

Ιόνιο Πανεπιστήμιο

Τμήμα Πληροφορικής

Δημιουργία μίας εφαρμογής κοινωνικής δικτύωσης για διαμοίραση και σχολιασμό προγραμματιστικού κώδικα

Πτυχιακή εργασία



Αλβανιτόπουλος Ευάγγελος(Π2013119)

Επιβλέπων: Μιχαήλ Στεφανιδάκης

2 ΜΑΙΟΥ 2018

ΠΕΡΙΛΗΨΗ

Ο σκοπός της πτυχιακής εργασίας είναι η δημιουργία μίας εφαρμογής κοινωνικής δικτύωσης για διαμοίραση και σχολιασμό προγραμματιστικού κώδικα. Η εφαρμογή δίνει τη δυνατότητα στους χρήστες να εγγραφούν στη σελίδα και να δημοσιοποιήσουν την προγραμματιστικού περιεχομένου ερώτησή τους. Με αυτόν τον τρόπο δημιουργείται μια κοινότητα ατόμων, τα οποία βοηθούν το ένα το άλλο λύνοντας απορίες και μοιράζοντας συμβουλές και γνώσεις. Επιπλέον, δημιουργείται μια βάση δεδομένων, η οποία περιέχει συμβουλές, γνώσεις και απορίες από έμπειρους προγραμματιστές αλλά και από άτομα τα οποία λιγότερης εμπειρίας.

Η εφαρμογή θα αναπτυχθεί με τη χρήση της Python ως γλώσσα προγραμματισμού και της MYSQL ως βάση δεδομένων.

ΠΕΡΙΕΧΟΜΕΝΑ

Περίληψη.....	3
ΚΕΦΑΛΑΙΟ 1.....	7
1.ΕΙΣΑΓΩΓΗ.....	7
1.1 ΣΚΟΠΟΣ.....	8
1.2 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ.....	9
1.3 ΤΕΧΝΟΛΟΓΙΕΣΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ.....	10
1.3.1 MYSQL PHERMYADMIN.....	11
1.3.2 MySQL.....	11
1.3.3 ATOM.....	12
1.3.4 BootstrapCDN.....	13
1.3.5 FACEBOOK.....	15
1.3.6 FLASK.....	16
1.3.6.1 FLASK ROUTING.....	17
1.3.6.2 TEMPLATE ENGINE.....	18
1.3.6.3 THREAD-LOCALS.....	18
1.3.6.4 ΕΓΚΑΤΑΣΤΑΣΗ FLASK.....	19
1.3.6.5 EXTENSIONS.....	19
1.3.6.6 BLUEPRINTS.....	20
1.3.6.7 WTFForms.....	21

ΚΕΦΑΛΑΙΟ 2.....	22
2. ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ.....	22
2.1 ΡΥΤΗΘΗ.....	22
2.1.1 ΔΟΜΗ ΚΑΙ ΣΥΝΤΑΞΗ.....	23
2.2 HTML.....	24
2.2.1 ΣΤΟΙΧΕΙΑ HTML.....	25
2.3 CSS.....	25
ΚΕΦΑΛΑΙΟ 3.....	26
3. ΕΦΑΡΜΟΓΗ.....	26
3.1 ΛΕΙΤΟΥΡΓΙΕΣ ΕΦΑΡΜΟΓΗΣ.....	26
3.1.1 ΟΙ ΧΡΗΣΤΕΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ.....	27
3.1.2 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ.....	29
3.1.2.1 ΠΙΝΑΚΕΣ.....	29
3.2 ΕΙΚΟΝΕΣ ΚΑΙ ΕΠΕΞΗΓΗΣΗ ΚΩΔΙΚΑ.....	31
3.2.1 ΑΡΧΙΚΗ ΣΕΛΙΔΑ.....	31
3.2.2 ΣΧΕΤΙΚΑ ΜΕ ΕΜΑΣ.....	32
3.2.3 REGISTER PAGE.....	33
3.2.4 LOGIN.....	35
3.2.5 DASHBOARD ΚΑΙ ΔΗΜΙΟΥΡΓΙΑ ΑΡΘΡΟΥ.....	36
3.2.6 ΛΙΣΤΑ ΑΡΘΡΩΝ.....	38

3.2.7 ΠΡΟΣΘΗΚΗ ΚΑΙ ΠΡΟΒΟΛΗ ΣΧΟΛΙΩΝ.....	40
3.2.8 ΑΠΟΣΥΝΔΕΣΗ.....	42
3.3 ΚΩΔΙΚΑΣ.....	43
3.3.1 Layout.....	44
3.3.2 ΑΡΧΙΚΗ ΣΕΛΙΔΑ.....	45
3.3.3 USERS.....	45
3.3.4 ARTICLES.....	51
3.3.5 COMMENTS.....	57
3.3.6 ΠΙΝΑΚΑΣ ΕΛΕΓΧΟΥ.....	62
ΚΕΦΑΛΑΙΟ 4.....	65
4. ΣΥΜΠΕΡΑΣΜΑΤΑ – ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ.....	65
4.1 ΣΥΜΠΕΡΑΣΜΑΤΑ.....	65
4.2 ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ.....	67
ΒΙΒΛΙΟΓΡΑΦΙΑ.....	68

ΚΕΦΑΛΑΙΟ 1

1.ΕΙΣΑΓΩΓΗ

Τις τελευταίες δεκαετίες, με την ραγδαία εξέλιξη της τεχνολογίας, ο άνθρωπος έχει βελτιώσει σε μεγάλο βαθμό την καθημερινότητά του. Τα προβλήματα που αντιμετωπίζει καθημερινά και οι ανάγκες που έχει ικανοποιούνται γρηγορότερα και αποτελεσματικότερα από ότι γινόταν στο παρελθόν. Οι μετακινήσεις του γίνονται γρηγορότερα και με μεγαλύτερη άνεση, η απόκτηση και η συντήρηση της τροφής του γίνεται αποτελεσματικότερα και για μεγαλύτερο χρονικό διάστημα σε σύγκριση με το παρελθόν.

Τα τελευταία χρόνια δίνεται ιδιαίτερη σημασία στην ικανοποίηση και στην βελτίωση δυο ακόμα αναγκών του ατόμου, της ψυχαγωγίας και της ενημέρωσης. Η εξέλιξη της τεχνολογίας και ειδικότερα η δημιουργία του World Wide Web, έχει συνεισφέρει θετικά στην επίτευξη αυτού του σκοπού. Το Web μπορούμε να το φανταστούμε σαν μια μεγάλη βιβλιοθήκη. Αυτή η βιβλιοθήκη περιέχει ένα μεγάλο όγκο από πληροφορίες, οι οποίες είναι προσβάσιμες από όλους τους χρήστες του διαδικτύου. Στην εποχή μας, κάθε άνθρωπος έχει τουλάχιστον έναν τρόπο πρόσβασης (προσωπικός υπολογιστής, laptop, smartphone) με τον οποίο μπορεί μπει στο Web και να χρησιμοποιήσει τις πληροφορίες που του προσφέρει.

Πλέον, το Web αποτελεί μέσω επικοινωνίας με φίλους, συγγενείς, γνωστούς και άλλους συνανθρώπους μας που βρίσκονται μιλιά μακριά. Να ενημερωθούμε για το τι συμβαίνει σε ολόκληρη την υφήλιο οποιαδήποτε στιγμή και σε γρήγορο χρονικό διάστημα.

Μπορούμε ακόμα να ενημερωθούμε και αναζητήσουμε όποια πληροφορία επιθυμήσουμε. Εάν θέλουμε να μάθουμε μια πληροφορία ή μια απορία τότε με την βοήθεια μιας μηχανής αναζήτησης θα βρούμε απάντηση σε όποιο θέμα αναζητούμε. Μπορούμε να κάνουμε πολλές συναλλαγές μέσω διαδικτύου όπως να πληρώσουμε λογαριασμούς σε ιδιώτες ή σε δημοσίους φορείς γρήγορα και με ασφάλεια. Οι δυνατότητες και οι προοπτικές του διαδικτύου είναι απεριόριστες.

1.1 ΣΚΟΠΟΣ

Όλες αυτές οι τεχνολογικές αλλαγές συνεισφέρουν στην εμφάνιση του προγραμματισμού σαν αντικείμενο αλλά και σαν επάγγελμα. Ο προγραμματισμός περιλαμβάνει πολλές γλώσσες προγραμματισμού, οι οποίες εξελίσσονται και βελτιώνονται συνεχώς. Κάθε γλώσσα έχει τα χαρακτηριστικά και τις ιδιαιτερότητες της καθώς και την χρησιμότητα της. Δηλαδή, δεν χρησιμοποιούνται όλες οι γλώσσες για τον ίδιο σκοπό. Κάθε προγραμματιστής, όταν μαθαίνει μια καινούργια γλώσσα ή όταν βελτιώνει τις γνώσεις του σε μια άλλη, αντιμετωπίζει δυσκολίες και προβλήματα.

Το διαδίκτυο είναι ένας εύκολος τρόπος να λύσει αυτά τα προβλήματα. Με την πλοήγηση του σε σελίδες σχετικές με το προγραμματιστικό πρόβλημα που αντιμετωπίζει και χρησιμοποιώντας την γνώση και την εμπειρία τρίτων, μπορεί να βρει τη λύση στο πρόβλημα του. Γενικότερα, ο προγραμματισμός είναι ένα αντικείμενο με το οποίο ασχολούνται όλο και περισσότεροι άνθρωποι στον κόσμο, όπου ο καθένας έχει τις δικές του εμπειρίες και γνώσεις, τις οποίες όμως μπορεί να μοιραστεί, έτσι ώστε να βοηθήσει τους υπόλοιπους .

Ο σκοπός της πτυχιακής εργασίας είναι η δημιουργία η μίας εφαρμογής κοινωνικής δικτύωσης για διαμοίραση και σχολιασμό προγραμματιστικού κώδικα.

1.2 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Τα άτομα που επισκέπτονται τη σελίδα έχουν την δυνατότητα να δουν τα άρθρα που έχει η σελίδα. Μπορούν δηλαδή να δουν τις ερωτήσεις και τις απαντήσεις που έδωσαν οι εγγεγραμμένοι χρήστες της σελίδας. Δεν μπορούν όμως να απαντήσουν ή να δημιουργήσουν μια ερώτηση. Για να το κάνουν αυτό, πρέπει να έχουν κάνει λογαριασμό στη σελίδα. Χρησιμοποιώντας ένα user-name, ένα password και το email τους, μπορούν να δημιουργήσουν λογαριασμό, ο οποίος θα τους επιτρέψει να υποβάλουν τις ερωτήσεις και τις απαντήσεις τους. Επιπλέον, οι χρήστες θα μπορούν να συνδεθούν και με τον λογαριασμό τους στο facebook και θα τους δίνεται η δυνατότητα να μοιράζονται τις αναρτήσεις της σελίδας με τους χρήστες του facebook. Επιπρόσθετα, οι χρήστες οι οποίοι έχουν δημιουργήσει κάποιου είδους δημοσίευσης, δηλαδή είτε ερώτηση είτε απάντηση, τους δίνεται η δυνατότητα επεξεργασίας και διαγραφής της δημοσίευσης. Τέλος, όταν ολοκληρώσουν την περιήγηση τους, οι εγγεγραμμένοι χρήστες, μπορούν να κάνουν αποσύνδεση από την σελίδα.

Στο πρώτο κεφάλαιο παρουσιάζονται ο σκοπός της πτυχιακής , η γενική περιγραφή της εφαρμογής και οι τεχνολογίες που χρησιμοποιήθηκαν. Οι τεχνολογίες που έχουν χρησιμοποιηθεί είναι η εφαρμογή mysql phpmyadmin, MySQL ως server, ο κειμενογράφος Atom, BootstrapCDN, facebook intergration και η βιβλιοθήκη flask.

Στο δεύτερο κεφάλαιο παρουσιάζονται τα εργαλεία που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής. Τα εργαλεία είναι η γλώσσα προγραμματισμού python, η γλώσσα σήμανσης υπερκείμενου html και η γλώσσα σήμανσης css.

Στο τρίτο κεφάλαιο παρουσιάζεται η εφαρμογή. Πρώτα αναφέρονται και αναλύονται οι βασικές λειτουργίες της εφαρμογής. Στη συνέχεια, παρουσιάζεται η εφαρμογή με διάφορες εικόνες για τον τρόπο λειτουργίας της εφαρμογής συνοδευόμενες με την ανάλογη επεξήγηση. Τέλος, γίνεται αναφορά του κώδικα της εφαρμογής.

Στο τέταρτο κεφάλαιο αναφέρονται τα τελικά συμπεράσματα για την εφαρμογή.

Στο πέμπτο κεφάλαιο αναγράφονται οι πηγές άντλησης πληροφοριών τόσο για την συγγραφή της αναφοράς όσο και της υλοποίησης της εφαρμογής.

1.3 ΤΕΧΝΟΛΟΓΙΕΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

Οι τεχνολογίες που έχουν χρησιμοποιηθεί είναι η εφαρμογή mysql phpmyadmin, MySQL ως server, ο κειμενογράφος Atom, BootstrapCDN , facebook intergration και η βιβλιοθήκη flask.

1.3.1 MYSQL PHPMYADMIN

Η εφαρμογή MySQL phpmysqladmin χρησιμοποιήθηκε για την δημιουργία και την διαχείριση της βάσης δεδομένων της εφαρμογής. Πρόκειται για μια LAMP εφαρμογή γραμμένη ειδικά για την διαχείριση εξυπηρετών MySQL. Είναι γραμμένη σε PHP και προσβάσιμη μέσω οποιoδήποτε περιηγητή ιστοσελίδων. Επιπλέον προσφέρει ένα γραφικό περιβάλλον για εργασίες σχετικές με τη διαχείριση βάσεων δεδομένων.

Η εφαρμογή αυτή είναι ένα σύνολο από php scripts, το οποίο δίνει τη δυνατότητα διαχείρισης των βάσεων δεδομένων που έχουμε. Επιπρόσθετα, μπορεί να διαχειριστεί ένα ολόκληρο mysql server ή και ακόμα απλές βάσεις δεδομένων όπου ο κάθε χρήστης έχει ένα λογαριασμό και μπορεί να δημιουργήσει και να διαχειριστεί τις δικές του βάσεις δεδομένων.

1.3.2 MySQL

Η MySQL είναι ένα γρήγορο σύστημα διαχείρισης σχεσιακής βάσης ανοιχτού κώδικα, που χρησιμοποιεί την SQL (Structured Query Language), μια γλώσσα για την προσθήκη, την πρόσβαση και την επεξεργασία δεδομένων σε μια βάση δεδομένων. Οι βάσεις δεδομένων χρησιμοποιούνται για την αποθήκευση, την αναζήτηση, την ταξινόμηση και γενικότερα στην διαχείριση δεδομένων.

Ένας MySQL server αποτελείται από ένα σύνολο από βάσεις δεδομένων (databases), όπου και αυτές με τη σειρά τους αποτελούνται από ένα σύνολο από πίνακες (tables). Κάθε πίνακας περιέχει γραμμές, οι οποίες ονομάζονται εγγραφές (records), και

στήλες, οι οποίες ονομάζονται πεδία (fields). Κάθε πεδίο, του πίνακα, μπορεί να περιέχει μόνο ένα τύπο πληροφορίας, οποίος ορίζεται όταν δημιουργείται ο πίνακας. Κάθε πίνακας μιας βάσης δεδομένων είναι σημαντικό να περιέχει μια στήλη σε κάθε σειρά, η οποία έχει διαφορετική τιμή έτσι ώστε κάθε εγγραφή, στον πίνακα, να αποκτήσει ένα μοναδικό χαρακτηριστικό, το οποίο επιτρέπει την αναφορά σε αυτήν. Το μοναδικό αυτό πεδίο το ορίζουμε κατά την δημιουργία του πίνακα και έχει την ιδιότητα του πρωτεύοντος κλειδιού (primary key). Τέλος, αξίζει να αναφερθεί ότι δεν υπάρχει όριο στον αριθμό των βάσεων δεδομένων που περιέχονται στον server, όπως και στον αριθμό των πινάκων που περιέχονται στην βάση δεδομένων αλλά και στην ποσότητα των πεδίων και εγγραφών που περιέχονται στους πίνακες.

1.3.3 ATOM

Το ATOM είναι ένα εργαλείο για την επεξεργασία κειμένου και πηγαίου κώδικα. Το εργαλείο αυτό δεν κοστίζει, είναι ανοιχτού κώδικα (open source) και υποστηρίζεται από τα περισσότερα λειτουργικά συστήματα. Επιπλέον, υποστηρίζει plug-ins γραμμένα σε Node.js και περιέχει Git Control, που αναπτύχθηκε από το GitHub. Το Git Control παρέχει μια διεπαφή GUI για την διαχείριση όλων των συχνών git εντολών. Είναι χρήσιμο και βολικό σε περιπτώσεις όπου ο κώδικας της εφαρμογής βρίσκεται στο GitHub. Το Atom είναι μια εφαρμογή για ηλεκτρονικούς υπολογιστές και κατασκευάστηκε με τη χρήση τεχνολογιών ιστού (web technologies) και συντηρείται από την κοινότητα.

Η εγκατάσταση του Atom είναι ιδιαίτερα εύκολη, καθώς το μόνο που πρέπει να κάνει κανείς είναι να πλοηγηθεί στη σελίδα <https://atom.io/> και στη συνέχεια να κατεβάσει το εργαλείο. Τέλος, πρέπει να κάνει εγκατάσταση το αρχείο που κατέβηκε και το εργαλείο είναι έτοιμο για χρήση. Αξίζει να σημειωθεί ότι ανάλογα με το λειτουργικό σύστημα που χρησιμοποιεί κανείς, πρέπει να επιλέξει και την κατάλληλη έκδοση του εργαλείου αλλά και ο τρόπος εγκατάστασης διαφέρει από λειτουργικό σε λειτουργικό.

1.3.4 BootstrapCDN

Το BootstrapCDN είναι ένα δωρεάν και δημόσιο δίκτυο παροχής περιεχομένου. Οι χρήστες του BootstrapCDN μπορούν να φορτώσουν CSS, JavaScript για τις μορφές τυπογραφίας, κουμπιά πλοήγησης και εικόνες απομακρυσμένα, από τους διακομιστές του. Το Bootstrap χρησιμοποιεί το παγκόσμιο δίκτυο προβολής περιεχομένου του MaxCDN, το οποίο κάνει τους ιστότοπους που χρησιμοποιούν την υπηρεσία τους ανθεκτικούς σε απροσδόκητα κύματα στην κυκλοφορία ιστού (wed traffic). Έχει σχετικά ελλιπή υποστήριξη για HTML5 και CSS, αλλά είναι συμβατό με όλους τους φυλλομετρητές (browsers). Το εργαλείο αυτό χρησιμοποιήθηκε για παροχή έτοιμου κώδικα για την μορφή-όψη της εφαρμογής.

Το Bootstrap είναι σπονδυλωτό και αποτελείται ουσιαστικά από μια σειρά στυλ (stylsheets) που εφαρμόζουν τα διάφορα συστατικά του πακέτου εργαλείων. Ένα στυλ που ονομάζεται bootstrap.less περιλαμβάνει τα συστατικά stylesheets. Οι προγραμματιστές μπορούν να προσαρμόσουν το αρχείο Bootstrap, επιλέγοντας τα στοιχεία που θέλουν να χρησιμοποιήσουν στο έργο τους.

Οι προσαρμογές είναι δυνατές σε περιορισμένη έκταση μέσω ενός κεντρικού στυλ διαμόρφωσης. Η χρήση γλώσσας στυλ επιτρέπει τη χρήση για μεταβλητές, λειτουργίες και φορείς (operators), ένθετους επιλογείς, γνωστά και ως μείγματα [mixin](#). Επιπλέον, παρέχει ένα σύνολο στυλ που παρέχουν βασικούς ορισμούς στυλ για όλα τα βασικά στοιχεία HTML. Αυτά παρέχουν ενιαία, σύγχρονη εμφάνιση για πίνακες, μορφοποίηση κειμένου, καθώς και στοιχεία μιας φόρμας.

Προκειμένου να χρησιμοποιηθεί το Bootstrap σε μια σελίδα HTML, ο προγραμματιστής κάνει λήψη του στύλ CSS και περιλαμβάνει μια σύνδεση στο αρχείο HTML. Επιπλέον, του δίνεται η δυνατότητα να χρησιμοποιήσει στοιχεία JavaScript, τα οποία όμως πρέπει να αναφέρονται μαζί με τη βιβλιοθήκη jQuery στο HTML αρχείο. Τέλος, ο προγραμματιστής επιλέγει σε μια φόρμα τα επιθυμητά συστατικά και τα προσαρμόζει, εάν είναι αναγκαίο, σε τιμές διαφόρων εναλλακτικών λύσεων για τις ανάγκες του. Στη συνέχεια δημιουργείται ένα πακέτο που περιλαμβάνει ήδη το προ-χτισμένο CSS στυλ.

1.3.5 FACEBOOK

Το facebook είναι ένας ιστότοπος κοινωνικής δικτύωσης, ο οποίος έκανε την εμφάνιση του στις 4 Φεβρουαρίου του 2004, από το τότε φοιτητή του Χάρβαρντ Mark Zuckerberg και τους επίσης φοιτητές στην ίδια σχολή Eduardo Saverin, Andrew McCollum, Dustin Moskovitz και Chris Hughes. Το όνομα της πλατφόρμας που δημιουργήθηκε προέρχεται από τα έγγραφα παρουσίασης των μελών της επιστημονικής κοινότητας των Αμερικανικών κολεγίων. Ο σκοπός της εφαρμογής ήταν να χρησιμοποιείται από τους φοιτητές του πανεπιστημίου του Χάρβαρντ. Αργότερα προστέθηκαν και χρήστες από άλλα πανεπιστήμια στην Αμερική και τελικά, πρόσβαση απέκτησαν και οι μαθητές των λυκείων και πλέον από το 2006 και μετά πλατφόρμα έγινε προσβάσιμη σε κάθε άνθρωπο στον πλανήτη. Μέχρι το 2012 η αξία της εταιρίας εκτιμήθηκε στα 104 δισεκατομμύρια δολάρια, τη μεγαλύτερη αποτίμηση μέχρι σήμερα για μια νεοσυσταθείσα εταιρία. Αξίζει να αναφερθεί ότι η εγγραφή στο facebook είναι δωρεάν και τα περισσότερα έσοδα της εταιρίας προέρχονται από διαφημίσεις.

Η πρόσβαση στην πλατφόρμα μπορεί να γίνει από τις περισσότερες συσκευές με σύνδεση στο ίντερνετ, όπως επιτραπέζιοι υπολογιστές, λάπτοπ, tablet και κινητά τηλέφωνα. Μετά την πραγματοποίηση της εγγραφής, οι χρήστες μπορούν να δημιουργήσουν ένα προσαρμοσμένο προφίλ που αναφέρει πληροφορίες για αυτούς, όπως όνομα, επάγγελμα, εκπαίδευση και ούτο καθεξής. Οι ενέργειες που μπορούν να κάνουν οι χρήστες, μετά την εγγραφή είναι η προσθήκη άλλων χρηστών, “φίλων”, η ανταλλαγή μηνυμάτων, ο διαμοιρασμός φωτογραφιών και βίντεο και η χρήση διάφορων εφαρμογών. Επιπλέον, δίνεται η δυνατότητα συμμετοχής σε

ομάδες χρηστών με κοινά ενδιαφέροντα, αλλά και κατηγοριοποίησης των φίλων. Σήμερα το facebook αποτελεί τη μεγαλύτερη πλατφόρμα κοινωνικής δικτυακής επικοινωνίας με περισσότερους από 1 δισεκατομμύριο ενεργούς χρήστες, γεγονός που το κατατάσει ως ένα από τα πιο δημοφιλή κοινωνικά δίκτυα του πλανήτη.

1.3.6 FLASK

Το Flask (φλασκί) είναι ένα micro περιβάλλον για την ανάπτυξη διαδικτυακών εφαρμογών, το οποίο είναι γραμμένο σε Python και είναι βασισμένο στο Werkzeug toolkit (WSGI) και στο Jinja2 template engine. Ο όρος micro εννοεί ότι το Flask δεν απαιτεί ειδικά εργαλεία ή βιβλιοθήκες για την λειτουργία του. Αυτό, βέβαια δεν σημαίνει ότι στερείται σε λειτουργικότητα. Στην πραγματικότητα, στόχος είναι να αποτελείται από απλές βασικές λειτουργίες αλλά και είναι και επεκτάσιμο. Δηλαδή, δεν περιλαμβάνει, επίπεδο αφαίρεσης βάσεων δεδομένων, επικύρωση φόρμας που περιέχει δεδομένα, ή οτιδήποτε άλλο που υπάρχει ήδη σε άλλες βιβλιοθήκες. Αντίθετα, το Flask υποστηρίζει επεκτάσεις, οι οποίες προσθέτουν την λειτουργικότητα που απαιτείται στις εφαρμογές.

Το Flask έχει πολλές τιμές, οι οποίες μπορούν αλλάξουν, με λογικές προεπιλογές και μερικές συνήθειες στην αρχή. Για παράδειγμα, τα template και static αρχεία, αποθηκεύονται σε υποκαταλόγους στο εσωτερικό του Python source tree της εφαρμογής. Αυτό, αν και δεν είναι αναγκαίο, μπορεί να αλλάξει εύκολα όταν ξεκινάει η διαδικασία της δημιουργίας μιας εφαρμογής.

Στην αρχή της διαδικασίας της δημιουργίας μιας εφαρμογής , ο προγραμματιστής μπορεί να βρεί διαθέσιμες μια ποικιλία από επεκτάσεις, έτοιμες για χρήση. Σε κάθε σημείο της δημιουργίας ο προγραμματιστής έχει τον απόλυτο έλεγχο σχετικά με τις αποφάσεις που αφορούν τον σχεδιασμό. Η λειτουργία του Flask βασίζεται περισσότερο στην ένωση μεταξύ των λειτουργιών που προσφέρει η Python.

Μια διαδικτυακή εφαρμογή σε Python και βασισμένη σε WSGI πρέπει να διαθέτει ένα κεντρικό αντικείμενο, το οποίο καλεί την πραγματική εφαρμογή. Στο Flask αυτή είναι μια περίπτωση χρήσης της κλάσης Flask. Κάθε εφαρμογή Flask πρέπει να δημιουργεί ένα στιγμιότυπο αυτής της κλάσης και να της μεταβιβάσει το όνομα της. Αυτό συμβαίνει γιατί μπορεί να υπάρχουν αντικείμενα εφαρμογών, τα οποία απαιτούν την ύπαρξη μόνο ενός (instance) αντικειμένου.

1.3.6.1 FLASK ROUTING

Το Flask χρησιμοποιεί το σύστημα δρομολόγησης (routing) Werkzeug, το οποίο έχει σχεδιαστεί να ταξινομεί τα δρομολόγια (routes) ανάλογα με την πολυπλοκότητα τους. Αυτό σημαίνει ότι η σειρά δήλωσης των routes δεν είναι σημαντική. Το γεγονός αυτό απαιτείται όταν η εφαρμογή χωρίζεται σε πολλαπλές ενότητες.

1.3.6.2 TEMPLATE ENGINE

Επιπλέον, το Flask αποφασίζει σχετικά με το template engine, το Jinja2. Αν και ο κάθε προγραμματιστής μπορεί να χρησιμοποιήσει το δικό template engine της επιλογής του, το Flask θα ρυθμίζει πάντα το Jinja2. Οι μηχανές προτύπων (template engines) είναι σαν τις γλώσσες προγραμματισμού. Τα template engines περίπου την ίδια λειτουργία μεταξύ τους, παρόλα αυτά, κάθε ένα έχει την δική του χρήση. Για παράδειγμα, το Jinja2 διαθέτει ένα εκτεταμένο σύστημα φίλτρων, έναν ορισμένο τρόπο για κληρονόμηση των templates και άλλα. Από την άλλη μεριά ο Genshi βασίζεται στην αξιολόγηση ροής XML, την κληρονομιά προτύπου, λαμβάνοντας υπόψη τη διαθεσιμότητα του XPath και άλλα.

Όσο αφορά την σύνδεση ενός template engine με μια εφαρμογή, το Flask χρησιμοποιεί τις λειτουργίες που προσφέρει ο Jinja2. Βέβαια, ο προγραμματιστής μπορεί να χρησιμοποιήσει τη δική του templating γλώσσα, αλλά μια επέκταση εξαρτάται από το Jinja.

1.3.6.3 THREAD-LOCALS

Μια απόφαση σχετικά με τον σχεδιασμό που έχει πάρει το Flask είναι ότι οι απλές εργασίες πρέπει να είναι απλές και να μην απαιτούν μεγάλα κομμάτια κώδικα, προκειμένου να επιτευχθούν. Για παράδειγμα, χρησιμοποιεί thread-local objects (αντικείμενα που χρησιμοποιούν νήματα) έτσι ώστε όταν χρειάζεται ένα αντικείμενο σε περισσότερες από μια εργασίες, να μην χρειάζεται να ζητείται κάθε φορά.

1.3.6.4 ΕΓΚΑΤΑΣΤΑΣΗ FLASK

Η εγκατάσταση του Flask χρειάζεται την τελευταία έκδοση της Python 3. Φυσικά, υποστηρίζονται και άλλες εκδόσεις python όπως Python 3.4, Python 2.7, και PyPy. Αξίζει να αναφερθεί ότι όταν γίνει η εγκατάσταση του Flask θα γίνει και η εγκατάσταση κάποιων εξαρτημάτων (Dependencies) ταυτόχρονα. Όπως έχει ήδη αναφερθεί δυο από αυτά είναι το WSGI και το Jinja, αλλά και το MarkupSafe, το ItsDangerous τα οποία σχετίζονται με την ασφάλεια και το Click, το οποίο επιτρέπει τη χρήση της εντολής flask αλλά και την προσθήκη προσαρμοσμένων εντολών. Υπάρχουν διαθέσιμα βέβαια και κάποια Dependencies, όπως το Watchdog, που χρησιμοποιείται για πιο γρήγορη φόρτωση του server, που είναι προαιρετικά.

Όταν κάποιος προγραμματιστής έχει αναλάβει περισσότερα από ένα project, είναι λογικό ότι απαιτούνται διάφορες εκδόσεις από βιβλιοθήκες, που όμως μπορεί να προκαλούν σφάλματα μεταξύ τους. Η λύση είναι η χρήση Virtual environments. Έτσι κάθε project εκτελείται στο δικό του εικονικό χώρο και δεν επηρεάζει τα υπόλοιπα.

1.3.6.5 EXTENSIONS

Τα extensions (επεκτάσεις) είναι πρόσθετα πακέτα που προσθέτουν λειτουργικότητα σε μια flask εφαρμογή. Για παράδειγμα, μια επέκταση ενδέχεται να προσθέσει υποστήριξη για την αποστολή μηνυμάτων ηλεκτρονικού ταχυδρομείου ή τη σύνδεση σε μια βάση δεδομένων. Ορισμένες επεκτάσεις προσθέτουν ολόκληρα νέα πλαίσια για να βοηθήσουν στη δημιουργία συγκεκριμένων τύπων εφαρμογών, όπως ένα ReST API. Μερικά extensions αναφέρονται παρακάτω.

1.3.6.6 BLUEPRINTS

Το Flask χρησιμοποιεί τα blueprints για την κατασκευή τμημάτων της εφαρμογής και για την υποστήριξη των ίδιων τμημάτων σε μία ή περισσότερες εφαρμογές. Τα Blueprints μπορούν να απλοποιήσουν σε μεγάλο βαθμό τη λειτουργία των μεγάλων εφαρμογών και να παράσχουν ένα κεντρικό μέσο για τις Flask επεκτάσεις έτσι ώστε να καταχωρούνται οι εργασίες σε μια εφαρμογή. Ένα αντικείμενο Blueprint λειτουργεί παρόμοια με ένα αντικείμενο Flask μιας εφαρμογής, αλλά στην πραγματικότητα δεν είναι εφαρμογή. Πρόκειται για είναι ένα σχέδιο για το πώς μπορείς να κατασκευάσεις ή να επεκτείνεις μια εφαρμογή.

Η χρήση blueprints είναι ιδανική σε μεγάλες εφαρμογές. Η χρήση τους θα επέτρεπε σε μια εργασία να καλέσει ένα αντικείμενο να προετοιμάσει διάφορες επεκτάσεις και να καταχωρίσει πολλά blueprints. Επιπλέον, γίνεται εφικτή η καταχώρηση blueprint σε ένα πρόθεμα URL. Αυτό σημαίνει ότι μπορεί να γίνει η καταχώρηση ενός blueprint, σε μια εφαρμογή, πολλές φορές με διαφορετικό URL. Τέλος, προσφέρονται template φίλτρα, στατικά αρχεία, templates και άλλα βοηθητικά προγράμματα μέσω blueprints.

Συνοψίζοντας, όπως έχει ήδη αναφερθεί το blueprint δεν είναι εφαρμογή αλλά, ένα σύνολο λειτουργιών που μπορούν να καταχωρηθούν σε μια εφαρμογή, ακόμα και πολλές φορές. Επιπλέον, είναι δυνατή η ύπαρξη πολλαπλών αντικειμένων εφαρμογής (application objects), που κάθε ένα θα έχει τις δικές του ρυθμίσεις και θα διαχειρίζονται από το WSGI. Τέλος, τα Blueprints παρέχουν διαχωρισμό στο επίπεδο Flask, μοιράζονται το config της εφαρμογής και μπορούν να αλλάξουν ένα αντικείμενο εφαρμογής όπως απαιτείται

με την εγγραφή τους. Το μειονέκτημα είναι ότι δεν μπορεί να καταργηθεί η καταχώριση ενός blueprint μόλις δημιουργηθεί μια εφαρμογή χωρίς να χρειάζεται να καταστραφεί ολόκληρο το αντικείμενο της εφαρμογής.

1.3.6.7 WTForms

Όταν μια εφαρμογή περιέχει δεδομένα φόρμας , για παράδειγμα όνομα, επώνυμο, email, τηλέφωνο κτλ, η προβολή τους από έναν φυλλομετρητή είναι ιδιαίτερα δύσκολη. Υπάρχουν βιβλιοθήκες που έχουν σχεδιαστεί για να διευκολύνουν τη διαχείριση αυτής της διαδικασίας. Ένας από αυτούς είναι η WTForms. Πρόκειται για μια flask extension, που βοηθάει στην εύκολη και γρήγορη δημιουργία και διαχείριση του απαραίτητου κώδικα.

ΚΕΦΑΛΑΙΟ 2

2. ΕΡΓΑΛΕΙΑ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΗΘΗΚΑΝ

Τα εργαλεία που χρησιμοποιήθηκαν είναι η γλώσσα προγραμματισμού python, η γλώσσα σήμανσης υπερκείμενου html και η γλώσσα σήμανσης css.

2.1 PYTHON

Η python είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου, η οποία δημιουργήθηκε το 1990 από τον Ολλανδό Γκβιντο βαν Ρόσσουμ (Guido van Rossum). Στόχος της python είναι η προσφορά ευκολίας κατά την ανάγνωση και την χρήση της. Το απλό συντακτικό της δίνει την δυνατότητα να εκφράζονται έννοιες σε λιγότερες γραμμές σε σύγκριση με άλλες γλώσσες όπως η C++ και η Java. Οι πολλές βιβλιοθήκες που διαθέτει, διευκολύνουν σε μεγάλο βαθμό συνηθισμένες εργασίες, γεγονός που την καθιστά εύκολη την εκμάθηση της.

Η ευκολίες που προσφέρει η python την καθιστούν ιδιαίτερα δημοφιλή ανάμεσα σε άτομα που θέλουν να αρχίσουν να μαθαίνουν προγραμματισμό, αλλά και σε άτομα που έχουν μεγαλύτερη εμπειρία. Το γεγονός αυτό έχει οδηγήσει στην διαθεσιμότητα της python σε πολλά λειτουργικά συστήματα. Ο κώδικας της Python μπορεί να πακεταριστεί σε αυτόνομα εκτελέσιμα προγράμματα για μερικά από τα πιο δημοφιλή λειτουργικά συστήματα, επιτρέποντας τη διανομή του βασισμένου σε Python λογισμικού για χρήση σε αυτά τα

περιβάλλοντα χωρίς να απαιτείται εγκατάσταση του διερμηνευτή της Python.

Σημαντικό είναι να αναφερθεί ο τρόπος με τον οποίο μπορεί να γίνει η συγγραφή python προγραμμάτων. Απαραίτητος είναι ένας κειμενογράφος, όπως το Atom που αναφέρθηκε παραπάνω, ή ένα περιβάλλον ανάπτυξης, το οποίο είναι λογισμικό για την ανάπτυξη εφαρμογών. Η python περιέχει ένα τέτοιο λογισμικό το IDLE, το οποίο είναι κατασκευασμένο από τον δημιουργό της python και δίνει τη δυνατότητα συγγραφής, επεξεργασίας και εκτέλεσης προγραμμάτων.

2.1.1 ΔΟΜΗ ΚΑΙ ΣΥΝΤΑΞΗ

Η python, για την δημιουργία εκτελέσιμου κώδικα χρησιμοποιεί έναν μεταγλωττιστή (compiler). Ένα χαρακτηριστικό της γλώσσας είναι ο τρόπος χρήσης των κενών διαστημάτων (whitespace). Για τον διαχωρισμό των συντακτικών δομών η python χρησιμοποιεί τα κενά διαστήματα, σε αντίθεση με άλλες γλώσσες προγραμματισμού οι οποίες χρησιμοποιούν σύμβολα. Επιπλέον, στη θέση των συμβόλων η python χρησιμοποιεί πλήρεις αγγλικές λέξεις και σε συνδυασμό με αυτά που αναφέρθηκαν παραπάνω, καθιστούν τον κώδικα της python ευανάγνωστο από όσους έχουν τη βασική γνώση της αγγλικής γλώσσας.

2.2 HTML

Η HTML (HYPERTEXT MARKUP LANGUAGE) είναι μια γλώσσα σήμανσης για τις ιστοσελίδες και τα στοιχεία της αποτελούν βασικά δομικά στοιχεία για αυτές. Προτάθηκε από τον Tim Berners Lee το 1991 στο Cern της Γενεύης. Υπάρχουν πολλές εκδόσεις της HTML από τότε που ξεκίνησε το WWW. Δεν είναι γλώσσα προγραμματισμού, αλλά γλώσσα σήμανσης υπερκείμενου. Ο σκοπός της είναι η μεταφερσιμότητα σε υπολογιστικά συστήματα. Περιέχει τα <tag>, που μέσα σε αυτά θέτουμε πως θα παρουσιαστούν διάφορα στοιχεία στην ιστοσελίδα μας. Οι φυλλομετρητές δεν εμφανίζουν τα <tag>, αλλά τα χρησιμοποιούν για την ερμήνευση του περιεχομένου της σελίδας.

Η χρήση της HTML σε μια σελίδα στο διαδίκτυο επιτρέπει τον συνδυασμό πολλών δεδομένων, όπως εικόνες, βίντεο, κείμενο, ήχο κτλ. Συνήθως, μαζί με την χρήση της HTML γίνεται και χρήση της CSS, η οποία θα αναλυθεί παρακάτω, για την διαμόρφωση της σελίδας. Επιπλέον, η HTML παρέχει τις μεθόδους για την δημιουργία δομημένων εγγράφων. Δηλαδή εγγράφων που αποτελούνται από το περιεχόμενο που μεταφέρουν και από τον κώδικα μορφοποίησης του περιεχομένου που καθορίζει τα δομικά στοιχεία για το κείμενο, όπως είναι οι κεφαλίδες, οι παράγραφοι, οι λίστες, οι σύνδεσμοι, οι παραθέσεις και άλλα.

2.2.1 ΣΤΟΙΧΕΙΑ HTML

Κάθε αρχείο html περιέχει στοιχεία html, τα οποία αποτελούνται από τρία κομμάτια. Το πρώτο είναι ένα ζεύγος από ετικέτες, η «ετικέτα εκκίνησης» και η «ετικέτα τερματισμού», οι οποίες καθορίζουν τα όρια του στοιχείου. Επιπλέον, αποτελούνται από τις ιδιότητες που βρίσκονται μέσα στην ετικέτα εκκίνησης και τέλος το κείμενο ή το γραφικό περιεχόμενο μεταξύ των ετικετών, το οποίο μπορεί να περιλαμβάνει και άλλα στοιχεία εμφωλευμένα μέσα του.

Οι ιδιότητες που αναφέρονται στα στοιχεία html, είναι υπεύθυνα για να ορίσουν τον σκοπό του κειμένου, για παράδειγμα αν θα είναι επικεφαλίδα, τον τρόπο με τον οποίο εμφανίζεται το κείμενο αλλά και για την σύνδεση του αρχείου με άλλα αρχεία.

2.3 CSS

Η *css (Cascading Style Sheets)* είναι μια γλώσσα υπολογιστή που ανήκει στην κατηγορία των γλωσσών στυλ που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης για παράδειγμα της html. Ελέγχει δηλαδή την εμφάνιση ενός εγγράφου ή μιας ιστοσελίδας ή γενικότερα ενός ιστοτόπου που γράφτηκε στις γλώσσες HTML και XHTML. Πρόκειται για μια γλώσσα η οποία μπορεί να αλλάξει τα χαρακτηριστικά μιας σελίδας, όπως χρώμα, στοίχιση κτλ, έτσι ώστε να δημιουργηθεί μια καλοσχεδιασμένη ιστοσελίδα.

ΚΕΦΑΛΑΙΟ 3

3. ΕΦΑΡΜΟΓΗ

Η δομή της εφαρμογής είναι απλή. Αρχικά, υπάρχει μια βάση δεδομένων, περισσότερα για την βάση θα αναφερθούν παρακάτω, η οποία αλληλεπιδρά με το λογισμικό της εφαρμογής που βρίσκεται στον server. Στη συνέχεια, ο server αλληλεπιδρά με τον φυλλομετρητή της επιλογής του χρήστη και εμφανίζονται τα ανάλογα αποτελέσματα.

3.1 ΛΕΙΤΟΥΡΓΙΕΣ ΕΦΑΡΜΟΓΗΣ

Η εφαρμογή είναι ένας ιστότοπος, και αποτελείται από σελίδες, που η κάθε μια έχει μοναδική χρήση και λειτουργία. Ο κώδικας της εφαρμογής είναι βασισμένος στη python και την html και η βάση δεδομένων έχει βασιστεί στην mysql. Υπάρχουν δυο τύποι χρήστη, η επισκέπτες και οι εγγεγραμμένοι χρήστες. Και οι δυο κατηγορίες έχουν παρόμοια λειτουργικότητα και δικαιώματα. Μπορούν να περιηγηθούν στη σελίδα και να δούν το περιεχόμενο των σελίδων της εφαρμογής. Η διαφορά είναι ότι η εγγεγραμμένοι χρήστες, αυτοί δηλαδή που έχουν δώσει κάποια στοιχεία στην εφαρμογή προκειμένου να αποκτήσουν λογαριασμό, μπορούν να αναρτήσουν ερωτήσεις και απαντήσεις.

3.1.1 ΟΙ ΧΡΗΣΤΕΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Σημαντικό είναι να αναφερθεί ο τρόπος με τον οποίο μπορεί να γίνει κανείς εγγεγραμμένος χρήστης. Το μόνο που απαιτείται από τον χρήστη είναι ένα username, το οποίο δηλώνει την ταυτότητα του στη σελίδα, ένας κωδικός (password) και τέλος ένα email. Δίνεται και η δυνατότητα σύνδεσης με τον λογαριασμό του στο facebook. Με αυτόν τον τρόπο θα μπορεί να μοιράζεται ερωτήσεις και απαντήσεις, από την εφαρμογή, με τους χρήστες του facebook.

Την στιγμή που θα μπει στην εφαρμογή κάποιος χρήστης, εμφανίζεται η αρχική σελίδα. Εκείνη της στιγμή, οι χρήστες έχουν κάποιες επιλογές. Αρχικά, τους δίνεται η δυνατότητα να συνδεθούν με τον λογαριασμό τους στην εφαρμογή, αν έχουν, αλλιώς μπορούν να δημιουργήσουν έναν. Αν δεν επιθυμούν να δημιουργήσουν λογαριασμό, τότε μπορούν να προχωρήσουν στο περιεχόμενο της εφαρμογής.

Στη συνέχεια, αφού οι χρήστες πραγματοποιήσουν μια από τις επιλογές που αναφέρθηκαν, αποκτούν πρόσβαση στο περιεχόμενο της εφαρμογής. Υπάρχει μια σελίδα που περιέχει όλες τις αναρτήσεις των χρηστών σε μία λίστα. Εκεί ανάλογα με αντικείμενο που αναζητούν μπορούν να επιλέξουν το κατάλληλο άρθρο, και να δούν το περιεχόμενο του. Το περιεχόμενο του περιλαμβάνει το κύριο άρθρο και τα σχόλια, αν υπάρχουν, των χρηστών.

Αν οι χρήστες τις εφαρμογής επιθυμούν να δημιουργήσουν οι ίδιοι μια ερώτηση, η μόνη απαίτηση είναι, όπως έχει ήδη αναφερθεί, η ύπαρξη λογαριασμού. Μπορούν να πλοηγηθούν στην κατάλληλη

σελίδα δημιουργίας άρθρου και να γράψουν την ερώτηση τους. Στη συνέχεια, απλά πρέπει να περιμένουν έτσι ώστε κάποιος από τους χρήστες που γνωρίζει την λύση και είναι πρόθυμος να βοηθήσει, απαντήσει με την μορφή σχολίου.

Από την άλλη μεριά, αν κάποιος χρήστης αποφασίσει να απαντήσει σε κάποια ερώτηση, το μόνο που έχει να κάνει είναι να μεταφερθεί στην ανάλογη σελίδα και να γράψει το σχόλιο του. Φυσικά, προϋπόθεση είναι να είναι εγγεγραμμένος χρήστης.

Αξίζει να αναφερθεί ότι κάθε χρήστης έχει την δυνατότητα επεξεργασίας της ανάρτησης του. Με την πλοήγηση στην κατάλληλη σελίδα, μπορεί να επεξεργαστεί την ανάρτηση του, είτε σχόλιο είτε άρθρο, και να την διαγράψει αν το επιθυμεί.

Τέλος, μετά την χρήση της σελίδας, οι χρήστες μπορούν να κλείσουν την εφαρμογή από τον φυλλομετρητή τους. Οι εγγεγραμμένοι χρήστες, αν θέλουν να προφυλάξουν τον λογαριασμό τους στην εφαρμογή, μπορούν να κάνουν αποσύνδεση πρίν την έξοδο.

3.1.2 ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ

Ο σκοπός μιας βάσης δεδομένων είναι να αποθηκεύει δεδομένα για τις εφαρμογές που χρειάζονται αποθήκευση δεδομένων. Για την υλοποίηση των λειτουργιών της εφαρμογής, η ύπαρξη μιας βάσης δεδομένων είναι απαραίτητη. Για αυτό τον λόγο χρησιμοποιήθηκε μια MySQL βάση δεδομένων.

Αρχικά, για την δημιουργία της και για λόγους ασφάλειας χρησιμοποιήθηκαν username και password, έτσι ώστε να πραγματοποιηθεί η δημιουργία της. Στη συνέχεια, κατασκευάστηκαν με την χρήση των κατάλληλων sql εντολών, οι πίνακες (tables) που θα περιέχουν τα κατάλληλα δεδομένα.

3.1.2.1 ΠΙΝΑΚΕΣ

Για τον σωστό σχεδιασμό και την αποτελεσματική υλοποίηση της εφαρμογής και της βάσης δεδομένων κατασκευάστηκαν τρεις πίνακες αποθήκευσης δεδομένων. Ένας πίνακας περιέχει τα άρθρα των χρηστών, ένας τα σχόλια των χρηστών και ο τελευταίος περιέχει τους χρήστες και τις πληροφορίες τους.

Ο πρώτος πίνακας, αυτός που περιέχει τα άρθρα, περιέχει πέντε στήλες. Κάθε στήλη περιέχει το id του πίνακα, που είναι ένας μοναδικός αριθμός για κάθε άρθρο (primary key) και τα στοιχεία του κάθε άρθρου. Συγκεκριμένα, περιέχονται ο τίτλος του κάθε άρθρου, το όνομα του χρήστη που δημιούργησε το άρθρο, το περιεχόμενο του καθώς και η ημερομηνία δημιουργίας του. Τα δεδομένα αυτά είναι απαραίτητα για την σωστή και αποτελεσματική λειτουργία της εφαρμογής.

Ο δεύτερος πίνακας περιέχει τα απαραίτητα δεδομένα των σχολίων. Ο πίνακας αποτελείται από πέντε στήλες που περιέχουν το id του πίνακα, τον όνομα του χρήστη που δημιούργησε το σχόλιο, το περιεχόμενο του σχολίου, την ημερομηνία δημιουργίας του και το id του αντίστοιχου άρθρου. Όπως και στον πίνακα, κάθε σχόλιο έχει το δικό του id, το οποίο είναι και το primary key του πίνακα. Τα βασικά δεδομένα του σχολίου, δηλαδή το όνομα του δημιουργού, το περιεχόμενο και η ημερομηνία δημιουργίας έχουν την ίδια χρήση με τα βασικά στοιχεία του πρώτου πίνακα. Η κύρια διαφορά είναι η ύπαρξη μιας στήλης που περιέχει το id του άρθρου που αντιστοιχεί. Κάθε άρθρο έχει και τα δικά του σχόλια. Ο αριθμός αυτός ορίζει σε ποιο άρθρο αντιστοιχεί το κάθε σχόλιο.

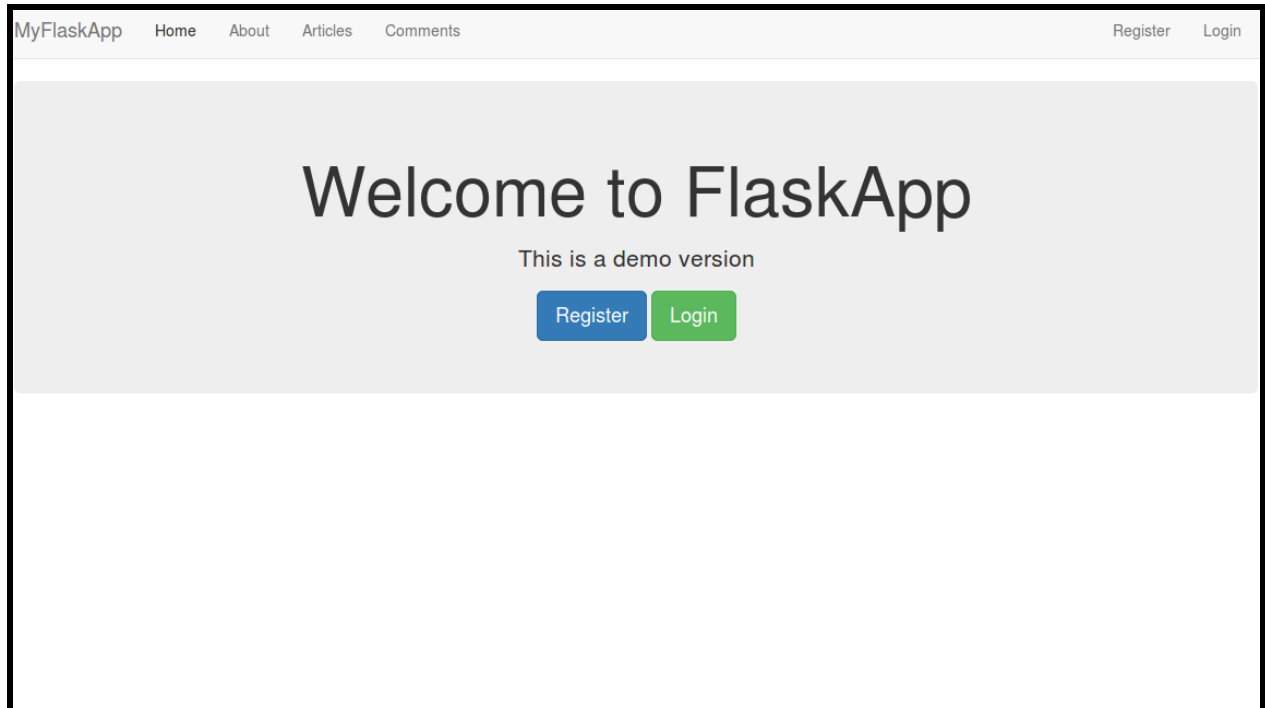
Ο τελευταίος πίνακας περιέχει τα δεδομένα των χρηστών. Συγκεκριμένα περιέχει το id των χρηστών (primary key), το όνομα τους, το email τους, το username, των κωδικό και την ημερομηνία εγγραφής τους στην εφαρμογή.

3.2 ΕΙΚΟΝΕΣ ΚΑΙ ΕΠΕΞΗΓΗΣΗ ΚΩΔΙΚΑ

Παρακάτω θα παρουσιαστούν εικόνες για επεξήγηση της λειτουργίας της εφαρμογής. Επιπλέον, θα γίνει αναφορά στον κώδικα των βασικών σελίδων της εφαρμογής.

3.2.1 ΑΡΧΙΚΗ ΣΕΛΙΔΑ

Όταν κάποιος επισκεφθεί την ιστοσελίδα, το πρώτο που αντικρίζει είναι η αρχική σελίδα (Home page) (ΕΙΚΟΝΑ 1). Εκεί αναφέρεται το όνομα της εφαρμογής και περιέχει και δυο κουμπιά, που είναι υπεύθυνα για την εγγραφή και την σύνδεση στην εφαρμογή. Επίσης, στην κορυφή της σελίδας υπάρχει μια μπάρα (navigation bar), που περιέχει κάποια κουμπιά για την πλοήγηση στην σελίδα.

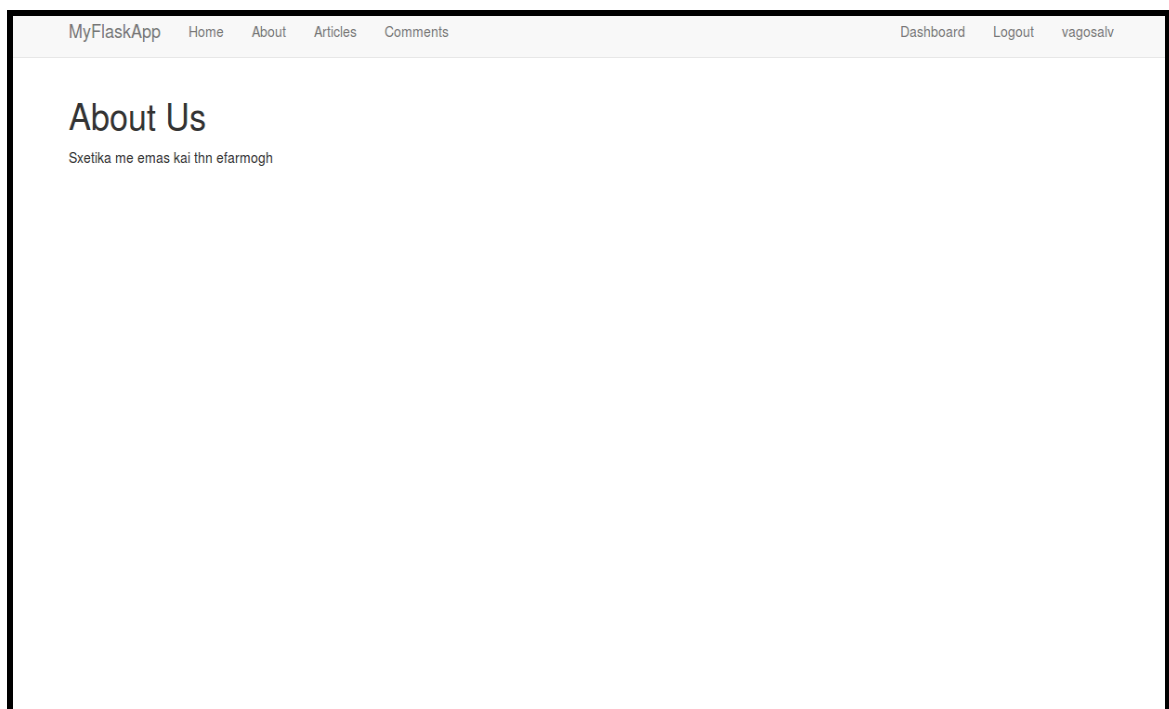


ΕΙΚΟΝΑ 1:ΑΡΧΙΚΗ ΣΕΛΙΔΑ

Συγκεκριμένα, αναγράφονται ένα κουμπί το οποίο οδηγεί στην αρχική σελίδα. Ένα κουμπί που αδηγεί σε μία σελίδα που αναφέρονται μερικές λεπτομέρειες για την εφαρμογή και δυο επιπλέον κουμπιά που οδηγούν στην λίστα με τα άρθρα και τα σχόλια που έχει κάνει ο χρήστης. Τέλος, στην άκρη της σελίδας υπάρχουν δυο κουμπιά για εγγραφή και σύνδεση και βρίσκονται εκεί σε περιπτώσεις όπου ο χρήστης, είναι σε κάποια άλλη σελίδα της εφαρμογής, και θέλει να πραγματοποιήσει μία από τις δυο ενέργειες.

3.2.2 ΣΧΕΤΙΚΑ ΜΕ ΕΜΑΣ

Στην επόμενη εικόνα (ΕΙΚΟΝΑ 2), παρουσιάζεται η σελίδα η οποία περιέχει μερικές πληροφορίες σχετικές με την εφαρμογή.



ΕΙΚΟΝΑ 2: ΣΧΕΤΙΚΑ ΜΕ ΕΜΑΣ

Επιπλέον, εδώ ο προγραμματιστής της σελίδας μπορεί να αναφέρει, εκτός από πληροφορίες για την σελίδα και πληροφορίες για τον ίδιο ή οτιδήποτε επιθυμεί και θεωρεί σημαντικό να αναφερθεί.

3.2.3 REGISTER PAGE

Στη συνέχεια, θα γίνει αναφορά στην σελίδα εγγραφής, register (ΕΙΚΟΝΑ 3). Οι χρήστες που θέλουν να δημιουργήσουν λογαριασμό, να εγγραφούν, στην εφαρμογή θα συναντήσουν αυτή τη σελίδα. Η σελίδα αυτή περιέχει κάποια πεδία τα οποία πρέπει να συμπληρωθούν από συγκεκριμένες πληροφορίες.



MyFlaskApp Home About Articles Comments Register Login

Register

Name

Email

Username

Password

Confirm Password

Submit

ΕΙΚΟΝΑ 3: REGISTER PAGE

Το πρώτο πεδίο που απαιτεί συμπλήρωση είναι το όνομα του χρήστη (name). Το πεδίο αυτό δεν έχει ιδιαίτερη σημασία στην λειτουργικότητα της σελίδας, και αποθηκεύεται για μελλοντική χρήση. Στη συνέχεια, ένα από τα σημαντικά πεδία που πρέπει να συμπληρωθούν είναι το Email. Το πεδίο αυτό είναι σημαντικό γιατί σε μελλοντική αναβάθμιση, θα αποστέλλεται ενημερωτικό μήνυμα σχετικά με τις δραστηριότητες της εφαρμογής.

Το επόμενο πεδίο, username, περιέχει το όνομα του χρήστη που εμφανίζει η εφαρμογή. Δηλαδή, στα άρθρα και στα σχόλια εμφανίζεται το περιεχόμενο του συγκεκριμένου πεδίου. Τα δυο τελευταία πεδία αφορούν την ασφάλεια των χρηστών και περιέχουν των κωδικό πρόσβασης, που έχει επιλέξει ο χρήστης. Το πρώτο πεδίο ορίζει τον κωδικό πρόσβασης και το δεύτερο ελέγχει αν ο χρήστης πληκτρολόγησε των κωδικό που ήθελε.

3.2.4 LOGIN

Σε περίπτωση όπου ο χρήστης έχει πραγματοποιήσει ήδη την εγγραφή του στην εφαρμογή και θέλει να συνδεθεί σε αυτήν, εμφανίζεται η σελίδα για την σύνδεση, login page (εικόνα 4).



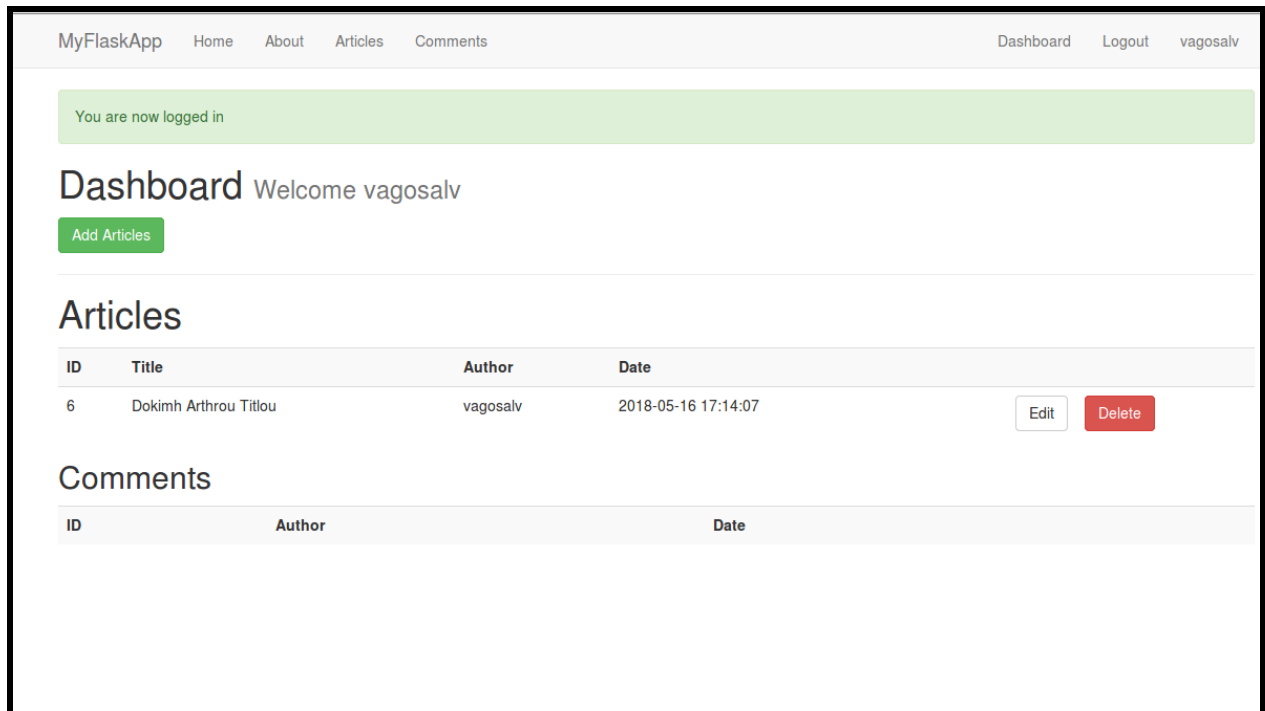
The image shows a web browser window displaying the login page of an application named 'MyFlaskApp'. The header bar is light gray and contains the application name on the left and navigation links ('Home', 'About', 'Articles', 'Comments', 'Register', 'Login') on the right. The main content area is white and features the title 'Login' in a large, dark font. Below the title, there are two input fields: one for 'Username' and one for 'Password'. Both fields are empty and have a light gray border. Below the password field, there is a blue button with the text 'Submit' in white. The entire page is framed by a thin black border.

ΕΙΚΟΝΑ 4: LOGIN PAGE

Σε αυτή τη σελίδα, ο χρήστης μπορεί να πραγματοποιήσει την σύνδεση του με τον λογαριασμό που διατηρεί στην εφαρμογή. Προκειμένου, να γίνει αυτό πρέπει να πληκτρολογήσει το username και το password που έχει επιλέξει. Σε περίπτωση λάθους σε ένα από τα δύο πεδίο εμφανίζεται κατάλληλο μήνυμα λάθους.

3.2.5 DASHBOARD ΚΑΙ ΔΗΜΙΟΥΡΓΙΑ ΑΡΘΡΟΥ

Όταν ο χρήστης συνδεθεί με επιτυχία στην εφαρμογή, μεταφέρεται στη σελίδα Dashboard (ΕΙΚΟΝΑ 5).

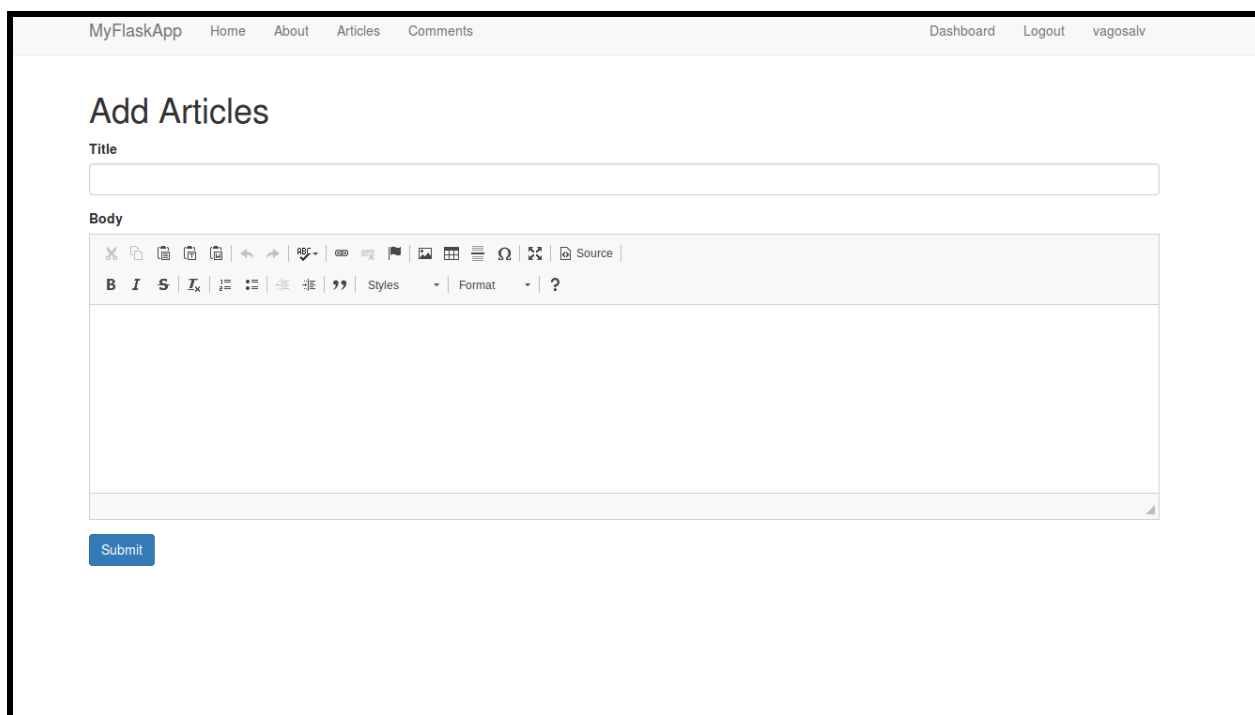


ΕΙΚΟΝΑ 5: DASHBOARD

Στην ουσία πρόκειται για μια σελίδα όπου εμφανίζει τις αναρτήσεις του χρήστη. Εδώ μπορεί να δει την κατάσταση των άρθρων και των σχολίων που έχει δημιουργήσει. Παράλληλα του δίνεται η δυνατότητα επεξεργασίας τους.

Επιπλέον, στη σελίδα αυτή υπάρχει ένα κουμπί του οποίου η χρήση είναι η δημιουργία καινούριου άρθρου. Με το πάτημα του

κουμπιού, ο χρήστης μεταφέρεται σε μια καινούρια σελίδα (ΕΙΚΟΝΑ 6), στην οποία μπορεί να δημιουργήσει ένα νέο άρθρο.



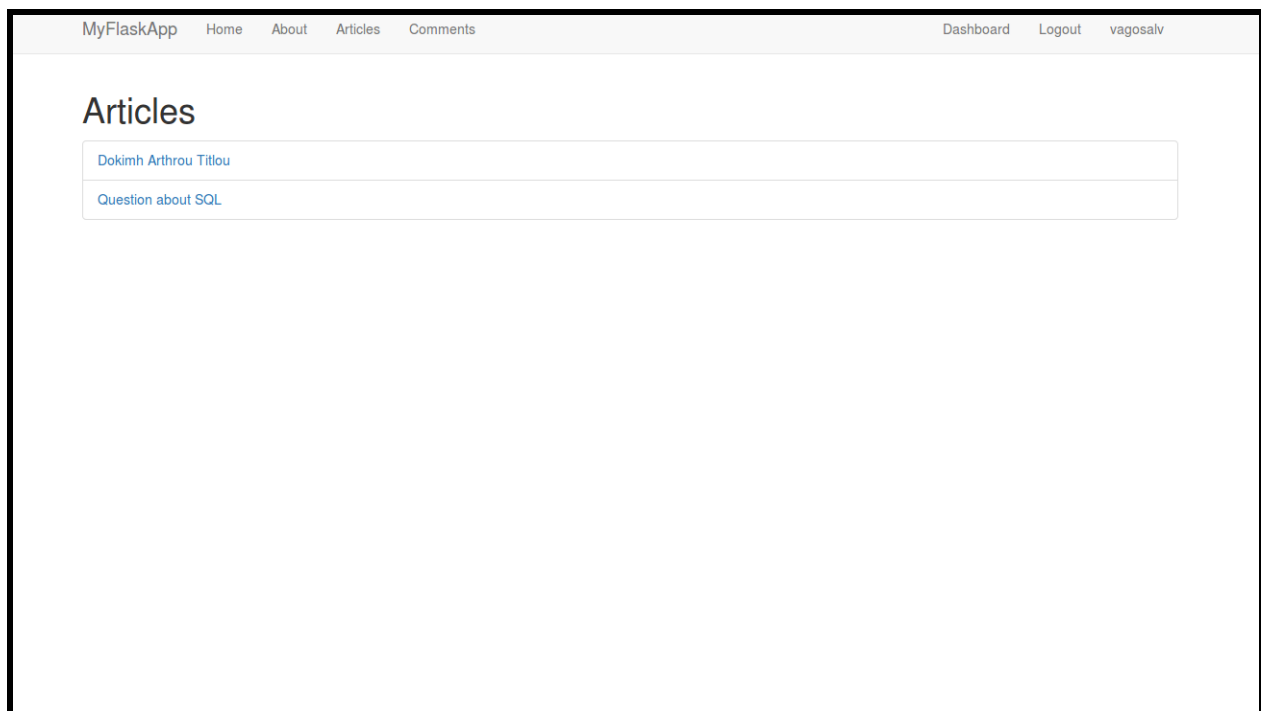
The screenshot shows a web application interface for adding articles. The header includes the application name 'MyFlaskApp' and navigation links: Home, About, Articles, Comments. On the right, there are links for Dashboard, Logout, and the user name 'vagosahv'. The main content area is titled 'Add Articles'. It features a 'Title' input field and a 'Body' text area. The 'Body' area is equipped with a rich text editor toolbar containing icons for bold, italic, strikethrough, text color, background color, bulleted list, numbered list, link, unlink, indent, outdent, and a 'Source' view toggle. Below the text area is a blue 'Submit' button.

ΕΙΚΟΝΑ 6: ΠΡΟΣΘΗΚΗ ΑΡΘΡΟΥ

Η νέα σελίδα που εμφανίζεται στον χρήστη, έχει πεδία για τον τίτλο του άρθρου και το κύριο μέρος του. Στον τίτλο, ο χρήστης βάζει το αντικείμενο που αφορά το άρθρο. Στο κύριο μέρος, συμπληρώνει τον λόγο που δημιουργίας του άρθρου. Επιπλέον, στο κύριο μέρος είναι διαθέσιμα και κάποια εργαλεία για την καλύτερη παρουσίαση του. Για παράδειγμα, όταν θέλει κάποιος χρήστης να αναρτήσει μια ερώτηση σχετική με ένα πρόβλημα που αντιμετώπισε κατά την δημιουργία ενός προγράμματος, στο κύριο μέρος μπορεί να βάλει το κομμάτι του κώδικα που τον δυσκολεύει. Τέλος, όταν τελειώσει με την δημιουργία του άρθρου, προκειμένου να αναρτηθεί, πατάει το κουμπί Submit.

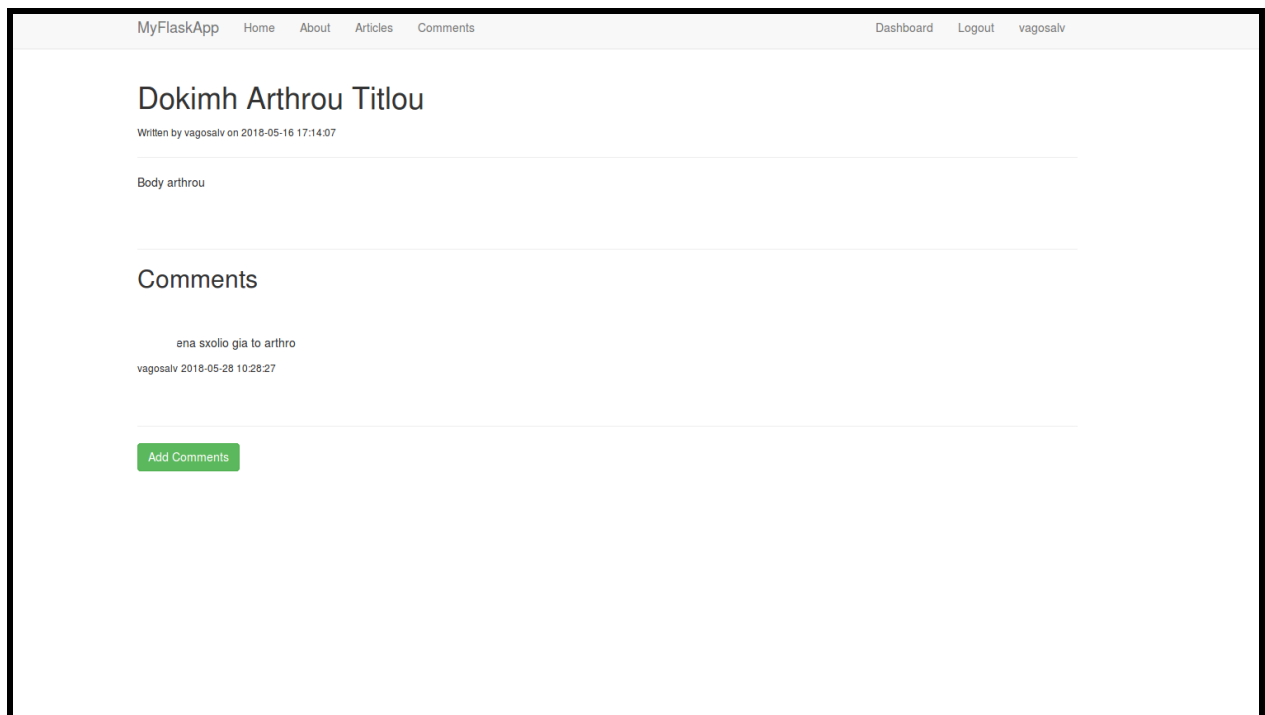
3.2.6 ΛΙΣΤΑ ΑΡΘΡΩΝ

Στη περίπτωση που ο χρήστης δεν θέλει να δημιουργήσει μια ανάρτηση, ή αν δεν είναι συνδεδεμένος χρήστης και θέλει να δει τη λίστα με τα άρθρα που είναι διαθέσιμα, τότε πατώντας το κουμπί από την επάνω μπάρα, μεταφέρεται στη σελίδα που περιέχει όλα τα άρθρα (ΕΙΚΟΝΑ 7).



ΕΙΚΟΝΑ 7: ΛΙΣΤΑ ΑΡΘΡΩΝ

Στη σελίδα αυτή, εμφανίζονται όλα τα άρθρα που είναι διαθέσιμα στην εφαρμογή. Ο τίτλος των άρθρων, αναφέρει και το τί περίπου θα περιέχει το άρθρο. Αφού βρεί ο χρήστης το κατάλληλο άρθρο, το επιλέγει και εμφανίζεται η σελίδα του άρθρου (ΕΙΚΟΝΑ 8).

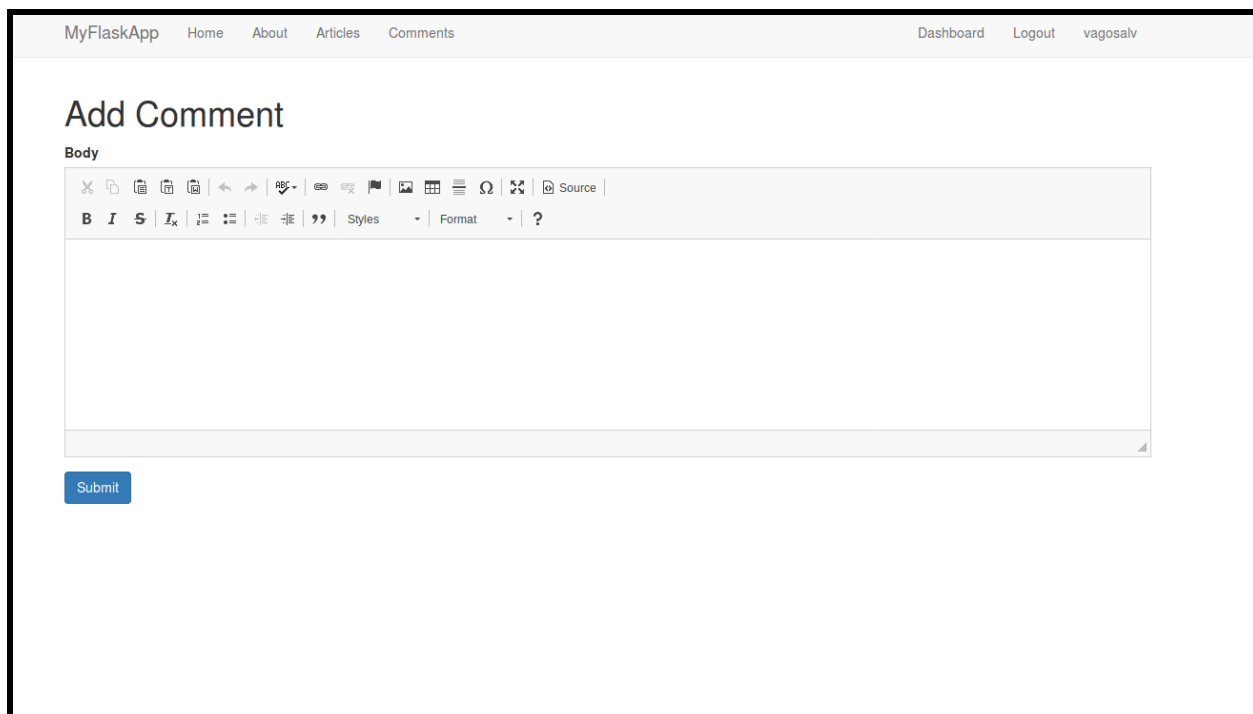


ΕΙΚΟΝΑ 8: ΕΣΩΤΕΡΙΚΟ ΤΟΥ ΑΡΘΡΟΥ

Με το άνοιγμα της σελίδας εμφανίζεται ο τίτλος του άρθρου μαζί με μερικές πληροφορίες. Πιο συγκεκριμένα, εμφανίζεται το username του δημιουργού του άρθρου και η ημερομηνία δημιουργίας του. Στη συνέχεια, εμφανίζεται το περιεχόμενο του άρθρου και πιο κάτω στη σελίδα υπάρχει η περιοχή για τα σχόλια. Εκεί, αναφέρεται το κύριο μέρος του σχολίου, το username του χρήστη που δημιούργησε το σχόλιο μαζί με την ημερομηνία δημιουργίας του. Στο τέλος της σελίδας υπάρχει ένα κουμπί που επιτρέπει την δημιουργία νέων σχολίων.

3.2.7 ΠΡΟΣΘΗΚΗ ΚΑΙ ΠΡΟΒΟΛΗ ΣΧΟΛΙΩΝ

Με το πάτημα του κουμπιού, ο χρήστης μεταφέρεται σε μια νέα σελίδα (ΕΙΚΟΝΑ 9), όπου μπορεί να γράψει τα δικά του σχόλια.



ΕΙΚΟΝΑ 9: ΔΗΜΙΟΥΡΓΙΑ ΣΧΟΛΙΩΝ

Στη σελίδα αυτή, ο χρήστης μπορεί να γράψει το δικό του σχόλιο στο κατάλληλο μέρος και στη συνέχεια, πατώντας το κατάλληλο κουμπί, να αναρτήσει το σχόλιο του.

Μια ακόμη λειτουργία που προσφέρει η εφαρμογή είναι η προβολή μιας λίστας που περιέχει όλα τα σχόλια που έχει κάνει ο χρήστης. Με την επιλογή της κατάλληλης επιλογής στην μπάρα περιήγησης, στην κορυφή της σελίδας, εμφανίζεται η σελίδα με την συγκεκριμένη λίστα (ΕΙΚΟΝΑ 10).

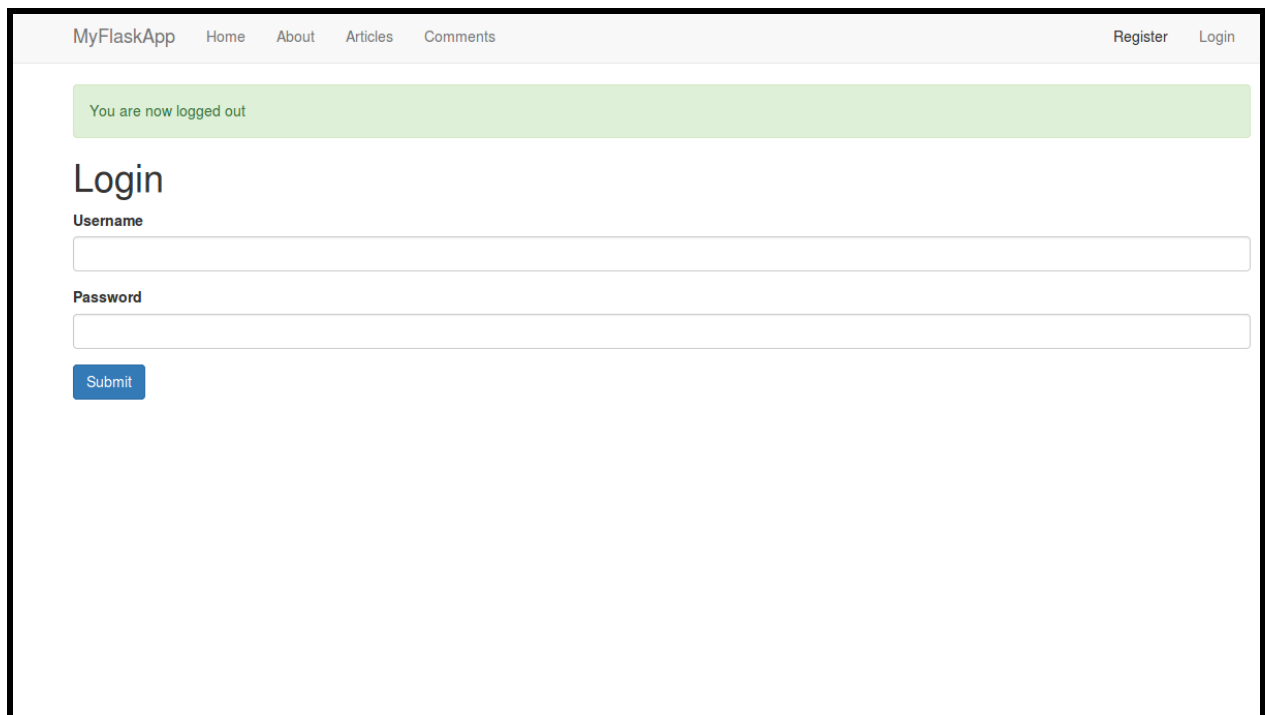
MyFlaskApp					Dashboard		Logout	vagosalv
Home	About	Articles	Comments					
ID	Author	Date	Body	Article_id				
11	vagosalv	2018-05-28 10:28:27	akoma ena sxolio gia to arthro	6	Edit	Delete		
12	vagosalv	2018-05-28 11:23:23	ssssxxxxoooooiiiiiooooo	6	Edit	Delete		

ΕΙΚΟΝΑ 10: ΛΙΣΤΑ ΜΕ ΤΑ ΣΧΟΛΙΑ

Στη σελίδα αυτή αναγράφονται όλα τα σχόλια που έχει κάνει ο ίδιος ο χρήστης, μαζί με τις κατάλληλες πληροφορίες. Πιο συγκεκριμένα, εμφανίζεται η ημερομηνία δημιουργίας του κάθε σχολίου αλλά και σε ποιο άρθρο αντιστοιχεί. Τέλος, προσφέρεται η δυνατότητα για την επεξεργασία ή την διαγραφή των σχολίων.

3.2.8 ΑΠΟΣΥΝΔΕΣΗ

Οι χρήστες που διαθέτουν λογαριασμό στη σελίδα, μετά το τέλος της πλοήγησης τους, τους δίνεται η δυνατότητα πριν κλείσουν την σελίδα, να αποσυνδεθούν από αυτή. Αυτή η ενέργεια επιτυγχάνεται με την επιλογή του κουμπιού Logout, που βρίσκεται στην μπάρα περιήγησης στον κορυφή των σελίδων. Με την επιλογή αυτή ο χρήστης μεταφέρεται στη σελίδα σύνδεσης αλλά αναφέρεται και το κατάλληλο μήνυμα επιτυχίας (ΕΙΚΟΝΑ 11).

The image is a screenshot of a web application interface. At the top, there is a navigation bar with the text 'MyFlaskApp' on the left and links for 'Home', 'About', 'Articles', and 'Comments' in the center. On the right side of the navigation bar are links for 'Register' and 'Login'. Below the navigation bar, a green banner displays the message 'You are now logged out'. Underneath this banner, the word 'Login' is prominently displayed. Below 'Login', there are two input fields: the first is labeled 'Username' and the second is labeled 'Password'. At the bottom of these fields is a blue button labeled 'Submit'.

ΕΙΚΟΝΑ 11: ΣΕΛΙΔΑ ΣΥΝΔΕΣΗΣ ΜΕ ΜΗΝΥΜΑ ΛΑΘΟΥΣ

Η διαδικασία αυτή, αν και δεν είναι αναγκαία, είναι χρήσιμη και αποτελεσματική σε θέματα ασφάλειας του χρήστη αλλά και της εφαρμογής.

3.3 ΚΩΔΙΚΑΣ

Κάποιες από τις γλώσσες προγραμματισμού που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής, όπως έχει αναφερθεί και παραπάνω, είναι η python και η html. Οι σελίδες που διαθέτει η εφαρμογή δημιουργήθηκαν με τη χρήση html, ενώ ο κώδικας που αφορά την λειτουργικότητα της με python. Παρακάτω θα γίνει αναφορά και θα δοθεί μια περιγραφή σχετικά με το πιο σημαντικό κομμάτι του κώδικα της σελίδας . Επιπρόσθετα, η για την διαδικασία της επεξήγησης του κώδικα θα εμφανίζονται και οι σχετικές εικόνες με το κάθε κομμάτι κώδικα.

Επιπλέον, αξίζει να αναφερθεί η δομή των αρχείων της εφαρμογής. Στον φάκελο της σελίδας υπάρχουν δυο υποφάκελοι, οι οποίοι περιέχουν ο ένας, templates, τα html αρχεία και ο άλλος, views, τα αρχεία python. Στον φάκελο templates υπάρχει ένας ακόμα υποφάκελος, includes, ο οποίος περιέχει αρχεία html που αφορούν τα μηνύματα που εμφανίζονται στην εφαρμογή αλλά και το navigation bar , που περιέχει τον κώδικα για το menu της εφαρμογής.

Αξίζει να διευκρινιστεί ότι, δεν θα γίνει επεξήγηση ολόκληρου του κώδικα της εφαρμογής, αλλά θα αναλυθούν τα πιο κρίσιμα κομμάτια της σελίδας.

3.3.1 LAYOUT

Το αρχείο αυτό περιέχει κώδικα που αφορά την μορφοποίηση και την λειτουργικότητα της σελίδας. Όπως φαίνεται και στην εικόνα 12 παρακάτω, το αρχείο αυτό καθορίζει τον τίτλο της εφαρμογής αλλά και την μορφοποίηση.

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8">
5   <title>MyFlaskApp</title>
6   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
7 </head>
8 <body>
9   {% include 'includes/_navbar.html' %}
10  <div class="container">
11    {% include 'includes/_messages.html' %}
12    {% block body %}{% endblock %}
13  </div>
14  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
15  <script src="//cdn.ckeditor.com/4.7.3/standard/ckeditor.js"></script>
16  <script type="text/javascript">
17    CKEDITOR.replace('editor')
18  </script>
19 </body>
20 </html>
```

ΕΙΚΟΝΑ 12: Layout.html

Πιο συγκεκριμένα, στο tag link (γραμμή 6) περιέχει τον σύνδεσμο που περιέχει το αρχείο css, το οποίο μορφοποιεί την εφαρμογή. Επιπλέον, στις γραμμές 14, 15, 16 περιλαμβάνονται, στα κατάλληλα tags, ο σύνδεσμος για την απαραίτητη javascript που απαιτείται για την σωστή λειτουργία της css.

Ο κώδικας που αναφέρεται στις γραμμές 9 και 11 έχει ως σκοπό να περιλαμβάνει τον κώδικα που βρίσκεται σε κάθε αρχείο και αφορά το menu της εφαρμογής (navigation bar) και τα σχόλια σε περίπτωση λάθους σύνδεσης.

3.3.2 ΑΡΧΙΚΗ ΣΕΛΙΔΑ

Ο κώδικας της αρχικής σελίδας περιέχει έναν τίτλο και τα κουμπιά για την σύνδεση και την εγγραφή.

```
1  {% extends 'layout.html' %}
2
3  {% block body %}
4  <div class="jumbotron text-center">
5      <h1>Welcome to FlaskApp</h1>
6      <p class="lead">This is a demo version</p>
7      {% if session.logged_in == NULL %}
8          <a href="/p13alva/register" class="btn btn-primary btn-lg">Register</a>
9          <a href="/p13alva/login" class="btn btn-success btn-lg">Login</a>
10     {% endif%}
11 </div>
12 {% endblock %}
13
```

ΕΙΚΟΝΑ 13: Home.html

Αξίζει να σημειωθεί ότι περιλαμβάνει το αρχείο html που αναφέρθηκε πιο πάνω (layout.html) με σκοπό την χρήση του αρχείου για τα μηνύματα λάθους κατά την σύνδεση και εγγραφή.

3.3.3 USERS

Τα αρχεία html που αφορούν την εγγραφή και την σύνδεση δεν έχουν κάποια ιδιαιτερότητα και για αυτό τον λόγο δεν θα γίνει αναφορά σε αυτά. Θα γίνει αναφορά όμως στο rpython αρχείο που αφορά την λειτουργικότητα αυτών των σελίδων. Εξαιτίας του μεγάλου όγκου του κώδικα, κρίθηκε αναγκαία η τμηματοποίηση του κώδικα σε ξεχωριστές εικόνες, για την καλύτερη επεξήγηση του.

```

1  from flask import Blueprint, Flask, render_template, flash, url_for, session, request, logging, redirect
2  from wtforms import Form, StringField, TextAreaField, PasswordField, validators
3  from passlib.hash import sha256_crypt
4  from functools import wraps
5  from flask_mysqldb import MySQL
6  from ..app import mysql
7
8
9  us = Blueprint('us', __name__, template_folder='templates')
10
11  #klash gia elenxo ths formas
12  class RegisterForm(Form):
13      name = StringField('Name', [validators.Length(min=1, max=50)])
14      username = StringField('Username', [validators.Length(min=4, max=25)])
15      email = StringField('Email', [validators.Length(min=6, max=50)])
16      password = PasswordField('Password', [
17          validators.DataRequired(),
18          validators.EqualTo('confirm', message='Passwords do not match')
19      ])
20      confirm = PasswordField('Confirm Password')
21
22

```

EIKONA 13.1: Users.py

Στην αρχή του κώδικα (εικόνα 13.1) φαίνονται τα imports που χρειάστηκαν και στη συνέχεια γίνεται η δήλωση του blueprint. Στη συνέχεια, υπάρχει μια κλάση σχετική με την εγγραφή των χρηστών. Η κλάση αυτή περιλαμβάνει ελέγχους για την ορθότητα των στοιχείων που πληκτρολογεί ο χρήστης καθώς και κάποιους περιορισμούς και συγκεκριμένα πεδία (μήκος).

```

23 #User register
24 @us.route('/register', methods=['GET', 'POST'])
25 def register():
26     form = RegisterForm(request.form)
27     if request.method == 'POST' and form.validate():
28         name = form.name.data
29         email = form.email.data
30         username = form.username.data
31         password = sha256_crypt.encrypt(str(form.password.data))
32
33         # Create cursor
34         c = mysql.db.cursor()
35
36         # Execute query
37         c.execute("INSERT INTO users(name, email, username, password) VALUES(%s, %s, %s, %s)", (name, email, username, password))
38
39         # Commit to DB
40         mysql.db.commit()
41
42         #Close connection
43         c.close()
44
45         flash ('You are now registered and can log in', 'Success')
46
47         redirect(url_for('us.index_page'))
48
49         return redirect(url_for('us.login'))
50     return render_template('register.html', form = form)
51
52

```

EIKONA 13.2: Users.py

Στη συνέχεια, υπάρχει μια συνάρτηση για την εισαγωγή των στοιχείων εγγραφής του χρήστη στην βάση δεδομένων. Αρχικά, εισάγονται σε μεταβλητές τα δεδομένα του χρήστη (γραμμή 28-31), γίνεται η ετοιμασία για την εισαγωγή των στοιχείων στη βάση δεδομένων (γραμμή 37) και στο τέλος πραγματοποιείται η εισαγωγή στην βάση (γραμμή 40). Στο τέλος, εμφανίζεται το κατάλληλο μήνυμα επιτυχίας (γραμμή 45) και γίνεται ανακατεύθυνση (γραμμή 47).

```

54 #User login
55 @us.route('/login', methods=['GET', 'POST'])
56 def login():
57     if request.method == 'POST':
58         #Get form fields
59         username = request.form['username']
60         password_candidate = request.form['password']
61         #Create cursor
62         c = mysql.db.cursor()
63         #Get user by username
64         result = c.execute("SELECT * FROM users WHERE username = %s" , [username])
65
66         if result > 0:
67             #Get stored hash
68             data = c.fetchone()
69             password = data[4]#htan 'password'
70
71             #Compare Passwords
72             if sha256_crypt.verify(password_candidate, password):
73                 #Passed
74                 session['logged_in'] = True
75                 session['username'] = username
76
77                 flash('You are now logged in', 'success')
78                 return redirect(url_for('hello.dashboard'))
79             else:
80                 error = 'Invalid login'
81                 return render_template('login.html', error=error )
82             #close connection
83             c.close()
84         else:
85             error = 'Username not found'
86             return render_template('login.html', error=error )
87
88
89     return render_template('login.html')
90

```

ΕΙΚΟΝΑ 13.3: Users.py

Η διαδικασία της σύνδεσης του χρήστη στην εφαρμογή περιλαμβάνει μια διαδικασία ελέγχου. Αυτή η διαδικασία περιλαμβάνει τον κώδικα που αναγράφεται στην εικόνα 13.3. Αρχικά, όταν ο χρήστης εισάγει στην σελίδα login τα στοιχεία του (γραμμή 59,60), γίνεται έλεγχος στην βάση δεδομένων για το username, δηλαδή αν είναι εγγεγραμμένος χρήστης (γραμμή 64). Αν βρεθεί αποτέλεσμα (γραμμή 66) τότε γίνεται έλεγχος του κωδικού πρόσβασης

(γραμμή 72). Αν είναι σωστός τότε ο χρήστης μεταφέρεται στην κατάλληλη σελίδα (γραμμή 78), αλλιώς μπορεί να ξαναπροσπαθήσει (γραμμή 81). Σε κάθε περίπτωση εμφανίζεται το κατάλληλο μήνυμα, είτε επιτυχίας είτε αποτυχίας.

```
92 #Check if user is logged_in (snippet)
93 def is_logged_in(f):
94     @wraps(f)
95     def wrap(*args, **kwargs):
96         if 'logged_in' in session:
97             return f(*args, **kwargs)
98         else:
99             flash('Unauthorized, Please login', 'danger')
100             return redirect(url_for('us.login'))
101     return wrap
102
103
104 #Logout
105 @us.route('/logout')
106 @is_logged_in
107 def logout():
108     session.clear()
109     flash('You are now logged out', 'success')
110     return redirect(url_for('us.login'))
111
```

ΕΙΚΟΝΑ 13.4: Users.py

Τέλος, υπάρχουν δυο ακόμα συναρτήσεις. Η πρώτη (γραμμή 93) είναι υπεύθυνη για τον έλεγχο της σύνδεσης αν δηλαδή ο χρήστης είναι συνδεδεμένος στην εφαρμογή. Σε περίπτωση που δεν είναι τότε εμφανίζεται το κατάλληλο μήνυμα λάθους(γραμμή 99) και γίνεται ανακατεύθυνση στην σελίδα σύνδεσης (login). Η συγκεκριμένη συνάρτηση είναι σημαντική για λόγους ασφαλείας καθώς επιτρέπει

μόνο τους εγγεγραμμένους χρήστες να δουν το περιεχόμενο της εφαρμογής.

Η τελευταία συνάρτηση είναι υπεύθυνη για την αποσύνδεση του χρήστη από την εφαρμογή. Πρώτα καλεί την συνάρτηση που αναφέρθηκε πριν (γραμμή 106) και μετά αποσυνδέει τον χρήστη από την σελίδα, εμφανίζοντας το κατάλληλο μήνυμα και τον μεταφέρει στην σελίδα σύνδεσης (login).

3.3.4 ARTICLES

Ένα από τα βασικά κομμάτια της εφαρμογής είναι αυτά που αφορούν τα άρθρα (articles). Παρακάτω θα γίνει αναφορά στον τρόπο με τον οποίο λειτουργούν τα άρθρα (εισαγωγή, επεξεργασία, διαγραφή) και στην συνέχεια ο τρόπος που εμφανίζονται.

```
1 from flask import Blueprint, Flask, render_template, flash, url_for, session, request, logging, redirect
2 from wtforms import Form, StringField, TextAreaField, PasswordField, validators
3 from passlib.hash import sha256_crypt
4 from functools import wraps
5 from flask_mysqldb import MySQL
6 from ..app import mysql
7 from .users import is_logged_in
8 #twra to evala
9 from .comments import *
10
11 art = Blueprint('art', __name__, template_folder='templates')
12
13
14 class ArticleForm(Form):
15     title = StringField('Title', [validators.Length(min=1, max=200)])
16     body = TextAreaField('Body', [validators.Length(min=10)])
17
18 #Articles
19 @art.route('/articles')
20 def articles():
21     #Create cursor
22     c = mysql.db.cursor()
23     #Get Articles
24     result = c.execute("SELECT * FROM articles")
25     articles = c.fetchall()
26     if result > 0 :
27         return render_template('articles.html', articles=articles)
28     else:
29         msg = 'No articles Found'
30         return render_template('articles.html', msg=msg)
31     #close connection
32     c.close()
33
34 # Gia otan anoigw to article na emfanizei to swsto perioxomeno
35 @art.route('/article/<string:id>/')
36 def article(id):
37     #Create cursor
38     c = mysql.db.cursor()
39     #Get Article
40     result = c.execute("SELECT * FROM comments WHERE article_id = %s", [id])
41     if result > 0 :
42         result = c.execute("SELECT articles.* , comments.* FROM articles, comments WHERE articles.id = %s and comments.article_id = articles.id ", [id])
43     else:
44         result = c.execute("SELECT * FROM articles WHERE id = %s", [id])
45
46     #Commit
47     article = c.fetchone()
48     return render_template('test.html', article=article)
49
```

EIKONA 14.1: articles.py

Στην εικόνα 14 φαίνεται ο κώδικας του αρχείου articles.py. Στην αρχή γίνονται οι εισαγωγές (imports) των κατάλληλων αρχείων για την

σωστή λειτουργία του κώδικα. Παρακάτω γίνεται η δήλωση του blueprint (γραμμή 11) και υπάρχει η κλάση (γραμμή 14) με την δομή του κάθε άρθρου. Κάθε άρθρο έχει τίτλο και κύριο μέρος, αλλά και περιορισμούς.

Στη συνέχεια, υπάρχει η συνάρτηση που είναι υπεύθυνη για την εμφάνιση όλων των άρθρων (γραμμή 18). Συγκεκριμένα, συνδέεται με την βάση δεδομένων (γραμμή 22-25) και ελέγχει το περιεχόμενο του σχετικού πίνακα. Αν ο πίνακας έχει δεδομένα τα εμφανίζει (γραμμή 27) αλλιώς εμφανίζει το κατάλληλο μήνυμα (γραμμή 29).

Στην περίπτωση όπου υπάρχουν άρθρα στην σελίδα και ο χρήστης επιλέξει ένα από αυτά τότε πρέπει να εμφανιστεί το σωστό άρθρο. Την απαίτηση αυτή την πραγματοποιεί η επόμενη συνάρτηση (γραμμή 34). Το κάθε άρθρο όταν εισάγεται στην βάση δεδομένων λαμβάνει έναν μοναδικό αριθμό (id). Οπότε η συνάρτηση χρησιμοποιεί αυτόν τον αριθμό για την αναγνώριση του σωστού άρθρου (γραμμή 40). Αν υπάρχει στην βάση δεδομένων άρθρο με τον συγκεκριμένο αριθμό τότε το εμφανίζει μαζί με τα σχόλια του άρθρου (γραμμή 42), αν υπάρχουν.

```

51 #για οταν ανοιγω το article na emfanizontai ta swsta comments
52 @art.route('/comment/<string:id/>')
53 def comment(id):
54     #Create cursor
55     c = mysql.db.cursor()
56     #Get Comment
57     result = c.execute("SELECT * FROM comments WHERE id = %s", [id])
58     comment = c.fetchone()
59     return render_template('comment.html', comment=comment)
60
61
62     #Add article
63 @art.route('/add_article', methods=['GET', 'POST'])
64 @is_logged_in
65 def add_article():
66     form = ArticleForm(request.form)
67     if request.method == 'POST' and form.validate():
68         title = form.title.data
69         body = form.body.data
70         #Create cursor
71         c = mysql.db.cursor()
72         #execute
73         c.execute("INSERT INTO articles(title, body, author) VALUES(%s, %s, %s)", (title, body, session['username']))
74         #commit
75         mysql.db.commit()
76         #close connection
77         c.close()
78         flash('Article created', 'success')
79         return redirect(url_for('hello.dashboard'))
80
81     return render_template('add_article.html', form=form)
82

```

EIKONA 14.2: articles.py

Επιπλέον, στο αρχείο περιλαμβάνεται και η συνάρτηση για την προσθήκη άρθρου. Αρχικά, καλείται η συνάρτηση για το αν ο χρήστης είναι συνδεδεμένος στην σελίδα (γραμμή 64) και να είναι τότε μπορεί να προσθέσει άρθρο. Στη συνέχεια, εισάγονται σε μεταβλητές ο τίτλος και το περιεχόμενο του άρθρου (γραμμή 68,69) και τοποθετούνται στην βάση δεδομένων. (γραμμή 70-77). Αν η διαδικασία αυτή ολοκληρώθηκε με επιτυχία τότε εμφανίζεται το κατάλληλο μήνυμα επιτυχίας (γραμμή 78).

```

83 #Edit article
84 @art.route('/edit_article/<string:id>', methods=['GET', 'POST'])
85 @is_logged_in
86 def edit_article(id):
87     #create cursor
88     c = mysql.db.cursor()
89     #get article by id
90     result = c.execute("SELECT * FROM articles WHERE id = %s",[id])
91     article = c.fetchone()
92     #Get form
93     form = ArticleForm(request.form)
94     #populate article form fields
95     form.title.data = article[1]
96     form.body.data = article[3]
97
98     if request.method == 'POST' and form.validate():
99         title = request.form['title']
100         body = request.form['body']
101         #Create cursor
102         c = mysql.db.cursor()
103         #execute
104         c.execute("UPDATE articles SET title=%s, body=%s WHERE id = %s",(title, body, id))
105         #commit
106         mysql.db.commit()
107         #close connection
108         c.close()
109         flash('Article Updated', 'success')
110         return redirect(url_for('hello.dashboard'))
111
112     return render_template('edit_article.html', form=form)
113
114
115 #Delete article
116 @art.route('/delete_article/<string:id>', methods=['POST'])
117 @is_logged_in
118 def delete_article(id):
119     #create cursor
120     c = mysql.db.cursor()
121     #execute
122     c.execute("DELETE FROM articles WHERE id = %s", [id])
123     #commit
124     mysql.db.commit()
125     #close connection
126     c.close()
127     flash('Article Deleted', 'success')
128     return redirect(url_for('hello.dashboard'))
129

```

EIKONA 14.3: articles.py

Ο χρήστης, όπως έχει ήδη αναφερθεί, έχει τη δυνατότητα επεξεργασίας του κάθε άρθρου που έχει ο ίδιος αναρτήσει στη σελίδα. Η επόμενη συνάρτηση (edit_article, γραμμή 83) δίνει στον χρήστη αυτή την δυνατότητα. Αρχικά, καλείται η συνάρτηση is_logged_in που έχει αναφερθεί παραπάνω και στη συνέχεια αρχίζει το κύριο μέρος της

συνάρτησης. Πρώτα, πρέπει να γίνει η επιλογή του σωστού άρθρου και να έρθουν από την βάση δεδομένων το περιεχόμενο του, τίτλος και κύριο μέρος (γραμμή 87-96), ώστε να συμπληρωθεί η φόρμα για την επεξεργασία του άρθρου. Αν ο χρήστης κάνει κάποια αλλαγή στο περιεχόμενο τότε, εισάγονται στην βάση δεδομένων (γραμμή 99-108) με τον ίδιο τρόπο που έγινε και στην δημιουργία του άρθρου.

Τέλος, η εφαρμογή προσφέρει και την δυνατότητα διαγραφής του άρθρου στον χρήστη. Στην τελευταία συνάρτηση, αφού πρώτα γίνει ο έλεγχος για το αν ο χρήστης είναι συνδεδεμένος, τότε μπορεί με την κλήση της συνάρτησης να διαγράψει το άρθρο. Το id του άρθρου καλείται από την βάση δεδομένων και αν υπάρχει τότε διαγράφεται από την βάση.

Αξίζει να αναφερθεί ο τρόπος με τον οποίο εμφανίζονται τα άρθρα και πιο συγκεκριμένα πως εμφανίζεται ένα συγκεκριμένο άρθρο. Η εμφάνιση της λίστας όλων είναι απλή διαδικασία καθώς εμφανίζεται το περιεχόμενο του πίνακα της βάσης δεδομένων που περιέχει όλα τα άρθρα.

Τέλος, για την επεξεργασία και την διαγραφή των άρθρων ελέγχεται αν ο συνδεδεμένος χρήστης στη σελίδα είναι και ο συγγραφέας του άρθρου. Παρακάτω παρουσιάζεται ο τρόπος με τον οποίο εμφανίζεται το κάθε άρθρο.

```

1  {% extends 'layout.html' %}
2
3  {% block body %}
4
5  <h1>{{article[1]}}</h1>
6  <small>Written by {{article[2]}} on {{article[4]}}</small>
7  <hr>
8  <div>
9      {{article[3] | safe}}
10
11 </div>
12 <br></br>
13 <hr>
14 <h2>Comments</h2>
15
16 <br></br>
17 <div>
18     {{article[7] | safe}}
19     <small>{{article[6] | safe}}</small>
20     <small>{{article[8] | safe}}</small>
21 </div>
22 <br></br>
23
24 <hr>
25 <a class="btn btn-success" href="/p13alva/add_comments/{{article[0]}}"> Add Comments</a>
26 <a class="btn btn-success" href="/p13alva/comments"> Show Comments</a>
27
28 {% endblock %}
29

```

EIKONA 15: articles.html

Όπως έχει ήδη αναφερθεί παραπάνω, κάθε άρθρο έχει τίτλο και περιεχόμενο. Ο πίνακας στη βάση δεδομένων αποθηκεύει και κάποια άλλα χαρακτηριστικά όπως το όνομα του συγγραφέα και η ημερομηνία δημιουργίας του άρθρου. Στην εικόνα 15 παρατηρείται πως αρχικά εμφανίζεται ο τίτλος του άρθρου (γραμμή 5) και μετά το όνομα του συγγραφέα και η ημερομηνία δημοσίευσης του (γραμμή 6). Τέλος, εμφανίζεται και το κύριο μέρος του άρθρου (γραμμή 9).

Σε περίπτωση που υπάρχουν και σχόλια για το συγκεκριμένο άρθρο, εμφανίζονται στο κάτω μέρος της σελίδας μαζί με τις απαραίτητες πληροφορίες, όπως όνομα συγγραφέα και ημερομηνία ανάρτησης. Στο τέλος, υπάρχει ένα κουμπί για την εισαγωγή νέων σχολίων.

3.3.5 COMMENTS

Ο κάθε χρήστης μπορεί να δει τα σχόλια που ο ίδιος έχει κάνει σε μία λίστα στη σελίδα. Εκεί βλέπει πληροφορίες σχετικά με το περιεχόμενο του σχολίου και σε ποιο άρθρο αντιστοιχεί. Επιπλέον, του δίνεται η δυνατότητα επεξεργασίας ή ακόμα διαγραφής των δικών του σχολίων. Παρακάτω θα γίνει αναφορά και θα δοθεί επεξήγηση σχετικά με τον κώδικα που του προσφέρει αυτές τις δυνατότητες καθώς και της λίστας των άρθρων της εφαρμογής.

```

1  from flask import Blueprint, Flask, render_template, flash, url_for, session, request, logging, redirect
2  from wtforms import Form, StringField, TextAreaField, PasswordField, validators
3  from passlib.hash import sha256_crypt
4  from functools import wraps
5  from flask_mysql import MySQL
6  from ..app import mysql
7  from .users import is_logged_in
8
9
10 com = Blueprint('com', __name__, template_folder='templates')
11
12 #Comments Form class
13 class CommentForm(Form):
14     body = TextAreaField('Body', [validators.Length(min=10)])
15
16 #Comments
17 @com.route('/comments')
18 def comments():
19     #Create cursor
20     c = mysql.db.cursor()
21     #Get Articles
22     result = c.execute("SELECT * FROM comments")
23     comments = c.fetchall()
24     if result > 0 :
25         return render_template('comments.html', comments=comments)
26     else:
27         msg = 'No comments Found'
28         return render_template('comments.html', msg=msg)
29     #close connection
30     c.close()
31

```

ΕΙΚΟΝΑ 16.1: comments.py

Όπως φαίνεται και στην εικόνα 16.1 στην αρχή γίνονται οι εισαγωγές (imports) των κατάλληλων αρχείων για την σωστή λειτουργία του κώδικα και γίνεται η δήλωση του blueprint (γραμμή 10). Η εμφάνιση της λίστας των σχολίων της εφαρμογής επιτυγχάνεται με την συνάρτηση comments (γραμμή 18), όπου πραγματοποιείται έλεγχος στην βάση δεδομένων της εφαρμογής για την ύπαρξη σχολίων. Το αποτέλεσμα της συνάρτησης είναι η εμφάνιση των σχολίων, σε περίπτωση που αυτά βρεθούν στην βάση δεδομένων ή η εμφάνιση του κατάλληλου μηνύματος σε αντίθετη περίπτωση.

```

35 #Add comment
36 @com.route('/add_comments/<string:article_id>', methods=['GET', 'POST'])
37 @is_logged_in
38 def add_comment(article_id):
39
40     form = CommentForm(request.form)
41     if request.method == 'POST' and form.validate():
42         body = form.body.data
43         #Create cursor
44         c = mysql.db.cursor()
45         #execute
46         c.execute("INSERT INTO comments(author, body, article_id) VALUES(%s, %s, %s)", (session['username'], body, [article_id]))
47         #commit
48         mysql.db.commit()
49         #close connection
50         c.close()
51         flash('Comment created', 'success')
52         return redirect(url_for('hello.dashboard'))
53
54     return render_template('add_comments.html', form=form)
55
56
57 #Edit comment
58 @com.route('/edit_comment/<string:id>', methods=['GET', 'POST'])
59 @is_logged_in
60 def edit_comment(id):
61     #create cursor
62     c = mysql.db.cursor()
63     #get comment by id
64     result = c.execute("SELECT * FROM comments WHERE id = %s",[id])
65     comment = c.fetchone()
66     #Get form
67     form = CommentForm(request.form)
68     #populate comment form fields
69     form.body.data = comment['body']
70
71     if request.method == 'POST' and form.validate():
72         body = request.form['body']
73         #Create cursor
74         c = mysql.db.cursor()
75         #execute
76         c.execute("UPDATE comments SET body=%s WHERE id =%s", (body, id))
77         #commit
78         mysql.db.commit()
79         #close connection
80         c.close()
81         flash('Comment Updated', 'success')
82         return redirect(url_for('hello.dashboard'))
83
84     return render_template('edit_comment.html', form=form)

```

EIKONA 16.2: comments.py

Κατά την διαδικασία προβολής ενός συγκεκριμένου άρθρου, ο χρήστης με την επιλογή του κατάλληλου κουμπιού μπορεί να προσθέσει το δικό του σχόλιο. Στην εικόνα 16.2 φαίνεται η συνάρτηση (γραμμή 60), η οποία εκπληρώνει αυτή την απαίτηση της εφαρμογής. Αρχικά, γίνεται έλεγχος έτσι ώστε να διαπιστωθεί ότι ο χρήστης είναι συνδεδεμένος (γραμμή 50) και στην συνέχεια το περιεχόμενο του άρθρου, το κύριο μέρος δηλαδή, εισέρχεται σε μία μεταβλητή (γραμμή

42), με σκοπό την εισαγωγή του στην βάση δεδομένων. Στο τέλος, ο χρήστης ενημερώνεται για την επιτυχία αυτής της ενέργειας.

Όπως αναφέρθηκε και πιο πάνω, ο χρήστης μπορεί να επεξεργαστεί το σχόλιο που έχει κάνει ο ίδιος και αυτή την ανάγκη την ικανοποιεί η δεύτερη συνάρτηση της εικόνας 16.2 (γραμμή 57). Αφού γίνει ο έλεγχος για το αν ο χρήστης που κάνει την ενέργεια είναι εγγεγραμμένος, τότε καλείται το σχόλιο που έχει επιλεγεί για επεξεργασία από την βάση δεδομένων (γραμμή 64), με κριτήριο το id , δηλαδή τον μοναδικό αριθμό του κάθε σχολίου. Στη συνέχεια, όταν ο χρήστης κάνει τις αλλαγές που επιθυμεί στο σχόλιο του και πατήσει το κουμπί ολοκλήρωσης, το νέο περιεχόμενο του άρθρου αντικαθιστά το περιεχόμενο του παλιού στη βάση δεδομένων (γραμμή 76). Σε κάθε περίπτωση ο χρήστης ενημερώνεται για επιτυχία της ενέργειας που έκανε.

```
86 #Delete comment
87 @com.route('/delete_comment/<string:id>', methods=['POST'])
88 @is_logged_in
89 def delete_comment(id):
90     #create cursor
91     c = mysql.db.cursor()
92     #execute
93     c.execute("DELETE FROM comments WHERE id = %s", [id])
94     #commit
95     mysql.db.commit()
96     #close connection
97     c.close()
98     flash('Comment Deleted', 'success')
99     return redirect(url_for('hello.dashboard'))
100
```

ΕΙΚΟΝΑ 16.3: comments.py

Τέλος, η τελευταία δυνατότητα που προσφέρεται είναι αυτή της διαγραφής του άρθρου. Ο κώδικας της συνάρτησης διαγραφής (εικόνα 16.3) είναι σχετικά απλός και η μόνη προϋπόθεση είναι ο χρήστης να αποτελεί μέρος των εγγεγραμμένων χρηστών της εφαρμογής. Στη συνέχεια, με την χρήση του μοναδικού αριθμού του άρθρου, διαγράφεται από την βάση δεδομένων με την κατάλληλη εντολή (γραμμή 93).

```
1  {% extends 'layout.html' %}
2
3  {% block body %}
4  <table class="table table-striped">
5      <tr>
6          <th>ID</th>
7          <th>Author</th>
8          <th>Date</th>
9          <th>Body</th>
10         <th>Article_id</th>
11         <th></th>
12     </tr>
13     {% for comment in comments %}
14         <tr>
15             <td>{{comment[0]}}</td>
16             <td>{{comment[1]}}</td>
17             <td>{{comment[3]}}</td>
18             <td>{{comment[2] | safe}}</td>
19             <td>{{comment[4]}}</td>
20             {% if session['username'] == comment[1] %}
21             <td><a href="p13alva/edit_comment/{{comment[0]}}" class="btn btn-default pull-right">Edit</a></td>
22             <td>
23                 <form action="{{url_for('com.delete_comment', id=comment[0])}}" method="post">
24                     <input type="hidden" name="_method" value="DELETE">
25                     <input type="submit" value="Delete" class="btn btn-danger">
26                 {% endif %}
27                 </form>
28             </td>
29         </tr>
30     {% endfor %}
31 </table>
32 {% endblock %}
```

EIKONA 17: comments.html

Στην εικόνα 17 φαίνεται η σελίδα που περιέχει την λίστα των σχολίων. Για κάθε σχόλιο εμφανίζονται κάποιες πληροφορίες για αυτό, όπως είναι το id, η ημερομηνία δημιουργίας του, ο χρήστης που το δημοσίευσε, το περιεχόμενο και το άρθρο στο οποίο ανήκει (γραμμή 15-19). Στο τέλος, σε περίπτωση που ο χρήστης που βλέπει την λίστα είναι και ο συγγραφέας του σχολίου, εμφανίζονται τα κουμπιά για επεξεργασία και διαγραφή του άρθρου.

3.3.6 ΠΙΝΑΚΑΣ ΕΛΕΓΧΟΥ

Ο πίνακας (dashboard) είναι μια σελίδα της εφαρμογής στην οποία ο χρήστης μπορεί να δει τα άρθρα και τα σχόλια που έχει κάνει ο ίδιος. Επιπλέον, στη σελίδα, δίπλα από κάθε άρθρο ή σχόλιο, φαίνονται τα κουμπιά της επεξεργασίας και της διαγραφής της δημοσίευσης. Ιδιαίτερο ενδιαφέρον έχει ο κώδικας της σελίδας του πίνακα ελέγχου, το αρχείο dashboard.html. Παρακάτω παρουσιάζεται σε εικόνες ο κώδικας και η αντίστοιχη επεξήγηση του.

```

1  {% extends 'layout.html' %}
2
3  {% block body %}
4  <h1>Dashboard <small> Welcome {{session.username}}</small></h1>
5  <a class="btn btn-success" href="/p13alva/add_article"> Add Articles</a>
6  <hr>
7  <h1>Articles</h1>
8  <table class="table table-striped">
9      <tr>
10         <th>ID</th>
11         <th>Title</th>
12         <th>Author</th>
13         <th>Date</th>
14         <th></th>
15         <th></th>
16     </tr>
17     {% for article in articles %}
18         <tr>
19             <td>{{article[0]}}</td>
20             <td>{{article[1]}}</td>
21             <td>{{article[2]}}</td>
22             <td>{{article[4]}}</td>
23             {% if session['username'] == article[2] %}
24             <td><a href="/p13alva/edit_article/{{article[0]}}" class="btn btn-default pull-right">Edit</a></td>
25             <td>
26                 <form action="{{url_for('art.delete_article', id=article[0])}}" method="post">
27                     <input type="hidden" name="_method" value="DELETE">
28                     <input type="submit" value="Delete" class="btn btn-danger">
29                 {% endif %}
30             </form>
31             </td>
32         </tr>
33     {% endfor %}
34 </table>

```

EIKONA 18.1: dashboard.html

Η σελίδα του πίνακα ελέγχου περιέχει δυο πίνακες, οι οποίοι περιέχουν ο ένας τα άρθρα και ο δεύτερος τα σχόλια που έχει δημοσιεύσει ο χρήστης. Ο πίνακας των άρθρων (γραμμή 10-13) περιέχει τον μοναδικό αριθμό των άρθρων, τον τίτλο, τον συγγραφέα και την ημερομηνία ανάρτησης. Επιπλέον, στον κώδικα υπάρχει και η γραμμή 23, η οποία ελέγχει αν ο συγγραφέας του άρθρου είναι ο συνδεδεμένος εκείνη την στιγμή χρήστης. Αν η παραπάνω γραμμή ισχύει τότε εμφανίζονται τα κουμπιά για την επεξεργασία και της διαγραφής.

```

35 <h2>Comments</h2>
36 <table class="table table-striped">
37   <tr>
38     <th>ID</th>
39     <th>Author</th>
40     <th>Date</th>
41     <th></th>
42     <th></th>
43   </tr>
44   {% for comment in comments %}
45     <tr>
46       <td>{{comment[1]}}</td>
47       <td>{{comment[2]}}</td>
48       <td>{{comment[3]}}</td>
49       {% if session['username'] == comment[1] %}
50       <td><a href="edit_comment/{{comment[0]}}" class="btn btn-default pull-right">Edit</a></td>
51       <td>
52         <form action="{{url_for('com.delete_comment', id=comment[0])}}" method="post">
53           <input type="hidden" name="_method" value="DELETE">
54           <input type="submit" value="Delete" class="btn btn-danger">
55         {% endif %}
56       </form>
57     </td>
58   </tr>
59   {% endfor %}
60 </table>
61
62
63
64 {% endblock %}

```

ΕΙΚΟΝΑ 18.2: dashboard.html

Ο δεύτερος πίνακας έχει παρόμοια λειτουργία με τον πρώτο. Περιέχει τα πεδία για το id, τον συγγραφέα και την ημερομηνία ανάρτησης (γραμμή 18-40). Στη συνέχεια, εμφανίζει τα σχόλια από την βάση δεδομένων και γίνεται ο έλεγχος του χρήστη, που αναφέρθηκε προηγουμένως (γραμμή 49). Σε περίπτωση που ο χρήστης είναι και ο συγγραφέας τότε εμφανίζονται και οι επιλογές για την επεξεργασία και την διαγραφή. Γενικότερα, το αρχείο dashboard.html προσφέρει στον χρήστη την δυνατότητα του ελέγχου των αναρτήσεων του αλλά και της επεξεργασίας και της διαγραφής τους. Έτσι ο χρήστης μπορεί να βλέπει την κατάσταση των άρθρων και των σχολίων του, δηλαδή αν υπήρξε κάποιο ενδιαφέρον από τρίτους.

ΚΕΦΑΛΑΙΟ 4

4. ΣΥΜΠΕΡΑΣΜΑΤΑ – ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Στο τελευταίο κεφάλαιο της παρούσας πτυχιακής εργασίας, θα γίνει αναφορά στα συμπεράσματα που προέκυψαν κατά την υλοποίηση της εφαρμογής. Επιπλέον, θα γίνει αναφορά σε προτάσεις που αφορούν το μέλλον της εφαρμογής, σε λειτουργικότητα και ανάπτυξη.

4.1 ΣΥΜΠΕΡΑΣΜΑΤΑ

Σκοπός της πτυχιακής εργασίας είναι η δημιουργία μιας εφαρμογής κοινωνικής δικτύωσης για διαμοίραση και σχολιασμό προγραμματιστικού κώδικα. Επιπρόσθετα, η βασική επιδίωξη ήταν η δημιουργία ενός φιλικού και λειτουργικού συστήματος, με σκοπό την διευκόλυνση των χρηστών κατά την χρήση και την πλοήγηση στο περιβάλλον της εφαρμογής.

Οι χρήστες στους οποίους απευθύνεται η εφαρμογή είναι μαθητές και φοιτητές και γενικότερα άτομα που ασχολούνται με τον προγραμματισμό. Στη συνέχεια, αποφασίστηκε η δομή της εφαρμογής καθώς και οι υπηρεσίες που θα προσφέρει αλλά και η λειτουργίες που θα διαθέτει. Τέλος, ιδιαίτερη προσοχή δόθηκε στον τρόπο πλοήγησης των χρηστών μέσα την εφαρμογή.

Η βασική διαφορά της εργασίας είναι η ανάπτυξη ενός συστήματος με την χρήση της python και την χρήση flask για την

υλοποίηση της. Προκειμένου η εφαρμογή να λειτουργεί σωστά και να ικανοποιεί τον σκοπό της δημιουργίας, πρέπει να διαθέτει μεγάλο όγκο πληροφοριών και δεδομένων. Η ανάγκη αυτή επιτυγχάνεται από τους χρήστες και συγκεκριμένα από τον μεγάλο αριθμό χρηστών, όπου ο κάθε ένας θα βρίσκεται σε διαφορετικό επίπεδο γνώσεων. Με αυτό τον τρόπο θα υπάρχει μια ποικιλία ερωτήσεων αλλά και απαντημένες ερωτήσεις, το οποίο είναι και ο βασικός σκοπός της εφαρμογής.

Για την εξασφάλιση πολλών χρηστών αλλά και δεδομένων, ιδανική θεωρείται η διάθεση της εφαρμογής σε σχολές πληροφορικής και γενικότερα σε σχολές όπου οι φοιτητές ασχολούνται με τον προγραμματισμό. Με αυτόν τον τρόπο η εφαρμογή θα γίνει γρήγορα γνωστή ανάμεσα στους φοιτητές, με αποτέλεσμα την διάθεση μεγάλου αριθμού χρηστών και δεδομένων. Η φήμη της εφαρμογής θα βοηθήσει στην γνωστοποίηση του συστήματος πέρα από τον χώρο της εκπαίδευσης, στη διάθεση του στον εργασιακό τομέα και στην χρήση του από επαγγελματίες προγραμματιστές.

4.2 ΜΕΛΛΟΝΤΙΚΗ ΕΡΓΑΣΙΑ

Η εφαρμογή με της κατάλληλες τροποποιήσεις θα μπορεί να διαθέτει και μηχανισμό ανταλλαγής προσωπικών μηνυμάτων (chat). Επιπλέον, θα μπορούσε να διαθέτει μια σελίδα με τις πιο δημοφιλείς ερωτήσεις, αλλά και μια άλλη με ερωτήσεις οι οποίες έχουν απαντηθεί.

Παράλληλα, θα μπορούσε να βελτιωθεί και η ασφάλεια της εφαρμογής. Να υπάρξει δηλαδή, για παράδειγμα μια σελίδα για την ανάκτηση ή την έκδοση καινούριου κωδικού πρόσβασης. Επιπλέον, θα μπορούσε να δημιουργηθεί μηχανισμός για την αποστολή ειδοποιήσεων στους χρήστες σχετικά με μια ερώτηση. Αν κάποιος απαντήσει σε κάποια ερώτηση, να στέλνεται ειδοποίηση στον δημιουργό της.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. MYSQL, A. B. MySQL. 2001.
2. <https://www.phpmyadmin.net/>
3. <https://www.mysql.com/>
4. <https://atom.io/>
5. <https://www.bootstrapcdn.com/>
6. ΤΣΑΠΆΛΟΥ, Καλομοίρα. Κοινωνικά δίκτυα και ηλεκτρονικό μάρκετινγκ. 2016.
7. <http://flask.pocoo.org/>
8. <https://docs.python.org/3/license.html>
9. <https://www.w3.org/People/Raggett/book4/ch02.html>
10. ΘΕΟΔΩΡΙΔΗΣ, Θεοχάρης. Σχεδιασμός και ανάπτυξη ενός πληροφοριακού συστήματος διαχείρισης θέσεων πρακτικής άσκησης για το Πανεπιστήμιο Δυτικής Μακεδονίας. 2016. PhD Thesis. Θεοδωρίδης Θεοχάρης.
11. <https://www.w3.org/Style/CSS20/history.html>