οι συνδέσεις στη βάση μόνο από το συγκεκριμένο μηχάνημα:

```
sudo ufw allow from 147.102.40.42 to any port 3306
```

Τελικό βήμα, αποτελεί να ορίσουμε στο αρχείο παραμετροποίησης του Moodle, το μηχάνημα που φιλοξενείται η βάση δεδομένων, αλλά και τον χρήστη που χρησιμοποιείται για την πρόσβαση σε αυτήν. Έτσι, αλλάζουμε τις παρακάτω μεταβλητές στο config.php του Moodle.

```
$CFG->dbhost    = '147.102.40.49';
$CFG->dbuser    = 'moodle_user_mysql';
$CFG->dbpass    = 'passwordformoodle_user_mysql;
```

8.6: Εγκατάσταση Nginx και ρύθμιση για τη λειτουργία του ως reverse proxy

Για την εγκατάσταση του Nginx εκτελούμε την παρακάτω εντολή:

```
sudo apt-get install nginx
```

Επιπλέον, διαγράφουμε το προεπιλεγμένο αρχείο παραμετροποίησής του και δημιουργούμε ένα νέο με όνομα **/etc/nginx/sites-available/reverse-proxy.conf** και το συνδέουμε με τη χρήση symbolic link με την εντολή:

```
sudo  ln -s /etc/nginx/sites-available/reverse-proxy.conf
/etc/nginx/sites-enabled/reverse-proxy.conf
```

Προκειμένου να μπορέσει να λειτουργήσει ο Nginx ως reverse proxy, κρίνεται αναγκαία η προσθήκη κατάλληλων ρυθμίσεων στο αρχείο παραμετροποίησής του **/etc/nginx/sites-available/reverse-proxy.conf.** Για το σκοπό αυτό, γίνεται χρήση του upstream module, μέσω του οποίου μπορούν να ορισθούν ομάδες από εξυπηρετητές που θα αποτελούν τους back-end εξυπηρετητή. Έτσι, προσθέτουμε τις παρακάτω ρυθμίσεις.

```
upstream backendmoodles {
        server 147.102.40.42:80 max_fails=2 fail_timeout=20s;
}
server{
        server_name bench-test.cn.ece.ntua.gr  ;
        keepalive_timeout   70;
        listen 443 ssl; # managed by Certbot
        listen [::]:443 ssl ipv6only=on; # managed by Certbot
        ssl_certificate /etc/letsencrypt/live/bench-test.cn.ece.ntua.gr/fullchain.pem; # managed
by Certbot
    ssl_certificate_key /etc/letsencrypt/live/bench-test.cn.ece.ntua.gr/privkey.pem; # managed
by Certbot
```

```
        include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
        ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot

location / {
        proxy_pass http://backendmoodles;
        proxy_http_version 1.1;
        proxy_set_header Connection "";
        proxy_set_header X-Forwarded-Proto https;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_next_upstream error timeout http_500;
}
}
```

Κατά τη χρήση του certbot δημιουργήθηκαν αυτόματα οι παραπάνω γραμμές που αναφέρονται στο certbot.
Η σημαντικότερη εντολή που προσθέτουμε και που μετατρέπει τον Nginx σε reverse proxy είναι η **proxy_pass http://backendmoodles**. Με αυτόν τον τρόπο, ορίζεται η προώθηση όλων των αιτημάτων προς τον ή τους εξυπηρετητές που ορίζονται στο upstream module. Επιπλέον, γίνεται χρήση ορισμένων παραμέτρων στο πεδίο location, οι οποίες έχουν μεγάλη σημασία για την επίδοση του Nginx. Αυτές επεξηγούνται παρακάτω:

- **proxy_http_version 1.1:** Από προεπιλογή, ο Nginx χρησιμοποιεί την έκδοση HTTP/1.0, επομένως με αυτόν τον τρόπο δηλώνουμε τη χρήση του HTTP/1.1. Η έκδοση HTTP/1.0 προσθέτει την επικεφαλίδα **Connection:close.** Αυτό έχει ως αποτέλεσμα να τερματίζονται και να κλείνουν οι συνδέσεις με το που τελειώνει κάποιο αίτημα.

- **proxy_set_header Connection "":** Ίσως είναι και η πιο σημαντική ρύθμιση, καθώς με αυτόν τον τρόπο αφαιρείται η επικεφαλίδα **Connection:close** προκειμένου να διατηρούνται ανοιχτές οι συνδέσεις.

- **proxy_set_header X-Forwarded-Proto https:** Κάθε απάντηση HTTP από τον back-end εξυπηρετητή μετατρέπεται σε HTTPS.

- **proxy_set_header Host $host:** Ορίζει τον πραγματικό host ο οποίος ζητήθηκε από τον client.

- **proxy_set_header X-Real-IP $remote_addr:** Προωθεί την πραγματική IP διεύθυνση του client στον back-end εξυπηρετητή, στον οποίο προωθούνται τα αιτήματα.

- **proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for:** Αποτελεί μία λίστα με τις IP διευθύνσεις των εξυπηρετητή, στους οποίους προωθήθηκαν τα αιτήματα ενός χρήστη.

- **proxy_next_upstream error timeout http_500:** Με αυτόν τον τρόπο, θεωρεί ο Nginx, ότι ο κωδικός HTTP 500 (Internal Server Error) από έναν εξυπηρετητή στο upstream module αποτελεί μία αποτυχημένη προσπάθεια ενός αιτήματος.

Παραμετροποίηση nginx.conf

Οι επιπλέον ρυθμίσεις που είναι πολύ κρίσιμες για την επίδοση του Nginx ορίζονται στο αρχείο παραμετροποίησης με όνομα nginx.conf. Αυτές είναι οι ακόλουθες:

- **worker_rlimit_nofile 40000 :** Έτσι επιτρέπουμε στον Nginx τη χρήση περισσότερων File Descriptors.

- **worker_connections 10000:** Με αυτόν τον τρόπο ορίζουμε το μέγιστο πλήθος από ταυτόχρονες συνδέσεις, τις οποίες μπορεί να διατηρεί ο worker process του Nginx.

Τέλος, πρέπει να αναφερθεί ότι ο Nginx χρησιμοποιείται σε συνδυασμό με τις αρχιτεκτονικές που περιγράφηκαν προηγουμένως, προκειμένου να διαπιστωθεί η διαφορά που τυχόν επιφέρει στην επίδοση ο reverse proxy εξυπηρετητή.

Ρύθμιση του wwwroot στο config.php του Moodle

Για τη λειτουργία του Nginx ως reverse proxy, οφείλουμε να δηλώσουμε στο αρχείο παραμετροποίησης του Moodle την διεύθυνση IP του. Έτσι προσθέτουμε τις παρακάτω επιλογές:

```
$CFG->wwwroot   = 'https://147.102.40.62';
$CFG->sslproxy = true;
```

Έτσι, ανακεφαλαιώνοντας με τις παραπάνω επιλογές, ο Nginx λειτουργεί επιτυχημένα ως reverse proxy προωθώντας όλα τα εισερχόμενα αιτήματα στον Moodle εξυπηρετητή (147.102.40.42).

8.7: Εγκατάσταση certbot: ρύθμιση HTTPS και SSL τερματισμός

Για τη μετατροπή της ιστοσελίδας από HTTP σε HTTPS, εγκαθίσταται το λογισμικό certbot. Το certbot, είναι λογισμικό ανοικτού κώδικα το οποίο χρησιμοποιεί αυτόματα Let's encrypt certificates για την ενεργοποίηση του HTPS [39].

Αρχικά, πρέπει να γνωρίζουμε το FQDN (Fully Qualified Domain Name) του proxy. Αυτό ανακαλύπτεται με την εντολή:

```
dig -x 147.102.40.62
```

Η απάντηση που λαμβάνεται είναι:

```
62.40.102.147.in-addr.arpa. 86400 IN    PTR     bench-test.cn.ece.ntua.gr.
```

Επομένως, για την εγκατάσταση του certbot επιτελούμε τα εξής:

```
sudo apt install certbot python3-certbot-nginx
```

Οπότε, για την ενεργοποίηση εκτελούμε:

```
sudo certbot --nginx -d bench-test.cn.ntua.gr
```

και επιλέγουμε τις επιθυμητές ρυθμίσεις που εμφανίζονται.

Οι αλλαγές που πραγματοποίησε το certbot εμφανίζονται στο αρχείο παραμετροποίησης του Nginx και θα παρουσιαστούν στη συνέχεια μαζί με τις αναγκαίες ρυθμίσεις για τη λειτουργία του ως reverse proxy.

8.8: Εγκατάσταση και ρύθμιση του Redis

Για την εγκατάστασή του ακολουθούμε τα παρακάτω βήματα.

```
sudo apt install redis-server
```

Στη συνέχεια ορίζουμε στο αρχείο παραμετροποίησής του, /etc/redis/redis.conf τη λειτουργία του, ως service του συστήματος.

```
supervised systemd
```

και για να φορτωθούν επιτυχημένα οι αλλαγές εκτελούμε:

```
sudo systemctl restart redis.service
```

Για να είναι το Redis προσβάσιμο από τους δύο Moodle εξυπηρετητές, οι οποίοι βρίσκονται σε διαφορετικά μηχανήματα, θα πρέπει να επιτραπεί η απομακρυσμένη πρόσβαση σε αυτό. Επομένως, ορίζουμε στο αρχείο παραμετροποίησης τα παρακάτω.

Αρχικά θέτουμε έναν ισχυρό κωδικό πρόσβασης και τον προσθέτουμε κατάλληλα:

```
requirepass password_for_redis
```

Στη συνέχεια, οφείλουμε να επιτρέψουμε την απομακρυσμένη πρόσβαση στη θύρα που τρέχει το Redis θέτοντας τις κατάλληλες εντολές για τα Moodles στο UFW:

```
sudo ufw allow from 147.102.40.42 to any port 6379
sudo ufw allow from 147.102.40.49 to any port 6379
```

Αντίστοιχα, ρυθμίζουμε και τη χρήση του Redis στους Redis clients (Moodle), εκκινώντας από την εγκατάσταση της επέκτασης του Redis για την PHP:

```
sudo apt-get install php7.4-redis
```

Στη συνέχεια, ορίζουμε το Redis ως “session handler” στο αρχείο παραμετροποίησης **/etc/php/7.4/fpm/php.ini** της PHP στους Moodle εξυπηρετητές.

```
session.save_handler = redis (από file που ήταν προηγουμένως)

session.save_path = "tcp://147.102.40.52:6379?auth=password_for_redis"
```

Επιπλέον, είναι απαραίτητη η ρύθμιση του Redis και από το γραφικό περιβάλλον του Moodle. Στο πεδίο **Site administration→Caching→Configuration**, επιλέγουμε **add instance** για τον καθορισμό του Redis προσθέτοντας τις κατάλληλες ρυθμίσεις:

Store name	*Redis*
Server	*147.102.40.52:6379*
Password	**********

```
Use serializer  igbinary serializer
```

Ακόμη, επιλέγουμε το σκοπό για τον οποίο επιθυμούμε να λειτουργήσει το Redis και αυτός γίνεται από το πεδίο **Stores used when no mapping is present** επιλέγοντας Redis ως store mapping για τα Application και Session.
Τέλος ορίζουμε και στο αρχείο παραμετροποίησης του Moodle (config.php) το Redis ως κρυφή μνήμη προσθέτοντας τις παρακάτω γραμμές:

```
$CFG->session_handler_class = '\core\session\redis';
$CFG->session_redis_host = '147.102.40.52';
$CFG->session_redis_port = 6379;
$CFG->session_redis_database = 0;
$CFG->session_redis_auth = 'password_for_redis';
$CFG->session_redis_prefix = '';
$CFG->session_redis_acquire_lock_timeout = 120;
$CFG->session_redis_lock_expire = 7200;
$CFG->session_redis_lock_retry = 100;
$CFG->session_redis_serializer_use_igbinary = true;
```

8.9: Εγκατάσταση και ρύθμιση NFS

Στον NFS host εκτελούμε τα παρακάτω:

```
sudo apt update
sudo apt install nfs-kernel-server
```

Ενώ στους NFS clients εγκαθιστούμε το πακέτο nfs-common:

```
sudo apt install nfs-common
```

Στη συνέχεια, στον NFS host δημιουργούμε τον κοινόχρηστο φάκελο:

```
sudo mkdir /var/nfs/host_moodledata  -p
```

έπειτα θέτουμε τον κατάλληλο owner:

```
sudo chown nobody:nogroup /var/nfs/host_moodledata
```

Επιπλέον, θα πρέπει να οριστούν και οι NFS clients, με τους οποίους θα διαμοιράζεται ο παραπάνω φάκελος και αυτό επιτυγχάνεται προσθέτοντας τις κατάλληλες γραμμές στο αρχείο /etc/exports.

```
/var/nfs/host_moodledata 147.102.40.49(rw,sync,no_subtree_check,no_root_squash)

/var/nfs/host_moodledata 147.102.40.42(rw,sync,no_subtree_check,no_root_squash)
```

Με τις παραπάνω επιλογές ορίζουμε τα ακόλουθα:

- **rw:** Παραχωρεί το δικαίωμα ανάγνωσης και εγγραφής (read και write) στους NFS clients στον κοινόχρηστο δικτυακό δίσκο.

- **sync:** Το NFS γράφει πρώτα τυχόν αλλαγές στο δίσκο πριν από κάθε απάντηση. Με αυτόν τον τρόπο, εξασφαλίζεται η συνέπεια των δεδομένων, ωστόσο με κόστος χαμηλότερης ταχύτητας για τις ενέργειες στον δίσκο.

- **no_subtree_check:** Δεν πραγματοποιείται ο έλεγχος subtree, δηλαδή αν το αρχείο είναι διαθέσιμο στο εξαγόμενο tree για κάθε αίτημα.

- **no_root_squash:** Από προεπιλογή, το NFS μετατρέπει απομακρυσμένα αιτήματα από έναν χρήστη με δικαιώματα διαχειριστή (root) σε έναν που δεν έχει root δικαιώματα. Με την επιλογή αυτή, απενεργοποιείται η συμπεριφορά αυτή.

Φυσικά, θα πρέπει να προστεθούν και οι κατάλληλοι κανόνες στο τείχος προστασίας (firewall) για την πρόσβαση των NFS clients στο δίσκο:

```
sudo ufw allow from 147.102.40.42 to any port nfs
sudo ufw allow from 147.102.40.49 to any port nfs
```

Για τη ρύθμιση του NFS στους clients, οι οποίοι είναι οι Moodle εξυπηρετητές, εγκαθιστούμε το πακέτο που αναφέρθηκε προηγουμένως. Στη συνέχεια, δημιουργούμε το φάκελο με το επιθυμητό όνομα.

```
sudo mkdir -p /home/theo/nfs_moodledata
```

Τέλος, συνδέουμε τον παραπάνω φάκελο με τον NFS host εκτελώντας:

```
sudo mount 147.102.40.52:/var/nfs/host_moodledata home/theo/nfs_moodledata/
```

Δημιουργία επιπλέον κοινόχρηστων αρχείων

Για την ορθή λειτουργία της συστάδας, είναι απαραίτητη η παρουσία των 4 παρακάτω κοινόχρηστων αρχείων μέσα στον κοινόχρηστο φάκελο nfs_moodledata:

- moodledata: το οποίο το μεταφέρουμε στον NFS host από το Moodle μέσω SCP.
- clustercache
- clustertmp
- backuptmp

Εκ των οποίων τα τρία τελευταία δημιουργούνται στον NFS host καθώς επιθυμούμε να είναι και αυτά κοινόχρηστα (μέσα στο directory /var/nfs/host_moodledata).

Ωστόσο, απαιτείται και η παρουσία δύο τοπικών αρχείων, επομένως δημιουργούμε τα localcache και localtmp τα οποία δεν είναι κοινόχρηστα.
Επιπλέον, όλα τα παραπάνω θα πρέπει να οριστούν και στο αρχείο παραμετροποίησης του Moodle (config.php) προσθέτοντας τις παρακάτω γραμμές:

```
$CFG->tempdir = '/home/theo/nfs_moodledata/clustertmp';
$CFG->cachedir = '/home/theo/nfs_moodledata/clustercache';
$CFG->localcachedir = '/home/theo/localcache';
$CFG->localrequestdir = '/home/theo/localtmp';
$CFG->backuptempdir = '/home/theo/nfs_moodledata/backuptmp';
```

Τέλος, απαιτείται και ο σωστός ορισμός του moodledata, ο οποίος γίνεται με την ακόλουθη προσθήκη.

```
$CFG->dataroot  = '/home/theo/nfs_moodledata/moodledata';
```

8.10: Εγκατάσταση Unison

Για την εγκατάσταση του Unison εφαρμόζουμε τα παρακάτω:

```
sudo apt-get install unison unison-all
```

Στη συνέχεια, επειδή το Unison λειτουργεί μέσω του SSH, πρέπει να δημιουργηθούν SSH κλειδιά για την αυθεντικοποίηση χωρίς κωδικό.

Για το σκοπό αυτό, στο Moodle1 εκτελούμε:

```
ssh-keygen -o -a 100 -t ed25519 -f ~/.ssh/moodle1key
```

Στη συνέχεια, αντιγράφουμε το δημόσιο κλειδί moolde1key.pub στο Moodle2 στον φάκελο /root/.ssh/authorized_keys, καθώς το Unison λειτουργεί επιτυχώς με την εκτέλεσή του ως root. Αντίστοιχα, πραγματοποιούμε την διαδικασία αυτή και για το Moodle2.

Κατόπιν, δημιουργούμε όπως και στην περίπτωση του NFS το κοινόχρηστο directory με όνομα unison_moodledata και μέσα σε αυτόν τα: clustercache', clustertmp και backuptmp. Φροντίζουμε επίσης μέσα στο κοινόχρηστο directory να μεταφέρουμε και το moodledata που περιέχει τα δεδομένα του συστήματος.

Στη συνέχεια, προσθέτουμε τις κατάλληλες γραμμές στο αρχείο **./unison/default.prf** και στους δύο Moodle εξυπηρετητές.

```
# Unison preferences file
root=/home/theo/unison_moodledata
root=ssh://root@moodle_ip//home/theo/unison_moodledata
sshargs=  -i /home/theo/.ssh/unisonmoodlekey
auto=true
batch=true
confirmbigdel=true
fastcheck=true
group=true
owner=true
```

Στα παραπάνω, όπου moodle_ip ορίζεται διεύθυνση IP του αντίστοιχου Moodle και ως unisonmoodlekey το αντίστοιχο κλειδί.

Επιπλέον, δημιουργούμε ένα script με όνομα myunison με τις επιθυμητές ρυθμίσεις:

```
#!/bin/bash

if ! (ps aux | grep unison | grep -v grep | grep -v unison.log | grep -v myunison ); then

  unison default

else

  date >> /home/theo/.unison/unison.log

fi
```

Τέλος, ορίζουμε ένα Cron job για τη συχνότητα με την οποία θα συγχρονίζονται τα αρχεία:

Συγκεκριμένα, στο Moodle1 προσθέτουμε τα εξής:

```
sudo crontab -e

* * * * * (sleep 15; bash /home/theo/.unison/myunison)   >/dev/null 2>&1
```

Ομοίως, στο Moodle2:

```
* * * * * (sleep 45; bash /home/theo/.unison/myunison)   >/dev/null 2>&1
```

Με αυτόν τον τρόπο, το Unison εκτελείται κάθε λεπτό, αλλά με μία καθυστέρηση 15 δευτερολέπτων στο Moodle1 και 45 δευτερολέπτων στο Moodle2. Η επιλογή αυτών των καθυστερήσεων γίνεται προκειμένου να είναι βέβαιο πως έχουν ολοκληρωθεί οι αντιγραφές του πρώτου μηχανήματος προτού εκκινήσει το δεύτερο.

Τέλος, θα πρέπει να αναφερθεί πως είναι αναγκαία η δημιουργία δύο χρηστών στη βάση δεδομένων που θα επιτρέπουν την πρόσβαση σε αυτήν από τους Moodle εξυπηρετητή. Επιπλέον, απαιτείται και η κατάλληλη δήλωση του κάθε χρήστη στο αρχείο παραμετροποίησης του εκάστοτε Moodle με τον τρόπο που παρουσιάστηκε στο κεφάλαιο 8.2.

Ρύθμιση Nginx ως load balancer

Αφού αναλύθηκε ο τρόπος υλοποίησης της συστάδας των Moodle, στο παρόν κεφάλαιο παρουσιάζονται οι ρυθμίσεις που επιτρέπουν στον Nginx να λειτουργεί ως load balancer. Ο Nginx ως load balancer αναλαμβάνει τον καταμερισμό του φόρτου εργασίας στους δύο Moodle εξυπηρετητές. Για το σκοπό αυτό, υπάρχουν διάφοροι αλγόριθμοι που μπορούν να χρησιμοποιηθούν.

Οι δύο αλγόριθμοι που χρησιμοποιήθηκαν είναι ο Round-robin και η μέθοδος hashing αναφορικά με την IP και θύρα του χρήστη. Για τον ορισμό του Nginx ως load balancer τροποποιούμε κατάλληλα το αρχείο παραμετροποίησης **/etc/nginx/sites-available/reverse-proxy.conf**, προσθέτοντας τις παρακάτω αλλαγές στο upstream module:

```
upstream backendmoodles {
#hash $binary_remote_addr$remote_port consistent;
keepalive 4;
server 147.102.40.42:80 max_fails=2 fail_timeout=20s;
server 147.102.40.49:80 max_fails=2 fail_timeout=20s;
}
```

Με το παραπάνω upstream module, ορίζουμε τη μέθοδο με Round-robin ως load balancing. Σε περίπτωση που επιθυμείται η επιλογή της μεθόδου με το hashing, αφαιρούμε από το σχόλιο την αντίστοιχη γραμμή.

Επίλογος

Σκοπός της παρούσας διπλωματικής εργασίας, αποτελεί η αξιολόγηση της επίδοσης του συστήματος διαχείρισης μαθημάτων Moodle. Για το σκοπό αυτό, χρησιμοποιήθηκε το Apache JMeter προκειμένου να διενεργηθούν τα τεστ της δοκιμής φορτίου που προσομοιώνουν τη συμπεριφορά των χρηστών του συστήματος. Οι αρχιτεκτονικές που σχεδιάστηκαν με σκοπό τη σύγκριση των αποτελεσμάτων και την εύρεση της βέλτιστης υλοποίησης είναι οι εξής:

- Moodle ως αυτοδύναμος εξυπηρετητής (Apache, MySQL και PHP σε έναν εξυπηρετητή)
- Moodle με τη βάση δεδομένων σε έναν αυτοδύναμο MySQL εξυπηρετητή
- Συστάδα δύο Moodle εξυπηρετητών με τη χρήση των λογισμικών NFS/Unison για το συγχρονισμό του κοινόχρηστου φακέλου moodledata, καθώς και του Redis για την ύπαρξη κρυφής μνήμης (cache).

Όπως γίνεται αντιληπτό από τα προηγούμενα κεφάλαια, το Moodle είναι ικανό να διαχειριστεί έναν υψηλό αριθμό ταυτόχρονων χρηστών. Όπως αποδείχθηκε και από τα αποτελέσματα των δοκιμών, η υπολογιστική ισχύς του αυτοδύναμου Moodle εξυπηρετητή φτάνει σε κορεσμό όσο αυξάνεται το πλήθος των χρηστών. Επομένως για την ύπαρξη υψηλού αριθμού ταυτόχρονων χρηστών που χρησιμοποιούν την πλατφόρμα είναι αναγκαία η επιλογή της αρχιτεκτονικής μιας συστάδας από Moodle εξυπηρετητές.

Μελλοντικές επεκτάσεις

Μια επέκταση και βελτιστοποίηση, ενδέχεται να επιφέρει η αντικατάσταση του Apache με τον Nginx. Παρόλο που στην υλοποίηση του συστήματος μελετήθηκε η βελτιστοποίηση του Apache με τη σωστή επιλογή του mpm_worker, ενδέχεται η αντικατάστασή του με τον Nginx στο ρόλο του εξυπηρετητή ιστού να προσφέρει καλύτερες επιδόσεις. Αυτό οφείλεται στον τρόπο λειτουργίας του Nginx σε σχέση με τον Apache αναφορικά με τον τρόπο διαχείρισης των HTTP αιτημάτων. Ο Nginx βασίζεται σε έναν asynchronous event-driven αλγόριθμο διαχείρισης αιτημάτων. Κάθε worker process μπορεί να διαχειρίζεται χιλιάδες αιτήματα, καθώς ο τρόπος λειτουργίας του βασίζεται στον συνεχή έλεγχο και την ασύγχρονη εκτέλεση του, μέσω ενός event loop [27]. Το πλεονέκτημα χρήσης του Nginx, το οποίο παρατηρήθηκε και από τα εξαγόμενα τεστ είναι η πολύ μικρή κατανάλωση μνήμης αλλά και επεξεργαστή.
Μια επέκταση με σκοπό την αύξηση της επίδοσης του Moodle ενδέχεται να αποτελεί η βελτιστοποίηση του κώδικα PHP. Είναι πιθανό ο ίδιος ο κώδικάς του να μην είναι βελτιστοποιημένος και να επιφέρει βελτιστοποιήσεις.

Βιβλιογραφία

[1] Wikipedia, “LAMP (software bundle)”. [Online].
Available: https://en.wikipedia.org/wiki/LAMP_(software_bundle)
[Accessed October 2022].

[2] Moodle, “About Moodle”. [Online].
Available: https://docs.moodle.org/400/en/About_Moodle
[Accessed October 2022].

[3] Moodle, “Test course generator”. [Online].
Available: https://docs.moodle.org/39/en/Test_course_generator
[Accessed October 2022].

[4] Moodle, “Installing plugins”. [Online].
Available: https://docs.moodle.org/311/en/Installing_plugins
[Accessed October 2022].

[5] Moodle, “Moodle Benchmark”. [Online].
Available: https://moodle.org/plugins/report_benchmark
[Accessed October 2022].

[6] Wikipedia, “Load testing”. [Online].
Available: https://en.wikipedia.org/wiki/Load_testing
[Accessed October 2022].

[7] Wikipedia, “Stress testing”. [Online].
Available: https://en.wikipedia.org/wiki/Stress_testing
[Accessed October 2022].

[8] BlazeMeter, “BlazeMeter”. [Online].
Available: https://www.blazemeter.com/product/blazemeter
[Accessed October 2022].

[9] LoadRunner, “LoadRunner Professional”. [Online].
Available: https://www.microfocus.com/en-us/products/loadrunner-professional/overview
[Accessed October 2022].

[10] WebLOAD, ”WebLOAD – Performance and Load Testing”. [Online].
Available: https://www.radview.com/webload-overview/

[11] Tsung, ”Download Tsung”. [Online].
Available: http://tsung.erlang-projects.org/

[12] Apache JMeter, “Overview”. [Online].
Available: https://jmeter.apache.org/
[Accessed October 2022].

[13] Apache JMeter, “Building a Test Plan”. [Online].
Available: https://jmeter.apache.org/usermanual/build-test-plan.html
[Accessed October 2022].

[14] Apache JMeter, “Functions, __time”. [Online].
Available: https://jmeter.apache.org/usermanual/functions.html#__time
[Accessed October 2022].

[15] Apache JMeter, “Thread Group”. [Online].
Available: https://jmeter.apache.org/usermanual/test_plan.html#thread_group
[Accessed October 2022].

[16] Apache JMeter, “HTTP Cookie Manager”. [Online].
Available: https://jmeter.apache.org/usermanual/component
[Accessed October 2022].

[17] Apache JMeter, “HTTP Request”. [Online].
Available: https://jmeter.apache.org/usermanual/component_reference.html#HTTP_Request
[Accessed October 2022].

[18] Apache JMeter, ”Users Manual: Properties Reference”. [Online].
Available:https://jmeter.apache.org/usermanual/properties_reference.html#httpclient_common_properties
[Accessed October 2022].

[19] Moodle, “Login token”. [Online].
Available: https://docs.moodle.org/dev/Login_token
[Accessed October 2022].

[20] Apache JMeter, “Regular Expression Extractor”. [Online].
Available:https://jmeter.apache.org/usermanual/component_reference.html#
[Accessed October 2022].

[21] Moodle, “Security: Cross-site request forgery”. [Online].
Available: https://docs.moodle.org/dev/Security:Cross-site_request_forgery
[Accessed October 2022].

[22] Apache JMeter, “Timers”. [Online].
Available: https://jmeter.apache.org/usermanual/component_reference.html#timers
[Accessed October 2022].

[23] Apache JMeter, “Precise Throughput Timer”. [Online].
Available: https://jmeter.apache.org/usermanual/component_reference.html#
[Accessed October 2022].

[24] Apache JMeter, “Load Test running”. [Online].
Available: https://jmeter.apache.org/usermanual/get-started.html#load_test_running
[Accessed October 2022].

[25] Wikipedia, “Apdex”. [Online].
Available: https://en.wikipedia.org/wiki/Apdex
[Accessed October 2022].

[26] Apache, “MaxRequestWorkers Directive”. [Online].
Available: https://httpd.apache.org/docs/2.4/mod/mpm_common.html#maxrequestworkers
[Accessed October 2022].

[27] DigitalOcean, “Apache vs Nginx: Practical Considerations”. [Online]
Available: https://www.digitalocean.com/community/tutorials/apache-vs-nginx-practical-considerations
[Accessed October 2022].

[28] Wikipedia, “Reverse proxy”. [Online].
Available: https://en.wikipedia.org/wiki/Reverse_proxy
[Accessed October 2022].

[29] Moodle, “Caching”. [Online].
Available: https://docs.moodle.org/400/en/Caching
[Accessed October 2022].

[30] Moodle, “Step-by-step Installation Guide for Ubuntu”. [Online].
Available: https://docs.moodle.org/310/en/Step-by-step_Installation_Guide_for_Ubuntu
[Accessed October 2022].

[31] Moodle, “Apache”. [Online].
Available: https://docs.moodle.org/400/en/Apache
[Accessed October 2022].

[32] Cloudways, “PHP-FPM Cuts Web App Loading Times by 300%”. [Online].
Available: https://www.cloudways.com/blog/php-fpm-on-cloud/
[Accessed October 2022].

[33] Moodle, “OPcache Configuration”. [Online].
Available: https://docs.moodle.org/310/en/OPcache#Configuration
[Accessed October 2022].

[34] Moodle, “Cron”. [Online].
Available: https://docs.moodle.org/311/en/Cron
[Accessed October 2022].

[35] Moodle, “Environment - max input vars”. [Online].
Available: https://docs.moodle.org/311/en/Environment_-_max_input_vars
[Accessed October 2022].

[36] Apache JMeter, ”Download Apache JMeter”. [Online].
https://jmeter.apache.org/download_jmeter.cgi
[Accessed October 2022].

[37] Apache JMeter, “Running JMeter”. [Online].
Available: https://jmeter.apache.org/usermanual/get-started.html#running
[Accessed October 2022].

[38] Moodle, “Moodle migration”. [Online].
Available: https://docs.moodle.org/400/en/Moodle_migration
[Accessed October 2022].

[39] Certbot, “About certbot”. [Online].
Available: https://certbot.eff.org/pages/about
[Accessed October 2022].