ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΗΣ ΙΣΤΟΥ ΓΙΑ ΤΗ ΔΙΑΧΕΙΡΙΣΗ ΤΩΝ ΧΡΗΣΤΩΝ ΤΟΥ ΕΡΓΑΣΤΗΡΙΟΥ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΓΕΩΡΓΙΟΣ Α. ΚΑΖΕΛΙΔΗΣ

Επιβλέπων: Νικόλαος Παπασπύρου
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΗΣ ΙΣΤΟΥ ΓΙΑ ΤΗ ΔΙΑΧΕΙΡΙΣΗ ΤΩΝ ΧΡΗΣΤΩΝ ΤΟΥ ΕΡΓΑΣΤΗΡΙΟΥ ΤΕΧΝΟΛΟΓΙΑΣ ΛΟΓΙΣΜΙΚΟΥ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΓΕΩΡΓΙΟΣ Α. ΚΑΖΕΛΙΔΗΣ

Επιβλέπων: Νικόλαος Παπασπύρου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 21η Οκτωβρίου 2019.

.....	.....	.....
Νικόλαος Παπασπύρου Καθηγητής Ε.Μ.Π.	Αριστείδης Παγουρτζής Αν. Καθηγητής Ε.Μ.Π.	Γεώργιος Στάμου Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2019

.....

Γεώργιος Α. Καζελίδης
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στα πλαίσια της διδασκαλίας αρκετών προπτυχιακών και μεταπτυχιακών μαθημάτων προγραμματισμού υπολογιστών, το Εργαστήριο Τεχνολογίας Λογισμικού, SoftLab, του Εθνικού Μετσόβιου Πολυτεχνείου παρέχει πρόσβαση σε διάφορες διαδικτυακές πλατφόρμες (Novice, Grader, Moodle και PLgrader), στις οποίες οι φοιτητές της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών μπορούν, μεταξύ άλλων, να επεξεργάζονται ή να παραδίδουν τις προγραμματιστικές εργασίες τους, αφού πρώτα εισέλθουν με κατάλληλα διαπιστευτήρια.

Σκοπός αυτής της διπλωματικής εργασίας είναι η ανάπτυξη της εφαρμογής ιστού SLUB (SoftLab UserBase) που επιτελεί την κεντρική διαχείριση και συσχέτιση λογαριασμών για τους χρήστες των πλατφορμών του SoftLab μέσω της δικής της βάσης δεδομένων, usermergeDB, ώστε να διευκολύνει την ταυτοποίηση χρηστών από τους υπεύθυνους του Εργαστηρίου, τόσο σε επίπεδο μίας όσο και σε επίπεδο πολλαπλών πλατφορμών. Το SLUB, του οποίου ο πηγαίος κώδικας και η τεκμηρίωση διατίθενται στη διεύθυνση https://git.softlab.ntua.gr/giorgkazelidis/userbase, αναπτύχθηκε σε στοίβα λογισμικού LAMP (Linux-Apache-MySQL-Python) μέσω του πλαισίου ιστού Django και επιστρατεύτηκε στη διεύθυνση http://slub.softlab.ntua.gr/, όπου ζητείται από τον τρέχοντα χρήστη να εισέλθει είτε ως διαχειριστής είτε ως απλός χρήστης της εφαρμογής, δηλαδή χρήστης μίας πλατφόρμας του Εργαστηρίου.

Από τη μία, ένας διαχειριστής του SLUB μπορεί να προσθέτει διαπιστευτήρια απλών χρηστών για οποιαδήποτε πλατφόρμα του SoftLab στη βάση usermergeDB μέσω κατάλληλα διαμορφωμένων αρχείων TXT, ενώ έχει επίσης τη δυνατότητα να δημιουργεί λεπτομερείς CSV αναφορές απλών χρηστών για οποιαδήποτε από τις προαναφερθείσες πλατφόρμες, χρησιμοποιώντας ως κριτήριο την ημερομηνία τελευταίας εισόδου στην εφαρμογή με διαπιστευτήρια της εκάστοτε επιλεγμένης πλατφόρμας. Από την άλλη, ένας απλός χρήστης του SLUB μπορεί όχι μόνο να επεξεργάζεται το προφίλ του, {όνομα, επώνυμο, αριθμός μητρώου στη Σχολή ΗΜΜΥ (αν υπάρχει), ηλεκτρονικό ταχυδρομείο}, αλλά και να το ανακτά από τη βάση usermergeDB, χρησιμοποιώντας ως κριτήρια τον αριθμό μητρώου και το ηλεκτρονικό ταχυδρομείο, σε περίπτωση που τα διαπιστευτήρια της πλατφόρμας εισόδου αξιοποιούνται για πρώτη ή αλλάζονται για ν-οστή φορά στην εφαρμογή.

Τέλος, ενεργοποιήθηκε στη διεύθυνση http://slub.softlab.ntua.gr/django-admin/ ο ιστότοπος διαχείρισης του Django, ο οποίος απευθύνεται αποκλειστικά σε χρήστες του ομώνυμου πλαισίου και δρα ως διαχειριστική διεπαφή για τη δημιουργία, την ανάγνωση, την ενημέρωση ή/και τη διαγραφή εγγραφών των πινάκων της βάσης usermergeDB.

Λέξεις κλειδιά

Αρχιτεκτονική πελάτη-διακομιστή, συνεδρίες HTTP, διαχείριση χρηστών, εφαρμογή ιστού SLUB (SoftLab UserBase), διαδικτυακός προγραμματισμός εμπρόσθιου τμήματος (HTML-CSS-JavaScript), στοίβα λογισμικού LAMP (Linux-Apache-MySQL-Python), πλαίσιο ιστού Django, αρχιτεκτονικό μόρφημα MTV, προδιαγραφή WSGI, ιστότοπος διαχείρισης του Django.

Abstract

In the context of teaching several undergraduate and postgraduate computer programming courses, the Software Engineering Laboratory, SoftLab, of the National Technical University of Athens provides access to various online platforms (Novice, Grader, Moodle and PLgrader), in which the students of the Electrical and Computer Engineering School can, inter alia, edit or submit their programming tasks, after first logging in with appropriate credentials.

The purpose of this diploma thesis is to develop the SLUB (SoftLab UserBase) web application that performs the central management and association of accounts for the users of the SoftLab platforms through its own database, usermergeDB, in order to facilitate user identification by the Laboratory managers, both at a single- and at a multiple-platform level. SLUB, whose source code and documentation are available at https://git.softlab.ntua.gr/giorgkazelidis/userbase, was developed in the LAMP (Linux-Apache-MySQL-Python) software stack via the Django web framework and was deployed at http://slub.softlab.ntua.gr/, where the current user is asked to log in either as an administrator or as a simple user of the application, i.e. user of a Laboratory platform.

On the one hand, an administrator of SLUB can add credentials of simple users for any SoftLab platform to usermergeDB through properly configured TXT files, while also being able to create detailed CSV reports of simple users for any of the aforementioned platforms, based on the date of last login to the application with credentials of the platform that is selected each time. On the other hand, a simple user of SLUB can not only edit his/her profile, {first name, last name, ECE ID (if any), email}, but also recover it from usermergeDB, by using the ECE ID and email as criteria, in case the credentials of the login platform are utilized for the first time or changed for the n-th one in the application.

Finally, activated at http://slub.softlab.ntua.gr/django-admin/ lies the Django admin site, which is targeted exclusively at users of the homonymous framework and acts as an administrative interface for creating, reading, updating or/and deleting records of the usermergeDB tables.

Keywords

Client-server architecture, HTTP sessions, user management, SLUB (SoftLab UserBase) web application, front-end web development (HTML-CSS-JavaScript), LAMP (Linux-Apache-MySQL-Python) software stack, Django web framework, MTV architectural pattern, WSGI specification, Django admin site.

Ευχαριστίες

Πριν από περίπου ενάμιση χρόνο ολοκλήρωσα τη μονοετή πρακτική άσκησή μου στη Nokia Networks Hellas, όπου ασχολήθηκα κυρίως με την επέκταση των χαρακτηριστικών μίας εφαρμογής ιστού σε στοίβα λογισμικού LTMJ (Linux-Tomcat-MySQL-Java) μέσω του ολοκληρωμένου περιβάλλοντος ανάπτυξης STS (Spring Tool Suite). Κατά την ενασχόλησή μου με αυτή την εφαρμογή, δυσκολεύτηκα μεν αρκετά, αλλά συνάμα εντυπωσιάστηκα από τις απεριόριστες δυνατότητες που προσφέρουν οι σύγχρονες τεχνολογίες λογισμικού τόσο στους μηχανικούς υπολογιστών, οι οποίοι αναπτύσσουν εφαρμογές ιστού, όσο και στους τελικούς χρήστες, οι οποίοι τις χρησιμοποιούν. Η περιέργεια να μάθω πώς είναι να δημιουργείς μία τέτοια εφαρμογή από την αρχή και η επιθυμία να εμπλουτίσω τις γνώσεις μου με τη χρήση του πλαισίου ιστού Django, το οποίο είναι αφενός προσιτό σε αρχάριους στον διαδικτυακό προγραμματισμό και αφετέρου αρκετά δημοφιλές στην αγορά εργασίας, αποτέλεσαν το εφαλτήριο αυτής της διπλωματικής.

Θα ήθελα να ευχαριστήσω θερμά τον κ. Νικόλαο Παπασπύρου που μου έδωσε την ευκαιρία να αναπτύξω μία τέτοια εφαρμογή για τη διαχείριση χρηστών του Εργαστηρίου Τεχνολογίας Λογισμικού και που συνέβαλε καθοριστικά στην ομαλή περάτωσή της, επιδεικνύοντας πάντα προθυμία και διαλλακτικότητα. Θα ήθελα επίσης να ευχαριστήσω τους συναδέλφους στη Nokia Networks Hellas που, μεταξύ άλλων, με βοήθησαν να αποκτήσω γνώσεις και εμπειρία πάνω σε θέματα ανάπτυξης εφαρμογών ιστού. Πάνω από όλα, όμως, θα ήθελα να ευχαριστήσω την οικογένειά μου που είναι πάντοτε δίπλα μου και με υποστηρίζει σε ό,τι και αν καταπιαστώ.

Το πόνημα αυτό είναι ιδιαίτερα αφιερωμένο στους παππούδες μου, Γεωργία και Ιωάννη, και στους γονείς μου, Δέσποινα και Απόστολο.

Γεώργιος Α. Καζελίδης,
Αθήνα, 21 Οκτωβρίου 2019

Περιεχόμενα

Κατάλογος σχημάτων

1 Εισαγωγή

1.1 Σκοπός

Στα πλαίσια της διδασκαλίας αρκετών προπτυχιακών και μεταπτυχιακών μαθημάτων προγραμματισμού υπολογιστών, το Εργαστήριο Τεχνολογίας Λογισμικού, SoftLab, του Εθνικού Μετσόβιου Πολυτεχνείου παρέχει πρόσβαση σε διάφορες διαδικτυακές πλατφόρμες (Novice, Grader, Moodle και PLgrader), στις οποίες οι φοιτητές της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών μπορούν, μεταξύ άλλων, να επεξεργάζονται ή να παραδίδουν τις προγραμματιστικές εργασίες τους, αφού πρώτα εισέλθουν με κατάλληλα διαπιστευτήρια.

Κατά κανόνα, κάθε πλατφόρμα του SoftLab αναφέρεται σε διαφορετικά μαθήματα προγραμματισμού, ενώ διαθέτει παράλληλα διαφορετική βάση δεδομένων και διαφορετικό μηχανισμό δημιουργίας/αλλαγής διαπιστευτηρίων χρηστών σε σχέση με τις υπόλοιπες. Ως εκ τούτου, είναι πιθανόν ένας χρήστης να έχει διαφορετικά διαπιστευτήρια ανά πλατφόρμα, ενώ ενδέχεται ακόμα και να τα αλλάξει σε περίπτωση που εκείνα χαθούν/ξεχαστούν· η υψηλή διασπορά και οι συχνές αλλαγές των διαπιστευτηρίων είναι ικανές να δυσχεράνουν την ταυτοποίηση χρηστών από τους υπεύθυνους του Εργαστηρίου, τόσο σε επίπεδο μίας όσο και σε επίπεδο πολλαπλών πλατφορμών.

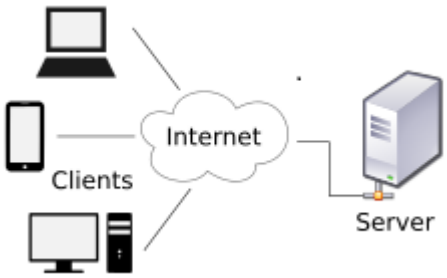
Σκοπός αυτής της διπλωματικής εργασίας είναι η ανάπτυξη της εφαρμογής ιστού SLUB (SoftLab UserBase) που επιτελεί την κεντρική διαχείριση και συσχέτιση λογαριασμών για τους χρήστες των πλατφορμών του SoftLab μέσω της δικής της βάσης δεδομένων, usermergeDB, ώστε να εξομαλύνει το προαναφερθέν πρόβλημα. Το SLUB, του οποίου ο πηγαίος κώδικας και η τεκμηρίωση διατίθενται στη διεύθυνση https://git.softlab.ntua.gr/giorgkazelidis/userbase, αναπτύχθηκε σε στοίβα λογισμικού LAMP (Linux-Apache-MySQL-Python) μέσω του πλαισίου ιστού Django και επιστρατεύτηκε στη διεύθυνση http://slub.softlab.ntua.gr/, όπου ζητείται από τον τρέχοντα χρήστη να εισέλθει είτε ως διαχειριστής είτε ως απλός χρήστης της εφαρμογής, δηλαδή χρήστης μίας πλατφόρμας του Εργαστηρίου.

1.2 Αρχιτεκτονική πελάτη-διακομιστή[1]

1.2.1 Επισκόπηση

Το μοντέλο πελάτη-διακομιστή (client-server model) είναι μία δομή κατανεμημένης εφαρμογής (distributed application structure) που διαμοιράζει καθήκοντα (tasks) ή φορτία εργασίας (workloads) μεταξύ των παρόχων ενός πόρου ή υπηρεσίας, που ονομάζονται διακομιστές (servers), και των αιτούντων τον πόρο ή την υπηρεσία, που ονομάζονται πελάτες (clients). Συχνά οι πελάτες και οι διακομιστές επικοινωνούν μέσω ενός δικτύου υπολογιστών σε ξεχωριστό υλισμικό (hardware) όπως φαίνεται στο Σχήμα 1.1, αλλά τόσο ο πελάτης όσο και ο διακομιστής μπορούν να διαμένουν στο ίδιο σύστημα. Ένας υπολογιστής-διακομιστής (server host) εκτελεί ένα ή περισσότερα προγράμματα διακομιστή που μοιράζονται τους πόρους τους με τους πελάτες. Ένας πελάτης δεν μοιράζεται κανέναν από τους πόρους του, αλλά αιτείται τη λειτουργία ενός περιεχομένου ή μίας υπηρεσίας διακομιστή. Συνεπώς, οι πελάτες ξεκινούν συνεδρίες επικοινωνίας (communication sessions) με διακομιστές που περιμένουν εισερχόμενα

αιτήματα. Παραδείγματα εφαρμογών υπολογιστή που χρησιμοποιούν το μοντέλο πελάτη-διακομιστή είναι το ηλεκτρονικό ταχυδρομείο (email), η δικτυακή εκτύπωση (network printing) και ο Παγκόσμιος Ιστός (World Wide Web).



Σχήμα 1.1: Αναπαράσταση της επικοινωνίας πελατών-διακομιστή μέσω του Διαδικτύου

1.2.2 Ρόλος πελάτη και διακομιστή

Οι διακομιστές ταξινομούνται από τις υπηρεσίες που παρέχουν. Για παράδειγμα, ένας διακομιστής ιστού (web server) παρέχει ιστοσελίδες (web pages) και ένας διακομιστής αρχείων (file server) παρέχει αρχεία υπολογιστών. Ένας κοινός πόρος μπορεί να είναι οποιοδήποτε από τα λογισμικά και ηλεκτρονικά στοιχεία του υπολογιστή διακομιστή, από προγράμματα και δεδομένα έως επεξεργαστές και συσκευές αποθήκευσης. Η κοινή χρήση πόρων ενός διακομιστή αποτελεί υπηρεσία.

Το αν ένας υπολογιστής είναι πελάτης, διακομιστής ή και τα δύο εξαρτάται από τη φύση της εφαρμογής που απαιτεί τις λειτουργίες της υπηρεσίας. Για παράδειγμα, ένας μόνο υπολογιστής μπορεί να εκτελεί συγχρόνως λογισμικό διακομιστή ιστού και διακομιστή αρχείων για να παρέχει διαφορετικά δεδομένα σε πελάτες που κάνουν διαφορετικά είδη αιτημάτων. Το λογισμικό πελάτη μπορεί επίσης να επικοινωνεί με λογισμικό διακομιστή εντός του ίδιου υπολογιστή. Η επικοινωνία μεταξύ διακομιστών, π.χ. για τον συγχρονισμό δεδομένων, αποκαλείται μερικές φορές επικοινωνία διακομιστή με διακομιστή (server-to-server communication).

1.2.3 Επικοινωνία πελάτη και διακομιστή

Γενικά, μία υπηρεσία είναι μία αφαιρετική αναπαράσταση των υπολογιστικών πόρων και ένας πελάτης δεν χρειάζεται να ασχολείται με τον τρόπο με τον οποίο ο διακομιστής λειτουργεί κατά την εκπλήρωση του αιτήματος και την παράδοση της απόκρισης. Ο πελάτης πρέπει μόνο να κατανοήσει την απόκριση με βάση το γνωστό πρωτόκολλο εφαρμογής, δηλαδή το περιεχόμενο και τη μορφοποίηση των δεδομένων για την ζητούμενη υπηρεσία.

Οι πελάτες και οι διακομιστές ανταλλάσσουν μηνύματα σε ένα μοτίβο μηνυμάτων αιτήματος-απόκρισης (request-response messaging pattern). Ο πελάτης στέλνει ένα αίτημα και ο διακομιστής επιστρέφει μία απόκριση. Αυτή η ανταλλαγή μηνυμάτων είναι ένα παράδειγμα επικοινωνίας μεταξύ διαδικασιών (inter-process communication). Για να επικοινωνούν, οι υπολογιστές πρέπει να έχουν μία κοινή

γλώσσα και πρέπει να ακολουθούν κανόνες, έτσι ώστε τόσο ο πελάτης όσο και ο διακομιστής να γνωρίζουν τι να περιμένουν. Η γλώσσα και οι κανόνες επικοινωνίας ορίζονται σε ένα πρωτόκολλο επικοινωνιών (communications protocol). Όλα τα πρωτόκολλα πελάτη-διακομιστή λειτουργούν στο στρώμα εφαρμογής της στοίβας TCP/IP (application layer of the TCP/IP - Transmission Control Protocol / Internet Protocol - stack). Το πρωτόκολλο στρώματος εφαρμογής ορίζει τα βασικά μοτίβα του διαλόγου. Για να επισημοποιήσει ακόμη περισσότερο την ανταλλαγή δεδομένων, ο διακομιστής μπορεί να υλοποιήσει μία διεπαφή προγραμματισμού εφαρμογών (Application Programming Interface - API). Η διεπαφή αυτή είναι ένα στρώμα αφαίρεσης για την πρόσβαση σε μία υπηρεσία. Περιορίζοντας την επικοινωνία σε μία συγκεκριμένη μορφή περιεχομένου (content format), διευκολύνει την ανάλυση (parsing). Αντιμετωπίζοντας την πρόσβαση αφαιρετικά, διευκολύνει την ανταλλαγή δεδομένων μεταξύ πλατφορμών.

Ένας διακομιστής ενδέχεται να λάβει αιτήματα από πολλούς διαφορετικούς πελάτες σε σύντομο χρονικό διάστημα, ενώ μπορεί να εκτελέσει μόνο έναν περιορισμένο αριθμό εργασιών (tasks) ανά πάσα στιγμή και βασίζεται σε ένα σύστημα χρονοδρομολόγησης (scheduling system) για να δώσει προτεραιότητα στα εισερχόμενα αιτήματα των πελατών που θα εξυπηρετήσει. Για να αποτραπεί η κατάχρηση και να μεγιστοποιηθεί η διαθεσιμότητα (availability/uptime), το λογισμικό διακομιστή ενδέχεται να περιορίσει τη διαθεσιμότητα σε πελάτες. Οι επιθέσεις άρνησης εξυπηρέτησης (Denial of Service, DoS, attacks) έχουν σχεδιαστεί για να εκμεταλλεύονται την υποχρέωση ενός διακομιστή να επεξεργάζεται αιτήματα υπερφορτώνοντας τον σε σύντομο χρονικό διάστημα με έναν υπερβολικά μεγάλο αριθμό αιτημάτων.

1.2.4 Συνεδρίες πελάτη-διακομιστή[2]

Οι περισσότερες συνεδρίες πελάτη-διακομιστή (client-server sessions) διατηρούνται από το στρώμα μεταφοράς της στοίβας TCP/IP (transport layer of the TCP/IP stack) και σε καθεμία αντιστοιχεί ακριβώς μία σύνδεση δικτύου (network connection). Ωστόσο, κάθε φάση συναλλαγής (transaction phase) μίας συνεδρίας ιστού/HTTP (web/HTTP session) δημιουργεί μία ξεχωριστή σύνδεση. Για να διατηρηθεί η συνέχεια της συνεδρίας (session continuity) μεταξύ των φάσεων, απαιτείται αναγνωριστικό συνεδρίας (session ID). Το αναγνωριστικό συνεδρίας είναι ενσωματωμένο (embedded) στους συνδέσμους (links) <A HREF> ή <FORM> των δυναμικών ιστοσελίδων έτσι ώστε να μεταφέρεται πίσω στο CGI (Common Gateway Interface) του διακομιστή. Το CGI χρησιμοποιεί έπειτα το αναγνωριστικό συνεδρίας για να εξασφαλίσει τη συνέχεια της συνεδρίας μεταξύ των φάσεων συναλλαγής. Ένα βασικό πλεονέκτημα της δημιουργίας μίας σύνδεσης ανά φάση (connection-per-phase) είναι ότι λειτουργεί καλά σε συνδέσεις χαμηλού εύρους ζώνης (low bandwidth connections), π.χ. μέσω διαποδιαμορφωτή (modem).

1.2.4.1 Διαχείριση συνεδριών διακομιστή ιστού

Το πρωτόκολλο μεταφοράς υπερκειμένου, HTTP (HyperText Transfer Protocol), είναι ακαταστατικό (stateless): ένας υπολογιστής-πελάτης (client computer) που εκτελεί έναν περιηγητή ιστού (web browser) πρέπει να εγκαθιδρύσει (establish) μία νέα σύνδεση/συνεδρία δικτύου με βάση το πρωτόκολλο ελέγχου μετάδοσης, TCP

(Transmission Control Protocol), στον διακομιστή ιστού για κάθε νέο αίτημα HTTP GET ή POST. Επομένως, ο διακομιστής δεν μπορεί να βασιστεί σε μία εγκαθιδρυμένη σύνδεση/συνεδρία TCP για περισσότερες από μία λειτουργίες (operations) HTTP GET ή POST. Η τεχνική με την οποία οι προγραμματιστές ιστού (web developers) «αναγκάζουν» το ακαταστατικό πρωτόκολλο HTTP να υποστηρίξει την κατάσταση συνεδρίας (session state) ονομάζεται διαχείριση συνεδριών (session management) και χρησιμοποιεί ένα cookie συνεδρίας, δηλαδή ένα HTTP cookie που περιέχει το αναγνωριστικό συνεδρίας.

Μετά την αυθεντικοποίηση (authentication) και είσοδο (login) του χρήστη του υπολογιστή-πελάτη στην εφαρμογή του διακομιστή, ο διακομιστής δημιουργεί και αποστέλλει στον πελάτη υπό μορφή cookie ένα μοναδικό αναγνωριστικό (unique identifier) για την τρέχουσα συνεδρία αλληλεπίδρασης. Ο πελάτης αποθηκεύει το cookie στη μνήμη ή στον δίσκο του. Με κάθε διαδοχικό αίτημα, ο πελάτης στέλνει το cookie πίσω στον διακομιστή και ο διακομιστής το χρησιμοποιεί για να «θυμηθεί» την κατάσταση της εφαρμογής για τον συγκεκριμένο πελάτη και να δημιουργήσει μία κατάλληλη απόκριση. Συνεπώς, η εφαρμογή του διακομιστή δεν χρειάζεται να ζητήσει εκ νέου το όνομα χρήστη (username) και τον αντίστοιχο κωδικό πρόσβασης (password) σε κανένα από τα επόμενα αιτήματα HTTP (GET ή POST) του πελάτη. Σε περίπτωση που η συνεδρία λήξει (expires), δηλαδή παρέλθει ένα προκαθορισμένο χρονικό διάστημα χωρίς δραστηριότητα χρήστη (user activity), η διαχείριση συνεδριών απαιτεί από τον χρήστη να επανεισέλθει (re-login).

Η χρήση cookie συνεδρίας συνεπάγεται ότι ο πελάτης μπορεί να διαχειριστεί μόνο το αναγνωριστικό συνεδρίας και ότι όλα τα δεδομένα συνεδρίας αποθηκεύονται στον διακομιστή (συνήθως σε μία βάση δεδομένων, στην οποία ο πελάτης δεν έχει άμεση πρόσβαση) και συσχετίζονται με το συγκεκριμένο αναγνωριστικό. Αποφεύγεται με αυτόν τον τρόπο οποιαδήποτε ανεπιθύμητη τροποποίηση των δεδομένων συνεδρίας από πλευράς πελάτη που μπορεί να προέλθει είτε τυχαία είτε κακόβουλα. Παραδείγματα ονομάτων που χρησιμοποιούν ορισμένες γλώσσες προγραμματισμού (programming languages) για το cookie συνεδρίας είναι τα JSESSIONID (JavaServer Pages - JSP), PHPSESSID (PHP: Hypertext Preprocessor - PHP), CGISESSID (CGI) και ASPSESSIONID (Active Server Pages - ASP).

1.3 Εφαρμογές ιστού[3]

1.3.1 Επισκόπηση

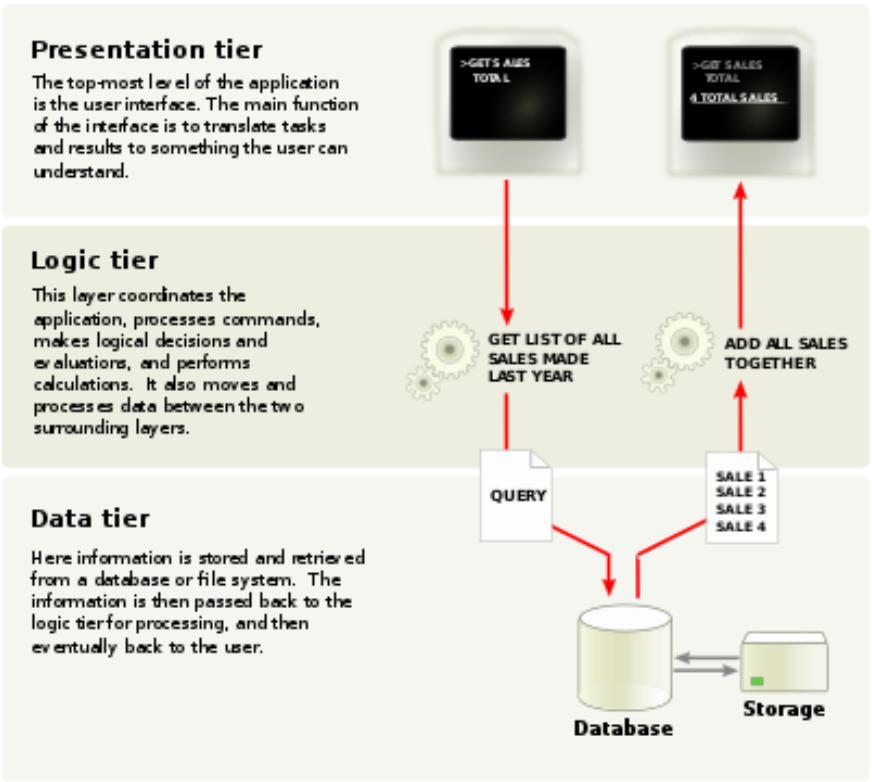
Στον διαδικτυακό προγραμματισμό (web development), εφαρμογή ιστού (web application ή web app) ονομάζεται ένα πρόγραμμα αρχιτεκτονικής πελάτη-διακομιστή που ο πελάτης, συμπεριλαμβανομένης της διεπαφής χρήστη (user interface) και της λογικής από πλευράς πελάτη (client-side logic), εκτελεί σε έναν περιηγητή ιστού (web browser). Στις ευρέως διαδεδομένες εφαρμογές ιστού περιλαμβάνονται το ταχυδρομείο ιστού (webmail) καθώς και οι λιανικές πωλήσεις και δημοπρασίες μέσω Διαδικτύου.

Οι εφαρμογές ιστού χρησιμοποιούν ιστοσελίδες που γράφονται σε τυπική μορφή, π.χ. HTML (HyperText Markup Language), CSS (Cascading Style Sheets) και JavaScript, και υποστηρίζονται από μία ποικιλία περιηγητών. Οι εφαρμογές ιστού μπορούν να θεωρηθούν ως μία συγκεκριμένη παραλλαγή λογισμικού πελάτη-διακομιστή όπου το

λογισμικό πελάτη μεταφορτώνεται (downloaded) στο μηχάνημα πελάτη όταν γίνεται επίσκεψη στη σχετική ιστοσελίδα χρησιμοποιώντας τυπικές διαδικασίες όπως το HTTP. Οι ενημερώσεις λογισμικού πελάτη ενδέχεται να πραγματοποιούνται κάθε φορά που γίνεται επίσκεψη στην ιστοσελίδα. Κατά τη διάρκεια της συνεδρίας, ο περιηγητής ερμηνεύει και εμφανίζει τις ιστοσελίδες και ενεργεί ως καθολικός πελάτης για οποιαδήποτε εφαρμογή ιστού.

1.3.2 Δομή

Οι εφαρμογές συνήθως διαιρούνται σε λογικά επίπεδα (tiers) και σε καθένα από αυτά ανατίθεται ένας ρόλος. Σε αντίθεση με τις παραδοσιακές εφαρμογές που αποτελούνται μόνο από ένα επίπεδο, το οποίο βρίσκεται στη μηχανή πελάτη, οι εφαρμογές ιστού είναι από τη φύση τους ν-επίπεδες (n-tiered). Αν και πολλές παραλλαγές είναι δυνατές, η πιο διαδεδομένη δομή είναι η τρι-επίπεδη εφαρμογή (three-tiered application) που παρουσιάζεται στο Σχήμα 1.2. Στην πιο κοινή μορφή της, τα τρία επίπεδα ονομάζονται κατά σειρά: παρουσίαση (presentation), εφαρμογή (application) και αποθήκευση (storage).



Σχήμα 1.2: Αναπαράσταση μίας τρι-επίπεδης εφαρμογής

Ένας περιηγητής είναι το πρώτο επίπεδο (παρουσίαση), ένας μηχανισμός που χρησιμοποιεί κάποια δυναμική τεχνολογία περιεχομένου ιστού (dynamic web content technology), π.χ. ASP, CGI, JSP, Node.js, Django ή Ruby on Rails, είναι το μεσαίο επίπεδο (λογική εφαρμογής) και μία βάση δεδομένων είναι το τρίτο επίπεδο (αποθήκευση). Ο περιηγητής στέλνει αιτήματα στο μεσαίο επίπεδο, το οποίο τα εξυπηρετεί κάνοντας επερωτήσεις (queries) και ενημερώσεις (updates) στη βάση δεδομένων και δημιουργεί μία διεπαφή χρήστη.

Η αρχιτεκτονική τριών επιπέδων έχει στόχο να επιτρέπει την ανεξάρτητη αναβάθμιση (upgrading) ή αντικατάσταση οποιουδήποτε από τα τρία επίπεδα ανάλογα με τις αλλαγές στις απαιτήσεις ή την τεχνολογία[4]. Για παράδειγμα, μία αλλαγή του λειτουργικού συστήματος (operating system) στο επίπεδο παρουσίασης θα επηρεάσει μόνο τον κώδικα διεπαφής χρήστη.

1.3.3 Ανάπτυξη

Η σύνταξη εφαρμογών ιστού συχνά απλοποιείται με τη χρήση πλαισίων εφαρμογών ιστού (web application frameworks). Τα πλαίσια αυτά διευκολύνουν την ταχεία ανάπτυξη εφαρμογών (rapid application development) επιτρέποντας σε μία ομάδα προγραμματιστών (development team) να επικεντρωθεί στα τμήματα της εφαρμογής που σχετίζονται άμεσα με τους στόχους της χωρίς να χρειάζεται να ασχολείται με κοινότυπα ζητήματα ανάπτυξης όπως η διαχείριση χρηστών (user management). Πολλά από τα πλαίσια που χρησιμοποιούνται αποτελούν λογισμικό ανοικτού κώδικα (open-source software).

Η χρήση πλαισίων εφαρμογών ιστού μπορεί συχνά να μειώσει τον αριθμό των σφαλμάτων σε ένα πρόγραμμα καθιστώντας τον κώδικα απλούστερο και επιτρέποντας σε μία ομάδα να επικεντρωθεί στο πλαίσιο, ενώ μία άλλη επικεντρώνεται σε μία συγκεκριμένη περίπτωση χρήσης (use case). Σε εφαρμογές που εκτίθενται σε συνεχείς προσπάθειες πειρατείας (hacking) στο Διαδίκτυο, τα προβλήματα που σχετίζονται με την ασφάλεια μπορούν να προκληθούν από σφάλματα στο πρόγραμμα. Τα πλαίσια μπορούν επίσης να προωθήσουν τη χρήση βέλτιστων πρακτικών όπως το GET μετά το POST.

1.3.4 Δοκιμές[5]

Οι εφαρμογές ιστού υπόκεινται στις ίδιες δοκιμές μονάδας, ενσωμάτωσης και συστήματος (unit, integration and system testing) με τις παραδοσιακές εφαρμογές επιφάνειας εργασίας (desktop applications). Στόχος των δοκιμών αυτών είναι αφενός η επαλήθευση (verification) της ορθής λειτουργίας της εφαρμογής και αφετέρου ο εντοπισμός των σφαλμάτων (error identification) που χρειάζονται διόρθωση. Οι δοκιμές των εφαρμογών ιστού, ωστόσο, έχουν κάποια ιδιαίτερα χαρακτηριστικά που τις διαφοροποιούν από εκείνες των παραδοσιακών εφαρμογών. Για παράδειγμα, οι εφαρμογές ιστού τείνουν να έχουν πολλές πληροφορίες που θα μπορούσαν να περιέχουν λάθη, παραλείψεις, πλεονασμούς, κ.λπ. Επίσης ενδέχεται να οργανώνονται σε πολλαπλά στρώματα (layers) και να επηρεάζονται από πολλαπλές, δυναμικές διαμορφώσεις (configurations). Επομένως, οι δοκιμές και ο εντοπισμός σφαλμάτων περιλαμβάνουν πιο περίπλοκες διαδικασίες όπως η συμπερίληψη λειτουργικής ανάλυσης (operational analysis inclusion) για κάθε στρώμα ή διαμόρφωση.

Οι πελάτες εφαρμογών ιστού ποικίλουν σε μεγάλο βαθμό, επομένως οι ομάδες προγραμματιστών ενδέχεται να πραγματοποιούν μερικές επιπρόσθετες δοκιμές αναφορικά με την ασφάλεια (security), την απόδοση, φόρτωση και πίεση (performance, load and stress), την επικύρωση HTML/CSS (HTML/CSS validation), την

προσβασιμότητα (accessibility), την ευχρηστία (usability) και τη συμβατότητα με τους διάφορους περιηγητές (cross-browser compatibility).

Πολλοί τύποι δοκιμών είναι αυτοματοποιήσιμοι (automatable). Σε επίπεδο συνιστωσών (component level), χρησιμοποιείται συχνά κάποιο από τα βοηθητικά πακέτα xUnit (xUnit packages), ενώ ένας οργανισμός μπορεί να δημιουργήσει και το δικό του πλαίσιο δοκιμής μονάδων (unit testing framework). Σε επίπεδο GUI (Graphical User Interface), χρήσιμα μπορεί να φανούν το Watir και το iMacros.

1.4 Πλαίσια εφαρμογών ιστού[6]

1.4.1 Επισκόπηση

Ένα πλαίσιο εφαρμογών ιστού (Web Application Framework - WAF) ή απλά πλαίσιο ιστού (WF) είναι ένα πλαίσιο λογισμικού (software framework) που έχει σχεδιαστεί για να υποστηρίζει την ανάπτυξη εφαρμογών ιστού, συμπεριλαμβανομένων υπηρεσιών, πόρων και API ιστού. Τα πλαίσια ιστού παρέχουν έναν τυπικό τρόπο κατασκευής (build) και επιστράτευσης (deployment) εφαρμογών στον Παγκόσμιο Ιστό. Στοχεύουν επίσης στην αυτοματοποίηση του φόρτου (overhead) από κοινότυπες εργασίες που εκτελούνται στον διαδικτυακό προγραμματισμό. Για παράδειγμα, πολλά πλαίσια ιστού παρέχουν βιβλιοθήκες (libraries) για πρόσβαση σε βάσεις δεδομένων (database access), συστήματα προτύπων ιστού (web template systems) και διαχείριση συνεδριών (session management) και συχνά προωθούν την επαναχρησιμοποίηση κώδικα (code reuse). Παρόλο που στοχεύουν συχνά στην ανάπτυξη δυναμικών ιστοτόπων (dynamic websites), είναι εφαρμόσιμα ακόμα και για στατικούς ιστοτόπους (static websites).

1.4.2 Τύποι αρχιτεκτονικών

1.4.2.1 Μόρφημα MVC[7]

Το αρχιτεκτονικό μόρφημα MVC (Model-View-Controller architectural pattern) χρησιμοποιείται συχνά για την ανάπτυξη διεπαφών χρήστη και διαιρεί μία εφαρμογή σε τρεις διασυνδεδεμένες συνιστώσες (interconnected components). Αυτό γίνεται για να διαχωριστούν οι εσωτερικές αναπαραστάσεις (internal representations) των πληροφοριών από τους τρόπους με τους οποίους οι πληροφορίες παρουσιάζονται και γίνονται δεκτές από τον χρήστη. Η χαλαρή σύζευξη (loose coupling) των συνιστωσών επιτρέπει την εύκολη και γρήγορη επαναχρησιμοποίηση κώδικα μεταξύ εφαρμογών και την παράλληλη ανάπτυξη (parallel development) των συνιστωσών χωρίς αλληλεξαρτήσεις.

Οι συνιστώσες του μορφήματος MVC είναι οι εξής:

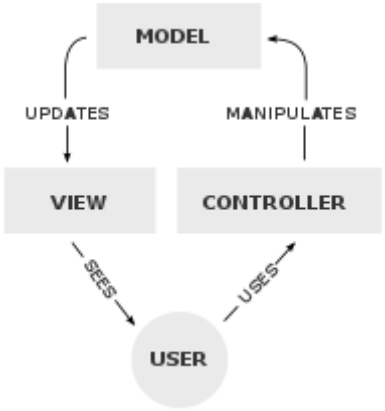
- Μοντέλο (Model): Η κεντρική συνιστώσα του μορφήματος. Είναι η δυναμική δομή δεδομένων (data structure) της εφαρμογής ανεξαρτήτως της διεπαφής χρήστη. Διαχειρίζεται άμεσα τα δεδομένα, τη λογική και τους κανόνες της εφαρμογής.
- Όψη (View): Οποιαδήποτε αναπαράσταση πληροφοριών, π.χ. γράφημα, διάγραμμα ή πίνακας. Είναι δυνατόν να υπάρξουν πολλαπλές όψεις των ίδιων πληροφοριών

όπως ένα διάγραμμα ράβδων (bar chart) για διαχείριση και μία προβολή πίνακα (tabular view) για λογιστική (accounting).

- Ελεγκτής (Controller): Αποδέχεται την είσοδο χρήστη (user input) και τη μετατρέπει σε εντολές (commands) για το μοντέλο ή την όψη.

Εκτός από τη διαίρεση της εφαρμογής σε αυτές τις συνιστώσες, το μόρφημα MVC ορίζει τις αλληλεπιδράσεις (interactions) μεταξύ τους όπως φαίνεται στο Σχήμα 1.3:

- Το μοντέλο είναι υπεύθυνο για τη διαχείριση των δεδομένων της εφαρμογής. Λαμβάνει την είσοδο χρήστη από τον ελεγκτή.
- Η όψη αποτελεί παρουσίαση του μοντέλου σε συγκεκριμένη μορφή.
- Ο ελεγκτής ανταποκρίνεται στην είσοδο χρήστη και αλληλεπιδρά με τα αντικείμενα μοντέλου δεδομένων (data model objects). Ο ελεγκτής λαμβάνει την είσοδο, προαιρετικά την επικυρώνει (validates) και στη συνέχεια τη μεταβιβάζει στο μοντέλο.



Σχήμα 2.3: Διάγραμμα αλληλεπιδράσεων μεταξύ των συνιστωσών του μορφήματος MVC

Πολλά πλαίσια ιστού ακολουθούν το μόρφημα MVC για να διαχωρίσουν το μοντέλο δεδομένων με επιχειρησιακούς κανόνες (business rules) από τη διεπαφή χρήστη. Αυτό θεωρείται γενικά μία καλή πρακτική καθώς οργανώνει τον κώδικα σε λογικές, συνεκτικές μονάδες (logical, cohesive modules), προωθεί την επαναχρησιμοποίηση κώδικα και επιτρέπει τη χρησιμοποίηση πολλαπλών διεπαφών. Στις εφαρμογές ιστού, αυτό επιτρέπει την εμφάνιση διαφορετικών όψεων, π.χ. ιστοσελίδων για τελικούς χρήστες (end users) και διεπαφών υπηρεσιών ιστού για απομακρυσμένες εφαρμογές (remote applications).

Τα περισσότερα πλαίσια MVC ακολουθούν μία αρχιτεκτονική ώθησης/ενέργειας (push-based/action-based architecture). Αυτά τα πλαίσια χρησιμοποιούν ενέργειες που κάνουν την απαιτούμενη επεξεργασία και στη συνέχεια «ωθούν» τα δεδομένα στο στρώμα όψεων για να απεικονίσουν τα αποτελέσματα. Τα πλαίσια Django, Ruby on Rails, Symfony, Spring MVC, Stripes, Sails.js, Diamond και CodeIgniter αποτελούν αντιπροσωπευτικά παραδείγματα αυτής της αρχιτεκτονικής. Υπάρχουν όμως και πλαίσια που ακολουθούν μία αρχιτεκτονική έλξης/συνιστώσας (pull-based/component-based architecture). Αυτά τα πλαίσια ξεκινούν με το στρώμα όψεων, το οποίο μπορεί έπειτα να «έλξει» τα αποτελέσματα από πολλαπλούς ελεγκτές ανάλογα με τις ανάγκες. Σε αυτή την αρχιτεκτονική, πολλαπλοί ελεγκτές μπορούν να συσχετιστούν με μία μόνο όψη. Τα πλαίσια Lift, Tapestry, JBoss Seam, JavaServer

Faces, (μ)Micro και Wicket είναι παραδείγματα της αρχιτεκτονικής έλξης. Τα πλαίσια Play, Struts, RIFE και ZK έχουν υποστήριξη για κλήσεις ελεγκτών που εντάσσονται τόσο σε εφαρμογές ώθησης όσο και έλξης.

Ορισμένα πλαίσια MVC υιοθετούν μία προσέγγιση ελαφριού πελάτη (thin client approach) που τοποθετεί σχεδόν ολόκληρη τη λογική μοντέλου-όψης-ελεγκτή στον διακομιστή. Αυτό αντικατοπτρίζεται σε πλαίσια όπως το Django, το Rails και το ASP.NET MVC. Σε αυτή την προσέγγιση, ο πελάτης στέλνει στον ελεγκτή είτε αιτήματα υπερσυνδέσμου (hyperlink requests) είτε υποβολές φόρμας (form submissions) και κατόπιν λαμβάνει μία πλήρη και ενημερωμένη ιστοσελίδα (ή άλλο έγγραφο) από την όψη. Το μοντέλο υπάρχει εξ ολοκλήρου στον διακομιστή. Άλλα πλαίσια όπως το AngularJS, το EmberJS, το JavaScriptMVC και το Backbone επιτρέπουν στις συνιστώσες MVC να εκτελούνται εν μέρει στον πελάτη.

1.4.2.2 Τρι-επίπεδη οργάνωση

Στην τρι-επίπεδη οργάνωση, οι εφαρμογές είναι δομημένες γύρω από τρία φυσικά επίπεδα: πελάτης, εφαρμογή και βάση δεδομένων. Η βάση δεδομένων αναφέρεται συνήθως σε ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational DataBase Management System - RDBMS). Η εφαρμογή περιέχει την επιχειρησιακή λογική (business logic) που εκτελείται σε έναν διακομιστή και επικοινωνεί με τον πελάτη χρησιμοποιώντας HTTP. Ο πελάτης σε εφαρμογές ιστού είναι ένας περιηγητής ιστού που εκτελεί HTML, η οποία παράγεται από το επίπεδο εφαρμογής. Ο όρος δεν πρέπει να συγχέεται με το MVC, όπου, αντίθετα από την τρι-επίπεδη αρχιτεκτονική, θεωρείται καλή πρακτική η απομάκρυνση της επιχειρησιακής λογικής από τον ελεγκτή, το «μεσαίο επίπεδο».

1.4.3 Υποστήριξη εφαρμογών

Τα πλαίσια δημιουργούνται για να υποστηρίξουν την κατασκευή εφαρμογών ιστού που βασίζονται σε μία μόνο γλώσσα προγραμματισμού, ποικίλοντας σε εστίαση από εργαλεία γενικού σκοπού (general purpose tools) όπως το Zend Framework, το Django και το Ruby on Rails, τα οποία επαυξάνουν τις δυνατότητες μίας συγκεκριμένης γλώσσας, σε προγραμματιζόμενα πακέτα μητρικής γλώσσας (native-language programmable packages), τα οποία δομούνται γύρω από μία συγκεκριμένη εφαρμογή χρήστη (user application) όπως τα συστήματα διαχείρισης περιεχομένου (content management systems), μερικά εργαλεία ανάπτυξης για κινητά τηλέφωνα (mobile development tools) και μερικά εργαλεία πύλης (portal tools).

1.4.3.1 Πλαίσια γενικού σκοπού για κατασκευή ιστοτόπων

Τα πλαίσια ιστού πρέπει να λειτουργούν σύμφωνα με τους αρχιτεκτονικούς κανόνες των περιηγητών και των πρωτοκόλλων ιστού όπως το HTTP, το οποίο είναι ακαταστατικό (stateless). Οι ιστοσελίδες εξυπηρετούνται από ένα διακομιστή και στη συνέχεια μπορούν να τροποποιηθούν από τον περιηγητή χρησιμοποιώντας JavaScript. Κάθε προσέγγιση έχει τα πλεονεκτήματα και τα μειονεκτήματά της.

Οι αλλαγές στη σελίδα από πλευράς διακομιστή συνήθως απαιτούν την ανανέωση της σελίδας (page refresh), επιτρέποντας όμως τη χρήση οποιασδήποτε γλώσσας και την αξιοποίηση περισσότερης υπολογιστικής ισχύος. Οι αλλαγές από πλευράς πελάτη επιτρέπουν την ενημέρωση της σελίδας (page update) κατά μικρά τμήματα προσιδιάζοντας σε εφαρμογή επιφάνειας εργασίας, αλλά περιορίζονται σε JavaScript και τρέχουν στον περιηγητή του χρήστη, ο οποίος ενδέχεται να έχει περιορισμένη υπολογιστική ισχύ. Συνήθως χρησιμοποιείται κάποιος συνδυασμός των δύο. Εφαρμογές που κάνουν βαριά χρήση της JavaScript ονομάζονται εφαρμογές μίας σελίδας (single-page applications) και συνήθως χρησιμοποιούν ένα πλαίσιο ιστού JavaScript από πλευράς πελάτη για την οργάνωση του κώδικα.

Πλαίσια ιστού από πλευράς διακομιστή είναι, μεταξύ άλλων, τα Spring MVC, Ruby on Rails, Laravel, Django, Sails.js, ASP.NET Core, Zend Framework και Express.js. Πλαίσια ιστού από πλευράς πελάτη είναι, μεταξύ άλλων, τα Backbone.js, AngularJS, Angular, EmberJS, ReactJS and Vue.js.

1.4.4 Χαρακτηριστικά

Τα πλαίσια καθορίζουν συνήθως τη ροή ελέγχου (control flow) ενός προγράμματος και επιτρέπουν στον χρήστη του πλαισίου να «επεμβαίνει» σε αυτή τη ροή εκθέτοντας διάφορα συμβάντα (events). Αυτό το σχεδιαστικό μόρφημα «αντιστροφής του ελέγχου» ("inversion of control" design pattern) θεωρείται ότι είναι μία καθοριστική αρχή ενός πλαισίου και ωφελεί τον κώδικα επιβάλλοντας μία κοινή ροή για μία ομάδα προγραμματιστών την οποία ο καθένας μπορεί να προσαρμόσει με παρόμοιο τρόπο. Για παράδειγμα, μερικά δημοφιλή «μικροπλαίσια» ("microframeworks") όπως το Sinatra της Ruby (το οποίο ενέπνευσε το Express.js) επιτρέπουν την προσάρτηση μεσισμικού (middleware hook) πριν και μετά από αιτήματα HTTP. Αυτές οι λειτουργίες μεσισμικού μπορούν να είναι οτιδήποτε και επιτρέπουν στον χρήστη να καθορίζει την καταγραφή (logging), τον έλεγχο ταυτότητας (authentication), τη διαχείριση συνεδριών (session management) και την ανακατεύθυνση (redirecting).

1.4.4.1 Σύστημα προτύπων ιστού[8]

Ένας κοινός στόχος μεταξύ έμπειρων προγραμματιστών ιστού είναι να αναπτύσσουν και να επιστρατεύουν στον Παγκόσμιο Ιστό εφαρμογές που είναι ευέλικτες και εύκολα συντηρήσιμες. Ένα σημαντικό στοιχείο για την επίτευξη αυτού του στόχου είναι ο διαχωρισμός της επιχειρησιακής λογικής από τη λογική παρουσίασης. Οι προγραμματιστές χρησιμοποιούν συστήματα προτύπων ιστού (με διάφορους βαθμούς επιτυχίας) για να διατηρήσουν αυτόν τον διαχωρισμό.

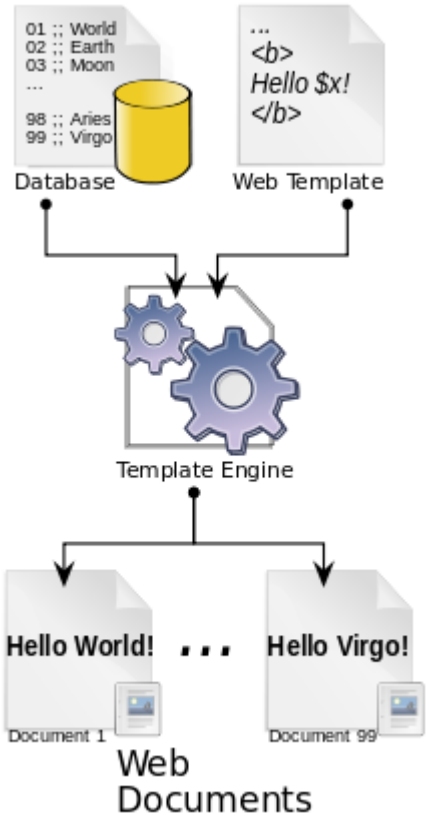
Κατά τη διαδικτυακή δημοσίευση (web publishing), ένα σύστημα προτύπων ιστού (web template system) επιτρέπει στους σχεδιαστές και προγραμματιστές ιστού (web designers and developers) να δουλεύουν με πρότυπα ιστού (web templates) για να δημιουργούν αυτόματα προσαρμοσμένες ιστοσελίδες όπως τα αποτελέσματα μίας αναζήτησης. Το σύστημα επαναχρησιμοποιεί τα στατικά στοιχεία ιστοσελίδων, ενώ καθορίζει τα δυναμικά στοιχεία με βάση τις παραμέτρους του αιτήματος ιστού. Τα πρότυπα ιστού μπορούν να υποστηρίξουν στατικό περιεχόμενο παρέχοντας βασική δομή και εμφάνιση. Οι προγραμματιστές μπορούν να υλοποιήσουν πρότυπα μέσω

συστημάτων διαχείρισης περιεχομένου, πλαισίων ιστού και συντακτών HTML (HTML editors).

Ένα σύστημα προτύπων ιστού αποτελείται από τα εξής:

- Μηχανή προτύπων (template engine): το κύριο στοιχείο επεξεργασίας του συστήματος.
- Πόροι περιεχομένου (content resources): ροές δεδομένων εισόδου (input data streams) όπως μία σχεσιακή βάση δεδομένων (relational database), ένας κατάλογος LDAP (Lightweight Directory Access Protocol), αρχεία XML (eXtensible Markup Language) και άλλα είδη τοπικών ή δικτυακών δεδομένων.
- Πόροι προτύπου (template resources): πρότυπα ιστού που καθορίζονται σύμφωνα με μία γλώσσα προτύπων (template language).

Οι πόροι προτύπου και περιεχομένου επεξεργάζονται και συνδυάζονται από τη μηχανή προτύπων για τη μαζική παραγωγή εγγράφων ιστού (web documents) όπως φαίνεται στο Σχήμα 1.4 (με τον όρο «έγγραφο ιστού» εννοούμε οποιαδήποτε μορφή εξόδου προς μετάδοση στο Διαδίκτυο μέσω HTTP ή άλλου διαδικτυακού πρωτοκόλλου).



Σχήμα 1.4: Αναπαράσταση ενός συστήματος προτύπων ιστού

1.4.4.2 Κρυφή μνήμη ιστού[9]

Μία κρυφή μνήμη ιστού (web cache) ή κρυφή μνήμη HTTP (HTTP cache) είναι μία τεχνολογία πληροφοριών (information technology) για την προσωρινή αποθήκευση (caching) εγγράφων ιστού όπως ιστοσελίδες, εικόνες και άλλοι τύποι πολυμέσων ιστού (web multimedia) για να μειωθεί η χρήση του εύρους ζώνης (bandwidth usage), το φορτίο του διακομιστή (server load) και η αντιληπτή «υστέρηση» ("lag"). Ένα σύστημα κρυφής μνήμης ιστού αποθηκεύει αντίγραφα των εγγράφων που διέρχονται από αυτό και τα επακόλουθα αιτήματα μπορούν να ικανοποιηθούν από την κρυφή μνήμη εάν πληρούνται συγκεκριμένες συνθήκες. Ένα σύστημα κρυφής μνήμης μπορεί να αναφέρεται είτε σε μία συσκευή είτε σε ένα πρόγραμμα υπολογιστή. Ορισμένα πλαίσια εφαρμογών παρέχουν μηχανισμούς για την προσωρινή αποθήκευση εγγράφων και την παράκαμψη διαφόρων σταδίων της προετοιμασίας της σελίδας όπως η πρόσβαση σε βάσεις δεδομένων ή η ερμηνεία προτύπων (template interpretation).

1.4.4.3 Έλεγχος πρόσβασης

Ορισμένα πλαίσια ιστού συνοδεύονται από συστήματα αυθεντικοποίησης (authentication) και εξουσιοδότησης (authorization), τα οποία επιτρέπουν στον διακομιστή ιστού να ταυτοποιεί (identify) τους χρήστες της εφαρμογής, και περιορίζουν την πρόσβαση σε λειτουργίες με βάση κάποια καθορισμένα κριτήρια. Για παράδειγμα, το Drupal παρέχει αφενός πρόσβαση βασισμένη σε ρόλους (role-based access) στις σελίδες και αφετέρου μία διεπαφή ιστού (web-based interface) για τη δημιουργία χρηστών και την ανάθεση ρόλων σε αυτούς.

Η διαδικασία εξουσιοδότησης διαφέρει από αυτήν της αυθεντικοποίησης[10]. Η αυθεντικοποίηση είναι η διαδικασία επαλήθευσης ότι «εσύ είσαι αυτός που ισχυρίζεσαι ότι είσαι», ενώ η εξουσιοδότηση είναι η διαδικασία επαλήθευσης ότι «εσύ επιτρέπεται να κάνεις αυτό που προσπαθείς να κάνεις». Αυτό δεν σημαίνει ότι η εξουσιοδότηση προϋποθέτει την αυθεντικοποίηση· ένας ανώνυμος πράκτορας (anonymous agent) θα μπορούσε να εξουσιοδοτηθεί για ένα περιορισμένο σύνολο ενεργειών (action set).

Μία βασική χρήση της αυθεντικοποίησης και της εξουσιοδότησης είναι ο έλεγχος πρόσβασης (access control). Όταν ένας χρήστης προσπαθεί να συνδεθεί (log in) στην εφαρμογή ενός διακομιστή ιστού, του ζητείται αρχικά να ταυτοποιηθεί με ένα (μοναδικό) όνομα σύνδεσης (login name) και να το υποστηρίξει με έναν κωδικό πρόσβασης (password). Κατόπιν ελέγχεται η αυθεντικότητα (authenticity) του συνδυασμού των διαπιστευτηρίων (credentials) μέσω αντιπαραβολής με ένα σύνολο έγκυρων εγγραφών ονόματος-κωδικού (login-password records) που διατηρεί ο διακομιστής, π.χ. σε μία βάση δεδομένων. Σε περίπτωση που ο συνδυασμός είναι έγκυρος/αυθεντικός, επιτυγχάνεται το βήμα της αυθεντικοποίησης του χρήστη και του επιτρέπεται η πρόσβαση στον διακομιστή. Στη συνέχεια, ένα σύνολο προκαθορισμένων δικαιωμάτων (permissions) και περιορισμών (restrictions) για το συγκεκριμένο όνομα σύνδεσης εκχωρείται (assigned) σε αυτόν τον χρήστη συνήθως με τη μορφή κάποιου ρόλου, π.χ. απλός χρήστης ή διαχειριστής (administrator), και ελέγχει την πρόσβασή του στις διάφορες λειτουργίες της εφαρμογής. Η πρόσβαση του χρήστη μπορεί επίσης να επηρεάζεται από τα χαρακτηριστικά (attributes), π.χ. ηλεκτρονικό ταχυδρομείο ή ηλικία, που έχει (ή δεν έχει) καταχωρήσει ο ίδιος στο προφίλ (profile) του[11]. Έτσι ολοκληρώνεται και το βήμα της εξουσιοδότησης.

1.4.4.4 Πρόσβαση, απεικόνιση και διαμόρφωση βάσης δεδομένων

Πολλά πλαίσια ιστού δημιουργούν ένα ενοποιημένο API (unified API) στο οπίσθιο τμήμα (backend) μίας βάσης δεδομένων, δίνοντας τη δυνατότητα στις εφαρμογές ιστού να δουλεύουν με μία ποικιλία από βάσεις δεδομένων χωρίς αλλαγές κώδικα και επιτρέποντας στους προγραμματιστές να ασχολούνται με έννοιες υψηλότερου επιπέδου (higher-level concepts). Επιπροσθέτως, μερικά αντικειμενοστρεφή (object-oriented) πλαίσια περιέχουν εργαλεία αντιστοίχισης (mapping tools) για την παροχή αντικειμενοσχεσιακής απεικόνισης (Object-Relational Mapping - ORM), η οποία αντιστοιχεί αντικείμενα σε πλειάδες (tuples).

Ορισμένα πλαίσια ελαχιστοποιούν τη διαμόρφωση (configuration) των εφαρμογών ιστού χρησιμοποιώντας ενδοσκόπηση (introspection) ή/και ακολουθώντας γνωστές συμβάσεις (conventions). Για παράδειγμα, πολλά πλαίσια Java χρησιμοποιούν το Hibernate ως στρώμα μονιμοποίησης δεδομένων (persistence layer), το οποίο μπορεί να δημιουργήσει ένα σχήμα βάσης δεδομένων (database schema) κατά τη διάρκεια εκτέλεσης ικανό να αποθηκεύσει μόνιμα (persisting) τις απαραίτητες πληροφορίες. Αυτό επιτρέπει στον σχεδιαστή εφαρμογών (application designer) να σχεδιάζει επιχειρηματικά αντικείμενα (business objects) χωρίς να χρειάζεται να καθορίζει ρητά ένα σχήμα βάσης δεδομένων. Πλαίσια όπως το Ruby on Rails μπορούν επίσης να λειτουργήσουν αντίστροφα, δηλαδή να ορίσουν ιδιότητες αντικειμένων μοντέλων (model object properties) κατά τη διάρκεια εκτέλεσης βάσει ενός σχήματος βάσης δεδομένων.

Τα πλαίσια ιστού ενδέχεται να παρέχουν και άλλα χαρακτηριστικά (features) όπως υποστήριξη συναλλαγών (transactional support) και εργαλεία μετεγκατάστασης βάσης δεδομένων (database migration tools).

1.4.4.5 Αντιστοίχιση διευθύνσεων URL

Το εργαλείο αντιστοίχισης ή δρομολόγησης διευθύνσεων URL (URL mapping or routing facility) ενός πλαισίου είναι ο μηχανισμός με τον οποίο το πλαίσιο ερμηνεύει τις διευθύνσεις URL (Uniform Resource Locator). Ορισμένα πλαίσια, όπως το Drupal και το Django, αντιπαραβάλλουν το παρεχόμενο URL με προκαθορισμένα μοτίβα (pattern matching) χρησιμοποιώντας κανονικές εκφράσεις (regular expressions), ενώ μερικά άλλα χρησιμοποιούν τεχνικές επανεγγραφής (rewriting techniques) για να μεταφράσουν το παρεχόμενο URL σε ένα που θα αναγνωρίσει ο υποκείμενος μηχανισμός. Μία άλλη τεχνική είναι αυτή της διάσχισης γραφημάτων (graph traversal) όπως αυτή που χρησιμοποιείται από το Zope, όπου ένα URL αποσυντίθεται σε βήματα που διασχίζουν ένα γράφημα αντικειμένων (μοντέλων και όψεων).

Ένα σύστημα αντιστοίχισης URL που χρησιμοποιεί αντιπαραβολή ή επανεγγραφή μοτίβων για τη δρομολόγηση και τη διαχείριση των αιτημάτων επιτρέπει τη χρήση συντομότερων και πιο «φιλικών URL» ("friendly URLs"), αυξάνοντας την απλότητα του ιστοτόπου και επιτρέποντας την καλύτερη ευρετηρίαση (indexing) από τις μηχανές αναζήτησης (search engines). Για παράδειγμα, ένα URL που τελειώνει με "/page.cgi?cat=science&topic=physics" θα μπορούσε να αλλαχθεί απλώς σε "/page/science/physics". Αυτό καθιστά το URL ευκολότερο στην ενθύμηση, ανάγνωση και γραφή για τους χρήστες και παρέχει στις μηχανές αναζήτησης καλύτερες

πληροφορίες σχετικά με τη διαρθρωτική διάταξη (structural layout) του ιστοτόπου. Μία προσέγγιση μέσω διάσχισης γραφημάτων τείνει επίσης να έχει ως αποτέλεσμα τη δημιουργία φιλικών URL. Ένα συντομότερο URL όπως το "/page/science" τείνει να υπάρχει από προεπιλογή (by default) καθώς αυτό είναι απλώς μία συντομότερη μορφή της μακρύτερης διάσχισης προς το "/page/science/physics".

1.5 Σύνοψη της εργασίας

Η υπόλοιπη εργασία οργανώνεται ως εξής:

- Το κεφάλαιο 2 αφορά τα εργαλεία που χρησιμοποιούνται για την ανάπτυξη, συμπεριλαμβανομένης της επιστράτευσης, της εφαρμογής ιστού SLUB. Αρχικά εξετάζονται οι έννοιες του εμπρόσθιου και του οπίσθιου τμήματος στην τεχνολογία λογισμικού και δίνεται ιδιαίτερη έμφαση στην εφαρμογή τους στον διαδικτυακό προγραμματισμό. Αφού παρουσιαστούν συνοπτικά οι πιο διαδεδομένες τεχνολογίες εμπρόσθιου και οπίσθιου τμήματος στον διαδικτυακό προγραμματισμό, κατόπιν μελετώνται λεπτομερώς εκείνες που αξιοποιούνται από το πλαίσιο ιστού Django για την ανάπτυξη του SLUB, δηλαδή οι γλώσσες HTML-CSS-JavaScript στο εμπρόσθιο τμήμα και οι συνιστώσες Linux-Apache-MySQL-Python της στοίβας λογισμικού LAMP στο οπίσθιο τμήμα. Τελικά αναλύονται τα χαρακτηριστικά του Django, το οποίο ακολουθεί το αρχιτεκτονικό μόρφημα μοντέλου-προτύπου-όψης, MTV, και την προδιαγραφή WSGI.
- Το κεφάλαιο 3 πραγματεύεται τα μοντέλα δεδομένων του SLUB και τον ιστότοπο διαχείρισης του Django. Πρώτα περιγράφεται η διαρθρωτική διάταξη του SLUB στο πλαίσιο Django ως ιεραρχία των αρχείων διαμόρφωσης και λειτουργικότητας της εφαρμογής usermerge στο έργο slub του Django. Γίνεται επίσης εμβάθυνση στο εικονικό περιβάλλον της Python, το οποίο περικλείει αφενός το προαναφερθέν έργο και αφετέρου όλα τα επιμέρους πακέτα της Python που απαιτούνται για την ορθή εκτέλεσή του. Έπειτα αναλύονται τα κύρια μοντέλα δεδομένων του SLUB και τα εγγενή μοντέλα αυθεντικοποίησης και εξουσιοδότησης χρηστών του Django, ενώ παρουσιάζεται και η απεικόνισή τους σε κατάλληλους πίνακες της βάσης usermergeDB μέσω του Django ORM. Τέλος, σκιαγραφείται ο ρόλος του ιστοτόπου διαχείρισης του Django ως διαχειριστική διεπαφή CRUD των πινάκων της βάσης δεδομένων και παρέχεται ένα τυπικό παράδειγμα αξιοποίησής του.
- Το κεφάλαιο 4 αναφέρεται στις όψεις λειτουργιών και στα πρότυπα ιστού του SLUB. Στην αρχή εξετάζεται η υλοποίηση των συνεδριών HTTP, των ACID-συμβατών συναλλαγών βάσης δεδομένων και των όψεων σφαλμάτων HTTP στο Django, καθώς και η εφαρμογή τους στα πλαίσια των όψεων του SLUB. Ακολούθως προσδιορίζεται η τυπική ροή λειτουργίας του SLUB, δηλαδή το σύνολο των συμβατικών βημάτων επιτέλεσης μίας λειτουργίας της εφαρμογής ιστού μέσω της αντίστοιχης όψης και του αντίστοιχου προτύπου HTML, που περιλαμβάνει, μεταξύ άλλων, ελέγχους πρόσβασης στην όψη, ελέγχους υποβολής-επικύρωσης-μετεπικύρωσης της φόρμας του προτύπου HTML και αξιοποίηση των σχετικών μοντέλων δεδομένων του SLUB. Στο τέλος χρησιμοποιείται η προαναφερθείσα ροή λειτουργίας ως βάση για τη διεξοδική μελέτη όλων των λειτουργιών ανά ρόλο (μοντέλο) χρήστη του SLUB, δηλαδή για τη μελέτη των όψεων λειτουργιών και των προτύπων ιστού που απευθύνονται (I) στους

διαχειριστές (χρήστες Admin), (II) στους απλούς χρήστες (χρήστες User), και (III) σε όλους τους χρήστες της εφαρμογής ιστού.

- Καταληκτικά, στο κεφάλαιο 5 επισημαίνονται η συνεισφορά και τα συμπεράσματα της εργασίας, ενώ προτείνονται ακόμα μελλοντικές δράσεις για τη βελτίωση των υπαρχόντων χαρακτηριστικών και την προσθήκη καινούριων στο SLUB, κυρίως μέσω της καλύτερης αξιοποίησης των δυνατοτήτων που παρέχονται από το Django.

2 Εργαλεία ανάπτυξης του SLUB

2.1 Εμπρόσθιο και οπίσθιο τμήμα[12]

2.1.1 Επισκόπηση

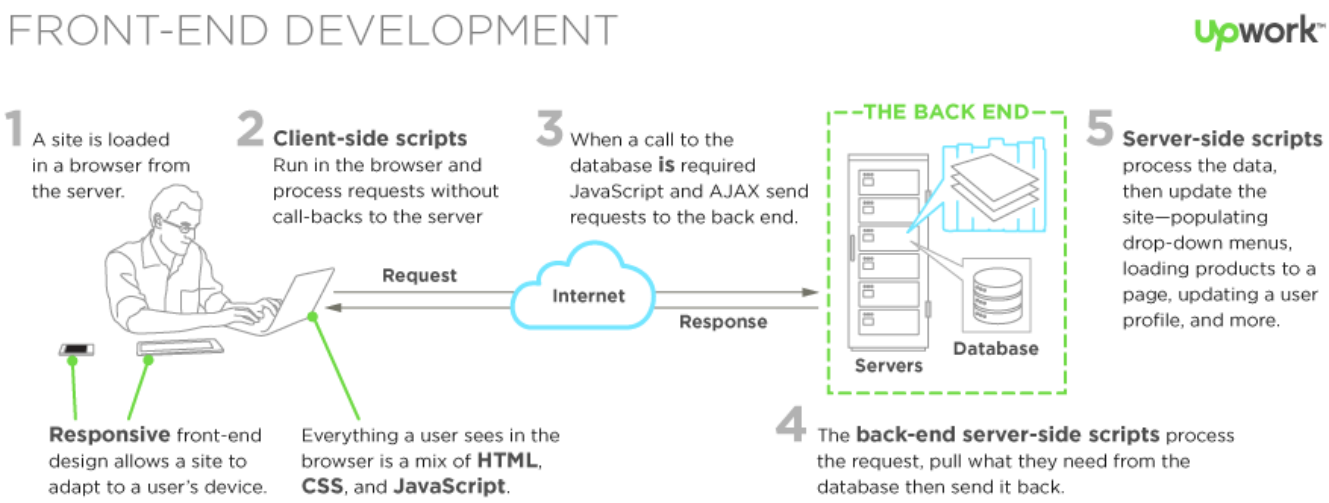
Στην τεχνολογία λογισμικού (software engineering), οι όροι εμπρόσθιο και οπίσθιο τμήμα (front end and back end) αναφέρονται στον διαχωρισμό ενδιαφερόντων (separation of concerns) μεταξύ του στρώματος παρουσίασης (presentation layer) - εμπρόσθιο τμήμα - και του στρώματος πρόσβασης δεδομένων (data access layer) - οπίσθιο τμήμα - ενός λογισμικού, ενός υλισμικού (hardware) ή μίας φυσικής υποδομής (physical infrastructure). Στο μοντέλο πελάτη-διακομιστή, ο πελάτης θεωρείται συνήθως το εμπρόσθιο τμήμα και ο διακομιστής το οπίσθιο τμήμα, ακόμη και όταν κάποια εργασία παρουσίασης γίνεται πραγματικά στον ίδιο τον διακομιστή.

Στην αρχιτεκτονική λογισμικού (software architecture), μπορεί να υπάρχουν πολλά στρώματα μεταξύ του υλισμικού και του τελικού χρήστη (end user). Το καθένα μπορεί να λέγεται ότι έχει ένα εμπρόσθιο και ένα οπίσθιο τμήμα. Το εμπρόσθιο τμήμα είναι μία αφαιρετική έννοια που απλοποιεί την υποκείμενη συνιστώσα παρέχοντας μία φιλική προς τον χρήστη διεπαφή (user-friendly interface), ενώ το οπίσθιο τμήμα συνήθως διαχειρίζεται την επιχειρησιακή λογική και την αποθήκευση δεδομένων. Ένας εμπειρικός κανόνας είναι ότι η πλευρά του πελάτη (ή «εμπρόσθιο τμήμα») είναι οποιαδήποτε συνιστώσα χειρίζεται ο χρήστης. Ο κώδικας από την πλευρά του διακομιστή (ή «οπίσθιο τμήμα») βρίσκεται κατά κανόνα στον διακομιστή, ο οποίος είναι συνήθως απομακρυσμένος φυσικά από τον χρήστη.

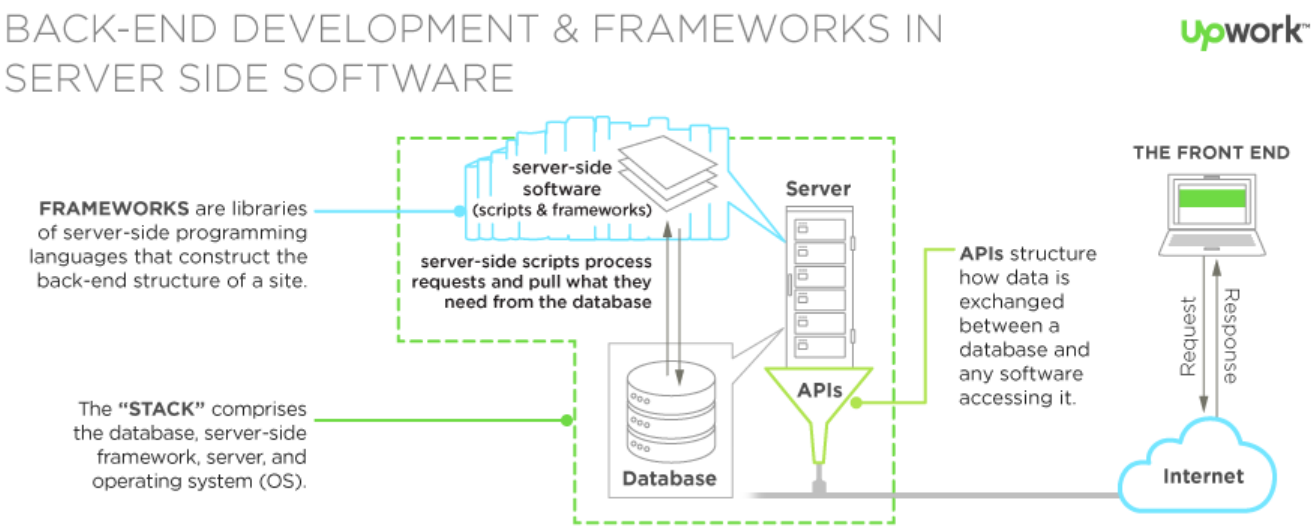
Σε μεγάλα υποσυστήματα υπολογιστών, ένας γραφικός διαχειριστής αρχείων (graphical file manager) δρα ως εμπρόσθιο τμήμα του συστήματος αρχείων (file system) του υπολογιστή και ένας φλοιός (shell) διασυνδέεται με το λειτουργικό σύστημα (operating system). Το εμπρόσθιο τμήμα είναι στραμμένο προς τον χρήστη, ενώ το οπίσθιο τμήμα εκκινεί τα προγράμματα του λειτουργικού συστήματος ως απόκριση. Επίσης, μία γραφική διεπαφή χρήστη (Graphical User Interface - GUI) δρα ως εμπρόσθιο περιβάλλον επιφάνειας εργασίας (desktop environment) αντί της διεπαφής γραμμής εντολών (Command-Line Interface - CLI), η οποία απαιτεί την εκμάθηση ειδικής ορολογίας και την απομνημόνευση εντολών.

2.1.2 Μελέτη περίπτωσης διαδικτυακού προγραμματισμού

Οι έννοιες του εμπρόσθιου και του οπίσθιου τμήματος είναι αρκετά διαδεδομένες στον διαδικτυακό προγραμματισμό όπου υπάρχουν τρία είδη εξειδίκευσης προγραμματιστών: οι προγραμματιστές εμπρόσθιου τμήματος (front-end developers), οι προγραμματιστές οπίσθιου τμήματος (back-end developers) και οι προγραμματιστές πλήρους στοίβας (full-stack developers)[13]. Όπως φαίνεται και στα Σχήματα 2.1 και 2.2[14], οι προγραμματιστές εμπρόσθιου τμήματος είναι υπεύθυνοι για τη συμπεριφορά και τα οπτικά χαρακτηριστικά (visuals) που εκτελούνται στον περιηγητή ιστού (πελάτη), οι προγραμματιστές οπίσθιου τμήματος ασχολούνται με τη λειτουργία των διακομιστών, ενώ οι προγραμματιστές πλήρους στοίβας συνδυάζουν τις αρμοδιότητες των δύο άλλων εξειδικεύσεων.



Σχήμα 2.1: Επισκόπηση προγραμματισμού εμπρόσθιου τμήματος



Σχήμα 2.2: Επισκόπηση προγραμματισμού οπίσθιου τμήματος

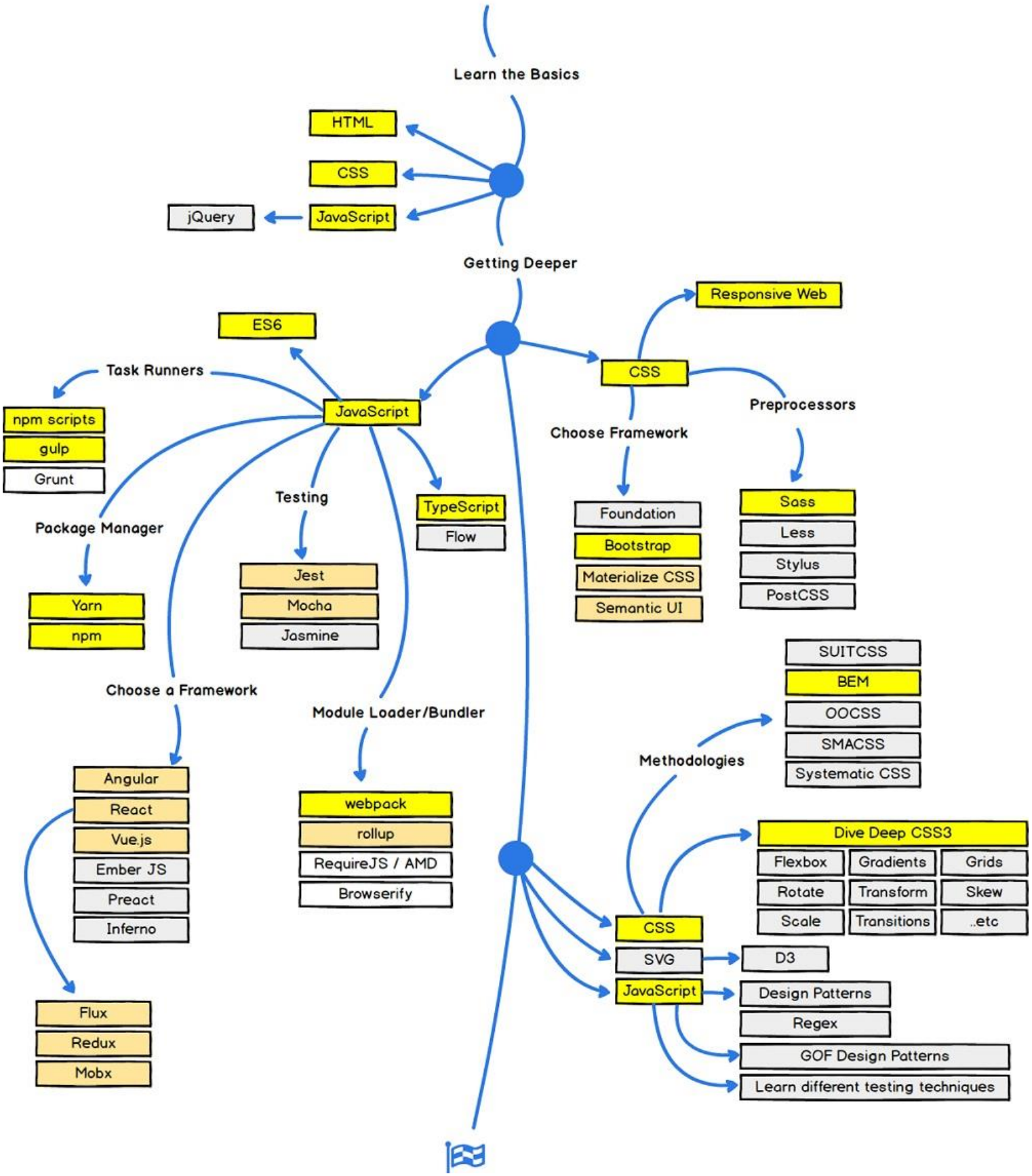
Παρόλο που ενδέχεται να εργάζονται πάνω στο ίδιο προϊόν (product), οι προγραμματιστές εμπρόσθιου τμήματος διαθέτουν ένα πολύ ξεχωριστό σύνολο δεξιοτήτων από εκείνο των προγραμματιστών οπίσθιου τμήματος. Οι δεξιότητες κάθε εξειδίκευσης παρουσιάζονται συνοπτικά παρακάτω, ενώ φαίνονται και στα Σχήματα 2.3 και 2.4[15]:

- Με εστίαση στο εμπρόσθιο τμήμα:
  - Γλώσσες σήμανσης και ιστού (markup and web languages) όπως HTML, CSS και JavaScript, και βοηθητικές βιβλιοθήκες (ancillary libraries) που χρησιμοποιούνται συνήθως σε αυτές τις γλώσσες όπως Sass (Syntactically awesome style sheets) ή jQuery
  - Ασύγχρονη διεκπεραίωση αιτημάτων (asynchronous request handling) και AJAX (Asynchronous JavaScript And XML)

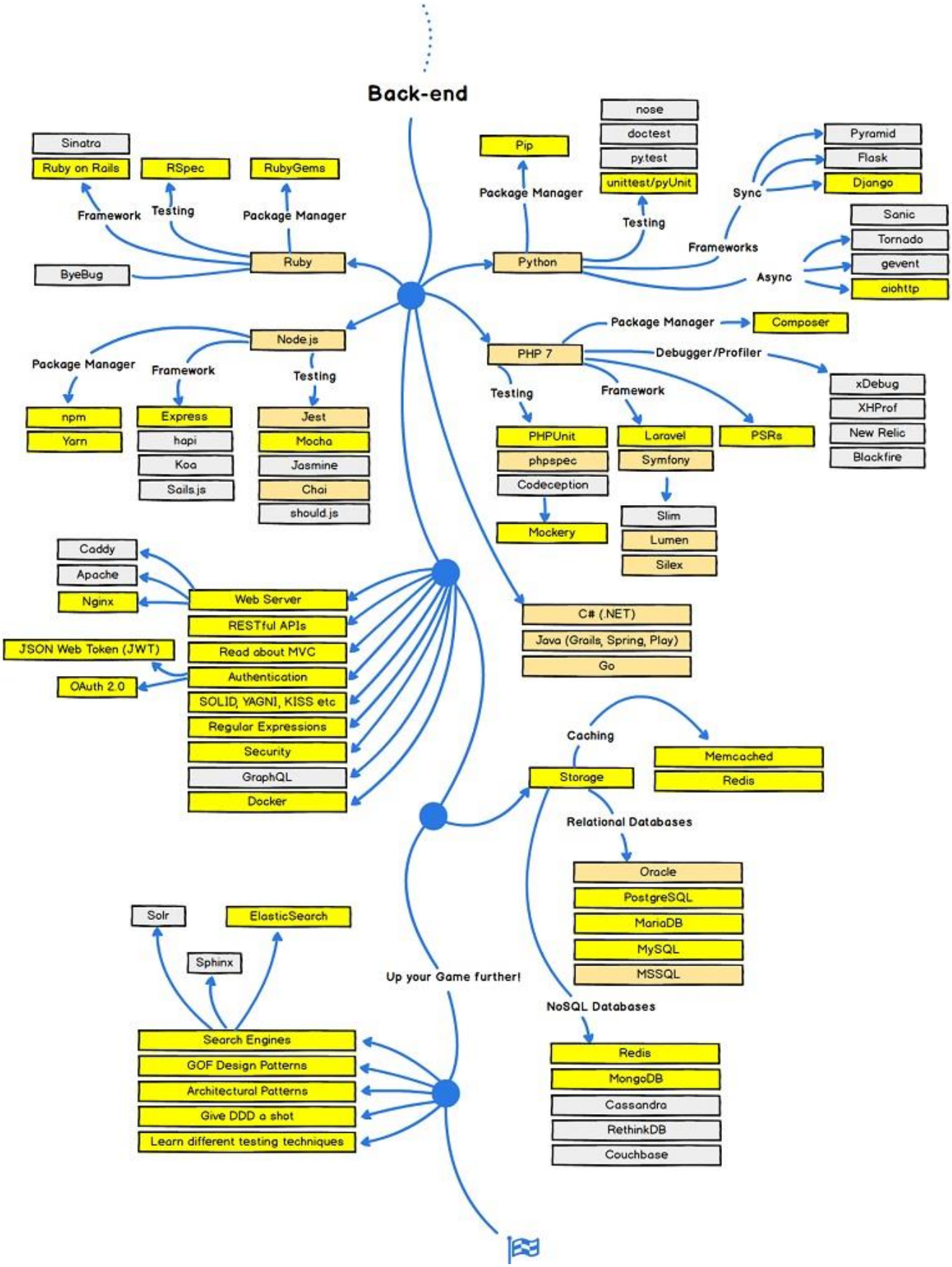
- Εφαρμογές μίας σελίδας (single-page applications) με χρήση πλαισίων όπως React, AngularJS ή Vue.js
- Απόδοση ιστού (web performance) με χρήση μετρικών (metrics), π.χ. FMP (First Meaningful Paint), TTI (Time To Interactive), FPS (Frames Per Second), κ.λπ.
- Αποκρίσιμος σχεδιασμός ιστού (responsive web design)
- Προβλήματα συμβατότητας με τους διάφορους περιηγητές (cross-browser compatibility) και αντιμετώπισή τους
- Δοκιμές από άκρο σε άκρο (end-to-end testing) με περιηγητή χωρίς GUI (headless browser)
- Αυτοματοποίηση κατασκευών (build automation) για τον μετασχηματισμό και δεσμοποίηση (bundling) αρχείων JavaScript, τη μείωση μεγέθους εικόνων, κ.λπ. με χρήση εργαλείων όπως webpack ή gulp
- Βελτιστοποίηση μηχανών αναζήτησης (search engine optimization)
- Θέματα προσβασιμότητας (accessibility concerns)
- Βασική χρήση εργαλείων επεξεργασίας εικόνων (image editing tools) όπως GIMP (GNU - GNU's Not Unix! - Image Manipulation Program) ή Photoshop
- Οπτική διεπαφή (visual interface)

- Με εστίαση στο οπίσθιο τμήμα:
  - Γλώσσες σεναρίων (scripting languages) όπως Node.js, PHP (PHP: Hypertext Preprocessor), Python, Ruby και Perl ή μεταγλωττισμένες γλώσσες (compiled languages) όπως C#, Java και Go
  - Αυτοματοποιημένα πλαίσια δοκιμών (automated testing frameworks) για τη χρησιμοποιούμενη γλώσσα
  - Πρόσβαση σε δεδομένα εφαρμογών (application data access)
  - Επιχειρησιακή λογική εφαρμογών (application business logic)
  - Διαχείριση βάσεων δεδομένων (database administration)
  - Κλιμακωσιμότητα (scalability)
  - Υψηλή διαθεσιμότητα (high availability)
  - Θέματα ασφάλειας (security concerns), αυθεντικοποίηση και εξουσιοδότηση (authentication and authorization)
  - Αρχιτεκτονική λογισμικού (software architecture)
  - Μετασχηματισμός δεδομένων (data transformation)
  - Μέθοδοι και λογισμικό αντιγράφων ασφαλείας (backup methods and software)

- Με εστίαση και στα δύο τμήματα:
  - Εργαλεία ελέγχου έκδοσης (version control tools) όπως GitHub, GitLab, Bitbucket, SourceForge ή Mercurial
  - Εργαλεία και πρωτόκολλα μεταφοράς αρχείων (file transfer tools and protocols) όπως FTP (File Transfer Protocol) ή rsync



Σχήμα 2.3: Επισκόπηση τεχνολογιών εμπρόσθιου τμήματος



Σχήμα 2.4: Επισκόπηση τεχνολογιών οπίσθιου τμήματος

2.1.3 Διαδικτυακός προγραμματισμός εμπρόσθιου τμήματος[16]

Ο διαδικτυακός προγραμματισμός εμπρόσθιου τμήματος (front-end web development) είναι η πρακτική της μετατροπής δεδομένων σε γραφική διεπαφή ώστε ο χρήστης να βλέπει και να αλληλεπιδρά ψηφιακά (digital interaction) με τα δεδομένα χρησιμοποιώντας HTML, CSS και JavaScript.

Διατίθενται διάφορα εργαλεία και πλατφόρμες (WordPress, Magento, κ.λπ.) που μπορούν να χρησιμοποιηθούν για την ανάπτυξη του εμπρόσθιου τμήματος ενός ιστοτόπου και η εύρεση εκείνων που ανταποκρίνονται καλύτερα σε συγκεκριμένες εργασίες (tasks) σηματοδοτεί τη διαφορά ενός καλά σχεδιασμένου, κλιμακώσιμου (scalable) ιστοτόπου από έναν που είναι επιρρεπής στην πειρατεία (hacking).

2.1.3.1 Γλώσσα σήμανσης υπερκειμένου

Η γλώσσα σήμανσης υπερκειμένου, HTML (HyperText Markup Language), αποτελεί τη ραχοκοκαλιά οποιασδήποτε διαδικασίας ανάπτυξης ιστοτόπου, χωρίς την οποία δεν μπορεί να υπάρξει καμία ιστοσελίδα. Υπερκείμενο (hypertext) σημαίνει ότι το κείμενο έχει συνδέσμους (links), που ονομάζονται υπερσύνδεσμοι (hyperlinks), ενσωματωμένους (embedded) σε αυτό. Όταν ένας χρήστης κάνει κλικ (clicks) σε μία λέξη ή μία φράση που έχει έναν υπερσύνδεσμο, ο τελευταίος θα φέρει μία άλλη ιστοσελίδα. Μία γλώσσα σήμανσης (markup language) υποδεικνύει ότι το κείμενο μπορεί να μετατραπεί σε εικόνες, πίνακες, συνδέσμους και άλλες αναπαραστάσεις. Ο κώδικας HTML είναι εκείνος που παρέχει ένα γενικό πλαίσιο για τον τρόπο εμφάνισης του ιστοτόπου. Η τελευταία έκδοση (version) της HTML ονομάζεται HTML5 και περιέχει νέους και αποδοτικούς τρόπους χειρισμού στοιχείων (elements) όπως τα αρχεία βίντεο και ήχου (video and audio files).

2.1.3.2 Διαδοχικά φύλλα ύφους

Τα διαδοχικά φύλλα ύφους, CSS (Cascading Style Sheets), ελέγχουν την παρουσίαση του ιστοτόπου και του επιτρέπουν να έχει τη δική του μοναδική εμφάνιση. Αυτό επιτυγχάνεται διατηρώντας φύλλα ύφους (style sheets), τα οποία «κάθονται» πάνω από άλλους κανόνες ύφους (style rules), και πυροδοτούνται (triggered) βάσει άλλων εισόδων (inputs) όπως το μέγεθος και η ανάλυση οθόνης συσκευής (device screen size and resolution).

2.1.3.3 JavaScript

Σε αντίθεση με το δηλωτικό γλωσσικό μοντέλο (declarative language model) της HTML, η JavaScript είναι μία προστακτική γλώσσα προγραμματισμού που βασίζεται σε συμβάντα (event-based imperative programming language) και χρησιμοποιείται για τον μετασχηματισμό μίας στατικής σελίδας HTML (static HTML page) σε μία δυναμική διεπαφή (dynamic interface). Ο κώδικας JavaScript μπορεί να χρησιμοποιήσει το μοντέλο αντικειμένων εγγράφου, DOM (Document Object Model), το οποίο παρέχεται από το πρότυπο HTML (HTML standard), για να χειρίζεται μία ιστοσελίδα ως απόκριση σε συμβάντα όπως η είσοδος χρήστη (user input).

Χρησιμοποιώντας την τεχνική AJAX (Asynchronous JavaScript And XML), ο κώδικας JavaScript μπορεί να ανακτήσει ενεργά περιεχόμενο (content) από τον ιστό (ανεξάρτητα από την αρχική ανάκτηση σελίδας HTML) και να αντιδράσει στα συμβάντα από πλευράς διακομιστή (server-side events), προσθέτοντας έτσι μία πραγματικά δυναμική φύση στην εμπειρία που προσφέρει η ιστοσελίδα (web page experience).

2.1.4 Διαδικτυακός προγραμματισμός οπίσθιου τμήματος[17]

Στην ανάπτυξη λογισμικού (software development), μία στοίβα λύσεων (solution stack) ή στοίβα λογισμικού (software stack) είναι ένα σύνολο υποσυστημάτων ή συνιστωσών λογισμικού (software subsystems or components) που απαιτούνται για τη δημιουργία μίας ολοκληρωμένης πλατφόρμας (complete platform), ώστε να μην χρειάζεται επιπρόσθετο λογισμικό για την υποστήριξη των εφαρμογών. Οι εφαρμογές λέγεται ότι «τρέχουν στην κορυφή της πλατφόρμας» ή ότι απλά «τρέχουν στην πλατφόρμα».

Στην περίπτωση του διαδικτυακού προγραμματισμού οπίσθιου τμήματος (back-end web development), η στοίβα ορίζεται ως το λειτουργικό σύστημα, ο διακομιστής ιστού, η βάση δεδομένων και η γλώσσα προγραμματισμού που απαιτούνται για την ανάπτυξη εφαρμογών ιστού. Μία άλλη εκδοχή στοίβας λογισμικού είναι: λειτουργικό σύστημα, μεσισμικό (middleware), βάση δεδομένων και εφαρμογές. Συνήθως, οι συνιστώσες μίας στοίβας λογισμικού αναπτύσσονται από διαφορετικούς προγραμματιστές, ανεξάρτητα η μία από την άλλη.

Ορισμένες συνιστώσες λογισμικού επιλέγονται πολύ συχνά μαζί για να συνθέσουν μία στοίβα με αποτέλεσμα η τελευταία να αναφέρεται με ένα όνομα που αντιπροσωπεύει το σύνολο, αντί να ονομάζει τα μέρη. Το όνομα αυτό είναι κατά βάση ένα ακρωνύμιο που αντιπροσωπεύει τις επιμέρους συνιστώσες.

Δύο αρκετά διαδεδομένες στοίβες λογισμικού είναι η LAMP (Linux-Apache-MySQL/MariaDB-Perl/PHP/Python) και η MEAN (MongoDB-Express.js-AngularJS-Node.js), οι οποίες διαθέτουν πολυάριθμες παραλλαγές:

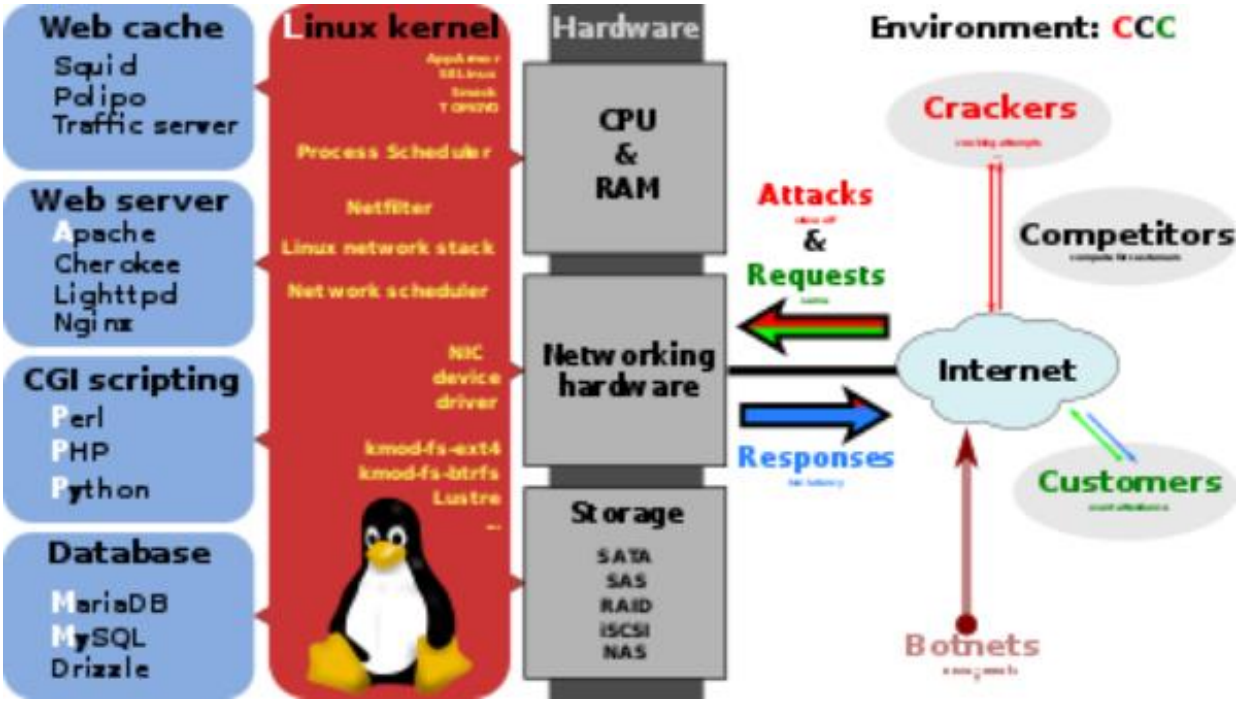
- LAPP (Linux-Apache-PostgreSQL-Perl/PHP/Python)
- LNMP (Linux-Nginx-MySQL/MariaDB-Perl/PHP/Python)
- LLMP (Linux-Lighttpd-MySQL/MariaDB-Perl/PHP/Python)
- MAMP (MacOS-Apache-MySQL/MariaDB-Perl/PHP/Python)
- WAMP (Windows-Apache-MySQL/MariaDB-Perl/PHP/Python)
- XAMPP (Linux/Windows/MacOS-Apache-MySQL/MariaDB-PHP-Perl)
- MERN (MongoDB-Express.js-React.js-Node.js)
- MEVN (MongoDB-Express.js-Vue.js-Node.js)
- MEEN (MongoDB-Express.js-Ember.js-Node.js)

2.2 Στοίβα λογισμικού LAMP

2.2.1 Επισκόπηση[18]

Η στοίβα υπηρεσιών ιστού (web service stack) LAMP, όπως φαίνεται στο Σχήμα 2.5, συντίθεται από τέσσερεις συνιστώσες λογισμικού ανοικτού κώδικα και το όνομά της αποτελεί ακρωνύμιο των ονομάτων τους: το λειτουργικό σύστημα Linux (Linux operating system), ο διακομιστής HTTP Apache (Apache HTTP Server), ένα εκ των συστημάτων διαχείρισης σχεσιακών βάσεων δεδομένων MySQL ή MariaDB (MySQL/MariaDB Relational DataBase Management System - RDBMS) και μία εκ των γλωσσών προγραμματισμού Perl, PHP ή Python (Perl/PHP/Python programming language).

Ως στοίβα λογισμικού, η LAMP είναι κατάλληλη για την κατασκευή δυναμικών ιστοτόπων (dynamic websites), όπως η Wikipedia και άλλα έργα του ιδρύματος Wikimedia (Wikimedia Foundation projects), και εφαρμογών ιστού (web applications) καθώς είναι ευέλικτη (flexible), προσαρμόσιμη (customizable), εύκολα αναπτυσσόμενη (easy to develop), εύκολα επιστρατεύσιμη στον ιστό (easy to deploy) και ασφαλής (secure), ενώ συνοδεύεται και από μία τεράστια κοινότητα υποστήριξης (support community) αφού είναι στοίβα ανοικτού κώδικα[19].



Σχήμα 2.5: Επισκόπηση της δομής και λειτουργίας της στοίβας LAMP

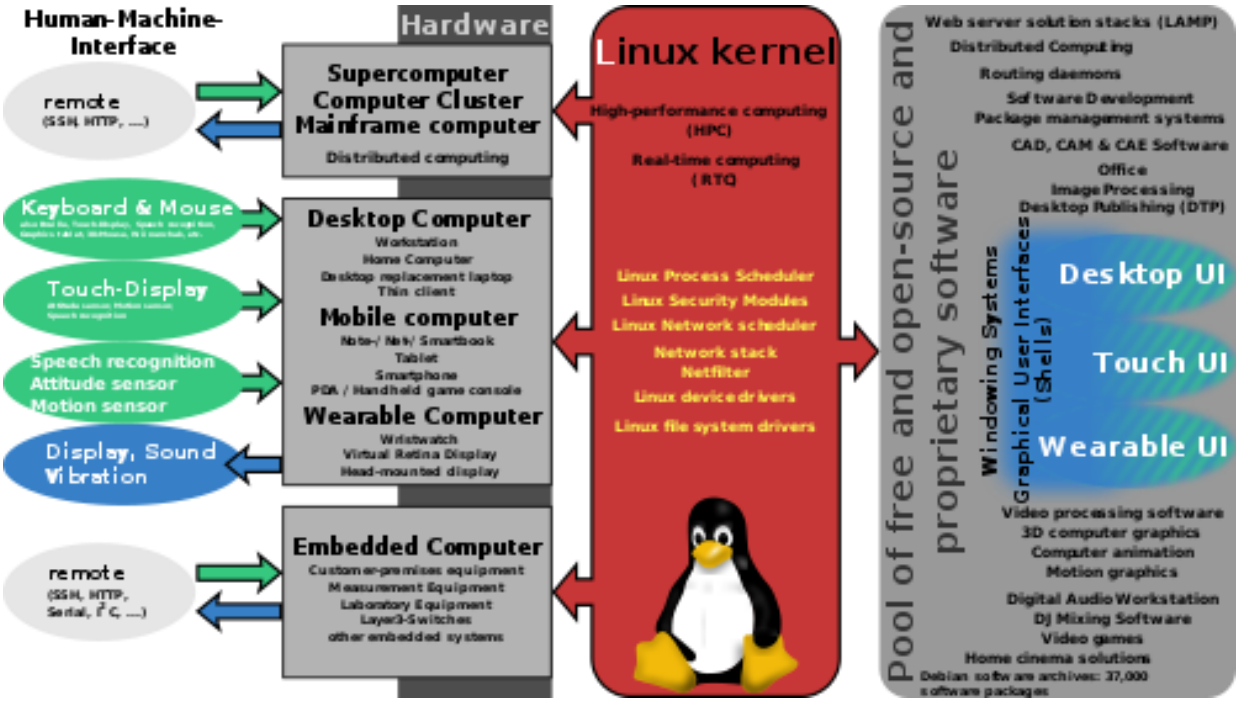
2.2.2 Συνιστώσες λογισμικού

Η εκδοχή της στοίβας LAMP που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής ιστού SLUB είναι η Linux-Apache-MySQL-Python.

2.2.2.1 Λειτουργικό σύστημα Linux[20]

Το Linux είναι ένα λειτουργικό σύστημα υπολογιστών τύπου Unix (Unix-like computer operating system) που συναρμολογείται (assembled) με βάση το μοντέλο ανάπτυξης και διανομής ελεύθερου και ανοικτού κώδικα λογισμικού (free and open-source software development and distribution). Οι περισσότερες διανομές Linux (Linux distributions), ως συλλογές λογισμικού που βασίζονται γύρω από τον πυρήνα του Linux (Linux kernel) και συχνά γύρω από ένα σύστημα διαχείρισης πακέτων (package management system), παρέχουν πλήρεις διατάξεις LAMP (complete LAMP setups) μέσω των πακέτων τους.

Το Linux αναπτύχθηκε αρχικά για προσωπικούς υπολογιστές (personal computers) με βάση την αρχιτεκτονική Intel x86, αλλά από τότε έχει μεταφερθεί (ported) σε περισσότερες πλατφόρμες από οποιοδήποτε άλλο λειτουργικό σύστημα. Το Linux είναι το κυρίαρχο λειτουργικό σύστημα σε διακομιστές και άλλα μεγάλα υπολογιστικά συστήματα (big iron systems) όπως οι κεντρικοί υπολογιστές (mainframe computers) και το μόνο λειτουργικό σύστημα που χρησιμοποιείται στους 500 κορυφαίους υπερυπολογιστές (TOP500 supercomputers) από τον Νοέμβριο του 2017 και έπειτα. Η σχετική επιτυχία του Linux αποδίδεται κατά κύριο λόγο στην ασφάλεια, την αξιοπιστία, το χαμηλό κόστος και την ελευθερία από το κλείδωμα σε προμηθευτή (vendor lock-in) που παρέχει.



Σχήμα 2.6: Επισκόπηση της δομής και των χρήσεων του πυρήνα του Linux

Όπως φαίνεται και στο Σχήμα 2.6, το Linux τρέχει επίσης σε ενσωματωμένα συστήματα (embedded systems), δηλαδή συσκευές των οποίων το λειτουργικό σύστημα είναι ενσωματωμένο στο υλικολογισμικό (firmware) και ειδικά προσαρμοσμένο στο σύστημα, όπως δρομολογητές (routers), συστήματα ελέγχου αυτοματισμών (automation controls), τηλεοράσεις, ψηφιακές συσκευές εγγραφής

βίντεο (digital video recorders), κονσόλες βιντεοπαιχνιδιών (video game consoles) και έξυπνα ρολόγια (smartwatches). Πολλά έξυπνα κινητά τηλέφωνα (smartphones) και υπολογιστές-ταμπλέτες (tablet computers) τρέχουν Android και άλλα παράγωγα του Linux (Linux derivatives). Λόγω της κυριαρχίας του Android στα smartphones, το Linux έχει τη μεγαλύτερη εγκατεστημένη βάση (installed base) από όλα τα λειτουργικά συστήματα γενικού σκοπού (general-purpose operating systems).

Το Linux είναι το πιο ευρέως χρησιμοποιούμενο λειτουργικό σύστημα σε διακομιστές ιστού. Σύμφωνα με τη W3Techs, τον Νοέμβριο του 2016 υπολογίστηκε ότι 66,7% των διακομιστών ιστού τρέχουν σε Linux/Unix και 33,4% τρέχουν σε Microsoft Windows. Οι πιο δημοφιλείς διανομές είναι οι Debian και Ubuntu, ενώ ακολουθούν οι RHEL (Red Hat Enterprise Linux), Fedora και CentOS.

2.2.2.2 Διακομιστής HTTP Apache[21]

Ο Apache είναι ελεύθερο (free), ανοικτού κώδικα (open-source) και ανεξάρτητο πλατφόρμας (cross-platform) λογισμικό διακομιστή ιστού (web server software). Είναι υψηλά κλιμακώσιμος (highly scalable), δηλαδή μπορεί εύκολα να χειριστεί παραπάνω από 10.000 ταυτόχρονες συνδέσεις, και υποστηρίζει μία ποικιλία χαρακτηριστικών (features), πολλά από τα οποία υλοποιούνται ως μεταγλωττισμένες μονάδες (compiled modules) που επεκτείνουν την κύρια λειτουργικότητά (core functionality) του. Αυτά μπορούν να κυμαίνονται από υποστήριξη γλωσσών προγραμματισμού από πλευράς διακομιστή (server-side programming language support), όπως Perl, Python, Tcl (Tool Command Language) και PHP, και χειρισμό στατικών αρχείων (static file handling), όπως σελίδες HTML και εικόνες, έως και σχήματα αυθεντικοποίησης (authentication schemes).

Μεταξύ άλλων, ο Apache περιλαμβάνει τις εξής μονάδες:
- Αυθεντικοποίησης: mod_access, mod_auth, mod_digest και mod_auth_digest
- Υποστήριξης SSL/TLS (Secure Sockets Layer / Transport Layer Security): mod_ssl
- Διαμεσολάβησης (proxy): mod_proxy
- Επανεγγραφής URL (URL rewriting): mod_rewrite
- Προσαρμοσμένης καταγραφής (custom logging): mod_log_config
- Φιλτραρίσματος (filtering): mod_include και mod_ext_filter
- Συμπίεσης (compression): mod_gzip
- Ανίχνευσης και πρόληψης εισβολών (intrusion detection and prevention): ModSecurity

Αντί να υλοποιεί μία μόνο αρχιτεκτονική, ο Apache παρέχει μία ποικιλία μονάδων πολλαπλών διεργασιών (MultiProcessing Modules - MPMs), οι οποίες του επιτρέπουν να εκτελείται είτε σε λειτουργία βασισμένη σε διεργασίες (process-based mode) είτε σε υβριδική λειτουργία (hybrid mode) που βασίζεται σε πολλαπλές διεργασίες και πολλαπλά νήματα ανά διεργασία (threads per process) είτε σε συμβανυβριδική λειτουργία (event-hybrid mode) που συνδυάζει την υβριδική λειτουργία με έναν ασύγχρονο βρόχο βασισμένο σε συμβάντα (asynchronous event-based loop) ώστε ο διακομιστής να ανταποκρίνεται καλύτερα στις απαιτήσεις κάθε συγκεκριμένης υποδομής (infrastructure). Επομένως, η επιλογή της MPM και της διαμόρφωσης (configuration) είναι σημαντική. Όπου απαιτούνται συμβιβασμοί στην απόδοση

(performance), ο Apache έχει σχεδιαστεί για να μειώνει τη χρονοκαθυστέρηση (latency) και να αυξάνει τη ρυθμαπόδοση (throughput) σε σχέση με τον απλό χειρισμό περισσότερων αιτημάτων, εξασφαλίζοντας έτσι συνεπή και αξιόπιστη επεξεργασία των αιτημάτων εντός εύλογων χρονικών πλαισίων (time-frames).

Ο Apache είναι ο πιο δημοφιλής διακομιστής ιστού στο δημόσιο Διαδίκτυο (public Internet). Τον Αύγουστο του 2018 υπολογίστηκε ότι εξυπηρετούσε το 39% όλων των ενεργών ιστοτόπων και το 35% του κορυφαίου εκατομμυρίου ιστοτόπων (top million websites).

2.2.2.3 Σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων MySQL[22]

Η MySQL είναι ένα πολυχρηστικό (multi-user), πολυνηματικό (multithreaded), ανεξάρτητο πλατφόρμας (cross-platform) και ανοικτού κώδικα (open-source) σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational DataBase Management System - RDBMS) που χρησιμοποιεί τη δηλωτική (declarative) γλώσσα δομημένων επερωτήσεων, SQL (Structured Query Language), για την επερώτηση, τον χειρισμό (εισαγωγή, ενημέρωση και διαγραφή - insertion, update and deletion), τον ορισμό (δημιουργία και τροποποίηση σχήματος - schema creation and modification) και τον έλεγχο πρόσβασης (access control) δεδομένων[23].

Μεταξύ των κύριων χαρακτηριστικών (major features) που διατίθενται στη MySQL (έκδοση 5.6) είναι:

- Ένα ευρύ υποσύνολο της ANSI (American National Standards Institute) SQL-99 και σχετικές επεκτάσεις (extensions)
- Αποθηκευμένες διαδικασίες (stored procedures) χρησιμοποιώντας μία διαδικαστική γλώσσα (procedural language) που είναι συμβατή με το πρότυπο SQL/PSM (SQL / Persistent Stored Modules standard)
- Εναύσματα (triggers)
- Δρομείς (cursors)
- Ενημερώσιμες όψεις (updatable views)
- Επιγραμμική (online) γλώσσα ορισμού δεδομένων (Data Definition Language - DDL) όταν χρησιμοποιείται η μηχανή αποθήκευσης InnoDB (InnoDB storage engine)
- Σχήμα πληροφοριών (information schema)
- Ένα σύνολο επιλογών κατάστασης λειτουργίας SQL (SQL Mode options) για τον έλεγχο της συμπεριφοράς κατά την εκτέλεση (runtime behavior), συμπεριλαμβανομένης μίας αυστηρής κατάστασης λειτουργίας (strict mode) για την καλύτερη συμμόρφωση με τα πρότυπα SQL (SQL standards)
- Συναλλαγές (transactions) με σημεία αποθήκευσης (savepoints) όταν χρησιμοποιείται η προεπιλεγμένη μηχανή αποθήκευσης InnoDB ή η μηχανή αποθήκευσης συστοιχίας NDB (Network DataBase, NDB, Cluster storage engine)
- Συμμόρφωση με τις ιδιότητες ACID (Atomicity, Consistency, Isolation and Durability properties) όταν χρησιμοποιούνται οι μηχανές αποθήκευσης InnoDB και συστοιχίας NDB
- Προσωρινή αποθήκευση επερωτήσεων στην κρυφή μνήμη (query caching)
- Εμφωλευμένες επερωτήσεις - υποερωτήματα - SELECT (SELECT nested queries - subqueries)

- Ευρετηρίαση και αναζήτηση πλήρους κειμένου (full-text indexing and searching)
- Βιβλιοθήκη ενσωματωμένων βάσεων δεδομένων (embedded database library)
- Πολλαπλές εγγενείς (native) μηχανές αποθήκευσης όπως InnoDB, MyISAM (Indexed Sequential Access Method), συγχώνευσης (Merge), μνήμης/σωρού (Memory/Heap), ομοσπονδιακή (Federated), αρχείου (Archive), CSV (Comma-Separated Values), μαύρης τρύπας (Blackhole) και συστοιχίας NDB, οι οποίες επιτρέπουν στον διαχειριστή (administrator) να επιλέξει αυτήν που είναι πιο αποτελεσματική για κάθε πίνακα (table) στην εφαρμογή (όταν χρησιμοποιούνται κάποιες μηχανές αποθήκευσης διαφορετικές από την προεπιλεγμένη - default - InnoDB, η MySQL δεν συμμορφώνεται με το πλήρες πρότυπο SQL για μερικές από τις υλοποιημένες λειτουργίες, συμπεριλαμβανομένων των αναφορών εξωτερικού κλειδιού - foreign key references)
- Υποστήριξη SSL
- Υποστήριξη Unicode
- Δημιουργία αντιγράφων ασφαλείας (backing up) από όλες τις μηχανές αποθήκευσης μέσω του εργαλείου mysqldump

Παρόλο που η MySQL ξεκίνησε ως εναλλακτική λύση χαμηλών προδιαγραφών (low-end alternative) αντί πιο ισχυρών ιδιόκτητων (proprietary) βάσεων δεδομένων, έχει σταδιακά εξελιχθεί ώστε να υποστηρίζει και ανάγκες υψηλότερης κλίμακας. Εξακολουθεί να χρησιμοποιείται αρκετά συχνά σε μικρής έως μεσαίας κλίμακας παρατάξεις ενός διακομιστή (single-server deployments), είτε ως συνιστώσα σε μία εφαρμογή ιστού που βασίζεται στη στοίβα LAMP είτε ως μεμονωμένος διακομιστής βάσεων δεδομένων (standalone database server). Μεγάλο μέρος της δημοφιλίας της MySQL προέρχεται από τη σχετική της απλότητα και ευκολία στη χρήση, η οποία γίνεται εφικτή μέσω ενός οικοσυστήματος εργαλείων ανοικτού κώδικα (open-source tools) όπως το phpMyAdmin. Σε μεσαία κλίμακα, η MySQL μπορεί να κλιμακωθεί μέσω επιστράτευσης (deployment) σε πιο ισχυρό υλισμικό (hardware) όπως, για παράδειγμα, συμβαίνει σε έναν πολυεπεξεργαστικό διακομιστή (multi-processor server) με αρκετά gigabytes μνήμης.

Η MySQL χρησιμοποιείται από πολλές εφαρμογές ιστού που οδηγούνται από βάσεις δεδομένων (database-driven web applications) όπως Drupal, Joomla!, phpBB (PHP Bulletin Board) και WordPress, καθώς και από πολλούς δημοφιλείς ιστοτόπους όπως Facebook, Twitter, Flickr και YouTube. Η MySQL έχει λάβει θετικές αξιολογήσεις (reviews) με τους αξιολογητές να παρατηρούν ότι «αποδίδει εξαιρετικά καλά στη μέση περίπτωση (average case)» και ότι οι «διεπαφές προγραμματιστών (developer interfaces) υπάρχουν, ενώ η τεκμηρίωση (documentation), η ανατροφοδότηση (feedback) στον πραγματικό κόσμο μέσω ιστοτόπων, κ.λπ. είναι πάρα πολύ καλές». Έχει επίσης δοκιμαστεί ότι είναι ένας «γρήγορος, σταθερός και αληθινά πολυχρηστικός, πολυνηματικός διακομιστής βάσεων δεδομένων SQL».

2.2.2.4 Γλώσσα προγραμματισμού Python[24]

Η Python είναι μία διερμηνευμένη (interpreted), υψηλού επιπέδου (high-level) και γενικού σκοπού γλώσσα προγραμματισμού (general-purpose programming language). Η φιλοσοφία σχεδίασής της (design philosophy) δίνει έμφαση στην αναγνωσιμότητα του κώδικα (code readability) και χρησιμοποιεί στοίχιση μη-εκτυπώσιμων χαρακτήρων (whitespace indentation), γνωστή και ως κανόνας του οφσάιντ (offside rule). Οι

γλωσσικές κατασκευές (language constructs) και η αντικειμενοστρεφής προσέγγισή της (object-oriented approach) στοχεύουν στο να βοηθούν τους προγραμματιστές να γράφουν σαφή και λογικό κώδικα για μικρής και μεγάλης κλίμακας έργα (projects).

Οι διερμηνείς (interpreters) της Python είναι διαθέσιμοι για πολλά λειτουργικά συστήματα. Ο προεπιλεγμένος και πιο ευρέως χρησιμοποιούμενος διερμηνέας της είναι ο CPython, ο οποίος είναι μία ανοικτού κώδικα (open-source) υλοποίηση αναφοράς (reference implementation) της γλώσσας γραμμένη σε C και Python. Πολλά λειτουργικά συστήματα περιλαμβάνουν την Python ως τυπική συνιστώσα (standard component). Διατίθεται με το AmigaOS 4, το FreeBSD (ως πακέτο), το NetBSD (Berkeley Software Distribution), το OpenBSD (ως πακέτο), το macOS και τις περισσότερες διανομές Linux, και μπορεί να χρησιμοποιηθεί από τη γραμμή εντολών / τερματικό (command line / terminal).

Η Python χρησιμοποιεί δυναμικό σύστημα τύπων (dynamic typing) και έναν συνδυασμό μέτρησης αναφορών (reference counting) και συλλογής σκουπιδιών με ανίχνευση κύκλων (cycle-detecting garbage collection) για τη διαχείριση μνήμης (memory management). Διαθέτει επίσης δυναμική επίλυση ονομάτων **(**dynamic name resolution**)** μέσω καθυστερημένης δέσμευσης (late binding), η οποία δεσμεύει τα ονόματα μεθόδων και μεταβλητών (method and variable names) κατά την εκτέλεση προγραμμάτων (program execution). Όπως και άλλες δυναμικές γλώσσες (dynamic languages), η Python χρησιμοποιείται συχνά ως γλώσσα σεναρίων (scripting language) καθώς και σε ένα ευρύ φάσμα πλαισίων χωρίς σενάρια (non-scripting contexts).

Η Python υποστηρίζει πολλαπλά προγραμματιστικά υποδείγματα (programming paradigms). Υποστηρίζει πλήρως τον αντικειμενοστρεφή (object-oriented) και τον δομημένο προγραμματισμό (structured programming), και πολλά από τα χαρακτηριστικά της (features) υποστηρίζουν τον συναρτησιακό (functional) και τον πτυχοστρεφή προγραμματισμό (aspect-oriented programming), συμπεριλαμβανομένων του μεταπρογραμματισμού (metaprogramming) και των μετααντικειμένων / μαγικών μεθόδων (metaobjects / magic methods). Πολλά άλλα υποδείγματα υποστηρίζονται μέσω επεκτάσεων (extensions), συμπεριλαμβανομένων του σχεδιασμού βάσει συμβάσεων (design by contract) και του λογικού προγραμματισμού (logic programming).

Η Python είναι μία γλώσσα μικρού πυρήνα (small core language) με μία μεγάλη πρότυπη βιβλιοθήκη (standard library) και έναν εύκολα επεκτάσιμο διερμηνέα (easily extensible interpreter). Αυτή η συμπαγής δομοστοιχείωση (modularity) έχει καταστήσει τη γλώσσα ιδιαίτερα δημοφιλή ως μέσο προσθήκης προγραμματιζόμενων διεπαφών (programmable interfaces) σε υπάρχουσες εφαρμογές. Η μεγάλη πρότυπη βιβλιοθήκη της, η οποία αποτελεί ένα από τα δυνατότερα στοιχεία της, παρέχει εργαλεία κατάλληλα για πολλές εργασίες (tasks). Από τον Μάρτιο του 2018, το ευρετήριο πακέτων Python, PyPI (Python Package Index), το επίσημο αποθετήριο (repository) για λογισμικό Python φτιαγμένο από τρίτους (third-party Python software), περιέχει πάνω από 130.000 πακέτα με ευρύ φάσμα λειτουργικότητας (functionality), όπως:

- Γραφικές διεπαφές χρήστη (GUI)
- Πλαίσια ιστού (web frameworks)
- Πολυμέσα (multimedia)
- Βάσεις δεδομένων (databases)

- Δικτύωση (networking)
- Πλαίσια δοκιμών (test frameworks)
- Αυτοματοποίηση (automation)
- Ιστοσυγκομιδή (web harvesting)
- Τεκμηρίωση (documentation)
- Διαχείριση συστημάτων (system administration)
- Επιστημονική υπολογιστική (scientific computing)
- Επεξεργασία κειμένων (text processing)
- Επεξεργασία εικόνων (image processing)
- Τεχνητή νοημοσύνη (artificial intelligence)
- Ασφάλεια πληροφοριακών συστημάτων (information security)

Η Python επιδιώκει μία απλούστερη, συνοπτικότερη σύνταξη και γραμματική, ενώ δίνει στους προγραμματιστές ελευθερία επιλογών κατά τη συγγραφή κώδικα. Σε αντίθεση με το σύνθημα (motto) της Perl «υπάρχουν περισσότεροι από ένας τρόποι να το κάνεις», η Python ακολουθεί τη σχεδιαστική φιλοσοφία «πρέπει να υπάρχει ένας - και κατά προτίμηση μόνο ένας - προφανής τρόπος να το κάνεις». Οι προγραμματιστές της Python προσπαθούν να αποφεύγουν την πρόωρη βελτιστοποίηση (premature optimization) και ως εκ τούτου απορρίπτουν τις επιδιορθώσεις (patches) σε μη-κρίσιμα κομμάτια (non-critical parts) της υλοποίησης αναφοράς CPython που θα πρόσφεραν οριακές αυξήσεις στην ταχύτητα εις βάρος της σαφήνειας.

Σύμφωνα με τον δείκτη προγραμματιστικής κοινότητας TIOBE (TIOBE Programming Community index), η Python κατατάσσεται σταθερά στις δέκα πιο δημοφιλείς γλώσσες προγραμματισμού από το 2003 και έπειτα, ενώ από τον Δεκέμβριο του 2018 είναι η τρίτη πιο δημοφιλής γλώσσα (πίσω από τις Java και C). Επιλέχθηκε επίσης ως η γλώσσα προγραμματισμού της χρονιάς (Programming Language of the Year) το 2007, το 2010 και το 2018. Τέλος, χρησιμοποιείται από πολλές μεγάλες οργανώσεις όπως Wikipedia, Google, Yahoo!, CERN (European Organization for Nuclear Research), NASA (National Aeronautics and Space Administration), Facebook, Amazon, Instagram, Spotify και Reddit, καθώς και από δημοφιλείς εφαρμογές όπως Ansible, BitTorrent, Celery, Dropbox και Mercurial.

2.3 Πλαίσιο ιστού Django[25]

2.3.1 Επισκόπηση

Το Django είναι ένα ελεύθερο (free) και ανοικτού κώδικα (open-source) πλαίσιο ιστού που βασίζεται στην Python (Python-based web framework) και το οποίο ακολουθεί το αρχιτεκτονικό μόρφημα MTV (Model-Template-View architectural pattern).

Ο πρωταρχικός στόχος του Django είναι να διευκολύνει τη δημιουργία πολύπλοκων ιστοτόπων που οδηγούνται από βάσεις δεδομένων (database-driven websites). Το πλαίσιο δίνει έμφαση στις δυνατότητες επαναχρησιμοποίησης (reusability) και «προσάρτησης» ("pluggability") συνιστωσών, στη χρήση λιγότερου κώδικα, στη χαμηλή σύζευξη (low coupling), στην ταχεία ανάπτυξη (rapid development) και στην αρχή της μη-επαναληψιμότητας (Don't Repeat Yourself, DRY, principle). Η Python χρησιμοποιείται σε όλη την έκταση του πλαισίου, ακόμα και για αρχεία ρυθμίσεων

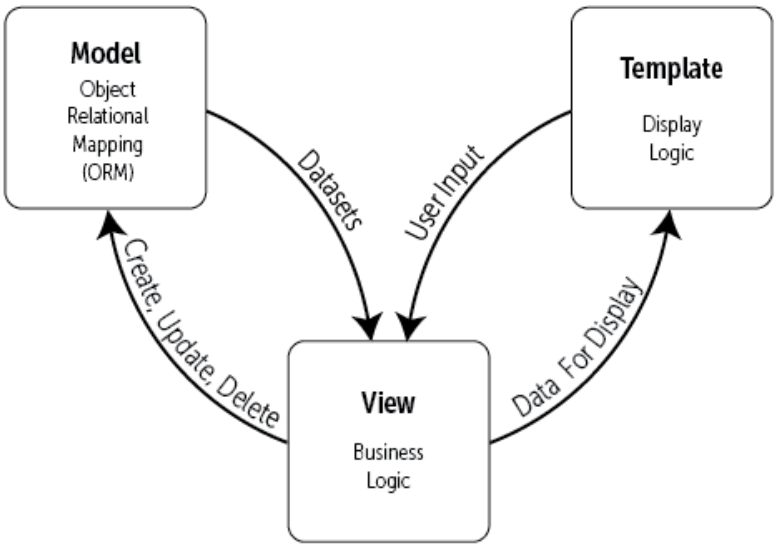
(settings files) και μοντέλα δεδομένων (data models). Το Django παρέχει επίσης μία προαιρετική διαχειριστική διεπαφή (administrative interface) για τη δημιουργία, ανάγνωση, ενημέρωση και διαγραφή (Create, Read, Update and Delete - CRUD) δεδομένων, η οποία δημιουργείται δυναμικά μέσω ενδοσκόπησης (introspection) και διαμορφώνεται (configured) μέσω μοντέλων διαχείρισης (admin models).

Ορισμένοι γνωστοί ιστότοποι που χρησιμοποιούν το Django είναι οι PBS (Public Broadcasting Service), Instagram, Mozilla, The Washington Times, Disqus, Bitbucket και Nextdoor.

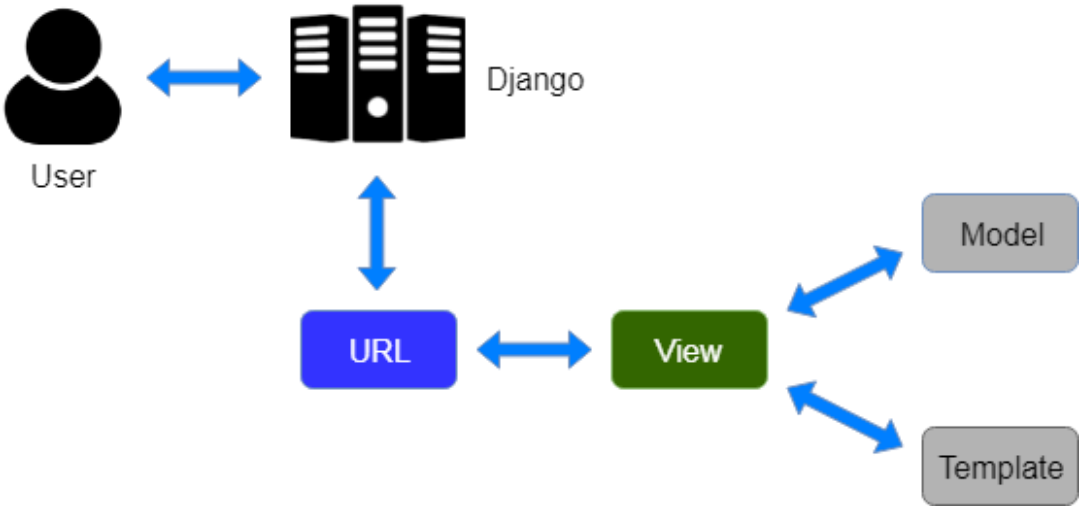
2.3.1.1 Αρχιτεκτονικό μόρφημα MTV[26]

Το Django είναι κατά βάση ένα πλαίσιο μοντέλου-όψης-ελεγκτή, MVC (Model-View-Controller), όπου οι συνιστώσες «όψη» ("view") και «ελεγκτής» ("controller") αντικαθίστανται από τις έννοιες «πρότυπο» ("template") και «όψη» αντίστοιχα. Το Django, δηλαδή, αποτελεί ένα πλαίσιο μοντέλου-προτύπου-όψης, MTV (Model-Template-View). Οι αλληλεπιδράσεις (interactions) μεταξύ των συνιστωσών του μορφήματος MTV καθώς και η σχετική ροή ελέγχου (control flow) στο πλαίσιο Django παρουσιάζονται συνοπτικά στα Σχήματα 2.7[27] και 2.8[28].

Σύμφωνα με την ερμηνεία που δίνει η ομάδα ανάπτυξης (development team) του Django για το μόρφημα MVC, η όψη περιγράφει τα δεδομένα που παρουσιάζονται στον χρήστη ανεξαρτήτως του τρόπου παρουσίασής τους. Υπάρχει, δηλαδή, μία λεπτή διάκριση μεταξύ περιεχόμενου (content) και παρουσίασης (presentation): η όψη περιγράφει το ποια δεδομένα βλέπει ο χρήστης, όχι το πώς τα βλέπει.



Σχήμα 2.7: Διάγραμμα αλληλεπιδράσεων μεταξύ των συνιστωσών του μορφήματος MTV



Σχήμα 2.8: Διάγραμμα ροής ελέγχου του μορφήματος MTV στο πλαίσιο Django

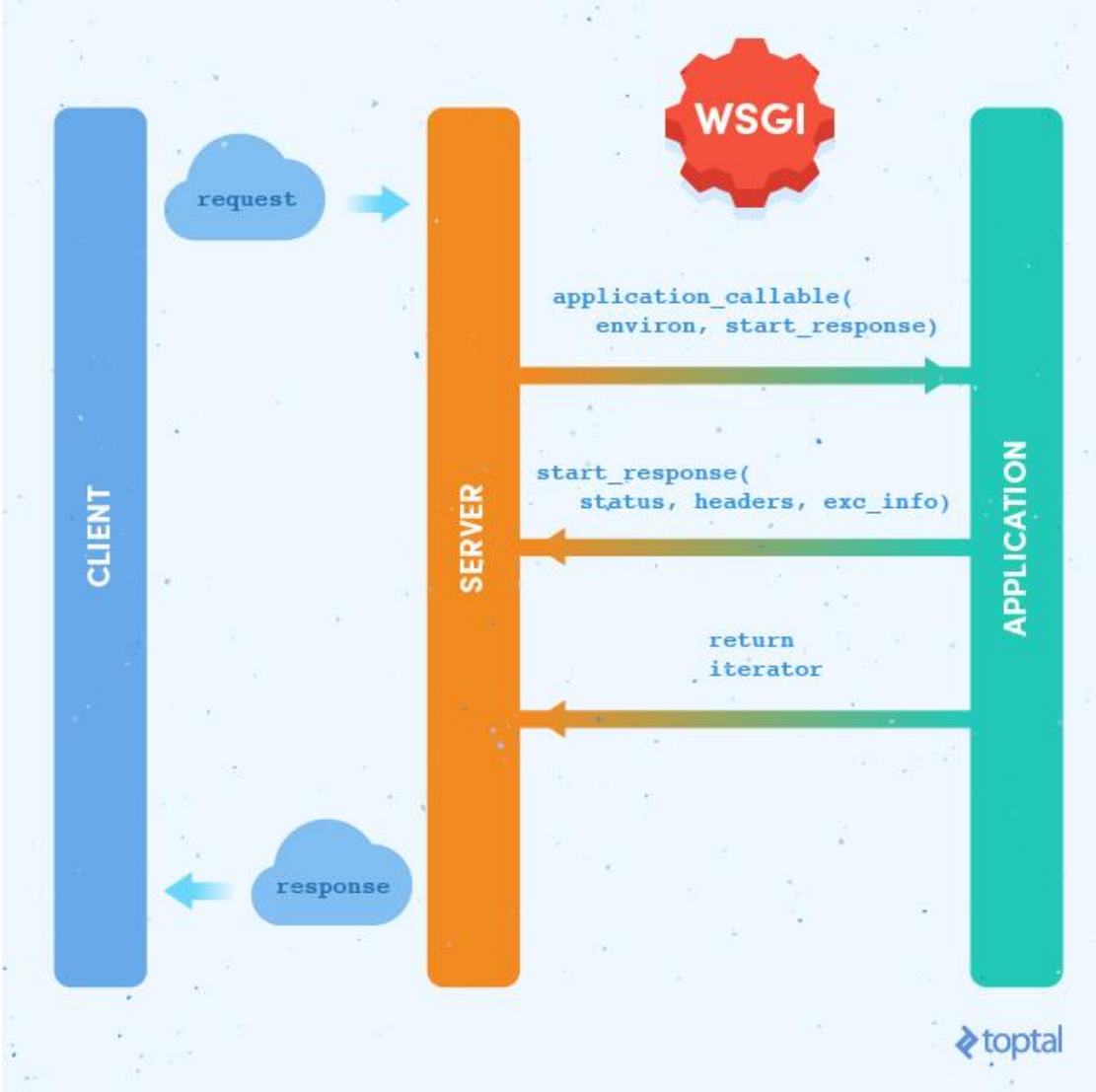
Επομένως, στην περίπτωση του Django, μία όψη είναι η Python συνάρτηση επιστροφής κλήσης (Python callback function) για ένα συγκεκριμένο URL επειδή αυτή η συνάρτηση επιστροφής κλήσης περιγράφει ποια δεδομένα θα παρουσιαστούν. Η όψη κανονικά αναθέτει (delegates) σε ένα πρότυπο την παρουσίαση των δεδομένων - το πώς, δηλαδή, παρουσιάζονται τα δεδομένα. Το πρότυπο μπορεί να είναι οποιοδήποτε πρότυπο/έγγραφο ιστού (web template/document), π.χ. σελίδα HTML, αρχείο CSV (Comma-Separated Values), κ.λπ. Ο ελεγκτής, τέλος, είναι το ίδιο το πλαίσιο: ο μηχανισμός που στέλνει ένα αίτημα στην κατάλληλη όψη σύμφωνα με τη διαμόρφωση των URL του Django (Django URL configuration).

2.3.1.2 Προδιαγραφή WSGI[29]

Η διεπαφή πύλης διακομιστή ιστού, WSGI (Web Server Gateway Interface), είναι μία απλή σύμβαση κλήσης (calling convention) για διακομιστές ιστού ώστε να διαβιβάζουν (forward) αιτήματα σε εφαρμογές ή πλαίσια ιστού που είναι γραμμένα σε Python. Το WSGI αρχικά ορίστηκε ως προδιαγραφή PEP (Python Enhancement Proposal) 333 το 2003, ενώ το PEP-3333, που δημοσιεύτηκε το 2010, ενημέρωνε (updates) την προηγούμενη προδιαγραφή (specification) σχετικά με την Python 3.

Το 2003, τα πλαίσια ιστού της Python γράφονταν συνήθως έτσι ώστε να συνδυάζονται με κάποια διεπαφή εκ των CGI (Common Gateway Interface), FastCGI και mod_python ή με κάποιο άλλο προσαρμοσμένο (custom) API ενός συγκεκριμένου διακομιστή ιστού. Σύμφωνα με το PEP-333:
Η Python διαθέτει αυτή τη στιγμή μία μεγάλη ποικιλία πλαισίων εφαρμογών ιστού, όπως το Zope, το Quixote, το Webware, το SkunkWeb, το PSO (Python Service Objects) και το Twisted Web. Αυτή η μεγάλη ποικιλία επιλογών μπορεί να αποτελέσει πρόβλημα για τους νέους χρήστες της Python, διότι σε γενικές γραμμές η επιλογή του πλαισίου ιστού περιορίζει την επιλογή των χρησιμοποιήσιμων (usable) διακομιστών ιστού και το αντίστροφο... Αντίθετα, παρόλο που η Java διαθέτει επίσης πολλά πλαίσια εφαρμογών ιστού, το "servlet" API της Java δίνει τη δυνατότητα σε εφαρμογές γραμμένες με οποιοδήποτε πλαίσιο εφαρμογών ιστού Java να τρέχουν σε οποιονδήποτε διακομιστή ιστού υποστηρίζει το servlet API.

Επομένως, το WSGI δημιουργήθηκε ως μία ανεξάρτητη υλοποίησης διεπαφή (implementation-agnostic interface) μεταξύ διακομιστών ιστού και εφαρμογών ή πλαισίων ιστού για την προώθηση κοινών πρακτικών κατά την ανάπτυξη φορητών (portable) εφαρμογών ιστού.



Σχήμα 2.9: Διάγραμμα αλληλεπιδράσεων μεταξύ των πλευρών της προδιαγραφής WSGI

Όπως φαίνεται και στο Σχήμα 2.9[30], το WSGI έχει δύο πλευρές:

- Την πλευρά διακομιστή/πύλης (server/gateway side): Πρόκειται συχνά για έναν πλήρη διακομιστή ιστού (full web server), όπως ο Apache ή ο Nginx, ή έναν ελαφρύ διακομιστή εφαρμογών (lightweight application server), όπως ο flup, που μπορεί να επικοινωνήσει με έναν διακομιστή ιστού.
- Την πλευρά εφαρμογής/πλαισίου (application/framework side): Πρόκειται για ένα κλητό αντικείμενο Python (Python callable) που παρέχεται από το πρόγραμμα ή το πλαίσιο της Python.

Μεταξύ του διακομιστή και της εφαρμογής μπορεί να υπάρχουν μία ή περισσότερες συνιστώσες μεσισμικού WSGI (WSGI middleware components), οι οποίες υλοποιούν και τις δύο πλευρές του API, συνήθως σε κώδικα Python. Το WSGI δεν προσδιορίζει ούτε το πώς πρέπει να εκκινείται ο διερμηνέας της Python (Python interpreter) ούτε το

πώς πρέπει να φορτώνεται ή να διαμορφώνεται (configured) το αντικείμενο της εφαρμογής (application object), αλλά αφήνει τα διάφορα πλαίσια (και διακομιστές) ιστού να υλοποιήσουν τις συγκεκριμένες ενέργειες με διαφορετικό συνήθως τρόπο το καθένα.

Μία συνιστώσα μεσισμικού WSGI είναι ένα κλητό αντικείμενο Python που αποτελεί το ίδιο μία εφαρμογή WSGI (WSGI application), αλλά μπορεί και να μεταβιβάζει (delegate) αιτήματα σε άλλες εφαρμογές WSGI. Αυτές οι εφαρμογές μπορούν οι ίδιες να είναι συνιστώσες μεσισμικού WSGI. Μία συνιστώσα μεσισμικού μπορεί να εκτελέσει διάφορες λειτουργίες όπως:

- Δρομολόγηση (routing) ενός αιτήματος σε διαφορετικά αντικείμενα εφαρμογών με βάση το URL προορισμού (target URL) μετά την ανάλογη αλλαγή των μεταβλητών περιβάλλοντος (environment variables)
- Εκτέλεση πολλαπλών εφαρμογών ή πλαισίων ταυτόχρονα στην ίδια διεργασία (process)
- Εξισορρόπηση φορτίου (load balancing) και απομακρυσμένη επεξεργασία (remote processing) με διαβίβαση (forwarding) αιτημάτων και αποκρίσεων μέσω ενός δικτύου
- Μετεπεξεργασία περιεχομένου (content post-processing) όπως χρησιμοποίηση φύλλων ύφους (style sheets) XSLT (eXtensible Stylesheet Language Transformations)

Πολλά πλαίσια ιστού υποστηρίζουν την προδιαγραφή WSGI, μερικά από τα οποία είναι:

- bjoern
- BlueBream
- Bottle
- CherryPy
- Django
- Flask
- Το webapp2 της μηχανής εφαρμογών της Google (Google App Engine)
- Gunicorn
- mod_wsgi για χρήση με τον Apache
- Η συνιστώσα WebOb του πακέτου Python Paste που είναι μία επέκταση WSGI (WSGI extension) και χρησιμοποιείται από το έργο Pylons (Pylons project)
- Pylons
- Pyramid
- Tornado
- Trac
- TurboGears
- uWSGI
- web2py

Αυτή τη στιγμή διατίθενται μονάδες περιτύλιξης (wrappers) για FastCGI, CGI, SCGI (Simple Common Gateway Interface), AJP (Apache JServ Protocol) με χρήση flup, Twisted, Apache με χρήση mod_wsgi ή mod_python, Nginx με χρήση ngx_http_uwsgi_module, και Microsoft IIS (Internet Information Services) με χρήση WFastCGI, isapi-wsgi, PyISAPIe (Internet Server Application Programming Interface) ή μίας πύλης ASP (Active Server Pages gateway).

2.3.2 Χαρακτηριστικά

2.3.2.1 Συνιστώσες

Παρόλο που έχει τη δική του ονοματολογία, π.χ. ονομάζει «όψεις» τα κλητά αντικείμενα (callable objects) που παράγουν τις αποκρίσεις HTTP, ο πυρήνας (core) του πλαισίου Django μπορεί να θεωρηθεί ως μία αρχιτεκτονική MVC:

- Σε επίπεδο μοντέλου (Model), αποτελείται από έναν αντικειμενοσχεσιακό απεικονιστή (Object-Relational Mapper - ORM) που μεσολαβεί μεταξύ μοντέλων δεδομένων (data models), τα οποία ορίζονται ως κλάσεις Python (Python classes), και μίας σχεσιακής βάσης δεδομένων (relational database).
- Σε επίπεδο όψης (View), αποτελείται από ένα σχήμα (schema) για την επεξεργασία αιτημάτων HTTP μέσω ενός συστήματος προτύπων ιστού (web template system).
- Σε επίπεδο ελεγκτή (Controller), αποτελείται από έναν διανεμητή URL (URL dispatcher) που βασίζεται σε κανονικές εκφράσεις (regular expressions).

Ο πυρήνας του πλαισίου περιλαμβάνει επίσης:

- Έναν ελαφρύ και μεμονωμένο διακομιστή ιστού (lightweight and standalone web server) για ανάπτυξη και δοκιμή (development and testing)
- Ένα σύστημα σειριοποίησης και επικύρωσης φορμών (form serialization and validation system) που μπορεί να μεταφράζει μεταξύ φορμών HTML (HTML forms) και τιμών κατάλληλων για αποθήκευση στη βάση δεδομένων (database storage)
- Ένα σύστημα προτύπων (template system) που αξιοποιεί την αντικειμενοστρεφή έννοια της κληρονομικότητας (object-oriented concept of inheritance)
- Ένα πλαίσιο προσωρινής αποθήκευσης (caching framework) που μπορεί να χρησιμοποιήσει οποιαδήποτε από τις διάφορες μεθόδους κρυφής μνήμης (cache methods)
- Υποστήριξη για κλάσεις μεσισμικού (middleware classes) που μπορούν να παρεμβαίνουν στα διάφορα στάδια επεξεργασίας αιτημάτων και να εκτελούν προσαρμοσμένες λειτουργίες (custom functions)
- Ένα εσωτερικό σύστημα διανεμητή (internal dispatcher system) που επιτρέπει στις συνιστώσες μίας εφαρμογής να επικοινωνούν τα συμβάντα (events) μεταξύ τους μέσω προκαθορισμένων σημάτων (predefined signals)
- Ένα σύστημα διεθνοποίησης (internationalization system) που περιλαμβάνει μεταφράσεις των συνιστωσών του ίδιου του Django σε μία ποικιλία γλωσσών
- Ένα σύστημα σειριοποίησης (serialization system) που μπορεί να παράγει και να διαβάζει XML (eXtensible Markup Language) ή/και JSON (JavaScript Object Notation) αναπαραστάσεις στιγμιοτύπων μοντέλων (model instance representations) του Django
- Ένα σύστημα για την επέκταση των δυνατοτήτων της μηχανής προτύπων (template engine)
- Μία διασύνδεση (interface) με το ενσωματωμένο (built-in) πλαίσιο δοκιμής μονάδων (unit test framework) της Python
- Μία διασύνδεση με την ενσωματωμένη μονάδα καταγραφής (logging module) της Python

2.3.2.2 Δεσμοποιημένες εφαρμογές

Η κύρια διανομή (main distribution) του Django περιλαμβάνει επίσης έναν αριθμό δεσμοποιημένων εφαρμογών (bundled applications) στο "contrib" πακέτο της όπως:

- Ένα επεκτάσιμο σύστημα αυθεντικοποίησης και εξουσιοδότησης (extensible authentication and authorization system)
- Τη δυναμική διαχειριστική διεπαφή (dynamic administrative interface)
- Εργαλεία για τη δημιουργία ροών διανομής (syndication feeds) RSS (RDF - Resource Description Framework - Site Summary / Rich Site Summary / Really Simple Syndication) και Atom
- Ένα πλαίσιο "Sites" ("Sites" framework) που επιτρέπει σε μία εγκατάσταση Django (Django installation) να τρέχει πολλαπλούς ιστοτόπους, τον καθένα με το δικό του περιεχόμενο και εφαρμογές
- Εργαλεία για τη δημιουργία χαρτών ιστοτόπων (sitemaps)
- Ενσωματωμένους μηχανισμούς μετρίασης (built-in mitigation mechanisms) για πλαστογράφηση αιτημάτων μεταξύ ιστοτόπων (Cross-Site Request Forgery - CSRF), εκτέλεση δέσμης ενεργειών (σεναρίων) μεταξύ ιστοτόπων (cross-Site Scripting - XSS), έγχυση SQL (SQL injection), σπάσιμο κωδικών πρόσβασης (password cracking) και άλλες τυπικές επιθέσεις ιστού (typical web attacks), οι περισσότεροι από τους οποίους είναι ενεργοποιημένοι από προεπιλογή (by default)
- Ένα πλαίσιο για τη δημιουργία εφαρμογών GIS (Geographic Information System)

2.3.2.3 Επεκτασιμότητα

Το σύστημα διαμόρφωσης (configuration system) του Django επιτρέπει την προσάρτηση (plugging-in) κώδικα τρίτων (third party code) σε ένα κανονικό έργο (regular project) υπό την προϋπόθεση ότι αυτός ακολουθεί τις συμβάσεις επαναχρησιμοποιήσιμης εφαρμογής (reusable app conventions). Περισσότερα από 2.500 πακέτα είναι διαθέσιμα για την επέκταση της αρχικής συμπεριφοράς του πλαισίου, παρέχοντας λύσεις σε ζητήματα που δεν αντιμετωπίζονται από το τελευταίο όπως η εγγραφή (registration), η αναζήτηση (search), η παροχή και κατανάλωση API (API provision and consumption), η διαχείριση περιεχομένου (content management), κ.λπ.

2.3.2.4 Παρατάξεις διακομιστών

Το Django μπορεί να εκτελεστεί σε συνδυασμό με τον Apache, τον Nginx (χρησιμοποιώντας WSGI), Gunicorn ή Cherokee (χρησιμοποιώντας τη μονάδα flup - flup module - της Python). Το Django μπορεί επίσης να εκκινήσει (launch) έναν διακομιστή ιστού που υποστηρίζει την προδιαγραφή FastCGI (FastCGI specification), όπως ο Lighttpd ή ο Hiawatha, και να τρέξει πίσω από αυτόν. Είναι επίσης δυνατόν να χρησιμοποιηθούν και άλλοι διακομιστές ιστού συμβατοί με την προδιαγραφή WSGI (WSGI-compliant web servers).

Το Django υποστηρίζει επίσημα τέσσερα οπίσθια τμήματα βάσεων δεδομένων (database backends): PostgreSQL, MySQL, SQLite και Oracle. Ο Microsoft SQL Server μπορεί να χρησιμοποιηθεί με το οπίσθιο τμήμα django-mssql στα λειτουργικά

συστήματα της Microsoft (Microsoft operating systems), ενώ παρομοίως υπάρχουν και εξωτερικά οπίσθια τμήματα (external backends) για τα συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMSs) Firebird, IBM (International Business Machines) Db2 και SAP (Systems, Applications and Products in data processing) SQL Anywhere. Υπάρχει ακόμα μία αποσχισμένη διακλάδωση (fork) του Django που ονομάζεται django-nonrel, η οποία υποστηρίζει βάσεις δεδομένων NoSQL (NoSQL databases) όπως η Datastore της μηχανής εφαρμογών της Google (Google App Engine) και η MongoDB.

3 Μοντέλα δεδομένων του SLUB και ιστότοπος διαχείρισης του Django

3.1 Διαρθρωτική διάταξη του SLUB στο πλαίσιο Django

3.1.1 Έργα και εφαρμογές Django[31]

Ο όρος «έργο» (project) περιγράφει μία εφαρμογή ιστού Django (Django web application). Το Python πακέτο έργου (project Python package) ορίζεται κυρίως από μία μονάδα ρυθμίσεων (settings module), αλλά συνήθως περιέχει και άλλα αρχεία. Για παράδειγμα, όταν τρέξουμε τη διαχειριστική εντολή (administrative command) startproject με όρισμα slub, λαμβάνουμε έναν κατάλογο έργου (project directory) slub που περιέχει ένα Python πακέτο slub με τα αρχεία settings.py, urls.py και wsgi.py. Το πακέτο έργου συχνά επεκτείνεται ώστε να περιλαμβάνει αρχεία όπως CSS, πρότυπα (templates) και fixtures - αρχεία σειριοποιημένων εγγραφών βάσης δεδομένων (serialized database records) σε μορφή JSON (JavaScript Object Notation), XML (eXtensible Markup Language) ή YAML (YAML Ain't Markup Language)[32] - που δεν συνδέονται με μία συγκεκριμένη εφαρμογή.

Ο ριζικός κατάλογος (root directory) ενός έργου, δηλαδή αυτός που περιέχει το αρχείο manage.py (βοηθητικό πρόγραμμα γραμμής εντολών - command-line utility - που επιτρέπει ποικιλότροπη αλληλεπίδραση με το έργο Django), είναι συνήθως ο περιέκτης (container) για όλες τις εφαρμογές του έργου που δεν έχουν εγκατασταθεί ξεχωριστά. Ο όρος «εφαρμογή» (application) περιγράφει ένα πακέτο Python που παρέχει κάποιο σύνολο χαρακτηριστικών (features). Οι εφαρμογές μπορούν να επαναχρησιμοποιηθούν σε διάφορα έργα.

Οι εφαρμογές περιλαμβάνουν έναν συνδυασμό μοντέλων, όψεων, προτύπων, ετικετών προτύπων (template tags), στατικών αρχείων, διευθύνσεων URL, μεσισμικού, κ.λπ. Συνδέονται γενικά σε έργα μέσω της ρύθμισης INSTALLED_APPS και προαιρετικά μέσω άλλων μηχανισμών όπως οι διαμορφώσεις URL (URLconfs), η ρύθμιση MIDDLEWARE ή η κληρονομικότητα προτύπων (template inheritance).

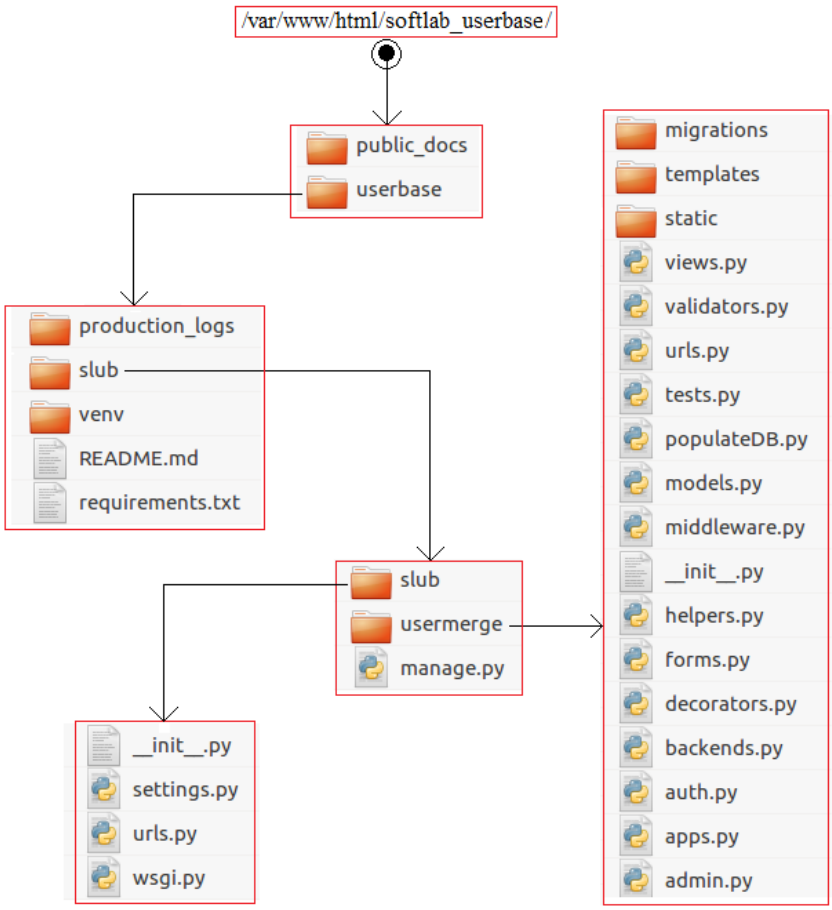
Μία εφαρμογή Django είναι απλά ένα σύνολο κώδικα που αλληλεπιδρά με διάφορα μέρη του πλαισίου και όχι κάποιο αντικείμενο Application (Application object) - δεν υπάρχει καν τέτοια κλάση Django (Django class). Ωστόσο, υπάρχουν μερικά σημεία όπου το Django πρέπει να αλληλεπιδρά με εγκατεστημένες εφαρμογές, κυρίως για διαμόρφωση (configuration) αλλά και για ενδοσκόπηση (introspection). Αυτός είναι ο λόγος για τον οποίο το μητρώο εφαρμογών (application registry), το οποίο ονομάζεται apps και διατίθεται στη μονάδα django.apps, διατηρεί μεταδεδομένα (metadata) σε ένα στιγμιότυπο AppConfig (AppConfig instance) του αρχείου apps.py για κάθε εγκατεστημένη εφαρμογή.

3.1.2 Έργο SLUB και εφαρμογή usermerge[33]

Η εφαρμογή ιστού SLUB (SoftLab UserBase) είναι ένα έργο Django, δηλαδή μία συλλογή ρυθμίσεων για ένα στιγμιότυπο Django (Django instance), η οποία συμπεριλαμβάνει τη διαμόρφωση των URL στο αρχείο urls.py, ένα σημείο εισόδου (entry point) για διακομιστές ιστού συμβατούς με την προδιαγραφή WSGI (WSGI-compatible web servers) στο αρχείο wsgi.py, καθώς και τη διαμόρφωση της βάσης δεδομένων (database configuration), τις επιλογές που αφορούν συγκεκριμένα το Django (Django-specific options) και τις ρυθμίσεις που αφορούν συγκεκριμένα τις εφαρμογές (application-specific settings) στο αρχείο settings.py.

Η βασική λειτουργικότητα (core functionality) του έργου, συμπεριλαμβανομένων των μοντέλων (βλ. models.py), των δοκιμών (βλ. tests.py), των URL (βλ. urls.py), των όψεων (βλ. views.py), των προτύπων HTML (βλ. κατάλογο templates), των στατικών αρχείων (βλ. κατάλογο static) και των προσαρμοσμένων ρυθμίσεων (custom settings) του συστήματος αυθεντικοποίησης (βλ. auth.py και backends.py) και του ιστοτόπου διαχείρισης του Django (βλ. admin.py), υλοποιείται στην Django εφαρμογή usermerge και η αντίστοιχη βάση δεδομένων ονομάζεται usermergeDB.

Ο πηγαίος κώδικας (source code) και η τεκμηρίωση (documentation) του SLUB διατίθενται στη διεύθυνση https://git.softlab.ntua.gr/giorgkazelidis/userbase, ενώ η διαρθρωτική διάταξή του (structural layout) στο πλαίσιο Django παρουσιάζεται συνοπτικά στο Σχήμα 3.1.



Σχήμα 3.1: Διαρθρωτική διάταξη του SLUB στο πλαίσιο Django

3.1.3 Εικονικό περιβάλλον του SLUB

Η ανάπτυξη (development) του SLUB και η επιστράτευσή του (deployment) στον ιστό πραγματοποιούνται εντός του εικονικού περιβάλλοντος (virtual environment) venv που ορίζεται στον ομώνυμο κατάλογο της εφαρμογής usermerge. Το venv, όπως και κάθε άλλο εικονικό περιβάλλον που δημιουργείται μέσω του εργαλείου virtualenv, είναι ένα απομονωμένο περιβάλλον Python (isolated Python environment) που έχει τους δικούς του καταλόγους εγκατάστασης (installation directories) και χώρο ονομάτων (namespace), δεν μοιράζεται τις βιβλιοθήκες του με άλλα εικονικά περιβάλλοντα και προαιρετικά δεν έχει πρόσβαση ούτε στις καθολικά εγκατεστημένες βιβλιοθήκες (globally installed libraries)[34]. Η χρήση εικονικών περιβαλλόντων είναι πιο πρακτική από την εγκατάσταση πακέτων σε επίπεδο συστήματος (systemwide package installation) καθώς παρέχει, μεταξύ άλλων, ευκολία αναβάθμισης πακέτων (package upgrading) μέσω του εργαλείου pip (Pip Installs Packages) και ενοποιημένο χειρισμό διαφόρων εκδόσεων Python (unified handling of different Python versions), ενώ επιτρέπει και την εγκατάσταση πακέτων χωρίς δικαιώματα διαχειριστή (administrator privileges)[35]. Στην περίπτωση του Django, το virtualenv μπορεί να παρέχει ένα αποκλειστικό περιβάλλον (dedicated environment), δηλαδή έναν ξεχωριστό κατάλογο εγκατεστημένων πακέτων, για καθένα από τα έργα μας έτσι ώστε να μην αλληλεπιδρούν μεταξύ τους[36]. Αν και δεν είναι υποχρεωτικό, αυτό θεωρείται μία βέλτιστη πρακτική που εξοικονομεί χρόνο και διευκολύνει την επιστράτευση ενός έργου στον ιστό.

Η λίστα όλων των πακέτων που είναι εγκατεστημένα (μέσω του pip) στο venv και των αντίστοιχων εκδόσεών τους περιέχεται στο αρχείο requirements.txt, το οποίο μας εξασφαλίζει, μεταξύ άλλων, την επαναληψιμότητα της εγκατάστασης (installation repeatability) των πακέτων και την προστασία από σφάλματα (bugs) και ασυμβατότητες (incompatibilities) νεότερων εκδόσεών τους (newly released versions)[37]. Διευκολύνει, δηλαδή, έναν οποιονδήποτε προγραμματιστή (ή και εμάς, αν χρειαστεί να ξαναδημιουργήσουμε το περιβάλλον) να εγκαταστήσει ανά πάσα στιγμή στο μηχάνημά του τα ίδια πακέτα χρησιμοποιώντας τις ίδιες εκδόσεις[38]. Τα πακέτα που αναφέρονται στο αρχείο requirements.txt είναι τα εξής:

- concurrent-log-handler (0.9.12)[39]: Παρέχει έναν επιπρόσθετο χειριστή αρχείων καταγραφής (log handler) για το πρότυπο πακέτο καταγραφής (standard logging package) της Python (Python Enhancement Proposal, PEP, 282) και υποστηρίζει τόσο συστήματα Windows όσο και POSIX (Portable Operating System Interface for UniX). Ο χειριστής ConcurrentRotatingFileHandler είναι μία άμεση - δίχως αλλαγές στον κώδικα ή τις ρυθμίσεις - αντικατάσταση (drop-in replacement) του πρότυπου χειριστή RotatingFileHandler της Python, δηλαδή παρέχει ένα σχήμα περιστροφής αρχείων (file rotation scheme) παρόμοιο με αυτό του RotatingFileHandler καθώς και ασφάλεια νήματος (thread safety) μέσω αντίστοιχων κλειδωμάτων (threading locks). Ειδικότερα, καταχωρεί συμβάντα (registers events) με τη μορφή εγγραφών (records) σε ένα αρχείο καταγραφής (log file), το οποίο περιστρέφεται (rotated) όταν φτάσει σε ένα συγκεκριμένο μέγεθος, και διατηρεί έναν συγκεκριμένο αριθμό αντιγράφων ασφαλείας (backups) για αυτό. Σε αντίθεση, όμως, με τον RotatingFileHandler, προσπαθεί να διατηρεί τις εγγραφές με κάθε κόστος επιτρέποντας στα αρχεία καταγραφής, όποτε χρειάζεται, να υπερβαίνουν το καθορισμένο μέγιστο μέγεθος περιστροφής (maximum rotation size). Χρησιμοποιεί επίσης κλείδωμα αρχείων (file locking) μέσω της βιβλιοθήκης

portalocker έτσι ώστε πολλαπλές διεργασίες (processes) να μπορούν να καταγράφουν ταυτόχρονα (concurrently) σε ένα αρχείο χωρίς απώλεια (dropping) ή αντικατάσταση/επικάλυψη (clobbering) εγγραφών.

Σύμφωνα με τις ρυθμίσεις καταγραφής στο αρχείο settings.py, ο χειριστής ConcurrentRotatingFileHandler χρησιμοποιείται στην παραγωγή (production), δηλαδή όταν το SLUB επιστρατεύεται στον ιστό και η ρύθμιση αποσφαλμάτωσης, DEBUG, είναι False, για την καταχώρηση όλων των εγγραφών, εξαιρουμένων των εγγραφών DEBUG, σε αρχεία καταγραφής που υπάρχουν στον κατάλογο production_logs. Οι γενικές εγγραφές INFO και WARNING καταχωρούνται στο αρχείο general_info_and_warnings.log, οι εγγραφές ERROR και CRITICAL καταχωρούνται στο αρχείο errors.log, οι εγγραφές INFO που καταγράφονται από τον καταγραφέα (logger) "usermerge.views.recover_user_profile" καταχωρούνται στο αρχείο changes_of_credentials_during_profile_recovery.log, ενώ οι εγγραφές DEBUG δεν καταχωρούνται πουθενά. Κάθε εγγραφή μορφοποιείται ούτως ώστε, εκτός από το κύριο μήνυμα (main message) που περιγράφει το αντίστοιχο συμβάν, να περιλαμβάνει και άλλες σχετικές λεπτομέρειες όπως η ημερομηνία-ώρα (date-time), το επίπεδο καταγραφής (logging level - DEBUG, INFO, WARNING, ERROR και CRITICAL), το όνομα αρχείου (file name), ο αριθμός γραμμής (line number), το όνομα συνάρτησης (function name), το όνομα καταγραφέα (logger name) και τα αναγνωριστικά διεργασίας-νήματος (process-thread IDs). Αξίζει να σημειωθεί ότι κατά την ανάπτυξη (development), δηλαδή όταν η ρύθμιση DEBUG είναι True, όλες οι εγγραφές, συμπεριλαμβανομένων των εγγραφών DEBUG, καταχωρούνται στην κονσόλα/τερματικό (console/terminal) - ροή (stream) sys.stderr του συστήματος - όπου τρέχει ο διακομιστής ανάπτυξης του Django (Django development server) μέσω της διαχειριστικής εντολής (administrative command) runserver.

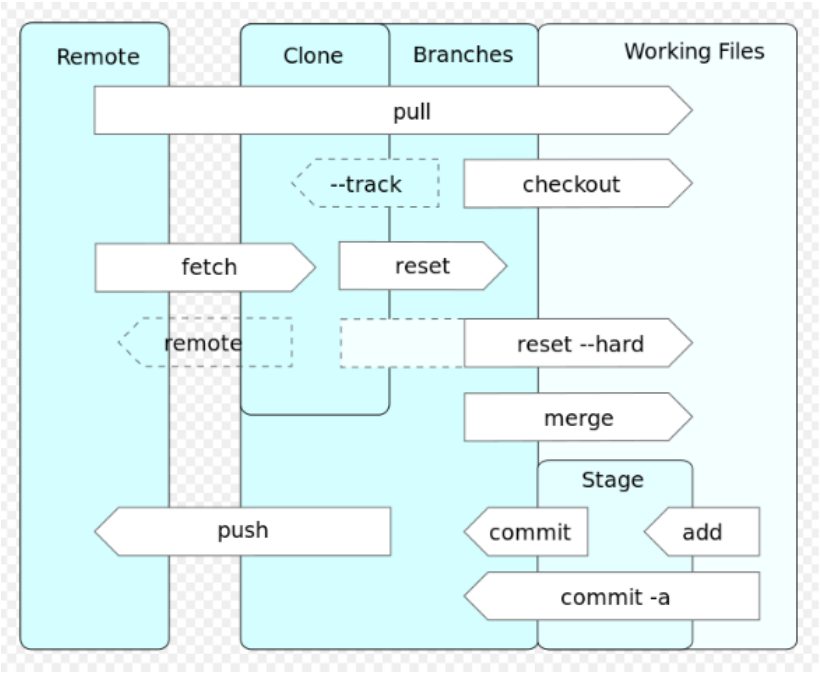
- Django (2.0.4)
- mod-wsgi (4.6.5)
- mysqlclient (1.3.12)[40]: Είναι ο εγγενής (native) και συνιστώμενος οδηγός (driver) της MySQL, ο οποίος υλοποιεί το API βάσεων δεδομένων της Python (Python Database API) που περιγράφεται στο PEP 249. Παρέχει ασφάλεια νήματος (thread safety) και κοινή χρήση συνδέσεων (connection pooling). Το Django διαθέτει έναν προσαρμογέα (adapter) για την πρόσβαση σε αυτόν τον οδηγό από το ORM (Object-Relational Mapper) του.
- pytz (2018.4)[41]: Παρέχει τη βάση δεδομένων Olson (Olson database) για τις ζώνες ώρας (time zones) στην Python. Αυτή η βιβλιοθήκη επιτρέπει ακριβείς και ανεξάρτητους πλατφόρμας (cross-platform) υπολογισμούς ζώνης ώρας χρησιμοποιώντας Python 2.4 ή νεότερη έκδοση. Επιλύει επίσης το ζήτημα των διφορούμενων ωρών στο τέλος της θερινής ώρας (Daylight Saving Time - DST), για την οποία διατίθενται περισσότερες πληροφορίες στην πρότυπη βιβλιοθήκη της Python (βλ. datetime.tzinfo). Υποστηρίζονται σχεδόν όλες οι ζώνες ώρας Olson.

Για πληρότητα, πρέπει να αναφερθούν και τα υπόλοιπα εργαλεία/συστήματα που είναι καθολικά εγκατεστημένα (globally installed) και χρησιμοποιούνται για την ανάπτυξη του SLUB και την επιστράτευσή του στον ιστό (βλ. τμήμα "ENVIRONMENT SPECIFICATIONS" στο αρχείο README.md):

- Ubuntu (16.04 LTS - Long Term Support)
- Python (3.5.2)

- Git (2.7.4)[42]: Το Git είναι ένα ελεύθερο (free), ανοικτού κώδικα (open-source), ανεξάρτητο πλατφόρμας (cross-platform) και ευρέως χρησιμοποιούμενο κατανεμημένο σύστημα ελέγχου έκδοσης (distributed version control system) για την παρακολούθηση αλλαγών (tracking changes) στον πηγαίο κώδικα (source code) κατά την ανάπτυξη λογισμικού. Έχει σχεδιαστεί για τον συντονισμό της εργασίας μεταξύ προγραμματιστών, αλλά μπορεί να χρησιμοποιηθεί και για την παρακολούθηση αλλαγών σε οποιοδήποτε σύνολο αρχείων. Οι στόχοι του περιλαμβάνουν την ταχύτητα (speed), την κλιμακωσιμότητα (scalability), την ακεραιότητα δεδομένων (data integrity), τη συμπίεση και το πακετάρισμα αντικειμένων (object compression and packing), τη συλλογή σκουπιδιών (garbage collection) και την υποστήριξη για κατανεμημένες, μη-γραμμικές ροές εργασίας (distributed, non-linear workflows). Όπως συμβαίνει με τα περισσότερα άλλα κατανεμημένα συστήματα ελέγχου έκδοσης και σε αντίθεση με τα περισσότερα συστήματα πελάτη-διακομιστή, κάθε κατάλογος Git (Git directory) σε έναν υπολογιστή είναι ένα τέλεια ανεπτυγμένο αποθετήριο (full-fledged repository) με πλήρες ιστορικό (complete history) και ολοκληρωμένες δυνατότητες παρακολούθησης έκδοσης (full version-tracking abilities), ανεξάρτητα από την πρόσβαση στο δίκτυο (network access) ή έναν κεντρικό διακομιστή (central server).

  Εφόσον το Git είναι ένα κατανεμημένο σύστημα ελέγχου έκδοσης, μπορεί να χρησιμοποιηθεί άμεσα (out of the box) ως διακομιστής. Διατίθεται με τον ενσωματωμένο δαίμονα (built-in daemon) git, ο οποίος εκκινεί έναν απλό διακομιστή TCP που εκτελείται με βάση το πρωτόκολλο Git και συνήθως ακούει στη θύρα (port) 9418. Ένας αποκλειστικός διακομιστής HTTP Git (dedicated HTTP Git server) παρέχει, μεταξύ άλλων, διάφορα χαρακτηριστικά (features) όπως η προσθήκη ελέγχου πρόσβασης (access control), η εμφάνιση των περιεχομένων ενός αποθετηρίου Git μέσω των διεπαφών ιστού και η διαχείριση πολλαπλών αποθετηρίων. Ένα ήδη υπάρχον αποθετήριο Git μπορεί να κλωνοποιηθεί (cloned) και να μοιραστεί για να χρησιμοποιηθεί από άλλους ως κεντρικό αποθετήριο (centralized repo). Μπορεί επίσης να προσπελαστεί μέσω απομακρυσμένου φλοιού (remote shell), αρκεί πρώτα να έχει εγκατασταθεί το λογισμικό Git και να έχει επιτραπεί στον χρήστη να εισέρχεται (log in). Στο Σχήμα 3.2 γίνεται μία επιγραμματική παρουσίαση των διάφορων ροών δεδομένων (data flows) και επιπέδων αποθήκευσης (storage levels) του συστήματος Git.
- pip (10.0.1)
- virtualenv (15.2.0)
- MySQL (Community) Server (5.7.24)
- Apache HTTP Server (2.4.18)



Σχήμα 3.2: Επισκόπηση των ροών δεδομένων και των επιπέδων αποθήκευσης του συστήματος Git

3.2 Κύρια μοντέλα δεδομένων του SLUB

3.2.1 Επισκόπηση

Τα κύρια μοντέλα δεδομένων (main data models) της εφαρμογής usermerge, και κατ' επέκταση του έργου SLUB, είναι:

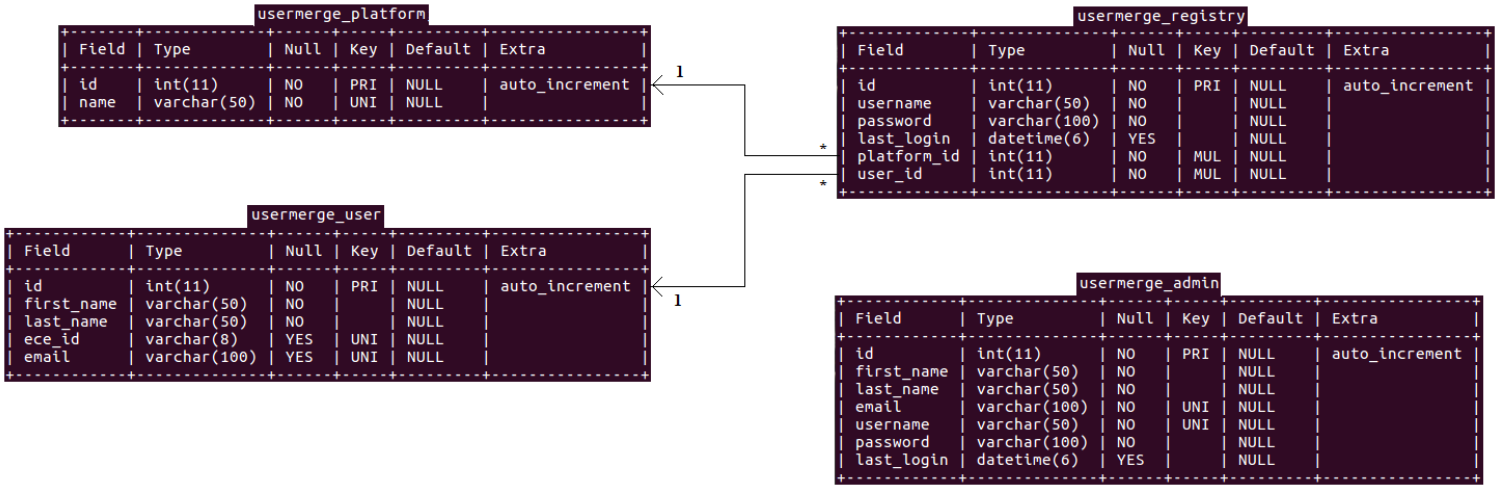
- Platform: Αντιπροσωπεύει οποιαδήποτε πλατφόρμα διατίθεται μέσω του Εργαστηρίου Τεχνολογίας Λογισμικού, SoftLab, και στην οποία μπορούν να εισέλθουν φοιτητές της Σχολής ΗΜΜΥ με κατάλληλα διαπιστευτήρια. Τέτοιες πλατφόρμες είναι οι Novice, Grader, Moodle και PLgrader.
- User: Αντιπροσωπεύει τον απλό χρήστη, δηλαδή φοιτητή της Σχολής ΗΜΜΥ, που εισέρχεται στο SLUB με σκοπό να συμπληρώσει/τροποποιήσει τα στοιχεία του προφίλ του (profile elements) για μία συγκεκριμένη πλατφόρμα, να συνδέσει το συμπληρωμένο προφίλ του με μία άλλη πλατφόρμα ή να αλλάξει τα διαπιστευτήριά του για μία τρίτη πλατφόρμα.
- Registry: Αντιπροσωπεύει τη συσχέτιση ενός απλού χρήστη με μία συγκεκριμένη πλατφόρμα (εγγραφή/καταχώρηση χρήστη στην πλατφόρμα) διατηρώντας τα διαπιστευτήρια εισόδου (login credentials), δηλαδή το όνομα χρήστη (username) και τον κωδικό πρόσβασης (password), του πρώτου για την τελευταία.
- Admin: Αντιπροσωπεύει τον διαχειριστή (συμπεριλαμβανομένων των διαπιστευτηρίων του για το SLUB), π.χ. υπεύθυνο του SoftLab, που εισέρχεται στο SLUB με σκοπό να εισαγάγει νέα διαπιστευτήρια ή να δημιουργήσει βάσει κριτηρίων αναφορές (criteria-based reports) απλών χρηστών για μία συγκεκριμένη πλατφόρμα.

Τα παραπάνω μοντέλα απεικονίζονται μέσω του Django ORM στους ομώνυμους πίνακες (tables) της βάσης usermergeDB (βλ. models.py), από όπου και χρησιμοποιούνται όχι μόνο για τις λειτουργίες/όψεις της εφαρμογής μας αλλά και για

την αλληλεπίδρασή της με τον ιστότοπο διαχείρισης Django (Django admin site). Οι προαναφερθέντες πίνακες δημιουργούνται μέσω της διαχειριστικής εντολής μετεγκατάστασης, migrate, και περιγράφονται στο Σχήμα 3.3.

Αναφορικά με τον πίνακα Registry, αξίζει να σημειωθούν τα εξής:

- Κάθε αντικείμενο Registry υποστηρίζει την αναφορική δράση (referential action) προστασίας/περιορισμού (PROTECT/RESTRICT) σε περίπτωση διαγραφής (ON DELETE) για τα αντικείμενα User και Platform στα οποία αναφέρεται (references) μέσω των εξωτερικών κλειδιών του (foreign keys). Αυτό σημαίνει ότι ένα αντικείμενο User ή Platform απαγορεύεται να διαγραφεί από τη βάση αν υπάρχει τουλάχιστον ένα αντικείμενο Registry που αναφέρεται σε αυτό. Με άλλα λόγια, μπορεί να διαγραφεί μόνο αν πρώτα διαγραφούν χειροκίνητα (manually) όλα τα αντικείμενα Registry που αναφέρονται σε αυτό[43]. Με αυτόν τον τρόπο διασφαλίζεται ότι τα εξωτερικά κλειδιά του πίνακα Registry δεν σπάνε σε περίπτωση διαγραφής αντικειμένων User ή/και Platform.
- Τα ζεύγη πεδίων (field pairs) {user_id, platform_id} και {username, platform_id} αποτελούν σύνθετα κλειδιά (composite keys) του πίνακα Registry, δηλαδή κάθε αντικείμενο Registry έχει μοναδικούς συνδυασμούς τιμών (unique value combinations) για τα προαναφερθέντα ζεύγη πεδίων. Συνεπώς, στα πλαίσια της εφαρμογής μας, ένας απλός χρήστης μπορεί να έχει το πολύ έναν λογαριασμό (εγγραφή) σε μία συγκεκριμένη πλατφόρμα, ο οποίος προσδιορίζεται μονοσήμαντα (uniquely) σε αυτή μέσω του αντίστοιχου ονόματος χρήστη (username).



Σχήμα 3.3: Επισκόπηση των πινάκων της βάσης usermergeDB που αντιστοιχούν στα κύρια μοντέλα δεδομένων του SLUB

3.2.2 Βασικές παρατηρήσεις

Στο αρχείο models.py έχουν καταγραφεί κάποιες βασικές παρατηρήσεις που αφορούν τη βάση usermergeDB και τα κύρια μοντέλα δεδομένων του SLUB και συνοδεύονται από τις σχετικές βιβλιογραφικές παραπομπές (bibliographic citations). Αναλυτικότερα:

(A) Το RDBMS που χρησιμοποιείται ως οπίσθιο τμήμα της βάσης usermergeDB είναι η MySQL. Το σύνολο χαρακτήρων (character set), δηλαδή η κωδικοποίηση (encoding),

της βάσης τίθεται σε utf8mb4 (8-bit Unicode Transformation Format, Max Bytes - 4) και η ταξινόμησή της (collation) τίθεται σε utf8mb4_0900_ai_ci (Accent Insensitive - Case Insensitive), utf8mb4_unicode_520_ci ή utf8mb4_unicode_ci όταν χρησιμοποιείται η έκδοση 8.0, 5.6/5.7 ή 5.5 της MySQL αντίστοιχα. Με αυτόν τον τρόπο επιτυγχάνεται καλύτερη αναπαράσταση και αντιπαραβολή δεδομένων Unicode στη βάση. Επιπροσθέτως, διασφαλίζεται ότι η μηχανή αποθήκευσης (storage engine) των πινάκων της είναι η InnoDB και ότι η SQL λειτουργία της (SQL mode) περιλαμβάνει την επιλογή (option) STRICT_TRANS_TABLES για την κατάλληλη υποστήριξη συναλλαγών (transaction support).

(Β) Η κενή τιμή (empty value) για τα πεδία ημερομηνίας-ώρας (DateTimeFields) είναι το NULL, το οποίο ερμηνεύεται ως None στην Python. Η κενή τιμή για τα πεδία χαρακτήρων (CharFields) και ηλεκτρονικού ταχυδρομείου (EmailFields) είναι η κενή / μηδενικού μήκους συμβολοσειρά (empty/zero-length string), "". Για μοναδικά (unique) CharFields και EmailFields, όπως το ece_id (Αριθμός Μητρώου, ΑΜ, στη Σχολή ΗΜΜΥ) και το email του μοντέλου User, η τιμή των οποίων ενδέχεται να μην δηλωθεί (declared) κατά τη δημιουργία/ενημέρωση των αντίστοιχων εγγραφών της βάσης, αυτό θα μπορούσε να οδηγήσει σε σφάλματα ακεραιότητας (integrity errors) λόγω διπλοτυπίας κλειδιού (key duplication). Για να αποφευχθούν τέτοια σφάλματα, το NULL θα πρέπει να χρησιμοποιείται ως κενή τιμή αυτών των πεδίων αντί της κενής συμβολοσειράς.

(Γ) Το πεδίο password ενός αντικειμένου User/Admin δεν πρέπει ποτέ να περιέχει τον ανεπεξέργαστο κωδικό πρόσβασης (raw password) του αντίστοιχου χρήστη. Θα πρέπει, αντίθετα, να περιέχει μία κατακερματισμένη εκδοχή (hashed version) του κωδικού που δημιουργείται από τη συνάρτηση django.contrib.auth.hashers.make_password() για βελτιωμένη ασφάλεια (enhanced security - βλ. populateDB.py για παραδείγματα). Από προεπιλογή, η προαναφερθείσα συνάρτηση δημιουργεί τον κατακερματισμένο κωδικό χρησιμοποιώντας τον αλγόριθμο PBKDF2 (Password-Based Key Derivation Function 2) με τιμή κατακερματισμού SHA-256 (256-bit Secure Hash Algorithm), ο οποίος είναι ένας μηχανισμός επέκτασης κωδικών πρόσβασης (password stretching mechanism) που συνιστάται από το NIST (National Institute of Standards and Technology). Ο μηχανισμός αυτός αρκεί για τις περισσότερες περιπτώσεις χρήσης (use cases) καθώς είναι αρκετά ασφαλής, απαιτώντας τεράστιες ποσότητες υπολογιστικού χρόνου (computing time) για να σπάσει. Ο κατακερματισμένος κωδικός πρόσβασης είναι μία συμβολοσειρά της μορφής *<hashing_algorithm>$<algorithm_iterations>$<random_salt>$<password_hash>* και έχει μήκος περίπου 80 χαρακτήρων, π.χ. "pbkdf2_sha256$15000$Pjun1TMGEQnM$lShdzU33covbDNiqGVDffdHh/86VaECJlaaNXchT0ew="[44].

(Δ) Τα πεδία id και last_login συμπληρώνονται/ενημερώνονται αυτόματα: κάθε μοντέλο λαμβάνει υπόρρητα (implicitly) ένα αυτόματο πεδίο (AutoField) id που δρα ως το αντίστοιχο πρωτεύον κλειδί (primary key), ενώ το πεδίο last_login ενός αντικειμένου User/Admin είναι αρχικά κενό και ενημερώνεται κάθε φορά που εκπέμπεται (emitted) το αντίστοιχο σήμα django.contrib.auth.signals.user_logged_in. Συνεπώς, τα πεδία αυτά δεν πρέπει να δηλώνονται κατά τη δημιουργία/ενημέρωση αντικειμένων των μοντέλων. Τα υπόλοιπα πεδία πρέπει να ικανοποιούν συγκεκριμένες απαιτήσεις (βλ. populateDB.py για παραδείγματα):

- Το πεδίο name ενός αντικειμένου Platform δεν πρέπει να είναι κενό. Η τιμή του μπορεί να αλλαχθεί έγκυρα (validly) και με συνέπεια (consistently), αλλά δεν πρέπει ποτέ να γίνει κενή. Ένα αντικείμενο Platform ενδείκνυται να δημιουργείται/αλλάζεται χειροκίνητα (manually) στη βάση, π.χ. μέσω του διαδραστικού φλοιού (interactive shell) ή ενός fixture του Django, και όχι στα πλαίσια κάποιας συνεδρίας (session context).
- Όλα τα πεδία ενός αντικειμένου Registry πρέπει να είναι μη-κενά (non-empty). Οι τιμές των πεδίων user_id, username και password μπορούν να αλλαχθούν έγκυρα και με συνέπεια (η τιμή του πεδίου platform_id δεν πρέπει ποτέ να αλλαχθεί), αλλά δεν πρέπει ποτέ να γίνουν κενές. Κάθε φορά που δημιουργείται ένα αντικείμενο Registry, θα πρέπει να δημιουργείται και ένα κενό αντικείμενο User που θα συσχετίζεται με αυτό (βλ. όψη import_user_credentials()).
- Όλα τα πεδία ενός αντικειμένου Admin πρέπει να είναι μη-κενά. Οι τιμές τους μπορούν να αλλαχθούν έγκυρα και με συνέπεια, αλλά δεν πρέπει ποτέ να γίνουν κενές. Ένα αντικείμενο Admin ενδείκνυται να δημιουργείται/αλλάζεται χειροκίνητα στη βάση, π.χ. μέσω του διαδραστικού φλοιού ή ενός fixture του Django, και όχι στα πλαίσια κάποιας συνεδρίας.
- Τα πεδία ενός αντικειμένου User πρέπει πάντα να βρίσκονται σε μία από τις ακόλουθες καταστάσεις (states): (1) όλα συμπληρωμένα (filled out), (2) όλα συμπληρωμένα εκτός από το ece_id ή (3) κανένα συμπληρωμένο. Οι επιτρεπόμενες μεταβάσεις (allowed transitions) μεταξύ των καταστάσεων είναι οι εξής: 1 → 1, 2 → 1, 2 → 2, 3 →1 και 3 → 2 (οι κενές τιμές των πεδίων μπορούν να αλλαχθούν σε έγκυρες και συνεπείς μη-κενές τιμές, ενώ οι μη-κενές τιμές των πεδίων μπορούν να αλλαχθούν έγκυρα και με συνέπεια, αλλά δεν πρέπει ποτέ να γίνουν κενές). Κάθε φορά που δημιουργείται ένα αντικείμενο User, όλα τα πεδία του θα πρέπει να είναι κενά (κατάσταση 3) και θα πρέπει να δημιουργείται ακριβώς ένα αντικείμενο Registry που θα αναφέρεται σε αυτό (βλ. όψη import_user_credentials()). Τα πεδία μπορούν στη συνέχεια να συμπληρωθούν/αλλαχθούν (κατάσταση 1/2) μόνο από τον αντίστοιχο χρήστη μέσω της φόρμας επεξεργασίας προφίλ χρήστη (user profile edit form), αφού πρώτα αυτός εισέλθει στο SLUB (βλ. όψη edit_user_profile()) - οποιοσδήποτε άλλος τρόπος συμπλήρωσης/αλλαγής των πεδίων δεν είναι σε καμία περίπτωση αποδεκτός.

(Ε) Ένα αντικείμενο User διατηρεί μία τιμή last_login για κάθε αντικείμενο Platform που σχετίζεται με αυτό μέσω του αντίστοιχου αντικειμένου Registry, ενώ ένα αντικείμενο Admin σχετίζεται μόνο με την «πλατφόρμα» SLUB, δηλαδή δεν σχετίζεται με κανένα αντικείμενο Platform, και συνεπώς διατηρεί ακριβώς μία τιμή last_login. Κάθε φορά που κάποιος απλός χρήστης ή διαχειριστής εισέρχεται στο SLUB, εκπέμπεται ένα κατάλληλο σήμα user_logged_in (βλ. συνάρτηση login() του αρχείου auth.py) που προκαλεί την ενημέρωση του πεδίου last_login του αντίστοιχου αντικειμένου User/Admin για την πλατφόρμα εισόδου (login platform) - «πλατφόρμα» SLUB στην περίπτωση αντικειμένου Admin - ανεξάρτητα από τα άλλα πεδία του. Ειδικότερα:

- Ένα μη-κενό αντικείμενο User πρέπει πάντα να διατηρεί μία μη-κενή τιμή last_login για καθένα από τα αντικείμενα Platform που σχετίζονται με αυτό (βλ. όψεις edit_user_profile() και recover_user_profile()). Αντίθετα, ένα κενό αντικείμενο User διατηρεί γενικά μία κενή τιμή last_login για το μοναδικό αντικείμενο Platform που σχετίζεται με αυτό (βλ. όψη import_user_credentials()).

Εξαίρεση αποτελεί η περίπτωση όπου ο απλός χρήστης εισέρχεται στο SLUB και, παρόλο που το προφίλ του είναι κενό, αυτός επιλέγει να εξέλθει (ή αποσυνδέεται - logged out - αυτόματα λόγω λήξης της συνεδρίας - session expiration) χωρίς να το συμπληρώσει· η τιμή last_login είναι τότε μη-κενή.

- Ένα αντικείμενο Admin διατηρεί γενικά μία μη-κενή τιμή last_login για την «πλατφόρμα» SLUB καθώς ο αντίστοιχος διαχειριστής θα πρέπει να εισέλθει σε αυτή για να αποκτήσει πρόσβαση σε οποιαδήποτε λειτουργία/όψη απευθύνεται σε διαχειριστές (Admin-targeted function/view). Εξαίρεση αποτελεί η περίπτωση όπου ο διαχειριστής δεν έχει εισέλθει ποτέ, αλλά το προφίλ του έχει ήδη συμπληρωθεί χειροκίνητα (όχι στα πλαίσια κάποιας συνεδρίας) στη βάση, π.χ. μέσω του διαδραστικού φλοιού ή ενός fixture του Django· η τιμή last_login είναι τότε κενή.

(Ζ) Δεδομένου ότι η υποστήριξη των ζωνών ώρας (time zone support) είναι ενεργοποιημένη (η ρύθμιση USE_TZ στο αρχείο settings.py είναι True), οι τιμές των DateTimeFields (αντικείμενα datetime της Python) αποθηκεύονται στη βάση στη μορφή αναφοράς (reference format) UTC (Coordinated Universal Time - Συντονισμένη Παγκόσμια Ώρα), η οποία δεν επηρεάζεται από τις αλλαγές DST (Daylight Saving Time - Ώρα Οικονομίας Ηλιακού Φωτός), και εμφανίζονται στα πρότυπα HTML στην τρέχουσα ζώνη ώρας, η οποία είναι "Europe/Athens" από προεπιλογή (καθορίζεται από τη ρύθμιση TIME_ZONE στο αρχείο settings.py).

3.3 Αυθεντικοποίηση χρηστών στο Django[45]

3.3.1 Επισκόπηση[46]

Το Django συνοδεύεται από ένα σύστημα αυθεντικοποίησης (authentication system), το οποίο διαχειρίζεται τους λογαριασμούς (accounts), τις ομάδες (groups) και τα δικαιώματα (permissions) χρηστών καθώς και τις συνεδρίες χρηστών που βασίζονται σε cookies (cookie-based user sessions). Χειρίζεται μάλιστα ταυτόχρονα τις «συζευγμένες» λειτουργίες ("coupled" functions) αυθεντικοποίησης και εξουσιοδότησης (authentication and authorization). Εν συντομία, η αυθεντικοποίηση επαληθεύει ότι ένας χρήστης είναι όντως εκείνος που ισχυρίζεται ότι είναι, ενώ η εξουσιοδότηση καθορίζει τι επιτρέπεται να κάνει ένας αυθεντικοποιημένος χρήστης (authenticated user).

Το σύστημα αυθεντικοποίησης αποτελείται από:

- Χρήστες: Πρόκειται για τους χρήστες του πλαισίου Django που αντιστοιχούν σε αντικείμενα του μοντέλου auth.User. Διαφέρουν μάλιστα από τους χρήστες της εφαρμογής usermerge, και κατ' επέκταση του έργου SLUB, που αντιστοιχούν σε αντικείμενα των μοντέλων usermerge.User (απλός χρήστης του SLUB) και usermerge.Admin (διαχειριστής του SLUB). Κατά κανόνα, τα δύο τελευταία μοντέλα αναφέρονται απλά ως User και Admin λόγω της συχνής χρήσης τους τόσο στον κώδικα όσο και στην τεκμηρίωση (documentation) του SLUB.
- Δικαιώματα: Δυαδικές σημαίες (binary flags), «ναι» ή «όχι», που καθορίζουν αν ένας χρήστης μπορεί να εκτελέσει μία συγκεκριμένη εργασία. Αντιστοιχούν σε αντικείμενα του μοντέλου auth.Permission.

- Ομάδες: Ένας γενικός τρόπος ανάθεσης ετικετών (labels) και δικαιωμάτων σε περισσότερους από έναν χρήστες. Αντιστοιχούν σε αντικείμενα του μοντέλου auth.Group.
- Ένα ρυθμιζόμενο (configurable) σύστημα κατακερματισμού κωδικών πρόσβασης (password hashing system)
- Φόρμες (forms) και εργαλεία όψεων (view tools) για την είσοδο χρηστών (logging in users) ή τον περιορισμό περιεχομένου (restricting content)
- Ένα σύστημα προσαρτήσιμων οπίσθιων τμημάτων (pluggable backend system)

3.3.2 Αντικείμενα auth.User

Τα αντικείμενα auth.User αποτελούν τον πυρήνα (core) του συστήματος αυθεντικοποίησης. Συνήθως αντιπροσωπεύουν τα άτομα που αλληλεπιδρούν με τον ιστότοπο της εφαρμογής μας και χρησιμοποιούνται για την ενεργοποίηση λειτουργιών όπως ο περιορισμός πρόσβασης (restricting access), η καταχώρηση προφίλ χρηστών (registering user profiles), η συσχέτιση περιεχομένου με τους δημιουργούς (associating content with creators), κ.λπ. Υπάρχει μόνο μία κλάση χρηστών (user class) στο πλαίσιο αυθεντικοποίησης του Django, δηλαδή οι «υπερχρήστες» ("superusers") και τα μέλη «προσωπικού» διαχείρισης (admin "staff" members) είναι απλώς αντικείμενα auth.User με καθορισμένα ειδικά χαρακτηριστικά (special attributes), όχι διαφορετικές κλάσεις αντικειμένων χρηστών.

Τα κύρια χαρακτηριστικά της προεπιλεγμένης (default) κλάσης χρηστών, auth.User, είναι το όνομα εισόδου (username), ο κωδικός πρόσβασης (password), το ηλεκτρονικό ταχυδρομείο (email), το μικρό όνομα (first_name) και το επώνυμο (last_name). Τα δύο πρώτα μάλιστα είναι απαιτούμενα πεδία (required fields) των αντικειμένων auth.User.

Αξίζει να σημειωθούν τα εξής[47]:
- Το πεδίο password διατηρεί μία τιμή κατακερματισμού (hash value), συμπεριλαμβανομένων μεταδεδομένων (metadata), για τον κωδικό πρόσβασης (το Django δεν αποθηκεύει τον ανεπεξέργαστο - raw - κωδικό πρόσβασης). Οι ανεπεξέργαστοι κωδικοί πρόσβασης μπορούν να είναι αυθαίρετα μεγάλοι και να περιέχουν οποιονδήποτε χαρακτήρα.
- Το πεδίο is_staff διατηρεί μία λογική τιμή (Boolean/logical value) / τιμή αληθείας (truth value), «Αληθές» ("True") ή «Ψευδές» ("False"), που καθορίζει αν ο χρήστης μπορεί να έχει πρόσβαση στον ιστότοπο διαχείρισης Django.
- Το πεδίο is_active διατηρεί μία τιμή αληθείας που καθορίζει αν ο λογαριασμός χρήστη (user account) πρέπει να θεωρείται ενεργός. Συνιστάται να τίθεται αυτή η σημαία (flag) σε False αντί να γίνεται διαγραφή λογαριασμών· με αυτόν τον τρόπο, αν οποιαδήποτε εφαρμογή μας διατηρεί εξωτερικά κλειδιά (foreign keys) σε χρήστες, τα κλειδιά αυτά δεν θα σπάσουν.
- Το πεδίο is_superuser διατηρεί μία τιμή αληθείας που δηλώνει ότι ο χρήστης έχει όλα τα δικαιώματα χωρίς να χρειάζεται να του ανατεθούν ρητά.

3.3.3 Αντικείμενα AnonymousUser[48]

Ανώνυμος χρήστης είναι ο χρήστης που δεν έχει αυθεντικοποιηθεί, δηλαδή δεν έχει παράσχει έγκυρες λεπτομέρειες αυθεντικοποίησης. Ωστόσο, αυτό δεν σημαίνει απαραίτητα ότι δεν εξουσιοδοτείται να κάνει τίποτα. Ειδικότερα, οι περισσότεροι ιστότοποι εξουσιοδοτούν τους ανώνυμους χρήστες να περιηγούνται στο μεγαλύτερο μέρος του ιστοτόπου, ενώ πολλοί επιτρέπουν την ανώνυμη δημοσίευση σχολίων (anonymous posting of comments), κ.λπ.

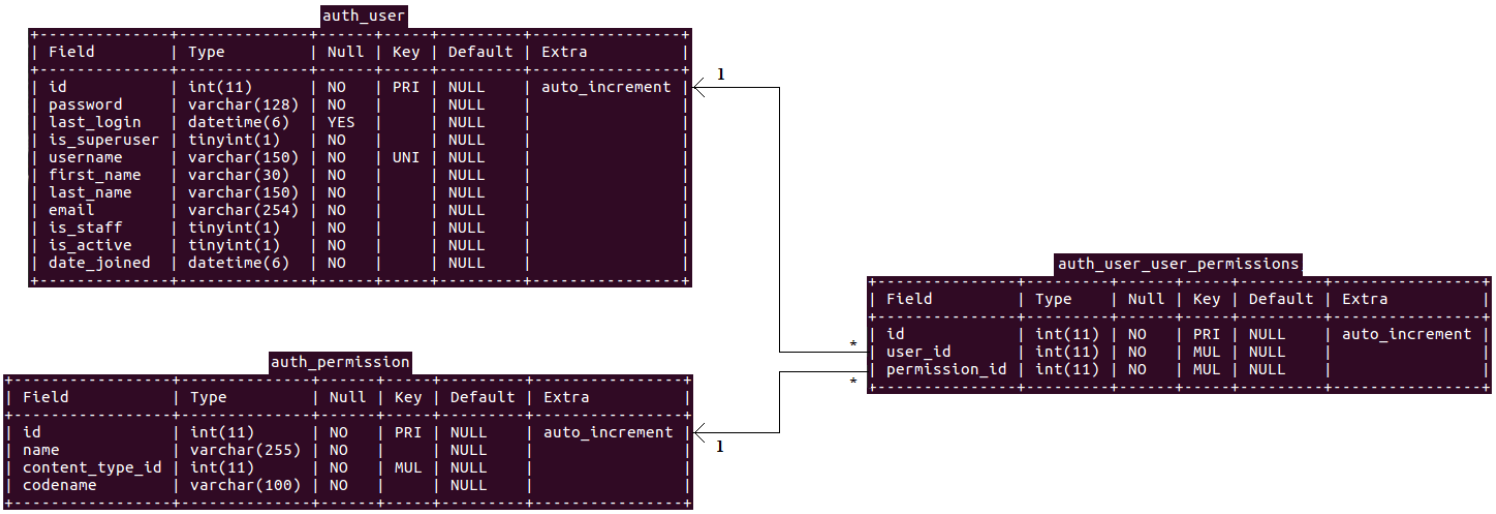
Το πλαίσιο δικαιωμάτων του Django δεν αποθηκεύει δικαιώματα για ανώνυμους χρήστες. Εντούτοις, το αντικείμενο χρήστη που διαβιβάζεται (passed) σε ένα οπίσθιο τμήμα αυθεντικοποίησης μπορεί να είναι ένα αντικείμενο AnonymousUser, επιτρέποντας στο οπίσθιο τμήμα να καθορίσει μία προσαρμοσμένη συμπεριφορά εξουσιοδότησης (custom authorization behavior) για ανώνυμους χρήστες. Αυτό είναι ιδιαίτερα χρήσιμο για τους συγγραφείς επαναχρησιμοποιήσιμων εφαρμογών, οι οποίοι μπορούν να μεταβιβάζουν (delegate) όλα τα ζητήματα εξουσιοδότησης στο οπίσθιο τμήμα αυθεντικοποίησης χωρίς να χρειάζονται ρυθμίσεις, π.χ. για τον έλεγχο ανώνυμης πρόσβασης (anonymous access control).

Η κλάση (auth.)AnonymousUser (django.contrib.auth.models.AnonymousUser) υλοποιεί τη διεπαφή auth.User (django.contrib.auth.models.User) με τις εξής διαφορές[47]:

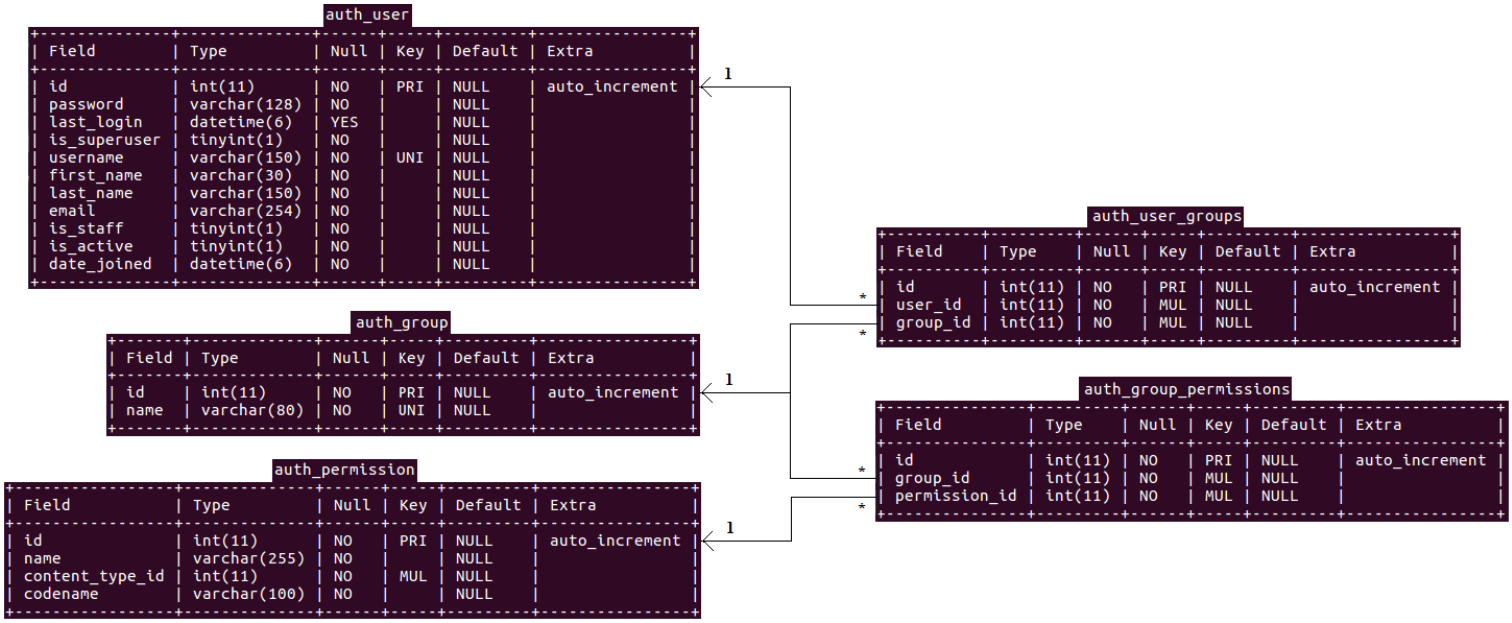
- Η τιμή του πεδίου id είναι πάντα None.
- Η τιμή του πεδίου username είναι πάντα η κενή συμβολοσειρά.
- Η μέθοδος get_username() επιστρέφει πάντα την κενή συμβολοσειρά.
- Η τιμή του μόνο-για-ανάγνωση χαρακτηριστικού (read-only attribute) is_anonymous είναι πάντα True.
- Η τιμή του μόνο-για-ανάγνωση χαρακτηριστικού is_authenticated είναι πάντα False.
- Οι τιμές των πεδίων is_staff και is_superuser είναι πάντα False.
- Η τιμή του πεδίου is_active είναι πάντα False.
- Οι τιμές των πεδίων groups και user_permissions είναι πάντα κενές.
- Οι μέθοδοι set_password(), check_password(), save() και delete() εγείρουν μία εξαίρεση (raise an exception) NotImplementedError της Python.

3.3.4 Εξουσιοδότηση

Το Django συνοδεύεται από ένα απλό σύστημα δικαιωμάτων (permissions system), το οποίο παρέχει έναν τρόπο εκχώρησης (assigning) δικαιωμάτων σε συγκεκριμένους χρήστες (άτομα) και ομάδες χρηστών (user_groups) αξιοποιώντας τα πολλά-προς-πολλά πεδία (many-to-many fields) user_permissions και group_permissions των αντικειμένων auth.User και auth.Group αντίστοιχα. Τα σχετικά μοντέλα απεικονίζονται μέσω του Django ORM στους ομώνυμους πίνακες της βάσης usermergeDB, από όπου και χρησιμοποιούνται κατά βάση από τον ιστότοπο διαχείρισης Django (Django admin site). Οι προαναφερθέντες πίνακες δημιουργούνται μέσω της διαχειριστικής εντολής μετεγκατάστασης, migrate, και περιγράφονται στα Σχήματα 3.4 και 3.5.



Σχήμα 3.4: Επισκόπηση των πινάκων της βάσης usermergeDB που σχετίζονται με τα δικαιώματα των χρηστών του πλαισίου Django σε ατομικό επίπεδο



Σχήμα 3.5: Επισκόπηση των πινάκων της βάσης usermergeDB που σχετίζονται με τα δικαιώματα των χρηστών του πλαισίου Django σε ομαδικό επίπεδο

Όταν οι εφαρμογές django.contrib.admin και django.contrib.auth είναι εγκατεστημένες (συμπεριλαμβάνονται στη λίστα της ρύθμισης INSTALLED_APPS), ο ιστότοπος διαχείρισης παρέχει έναν βολικό τρόπο για την εμφάνιση και διαχείριση χρηστών, ομάδων και δικαιωμάτων. Χρήστες μπορούν να δημιουργούνται και να διαγράφονται όπως οποιοδήποτε άλλο μοντέλο Django, ενώ είναι επίσης δυνατόν να δημιουργούνται ομάδες και να εκχωρούνται δικαιώματα σε χρήστες ή ομάδες. Τέλος, αποθηκεύεται και εμφανίζεται ένα αρχείο καταγραφής (log) που περιέχει τις τροποποιήσεις των μοντέλων από τους χρήστες εντός του ιστοτόπου διαχείρισης.

3.3.4.1 Δικαιώματα

Ο ιστότοπος διαχείρισης Django αξιοποιεί τα δικαιώματα ως εξής:

- Η πρόσβαση στη φόρμα προσθήκης ("add" form) και η προσθήκη ενός αντικειμένου περιορίζονται στους χρήστες με το δικαίωμα προσθήκης ("add" permission) για αυτόν τον τύπο αντικειμένων.
- Η πρόσβαση στη σελίδα επιλογής αντικειμένου ("change list" page) και στην αντίστοιχη φόρμα επεξεργασίας/αλλαγής ("change" form) καθώς και η επεξεργασία/αλλαγή ενός αντικειμένου περιορίζονται στους χρήστες με το δικαίωμα αλλαγής ("change" permission) για αυτόν τον τύπο αντικειμένων.
- Η διαγραφή ενός αντικειμένου περιορίζεται στους χρήστες με το δικαίωμα διαγραφής ("delete" permission) για αυτόν τον τύπο αντικειμένων.

Όταν η εφαρμογή django.contrib.auth είναι εγκατεστημένη, διασφαλίζει από προεπιλογή ότι δημιουργούνται τα τρία προαναφερθέντα δικαιώματα για κάθε μοντέλο Django που ορίζεται σε μία από τις εγκατεστημένες εφαρμογές μας.

Τα δικαιώματα μπορούν να οριστούν όχι μόνο ανά τύπο αλλά και ανά συγκεκριμένο στιγμιότυπο αντικειμένου (object instance). Χρησιμοποιώντας τις μεθόδους has_add_permission(), has_change_permission() και has_delete_permission() που παρέχονται από την κλάση ModelAdmin, είναι δυνατόν να προσαρμόσουμε (customize) τα δικαιώματα για διαφορετικά στιγμιότυπα αντικειμένων του ίδιου τύπου.

3.3.4.2 Ομάδες

Τα μοντέλα django.contrib.auth.models.Group είναι ένας γενικός τρόπος κατηγοριοποίησης χρηστών που μας επιτρέπει να εφαρμόζουμε δικαιώματα ή κάποια άλλη ετικέτα (label) σε αυτούς. Ένας χρήστης μπορεί να ανήκει σε οποιονδήποτε αριθμό ομάδων.

Ένας χρήστης μέσα σε μία ομάδα έχει αυτομάτως τα δικαιώματα που έχουν χορηγηθεί σε αυτήν. Για παράδειγμα, αν η ομάδα "Site editors" έχει το δικαίωμα can_edit_home_page, οποιοσδήποτε χρήστης μέσα σε αυτή θα έχει το ίδιο δικαίωμα. Πέραν των δικαιωμάτων, οι ομάδες είναι ένας βολικός τρόπος κατηγοριοποίησης χρηστών για να τους αποδοθεί κάποια ετικέτα ή εκτεταμένη λειτουργικότητα (extended functionality). Για παράδειγμα, μπορούμε να δημιουργήσουμε μία ομάδα "Special users" και να γράψουμε κώδικα που θα δίνει πρόσβαση σε ένα τμήμα του ιστοτόπου της εφαρμογής μας ή θα στέλνει μηνύματα ηλεκτρονικού ταχυδρομείου μόνο στα μέλη της.

3.4 Ιστότοπος διαχείρισης του Django[49, 50]

3.4.1 Επισκόπηση[51]

Μία από τις πιο ισχυρές συνιστώσες του Django είναι η αυτόματη διεπαφή διαχείρισης (automatic admin interface). Διαβάζει μεταδεδομένα (metadata) από τα μοντέλα της εφαρμογής μας για να παρέχει μία γρήγορη, μοντελοκεντρική (model-centric) διασύνδεση, όπου οι αξιόπιστοι χρήστες (trusted users) μπορούν να διαχειριστούν περιεχόμενο στον ιστότοπό μας. Η διεπαφή συνιστάται να χρησιμοποιείται ως ένα

εργαλείο εσωτερικής διαχείρισης (internal management tool) του ιστοτόπου καθώς παρέχει πολλές δυνατότητες προσαρμογής περιεχομένου (content customization). Δεν θα πρέπει, όμως, να αποτελεί τη βάση για την κατασκευή ολόκληρου του εμπρόσθιου τμήματός μας καθώς οι δυνατότητες που παρέχει δεν είναι συνήθως αρκετές για όλες τις περιπτώσεις χρήσης (use cases) του ιστοτόπου. Η χρήση μίας πιο διαδικασιοκεντρικής (process-centric) διεπαφής, η οποία περιγράφει με σαφήνεια τις λεπτομέρειες υλοποίησης των πινάκων (tables) και των πεδίων (fields) της βάσης δεδομένων, μπορεί να επιτευχθεί μέσω ορισμού κατάλληλων λειτουργιών/όψεων διαχείρισης Django (Django admin functions/views), όπως εκείνες που περιγράφονται στην ενότητα 3.4.3.

Ο ιστότοπος διαχείρισης Django (Django admin site) είναι προσβάσιμος μέσω της διεύθυνσης http://slub.softlab.ntua.gr/django-admin/, αφού πρώτα δημιουργήσουμε μέσω της εντολής createsuperuser έναν υπερχρήστη που θα μπορεί να εισέρχεται (log in) σε αυτόν. Αξίζει επίσης να σημειωθεί ότι τα στατικά αρχεία, π.χ. CSS, WOFF (Web Open Font Format), SVG (Scalable Vector Graphics) και JS (JavaScript), τα οποία συνδέονται στις σελίδες του ιστοτόπου διαχείρισης, διατηρούνται στον κατάλογο django/contrib/admin/static/admin/[52] και παρατίθενται στη διεύθυνση http://slub.softlab.ntua.gr/static/admin/ μέσω χρήσης της οδηγίας (directive) Alias στις ρυθμίσεις του Apache για τον ιστότοπο της εφαρμογής μας (βλ. βήμα (Ο) του τμήματος "INSTALLING AND CONFIGURING THE APACHE WEB SERVER TO DEPLOY THE PROJECT" στο αρχείο README.md).

3.4.2 Φιλοσοφία διαχείρισης

Η δημιουργία ενός ιστοτόπου διαχείρισης που επιτρέπει στο προσωπικό (staff) ή στους πελάτες μας (clients) να προσθέτουν, να αλλάζουν και να διαγράφουν περιεχόμενο είναι μία κουραστική δουλειά - η δημιουργία διεπαφών οπίσθιου τμήματος μόνο για τη διαχείριση περιεχομένου δεν απαιτεί μεγάλη δημιουργικότητα. Για αυτόν τον λόγο, το Django αυτοματοποιεί εξ ολοκλήρου τη δημιουργία διεπαφών διαχείρισης για μοντέλα.

Μόλις οριστούν τα μοντέλα της εφαρμογής μας, το Django μπορεί να δημιουργήσει αυτόματα μία επαγγελματική, δυναμική, έτοιμη για παραγωγή (production-ready) διεπαφή διαχείρισης - έναν ιστότοπο που επιτρέπει στους αυθεντικοποιημένους χρήστες (authenticated users) να τροποποιούν αντικείμενα των μοντέλων. Για να επιτευχθεί αυτό, αρκεί να καταχωρήσουμε (register) τα μοντέλα στον ιστότοπο διαχείρισης.

Στο Django επικρατεί σαφής διαχωρισμός μεταξύ των «εκδοτών περιεχομένου» ("content publishers") και του «δημόσιου» ιστοτόπου ("public" site): οι διαχειριστές του ιστοτόπου (site managers) χρησιμοποιούν το σύστημα για να προσθέτουν περιεχόμενο που εμφανίζεται στον δημόσιο ιστότοπο. Το Django επιλύει το πρόβλημα της δημιουργίας μίας ενοποιημένης (unified) διεπαφής που επιτρέπει στους διαχειριστές του ιστοτόπου να επεξεργάζονται περιεχόμενο. Η διεπαφή διαχείρισης δεν προορίζεται να χρησιμοποιηθεί από τους επισκέπτες του ιστοτόπου (site visitors), αλλά από τους διαχειριστές του.

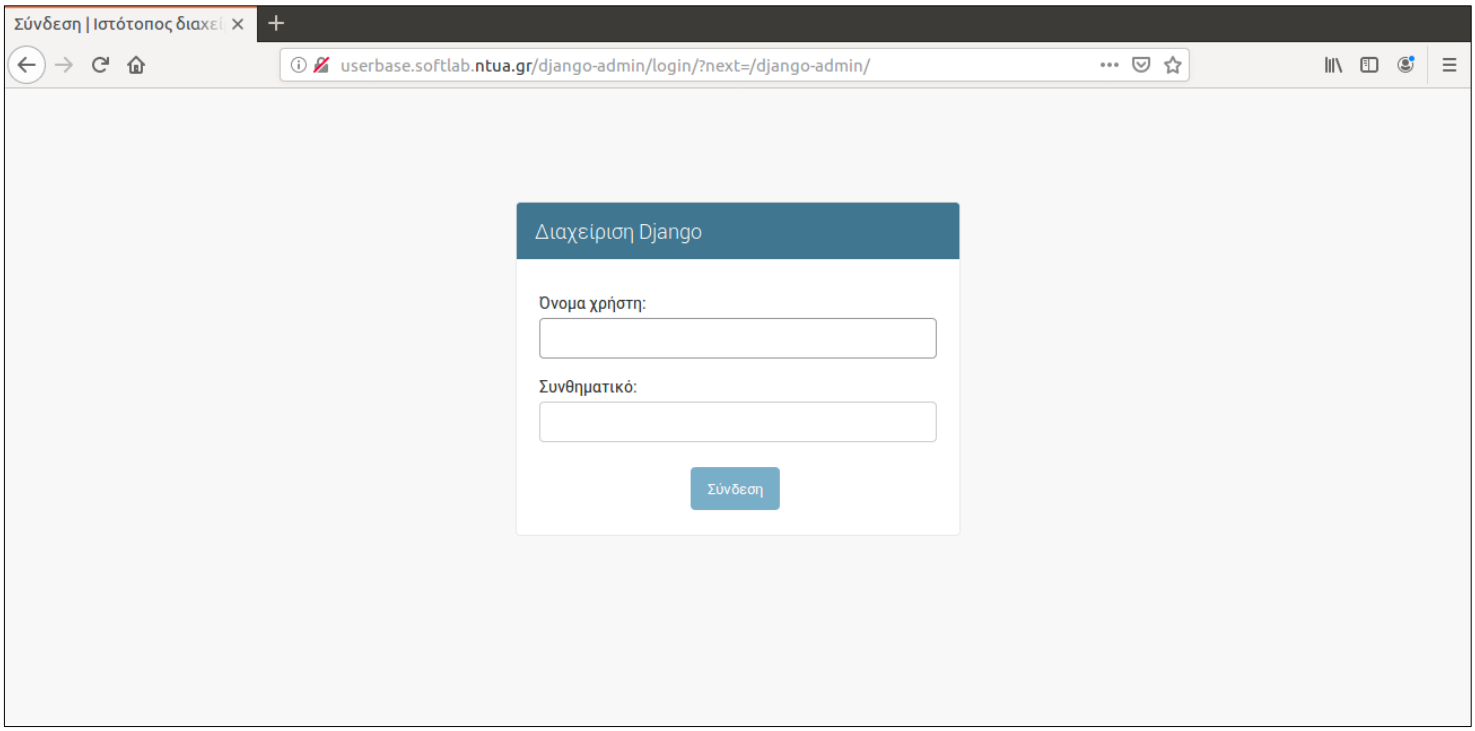
3.4.3 Τυπική ροή εργασίας και χρήση

Μία τυπική ροή εργασίας (workflow) για τη δημιουργία εφαρμογών Django είναι να δημιουργούμε μοντέλα και να εκκινούμε τον ιστότοπο διαχείρισης όσο πιο γρήγορα γίνεται, έτσι ώστε τα μέλη προσωπικού (ή οι πελάτες) μας να μπορούν να ξεκινήσουν τη συμπλήρωση (population) της βάσης δεδομένων. Στη συνέχεια, αναπτύσσουμε τον τρόπο παρουσίασης των δεδομένων στο κοινό.

Ακολουθεί ένα παράδειγμα τυπικής χρήσης του ιστοτόπου διαχείρισης:

(Α) Δημιουργία ενός χρήστη του ιστοτόπου (admin user): Πρώτα θα πρέπει να δημιουργήσουμε έναν χρήστη που μπορεί να εισέλθει στον ιστότοπο διαχείρισης. Αυτό επιτυγχάνεται μέσω εκτέλεσης της διαχειριστικής εντολής (administrative command) createsuperuser που μας προτρέπει να συμπληρώσουμε αναγκαστικά το επιθυμητό όνομα χρήστη (username) και κωδικό πρόσβασης (password) και προαιρετικά το επιθυμητό ηλεκτρονικό ταχυδρομείο (email).

(Β) Μετάβαση στον ιστότοπο μέσω περιηγητή ιστού: Ο ιστότοπος διαχείρισης είναι ενεργοποιημένος από προεπιλογή. Όταν μεταβούμε μέσω ενός περιηγητή ιστού στη διεύθυνση http://slub.softlab.ntua.gr/django-admin/, θα εμφανιστεί στην οθόνη η σελίδα σύνδεσης/εισόδου (login page) του ιστοτόπου που φαίνεται στο Σχήμα 3.6.



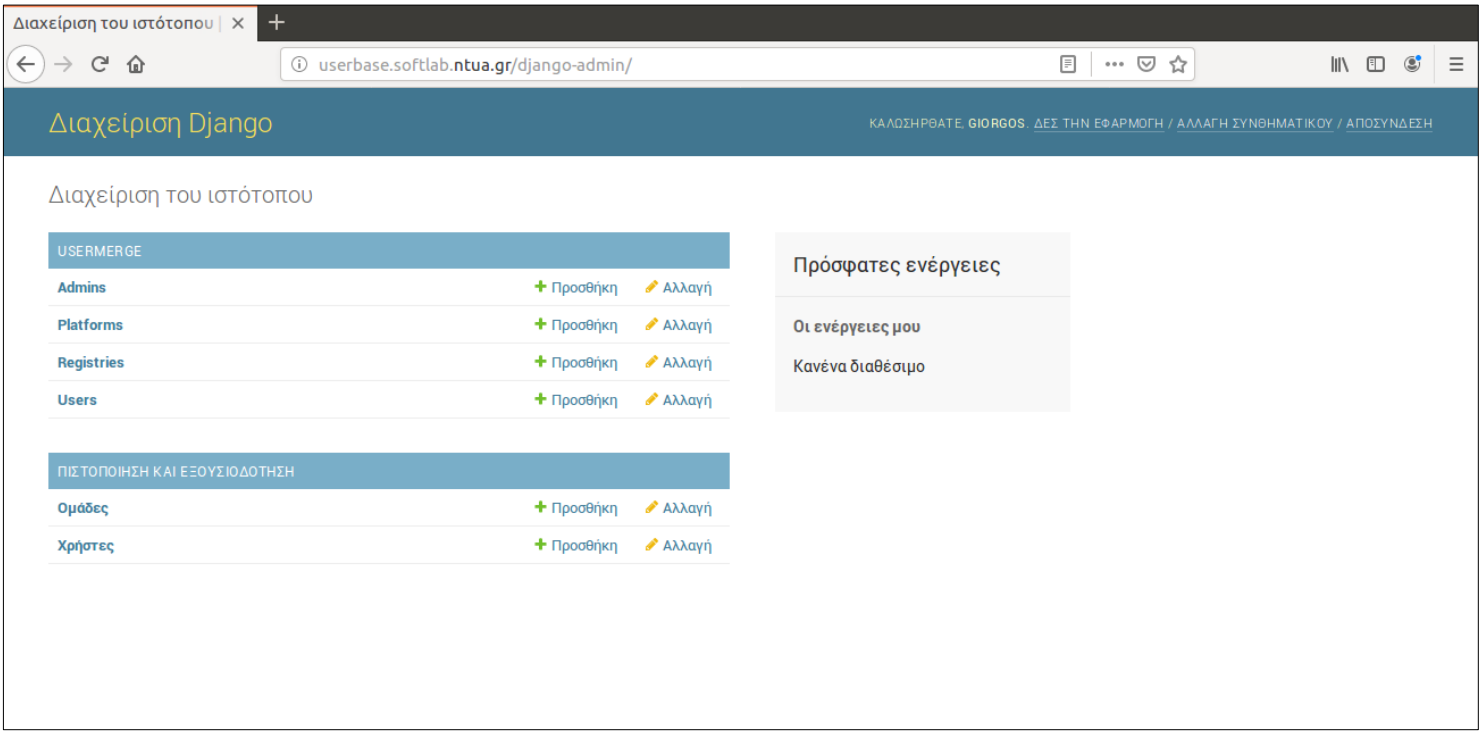
Σχήμα 3.6: Σελίδα σύνδεσης του ιστοτόπου διαχείρισης Django

(Γ) Είσοδος στον ιστότοπο: Δοκιμάζουμε να εισέλθουμε με τον λογαριασμό υπερχρήστη (superuser account) που δημιουργήσαμε σε προηγούμενο βήμα. Στην οθόνη θα πρέπει να εμφανιστεί η σελίδα ευρετηρίου (index page) του ιστοτόπου διαχείρισης. Θα πρέπει ακόμα να εμφανιστούν μερικοί τύποι επεξεργάσιμου

περιεχομένου (editable content): Ομάδες (Groups) και Χρήστες (Users). Οι τελευταίοι παρέχονται από το πλαίσιο αυθεντικοποίησης του Django, django.contrib.auth (στους Χρήστες - χρήστες του πλαισίου Django, όχι της εφαρμογής usermerge - περιλαμβάνεται και ο χρησιμοποιούμενος υπερχρήστης).

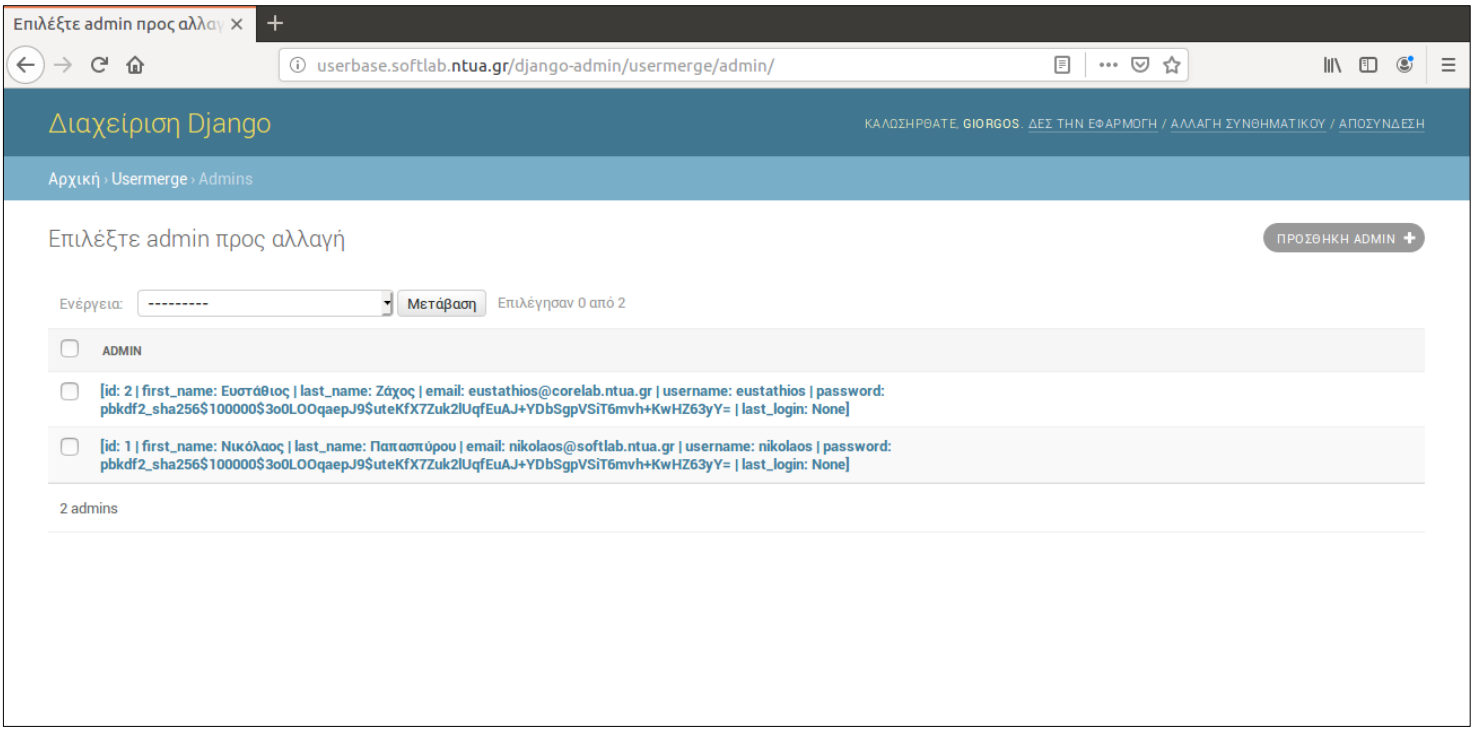
(Δ) Καταχώρηση των μοντέλων στον ιστότοπο: Αν δεν καταχωρήσουμε τα μοντέλα της εφαρμογής μας στον ιστότοπο διαχείρισης, αυτά δεν εμφανίζονται στη σελίδα ευρετηρίου του. Πρέπει, δηλαδή, να δηλώσουμε στον ιστότοπο ότι τα αντικείμενα των μοντέλων είναι τροποποιήσιμα (modifiable) μέσω μίας διεπαφής διαχείρισης. Για να γίνει αυτό, χρησιμοποιούμε τη μέθοδο register() της μονάδας django.contrib.admin.site στο αρχείο admin.py για καθένα από τα μοντέλα.

(Ε) Εξερεύνηση της αυτόματης λειτουργικότητας του ιστοτόπου: Αφού καταχωρηθούν τα μοντέλα, το Django γνωρίζει ότι θα πρέπει να εμφανίζονται στη σελίδα ευρετηρίου του ιστοτόπου διαχείρισης όπως φαίνεται στο Σχήμα 3.7.

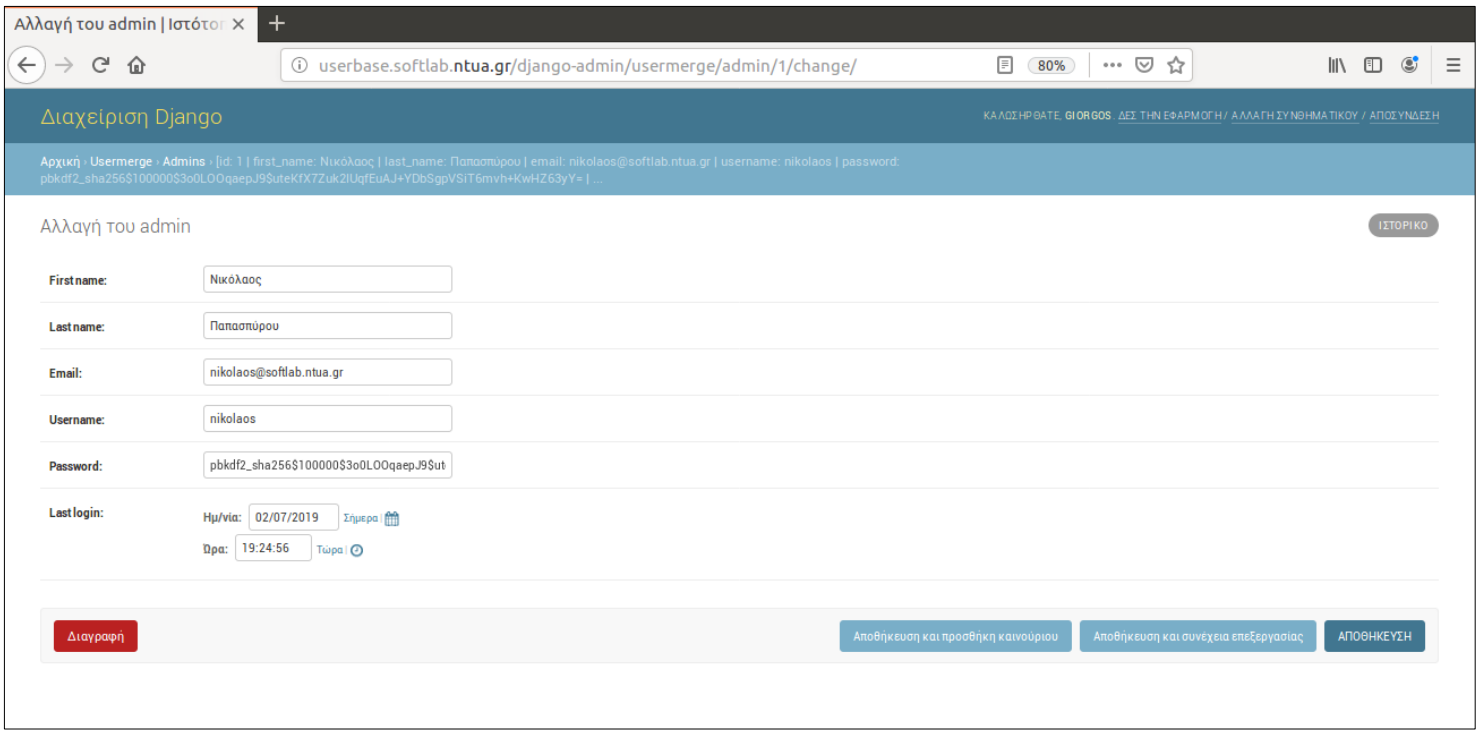


Σχήμα 3.7: Σελίδα ευρετηρίου του ιστοτόπου διαχείρισης Django

Η αυτόματη λειτουργικότητα του ιστοτόπου μπορεί να ελεγχθεί μέσω οποιουδήποτε μοντέλου της εφαρμογής μας διότι η διεπαφή διαχείρισης δρα με τον ίδιο τρόπο για όλα τα μοντέλα από προεπιλογή. Επιλέγουμε, όμως, να εξετάσουμε το μοντέλο Admin λόγω των πολλών και διαφορετικών τύπων πεδίων (field types) που διαθέτει. Κάνουμε κλικ στην επιλογή Admins και μεταφερόμαστε στη σελίδα επιλογής αντικειμένου ("change list" page) του μοντέλου. Όπως φαίνεται στο Σχήμα 3.8, αυτή η σελίδα εμφανίζει όλα τα αντικείμενα Admin, δηλαδή τους διαχειριστές της εφαρμογής μας, που διατηρούνται στη βάση δεδομένων και μας επιτρέπει να επιλέξουμε έναν για να τον αλλάξουμε. Για παράδειγμα, υπάρχει ο διαχειριστής «Νικόλαος Παπασπύρου», τον οποίο και επιλέγουμε για επεξεργασία όπως φαίνεται στο Σχήμα 3.9.



Σχήμα 3.8: Σελίδα επιλογής αντικειμένου του μοντέλου Admin στον ιστότοπο διαχείρισης Django



Σχήμα 3.9: Σελίδα επεξεργασίας ενός αντικειμένου Admin στον ιστότοπο διαχείρισης Django

Αναφορικά με τη φόρμα επεξεργασίας/αλλαγής ("change" form) του αντικειμένου, αξίζει να σημειωθούν τα εξής:

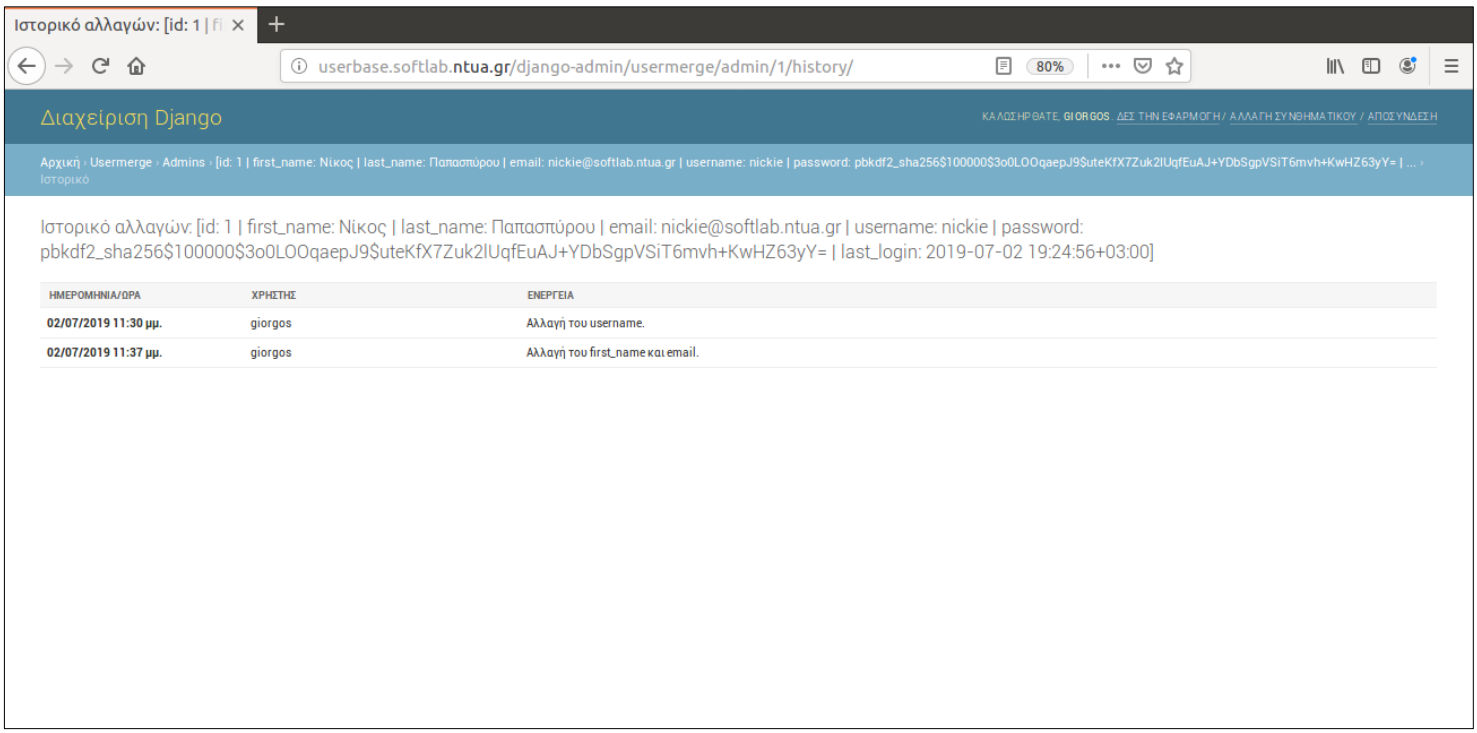
- Η φόρμα δημιουργείται αυτόματα από το μοντέλο Admin.

- Οι διάφοροι τύποι πεδίων του μοντέλου, π.χ. τα πεδία χαρακτήρων (CharFields), το πεδίο ηλεκτρονικού ταχυδρομείου (EmailField) και το πεδίο ημερομηνίας-ώρας (DateTimeField), αντιστοιχούν στο κατάλληλο γραφικό στοιχείο εισόδου HTML (HTML input widget). Κάθε τύπος πεδίου γνωρίζει πώς να εμφανίζεται στον ιστότοπο διαχείρισης.
- Κάθε DateTimeField αποκτά αυτόματες συντομεύσεις (shortcuts) JavaScript. Οι ημερομηνίες αποκτούν μία συντόμευση «Σήμερα» και ένα αναδυόμενο παράθυρο (pop-up window) ημερολογίου, ενώ οι ώρες αποκτούν μία συντόμευση «Τώρα» και ένα βολικό αναδυόμενο παράθυρο που παραθέτει συχνά εισαγόμενες χρονικές στιγμές, π.χ. «Μεσάνυχτα» ή «Μεσημέρι».

Το κάτω μέρος της σελίδας μάς δίνει μερικές επιλογές (options):

- Αποθήκευση (Save): Αποθηκεύει τις αλλαγές και επιστρέφει στη σελίδα επιλογής αντικειμένου για αυτόν τον τύπο αντικειμένων.
- Αποθήκευση και συνέχεια επεξεργασίας (Save and continue editing): Αποθηκεύει τις αλλαγές και επαναφορτώνει (reloads) τη σελίδα επεξεργασίας για αυτό το αντικείμενο.
- Αποθήκευση και προσθήκη άλλου αντικειμένου (Save and add another): Αποθηκεύει τις αλλαγές και φορτώνει μία νέα, κενή φόρμα (blank form) για αυτόν τον τύπο αντικειμένων.
- Διαγραφή (Delete): Εμφανίζει μία σελίδα επιβεβαίωσης διαγραφής (delete confirmation page).

Στο πάνω μέρος της σελίδας υπάρχει η επιλογή «Ιστορικό» ("History"), η οποία εμφανίζει μία σελίδα με τη λίστα όλων των αλλαγών που έχει υποστεί το εξεταζόμενο αντικείμενο μέσω του ιστοτόπου διαχείρισης. Όπως φαίνεται στο Σχήμα 3.10, κάθε αλλαγή συνοδεύεται από μία χρονοσφραγίδα (timestamp) και το όνομα χρήστη (username) του ατόμου που έκανε την αλλαγή.



Σχήμα 3.10: Σελίδα ιστορικού αλλαγών ενός αντικειμένου Admin στον ιστότοπο διαχείρισης Django

Οι αλλαγές αυτές αποθηκεύονται στη βάση δεδομένων ως εγγραφές django.contrib.admin.models.LogEntry και αντλούνται από τον πίνακα django_admin_log, στον οποίο διατηρείται το πλήρες ιστορικό αλλαγών (complete change history), δηλαδή το ιστορικό αλλαγών για όλα τα αντικείμενα ανεξαρτήτως μοντέλου[53]. Ο προαναφερθείς πίνακας περιγράφεται στο Σχήμα 3.11.

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
action_time	datetime(6)	NO		NULL	
object_id	longtext	YES		NULL	
object_repr	varchar(200)	NO		NULL	
action_flag	smallint(5) unsigned	NO		NULL	
change_message	longtext	NO		NULL	
content_type_id	int(11)	YES	MUL	NULL	
user_id	int(11)	NO	MUL	NULL	

Σχήμα 3.11: Επισκόπηση του πίνακα django_admin_log της βάσης usermergeDB

4 Όψεις λειτουργιών και πρότυπα ιστού του SLUB

Σε αντίθεση με το προηγούμενο κεφάλαιο που πραγματευόταν τα μοντέλα δεδομένων του SLUB, το παρόν κεφάλαιο εστιάζει στις όψεις λειτουργιών και στα πρότυπα ιστού της εφαρμογής. Συγκεκριμένα, αναλύει διεξοδικά τη δόμηση και χρησιμοποίηση των μονάδων views.py και templates, καθώς και των υπολοίπων μονάδων του καταλόγου usermerge που τις πλαισιώνουν λειτουργικά, όπως οι urls.py, decorators.py, forms.py, validators.py, models.py και helpers.py (σύνολο βοηθητικών συναρτήσεων που συμβάλλουν στη σωστή εκτέλεση και βελτιωμένη δομή/δομοστοιχείωση - structure/modularity - των συναρτήσεων των κύριων μονάδων, π.χ. views.py). Ο πηγαίος κώδικας και η τεκμηρίωση των προαναφερθεισών μονάδων διατίθενται στη διεύθυνση https://git.softlab.ntua.gr/giorgkazelidis/userbase, ενώ βασίζονται κατά κύριο λόγο στην επίσημη τεκμηρίωση του πλαισίου Django (έκδοση 2.0) που είναι προσβάσιμη μέσω της διεύθυνσης https://docs.djangoproject.com/en/2.0/. Ως εκ τούτου, η τεκμηρίωση των προαναφερθεισών μονάδων περιλαμβάνει, μεταξύ άλλων, τα URL των αντίστοιχων ενοτήτων της τεκμηρίωσης Django.

4.1 Συνεδρίες HTTP, συναλλαγές βάσης δεδομένων και όψεις σφαλμάτων HTTP στο Django

4.1.1 Συνεδρίες HTTP[54]

4.1.1.1 Επισκόπηση

Το Django παρέχει πλήρη υποστήριξη για ανώνυμες συνεδρίες (anonymous sessions). Το πλαίσιο συνεδριών (session framework) επιτρέπει την αποθήκευση και ανάκτηση αυθαίρετων δεδομένων ανά επισκέπτη του ιστοτόπου (site visitor). Αποθηκεύει τα δεδομένα στην πλευρά του διακομιστή και εκτελεί αδιαφανώς (abstracts) την αποστολή και λήψη των cookies. Τα cookies περιέχουν ένα αναγνωριστικό συνεδρίας (session ID) και όχι τα ίδια τα δεδομένα.

Το πλαίσιο συνεδριών Django είναι εξ ολοκλήρου και αποκλειστικά βασισμένο σε cookies. Σε αντίθεση με την PHP, δεν καταφεύγει στην τοποθέτηση του αναγνωριστικού συνεδρίας στις διευθύνσεις URL ως ύστατη λύση. Αυτή είναι μία σκόπιμη απόφαση σχεδιασμού (intentional design decision) καθώς η συμπεριφορά της PHP όχι μόνο ασχημαίνει τις διευθύνσεις URL, αλλά καθιστά και τον ιστότοπο της εφαρμογής (και τον ιστότοπο διαχείρισης Django) ευάλωτο σε κλοπή του αναγνωριστικού συνεδρίας (session-ID theft) μέσω της κεφαλίδας (header) "Referer".

Από προεπιλογή, οι συνεδρίες είναι επίμονες (persistent), δηλαδή δεν απαιτείται οι χρήστες να εισέρχονται στην εφαρμογή (ή στον ιστότοπο διαχείρισης Django) κάθε φορά που ανοίγουν τον περιηγητή ιστού τους (η ρύθμιση SESSION_EXPIRE_AT_BROWSER_CLOSE έχει τεθεί σε False). Συγκεκριμένα, τα cookies των συνεδριών αποθηκεύονται στους περιηγητές ιστού των χρηστών για όσο χρονικό διάστημα ορίζεται στη ρύθμιση SESSION_COOKIE_AGE.

4.1.1.2 Υλοποίηση μέσω του πίνακα django_session

Οι συνεδρίες υλοποιούνται μέσω της εφαρμογής django.contrib.sessions και του μεσισμικού django.contrib.sessions.middleware.SessionMiddleware που συμπεριλαμβάνονται στις ρυθμίσεις INSTALLED_APPS και MIDDLEWARE του αρχείου settings.py αντίστοιχα. Από προεπιλογή, το Django χρησιμοποιεί το οπίσθιο τμήμα django.contrib.sessions.backends.db για να αποθηκεύει τις συνεδρίες στη βάση δεδομένων, δηλαδή τις αντιμετωπίζει ως αντικείμενα του μοντέλου django.contrib.sessions.models.Session που απεικονίζεται μέσω του Django ORM στον πίνακα django_session της βάσης usermergeDB. Ο προαναφερθείς πίνακας δημιουργείται μέσω της διαχειριστικής εντολής μετεγκατάστασης, migrate, και περιγράφεται στο Σχήμα 4.1.

```
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| session_key  | varchar(40) | NO   | PRI | NULL    |       |
| session_data | longtext    | NO   |     | NULL    |       |
| expire_date  | datetime(6) | NO   | MUL | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
```

Σχήμα 4.1: Επισκόπηση του πίνακα django_session της βάσης usermergeDB

Κάθε αντικείμενο Session διαθέτει τα εξής πεδία:

- session_key: Αποτελεί το πρωτεύον κλειδί (primary key) του πίνακα django_session και μπορεί να περιέχει έως και 40 χαρακτήρες. Η τρέχουσα υλοποίηση δημιουργεί μία συμβολοσειρά 32 χαρακτήρων (32-character string) που είναι μία τυχαία ακολουθία (random sequence) από ψηφία (digits) και πεζά γράμματα ASCII (lowercase ASCII - American Standard Code for Information Interchange - letters).
- session_data: Είναι μία συμβολοσειρά που περιέχει ένα λεξικό συνεδρίας (session dictionary), το οποίο είναι σειριοποιημένο σε μορφή JSON (JSON-serialized) και κωδικοποιημένο σε σχήμα Base64 (Base64-encoded). Σύμφωνα με τα αρχεία auth.py[45] και backends.py[48], το λεξικό αυτό περιλαμβάνει τα εξής ζεύγη κλειδιού-τιμής (key-value pairs):
  - "_auth_user_id" → Αναγνωριστικό του χρήστη που έχει εισέλθει στο SLUB ή στον ιστότοπο διαχείρισης Django
  - "_auth_platform" → Όνομα της πλατφόρμας εισόδου (login platform). Το ζεύγος αυτό ορίζεται μόνο στην περίπτωση εισόδου του χρήστη στο SLUB (το μοντέλο του χρήστη είναι usermerge.User ή usermerge.Admin) και όχι στην περίπτωση εισόδου του στον ιστότοπο διαχείρισης Django (το μοντέλο του είναι auth.User). Ειδικότερα, το όνομα της πλατφόρμας λαμβάνεται από το αντίστοιχο πεδίο της φόρμας εισόδου (login form) του SLUB - η φόρμα εισόδου του ιστοτόπου διαχείρισης Django δεν διαθέτει τέτοιο πεδίο - ακριβώς μετά την επιτυχή αυθεντικοποίηση του χρήστη. Αν το προαναφερθέν όνομα είναι "SLUB", τότε το μοντέλο του χρήστη είναι usermerge.Admin· αν το όνομα είναι "Novice", "Grader", "Moodle" ή "PLgrader", τότε το μοντέλο του χρήστη είναι usermerge.User.
  - "_auth_user_backend" → Μονοπάτι (path) του χρησιμοποιούμενου οπίσθιου τμήματος αυθεντικοποίησης. Με βάση τη ρύθμιση

AUTHENTICATION_BACKENDS του αρχείου settings.py, το χρησιμοποιούμενο οπίσθιο τμήμα είναι είτε το django.contrib.auth.backends.ModelBackend σε περίπτωση αυθεντικοποίησης του χρήστη για είσοδο στον ιστότοπο διαχείρισης Django είτε το usermerge.backends.AppUserModelBackend σε περίπτωση αυθεντικοποίησης του χρήστη για είσοδο στο SLUB.

- "_auth_user_hash" → Τιμή κατακερματισμού για τον αντίστοιχο, κωδικοποιημένο μέσω του αλγορίθμου PBKDF2-SHA256 (Password-Based Key Derivation Function 2 using the 256-bit Secure Hash Algorithm), κωδικό πρόσβασης. Η προαναφερθείσα τιμή είναι ένας κωδικός αυθεντικοποίησης μηνύματος που βασίζεται σε κατακερματισμό, HMAC (Hash-based Message Authentication Code), μέσω του αλγορίθμου SHA-1 (160-bit SHA) και υπολογίζεται μέσω της μεθόδου get_session_auth_hash() του μοντέλου του εισελθόντος χρήστη ακριβώς μετά την επιτυχή αυθεντικοποίηση του τελευταίου στο SLUB ή στον ιστότοπο διαχείρισης Django.

  Μετά την εκτέλεση του SessionMiddleware και πριν τη διαβίβαση του αιτήματος HTTP του χρήστη στην αντίστοιχη όψη του SLUB ή του ιστοτόπου διαχείρισης Django, το Django εκτελεί το μεσισμικό AuthenticationMiddleware που, μεταξύ άλλων, υπολογίζει τον HMAC του χρήστη και τον συγκρίνει με εκείνον (τιμή "_auth_user_hash") της συνεδρίας. Αν οι HMAC είναι ίδιοι, δηλαδή ο κωδικός πρόσβασης του χρήστη παραμείνει αμετάβλητος, τότε το αίτημα θα διαβιβαστεί (passed) στην αντίστοιχη όψη για περαιτέρω επεξεργασία. Αν οι HMAC διαφέρουν, δηλαδή ο κωδικός πρόσβασης του χρήστη αλλαχθεί, τότε η συνεδρία θα ακυρωθεί (invalidated)· με άλλα λόγια, το αντίστοιχο αντικείμενο Session θα διαγραφεί από τον πίνακα django_session (το cookie συνεδρίας θα διαγραφεί ταυτόχρονα από την κρυφή μνήμη του περιηγητή ιστού - web/browser cache - του χρήστη) και ο χρήστης θα αποσυνδεθεί (logged out) αυτόματα. Η προαναφερθείσα σύγκριση εκτελείται για κάθε αίτημα, το οποίο αποστέλλεται (μέσω του περιηγητή ιστού) από τον χρήστη, και αφορά όλες τις (ενεργές) συνεδρίες, οι οποίες εγκαθιδρύονται (established) από αυτόν στο SLUB ή στον ιστότοπο διαχείρισης Django.

- expire_date: Είναι ένα αντικείμενο datetime της Python που καθορίζει πότε θα λήξει η συνεδρία. Οι ληγμένες συνεδρίες δεν είναι διαθέσιμες σε έναν χρήστη, αλλά ενδέχεται να παραμείνουν αποθηκευμένες στη βάση δεδομένων έως ότου εκτελεστεί η διαχειριστική εντολή (administrative command) clearsessions.

Το SessionMiddleware εξασφαλίζει, μεταξύ άλλων, ότι κάθε αντικείμενο django.http.request.HttpRequest - το πρώτο όρισμα (argument) σε οποιαδήποτε όψη Django που, κατά κανόνα, ονομάζεται request - θα έχει ένα χαρακτηριστικό (attribute) session, ήτοι request.session, το οποίο είναι ένα λεξικοειδές αντικείμενο (dictionary-like object) της κλάσης django.contrib.sessions.backends.db.SessionStore που διευκολύνει τη διαχείριση των δεδομένων συνεδρίας (session data management) στο αντίστοιχο αντικείμενο Session. Από προεπιλογή, το Django αποθηκεύει στον πίνακα django_session μόνο όταν έχει τροποποιηθεί η συνεδρία, δηλαδή αν έχει εκχωρηθεί ή διαγραφεί τουλάχιστον μία από τις τιμές του αντίστοιχου λεξικού. Αξίζει να σημειωθεί ότι το cookie συνεδρίας αποστέλλεται μόνο όταν έχει δημιουργηθεί ή τροποποιηθεί η συνεδρία (ομοίως, η παράμετρος expires του cookie συνεδρίας ενημερώνεται κάθε

φορά που αυτό αποστέλλεται). Η συνεδρία δεν αποθηκεύεται αν ο κωδικός κατάστασης (status code) της απόκρισης είναι "500 Server Error".

4.1.1.3 Εκκαθάριση του πίνακα django_session

Καθώς οι χρήστες δημιουργούν νέες συνεδρίες στον ιστότοπο της εφαρμογής (και στον ιστότοπο διαχείρισης Django), ενδέχεται να συσσωρεύονται (accumulate) δεδομένα συνεδριών στον πίνακα django_session της βάσης δεδομένων με αποτέλεσμα αυτός να μεγαλώνει ανεξέλεγκτα. Ειδικότερα, όταν ένας χρήστης εισέλθει στο SLUB (ή στον ιστότοπο διαχείρισης Django), το Django προσθέτει μία σχετική εγγραφή στον πίνακα django_session. Το Django ενημερώνει αυτή την εγγραφή κάθε φορά που αλλάζονται τα δεδομένα της συνεδρίας. Αν ο χρήστης εξέλθει χειροκίνητα (manually), το Django διαγράφει την εγγραφή· αν, όμως, δεν εξέλθει, η εγγραφή δεν θα διαγραφεί ποτέ. Το γεγονός αυτό μπορεί να προκαλέσει τη συσσώρευση περιττών εγγραφών στον πίνακα django_session καθώς το Django δεν παρέχει αυτόματη εκκαθάριση (automatic purging) των ληγμένων συνεδριών (expired sessions)· παρέχει, ωστόσο, τη διαχειριστική εντολή εκκαθάρισης clearsessions για αυτόν τον σκοπό. Ως εκ τούτου, συνιστάται η χειροκίνητη κλήση της προαναφερθείσας εντολής σε τακτική βάση, π.χ. ως μία καθημερινή εργασία cron (daily cron job).

4.1.1.4 Αυθεντικοποίηση χρήστη στα εισερχόμενα αιτήματα HTTP

Το Django χρησιμοποιεί το πλαίσιο συνεδριών και το μεσισμικό AuthenticationMiddleware για να προσαρτήσει (hook) το σύστημα αυθεντικοποίησης (authentication system) στα εισερχόμενα (incoming) αιτήματα HTTP. Το AuthenticationMiddleware, το οποίο ορίζεται στο αρχείο middleware.py[55], πρέπει να έπεται του SessionMiddleware στη λίστα της ρύθμισης MIDDLEWARE, και κατ' επέκταση στη σειρά εκτέλεσης (execution order), καθώς το τελευταίο διαχειρίζεται τα δεδομένα συνεδρίας που είναι απαραίτητα για την εκτέλεση του πρώτου.

Μέσω του AuthenticationMiddleware, το Django προσθέτει το χαρακτηριστικό user, ήτοι request.user, που αντιπροσωπεύει τον τρέχοντα χρήστη σε κάθε εισερχόμενο αντικείμενο HttpRequest. Αν ο χρήστης δεν έχει εισέλθει στο SLUB ή στον ιστότοπο διαχείρισης Django, αυτό το χαρακτηριστικό τίθεται σε ένα στιγμιότυπο AnonymousUser, ειδάλλως είναι ένα στιγμιότυπο usermerge.User, usermerge.Admin ή auth.User. Συγκεκριμένα, η υλοποίηση του AuthenticationMiddleware ελέγχει αν υπάρχει το κλειδί "_auth_platform" στο request.session· αν υπάρχει, τότε το request.user προσδιορίζεται μέσω της συνάρτησης usermerge.auth.get_user() (χρησιμοποιείται η μονάδα αυθεντικοποίησης - authentication module - της εφαρμογής, usermerge.auth, δηλαδή το αρχείο auth.py), ειδάλλως μέσω της συνάρτησης django.contrib.auth.get_user() (χρησιμοποιείται η προεπιλεγμένη μονάδα αυθεντικοποίησης του Django, django.contrib.auth).

Για τον έλεγχο εισόδου (login control) του τρέχοντα χρήστη σε μία όψη του SLUB ή του ιστοτόπου διαχείρισης Django, αξιοποιείται το μόνο-για-ανάγνωση χαρακτηριστικό (read-only attribute) is_authenticated του αντίστοιχου στιγμιοτύπου, ήτοι request.user.is_authenticated, το οποίο είναι πάντα False στα στιγμιότυπα AnonymousUser και πάντα True στα στιγμιότυπα usermerge.User, usermerge.Admin

και auth.User. Η αξιοποίηση του request.user.is_authenticated γίνεται συνήθως έμμεσα μέσω του διακοσμητή django.contrib.auth.decorators.login_required()[56] που αποτελεί μία βολική συντόμευση (shortcut) για τον προαναφερθέντα έλεγχο εισόδου. Αν ο χρήστης δεν έχει εισέλθει, ο διακοσμητής τον ανακατευθύνει (redirects) στην ιστοσελίδα εισόδου (login page) του SLUB - το προαιρετικό όρισμα (optional argument) login_url του διακοσμητή ή η ρύθμιση LOGIN_URL του αρχείου settings.py πρέπει να τεθεί στη σχετική διεύθυνση (relative URL) εισόδου του SLUB, "/", δηλαδή να αντιστοιχεί στην απόλυτη διεύθυνση (absolute URL) http://slub.softlab.ntua.gr/ (η ρύθμιση, η οποία είναι "/accounts/login/" από προεπιλογή και οδηγεί σε μία απόκριση με κωδικό κατάστασης "404 Not Found" μέσω της διεύθυνσης http://slub.softlab.ntua.gr/accounts/login/, λαμβάνεται υπόψιν μόνο όταν το όρισμα, το οποίο είναι None από προεπιλογή, δεν έχει καθοριστεί). Αν ο χρήστης έχει εισέλθει, ο διακοσμητής εκτελεί την όψη κανονικά - ο κώδικας της όψης είναι ελεύθερος να υποθέσει ότι ο χρήστης έχει εισέλθει.

Αξίζει να σημειωθεί ότι οι διακοσμητές User_login_required() και Admin_login_required() του αρχείου decorators.py, οι οποίοι περιγράφονται στην ενότητα 4.2.2, χρησιμοποιούν εσωτερικά τον διακοσμητή login_required() για να πραγματοποιήσουν τον έλεγχο εισόδου του τρέχοντα χρήστη σε μία όψη του SLUB.

4.1.2 Συναλλαγές βάσης δεδομένων[57]

4.1.2.1 Προεπιλεγμένη συμπεριφορά του Django

Το Django χρησιμοποιεί συναλλαγές (transactions) ή σημεία αποθήκευσης (savepoints) αυτόματα για να εγγυηθεί την ακεραιότητα (integrity) των λειτουργιών ORM (ORM operations) που απαιτούν πολλαπλές επερωτήσεις (queries), ιδίως επερωτήσεις delete() και update() της κλάσης django.db.models.query.QuerySet. Μία συναλλαγή είναι ένα ατομικό σύνολο επερωτήσεων βάσης δεδομένων. Ακόμα και αν η εφαρμογή καταρρεύσει (crashes), η βάση δεδομένων εγγυάται ότι θα εφαρμοστούν είτε όλες οι αλλαγές είτε καμία από αυτές. Ένα σημείο αποθήκευσης είναι ένας δείκτης (marker) μέσα σε μία συναλλαγή που επιτρέπει την αναίρεση (rollback) ενός μέρους αυτής αντί του συνόλου της.

Για τη διευκόλυνση των προγραμματιστών εφαρμογών, η προεπιλεγμένη συμπεριφορά του Django είναι να τρέχει σε λειτουργία αυτόματης εκτέλεσης (autocommit mode). Όταν η λειτουργία αυτή είναι ενεργοποιημένη (η ρύθμιση AUTOCOMMIT της βάσης δεδομένων έχει τεθεί σε True στο αρχείο settings.py) και καμία συναλλαγή δεν είναι ενεργή, κάθε επερώτηση SQL εμπερικλείεται (enclosed) στη δική της συναλλαγή. Με άλλα λόγια, όχι μόνο κάθε τέτοια επερώτηση εκκινεί μία συναλλαγή, αλλά η συναλλαγή επίσης εκτελείται (committed) ή αναιρείται (rolled back) αυτόματα στη βάση δεδομένων, ανάλογα με το αν η επερώτηση πέτυχε.

4.1.2.2 Εφαρμογή στις όψεις του SLUB

Σε κάποιες όψεις του SLUB, όπου απαιτείται ατομική εκτέλεση μίας σειράς, τουλάχιστον δύο, επερωτήσεων στη βάση δεδομένων για την εξυπηρέτηση του αντίστοιχου αιτήματος HTTP, εφαρμόζεται εξειδικευμένη διαχείριση συναλλαγών

(transaction management). Κάθε τέτοια όψη, συμπεριλαμβανομένου του αντίστοιχου αιτήματος HTTP, περιτυλίσσεται (wrapped) από τον διακοσμητή (decorator) atomic() που την εμπερικλείει (encloses) σε μία συναλλαγή. Στη συναλλαγή αυτή περικλείεται μόνο η εκτέλεση της όψης, δηλαδή το μεσισμικό και η απεικόνιση (rendering) του προτύπου-απόκρισης (template response) που σχετίζονται με την όψη εκτελούνται έξω από τη συναλλαγή.

Πριν κληθεί η όψη, το Django εκκινεί τη σχετική συναλλαγή. Αν η απόκριση παραχθεί χωρίς προβλήματα, το Django θα εκτελέσει τη συναλλαγή. Αν η όψη παραγάγει μία εξαίρεση, το Django θα αναιρέσει τη συναλλαγή. Στον κώδικα της όψης μπορούν επίσης να πραγματοποιηθούν υποσυναλλαγές (subtransactions) με χρήση σημείων αποθήκευσης· αυτό επιτυγχάνεται συνήθως μέσω του διαχειριστή πλαισίου (context manager) atomic(). Στο τέλος της όψης, όμως, θα εκτελεστούν είτε όλες είτε καμία από τις αλλαγές.

Τα σημεία αποθήκευσης δεν είναι ιδιαιτέρως χρήσιμα όταν χρησιμοποιείται η λειτουργία autocommit. Ωστόσο, μόλις ανοιχθεί μία συναλλαγή μέσω του διακοσμητή atomic(), δημιουργείται μία σειρά επερωτήσεων βάσης δεδομένων που αναμένουν μία εντολή COMMIT ή ROLLBACK. Αν δοθεί (issued) μία εντολή ROLLBACK, θα αναιρεθεί ολόκληρη η συναλλαγή. Σε αντίθεση με τη συνάρτηση django.db.transaction.rollback() που θα πραγματοποιούσε πλήρη αναίρεση της συναλλαγής, τα σημεία αποθήκευσης παρέχουν τη δυνατότητα λεπτόκοκκης (fine-grained) αναίρεσής της.

Όταν ο διακοσμητής atomic() είναι εμφωλευμένος (nested), δημιουργεί ένα σημείο αποθήκευσης για να επιτρέψει μερική εκτέλεση ή αναίρεση (partial commit or rollback). Το σημείο αποθήκευσης μπορεί να χρησιμοποιηθεί για την ανάνηψη από ένα σφάλμα βάσης δεδομένων (database error recovery) μέσω πραγματοποίησης μίας μερικής αναίρεσης.

4.1.2.3 Διαχείριση μέσω της συνάρτησης atomic()

Η ατομικότητα (atomicity) είναι η καθοριστική ιδιότητα (defining property) των συναλλαγών βάσης δεδομένων. Η συνάρτηση atomic(), η οποία χρησιμοποιείται τόσο ως διακοσμητής (decorator) όσο και ως διαχειριστής πλαισίου (context manager), ορίζεται στη μονάδα django.db.transaction και επιτρέπει τη δημιουργία ενός μπλοκ κώδικα (code block), μέσα στο οποίο είναι εγγυημένη η ατομικότητα στη βάση δεδομένων. Αν το μπλοκ κώδικα ολοκληρωθεί επιτυχώς, οι αλλαγές θα εκτελεστούν στη βάση δεδομένων. Αν υπάρξει κάποια εξαίρεση, οι αλλαγές θα αναιρεθούν.

Τα ατομικά μπλοκ (atomic blocks) μπορούν να εμφωλεύονται (nested). Σε αυτή την περίπτωση, όταν ένα εσωτερικό μπλοκ (inner block) ολοκληρώνεται επιτυχώς, τα αποτελέσματά του μπορούν ακόμα να αναιρεθούν αν εγερθεί μία εξαίρεση στο εξωτερικό μπλοκ (outer block) σε μία μεταγενέστερη χρονική στιγμή. Η εμφώλευση ενός ατομικού μπλοκ σε ένα try/except μπλοκ της Python επιτρέπει μάλιστα τον φυσικό χειρισμό σφαλμάτων ακεραιότητας (integrity error handling).

Προκειμένου να διασφαλιστεί η ατομικότητα, η συνάρτηση atomic() απενεργοποιεί ορισμένα API. Οποιαδήποτε προσπάθεια εκτέλεσης, αναίρεσης ή αλλαγής της

κατάστασης αυτόματης εκτέλεσης (autocommit state) της σύνδεσης βάσης δεδομένων (database connection) μέσα σε ένα ατομικό μπλοκ εγείρει μία εξαίρεση (raises an exception). Η συνάρτηση atomic() μπορεί να χρησιμοποιηθεί όταν η λειτουργία autocommit είναι απενεργοποιημένη· τότε χρησιμοποιεί μόνο σημεία αποθήκευσης, ακόμα και για το εξώτατο μπλοκ (outermost block).

Στο παρασκήνιο, ο κώδικας διαχείρισης συναλλαγών του Django δρα κατά κανόνα ως εξής:

- Ανοίγει μία συναλλαγή κατά την είσοδο στο εξώτατο ατομικό μπλοκ.
- Δημιουργεί ένα σημείο αποθήκευσης κατά την είσοδο σε ένα εσωτερικό ατομικό μπλοκ.
- Απελευθερώνει (releases) το σημείο αποθήκευσης ή επαναφέρει (rolls back) τη συναλλαγή σε αυτό κατά την έξοδο από ένα εσωτερικό μπλοκ.
- Εκτελεί ή αναιρεί τη συναλλαγή κατά την έξοδο από το εξώτατο μπλοκ.

Η δημιουργία σημείων αποθήκευσης για τα εσωτερικά μπλοκ μπορεί να απενεργοποιηθεί θέτοντας το όρισμα (argument) savepoint σε False. Αν συμβεί κάποια εξαίρεση, το Django θα πραγματοποιήσει την επαναφορά (rollback) κατά την έξοδο από το πρώτο γονικό μπλοκ (first parent block) που διαθέτει σημείο αποθήκευσης· αν δεν υπάρχει τέτοιο, η επαναφορά θα πραγματοποιηθεί από το εξώτατο μπλοκ. Η ατομικότητα εξακολουθεί να είναι εγγυημένη από την εξωτερική συναλλαγή. Αυτή η επιλογή (option) πρέπει να χρησιμοποιείται μόνο αν το λειτουργικό κόστος (overhead) των σημείων αποθήκευσης είναι αισθητό, καθώς έχει το μειονέκτημα ότι σπάει τον χειρισμό σφαλμάτων που αναφέρθηκε παραπάνω.

4.1.2.4 Ταυτοχρονισμός μέσω της μεθόδου select_for_update()[58]

Οι συναλλαγές συχνά εκτελούνται ταυτόχρονα, π.χ. πολλαπλές συναλλαγές μπορούν να διαβάζουν και να γράφουν σε έναν πίνακα την ίδια χρονική στιγμή, με σκοπό την ταχύτερη εξυπηρέτηση των αιτημάτων HTTP από τις όψεις της εφαρμογής. Σε αυτή την περίπτωση, κάθε συναλλαγή πρέπει να είναι απομονωμένη από τις άλλες, δηλαδή δεν πρέπει ούτε να επηρεάζει ούτε να επηρεάζεται από την εκτέλεση των υπολοίπων, ώστε το αποτέλεσμά της να είναι συνεπές (consistent) ως προς τους περιορισμούς (constraints), π.χ. πρωτεύοντα/εξωτερικά κλειδιά, και σωστό ως προς την εκάστοτε κατάσταση (στιγμιότυπο) της βάσης δεδομένων.

Η απομόνωση (isolation) των συναλλαγών επιτυγχάνεται μέσω του ελέγχου ταυτοχρονισμού (concurrency control), ο οποίος διασφαλίζει ότι η ταυτόχρονη εκτέλεση (concurrent execution) των συναλλαγών έχει τον ίδιο αντίκτυπο στη βάση δεδομένων με τη σειριακή εκτέλεσή τους (serial execution)[59]. Ο έλεγχος ταυτοχρονισμού υλοποιείται στο Django μέσω της μεθόδου select_for_update() της κλάσης django.db.models.query.QuerySet, η οποία χρησιμοποιείται, όποτε χρειάζεται, στις όψεις του SLUB που περιτυλίσσονται από τον διακοσμητή atomic(), και αποτρέπει την ανάπτυξη συνθηκών ανταγωνισμού (race conditions) μεταξύ ταυτόχρονων συναλλαγών στη βάση usermergeDB.

Η μέθοδος select_for_update() επιστρέφει ένα σύνολο επερώτησης (queryset) που θα κλειδώσει σειρές (lock rows) μέχρι το τέλος της συναλλαγής, δημιουργώντας μία SQL δήλωση (SQL statement) SELECT FOR UPDATE σε υποστηριζόμενες βάσεις

δεδομένων. Όταν αποτιμηθεί το σύνολο επερώτησης, όλες οι αντίστοιχες εγγραφές θα κλειδωθούν μέχρι το τέλος του μπλοκ συναλλαγής και, κατά συνέπεια, οι υπόλοιπες συναλλαγές δεν θα επιτρέπεται να αλλάξουν ή να αποκτήσουν κλειδώματα για αυτές.

Αν μία άλλη συναλλαγή έχει ήδη αποκτήσει ένα κλείδωμα (acquired a lock) για μία από τις επιλεγμένες σειρές (selected rows), η επερώτηση συνήθως μπλοκάρει (block) μέχρι να απελευθερωθεί το κλείδωμα (released the lock). Για την αποτίμηση του συνόλου επερώτησης (queryset evaluation), δηλαδή, λαμβάνονται υπόψιν και οι σειρές που είναι αρχικά κλειδωμένες από άλλες συναλλαγές· τα ορίσματα (arguments) nowait και skip_locked τίθενται σε False (τα προαναφερθέντα ορίσματα είναι αμοιβαίως αποκλειόμενα - mutually exclusive - και οποιαδήποτε κλήση της μεθόδου select_for_update() με αυτά ταυτοχρόνως ενεργοποιημένα - True - οδηγεί κατά κανόνα σε μία εξαίρεση ValueError της Python).

Από προεπιλογή, η μέθοδος select_for_update() κλειδώνει όλες τις σειρές που έχουν επιλεγεί από την επερώτηση - το όρισμα of τίθεται σε (), δηλαδή στην κενή πλειάδα (empty tuple). Για παράδειγμα, εκτός από τις σειρές του μοντέλου αναφοράς (reference model) του συνόλου επερώτησης, κλειδώνονται και εκείνες των σχετικών αντικειμένων που καθορίζονται στη μέθοδο select_related() της κλάσης django.db.models.query.QuerySet.

Το οπίσθιο τμήμα django.db.backends.mysql, το οποίο αποτελεί το RDBMS της εφαρμογής, υποστηρίζει τη μέθοδο select_for_update() (με τις προεπιλεγμένες τιμές των ορισμάτων που αναφέρθηκαν παραπάνω), αλλά δεν υποστηρίζει τα ορίσματά της. Επομένως, οποιαδήποτε κλήση της στο SLUB με nowait=True, skip_locked=True ή of≠() εγείρει μία εξαίρεση django.db.NotSupportedError για να αποτρέψει αιφνίδιο μπλοκάρισμα (blocking) του κώδικα.

Στα πλαίσια ενός οπίσθιου τμήματος βάσης δεδομένων που υποστηρίζει την SQL δήλωση SELECT FOR UPDATE, η αποτίμηση ενός συνόλου επερώτησης με τη μέθοδο select_for_update() σε λειτουργία autocommit οδηγεί σε μία εξαίρεση django.db.transaction.TransactionManagementError επειδή οι σειρές δεν κλειδώνονται σε εκείνη την περίπτωση (απαιτείται εγκαθίδρυση συναλλαγής - transaction establishment - μέσω της συνάρτησης atomic()). Αν η προαναφερθείσα αποτίμηση επιτρεπόταν, θα διευκόλυνε την αλλοίωση των δεδομένων (data corruption).

4.1.3 Όψεις σφαλμάτων HTTP[60]

Το Django διαθέτει από προεπιλογή μερικές όψεις για τον χειρισμό σφαλμάτων HTTP (HTTP error handling).

4.1.3.1 Όψη 404 (Not Found)

Όταν εγείρεται μία εξαίρεση django.http.Http404 μέσα από μία όψη, το Django φορτώνει από προεπιλογή την εξειδικευμένη όψη django.views.defaults.page_not_found() για τον χειρισμό των σφαλμάτων 404. Η προαναφερθείσα όψη είτε παράγει ένα πολύ απλό μήνυμα "Not Found" είτε φορτώνει

και απεικονίζει (renders) το πρότυπο 404.html, όπως φαίνεται στο Σχήμα 4.2, αν αυτό έχει δημιουργηθεί στον ριζικό κατάλογο προτύπων (root template directory), templates.

Η προεπιλεγμένη όψη 404 θα διαβιβάσει (pass) δύο μεταβλητές στο πρότυπο: τη request_path, η οποία είναι η διεύθυνση URL που οδήγησε στο σφάλμα, και την exception, η οποία είναι μία χρήσιμη αναπαράσταση της εξαίρεσης που πυροδότησε (triggered) την όψη, π.χ. περιέχει το μήνυμα - αν υπήρξε τέτοιο - που διαβιβάστηκε σε ένα συγκεκριμένο στιγμιότυπο Http404.

Πρέπει ακόμα να σημειωθούν τα εξής:

- Η όψη 404 καλείται επίσης όταν το Django ελέγξει κάθε κανονική έκφραση (regular expression) στη διαμόρφωση URL (URLconf) και δεν βρει καμία αντιστοιχία.
- Στην όψη 404 διαβιβάζεται ένα στιγμιότυπο django.template.RequestContext, μέσω του οποίου αυτή θα έχει πρόσβαση σε μεταβλητές, π.χ. ρύθμιση MEDIA_URL, που παρέχονται από τους επεξεργαστές πλαισίων προτύπων (template context processors).
- Αν η ρύθμιση DEBUG έχει τεθεί σε True στο αρχείο settings.py, τότε η όψη 404 δεν θα χρησιμοποιηθεί ποτέ· αντίθετα, θα εμφανιστεί η διαμόρφωση URL με κάποιες πληροφορίες αποσφαλμάτωσης (debug information).

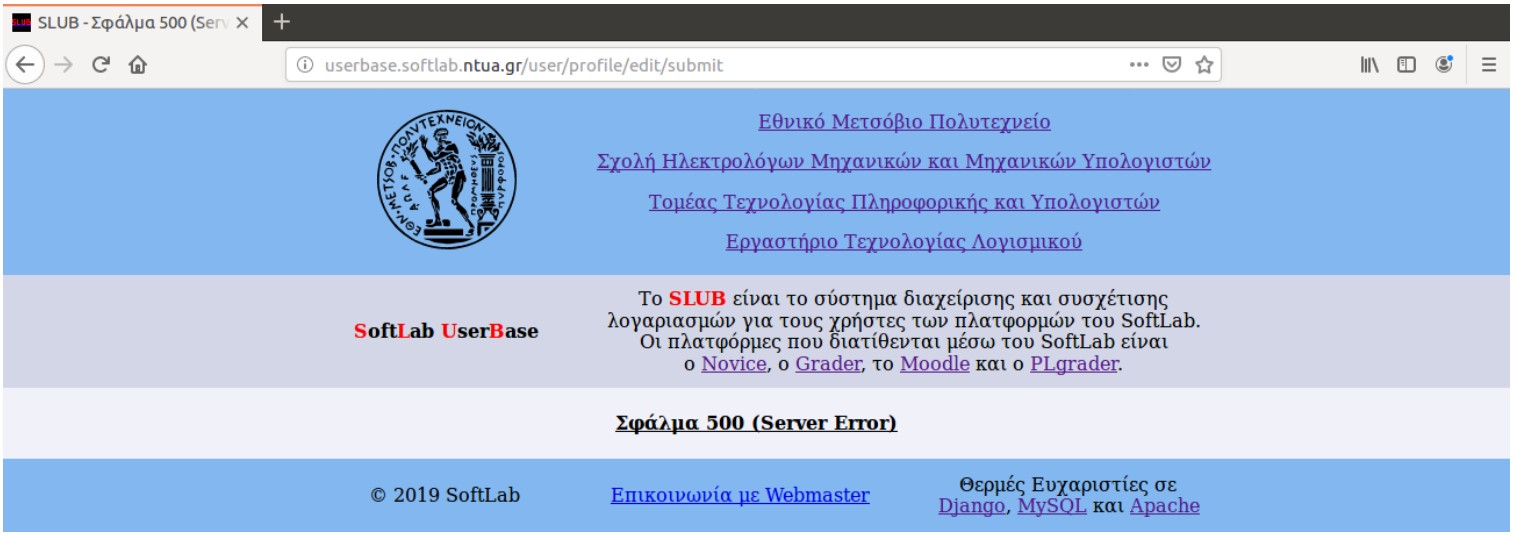
Σχήμα 4.2: Σελίδα σφάλματος HTTP 404 (Not Found)

4.1.3.2 Όψη 500 (Server Error)

Ομοίως, το Django αντιμετωπίζει με εξειδικευμένο τρόπο τα σφάλματα χρόνου εκτέλεσης (runtime errors) στον κώδικα των όψεων. Αν μία όψη έχει ως αποτέλεσμα μία εξαίρεση, το Django θα καλέσει από προεπιλογή την όψη django.views.defaults.server_error(), η οποία είτε παράγει ένα πολύ απλό μήνυμα "Server Error" είτε φορτώνει και απεικονίζει το πρότυπο 500.html, όπως φαίνεται στο Σχήμα 4.3, αν αυτό έχει δημιουργηθεί στον ριζικό κατάλογο προτύπων, templates.

Η προεπιλεγμένη όψη 500 δεν διαβιβάζει μεταβλητές στο πρότυπο 500.html και απεικονίζεται με ένα κενό στιγμιότυπο django.template.Context για να μειώσει την πιθανότητα επιπρόσθετων σφαλμάτων.

Αν η ρύθμιση DEBUG έχει τεθεί σε True στο αρχείο settings.html, τότε η όψη 500 δεν θα χρησιμοποιηθεί ποτέ· αντίθετα, θα εμφανιστούν τα ίχνη της στοίβας κλήσεων (call stack traceback) με κάποιες πληροφορίες αποσφαλμάτωσης.



Σχήμα 4.3: Σελίδα σφάλματος HTTP 500 (Server Error)

4.1.3.3 Όψη 403 (Forbidden)

Στο ίδιο πνεύμα με τις όψεις 404 και 500, το Django έχει μία όψη για τον χειρισμό των σφαλμάτων 403. Αν μία όψη έχει ως αποτέλεσμα μία εξαίρεση 403, τότε το Django θα καλέσει από προεπιλογή την όψη django.views.defaults.permission_denied().

Αυτή η όψη φορτώνει και απεικονίζει το πρότυπο 403.html του ριζικού καταλόγου προτύπων, templates, όπως φαίνεται στο Σχήμα 4.4· αντίθετα, αν δεν υπάρχει αυτό το αρχείο, παραθέτει το κείμενο "403 Forbidden" σύμφωνα με το τμήμα (section) 6.5.3 του RFC (Request For Comments) 7231 (της προδιαγραφής HTTP 1.1). Το πλαίσιο του προτύπου (template context) περιέχει τη μεταβλητή exception, η οποία είναι η συμβολοσειραϊκή αναπαράσταση (string representation) της εξαίρεσης που πυροδότησε την όψη.

Η προαναφερθείσα όψη πυροδοτείται μέσω έγερσης μίας εξαίρεσης (raising an exception) django.core.exceptions.PermissionDenied και μπορεί να χρησιμοποιηθεί για να απαγορεύσει την πρόσβαση σε μία όψη.

Σχήμα 4.4: Σελίδα σφάλματος HTTP 403 (Forbidden)

4.1.3.4 Όψη 400 (Bad Request)

Όταν μία εξαίρεση django.core.exceptions.SuspiciousOperation εγείρεται στο Django, μπορεί να αντιμετωπιστεί από μία συνιστώσα του Django, π.χ. μέσω επαναφοράς (resetting) των δεδομένων συνεδρίας. Αν δεν αντιμετωπιστεί συγκεκριμένα, το Django θα θεωρήσει το τρέχον αίτημα ως «κακό αίτημα» αντί για σφάλμα διακομιστή.

Η όψη django.views.defaults.bad_request() είναι παρόμοια με την όψη server_error(), αλλά επιστρέφει με τον κωδικό κατάστασης (status code) 400 που δηλώνει ότι η κατάσταση σφάλματος ήταν το αποτέλεσμα μίας λειτουργίας πελάτη (client operation). Από προεπιλογή, στο πλαίσιο του προτύπου 400.html, το οποίο παρουσιάζεται στο Σχήμα 4.5, δεν διαβιβάζεται τίποτα από όσα σχετίζονται με την εξαίρεση που πυροδότησε την όψη, καθώς το μήνυμα εξαίρεσης ενδέχεται να περιέχει ευαίσθητες πληροφορίες (sensitive information), π.χ. μονοπάτια συστήματος αρχείων (filesystem paths).

Η όψη bad_request() χρησιμοποιείται επίσης μόνο όταν η ρύθμιση DEBUG είναι False.

Σχήμα 4.5: Σελίδα σφάλματος HTTP 400 (Bad Request)

4.2 Τυπική ροή λειτουργίας του SLUB

4.2.1 Επισκόπηση

Το SLUB παρέχει διάφορες λειτουργίες στους χρήστες του με βάση τον ρόλο/μοντέλο τους. Στους απλούς χρήστες ή χρήστες User παρέχει τις λειτουργίες επεξεργασίας, αναζήτησης (για ανάκτηση) και ανάκτησης προφίλ απλού χρήστη, ενώ στους διαχειριστές ή χρήστες Admin παρέχει τις λειτουργίες προσθήκης διαπιστευτηρίων και δημιουργίας αναφορών για απλούς χρήστες. Υπάρχουν, βέβαια, και λειτουργίες που απευθύνονται τόσο στους χρήστες User όσο και στους χρήστες Admin, όπως η είσοδος στην εφαρμογή και η έξοδος από αυτήν, καθώς και συμπληρωματικές λειτουργίες (supplementary functions) που διευκολύνουν την εκτέλεση των προαναφερθέντων κύριων λειτουργιών (primary functions), όπως η εμφάνιση των αρχικών ιστοσελίδων για τους χρήστες User/Admin και των προεπιλεγμένων ιστοσελίδων για τις κύριες λειτουργίες.

Η επιχειρηματική λογική, δηλαδή το σύνολο των βημάτων/ενεργειών, μίας λειτουργίας εμπεριέχεται στην αντίστοιχη όψη του SLUB στο αρχείο views.py[61], δηλαδή κάθε όψη του SLUB επιτελεί και μία λειτουργία του. Κάθε όψη συσχετίζεται μέσω της διαμόρφωσης URL του Django (βλ. slub/slub/urls.py και slub/usermerge/urls.py[62]) με ένα μοναδικό URL, το οποίο μπορεί να χρησιμοποιηθεί από τον χρήστη για την προσπέλαση της αντίστοιχης λειτουργίας μέσω του περιηγητή ιστού (πελάτη), εφόσον πρώτα πραγματοποιηθούν επιτυχώς οι σχετικοί έλεγχοι πρόσβασης.

Οι όψεις συνήθως αξιοποιούν τα μοντέλα δεδομένων της εφαρμογής (βλ. models.py[63]) και επιστρέφουν ένα σχετικό πρότυπο ιστού με κωδικοποίηση UTF-8 (8-bit Unicode Transformation Format encoding) ως έξοδο (εξαίρεση αποτελούν οι συμπληρωματικές όψεις/λειτουργίες που απλά επιστρέφουν ένα τυποποιημένο πρότυπο HTML ως έξοδο). Ειδικότερα, συναλλάσσονται με τη βάση usermergeDB για να δημιουργήσουν, να διαβάσουν, να ενημερώσουν ή/και να διαγράψουν εγγραφές των πινάκων της και στη συνέχεια επιστρέφουν ένα πρότυπο ιστού, π.χ. HTML ή CSV, με τα σχετικά μηνύματα ή/και δεδομένα επιτυχίας/σφαλμάτων.

Κάθε όψη συσχετίζεται γενικά με ένα συγκεκριμένο πρότυπο HTML (ιστοσελίδα) που ορίζεται στον κατάλογο templates[64] και συχνά περιέχει μία φόρμα, μέσω της οποίας ο χρήστης πρέπει να δηλώσει στην αντίστοιχη όψη τα απαιτούμενα δεδομένα εισόδου για την επιτέλεση της λειτουργίας της, και ειδικότερα για τη συναλλαγή της με τη βάση δεδομένων. Η καταλληλότητα των δηλωθεισών τιμών για τα πεδία της σχετικής φόρμας ελέγχεται τόσο κατά την υποβολή τους στην πλευρά του πελάτη όσο και μετά από την υποβολή τους στην πλευρά του διακομιστή.

4.2.2 Βασικές παρατηρήσεις και βήματα

Πριν αναλυθεί η τυπική ροή λειτουργίας του SLUB, πρέπει να σημειωθούν τα εξής:

- Για τη λήψη μίας στατικής ιστοσελίδας (συμπεριλαμβανομένων των συνδεδεμένων αρχείων CSS, ICO - ICOn, SVG - Scalable Vector Graphics και PNG - Portable Network Graphics) που δεν συνοδεύεται από υποβολή δεδομένων μέσω φόρμας, ο περιηγητής ιστού αποστέλλει στον διακομιστή του SLUB, slub.softlab.ntua.gr / slub.softlab.ece.ntua.gr, ένα αίτημα HTTP GET (τα στατικά αρχεία, τα οποία

συνδέονται στις ιστοσελίδες της εφαρμογής, διατηρούνται στον κατάλογο slub/usermerge/static/[65] και παρατίθενται στη διεύθυνση http://slub.softlab.ntua.gr/static/ μέσω χρήσης της οδηγίας - directive - Alias στις ρυθμίσεις του Apache για τον ιστότοπο της εφαρμογής - βλ. βήμα (O) του τμήματος "INSTALLING AND CONFIGURING THE APACHE WEB SERVER TO DEPLOY THE PROJECT" στο αρχείο README.md). Αντίθετα, για τη λήψη ενός (στατικού) προτύπου ιστού που συνοδεύεται από υποβολή δεδομένων μέσω φόρμας, ο περιηγητής ιστού αποστέλλει στον διακομιστή του SLUB ένα αίτημα HTTP POST. Το εκάστοτε αίτημα προωθείται από τον διακομιστή στον κατανεμητή URL του Django και ο τελευταίος το οδηγεί στην αρμόδια όψη του SLUB, η οποία επιστρέφει μέσω του διακομιστή το ζητηθέν πρότυπο ιστού, π.χ. HTML ή CSV, ως απόκριση (με κατάσταση - status - "200 OK") στον περιηγητή.

Η χρήση της μεθόδου POST σε ένα αίτημα HTTP, η οποία αφορά την υποβολή δεδομένων μέσω φόρμας HTML, εξασφαλίζει ότι τα δεδομένα θα αποθηκευτούν στο σώμα του αιτήματος (request body), το οποίο δεν υπόκειται γενικά σε περιορισμούς μήκους δεδομένων, και ότι το αίτημα δεν θα αποθηκευτεί στην κρυφή μνήμη ιστού, δεν θα εισαχθεί στο ιστορικό περιήγησης (ή άλλο αρχείο καταγραφής διακομιστή) και δεν θα δεχθεί σελιδοδείκτη (bookmark). Τα ακριβώς αντίθετα ισχύουν στην περίπτωση χρήσης της μεθόδου GET, η οποία «αναγκάζει» τα δεδομένα να αποσταλούν ως περιορισμένου μήκους συμβολοσειρά (ζεύγη κλειδιού-τιμής) μαζί με το URL του αιτήματος. Συνεπώς, ένα αίτημα HTTP GET δεν πρέπει ποτέ να χρησιμοποιείται για την αποστολή δεδομένων που έχουν πολύ μεγάλο μήκος, είναι σε δυαδική μορφή (π.χ. εικόνες και αρχεία κειμένου), είναι ευαίσθητα (π.χ. κωδικοί πρόσβασης) ή/και τροποποιούν την κατάσταση (π.χ. βάση δεδομένων) του συστήματος διακομιστή, ενώ ένα αίτημα POST, το οποίο μπορεί να συνδυαστεί με άλλους μηχανισμούς προστασίας του Django όπως ο μηχανισμός CSRF (Cross-Site Request Forgery), ενδείκνυται για τέτοιες περιπτώσεις όπου, μεταξύ άλλων, απαιτείται ασφάλεια και έλεγχος πρόσβασης.

- Μία λειτουργία απευθύνεται συνήθως σε εισελθόντες (συνδεδεμένους) χρήστες ενός συγκεκριμένου ρόλου/μοντέλου και η αντίστοιχη όψη πρέπει, ως εκ τούτου, να είναι προσπελάσιμη μόνο από αυτούς. Ειδικότερα, η όψη περιτυλίσσεται (wrapped) από ακριβώς έναν διακοσμητή εκ των User_login_required() και Admin_login_required(), οι οποίοι ορίζονται στο αρχείο decorators.py[66] και εκτελούν τους σχετικούς ελέγχους πρόσβασης.

  Όταν ένα αίτημα HTTP φθάσει στην όψη από τον περιηγητή ιστού, ο διακοσμητής αξιοποιεί τις πληροφορίες που παρέχονται για αυτό από τη διαχείριση συνεδριών του Django και ελέγχει αφενός ότι ο χρήστης έχει εισέλθει στο SLUB - αν όχι (το μοντέλο χρήστη είναι AnonymousUser), ο διακοσμητής δεν επιτρέπει στην όψη να επιτελέσει τη λειτουργία της, αλλά ανακατευθύνει (redirects) τον περιηγητή στην ιστοσελίδα εισόδου/σύνδεσης ώστε να υποδείξει στον χρήστη ότι απαιτείται είσοδος στην εφαρμογή για πρόσβαση στη λειτουργία· ο διακοσμητής προκαλεί την ανακατεύθυνση στην προαναφερθείσα ιστοσελίδα αποστέλλοντας την αντίστοιχη απόκριση με κατάσταση "302 Found" στον περιηγητή μέσω του διακομιστή του SLUB - και αφετέρου ότι έχει τον απαιτούμενο ρόλο/μοντέλο, User ή Admin, για να προσπελάσει την όψη - αν όχι (το μοντέλο χρήστη είναι είτε αντίθετο αυτού που απαιτεί ο διακοσμητής είτε auth.User), ο διακοσμητής δεν επιτρέπει στην όψη να επιτελέσει τη λειτουργία της, αλλά επιστρέφει μέσω του διακομιστή μία ιστοσελίδα σφάλματος HTTP (HTTP error page) με κωδικό (code)

"403 Forbidden" στον περιηγητή ώστε να υποδείξει στον χρήστη ότι απαγορεύεται σε χρήστες με αυτόν τον ρόλο να αποκτήσουν πρόσβαση στη λειτουργία· ο διακοσμητής προκαλεί την επιστροφή της προαναφερθείσας ιστοσελίδας εγείροντας μία εξαίρεση (raising an exception) PermissionDenied. Αν ο χρήστης έχει εισέλθει στο SLUB και έχει τον απαιτούμενο ρόλο/μοντέλο, τότε ο διακοσμητής τού επιτρέπει να προσπελάσει την όψη, και κατ' επέκταση τη λειτουργία που αυτή επιτελεί.

Τα βήματα της τυπικής ροής μίας (κύριας) λειτουργίας του SLUB είναι τα εξής:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας: Ο χρήστης του SLUB επιθυμεί να χρησιμοποιήσει μία συγκεκριμένη λειτουργία της εφαρμογής. Συνεπώς, θα πρέπει να μεταβεί μέσω του περιηγητή ιστού στην αντίστοιχη σελίδα (πρότυπο HTML) του ιστοτόπου της εφαρμογής, http://slub.softlab.ntua.gr/, εισάγοντας το σχετικό URL στην μπάρα διευθύνσεων (address bar) ή επιλέγοντας τον σχετικό υπερσύνδεσμο από άλλη σελίδα του ιστοτόπου. Η μετάβαση στην (προεπιλεγμένη) ιστοσελίδα επιτρέπεται μόνο αν επιτύχει ο σχετικός έλεγχος πρόσβασης με βάση τον ρόλο του χρήστη (αν ορίζονται και άλλοι παρεμφερείς έλεγχοι στην αρχή της αντίστοιχης συμπληρωματικής όψης, τότε πρέπει να επιτύχουν και αυτοί), ειδάλλως απαγορεύεται.

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη: Αν επιτραπεί στον χρήστη η μετάβαση στην ιστοσελίδα της λειτουργίας, τότε αυτός θα αποκτήσει πρόσβαση και στη σχετική φόρμα, μέσω της οποίας θα πρέπει να δηλώσει (συμπληρώσει/ενημερώσει) στην αντίστοιχη (κύρια) όψη τα απαιτούμενα δεδομένα εισόδου για την επιτέλεση της λειτουργίας, και ειδικότερα για τη συναλλαγή με τη βάση usermergeDB. Η καταλληλότητα των δηλωθεισών τιμών για τα πεδία της φόρμας ελέγχεται κατά την υποβολή τους στον διακομιστή του SLUB μέσω του περιηγητή ιστού.

Οι περιορισμοί για τις τιμές των πεδίων ορίζονται στον πηγαίο κώδικα HTML5 της ιστοσελίδας, αφορούν την απαίτηση συμπλήρωσης, τον τύπο, το (ελάχιστο και μέγιστο) μήκος ή/και το μοτίβο κανονικής έκφρασης (regex pattern) της τιμής ενός πεδίου και ελέγχονται για όλα τα πεδία από τη στιγμή που θα πατηθεί το κουμπί υποβολής (submit button) της φόρμας. Αν παραβιαστεί κάποιος περιορισμός σε ένα πεδίο, τότε ο έλεγχός του αποτυγχάνει, δηλαδή η τιμή του θεωρείται ακατάλληλη ως προς τον συγκεκριμένο περιορισμό και οι υπόλοιποι περιορισμοί του δεν ελέγχονται καν. Αν αποτύχει ο έλεγχος για κάποιο πεδίο, τότε αποτυγχάνει η υποβολή ολόκληρης της φόρμας, «κοκκινίζουν» περιμετρικά όλα τα πεδία με ακατάλληλες τιμές και εμφανίζεται μόνο στο πρώτο από αυτά (με βάση τη σειρά παρουσίασής τους στη φόρμα) ένα αναδυόμενο παράθυρο με το σχετικό μήνυμα σφάλματος. Η υποβολή της φόρμας επιτυγχάνει μόνο όταν επιτύχουν οι έλεγχοι για όλα τα πεδία.

Οι έλεγχοι των πεδίων επαναλαμβάνονται με τον ίδιο τρόπο κάθε φορά που επιχειρείται υποβολή της φόρμας. Επειδή μάλιστα πραγματοποιούνται στην πλευρά του πελάτη, είναι ευάλωτοι τόσο σε τυχαίες όσο και σε κακόβουλες HTML/JavaScript τροποποιήσεις της φόρμας από τον χρήστη μέσω των προγραμματιστικών εργαλείων, π.χ. «Επιθεώρηση (Στοιχείων)» - "(Element) Inspector" και «Κονσόλα (Ιστού)» - "(Web) Console", του περιηγητή ιστού. Επομένως, οι έλεγχοι στην πλευρά του πελάτη είναι πιο πολύ τυπικού χαρακτήρα καθώς είναι γενικά απλοί και αναξιόπιστοι.

(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας: Όταν επιτύχει η υποβολή της φόρμας στην πλευρά του πελάτη, ο περιηγητής ιστού αποστέλλει το αντίστοιχο αίτημα HTTP POST, στο σώμα του οποίου έχουν αποθηκευτεί οι υποβληθείσες τιμές για τα πεδία της φόρμας, στην αρμόδια όψη μέσω του διακομιστή του SLUB.

Αρχικά εκτελείται ο έλεγχος πρόσβασης στην όψη με βάση τον ρόλο του χρήστη· η προσπέλαση της όψης επιτρέπεται στον χρήστη μόνο αν επιτύχει ο συγκεκριμένος έλεγχος (αν ορίζονται και άλλοι παρεμφερείς έλεγχοι στην αρχή της όψης, τότε πρέπει να επιτύχουν και αυτοί), ειδάλλως απαγορεύεται. Αν ο έλεγχος πρόσβασης επιτύχει (και επιτύχουν και όλοι οι παρεμφερείς έλεγχοι), ακολουθούν οι έλεγχοι επικύρωσης της φόρμας, δηλαδή οι έλεγχοι που αφορούν την επικύρωση των υποβληθεισών τιμών για τα πεδία της φόρμας. Οι έλεγχοι αυτοί εκτελούνται στην αρχή της όψης μέσω μίας φόρμας Django, η οποία ορίζεται στο αρχείο forms.py[67] και είναι συμβατή ως προς τα πεδία με την αντίστοιχη φόρμα HTML. Παρόλο που οι έλεγχοι εκτελούνται με παρόμοιο τρόπο και στις δύο φόρμες, η φόρμα Django περιλαμβάνει γενικά περισσότερους και πιο εξειδικευμένους ελέγχους πεδίων από την αντίστοιχη φόρμα HTML, οι οποίοι πραγματοποιούνται με απόλυτη συνέπεια στην πλευρά του διακομιστή καθώς δεν είναι ευάλωτοι σε HTML/JavaScript τροποποιήσεις από τον χρήστη. Συνεπώς, οι έλεγχοι στην πλευρά του διακομιστή είναι όχι μόνο πιο σύνθετοι αλλά και πιο αξιόπιστοι από τους αντίστοιχους στην πλευρά του πελάτη.

Η φόρμα Django μπορεί να εφαρμόσει έναν ή περισσότερους επικυρωτές τιμής σε καθένα από τα πεδία της, οι οποίοι ορίζονται στο αρχείο validators.py[68] και ο καθένας από αυτούς ελέγχει, ανεξάρτητα από τους άλλους, ότι η τιμή του εκάστοτε πεδίου, στο οποίο εφαρμόζεται, ικανοποιεί έναν συγκεκριμένο περιορισμό. Η φόρμα Django μπορεί επίσης να επιβάλει περιορισμούς τιμών ανά ομάδες (αλληλεξαρτώμενων) πεδίων μέσω της μεθόδου django.forms.Form.clean()· οι περιορισμοί αυτοί ελέγχονται πάντα μετά την ολοκλήρωση των ελέγχων για τους περιορισμούς τιμής ανά πεδίο και ανεξάρτητα από τα αποτελέσματα των τελευταίων. Αν παραβιαστεί κάποιος περιορισμός σε ένα πεδίο ή μία ομάδα πεδίων, τότε ο έλεγχός του/της αποτυγχάνει· αν αποτύχει τουλάχιστον ένας έλεγχος πεδίου ή ομάδας πεδίων, τότε ολόκληρη η φόρμα Django θεωρείται άκυρη. Σε αυτή την περίπτωση, η όψη δεν επιτελεί τη λειτουργία της, αλλά επιστρέφει μέσω του διακομιστή το αντίστοιχο πρότυπο HTML, το οποίο περιέχει ένα μήνυμα σφάλματος επικύρωσης (django.core.exceptions.ValidationError message) για καθέναν από τους αποτυχημένους ελέγχους πεδίων ή/και ομάδων πεδίων της φόρμας Django, ως απόκριση στον περιηγητή.

Αν οι έλεγχοι για όλα τα πεδία και όλες τις ομάδες πεδίων επιτύχουν, τότε η φόρμα Django θεωρείται έγκυρη και οι τιμές των πεδίων της θα αποτελέσουν δεδομένα εισόδου για τις κύριες ενέργειες της όψης. Μεταξύ άλλων, η όψη θα αξιοποιήσει τα δεδομένα εισόδου για να συναλλαχθεί με τη βάση usermergeDB και να δημιουργήσει, να διαβάσει, να ενημερώσει ή/και να διαγράψει τις κατάλληλες εγγραφές των πινάκων της. Αν η συναλλαγή αποτύχει λόγω μίας ή περισσότερων ασυμφωνιών (ασυμβατοτήτων) μεταξύ των δεδομένων εισόδου και της κατάστασης (στιγμιοτύπου) της βάσης, τότε η όψη δεν θα επιτελέσει τη λειτουργία της, αλλά θα επιστρέψει μέσω του διακομιστή το αντίστοιχο πρότυπο HTML, το οποίο θα περιέχει ένα μήνυμα σφάλματος μετεπικύρωσης (post-validation error message), συμπεριλαμβανομένων των σχετικών δεδομένων εισόδου (και του σχετικού κωδικού σφάλματος - error code - που περιλαμβάνεται στο πλαίσιο του προτύπου/σελίδας - template/page context - αλλά

συνήθως δεν εμφανίζεται κατά την προβολή αυτού/αυτής), για καθεμία από τις προαναφερθείσες ασυμφωνίες, ως απόκριση στον περιηγητή. Αντίθετα, αν η συναλλαγή επιτύχει, τότε η όψη θα επιτελέσει τη λειτουργία της και θα επιστρέψει μέσω του διακομιστή το αντίστοιχο πρότυπο ιστού, π.χ. HTML ή CSV, το οποίο θα περιέχει είτε το σχετικό μήνυμα επιτυχίας, συμπεριλαμβανομένων των δεδομένων εισόδου, είτε τα δεδομένα εισόδου μαζί με τις σχετικές πληροφορίες που αντλήθηκαν επιτυχώς από τη βάση usermergeDB στα πλαίσια της συγκεκριμένης λειτουργίας, ως απόκριση στον περιηγητή.

Αν η όψη δεν επιτελέσει τη λειτουργία της λόγω ενός ή περισσότερων από τα προαναφερθέντα σφάλματα, ο χρήστης μπορεί να επαναλάβει το παρόν βήμα από την αρχή συμπληρώνοντας/ενημερώνοντας ξανά τη φόρμα στην αντίστοιχη ιστοσελίδα και υποβάλλοντάς την, όπως περιγράφεται στο προηγούμενο βήμα, αφού πρώτα λάβει υπόψιν τα σχετικά μηνύματα σφαλμάτων. Ο χρήστης μπορεί γενικά να επαναλάβει το παρόν βήμα, και κατ' επέκταση και το προηγούμενο από αυτό, όσες φορές θέλει, ακόμα και αν η όψη επιτελέσει επιτυχώς τη λειτουργία της.

4.3 Όψεις λειτουργιών και πρότυπα ιστού που απευθύνονται σε όλους τους χρήστες του SLUB

4.3.1 Είσοδος στο SLUB

Η λειτουργία εισόδου στο SLUB απευθύνεται σε όλους τους χρήστες του SLUB και η ροή της βασίζεται σε εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας:

- Συμπληρωματική όψη display_default_login_page():
  - URL: http://slub.softlab.ntua.gr/ (σελίδα ευρετηρίου - index page - του ιστοτόπου της εφαρμογής)
  - Κύριο σώμα: Έλεγξε αν ο τρέχων χρήστης έχει εισέλθει. Αν όχι (το μοντέλο του είναι AnonymousUser), εμφάνισε τη σελίδα εισόδου, login.html, στην προεπιλεγμένη μορφή της (βλ. Σχήμα 4.6), δηλαδή χωρίς μηνύματα που σχετίζονται με την υποβολή της φόρμας. Διαφορετικά, έλεγξε αν το μοντέλο του χρήστη είναι auth.User. Αν ναι (ο χρήστης έχει εισέλθει στον ιστότοπο διαχείρισης Django), εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (ο χρήστης απαγορεύεται να προσπελάσει οποιαδήποτε σελίδα/πρότυπο της εφαρμογής). Διαφορετικά (ο χρήστης έχει εισέλθει στον ιστότοπο της εφαρμογής και το μοντέλο του είναι usermerge.{User/Admin}), εμφάνισε τη σελίδα συνέχισης συνεδρίας, session_continuation.html (βλ. Σχήμα 4.7), η οποία τον ενημερώνει ότι η συνεδρία του είναι ακόμα ενεργή και του παρέχει τις επιλογές είτε να μεταβεί στην αρχική σελίδα που αντιστοιχεί στο μοντέλο του (αν επιθυμεί να συνεχίσει την πλοήγησή του - navigation - σε αυτή τη συνεδρία) είτε να εξέλθει από τον ιστότοπο της εφαρμογής (αν επιθυμεί να εκκινήσει μία νέα συνεδρία ως άλλος χρήστης).

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη:

- Περιορισμοί για τις τιμές των πεδίων της HTML φόρμας login_form (για παράδειγμα σφάλματος υποβολής, βλ. Σχήμα 4.8):
  - Πλατφόρμα εισόδου:
    - Απαίτηση συμπλήρωσης
  - Όνομα χρήστη:
    - Απαίτηση συμπλήρωσης
    - Ελάχιστο μήκος: 8 χαρακτήρες
    - Μέγιστο μήκος: 50 χαρακτήρες
  - Κωδικός πρόσβασης:
    - Απαίτηση συμπλήρωσης
    - Ελάχιστο μήκος: 8 χαρακτήρες
    - Μέγιστο μήκος: 50 χαρακτήρες
- Λοιπές παρατηρήσεις:
  - Οι διαθέσιμες τιμές προς επιλογή στην αναπτυσσόμενη λίστα (drop-down list) του πεδίου της πλατφόρμας εισόδου είναι η "(διαχειριστής του) SLUB" και εκείνες που έχουν αποθηκευτεί στον πίνακα usermerge_platform της βάσης δεδομένων, δηλαδή οι "(χρήστης του) Novice", "(χρήστης του) Grader", "(χρήστης του) Moodle" και "(χρήστης του) PLgrader".

(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας:

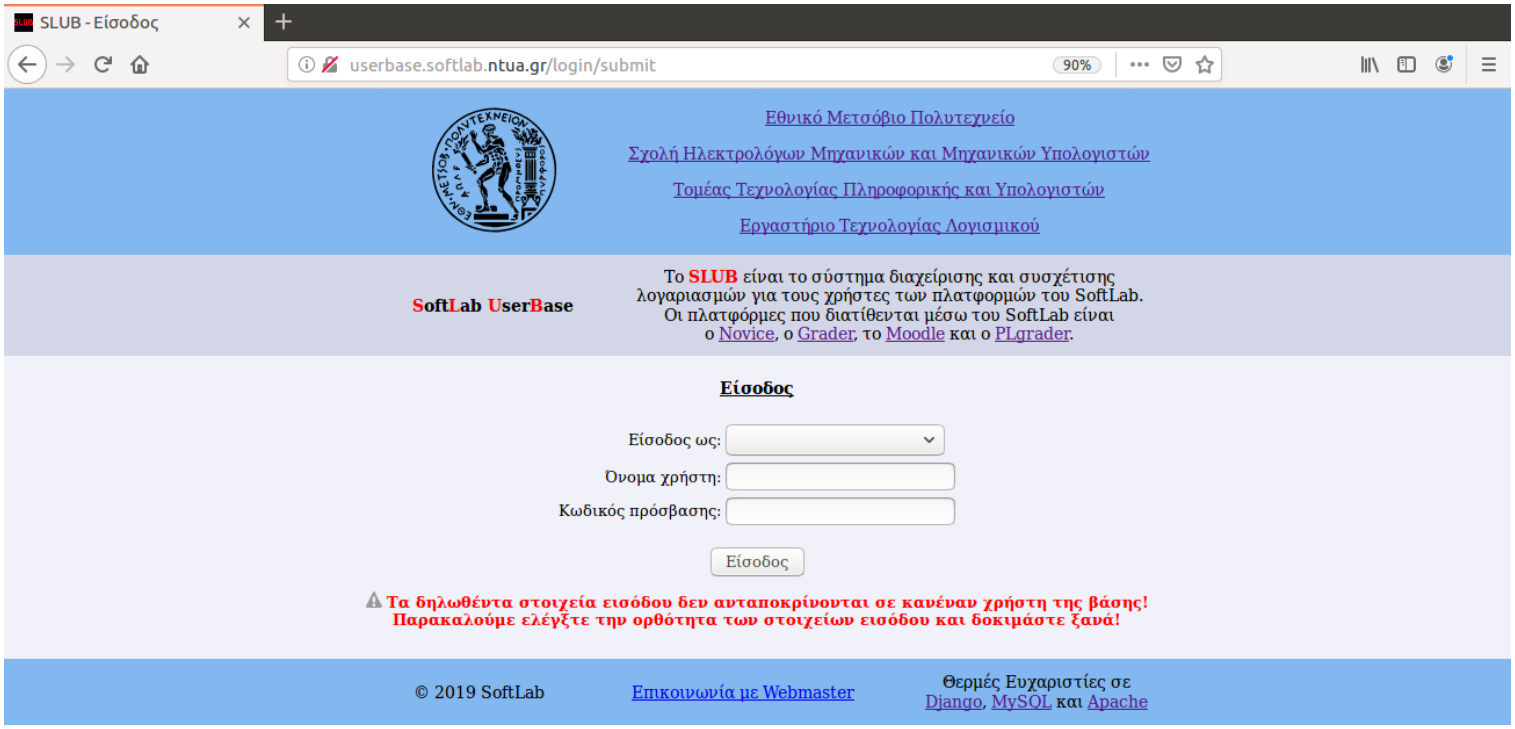
- Κύρια όψη log_in():
  - URL: http://slub.softlab.ntua.gr/login/submit
  - Κύριο σώμα: Έλεγξε αν ο τρέχων χρήστης έχει εισέλθει. Αν ναι (ο χρήστης έχει εισέλθει είτε στον ιστότοπο διαχείρισης Django ως ένα στιγμιότυπο auth.User είτε στον ιστότοπο της εφαρμογής ως ένα στιγμιότυπο usermerge.{User/Admin}), εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (ο χρήστης έχει εισέλθει είτε στον ιστότοπο διαχείρισης Django και, ως εκ τούτου, απαγορεύεται να προσπελάσει οποιαδήποτε όψη της εφαρμογής είτε στον ιστότοπο της εφαρμογής και, ως εκ τούτου, απαγορεύεται να ξαναεπιχειρήσει είσοδο στον τελευταίο). Διαφορετικά (ο χρήστης είναι ένα στιγμιότυπο AnonymousUser), σύλλεξε τα δεδομένα, δηλαδή την πλατφόρμα εισόδου, το όνομα χρήστη και τον κωδικό πρόσβασης, που υποβλήθηκαν μέσω της φόρμας εισόδου και επικύρωσέ τα. Αν δεν είναι έγκυρα, εμφάνισε τα κατάλληλα μηνύματα σφαλμάτων επικύρωσης στη σελίδα εισόδου. Διαφορετικά, αυθεντικοποίησε τα διαπιστευτήρια (όνομα χρήστη και κωδικό πρόσβασης) για την επιλεγμένη πλατφόρμα εισόδου. Αν αυτά δεν προσδιορίσουν κανένα στιγμιότυπο usermerge.{User/Admin} στη βάση δεδομένων, εμφάνισε το κατάλληλο μήνυμα σφάλματος μετεπικύρωσης στη σελίδα εισόδου και συμπερίλαβε τον αντίστοιχο κωδικό σφάλματος, "non_existent_credentials_pair_for_selected_login_platform", στο πλαίσιο της σελίδας (βλ. Σχήμα 4.9). Διαφορετικά, εισάγαγε τον τρέχοντα χρήστη στον ιστότοπο της εφαρμογής ως το προσδιορισμένο (αυθεντικοποιημένο) στιγμιότυπο usermerge.{User/Admin} και ανακατεύθυνέ τον στην αρχική σελίδα που αντιστοιχεί στο μοντέλο του.

- Περιορισμοί για τις τιμές των πεδίων της Django φόρμας LoginForm - οι περιορισμοί για τις τιμές όλων των πεδίων ελέγχονται μετά την αφαίρεση (stripping) των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων τους (leading and trailing whitespace characters):
  - Πλατφόρμα εισόδου:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Επιλογή τιμής μεταξύ εκείνων που διατίθενται στην αντίστοιχη αναπτυσσόμενη λίστα (σφάλμα "platform_is_not_SLUB_and_does_not_exist_in_DB")
  - Όνομα χρήστη:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Ελάχιστο μήκος (σφάλμα "min_length"): 8 χαρακτήρες
    - Μέγιστο μήκος (σφάλμα "max_length"): 50 χαρακτήρες
  - Κωδικός πρόσβασης:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Ελάχιστο μήκος (σφάλμα "min_length"): 8 χαρακτήρες
    - Μέγιστο μήκος (σφάλμα "max_length"): 50 χαρακτήρες

Σχήμα 4.6: Σελίδα εισόδου του SLUB στην προεπιλεγμένη μορφή της

Σχήμα 4.7: Σελίδα συνέχισης συνεδρίας του SLUB

Σχήμα 4.8: Εμφάνιση ενός μηνύματος σφάλματος υποβολής στη σελίδα εισόδου του SLUB



Σχήμα 4.9: Εμφάνιση του μηνύματος σφάλματος μετεπικύρωσης στη σελίδα εισόδου του SLUB

4.3.2 Έξοδος από το SLUB

Η λειτουργία εξόδου από το SLUB απευθύνεται σε όλους τους χρήστες του SLUB, προϋποθέτει την προγενέστερη επιτέλεση της λειτουργίας εισόδου στο SLUB και έχει το ακριβώς αντίθετο αποτέλεσμα από την τελευταία. Η λειτουργία εξόδου δεν σχετίζεται με κάποια συγκεκριμένη σελίδα ή φόρμα HTML· μέσω του υπερσυνδέσμου «Έξοδος» στην μπάρα συνεδρίας (session bar), μπορεί μάλιστα να επιτελεστεί ανά πάσα στιγμή από οποιαδήποτε σελίδα του SLUB (πρότυπο-απόκριση με κατάσταση "200 OK") στα πλαίσια μίας συνεδρίας. Ως εκ τούτου, η ροή της, η οποία διαφοροποιείται αρκετά από εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2, σχηματοποιείται ως εξής:

- Κύρια όψη log_out():
  - URL: http://slub.softlab.ntua.gr/logout
  - Διακοσμητές:
    - login_required(): Το μοντέλο του χρήστη της συνεδρίας μπορεί να είναι είτε auth.User είτε usermerge.{User/Admin}.
  - Κύριο σώμα: Έλεγξε αν το μοντέλο του χρήστη της συνεδρίας είναι auth.User. Αν ναι (ο χρήστης έχει εισέλθει στον ιστότοπο διαχείρισης Django), εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (ο χρήστης απαγορεύεται να προσπελάσει οποιαδήποτε όψη της εφαρμογής). Διαφορετικά (ο χρήστης έχει εισέλθει στον ιστότοπο της εφαρμογής και το μοντέλο του είναι usermerge.{User/Admin}), αποσύνδεσέ τον (εκπέμποντας - emitting - ταυτόχρονα το σήμα django.contrib.auth.signals.user_logged_out μέσω της συνάρτησης django.contrib.auth.logout()) και ανακατεύθυνέ τον στην προεπιλεγμένη σελίδα εισόδου.

- Λοιπές παρατηρήσεις:
  - Εκτός από τον υπερσύνδεσμο «Έξοδος», η μπάρα συνεδρίας περιλαμβάνει επίσης την κατάσταση συνεδρίας (session status), «Έχετε εισέλθει επιτυχώς!» / «Έχετε εισέλθει ως {ΟΝΟΜΑ και ΕΠΩΝΥΜΟ που έχουν συμπληρωθεί στο προφίλ του χρήστη}!», και τον υπερσύνδεσμο «Αρχική Σελίδα», ενώ εμφανίζεται σε όλες τις σελίδες του SLUB εκτός από τη σελίδα εισόδου και τις σελίδες σφαλμάτων HTTP.

4.4 Όψεις λειτουργιών και πρότυπα ιστού που απευθύνονται στους διαχειριστές του SLUB

4.4.1 Εμφάνιση αρχικής σελίδας διαχειριστή

Το SLUB παρέχει σε κάθε διαχειριστή του τις λειτουργίες προσθήκης διαπιστευτηρίων και δημιουργίας αναφορών για απλούς χρήστες, οι οποίες μπορούν να προσπελαστούν μέσω των αντίστοιχων υπερσυνδέσμων στην αρχική σελίδα διαχειριστή, admin_home.html (βλ. Σχήμα 4.10). Η λειτουργία εμφάνισης αυτής της σελίδας απευθύνεται στους διαχειριστές του SLUB και προϋποθέτει την προγενέστερη επιτέλεση της λειτουργίας εισόδου στο SLUB. Για αυτόν τον λόγο, επιτελείται μέσω της συμπληρωματικής όψης display_admin_home_page() που συσχετίζεται με το URL http://slub.softlab.ntua.gr/admin/home και περιτυλίσσεται από τον διακοσμητή Admin_login_required(). Μέσω του υπερσυνδέσμου «Αρχική Σελίδα» στην μπάρα συνεδρίας, η λειτουργία εμφάνισης της αρχικής σελίδας διαχειριστή μπορεί μάλιστα να επιτελεστεί ανά πάσα στιγμή από οποιαδήποτε σελίδα του SLUB (πρότυπο-απόκριση με κατάσταση "200 OK") στα πλαίσια μίας συνεδρίας διαχειριστή.

Σχήμα 4.10: Αρχική σελίδα διαχειριστή του SLUB

4.4.2 Προσθήκη διαπιστευτηρίων χρηστών

Η λειτουργία προσθήκης διαπιστευτηρίων (απλών) χρηστών απευθύνεται στους διαχειριστές του SLUB και η ροή της βασίζεται σε εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας:

- Συμπληρωματική όψη display_default_user_credentials_import_page():
  - URL: http://slub.softlab.ntua.gr/admin/user-credentials-import/default
  - Διακοσμητές:
    - Admin_login_required()
  - Κύριο σώμα: Εμφάνισε τη σελίδα προσθήκης διαπιστευτηρίων χρηστών, user_credentials_import.html, στην προεπιλεγμένη μορφή της (βλ. Σχήμα 4.11), δηλαδή χωρίς μηνύματα που σχετίζονται με την υποβολή της φόρμας.

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη:

- Περιορισμοί για τις τιμές των πεδίων της HTML φόρμας user_credentials_import_form (για παράδειγμα σφάλματος υποβολής, βλ. Σχήμα 4.12):
  - Πλατφόρμα διαπίστευσης (accreditation platform):
    - Απαίτηση συμπλήρωσης
  - Αρχείο διαπιστευτηρίων:
    - Απαίτηση συμπλήρωσης
    - Τύπος μέσου/περιεχομένου (media/content type) ή τύπος MIME (Multipurpose Internet Mail Extensions): text/plain (π.χ. επέκταση ονόματος αρχείου - filename extension - ".txt")
- Λοιπές παρατηρήσεις:
  - Οι διαθέσιμες τιμές προς επιλογή στην αναπτυσσόμενη λίστα (drop-down list) του πεδίου της πλατφόρμας διαπίστευσης είναι εκείνες που έχουν αποθηκευτεί στον πίνακα usermerge_platform της βάσης δεδομένων, δηλαδή οι "Novice", "Grader", "Moodle" και "PLgrader".
  - Δεδομένου ότι η μέθοδος υποβολής των δεδομένων της φόρμας (χαρακτηριστικό method του HTML στοιχείου <form>) είναι "post", η επιφόρτωση (upload) του αρχείου διαπιστευτηρίων στον διακομιστή επιτυγχάνει μόνο όταν ο τύπος κωδικοποίησης των υποβαλλόμενων δεδομένων (χαρακτηριστικό enctype του HTML στοιχείου <form>) είναι "multipart/form-data" (κανένας χαρακτήρας δεν κωδικοποιείται)[69].

(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας:

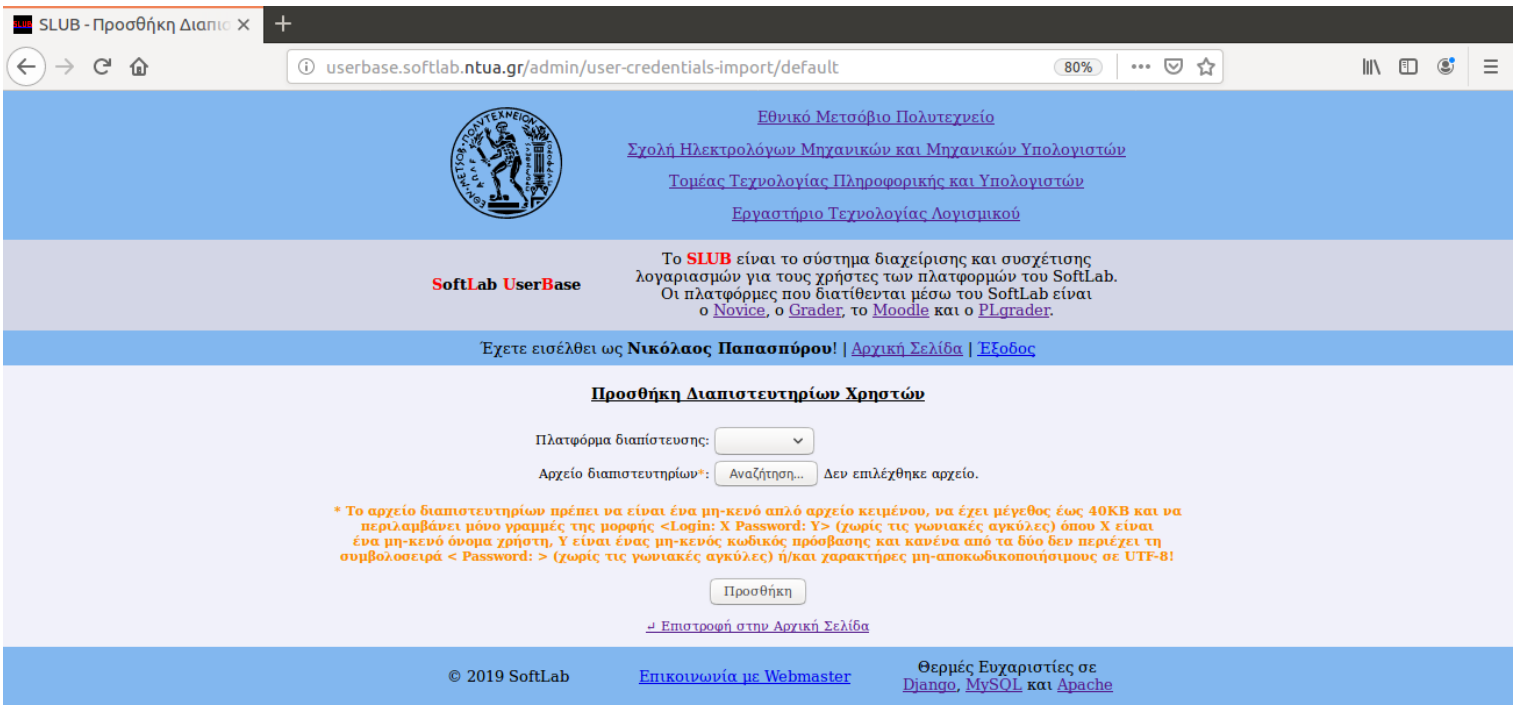
- Κύρια όψη import_user_credentials():
  - URL: http://slub.softlab.ntua.gr/admin/user-credentials-import/submit
  - Διακοσμητές:
    - Admin_login_required()
    - transaction.atomic()
  - Κύριο σώμα: Σύλλεξε τα δεδομένα, δηλαδή την πλατφόρμα διαπίστευσης και το αρχείο διαπιστευτηρίων (για παράδειγμα αρχείου διαπιστευτηρίων με τύπο TXT, μέγεθος 2KB και κωδικοποίηση UTF-8, βλ. Σχήμα 4.13),

που υποβλήθηκαν μέσω της φόρμας προσθήκης διαπιστευτηρίων χρηστών και επικύρωσέ τα. Αν δεν είναι έγκυρα, εμφάνισε τα κατάλληλα μηνύματα σφαλμάτων επικύρωσης στη σελίδα προσθήκης διαπιστευτηρίων χρηστών. Διαφορετικά, εξέτασε κάθε γραμμή του αρχείου διαπιστευτηρίων, ελέγχοντας πρώτον τη μορφή της γραμμής, δεύτερον το μήκος των διαπιστευτηρίων και τρίτον τη μοναδικότητα του σύνθετου κλειδιού (composite key) {username, platform_id} στη βάση δεδομένων. Αν κάποιος από τους ελέγχους αποτύχει, παράλειψε τους υπόλοιπους ελέγχους της εξεταζόμενης γραμμής, μάρκαρέ την ως εσφαλμένη για τον συγκεκριμένο έλεγχο και κατόπιν συνέχισε στην επόμενη γραμμή. Αν δεν αποτύχει κανένας από τους ελέγχους, μάρκαρε την εξεταζόμενη γραμμή ως σωστή, δημιούργησε ένα νέο κενό στιγμιότυπο User στη βάση δεδομένων, κωδικοποίησε (μέσω κατακερματισμού) τον κωδικό πρόσβασης της γραμμής, δημιούργησε ένα νέο στιγμιότυπο Registry στη βάση δεδομένων, το οποίο συσχετίζει το δημιουργηθέν στιγμιότυπο User με την πλατφόρμα διαπίστευσης, το όνομα χρήστη της γραμμής και τον κωδικοποιηθέντα κωδικό πρόσβασης, και μετά συνέχισε στην επόμενη γραμμή. Όταν εξεταστούν όλες οι γραμμές, εμφάνισε τα κατάλληλα μηνύματα επιτυχίας και σφαλμάτων μετεπικύρωσης στη σελίδα προσθήκης διαπιστευτηρίων χρηστών και συμπερίλαβε όλους τους αριθμούς (αναγνωριστικά) γραμμών, οι οποίοι έχουν ομαδοποιηθεί ανά αποτέλεσμα/μαρκάρισμα προσθήκης διαπιστευτηρίων (ομάδες "credentials_import_success_line_nos", "wrong_line_format_error_line_nos", "prohibited_credentials_length_error_line_nos" και "duplicate_platform_username_error_line_nos"), στο πλαίσιό της (για παράδειγμα, βλ. Σχήμα 4.14 που σχετίζεται με το αρχείο του Σχήματος 4.13).

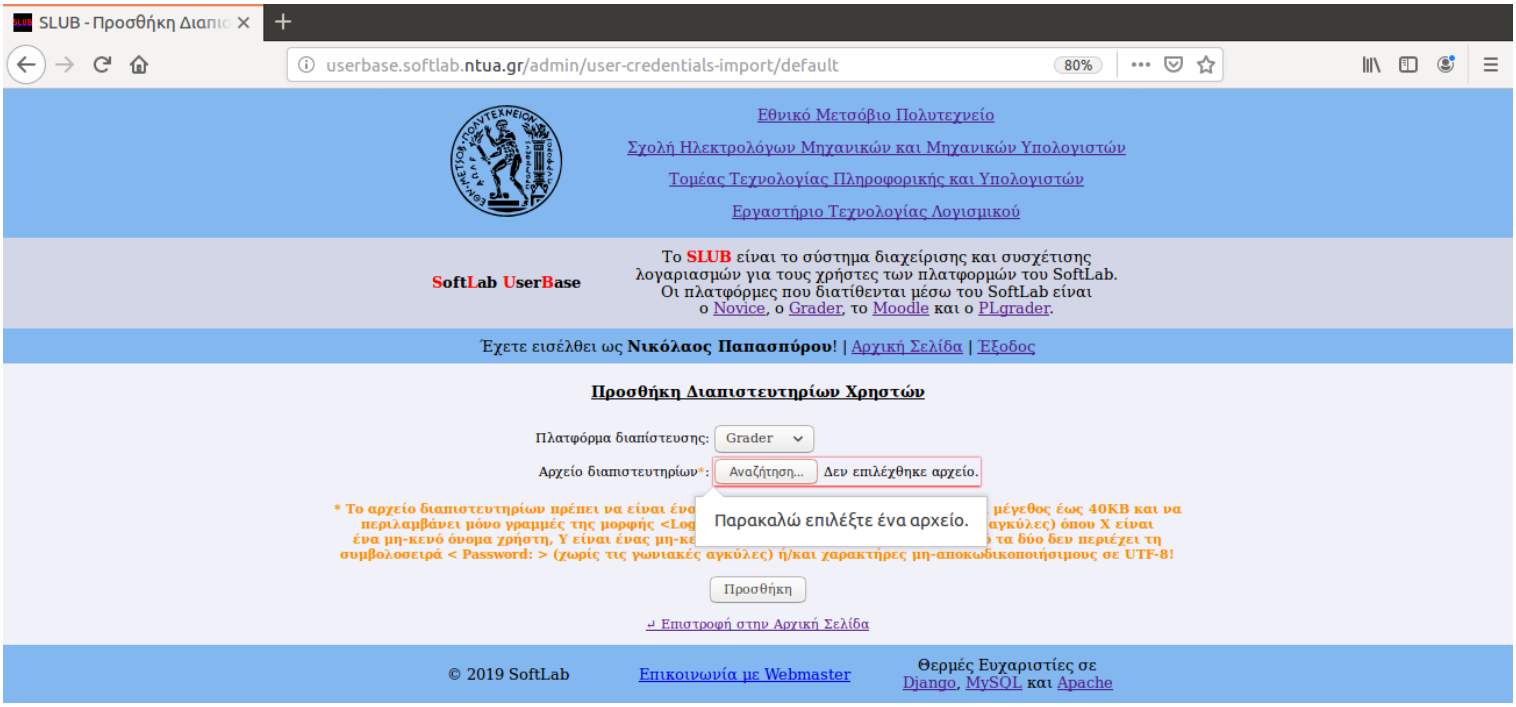
- Περιορισμοί για τις τιμές των πεδίων της Django φόρμας UserCredentialsImportForm:
  - Πλατφόρμα διαπίστευσης - οι περιορισμοί για την τιμή αυτού του πεδίου ελέγχονται μετά την αφαίρεση (stripping) των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων της (leading and trailing whitespace characters):
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Επιλογή τιμής μεταξύ εκείνων που διατίθενται στην αντίστοιχη αναπτυσσόμενη λίστα (σφάλμα "platform_does_not_exist_in_DB")
  - Αρχείο διαπιστευτηρίων:
    - Απαίτηση συμπλήρωσης και μη-κενού ονόματος (σφάλμα "required")
    - Τύπος κωδικοποίησης στην αντίστοιχη φόρμα (σφάλμα "invalid"): multipart/form-data
    - Ολοκληρωμένη παρουσία στο σώμα του αντίστοιχου αιτήματος HTTP POST που αποστέλλεται επιτυχώς στον διακομιστή, δηλαδή δεν συνοδεύεται από κατάρρευση περιηγητή ιστού (web browser crash), σφάλμα σύνδεσης δικτύου (network connection error) ή/και άλλο αστάθμιστο παράγοντα (σφάλμα "missing")
    - Μέγιστο μήκος ονόματος (σφάλμα "max_length"): 50 χαρακτήρες
    - Μη-κενό περιεχόμενο (σφάλμα "empty")

- Τύπος μέσου/περιεχομένου ή τύπος MIME (σφάλμα "credentials_file_content_type_is_not_plain_text"): text/plain (π.χ. επέκταση ονόματος αρχείου ".txt")
- Μέγιστο μέγεθος (σφάλμα "credentials_file_size_exceeds_40KB"): 40KB

- Λοιπές παρατηρήσεις:
  - Το αρχείο διαπιστευτηρίων πρέπει να είναι ένα μη-κενό απλό αρχείο κειμένου, να έχει μέγεθος έως 40KB (KiloBytes) και να περιλαμβάνει μόνο γραμμές της μορφής <Login: X Password: Y> (χωρίς τις γωνιακές αγκύλες) όπου X είναι ένα μη-κενό όνομα χρήστη, Y είναι ένας μη-κενός κωδικός πρόσβασης και κανένα από τα δύο δεν περιέχει τη συμβολοσειρά < Password: > (χωρίς τις γωνιακές αγκύλες) ή/και χαρακτήρες μη-αποκωδικοποιήσιμους (non-decodable) σε UTF-8.
  - Σε κάθε γραμμή του αρχείου διαπιστευτηρίων ελέγχεται ότι τα διαπιστευτήρια, δηλαδή το όνομα χρήστη και ο κωδικός πρόσβασης, έχουν μήκος τουλάχιστον 8 και το πολύ 50 χαρακτήρων έκαστο μετά την αφαίρεση των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων τους.



Σχήμα 4.11: Σελίδα προσθήκης διαπιστευτηρίων χρηστών του SLUB στην προεπιλεγμένη μορφή της

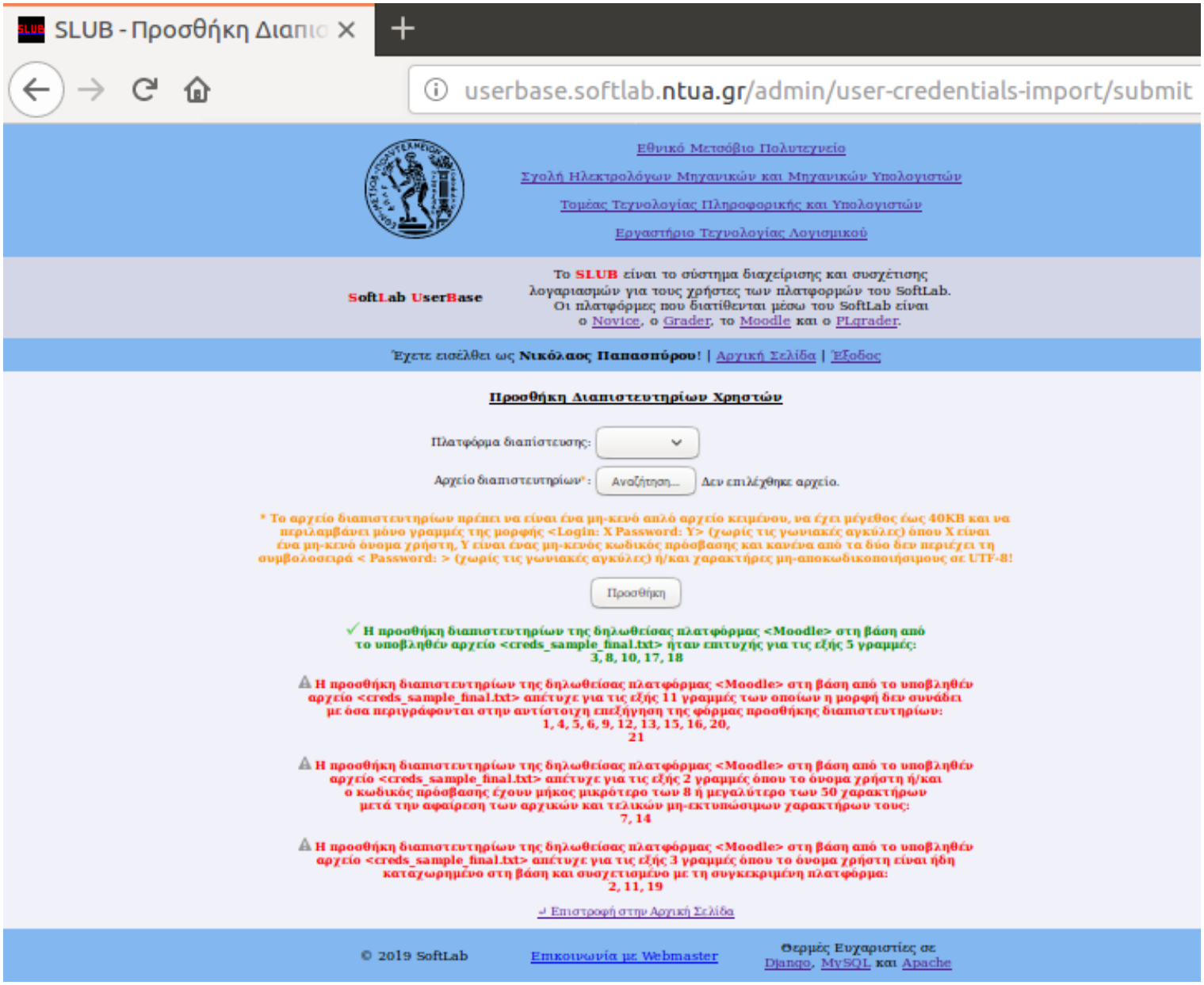


Σχήμα 4.12: Εμφάνιση ενός μηνύματος σφάλματος υποβολής στη σελίδα προσθήκης διαπιστευτηρίων χρηστών του SLUB

```
Login: NOpassword
Login: an_already_submitted_platform_login Password: PassTheWord!
Login:          leading_and_trailing_whitespace          Password:    leading_and_trailing_whitespace
Login:  Password: empty_login
Login: NOintermediateSpacePassword: The next line will be a blank one!

Login:   <8Chars                    Password: Scrooge$MacDuck
Login: 日本語が難しい! Password: ΤαΙαπωνικάΕίναιΔύσκολα!
Login: login#09 Password: extra_separator Password: password#09
Login: LineFeed:\n, CarriageReturn:\r, HorizontalTab:\t Password: escape_sequence_check
Login: LineFeed:\n, CarriageReturn:\r, HorizontalTab:\t Password: SeeAboveLine!
Login: nextCharIsTabNotSpace    Password without_colon
 <Login: space&angle_bracket@line_start Password:NOin_betweenSpace
Login: hello2world Password:   length(0123456789* 0123456789% 0123456789- 0123456789? 0123456789)>50Chars
Login: before_newline
after_newline Password: PassTheWord!
Login: Both the Login: andPassword: have >8 and <50 chars Password: Huey        [TAB]Dewey      {TAB}Louie
Login: 0Donald^Duck Password: Huey      [TAB]Dewey      {TAB}Louie
Login: another_already_submitted_platform_login Password: Aloha~Mundo
Login: credentials_in_2_lines Pass
word: Pass_newline_word:
```

Σχήμα 4.13: Ενδεικτικό αρχείο διαπιστευτηρίων που υποβλήθηκε μέσω της φόρμας προσθήκης διαπιστευτηρίων χρηστών του SLUB



Σχήμα 4.14: Εμφάνιση μηνυμάτων επιτυχίας και σφαλμάτων μετεπικύρωσης στη σελίδα προσθήκης διαπιστευτηρίων χρηστών του SLUB

4.4.3 Δημιουργία αναφοράς χρηστών

Η λειτουργία δημιουργίας αναφοράς (απλών) χρηστών απευθύνεται στους διαχειριστές του SLUB και η ροή της βασίζεται σε εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας:

- Συμπληρωματική όψη display_default_user_report_creation_page():
  - URL: http://slub.softlab.ntua.gr/admin/user-report-creation/default
  - Διακοσμητές:
    - Admin_login_required()
  - Κύριο σώμα: Εμφάνισε τη σελίδα δημιουργίας αναφοράς χρηστών, user_report_creation.html, στην προεπιλεγμένη μορφή της (βλ. Σχήμα

4.15), δηλαδή χωρίς μηνύματα που σχετίζονται με την υποβολή της φόρμας.

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη:

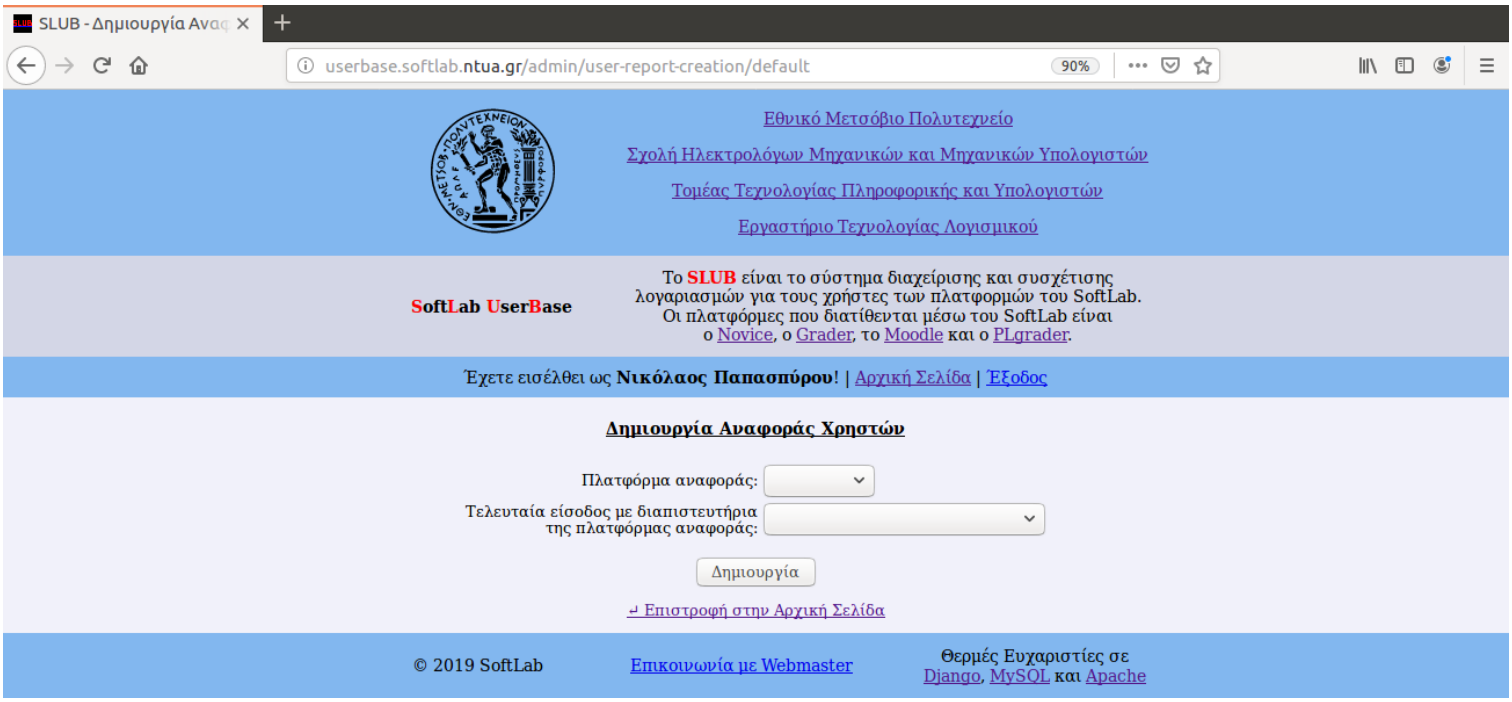
- Περιορισμοί για τις τιμές των πεδίων της HTML φόρμας user_report_creation_form (για παράδειγμα σφάλματος υποβολής, βλ. Σχήμα 4.16):
  - Πλατφόρμα αναφοράς (reference platform):
    - Απαίτηση συμπλήρωσης
  - Τελευταία είσοδος με διαπιστευτήρια της πλατφόρμας αναφοράς:
    - Απαίτηση συμπλήρωσης
- Λοιπές παρατηρήσεις:
  - Οι διαθέσιμες τιμές προς επιλογή στην αναπτυσσόμενη λίστα (drop-down list) του πεδίου της πλατφόρμας αναφοράς είναι εκείνες που έχουν αποθηκευτεί στον πίνακα usermerge_platform της βάσης δεδομένων, δηλαδή οι "Novice", "Grader", "Moodle" και "PLgrader".
  - Οι διαθέσιμες τιμές προς επιλογή στην αναπτυσσόμενη λίστα του πεδίου της τελευταίας εισόδου (με διαπιστευτήρια της πλατφόρμας αναφοράς) είναι οι "εντός του τελευταίου εξαμήνου" ("last_six_months"), "εντός του τελευταίου έτους" ("last_year") και "από την αρχή του χρόνου" ("all_time").

(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας:

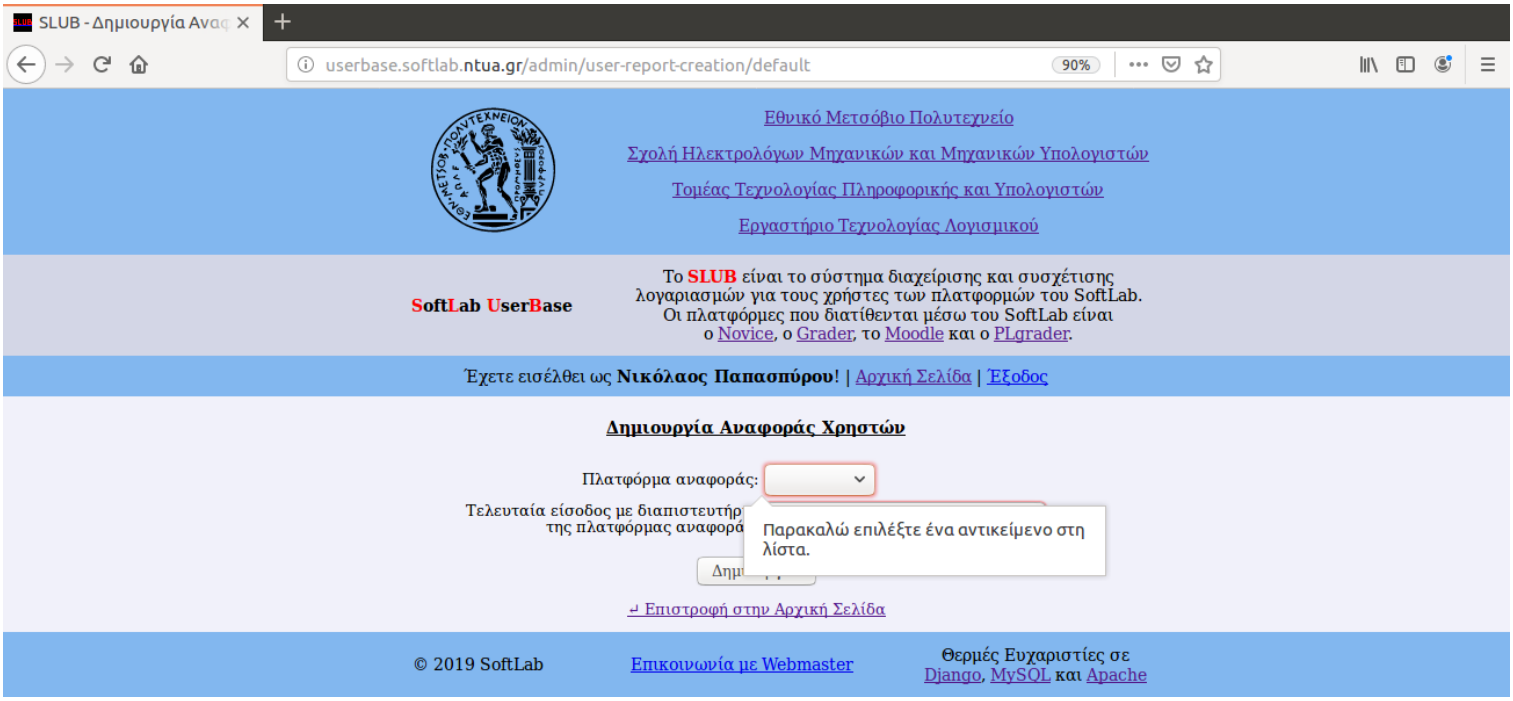
- Κύρια όψη create_user_report():
  - URL: http://slub.softlab.ntua.gr/admin/user-report-creation/submit
  - Διακοσμητές:
    - Admin_login_required()
  - Κύριο σώμα: Σύλλεξε τα δεδομένα, δηλαδή την πλατφόρμα αναφοράς και την τελευταία είσοδο (με διαπιστευτήρια της πλατφόρμας αναφοράς), που υποβλήθηκαν μέσω της φόρμας δημιουργίας αναφοράς χρηστών και επικύρωσέ τα. Αν δεν είναι έγκυρα, εμφάνισε τα κατάλληλα μηνύματα σφαλμάτων επικύρωσης στη σελίδα δημιουργίας αναφοράς χρηστών. Διαφορετικά, χρησιμοποίησέ τα για να επερωτήσεις (query) τη βάση δεδομένων για τα ζητούμενα στιγμιότυπα Registry και απαίτησε αυτά να συσχετίζονται με μη-κενά στιγμιότυπα User (προφίλ χρηστών) που διαθέτουν ουσιαστικές πληροφορίες για αναφορά (τα κενά στιγμιότυπα User δεν περιέχουν καμία απολύτως πληροφορία και, κατά συνέπεια, παραλείπονται μαζί με τα στιγμιότυπα Registry που τους αντιστοιχούν). Αν δεν ανακτηθεί κανένα τέτοιο στιγμιότυπο Registry από τη βάση δεδομένων, εμφάνισε το κατάλληλο μήνυμα σφάλματος μετεπικύρωσης στη σελίδα δημιουργίας αναφοράς χρηστών και συμπερίλαβε τον αντίστοιχο κωδικό σφάλματος, "unsatisfied_report_creation_criteria", στο πλαίσιο της σελίδας (βλ. Σχήμα 4.17). Διαφορετικά, δημιούργησε τη CSV (Comma-Separated Values) αναφορά με το κατάλληλο όνομα - π.χ. SLUB_user_report_Moodle_all_time_12072019_155955.csv, όπου τα κριτήρια δημιουργίας αναφοράς {πλατφόρμα αναφοράς, τελευταία είσοδος} είναι {"Moodle", "από την αρχή του χρόνου"} και η ημερομηνία δημιουργίας, η οποία (όπως και κάθε άλλη ημερομηνία που εμφανίζεται

στην αναφορά) έχει αναχθεί από το πρότυπο UTC (Coordinated Universal Time standard) στη ζώνη ώρας "Europe/Athens", είναι "12/07/2019 15:59:55" - και περιεχόμενο, δηλαδή τις λεπτομέρειες δημιουργίας καθώς και τις τιμές και μετρικές των πεδίων (field metrics - συνολικός αριθμός αναφερθέντων χρηστών και αριθμός αναφερθέντων χρηστών με ακαταχώρητο αριθμό μητρώου) που αντλήθηκαν από τα ανακτηθέντα στιγμιότυπα Registry και τα συσχετισμένα με αυτά στιγμιότυπα User (για παράδειγμα αναφοράς χρηστών με τις προαναφερθείσες τιμές κριτηρίων, βλ. Σχήμα 4.18), και εξάγαγέ την (export) ως συνημμένο αρχείο (file attachment - για παράδειγμα, βλ. Σχήμα 4.19 που σχετίζεται με την αναφορά του Σχήματος 4.18).

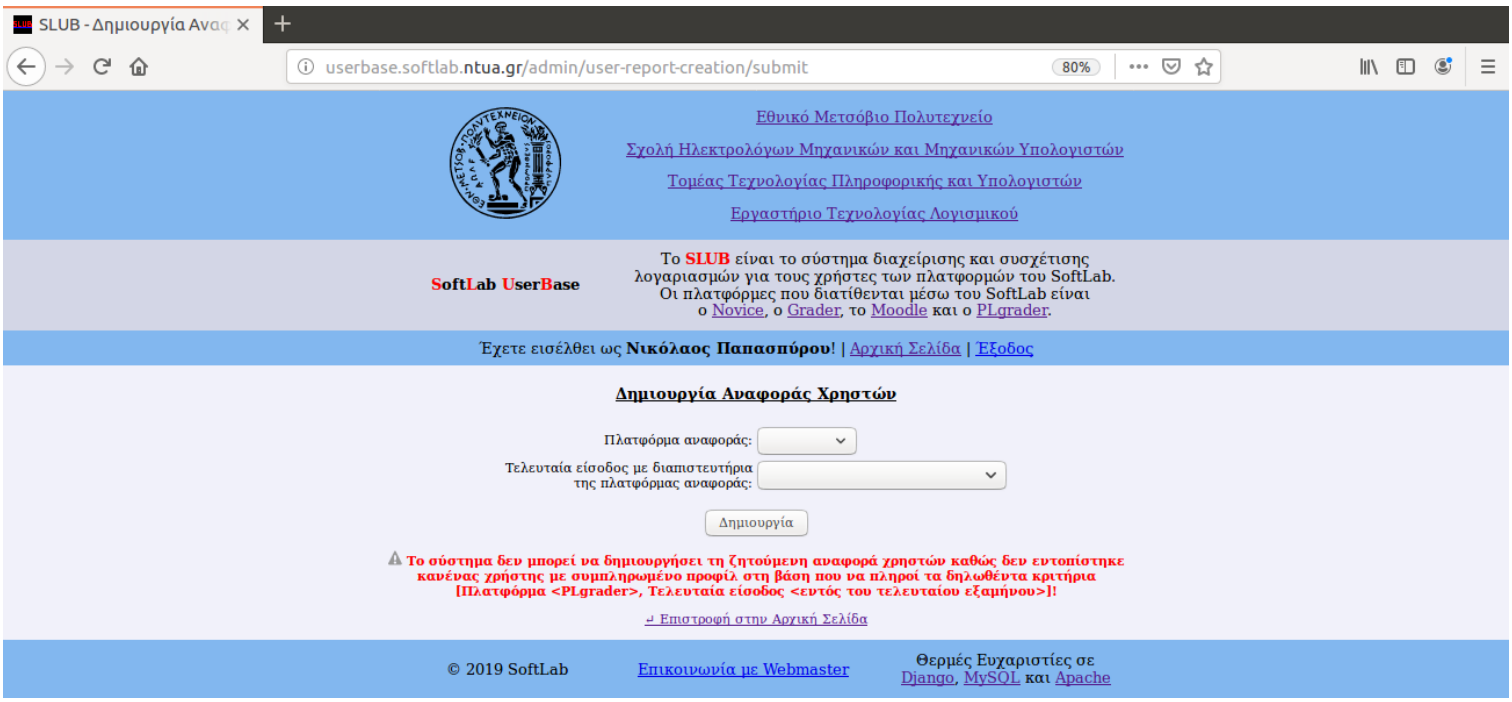
- Περιορισμοί για τις τιμές των πεδίων της Django φόρμας UserReportCreationForm - οι περιορισμοί για τις τιμές όλων των πεδίων ελέγχονται μετά την αφαίρεση (stripping) των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων τους (leading and trailing whitespace characters):
  - Πλατφόρμα αναφοράς:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Επιλογή τιμής μεταξύ εκείνων που διατίθενται στην αντίστοιχη αναπτυσσόμενη λίστα (σφάλμα "platform_does_not_exist_in_DB")
  - Τελευταία είσοδος με διαπιστευτήρια της πλατφόρμας αναφοράς:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Επιλογή τιμής μεταξύ εκείνων που διατίθενται στην αντίστοιχη αναπτυσσόμενη λίστα (σφάλμα "last_login_is_not_selected_from_provided_list")



Σχήμα 4.15: Σελίδα δημιουργίας αναφοράς χρηστών του SLUB στην προεπιλεγμένη μορφή της



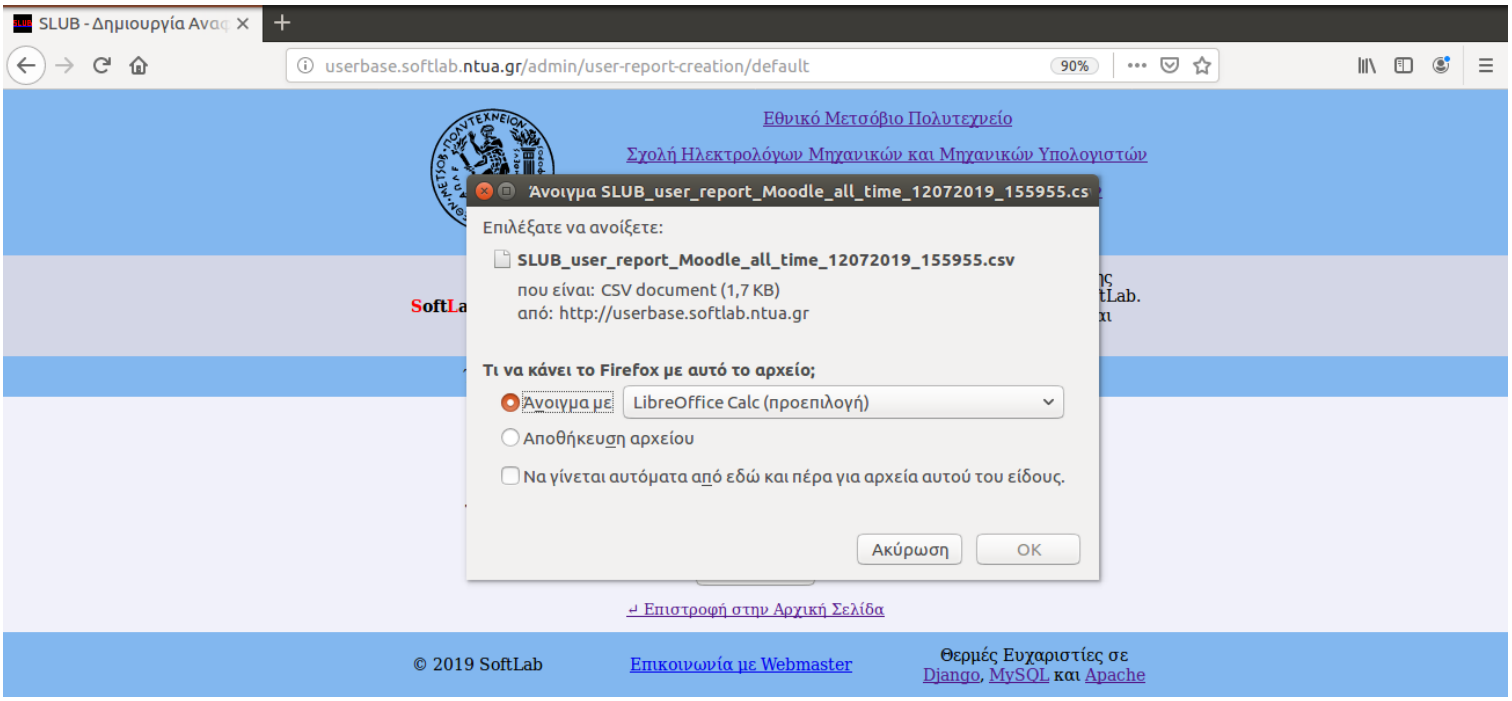
Σχήμα 4.16: Εμφάνιση ενός μηνύματος σφάλματος υποβολής στη σελίδα δημιουργίας αναφοράς χρηστών του SLUB



Σχήμα 4.17: Εμφάνιση του μηνύματος σφάλματος μετεπικύρωσης στη σελίδα δημιουργίας αναφοράς χρηστών του SLUB

	A	B	C	D	E	F	G
1	SoftLab UserBase:	Αναφορά Χρηστών					
2							
3	Δημιουργία από:	Νικόλαος Παπασπύρου <nikolaos@softlab.ntua.gr>					
4	Ημερομηνία δημιουργίας:	12/07/2019 16:45:38					
5							
6	Πλατφόρμα αναφοράς:	Moodle (αναγνωριστικό: 3)					
7	Τελευταία είσοδος με διαπιστευτήρια						
8	της πλατφόρμας αναφοράς:	από την αρχή του χρόνου					
9							
10							
11	Αναγνωριστικό χρήστη	Όνομα	Επώνυμο	Αριθμός μητρώου	Ηλεκτρονικό ταχυδρομείο	Τελευταία είσοδος	Όνομα χρήστη
12	5	Αντώνιος	Αντωνίου	03108951	aantoniou@highwind.com	05/10/2010 12:55:38	AntonAnton_5
13	8	Δήμητρα	Δημητρίου	03111963	ddimitriou@yaahoo.com	10/11/2011 14:45:20	DimiDimi_8
14	13	Κωνσταντίνος	Κωνσταντίνου	03114972	kkonsta@gogglemailz.com	15/12/2014 16:35:03	KonKon_13
15	21	Ιωάννης	Ιωάννου	[Δεν έχει καταχωρηθεί]	iioannou@thunderlook.com	20/03/2017 17:25:14	Ioanloan_21
16	34	Παναγιώτα	Παναγιώτου	03117984	ppanag@outmail.com	25/04/2018 19:15:59	PanagPanag_34
17	55	Αλεξία	Αλεξίου	[Δεν έχει καταχωρηθεί]	aalexiou@swiftbird.com	30/05/2019 21:05:47	AlexAlex_55
18							
19							
20	Συνολικός αριθμός αναφερθέντων χρηστών:	6					
21	Αριθμός αναφερθέντων χρηστών με						
22	ακαταχώρητο αριθμό μητρώου:	4					

Σχήμα 4.18: Ενδεικτική αναφορά χρηστών που δημιουργήθηκε μέσω της σελίδας δημιουργίας αναφοράς χρηστών του SLUB



Σχήμα 4.19: Εξαγωγή μίας αναφοράς χρηστών ως συνημμένο αρχείο στη σελίδα δημιουργίας αναφοράς χρηστών του SLUB

4.5 Όψεις λειτουργιών και πρότυπα ιστού που απευθύνονται στους απλούς χρήστες του SLUB

4.5.1 Εμφάνιση αρχικής σελίδας απλού χρήστη

Το SLUB παρέχει σε κάθε απλό χρήστη του τις λειτουργίες επεξεργασίας, αναζήτησης (για ανάκτηση) και ανάκτησης προφίλ απλού χρήστη· η λειτουργία επεξεργασίας προφίλ μπορεί να προσπελαστεί άμεσα μέσω του αντίστοιχου υπερσυνδέσμου στην αρχική σελίδα απλού χρήστη, user_home.html, ενώ οι συζευγμένες λειτουργίες αναζήτησης και ανάκτησης προφίλ, οι οποίες συσχετίζονται αμφότερες με τη σελίδα ανάκτησης προφίλ χρήστη, μπορούν να προσπελαστούν έμμεσα (και μόνο όταν το προφίλ του χρήστη είναι κενό) μέσω του αντίστοιχου υπερσυνδέσμου στη σελίδα επεξεργασίας προφίλ χρήστη.

Η λειτουργία εμφάνισης της σελίδας user_home.html απευθύνεται στους απλούς χρήστες του SLUB και προϋποθέτει την προγενέστερη επιτέλεση της λειτουργίας εισόδου στο SLUB. Για αυτόν τον λόγο, επιτελείται μέσω της συμπληρωματικής όψης display_user_home_page() που συσχετίζεται με το URL http://slub.softlab.ntua.gr/user/home και περιτυλίσσεται από τον διακοσμητή User_login_required(). Μέσω του υπερσυνδέσμου «Αρχική Σελίδα» στην μπάρα συνεδρίας, η λειτουργία εμφάνισης της αρχικής σελίδας απλού χρήστη μπορεί μάλιστα να επιτελεστεί ανά πάσα στιγμή από οποιαδήποτε σελίδα του SLUB (πρότυπο-απόκριση με κατάσταση "200 OK") στα πλαίσια μίας συνεδρίας απλού χρήστη.

Αν όλα τα στοιχεία προφίλ (profile elements) ενός απλού χρήστη του SLUB (στιγμιότυπο User) είναι κενά, τότε η αντίστοιχη αρχική σελίδα θα περιέχει ένα ενημερωτικό μήνυμα που τον προτρέπει να τα συμπληρώσει άμεσα μέσω της λειτουργίας επεξεργασίας προφίλ (βλ. Σχήμα 4.20). Ομοίως, αν το μόνο κενό στοιχείο προφίλ του χρήστη είναι το ΑΜ (Αριθμός Μητρώου στη Σχολή ΗΜΜΥ), τότε η αντίστοιχη αρχική σελίδα θα περιέχει ένα ενημερωτικό μήνυμα που τον προτρέπει να το συμπληρώσει άμεσα, αν το διαθέτει (ειδάλλως, π.χ. ένας φοιτητής που εισάγεται στη Σχολή μέσω των κατατακτήριων εξετάσεων και δεν αποκτά άμεσα ΑΜ, μπορεί προσωρινά να το παραλείψει), μέσω της προαναφερθείσας λειτουργίας (βλ. Σχήμα 4.21). Τέλος, αν όλα τα στοιχεία προφίλ του χρήστη είναι συμπληρωμένα, τότε η αντίστοιχη αρχική σελίδα δεν θα περιέχει κανένα σχετικό ενημερωτικό μήνυμα.

Σχήμα 4.20: Αρχική σελίδα απλού χρήστη του SLUB όταν όλα τα στοιχεία προφίλ του χρήστη είναι κενά

Σχήμα 4.21: Αρχική σελίδα απλού χρήστη του SLUB όταν ο αριθμός μητρώου είναι το μόνο κενό στοιχείο προφίλ του χρήστη

4.5.2 Επεξεργασία προφίλ χρήστη

Η λειτουργία επεξεργασίας προφίλ (απλού) χρήστη απευθύνεται στους απλούς χρήστες του SLUB και η ροή της βασίζεται σε εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας:

- Συμπληρωματική όψη display_default_user_profile_edit_page():
  - URL: http://slub.softlab.ntua.gr/user/profile/edit/default
  - Διακοσμητές:
    - User_login_required()
  - Κύριο σώμα: Εμφάνισε τη σελίδα επεξεργασίας προφίλ χρήστη, user_profile_edit.html, στην προεπιλεγμένη μορφή της, δηλαδή χωρίς μηνύματα που σχετίζονται με την υποβολή της φόρμας.
- Λοιπές παρατηρήσεις:
  - Αν το προφίλ ενός απλού χρήστη του SLUB (στιγμιότυπο User) είναι κενό (όλα τα στοιχεία προφίλ του χρήστη είναι κενά), τότε η φόρμα της αντίστοιχης σελίδας επεξεργασίας προφίλ θα είναι επίσης κενή και θα συνοδεύεται από το εξής ενημερωτικό μήνυμα:

    ✪ Αν καταχωρήσατε τα στοιχεία του προφίλ σας κατά τη διάρκεια παλιότερης εισόδου σας στο σύστημα και αυτά δεν εμφανίζονται τώρα στην παρακάτω φόρμα (είτε γιατί έχετε αλλάξει τα διαπιστευτήρια της πλατφόρμας εισόδου είτε γιατί έχετε εισέλθει για πρώτη φορά με διαπιστευτήρια αυτής της πλατφόρμας), παρακαλούμε ανακτήστε τα από τη βάση χωρίς να συμπληρώσετε τα πεδία της παρακάτω φόρμας!

    →Ανάκτηση χωρίς Συμπλήρωση

    που τον παραπέμπει στη λειτουργία αναζήτησης προφίλ (για ανάκτηση) μέσω του περιεχόμενου υπερσυνδέσμου «Ανάκτηση χωρίς Συμπλήρωση» (βλ. Σχήμα 4.22). Αντίθετα, αν το προφίλ του χρήστη είναι συμπληρωμένο (είτε όλα τα στοιχεία προφίλ του χρήστη είναι συμπληρωμένα είτε ο αριθμός μητρώου είναι το μόνο κενό στοιχείο προφίλ του), τότε η φόρμα της αντίστοιχης σελίδας επεξεργασίας προφίλ θα είναι επίσης συμπληρωμένη με τα στοιχεία προφίλ του χρήστη και δεν θα συνοδεύεται από κανένα σχετικό ενημερωτικό μήνυμα / υπερσύνδεσμο (για παράδειγμα, βλ. Σχήμα 4.23).

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη:

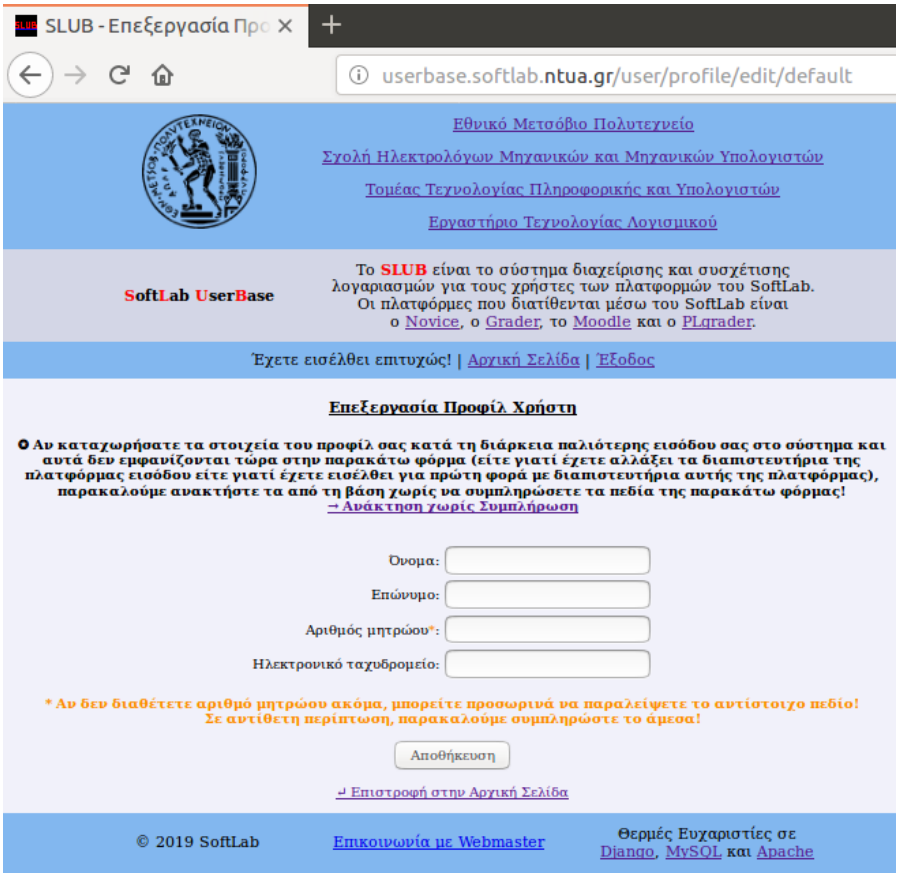
- Περιορισμοί για τις τιμές των πεδίων της HTML φόρμας user_profile_edit_form (για παράδειγμα σφάλματος υποβολής, βλ. Σχήμα 4.24):
  - Όνομα:
    - Απαίτηση συμπλήρωσης
    - Μέγιστο μήκος: 50 χαρακτήρες
  - Επώνυμο:
    - Απαίτηση συμπλήρωσης
    - Μέγιστο μήκος: 50 χαρακτήρες
  - Αριθμός μητρώου:
    - Μέγιστο μήκος: 8 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης: 031[0-9]{5}

- Ηλεκτρονικό ταχυδρομείο:
  - Απαίτηση συμπλήρωσης
  - Μέγιστο μήκος: 100 χαρακτήρες
  - Μοτίβο κανονικής έκφρασης: Περιγράφεται αναλυτικά στη διεύθυνση [70].

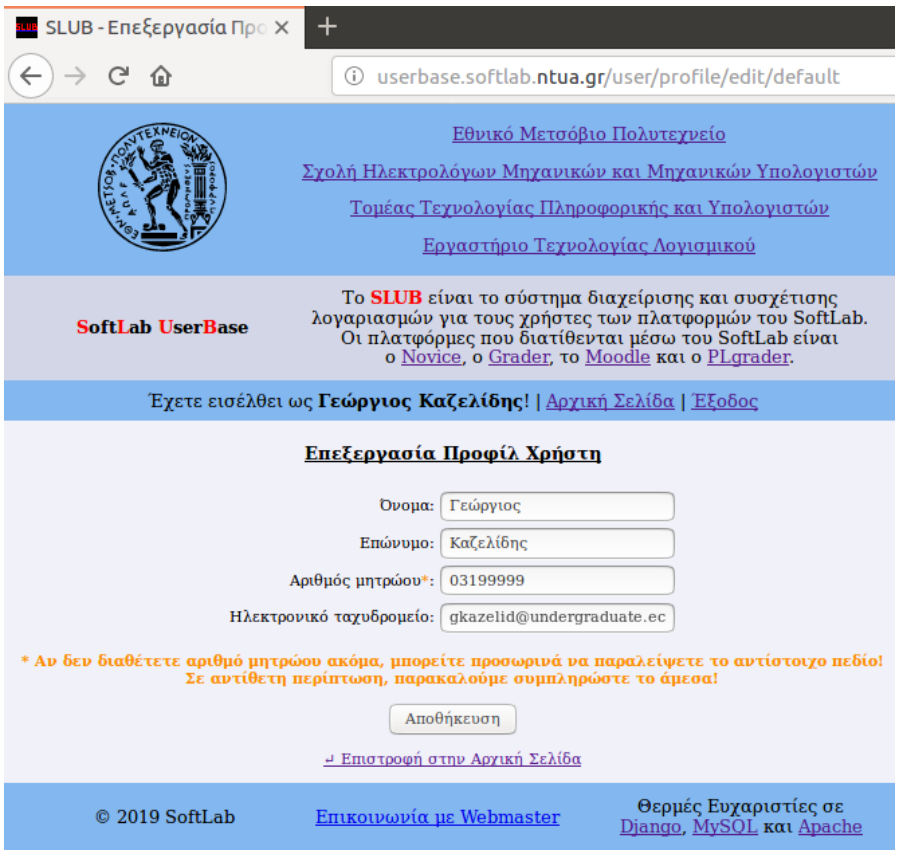
(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας:

- Κύρια όψη edit_user_profile():
  - URL: http://slub.softlab.ntua.gr/user/profile/edit/submit
  - Διακοσμητές:
    - User_login_required()
    - transaction.atomic(): Ο ταυτοχρονισμός των ατομικών συναλλαγών στο κύριο σώμα της όψης επιτυγχάνεται χρησιμοποιώντας τη μέθοδο select_for_update() για την ανάκτηση (retrieval) του στιγμιοτύπου session_user από τον πίνακα usermerge_user της βάσης δεδομένων.
  - Κύριο σώμα: Σύλλεξε τα δεδομένα, δηλαδή το όνομα, το επώνυμο, τον αριθμό μητρώου και το ηλεκτρονικό ταχυδρομείο, που υποβλήθηκαν μέσω της φόρμας επεξεργασίας προφίλ χρήστη και επικύρωσέ τα. Αν δεν είναι έγκυρα, εμφάνισε τα κατάλληλα μηνύματα σφαλμάτων επικύρωσης στη σελίδα επεξεργασίας προφίλ χρήστη. Διαφορετικά, έλεγξε αν τα πεδία της φόρμας αλλάχθηκαν σε σχέση με τα αντίστοιχα πεδία του προφίλ του χρήστη της συνεδρίας. Αν δεν έγινε καμία αλλαγή, εμφάνισε το κατάλληλο μήνυμα σφάλματος μετεπικύρωσης στη σελίδα επεξεργασίας προφίλ χρήστη και συμπερίλαβε τον αντίστοιχο κωδικό σφάλματος, "unchanged_profile_form_fields", στο πλαίσιο της σελίδας. Διαφορετικά, έλεγξε αν οι επικυρωθείσες τιμές του αριθμού μητρώου και του ηλεκτρονικού ταχυδρομείου επαρκούν για να τροποποιήσουν με συνέπεια το προφίλ του χρήστη της συνεδρίας στη βάση δεδομένων, λαμβάνοντας υπόψιν τις τιμές των αντίστοιχων πεδίων του τελευταίου:
    - ✓ Αν επαρκούν, έλεγξε ποια πεδία της φόρμας αλλάχθηκαν, ενημέρωσε το προφίλ του χρήστη της συνεδρίας στη βάση δεδομένων με τις αντίστοιχες επικυρωθείσες τιμές και εμφάνισε το ενημερωμένο προφίλ μαζί με το κατάλληλο μήνυμα επιτυχίας στη σελίδα επεξεργασίας προφίλ χρήστη (για παράδειγμα, βλ. Σχήμα 4.25).
    - ✓ Διαφορετικά, δημιούργησε τα κατάλληλα μηνύματα σφαλμάτων μετεπικύρωσης χρησιμοποιώντας τις επικυρωθείσες τιμές του αριθμού μητρώου ή/και του ηλεκτρονικού ταχυδρομείου της φόρμας μαζί με τους αντίστοιχους κωδικούς σφαλμάτων, "registered_ece_id_was_cleared" (στην περίπτωση αυτού του σφάλματος, ο επικυρωθείς αριθμός μητρώου είναι None, δηλαδή κενός, και αντ' αυτού χρησιμοποιείται ο μη-κενός αριθμός μητρώου του χρήστη της συνεδρίας), "duplicate_ece_id" ή/και "duplicate_email" (τα σφάλματα "registered_ece_id_was_cleared" και "duplicate_ece_id" δεν μπορούν ποτέ να συμβούν ταυτόχρονα), εμφάνισέ τα στη σελίδα επεξεργασίας προφίλ χρήστη και συμπερίλαβε τους κωδικούς σφαλμάτων στο πλαίσιο της σελίδας (για παράδειγμα, βλ. Σχήμα 4.26).

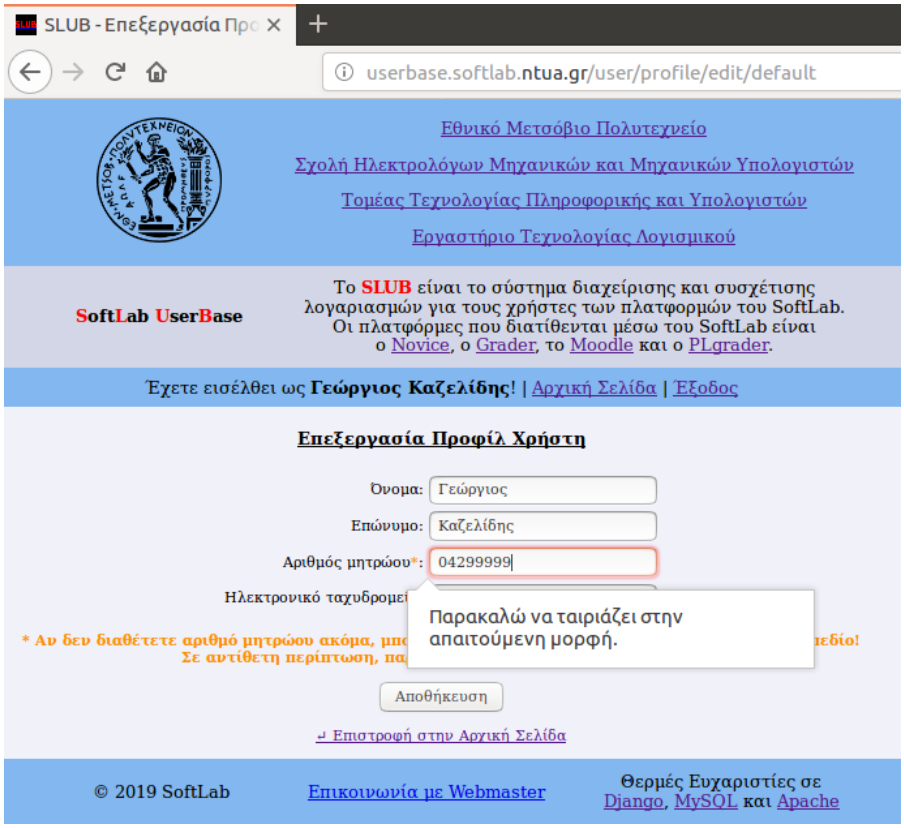
- Περιορισμοί για τις τιμές των πεδίων της Django φόρμας UserProfileEditForm - οι περιορισμοί για τις τιμές όλων των πεδίων, εξαιρουμένου του πεδίου του αριθμού μητρώου, ελέγχονται μετά την αφαίρεση (stripping) των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων τους (leading and trailing whitespace characters):
  - Όνομα:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Μέγιστο μήκος (σφάλμα "max_length"): 50 χαρακτήρες
  - Επώνυμο:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Μέγιστο μήκος (σφάλμα "max_length"): 50 χαρακτήρες
  - Αριθμός μητρώου:
    - Μέγιστο μήκος (σφάλμα "max_length"): 8 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης (σφάλμα "invalid"): 031[0-9]{5}
      Με άλλα λόγια, το μοτίβο είναι 031YYSSS, όπου YY είναι το έτος εισαγωγής (year of admission) στη Σχολή και SSS είναι ο σχετικός αύξων αριθμός (serial number).
    - Απαγόρευση του μοτίβου 031YY000 (σφάλμα "ece_id_is_031YY000")
  - Ηλεκτρονικό ταχυδρομείο:
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Μέγιστο μήκος (σφάλμα "max_length"): 100 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης (σφάλμα "invalid"): Περιγράφεται αναλυτικά στη διεύθυνση [71].



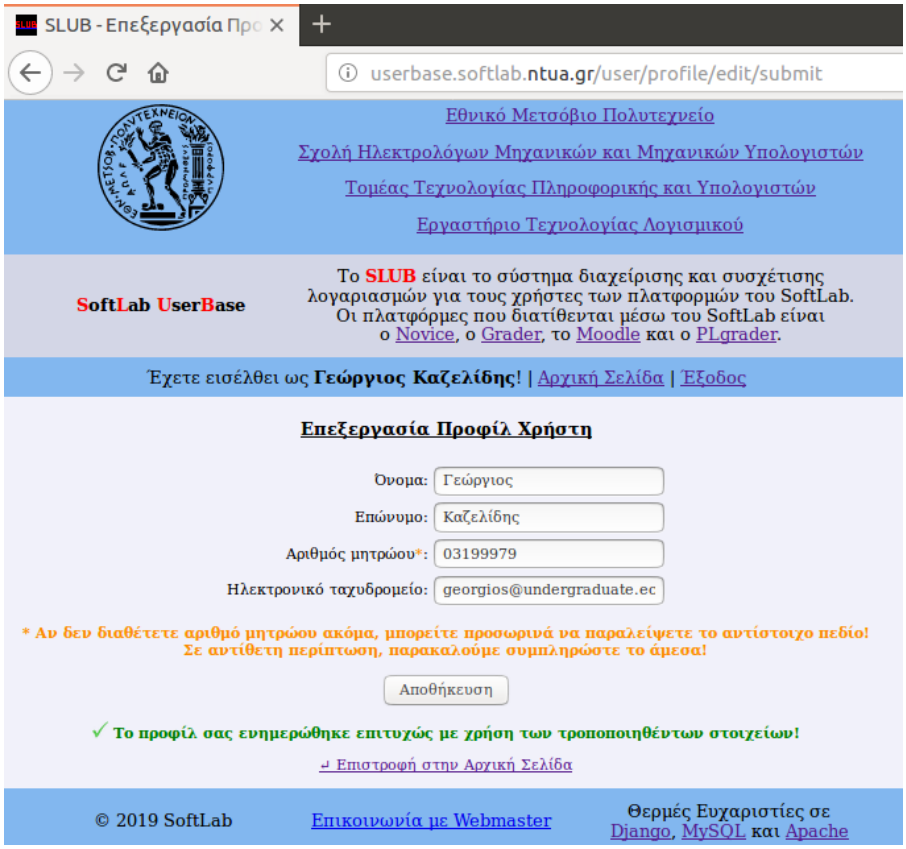
Σχήμα 4.22: Σελίδα επεξεργασίας προφίλ χρήστη του SLUB στην προεπιλεγμένη μορφή της όταν το προφίλ του χρήστη είναι κενό



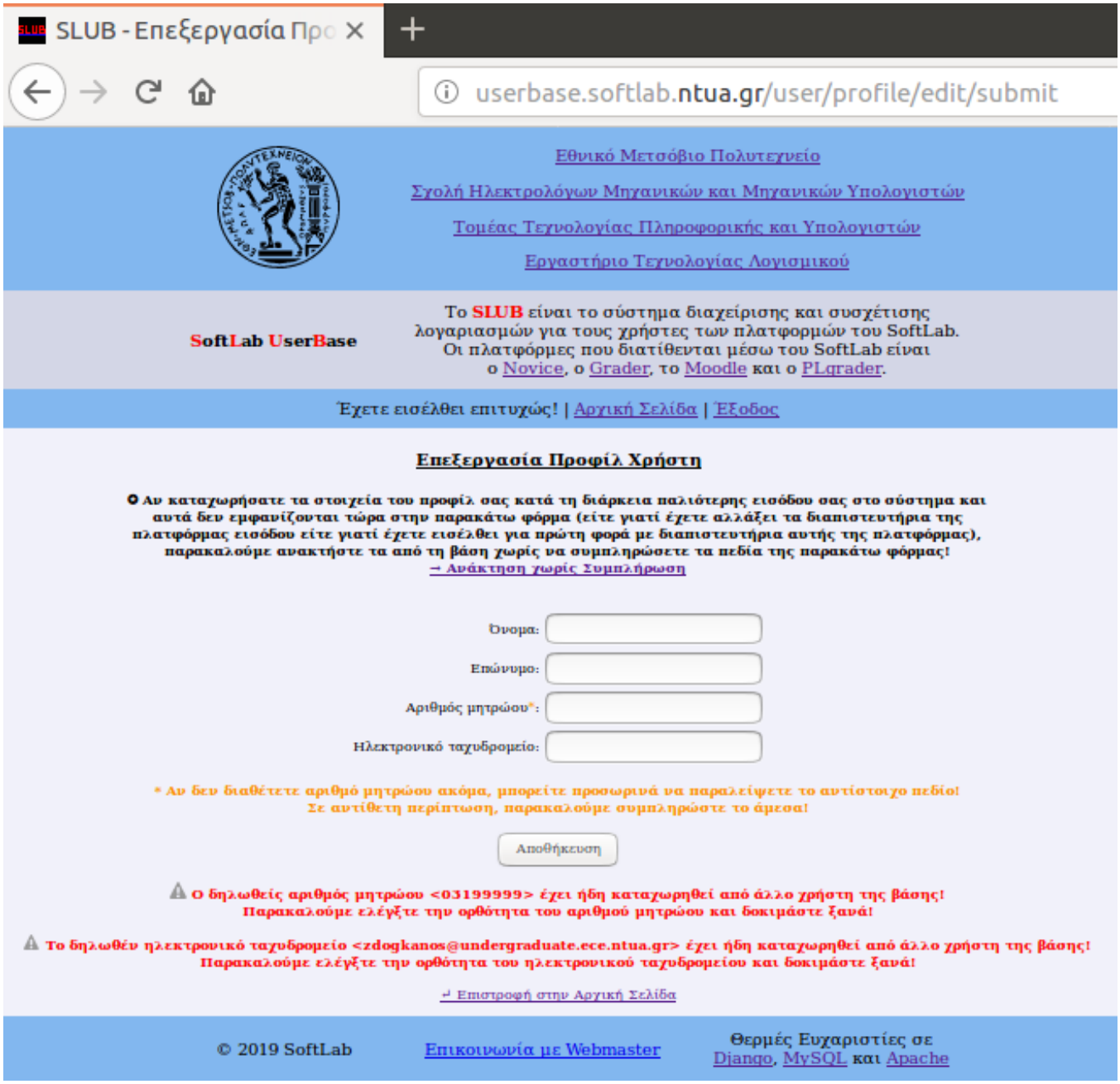
Σχήμα 4.23: Σελίδα επεξεργασίας προφίλ χρήστη του SLUB στην προεπιλεγμένη μορφή της όταν το προφίλ του χρήστη είναι συμπληρωμένο



Σχήμα 4.24: Εμφάνιση ενός μηνύματος σφάλματος υποβολής στη σελίδα επεξεργασίας προφίλ χρήστη του SLUB



Σχήμα 4.25: Εμφάνιση του μηνύματος επιτυχίας και του ενημερωμένου προφίλ του χρήστη στη σελίδα επεξεργασίας προφίλ χρήστη του SLUB



Σχήμα 4.26: Εμφάνιση μηνυμάτων σφαλμάτων μετεπικύρωσης στη σελίδα επεξεργασίας προφίλ χρήστη του SLUB

4.5.3 Αναζήτηση προφίλ χρήστη για ανάκτηση

Η λειτουργία αναζήτησης προφίλ (απλού) χρήστη για ανάκτηση απευθύνεται στους απλούς χρήστες του SLUB και η ροή της βασίζεται σε εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2:

(Α) Έλεγχοι πρόσβασης και μετάβαση στην ιστοσελίδα της λειτουργίας:

- Συμπληρωματική όψη display_default_user_profile_recovery_page():
  - URL: http://slub.softlab.ntua.gr/user/profile/recovery/default
  - Διακοσμητές:
    - User_login_required()

- Κύριο σώμα: Έλεγξε αν το προφίλ του χρήστη της συνεδρίας είναι κενό στη βάση δεδομένων. Αν όχι, εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (αν το προφίλ του χρήστη είναι ήδη μη-κενό, δεν έχει νόημα να προσδιοριστεί και να ανακτηθεί για αυτόν ένα μη-κενό προφίλ από τη βάση δεδομένων - βλ. όψεις search_for_recovery_user_profile() και recover_user_profile() παρακάτω). Διαφορετικά, εμφάνισε τη σελίδα ανάκτησης προφίλ χρήστη, user_profile_recovery.html, στην προεπιλεγμένη μορφή της (βλ. Σχήμα 4.27), δηλαδή χωρίς μηνύματα που σχετίζονται με την υποβολή της φόρμας.

(Β) Συμπλήρωση της φόρμας στην ιστοσελίδα και έλεγχοι υποβολής της από πλευράς πελάτη:

- Περιορισμοί για τις τιμές των πεδίων της HTML φόρμας user_profile_recovery_form (για παράδειγμα σφάλματος υποβολής, βλ. Σχήμα 4.28):
  - Αριθμός μητρώου:
    - Μέγιστο μήκος: 8 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης: 031[0-9]{5}
  - Ηλεκτρονικό ταχυδρομείο:
    - Απαίτηση συμπλήρωσης
    - Μέγιστο μήκος: 100 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης: Περιγράφεται αναλυτικά στη διεύθυνση [70].

(Γ) Έλεγχοι πρόσβασης στην όψη και επικύρωσης της φόρμας από πλευράς διακομιστή και επιτέλεση της λειτουργίας:

- Κύρια όψη search_for_recovery_user_profile():
  - URL: http://slub.softlab.ntua.gr/user/profile/recovery/search
  - Διακοσμητές:
    - User_login_required()
  - Κύριο σώμα: Έλεγξε αν το προφίλ του χρήστη της συνεδρίας είναι κενό στη βάση δεδομένων. Αν όχι, εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (αν το προφίλ του χρήστη είναι ήδη μη-κενό, δεν έχει νόημα να αναζητηθεί για αυτόν ένα μη-κενό προφίλ προς ανάκτηση από τη βάση δεδομένων). Διαφορετικά, σύλλεξε τα δεδομένα, δηλαδή τον αριθμό μητρώου και το ηλεκτρονικό ταχυδρομείο, που υποβλήθηκαν μέσω της φόρμας ανάκτησης προφίλ χρήστη και επικύρωσέ τα. Αν δεν είναι έγκυρα, εμφάνισε τα κατάλληλα μηνύματα σφαλμάτων επικύρωσης στη σελίδα ανάκτησης προφίλ χρήστη. Διαφορετικά, έλεγξε αν επαρκούν για να προσδιορίσουν ένα προφίλ ανάκτησης, δηλαδή ένα μη-κενό προφίλ χρήστη (στιγμιότυπο User) προς ανάκτηση, στη βάση δεδομένων (το προφίλ αυτό, αν υπάρχει, θα αντιστοιχεί τόσο στον επικυρωθέντα αριθμό μητρώου όσο και στο επικυρωθέν ηλεκτρονικό ταχυδρομείο):
    - ✓ Αν δεν επαρκούν, δημιούργησε τα κατάλληλα μηνύματα σφαλμάτων μετεπικύρωσης χρησιμοποιώντας τις επικυρωθείσες τιμές του αριθμού μητρώου ή/και του ηλεκτρονικού ταχυδρομείου της φόρμας μαζί με τους αντίστοιχους κωδικούς σφαλμάτων, "ece_id_and_email_exist_in_different_profiles",

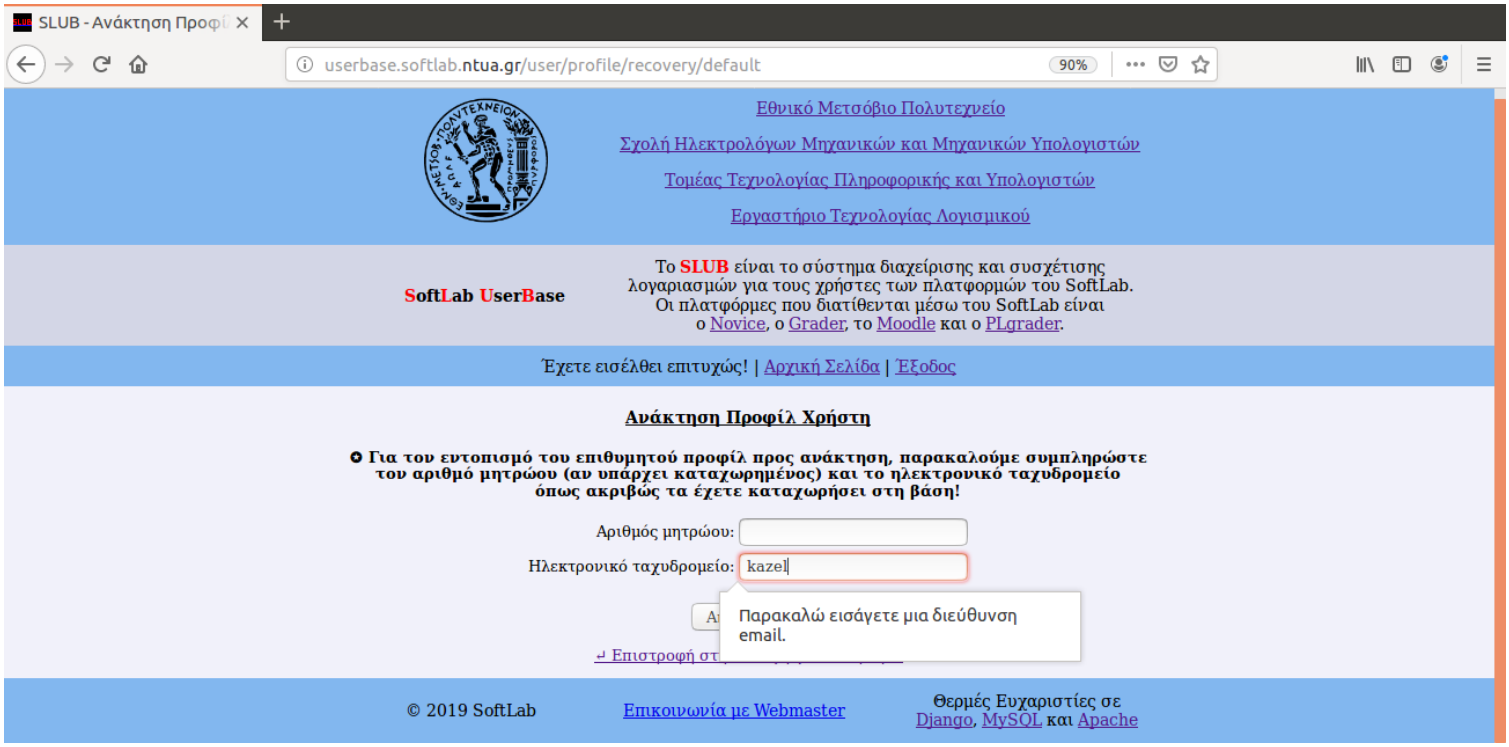
"non_existent_ece_id" ή/και "non_existent_email" (το σφάλμα "ece_id_and_email_exist_in_different_profiles" πάντα συμβαίνει μεμονωμένα, δηλαδή δεν μπορεί ποτέ να συμβεί ταυτόχρονα με κάποιο από τα σφάλματα "non_existent_ece_id" και "non_existent_email"), εμφάνισέ τα στη σελίδα ανάκτησης προφίλ χρήστη και συμπερίλαβε τους κωδικούς σφαλμάτων στο πλαίσιο της σελίδας (για παράδειγμα, βλ. Σχήμα 4.29). Επειδή δεν προσδιορίστηκε κανένα προφίλ ανάκτησης στη βάση δεδομένων, απότρεψε τη λειτουργία ανάκτησης προφίλ εξασφαλίζοντας ότι η τιμή "recov_user_id" δεν υπάρχει στα δεδομένα συνεδρίας (αν υπάρχει, διάγραψέ την).

- ✓ Διαφορετικά, εμφάνισε το κατάλληλο μήνυμα επιτυχίας στη σελίδα ανάκτησης προφίλ χρήστη (για παράδειγμα, βλ. Σχήμα 4.30), λαμβάνοντας υπόψιν το προσδιορισθέν προφίλ ανάκτησης. Αυτό το μήνυμα (I) παρουσιάζει στον χρήστη της συνεδρίας τις τιμές όλων των πεδίων του προαναφερθέντος προφίλ, (II) τον πληροφορεί ότι αν επιλέξει να το ανακτήσει, τότε τα διαπιστευτήριά του για την πλατφόρμα εισόδου θα συσχετιστούν με αυτό (θα αντικαταστήσουν τυχόν προηγούμενα διαπιστευτήρια που συσχετίζονταν με αυτό) και εκείνος θα αποσυνδεθεί αυτόματα από την εφαρμογή, και (III) τον παραπέμπει στη λειτουργία ανάκτησης προφίλ μέσω του περιεχόμενου υπερσυνδέσμου «Ανάκτηση και Έξοδος». Τέλος, επίτρεψε τη λειτουργία ανάκτησης προφίλ αποθηκεύοντας το αναγνωριστικό του προσδιορισθέντος προφίλ ανάκτησης ως τιμή "recov_user_id" στα δεδομένα συνεδρίας (το αναγνωριστικό αυτό αντικαθιστά τυχόν προηγούμενη τιμή "recov_user_id" των δεδομένων συνεδρίας).

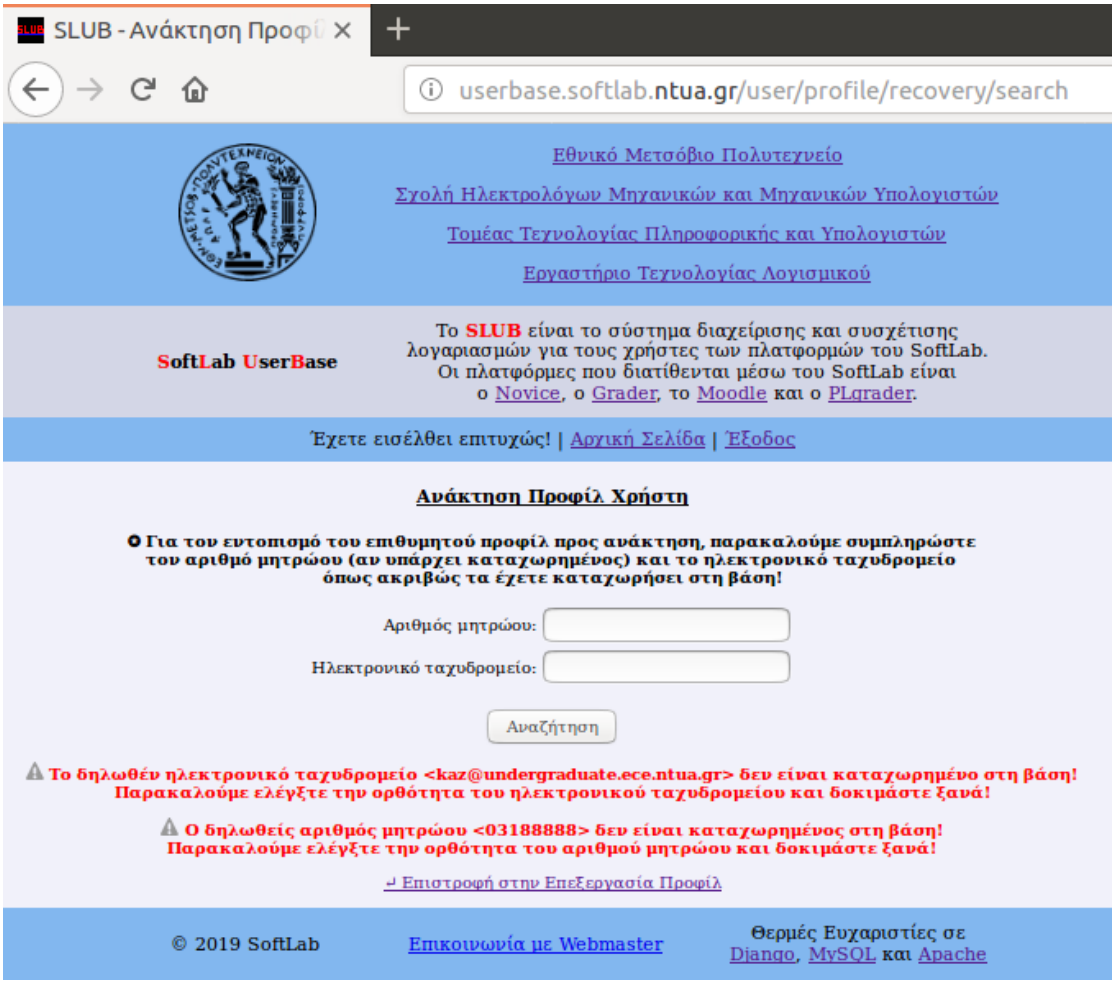
- Περιορισμοί για τις τιμές των πεδίων της Django φόρμας UserProfileRecoveryForm:
  - Αριθμός μητρώου:
    - Μέγιστο μήκος (σφάλμα "max_length"): 8 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης (σφάλμα "invalid"): 031[0-9]{5}
      Με άλλα λόγια, το μοτίβο είναι 031YYSSS, όπου YY είναι το έτος εισαγωγής (year of admission) στη Σχολή και SSS είναι ο σχετικός αύξων αριθμός (serial number).
    - Απαγόρευση του μοτίβου 031YY000 (σφάλμα "ece_id_is_031YY000")
  - Ηλεκτρονικό ταχυδρομείο - οι περιορισμοί για την τιμή αυτού του πεδίου ελέγχονται μετά την αφαίρεση (stripping) των αρχικών και τελικών μη-εκτυπώσιμων χαρακτήρων της (leading and trailing whitespace characters):
    - Απαίτηση συμπλήρωσης (σφάλμα "required")
    - Μέγιστο μήκος (σφάλμα "max_length"): 100 χαρακτήρες
    - Μοτίβο κανονικής έκφρασης (σφάλμα "invalid"): Περιγράφεται αναλυτικά στη διεύθυνση [71].
- Λοιπές παρατηρήσεις:
  - Η λειτουργία αναζήτησης προφίλ χρήστη (για ανάκτηση) αποτελεί το προπαρασκευαστικό στάδιο της λειτουργίας ανάκτησης προφίλ χρήστη, δηλαδή η παρούσα όψη, search_for_recovery_user_profile(), χρησιμοποιείται για την κατάλληλη διαμόρφωση των αντίστοιχων

πλαισίων συνεδρίας και σελίδας (session and page contexts) πριν την εκτέλεση της όψης recover_user_profile() που περιγράφεται στην ενότητα 4.5.4. Ειδικότερα για το πλαίσιο συνεδρίας, η κατάσταση της τιμής "recov_user_id" τροποποιείται κατάλληλα στα δεδομένα συνεδρίας για να αντιπροσωπεύει το πιο πρόσφατο αποτέλεσμα της παρούσας όψης, δηλαδή η τιμή "recov_user_id" είτε θα υπάρχει στα δεδομένα συνεδρίας και θα αντιπροσωπεύει το αναγνωριστικό του προσδιορισθέντος προφίλ ανάκτησης είτε δεν θα υπάρχει στα δεδομένα συνεδρίας και θα υποδεικνύει ότι δεν προσδιορίστηκε κανένα προφίλ ανάκτησης. Η προαναφερθείσα τιμή δεν διαβιβάζεται ποτέ στην όψη recover_user_profile() ως παράμετρος του αντίστοιχου URL· σε αντίθετη περίπτωση, θα ήταν ευάλωτη σε χειροκίνητες αλλαγές (π.χ. HTML/JavaScript τροποποιήσεις) και, κατά συνέπεια, θα μπορούσε να παρακάμψει το εκάστοτε προφίλ ανάκτησης που θα προσδιοριζόταν δεόντως από την παρούσα όψη.

Σχήμα 4.27: Σελίδα ανάκτησης προφίλ χρήστη του SLUB στην προεπιλεγμένη μορφή της



Σχήμα 4.28: Εμφάνιση ενός μηνύματος σφάλματος υποβολής στη σελίδα ανάκτησης προφίλ χρήστη του SLUB



Σχήμα 4.29: Εμφάνιση μηνυμάτων σφαλμάτων μετεπικύρωσης στη σελίδα ανάκτησης προφίλ χρήστη του SLUB

Σχήμα 4.30: Εμφάνιση του μηνύματος επιτυχίας στη σελίδα ανάκτησης προφίλ χρήστη του SLUB

4.5.4 Ανάκτηση προφίλ χρήστη

Σύμφωνα με όσα αναφέρθηκαν στην ενότητα 4.5.3, η λειτουργία ανάκτησης προφίλ χρήστη αποτελεί τη φυσική συνέχεια της λειτουργίας αναζήτησης προφίλ χρήστη (για ανάκτηση), δηλαδή η όψη recover_user_profile() πραγματοποιεί τους ίδιους ελέγχους πρόσβασης με την όψη search_for_recovery_user_profile() και βασίζεται στα αντίστοιχα πλαίσια συνεδρίας και σελίδας που διαμορφώνονται κατάλληλα από την τελευταία εκ των προτέρων. Ως εκ τούτου, η ροή της λειτουργίας ανάκτησης προφίλ, η οποία διαφοροποιείται αρκετά από εκείνην που περιγράφηκε τυπικά στην ενότητα 4.2.2, σχηματοποιείται ως εξής:

- Κύρια όψη recover_user_profile():
  - URL: http://slub.softlab.ntua.gr/user/profile/recovery/submit
  - Διακοσμητές:
    - User_login_required()

        - transaction.atomic(): Ο ταυτοχρονισμός των ατομικών συναλλαγών στο κύριο σώμα της όψης επιτυγχάνεται χρησιμοποιώντας τη μέθοδο select_for_update() για την ανάκτηση (retrieval) του στιγμιοτύπου session_registry από τον πίνακα usermerge_registry της βάσης δεδομένων.
    - Κύριο σώμα: Έλεγξε αν το προφίλ του χρήστη της συνεδρίας είναι άδειο στη βάση δεδομένων και αν η τιμή "recov_user_id" υπάρχει στα δεδομένα συνεδρίας, δηλαδή αν προσδιορίστηκε ένα προφίλ ανάκτησης στη βάση δεδομένων μέσω της όψης search_for_recovery_user_profile(). Αν κάποιος από τους ελέγχους αποτύχει, εμφάνισε μία σελίδα HTTP 403 (Forbidden) εγείροντας μία εξαίρεση PermissionDenied (είναι αδύνατον να ανακτηθεί ένα απροσδιόριστο μη-κενό προφίλ - στιγμιότυπο User - από τη βάση δεδομένων, ενώ αν το προφίλ του χρήστη της συνεδρίας είναι ήδη μη-κενό, δεν έχει νόημα να ανακτηθεί για αυτόν ούτε καν ένα προσδιορισθέν). Διαφορετικά, διάγραψε τυχόν προηγούμενα διαπιστευτήρια που διατηρούνταν από τον χρήστη ανάκτησης για την πλατφόρμα εισόδου στη βάση δεδομένων και συσχέτισέ τον με τα αντίστοιχα διαπιστευτήρια του χρήστη της συνεδρίας. Συγκεκριμένα, διάγραψε το στιγμιότυπο Registry που αντιστοιχεί τόσο στον χρήστη ανάκτησης όσο και στην πλατφόρμα εισόδου, αν αυτό υπάρχει στη βάση δεδομένων, και ενημέρωσε το υπάρχον στιγμιότυπο Registry που αντιστοιχεί τόσο στον χρήστη της συνεδρίας όσο και στην πλατφόρμα εισόδου, ώστε να αναφέρεται πλέον στον χρήστη ανάκτησης. Στη συνέχεια, διάγραψε το προφίλ του χρήστη της συνεδρίας για να ολοκληρωθεί η συγχώνευσή του με τον χρήστη ανάκτησης και αν τα διαπιστευτήρια του τελευταίου για την πλατφόρμα εισόδου αλλάχθηκαν, δηλαδή αν εκείνος διατηρούσε προηγούμενα διαπιστευτήρια πριν λάβει τα αντίστοιχα του χρήστη της συνεδρίας, χρησιμοποίησε τον καταγραφέα "usermerge.views.recover_user_profile" για την καταγραφή του κατάλληλου μηνύματος INFO, το οποίο θα περιλαμβάνει το προηγούμενο και το τρέχον όνομα χρήστη (διαπιστευτήριο-κλειδί) του χρήστη ανάκτησης για την πλατφόρμα εισόδου. Τέλος, αποσύνδεσε τον χρήστη της συνεδρίας από την εφαρμογή (διάγραψε τα δεδομένα και το cookie της συνεδρίας καθώς αυτή βασίζεται σε ένα ανύπαρκτο πλέον προφίλ και, ως εκ τούτου, είναι άκυρη/ασυνεπής) και ανακατεύθυνέ τον στην προεπιλεγμένη σελίδα εισόδου.
- Λοιπές παρατηρήσεις:
    - Η μορφή των μηνυμάτων INFO, τα οποία καταγράφονται από τον καταγραφέα "usermerge.views.recover_user_profile", περιγράφεται από την printf-οειδή (printf-like) συμβολοσειρά "user_id: %d | platform: %s (id: %d) | old_username: %s | new_username: %s" της Python, όπου "%d" είναι το προσδιοριστικό μετατροπής (conversion specifier) σε ακέραιο αριθμό μέσω της ενσωματωμένης συνάρτησης (built-in function) int() της Python και "%s" είναι το προσδιοριστικό μετατροπής σε συμβολοσειρά μέσω της ενσωματωμένης συνάρτησης str() της Python[72].
    - Από τη μία, η συνάρτηση django.contrib.auth.logout() εκπέμπει (emits) το σήμα django.contrib.auth.signals.user_logged_out που, μεταξύ άλλων, παρέχει το προφίλ του αποσυνδεόμενου χρήστη στις αντίστοιχες συναρτήσεις χειρισμού (handler functions). Από την άλλη, το προφίλ του χρήστη της συνεδρίας διαγράφεται από τη βάση δεδομένων στα πλαίσια

της παρούσας όψης, recover_user_profile(), και δεν μπορεί να προσπελαστεί από καμία συνάρτηση. Επομένως, η παρούσα όψη δεν καλεί την όψη log_out() (βλ. ενότητα 4.3.2) που αξιοποιεί την προαναφερθείσα συνάρτηση logout(), αλλά αποσυνδέει τον χρήστη της συνεδρίας από την εφαρμογή (πρώτα διαγράφει τα δεδομένα και το cookie της συνεδρίας και κατόπιν θέτει ένα στιγμιότυπο AnonymousUser ως τρέχοντα χρήστη στο σχετικό αίτημα, request.user) και τον ανακατευθύνει στην προεπιλεγμένη σελίδα εισόδου χειροκίνητα.

5 Συμπεράσματα

5.1 Συνεισφορά

Η συνεισφορά της εργασίας και τα συμπεράσματα που προέκυψαν είναι τα παρακάτω:

- Αναπτύχθηκε και επιστρατεύτηκε στη διεύθυνση http://slub.softlab.ntua.gr/ η εφαρμογή ιστού SLUB, η οποία επιτελεί την κεντρική διαχείριση και συσχέτιση λογαριασμών για τους χρήστες των πλατφορμών του SoftLab μέσω της δικής της βάσης δεδομένων, usermergeDB, ώστε να διευκολύνει την ταυτοποίηση χρηστών από τους υπεύθυνους του Εργαστηρίου, τόσο σε επίπεδο μίας όσο και σε επίπεδο πολλαπλών πλατφορμών. Για την ανάπτυξη της εφαρμογής, της οποίας ο πηγαίος κώδικας και η τεκμηρίωση διατίθενται στη διεύθυνση https://git.softlab.ntua.gr/giorgkazelidis/userbase, χρησιμοποιήθηκαν οι γλώσσες HTML-CSS-JavaScript στο εμπρόσθιο τμήμα και οι συνιστώσες Linux-Apache-MySQL-Python της στοίβας λογισμικού LAMP στο οπίσθιο τμήμα.

  Οι προαναφερθείσες τεχνολογίες συνδυάστηκαν μέσω του πλαισίου ιστού Django, το οποίο ακολουθεί το αρχιτεκτονικό μόρφημα μοντέλου-προτύπου-όψης, MTV, και την προδιαγραφή WSGI. Σύμφωνα με το μόρφημα MTV, υλοποιήθηκαν (I) αντιπροσωπευτικά μοντέλα τόσο για τους απλούς χρήστες και διαχειριστές της εφαρμογής όσο και για τις πλατφόρμες του SoftLab και τις διαπιστευτικές συσχετίσεις των απλών χρηστών με αυτές, τα οποία απεικονίστηκαν σε κατάλληλους πίνακες της βάσης usermergeDB μέσω του Django ORM, (II) επιχειρηματικές όψεις που συναλλάσσονται με τη βάση usermergeDB, δηλαδή δημιουργούν, διαβάζουν, ενημερώνουν ή/και διαγράφουν εγγραφές των πινάκων της, για την επιτέλεση των λειτουργιών της εφαρμογής ανά ρόλο/μοντέλο χρήστη, και (III) διαδραστικά πρότυπα για τη δήλωση των απαιτούμενων δεδομένων εισόδου και την παρουσίαση των μηνυμάτων ή/και δεδομένων επιτυχούς/ανεπιτυχούς εξόδου των προαναφερθεισών λειτουργιών/όψεων στον ιστό.

  Τέλος, ενεργοποιήθηκε στη διεύθυνση http://slub.softlab.ntua.gr/django-admin/ ο ιστότοπος διαχείρισης του Django, ο οποίος απευθύνεται αποκλειστικά στους χρήστες του ομώνυμου πλαισίου και δρα ως διαχειριστική διεπαφή CRUD των πινάκων της βάσης usermergeDB.
- Παρατηρήθηκε η απουσία άμεσης διασύνδεσης των λογαριασμών χρηστών στο SLUB με τους αντίστοιχους σε κάποιες από τις πλατφόρμες του SoftLab. Από τη μία, η εφαρμογή επιτελεί άμεσα, δηλαδή μέσω των τρεχουσών λειτουργιών της, την κεντρική διαχείριση και συσχέτιση λογαριασμών για πλατφόρμες όπως ο Novice (και ο Grader, στη βάση δεδομένων του οποίου καταχωρούνται συνήθως τα διαπιστευτήρια των χρηστών του Novice), για τις οποίες τα διαπιστευτήρια παρέχονται στατικά στους χρήστες, δηλαδή παράγονται μαζικά από μία γεννήτρια συνάρτηση (generator function) και κατόπιν διαμοιράζονται στους φοιτητές της Σχολής ΗΜΜΥ (απλούς χρήστες του SLUB) από τους υπεύθυνους του SoftLab (διαχειριστές του SLUB), με σκοπό τη δημιουργία/αλλαγή λογαριασμών σε αυτές. Από την άλλη, η εφαρμογή αδυνατεί να επιτελέσει άμεσα την κεντρική διαχείριση και συσχέτιση λογαριασμών για πλατφόρμες όπως το Moodle (και ο PLgrader, ο οποίος αξιοποιεί τα διαπιστευτήρια των χρηστών που είναι καταχωρημένα στη βάση δεδομένων του Moodle), για τις οποίες τα διαπιστευτήρια

δημιουργούνται/αλλάζονται δυναμικά από τους χρήστες που εγγράφονται (sign up) σε αυτές.

Στην τελευταία περίπτωση, οι υπεύθυνοι του SoftLab πρέπει να αντλούν τακτικά, π.χ. καθημερινά, τα εκάστοτε νεοδημιουργηθέντα/νεοαλλαχθέντα διαπιστευτήρια από τις βάσεις δεδομένων των πλατφορμών και να τα εισάγουν/ενημερώνουν στον πίνακα usermerge_registry της βάσης usermergeDB, λαμβάνοντας υπόψιν το σύνθετο κλειδί (composite key) {username, platform_id}, ούτως ώστε οι φοιτητές να μπορούν να εισέλθουν στην εφαρμογή με τα προαναφερθέντα διαπιστευτήρια και να τα συσχετίσουν με το προφίλ τους· δεδομένου ότι οι κωδικοί πρόσβασης που είναι καταχωρημένοι στη βάση usermergeDB είναι κωδικοποιημένοι σε σχήμα PBKDF2-SHA256 (Password-Based Key Derivation Function 2 using the 256-bit Secure Hash Algorithm) και ότι εκείνοι που αντλούνται από τις βάσεις δεδομένων των πλατφορμών είναι κατά πάσα πιθανότητα κωδικοποιημένοι σε ένα σχήμα X, π.χ. SHA-1 (160-bit SHA), MD5 (Message-Digest algorithm 5) ή ακόμα και PBKDF2-SHA256, το οποίο ενδέχεται να είναι διαφορετικό στη βάση δεδομένων καθεμίας από τις πλατφόρμες, πρέπει να διασφαλιστεί μέσω των συναρτήσεων check_password() και make_password() της μονάδας django.contrib.auth.hashers[73] ότι (I) οι τελευταίοι θα κωδικοποιούνται σε σχήμα PBKDF2-SHA256 κατά την εισαγωγή/ενημέρωσή τους στον πίνακα usermerge_registry της βάσης usermergeDB, και (II) οι ανεπεξέργαστοι (raw) κωδικοί πρόσβασης για καθεμία από τις πλατφόρμες θα κωδικοποιούνται πρώτα στο εκάστοτε σχήμα X και έπειτα σε PBKDF2-SHA256 κατά την αυθεντικοποίηση των φοιτητών στην εφαρμογή (απαιτούνται πιθανότατα κατάλληλες τροποποιήσεις στη μέθοδο authenticate() της κλάσης AppUserModelBackend στο αρχείο backends.py).

Γίνεται, λοιπόν, σαφές ότι χρειάζεται να υλοποιηθεί (τουλάχιστον) μία συνάρτηση/κλάση Django που θα αυτοματοποιεί, π.χ. μέσω εργασίας cron (cron job), τις προαναφερθείσες ενέργειες των υπεύθυνων του SoftLab και, κατά συνέπεια, θα επιτρέπει τον βολικό και άμεσο συγχρονισμό (synchronization) των διαπιστευτηρίων στη βάση usermergeDB με τα αντίστοιχα στις βάσεις δεδομένων των πλατφορμών που συμπεριφέρονται όπως το Moodle (και ο PLgrader). Για τον χρονοπρογραμματισμό εργασιών (job scheduling) στο Django με χρήση του δαίμονα (daemon) cron διατίθενται διάφορα πακέτα στο ευρετήριο PyPI (Python Package Index), όπως το django-cron[74], το django-crontab[75] και το django-kronos[76].

5.2 Μελλοντική βελτίωση και επέκταση

Με αφορμή το τελευταίο συμπέρασμα που προέκυψε από την εργασία στην ενότητα 5.1, παρατίθεται πιο κάτω μία (μη-εξαντλητική) λίστα προτεινόμενων μελλοντικών δράσεων για τη βελτίωση των υπαρχόντων χαρακτηριστικών και την προσθήκη καινούριων στο SLUB (βλ. τμήμα "LIST OF SUGGESTED TO-DO ACTIONS" στο αρχείο README.md):

- Ορισμός μίας εργασίας cron[77] που χρησιμοποιεί τη διαχειριστική εντολή mysqldump της MySQL για την τακτική δημιουργία αντιγράφων ασφαλείας (backups) της βάσης δεδομένων του SLUB, δηλαδή αρχείων κατεκτύπωσης (dump files) που επιτρέπουν την ανάνηψη (recovery) της βάσης usermergeDB σε

περίπτωση κατάρρευσης συστήματος (system crash), αστοχίας υλισμικού (hardware failure), κατά λάθος διαγραφής δεδομένων, κ.λπ.[78]

- Ορισμός μίας εργασίας cron που χρησιμοποιεί τη διαχειριστική εντολή clearsessions του Django για την τακτική εκκαθάριση των ληγμένων συνεδριών από τον πίνακα django_session της βάσης usermergeDB και την αποτροπή της ανεξέλεγκτης πλήθυνσής του[79]
- Δημιουργία και χρήση μίας όψης (και του αντίστοιχου προτύπου) που απευθύνεται σε στιγμιότυπα Admin και πραγματοποιεί μαζική διαγραφή (bulk/batch deletion) στιγμιοτύπων User και Registry βάσει συγκεκριμένων κριτηρίων, π.χ. κενότητας του απαιτούμενου πεδίου email ή/και «παλαιότητας» της τιμής του πεδίου last_login, για την εξασφάλιση της ελεγχόμενης πλήθυνσης των πινάκων usermerge_user και usermerge_registry της βάσης usermergeDB - βλ. διεύθυνση [80] και σχετικές παρατηρήσεις στο αρχείο models.py
- Δημιουργία και εκτέλεση αυτοματοποιημένων δοκιμών, π.χ. δοκιμών μονάδων (unit tests), μέσω των αντίστοιχων εργαλείων του Django[81]
- Καλύτερη αξιοποίηση των δυνατοτήτων του ιστοτόπου διαχείρισης του Django[51]
- Χρησιμοποίηση των μεθόδων επικύρωσης μοντέλων του Django[82]
- Χρησιμοποίηση των σχημάτων αναβάθμισης (upgrading) και επικύρωσης κωδικών πρόσβασης του Django[83, 84]
- Καλύτερη αξιοποίηση των χαρακτηριστικών ασφαλείας του Django, π.χ. ενεργοποίηση της προστασίας HTTPS/SSL (HTTP Secure / Secure Sockets Layer protection), ενεργοποίηση της επικύρωσης κεφαλίδας (header) "Host", κ.λπ.[85]

Βιβλιογραφία

1. https://en.wikipedia.org/wiki/Client%E2%80%93server_model
2. https://en.wikipedia.org/wiki/Session_(computer_science)
3. https://en.wikipedia.org/wiki/Web_application
4. https://en.wikipedia.org/wiki/Multitier_architecture
5. https://en.wikipedia.org/wiki/Web_application_development
6. https://en.wikipedia.org/wiki/Web_framework
7. https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
8. https://en.wikipedia.org/wiki/Web_template_system
9. https://en.wikipedia.org/wiki/Web_cache
10. https://en.wikipedia.org/wiki/Authentication
11. https://en.wikipedia.org/wiki/Computer_access_control
12. https://en.wikipedia.org/wiki/Front_and_back_ends
13. https://en.wikipedia.org/wiki/Web_development
14. https://www.upwork.com/hiring/development/a-beginners-guide-to-back-end-development/
15. https://www.w3schools.com/whatis/
16. https://en.wikipedia.org/wiki/Front-end_web_development
17. https://en.wikipedia.org/wiki/Solution_stack
18. https://en.wikipedia.org/wiki/LAMP_(software_bundle)
19. https://www.upwork.com/hiring/development/back-end-web-developer/
20. https://en.wikipedia.org/wiki/Linux
21. https://en.wikipedia.org/wiki/Apache_HTTP_Server
22. https://en.wikipedia.org/wiki/MySQL
23. https://en.wikipedia.org/wiki/SQL
24. https://en.wikipedia.org/wiki/Python_(programming_language)
25. https://en.wikipedia.org/wiki/Django_(web_framework)
26. https://docs.djangoproject.com/en/2.0/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names
27. https://djangobook.com/mdj2-django-structure/
28. https://www.javatpoint.com/django-mvt
29. https://en.wikipedia.org/wiki/Web_Server_Gateway_Interface
30. https://www.toptal.com/python/pythons-wsgi-server-application-interface
31. https://docs.djangoproject.com/en/2.0/ref/applications/
32. https://docs.djangoproject.com/en/2.0/howto/initial-data/
33. https://docs.djangoproject.com/en/2.0/intro/tutorial01/
34. https://virtualenv.pypa.io/en/latest/
35. https://docs.djangoproject.com/en/2.0/topics/install/
36. https://docs.djangoproject.com/en/2.0/howto/windows/
37. https://pip.pypa.io/en/stable/user_guide/
38. https://docs.python-guide.org/dev/virtualenvs/
39. https://pypi.org/project/concurrent-log-handler/0.9.12/
40. https://docs.djangoproject.com/en/2.0/ref/databases/#mysql-db-api-drivers
41. https://pypi.org/project/pytz/2018.4/
42. https://en.wikipedia.org/wiki/Git

43. https://stackoverflow.com/questions/38388423/what-does-on-delete-do-on-django-models
44. https://tech.marksblogg.com/passwords-in-django.html
45. https://docs.djangoproject.com/en/2.0/topics/auth/default/
46. https://docs.djangoproject.com/en/2.0/topics/auth/
47. https://docs.djangoproject.com/en/2.0/ref/contrib/auth/
48. https://docs.djangoproject.com/en/2.0/topics/auth/customizing/#authorization-for-anonymous-users
49. https://docs.djangoproject.com/en/2.0/intro/tutorial02/#introducing-the-django-admin
50. https://docs.djangoproject.com/en/2.0/intro/overview/#a-dynamic-admin-interface-it-s-not-just-scaffolding-it-s-the-whole-house
51. https://docs.djangoproject.com/en/2.0/ref/contrib/admin/
52. https://docs.djangoproject.com/en/2.0/howto/deployment/wsgi/modwsgi/#serving-the-admin-files
53. http://djangoweekly.com/blog/post/viewbrowse-all-django-admin-edits-recent-actions-listing
54. https://docs.djangoproject.com/en/2.0/topics/http/sessions/
55. https://docs.djangoproject.com/en/2.0/topics/http/middleware/
56. https://docs.djangoproject.com/en/2.0/topics/auth/default/#the-login-required-decorator
57. https://docs.djangoproject.com/en/2.0/topics/db/transactions/
58. https://docs.djangoproject.com/en/2.0/ref/models/querysets/#select-for-update
59. https://en.wikipedia.org/wiki/ACID
60. https://docs.djangoproject.com/en/2.0/ref/views/#error-views
61. https://docs.djangoproject.com/en/2.0/topics/http/views/
62. https://docs.djangoproject.com/en/2.0/topics/http/urls/
63. https://docs.djangoproject.com/en/2.0/ref/models/
64. https://docs.djangoproject.com/en/2.0/ref/templates/
65. https://docs.djangoproject.com/en/2.0/ref/contrib/staticfiles/
66. https://docs.djangoproject.com/en/2.0/search/?q=decorator
67. https://docs.djangoproject.com/en/2.0/ref/forms/
68. https://docs.djangoproject.com/en/2.0/ref/validators/
69. https://www.w3schools.com/tags/att_form_enctype.asp
70. https://html.spec.whatwg.org/multipage/input.html#valid-e-mail-address
71. https://github.com/django/django/blob/stable/2.0.x/django/core/validators.py#L167-L179
72. https://docs.python.org/3.5/library/stdtypes.html#printf-style-string-formatting
73. https://docs.djangoproject.com/en/2.0/topics/auth/passwords/#module-django.contrib.auth.hashers
74. https://pypi.org/project/django-cron/
75. https://pypi.org/project/django-crontab/
76. https://pypi.org/project/django-kronos/
77. https://www.ostechnix.com/a-beginners-guide-to-cron-jobs/
78. https://dev.mysql.com/doc/refman/5.7/en/backup-and-recovery.html

79. https://docs.djangoproject.com/en/2.0/topics/http/sessions/#clearing-the-session-store
80. https://docs.djangoproject.com/en/2.0/ref/models/instances/#deleting-objects
81. https://docs.djangoproject.com/en/2.0/topics/testing/
82. https://docs.djangoproject.com/en/2.0/ref/models/instances/#validating-objects
83. https://docs.djangoproject.com/en/2.0/topics/auth/passwords/#password-upgrading
84. https://docs.djangoproject.com/en/2.0/topics/auth/passwords/#module-django.contrib.auth.password_validation
85. https://docs.djangoproject.com/en/2.0/topics/security/