ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

Τμήμα Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Πανεπιστήμιο Θεσσαλίας

ΔΥΝΑΜΙΚΗ ΔΙΑΔΙΚΤΥΑΚΗ ΕΦΑΡΜΟΓΗ ΜΕ ΧΡΗΣΗ NoSQL ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

DYNAMIC WEB APPLICATION USING A NoSQL DATABASE

Διπλωματική εργασία

Αθανάσιος Παπατζέλος

Επιβλέπων Καθηγητής: Βασιλακόπουλος Μιχαήλ

Αναπληρωτής Καθηγητής Π.Θ

Βόλος, 2017

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Πανεπιστήμιο Θεσσαλίας

Διπλωματική Εργασία με τίτλο:

ΔΥΝΑΜΙΚΗ ΔΙΑΔΙΚΤΥΑΚΗ ΕΦΑΡΜΟΓΗ ΜΕ ΧΡΗΣΗ NoSQL ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

DYNAMIC WEB APPLICATION USING NoSQL DATABASE

Αθανάσιος Παπατζέλος

Επιβλέπων Α'
Μιχαήλ Βασιλακόπουλος
Αναπληρωτής Καθηγητής

Επιβλέπων Β '
Τσομπανοπούλου Παναγιώτα
Αναπληρώτρια Καθηγήτρια

Βόλος, 2017

Ευχαριστίες

Ευχαριστώ τον επιβλέποντα καθηγητή μου κ. Βασιλακόπουλο Μιχαήλ για την πολύτιμη βοήθεια του, την διαρκή καθοδήγηση και τις σημαντικές συμβουλές του καθ' όλη την διάρκεια της παρούσας εργασίας.

Περίληψη

Η ανάγκη για διασκέδαση και ψυχαγωγία είναι έμφυτη στον άνθρωπο. Ανέκαθεν προσπαθούσε να βρει τρόπους για να το επιτύχει αυτό. Ένας από αυτούς είναι να βγαίνει έξω για καφέ και η επιλογή κατάλληλου καταστήματος είναι σημαντική. Για αυτό τον λόγο δημιουργήσαμε μία δυναμική διαδικτυακή εφαρμογή, με την οποία οι χρήστες μπορούν να εντοπίζουν και να αξιολογούν τα σχετικά καταστήματα, καθώς και να συζητούν μεταξύ τους, με σκοπό την βελτίωση της διασκέδασης τους.

Σκοπός της παρούσας διπλωματικής εργασίας είναι να δούμε πώς ανταποκρίνεται στη δημιουργία δυναμικών διαδικτυακών εφαρμογών μία NoSQL βάση δεδομένων, η ArangoDB, σε συνδυασμό με γλώσσες προγραμματισμού PHP, HTML και Javascript.

Και οι 3 παραπάνω γλώσσες προγραμματισμού χρησιμοποιούνται για τη δημιουργία σελίδων web με δυναμικό περιεχόμενο.

Η γλώσσα προγραμματισμού PHP χρησιμοποιείται για την επικοινωνία με τον διακομιστή (server).

Οι γλώσσες προγραμματισμού HTML και Javascript καθώς και επιπρόσθετα εργαλεία τους (Ajax, DOM) χρησιμοποιούνται για την απεικόνιση των δεδομένων στον χρήστη καθώς και την αλληλεπίδραση του χρήστη με την εφαρμογή.

Η εφαρμογή που δημιουργήσαμε είναι κατάλληλη τόσο για προσωπικό υπολογιστή, όσο και για κινητή συσκευή (κινητό, ταμπλέτα). Παρουσιάζουμε την ανάπτυξη και τη χρήση της και αξιολογούμε το τελικό αποτέλεσμα.

Abstract

The need for fun has always been part of human nature. People always try to find ways to achieve this goal. One of them is to go out for coffee and selecting an appropriate shop is important. That's why we've created a dynamic web applications that lets users identify and evaluate businesses, as well as, chat with each other to improve their entertainment.

The purpose of this diploma thesis is to examine how a NoSQL database, ArangoDB, combined with PHP, HTML and Javascript programming languages can be used for developing dynamic web applications.

All three of these programming languages are used to create web pages with dynamic content.

The programming language PHP is used for communication with the server.

The HTML and Javascript programming languages and their additional tools (Ajax, DOM) are used to display user data as well as to interact with the application.

The application we developed is suitable for personal computers, as well as, for mobile devices (mobile phone, or tablet). We present its development and use and evaluate the final result.

Λίστα πινάκων και σχημάτων

Πίνακας περιεχομένων

Κεφάλαιο 1: Εισαγωγή

1.1 Τι είναι μία διαδικτυακή εφαρμογή;

Διαδικτυακή εφαρμογή (web application ή web app) ονομάζεται κάθε εφαρμογή η οποία είναι διαθέσιμη στους χρήστες μέσω του διαδικτύου (Internet) και ο χρήστης χρησιμοποιεί μόνο τον περιηγητή του για να την χρησιμοποιήσει. Οι εφαρμογές αυτές συνήθως εκτελούνται σε ισχυρές υπολογιστικές μηχανές οι οποίες έχουν τον ρόλο του σταθμού εξυπηρέτησης και παρέχουν τις υπηρεσίες τους σε περισσότερους του ενός χρήστη [1].

1.1.1 Πλεονεκτήματα

- **Άμεση πρόσβαση από οποιαδήποτε συσκευή:** Οι χρήστες των διαδικτυακών εφαρμογών έχουν άμεση προσβασιμότητα στις εφαρμογές που θέλουν να χρησιμοποιήσουν από οποιονδήποτε υπολογιστή ή άλλη συσκευή έχει ίντερνετ χωρίς την εγκατάσταση κάποιου επιπρόσθετου λογισμικού. Η μόνη απαραίτητη εφαρμογή είναι ο περιηγητής διαδικτύου ο οποίος είναι προ εγκατεστημένος σε όλα τα λειτουργικά συστήματα ακόμα και στις φορητές συσκευές αλλά και στα κινητά τηλέφωνα. Η ιδιότητα αυτή των διαδικτυακών εφαρμογών είναι ιδιαίτερα σημαντική για μεγάλες επιχειρήσεις με πολλούς χρήστες που στην περίπτωση της τοπικής εφαρμογής θα έπρεπε να εγκατασταθεί η εφαρμογή σε κάθε ένα υπολογιστή ξεχωριστά.
- **Δυνατότητα χρήσης ανεξαρτήτου τοποθεσίας:** Ως συνέχεια του παραπάνω οι χρήστες των διαδικτυακών εφαρμογών μπορούν να τις χρησιμοποιούν ακόμα και αν δεν βρίσκονται στον χώρο εργασίας τους. Η δυνατότητα αυτή δίνει ευελιξία στους χρήστες ώστε να χρησιμοποιούν τις εφαρμογές οπουδήποτε στιγμή αυτοί επιθυμούνε

επιτρέποντας τους ακόμα και να εργάζονται από απομακρυσμένες περιοχές ή και από το σπίτι τους.

- **Συμβατές με όλα τα λειτουργικά συστήματα:** Ένα ακόμα πλεονέκτημα των διαδικτυακών εφαρμογών είναι ότι είναι συμβατές με όλα τα λειτουργικά συστήματα. Καθώς η εφαρμογή εκτελείτε μέσω του περιηγητή του διαδικτύου και όχι στον υπολογιστή του χρήστη την κάνει ικανή να εκτελείται σε όλα τα λειτουργικά συστήματα. Η ιδιότητα αυτή οφείλεται επίσης και στην προτυποποίηση των γλωσσών προγραμματισμού τις οποίες χρησιμοποιεί η εφαρμογή.

- **Δεν καταναλώνουν πόρους:** Ως συνέχεια του παραπάνω και εφόσον οι διαδικτυακές εφαρμογές δεν εκτελούνται στον υπολογιστή του χρήστη δεν καταναλώνουν και πόρους από το σύστημα. Για τον λόγο αυτό οι εφαρμογές διαδικτύου είναι ιδιαίτερα ελαφριές για την υπολογιστική μονάδα.

- **Δεν καταλαμβάνουν χώρο:** Ακολουθώντας την ίδια λογική με νωρίτερα οι εφαρμογές αυτές δεν καταλαμβάνουν καθόλου ή σχεδόν καθόλου χώρο στον δίσκο του χρήστη αφού το σύνολο της εφαρμογής είναι αποθηκευμένο στον εξυπηρετητή και μόνο κατά την χρήση της εφαρμογής μπορεί να υπάρχει μεταφορά δεδομένων προς την υπολογιστική μονάδα του χρήστη και μόνο στην περίπτωση που ο χρήστης το επιθυμεί.
- **Γρήγορη αναβάθμιση:** Σημαντικό πλεονέκτημα συγκριτικά με τις τοπικές εφαρμογές εμφανίζεται στις περιπτώσεις που η εφαρμογή χρειάζεται κάποια αναβάθμιση. Σε μια κλασική τοπική εφαρμογή η αναβάθμιση του συστήματος θα πρέπει να γίνει σε κάθε ένα υπολογιστή ξεχωριστά πράγμα που απαιτεί χρόνο και χρήμα. Αντίθετα σε μια διαδικτυακή εφαρμογή η αναβάθμιση πραγματοποιείται μόνο στον εξυπηρετητή που φιλοξενεί την εφαρμογή και ταυτόχρονα το αναβαθμισμένο πρόγραμμα είναι διαθέσιμο σε όλους τους χρήστες. Με τον τρόπο αυτό εξοικονομείτε χρόνος ο οποίος είναι ιδιαίτερα πολύτιμος κυρίως για τις μεγάλες επιχειρήσεις ενώ ως συνέπεια του παραπάνω σημαντικά μειωμένο είναι και το

κόστος της αναβάθμισης μιας και απαιτείται λιγότερο εργατικό δυναμικό για την διεκπεραίωση της αναβάθμισης.

- **Νέο βελτιωμένο περιβάλλον:** Ένα ακόμα πλεονέκτημα των διαδικτυακών εφαρμογών είναι ότι πλέον με την εμφάνιση της HTML5 είναι δυνατό ο δημιουργός της εφαρμογής να την εμπλουτίσει έτσι ώστε να είναι πιο φιλική, εύχρηστη και ευχάριστη προς τον χρήστη με εύκολο τρόπο. Παλαιότερα οι εφαρμογές αυτές υστερούσαν στην εμφάνιση, πλέον είναι ιδιαίτερα εύκολα να εμπλουτιστούν.

- **Δυνατότητα χρήσης και εκτός διαδικτύου - ενδοδικτύου:** Ένα ακόμα πλεονέκτημα των σύγχρονων διαδικτυακών εφαρμογών (εφαρμογές με χρήση HTML5 είναι η δυνατότητα της εκτός διαδικτύου χρήσης μιας διαδικτυακής εφαρμογής με την προϋπόθεση ότι η εφαρμογή έχει κατασκευασθεί με ανάλογο τρόπο. Για παράδειγμα αν για κάποιο λόγο η σύνδεση στο διαδίκτυο διακοπεί αυτό δεν επηρεάζει τον χρήστη ο οποίος συνεχίζει να χρησιμοποιεί την εφαρμογή κανονικά. Αυτό επιτυγχάνεται από τον περιηγητή ο οποίος κρατάει ένα αντίγραφο από τα αρχεία τα οποία είναι απαραίτητα για την εκτός δικτύου χρήση της εφαρμογής, στον υπολογιστή του χρήστη, και τα χρησιμοποιεί όταν αυτό κριθεί απαραίτητο. Η συγκεκριμένη δυνατότητα δεν είναι διαθέσιμη σε όλες τις εφαρμογές που χρησιμοποιούν HTML5 αλλά μόνο σε αυτές που έχει υπάρξει πρόβλεψη για χρήση της εφαρμογής και εκτός διαδικτύου ή ενδοδικτύου.

1.1.2 Μειονεκτήματα

- **Χρήση της εφαρμογής εκτός διαδικτύου:** Ένα μειονέκτημα που ταυτόχρονα είναι και πλεονέκτημα είναι η χρήση της εφαρμογής εκτός διαδικτύου. Προκειμένου να γίνει αυτό εφικτό θα πρέπει να έχει γίνει πρόβλεψη κατά την σχεδίαση της εφαρμογής και να έχουν ληφθεί τα κατάλληλα μέτρα. Σε περίπτωση που κάτι τέτοιο δεν έχει γίνει τότε η εφαρμογή δεν είναι δυνατό να χρησιμοποιηθεί χωρίς την σύνδεση του χρήση με το διαδίκτυο ή το ενδοδίκτυο της εταιρίας.

- **Αδυναμία χρήσης χωρίς σύνδεση στο διαδίκτυο:** Σαν συνέχεια του παραπάνω, εφαρμογές που δεν έχουν κατασκευασθεί με χρήση της τελευταίας έκδοσης της HTML5 δηλαδή παλαιότερες εφαρμογές δεν είναι δυνατόν να χρησιμοποιηθούν αν δεν υπάρχει σύνδεση με το διαδίκτυο ή το ενδοδίκτυο. Το παραπάνω αποτελεί και το κύριο μειονέκτημα των παλαιότερων διαδικτυακών εφαρμογών καθώς αν για κάποιο λόγο η σύνδεση του χρήστη διακοπεί τότε αυτός δεν μπορεί να χρησιμοποιήσει καθόλου την εφαρμογή. Σε περίπτωση που μια παλαιότερη διαδικτυακή εφαρμογή θελήσει να αυξήσει τις δυνατότητες της προκειμένου να μπορεί να λειτουργεί και εκτός διαδικτύου τότε αυτή θα πρέπει να ξανακατασκευασθεί σε μεγάλο μέρος της από την αρχή.

- **Μη πλήρη συμβατότητα των περιηγητών:** Ένα ακόμα μειονέκτημα που αφορά την τελευταία έκδοση της HTML5 είναι η μη πλήρη συμβατότητα των περιηγητών με την έκδοση αυτή. Αν και τα πλεονεκτήματα και οι δυνατότητες της HTML5 είναι πολλά, αρκετοί από τους περιηγητές δεν είναι ακόμα πλήρως συμβατοί με αυτά.

  Έτσι δεν γίνεται πλήρη χρήση των δυνατοτήτων αυτών πράγμα που περιορίζει τους προγραμματιστές που έχουν αναλάβει ένα έργο. Επίσης σε περίπτωση που δεν έχει προβλεφθεί η μη λειτουργία κάποιου χαρακτηριστικού της εφαρμογής σε κάποιον περιηγητή αυτό μπορεί να δημιουργήσει προβλήματα στην εφαρμογή με αποτέλεσμα να μην λειτουργεί σωστά ή να μην λειτουργεί καθόλου. Για το λόγο αυτό ο κατασκευαστής της εφαρμογής με τον πελάτη πρέπει από κοινού να αποφασίζουνε ποιος περιηγητής θα είναι ο προτεινόμενος για την εφαρμογή αλλά ταυτόχρονα να προβλέπεται και η περίπτωση χρήσης άλλων περιηγητών. Ένας καλός τρόπος για να ελεγχθεί η συμβατότητα του περιηγητής μας με την HTML5 είναι τα διάφορα διαδικτυακά τεστ που αξιολογούν τις δυνατότητες του.

- **Άμεση αναβάθμιση:** Ένα ακόμα χαρακτηριστικό παράδειγμα πλεονεκτήματος και μειονεκτήματος ταυτόχρονα αποτελεί και η αναβάθμιση της εφαρμογής. Στην περίπτωση της τοπικής εφαρμογής μια επιχείρηση μπορεί να αναβαθμίσει την εφαρμογή που χρησιμοποιεί όποτε αυτή το κρίνει αναγκαίο κρίνοντας το κόστος αναβάθμισης,την αξιοπιστία της νέας εφαρμογής αλλά και το χρόνο που θα χρειαστούν οι υπάλληλοι της ώστε να προσαρμοστούν στην νέα έκδοση. Αντίθετα στις διαδικτυακές εφαρμογές η αναβάθμιση γίνεται χωρίς πρώτα να ερωτηθούν όλοι οι χρήστες. Για παράδειγμα στην περίπτωση που η ερχόμενη αναβάθμιση μιας τοπικής

εφαρμογής έχει σφάλματα τότε μπορεί κάποιος χρήστης να μην πραγματοποιήσει την αναβάθμιση έως ότου διορθωθούν αυτά. Στην περίπτωση όμως της διαδικτυακής εφαρμογής ο χρήστης δεν μπορεί να αποτρέψει την αναβάθμιση αυτή.

- **Πιθανή μη συμβατότητα κάποιων στοιχείων της εφαρμογής με μια μελλοντική έκδοση του περιηγητή μας:** Τέλος μειονέκτημα είναι και η πιθανή μη συμβατότητα κάποιον στοιχείων του προγράμματος με μια μελλοντική έκδοση του περιηγητή μας. Αυτό συμβαίνει διότι συχνά παρατηρείται το φαινόμενο να εγκαταλείπονται κάποια υποστηριζόμενα στοιχεία από τους περιηγητές αν η δημιουργός εταιρία κρίνει ότι αυτά δεν έχουν μέλλον στις εφαρμογές διαδικτύου. Το αποτέλεσμα αυτών των αποφάσεων είναι η δυσλειτουργία κάποιων παλαιότερων εφαρμογών καθιστώντας αναγκαία την αναβάθμιση τους.

Κεφάλαιο 2: Εργαλεία Ανάπτυξης και Σχεδιασμού

2.1 Τεχνολογίες που χρησιμοποιήθηκαν

• **HTML** (Hyper Text Markup Language)

Είναι η κύρια γλώσσα κατασκευής ιστοσελίδων και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων [17].

• **CSS** (Cascading Style Sheets)

Είναι μια γλώσσα υπολογιστή που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης. Χρησιμοποιείται δηλαδή για τον έλεγχο της εμφάνισης ενός εγγράφου που γράφτηκε στις γλώσσες HTML και XHTML , δηλαδή για τον έλεγχο της εμφάνισης μιας ιστοσελίδας και γενικότερα ενός ιστότοπου. Η CSS είναι μια γλώσσα υπολογιστή προορισμένη να αναπτύσσει στιλιστικά μια ιστοσελίδα δηλαδή να διαμορφώνει περισσότερα χαρακτηριστικά, χρώματα, στοίχιση και δίνει περισσότερες δυνατότητες σε σχέση με την HTML. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται ως απαραίτητη.

• **JavaScript**

Είναι γλώσσα προγραμματισμού που έχει σαν σκοπό την παραγωγή δυναμικού περιεχομένου και την εκτέλεση κώδικα στην πλευρά του πελάτη (client-side).

• **PHP** (Hypertext Preprocessor)

Είναι γλώσσα προγραμματισμού για τη δημιουργία ιστοσελίδων δυναμικού περιεχόμενου που εκτελείται στον εξυπηρετητή (server-side).Η PHP αποτελεί μια από τις πιο διαδεδομένες τεχνολογίες στο Παγκόσμιο Ιστό, καθώς χρησιμοποιείται από πληθώρα εφαρμογών και ιστότοπων. Η ευρύτητα στη χρήση της είναι απόρροια της ευκολίας που παρουσιάζει ο προγραμματισμός με αυτή αλλά και στο γεγονός πως είναι μια γλώσσα η οποία βρίσκεται σχεδόν σε κάθε διακομιστή [16].

Τα παρακάτω «εργαλεία» δεν είναι γλώσσες προγραμματισμού από μόνες τους αλλά συνδυασμοί γλωσσών και τεχνικών προγραμματισμού:

• **jQuery**

Είναι μια βιβλιοθήκη JavaScript σχεδιασμένη για να απλοποιεί τον προγραμματισμό στην πλευρά του πελάτη (client-side scripting) της [11].

• **Ajax** (Asynchronous JavaScript and XML).

Είναι μια ομάδα αλληλένδετων τεχνικών ανάπτυξης για τη δημιουργία ασύγχρονων εφαρμογών. Μέσω της javascript προγραμματίζουμε τον client και μέσω της php προγραμματίζουμε την επικοινωνία με τον server.Με την βοήθεια της Ajax μεταφέρουμε μεταβλητές javascript (client side) σε php (server side) χωρίς την ανάγκη ανανέωσης της σελίδας [15].

• **DOM (**Document Object Model**)**.

Το μοντέλο αντικειμένου εγγράφου (DOM Object Model - DOM) είναι μια διεπαφή προγραμματισμού εφαρμογών πλατφόρμας και γλώσσας που αντιμετωπίζει ένα έγγραφο HTML, XHTML ή XML ως δομή δέντρου όπου κάθε κόμβος είναι ένα αντικείμενο που αντιπροσωπεύει ένα μέρος του εγγράφου. Τα αντικείμενα μπορούν να επεξεργαστούν προγραμματικά και οποιεσδήποτε ορατές

μεταβολές που προκύπτουν ως αποτέλεσμα μπορεί στη συνέχεια να αντανακλώνται στην απεικόνιση του εγγράφου [8].

• **Apache HTTP server**

Είναι ένας εξυπηρετητής του παγκόσμιου ιστού (web). Όποτε ένας χρήστης επισκέπτεται ένα ιστότοπο το πρόγραμμα πλοήγησης (browser) επικοινωνεί με έναν διακομιστή (server) μέσω του πρωτοκόλλου HTTP, ο οποίος παράγει τις ιστοσελίδες και τις αποστέλλει στο πρόγραμμα πλοήγησης. Ο Apache είναι ένας από τους δημοφιλέστερους εξυπηρετητές ιστού, εν μέρει γιατί λειτουργεί σε διάφορες πλατφόρμες όπως τα Windows, το Linux, το Unix και το Mac OS X. Κυκλοφόρησε υπό την άδεια λογισμικού Apache και είναι λογισμικό ανοιχτού κώδικα. Συντηρείται από μια κοινότητα ανοικτού κώδικα με επιτήρηση από το Ίδρυμα Λογισμικού Apache (Apache Software Foundation).

Ο Apache χρησιμοποιείται και σε τοπικά δίκτυα σαν διακομιστής συνεργαζόμενος με συστήματα διαχείρισης Βάσης Δεδομένων π.χ. Oracle, MySQL, ArangoDB [14].

2.2 NoSQL databases

Μια βάση δεδομένων **NoSQL** ("μη SQL" ή "μη σχεσιακή") ,παρέχει έναν μηχανισμό αποθήκευσης και ανάκτησης δεδομένων που μοντελοποιείται σε μέσα διαφορετικά από τις πινακοποιημένες σχέσεις που χρησιμοποιούνται στις σχεσιακές βάσεις δεδομένων. Αυτές οι βάσεις δεδομένων υπήρχαν από τα τέλη της δεκαετίας του 1960, αλλά δεν έλαβαν την ονομασία "NoSQL" μέχρι να αυξηθεί η δημοτικότητα τους στις αρχές του εικοστού πρώτου αιώνα. Βάσεις δεδομένων NoSQL χρησιμοποιούνται όλο και περισσότερο σε εφαρμογές μεγάλου όγκου δεδομένων (big data) και σε πραγματικό χρόνο (real time). Τα συστήματα NoSQL καλούνται επίσης μερικές φορές "όχι μόνο SQL" για να τονίσουν ότι μπορούν να υποστηρίξουν γλώσσες ερωτήσεων (query languages) τύπου SQL.

2.2.1 NoSQL vs SQL

Οι βάσεις δεδομένων SQL καλούνται κυρίως ως σχεσιακές βάσεις δεδομένων (RDBMS),ενώ η βάση δεδομένων NoSQL ονομάζεται κυρίως ως μη σχεσιακή ή κατανεμημένη βάση δεδομένων [2][3][5][13].

•Οι βάσεις δεδομένων SQL είναι βάσεις δεδομένων που βασίζονται σε πίνακες, ενώ οι βάσεις δεδομένων NoSQL βασίζονται σε έγγραφα, ζεύγη κλειδιών-τιμών, βάσεις δεδομένων γραφικών ή ευρείας στήλης. Αυτό σημαίνει ότι οι βάσεις δεδομένων SQL αντιπροσωπεύουν δεδομένα σε μορφή πινάκων που αποτελείται από n αριθμό σειρών δεδομένων ενώ οι βάσεις δεδομένων NoSQL είναι η συλλογή ζευγαριών κλειδιών-τιμών, εγγράφων, βάσεων δεδομένων γραφικών ή στήλες που δεν έχουν τυπικούς ορισμούς σχήματος που πρέπει να τηρηθούν.

•Οι βάσεις δεδομένων SQL έχουν προκαθορισμένο σχήμα, ενώ οι βάσεις δεδομένων NoSQL διαθέτουν δυναμικό σχήμα για μη δομημένα δεδομένα.

•Οι βάσεις δεδομένων SQL είναι κατακόρυφα κλιμακούμενες, ενώ οι βάσεις δεδομένων NoSQL είναι οριζόντια κλιμακούμενες. Οι βάσεις δεδομένων SQL κλιμακώνονται αυξάνοντας την ιπποδύναμη του υλικού. Οι βάσεις δεδομένων NoSQL κλιμακώνουν αυξάνοντας τους διακομιστές βάσεων δεδομένων στην ομάδα πόρων για να μειώσουν το φορτίο.

•Οι βάσεις δεδομένων SQL χρησιμοποιούν SQL (δομημένη γλώσσα ερωτήματος) για τον καθορισμό και τον χειρισμό των δεδομένων. Στη βάση δεδομένων NoSQL, τα ερωτήματα εστιάζονται στη συλλογή εγγράφων. Μερικές φορές ονομάζεται επίσης UnQL (Unstructured Query Language).Η σύνταξη της χρήσης της UnQL ποικίλλει από βάση δεδομένων σε βάση δεδομένων.

Παραδείγματα βάσεων δεδομένων SQL: MySql, Oracle, Sqlite, Postgres και MS-SQL. Παραδείγματα βάσεων δεδομένων NoSQL: ArangoDB, MongoDB, BigTable, Redis, RavenDb, Cassandra, Hbase, Neo4j και CouchDb.

•**Για σύνθετα ερωτήματα:** Οι βάσεις δεδομένων SQL είναι κατάλληλες για το πολύπλοκο περιβάλλον που απαιτεί πολύ προσοχή, ενώ οι βάσεις δεδομένων NoSQL δεν είναι κατάλληλες για σύνθετα ερωτήματα. Οι NoSQL δεν έχουν πρότυπες διεπαφές για την εκτέλεση πολύπλοκων ερωτημάτων και τα ίδια τα ερωτήματα στη NoSQL δεν είναι τόσο ισχυρά όσο στην ερωτηματική γλώσσα SQL .

•**Για τον τύπο των δεδομένων που πρέπει να αποθηκευτούν:** Οι βάσεις δεδομένων SQL δεν είναι οι πλέον κατάλληλες για την αποθήκευση ιεραρχικών δεδομένων. Όμως, η βάση δεδομένων NoSQL ταιριάζει καλύτερα για την ιεραρχική αποθήκευση δεδομένων καθώς ακολουθεί τον τρόπο ζευγών κλειδιών-τιμών για την αποθήκευση δεδομένων παρόμοια με τα δεδομένα JSON.Η βάση δεδομένων NoSQL προτιμάται ιδιαίτερα για μεγάλο σύνολο δεδομένων (δηλ. Big Data).

•**Για την επεκτασιμότητα:** Στις περισσότερες τυπικές καταστάσεις, οι βάσεις δεδομένων SQL είναι κατακόρυφα κλιμακωτές. Μπορούμε να διαχειριστούμε το αυξανόμενο φορτίο αυξάνοντας την CPU, RAM, SSD, κλπ, σε ένα μόνο διακομιστή. Από την άλλη πλευρά, οι βάσεις δεδομένων NoSQL είναι οριζόντια κλιμακωτές. Μπορούμε να προσθέσουμε μερικούς ακόμα εξυπηρετητές εύκολα στην υποδομή της βάσης δεδομένων NoSQL για να χειριστούμε τη μεγάλη κίνηση.

•**Για εφαρμογές βασισμένες σε υψηλές συναλλαγές:** Οι βάσεις δεδομένων SQL ταιριάζουν καλύτερα στις εφαρμογές τύπου "βαρέως τύπου", καθώς είναι πιο σταθερές και υπόσχονται την ατομικότητα καθώς και την ακεραιότητα των δεδομένων. Παρόλο που μπορούμε να χρησιμοποιήσουμε NoSQL για σκοπούς συναλλαγών, εξακολουθεί να μην είναι συγκρίσιμη και ικανοποιητική σε υψηλό φορτίο και για πολύπλοκες συναλλαγές.

•**Για υποστήριξη:** Εξαιρετική υποστήριξη είναι διαθέσιμη για όλες τις βάσεις δεδομένων SQL από τους προμηθευτές τους. Για μερικές βάσεις δεδομένων NoSQL πρέπει να βασιζόμαστε στην υποστήριξη της κοινότητας και μόνο περιορισμένοι εξωτερικοί εμπειρογνώμονες είναι διαθέσιμοι για να εγκαταστήσουν και να αναπτύξουν τις εφαρμογές NoSQL μεγάλης κλίμακας.

•**Για ιδιότητες:** Οι βάσεις δεδομένων SQL δίνουν έμφαση στις ιδιότητες του ACID (Ατομικότητα, Συνάφεια, Απομόνωση και Ανθεκτικότητα), ενώ η βάση δεδομένων NoSQL ακολουθεί το θεώρημα CAP του Brewers (ανοχή συνεκτικότητας, διαθεσιμότητας και διαχωρισμού)

Οι διαφορές ανάμεσα στις SQL και NoSQL βάσεις δεδομένων συνοψίζονται στον πίνακα 2.1:

Index	SQL	NoSQL
1)	Databases are categorized as Relational Database Management System (RDBMS).	NoSQL databases are categorized as Non-relational or distributed database system.
2)	SQL databases have fixed or static or predefined schema.	NoSQL databases have dynamic schema.
3)	SQL databases display data in form of tables so it is known as table-based database.	NoSQL databases display data as collection of key-value pair, documents, graph databases or wide-column stores.
4)	SQL databases are vertically scalable.	NoSQL databases are horizontally scalable.
5)	SQL databases use a powerful language "Structured Query Language" to define and manipulate the data.	In NoSQL databases, collection of documents are used to query the data. It is also called unstructured query language. It varies from database to database.
6)	SQL databases are best suited for complex queries.	NoSQL databases are not so good for complex queries because these are not as powerful as SQL queries.
7)	SQL databases are not best suited for hierarchical data storage.	NoSQL databases are best suited for hierarchical data storage.
8)	MySQL, Oracle, Sqlite, PostgreSQL and MS-SQL etc. are the example of SQL database.	MongoDB, BigTable, Redis, RavenDB, Cassandra, Hbase, Neo4j, CouchDB etc. are the example of nosql database

Πίνακας 2.1: Διαφορές ανάμεσα σε SQL και NoSQL.

2.3 ArangoDB

Η ArangoDB είναι μια μητρική βάση πολλαπλών μοντέλων NoSQL που αναπτύχθηκε από την triAGENS GmbH. Σε ένα βιβλίο που δημοσιεύθηκε το 2015, αναφέρεται ως η πιο δημοφιλής βάση δεδομένων NoSQL που διέθετε άδεια ανοικτού κώδικα. Έχει επίσης αναφερθεί ως παγκόσμια βάση δεδομένων. Οι δημιουργοί του αναφέρονται σε μια βάση δεδομένων "φυσικού πολυ-μοντέλου" που υποδηλώνει ότι έχει σχεδιαστεί ειδικά για να επιτρέπει την αποθήκευση δεδομένων κλειδιού / τιμής, εγγράφου και γραφήματος μαζί και την αναζήτηση με κοινή γλώσσα.

Η ArangoDB παρέχει κλιμακούμενα, υψηλής απόδοσης ερωτήματα όταν εργαζόμαστε με δεδομένα γραφημάτων. Η βάση δεδομένων χρησιμοποιεί JSON ως προεπιλεγμένη μορφή αποθήκευσης,αλλά εσωτερικά χρησιμοποιεί το VelocyPack του ArangoDB - μια γρήγορη και συμπαγή δυαδική μορφή για σειριοποίηση και αποθήκευση. Η ArangoDB μπορεί να αποθηκεύσει εγγενώς ένα ένθετο αντικείμενο JSON ως είσοδο δεδομένων μέσα σε μια συλλογή. Επομένως, δεν χρειάζεται να αποσυναρμολογήσουμε τα προκύπτοντα αντικείμενα JSON. Επομένως, τα αποθηκευμένα δεδομένα απλώς θα κληρονομήσουν τη δομή δέντρου των δεδομένων XML.

Η ArangoDB λειτουργεί σε ένα διανεμημένο σύμπλεγμα σε αντίθεση με κάποιες άλλες υπάρχουσες βάσεις δεδομένων γραφικών και είναι το πρώτο ΣΔΒΔ (Σύστημα Διαχείρισης Βάσεων Δεδομένων) που είναι πιστοποιημένο για το Distributed Cluster Operating System (DC / OS).

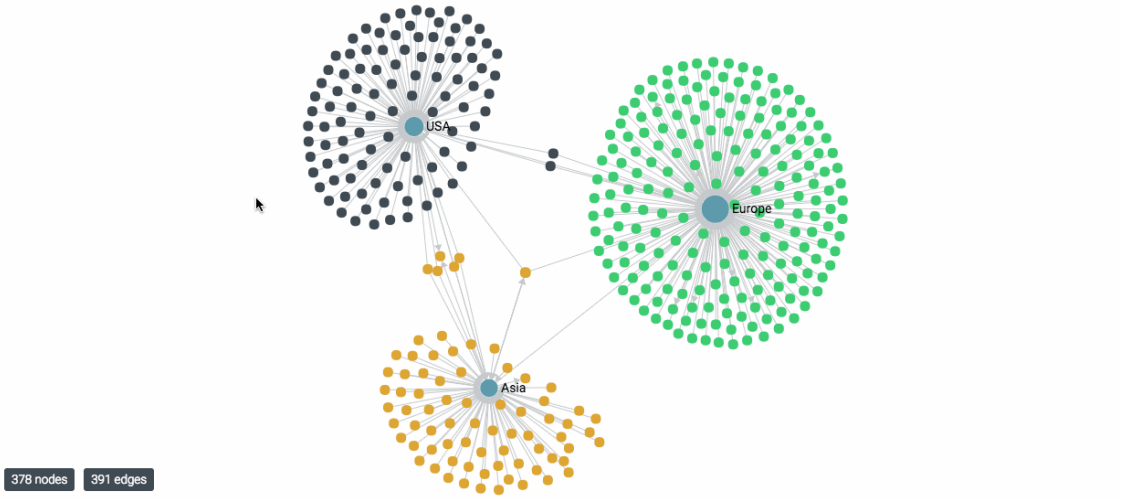
Η ArangoDB παρέχει ενσωμάτωση με τις εγγενείς μικρουπηρεσίες της JavaScript απευθείας πάνω από το ΣΔΒΔ χρησιμοποιώντας το πλαίσιο Foxx, το οποίο είναι ανάλογο με το πολυνηματικό NodeJS.

Η βάση δεδομένων έχει και γλώσσα ερωτήματος AQL **(Arango Query Language)** και παρέχει GraphQL για να γράψει ευέλικτες υπηρεσίες εγγενών ιστών απευθείας πάνω από το ΣΔΒΔ (Σύστημα Διαχείρισης Βάσεων Δεδομένων) [9][10].

2.3.1 Γραφικό περιβάλλον της ArangoDB

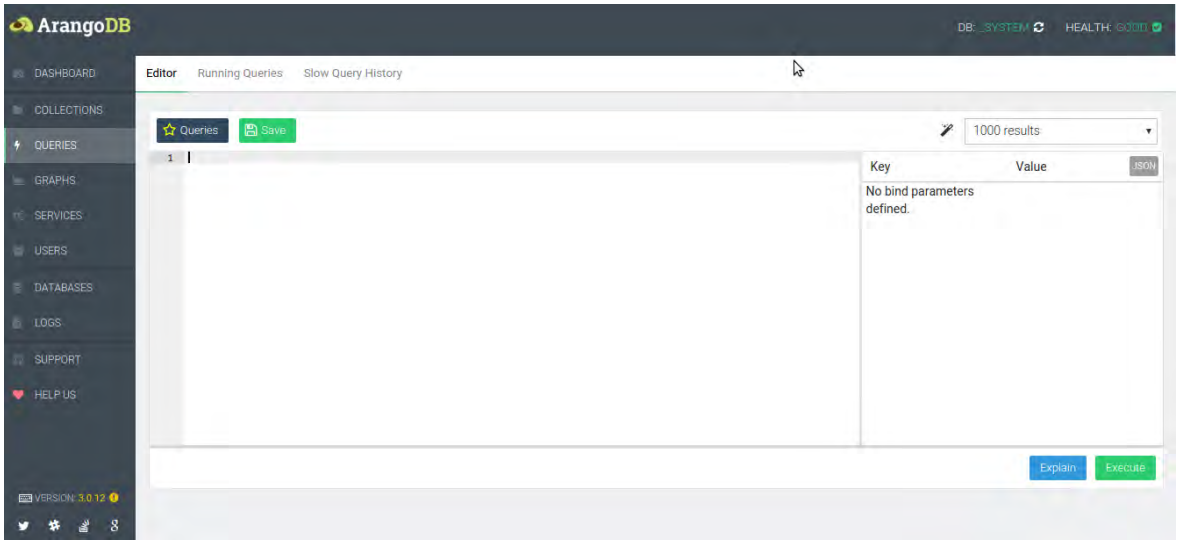
Η **ArangoDB** παρέχει μία διεπαφή ιστού με τις εξής λειτουργίες για την ευκολότερη διαχείριση της βάσης δεδομένων [18].

- Graph viewer: Πρόκειται για ένα εργαλείο για την απεικόνιση των γραφημάτων της βάσης δεδομένων, το οποιο δείχνει τις κορυφές και τα άκρα σύνδεσης.



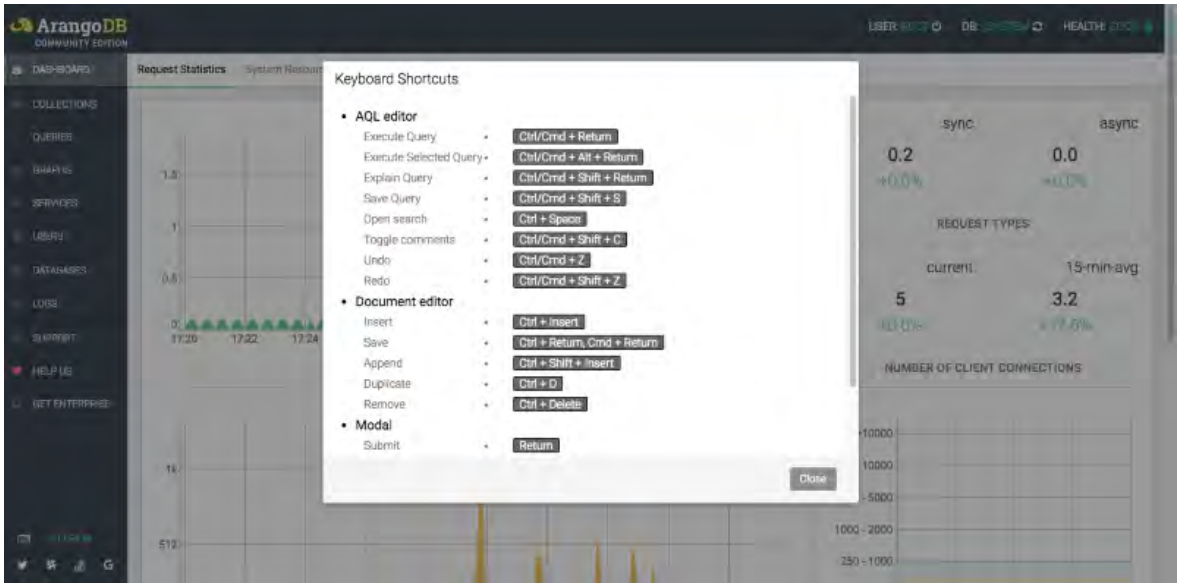
Σχήμα 2.2: Γραφική αναπαράσταση της βάσης δεδομένων

- AQL Editor: Είναι ένας επεξεργαστής κειμένου με τον οποίο μπορούμε να δοκιμάσουμε ερωτήματα AQL στην βάση δεδομένων για την διερεύνηση ζητημάτων και τη βελτίωση της συνολικής απόδοσης των ερωτημάτων.



Σχήμα 2.3: Επεξεργαστής ερωτημάτων AQL

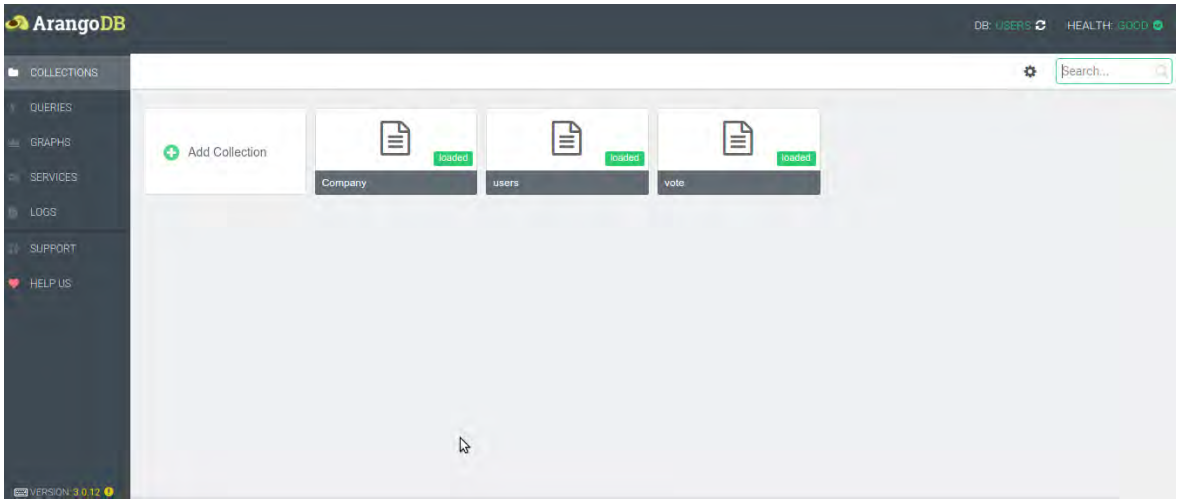
- Shortcuts Overview: Η διασύνδεση υποστηρίζει συντομεύσεις πληκτρολογίου, καθιστώντας ευκολότερη και ταχύτερη την εκτέλεση εργασιών στο γραφικό



περιβάλλον χρήστη.

Σχήμα 2.4: Συντομεύσεις πληκτρολογίου

Στην δική μας εφαρμογή η βάση δεδομένων αποτελείται από τις 3 συλλογές όπως φαίνονται στο σχήμα:



Σχήμα 2.5: Αναπαράσταση από συλλογές της βάσης δεδομένων της εφαρμογής CoffeFun.

2.3.2 AQL vs SQL

Η **γλώσσα ερωτημάτων ArangoDB (AQL (Arango Query Language))** είναι παρόμοια με τη **δομημένη γλώσσα ερωτήματος (SQL).** Και οι δύο υποστηρίζουν την ανάγνωση και τροποποίηση δεδομένων συλλογής, ωστόσο η AQL δεν υποστηρίζει λειτουργίες ορισμού δεδομένων, όπως η δημιουργία και η απομάκρυνση βάσεων δεδομένων, συλλογών και ευρετηρίων.

Παρόλο που ορισμένες λέξεις-κλειδιά επικαλύπτονται, η σύνταξη AQL διαφέρει από την SQL. Για παράδειγμα, οι λέξεις-κλειδιά της SQL WHERE και της AQL FILTER είναι ισοδύναμες με το ότι και οι δύο ορίζουν συνθήκες για την επιστροφή αποτελεσμάτων. Η SQL χρησιμοποιεί προκαθορισμένη ακολουθία για να προσδιορίσει πού πρέπει να εμφανιστεί η λέξη-κλειδί WHERE στη δήλωση. Στην AQL, οι λέξεις-κλειδιά εκτελούνται από αριστερά προς τα δεξιά, έτσι η θέση μιας λέξης-κλειδί FILTER στο ερώτημα καθορίζει την προτεραιότητά της [4].

Παρακάτω είναι ένας πίνακας με τους όρους και των δύο συστημάτων :

SQL	AQL
database	database
table	collection
row	document
column	attribute
table joins	collection joins
primary key	primary key (automatically present on `_key` attribute)
index	index

Πίνακας 2.6: Αντιστοίχιση όρων μεταξύ SQL και AQL.

Κεφάλαιο 3: Σχεδιασμός και Υλοποίηση

3.1 Εκδόσεις εφαρμογής

Το σύστημα μας έχει δύο εκδόσεις, η μία είναι για οθόνες με πλάτος μεγαλύτερο απο 800 πίξελς και η άλλη για μικρότερες ή ίσες απο 800 πίξελς.Η βασική διαφορά και στις δύο εκδόσεις είναι ο διαφορετικός κώδικας **css** ώστε να εμφανίζεται η ιστοσελίδα καλύτερα στην οθόνη,καθώς μοιράζονται το πιο σημαντικό κομμάτι κώδικα που υλοποιεί όλες τις λειτουργίες.

3.2 Ιεραρχία Καταλόγων

Όλα τα αρχεία μας βρίσκονται στον κατάλογο **/var/www/html/ .**Τα αρχεία κώδικα που τρέχουνε για σταθερό υπολογιστή βρίσκονται σε αυτόν το κατάλογο. Αντιθέτως τα αρχεία που τρέχουνε για κινητό τηλέφωνο βρίσκονται στον υποκατάλογο mobile_version.Ωστόσο και οι δύο εκδόσεις χρησιμοποιούν απο κοινού το αρχείο db.php το οποίο περιέχει όλες τις βασικές συναρτήσεις που υλοποιούν τις βασικές λειτουργίες της εφαρμογής.

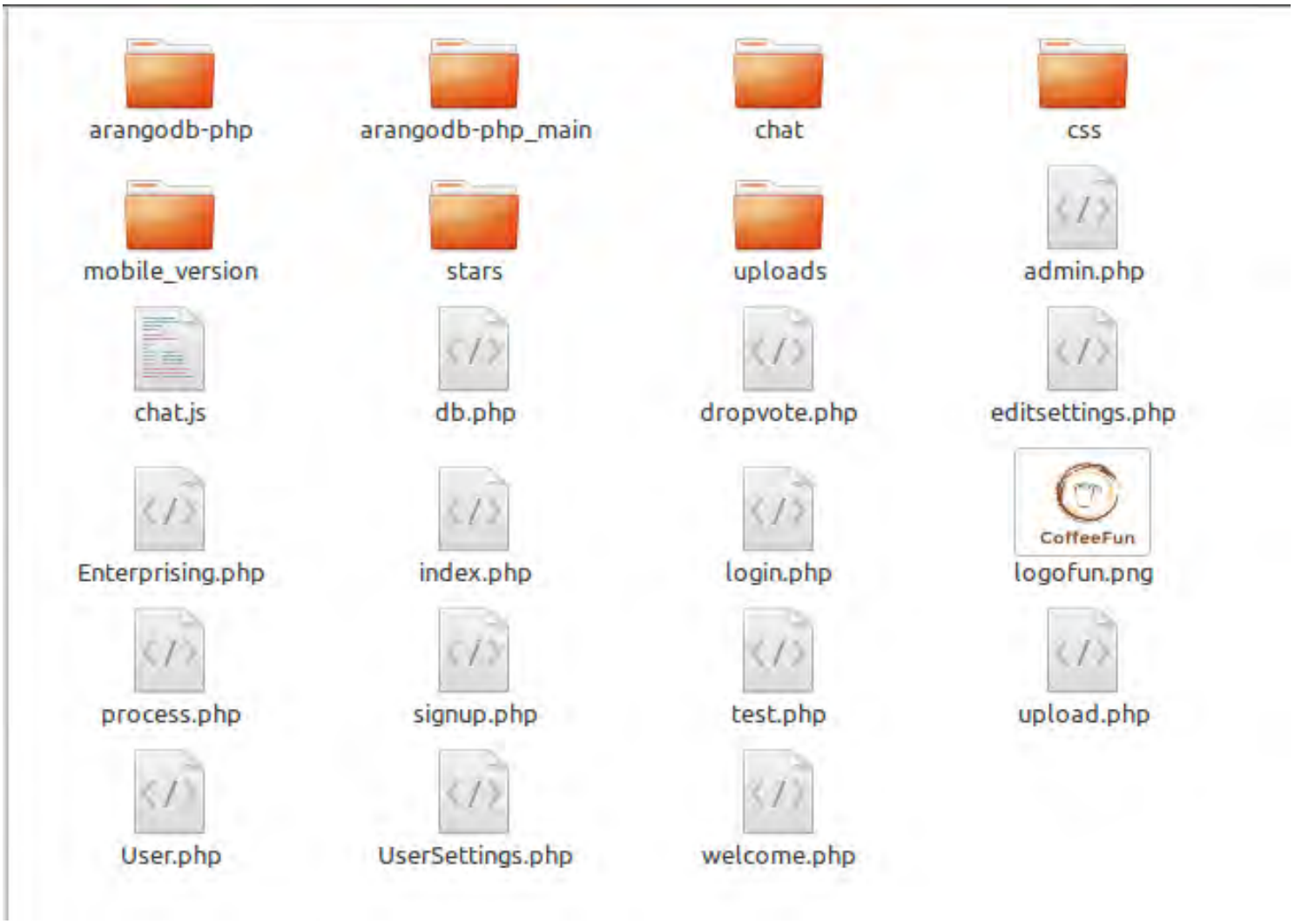
Ο υποκατάλογος stars περιέχει τις εικόνες απο αστέρια για την αξιολόγηση των καφετεριών.

Ο υποκατάλογος uploads περιέχει τις φωτογραφίες των καφετεριών.

Ο υποκατάλογος chat περιέχει για κάθε καφετέρια ένα ξεχωριστό αρχείο κειμένου (txt file) με το όνομα της εκάστοτε καφετέριας,κρατώντας σε αυτό το ιστορικό συνομιλίας των χρηστών.

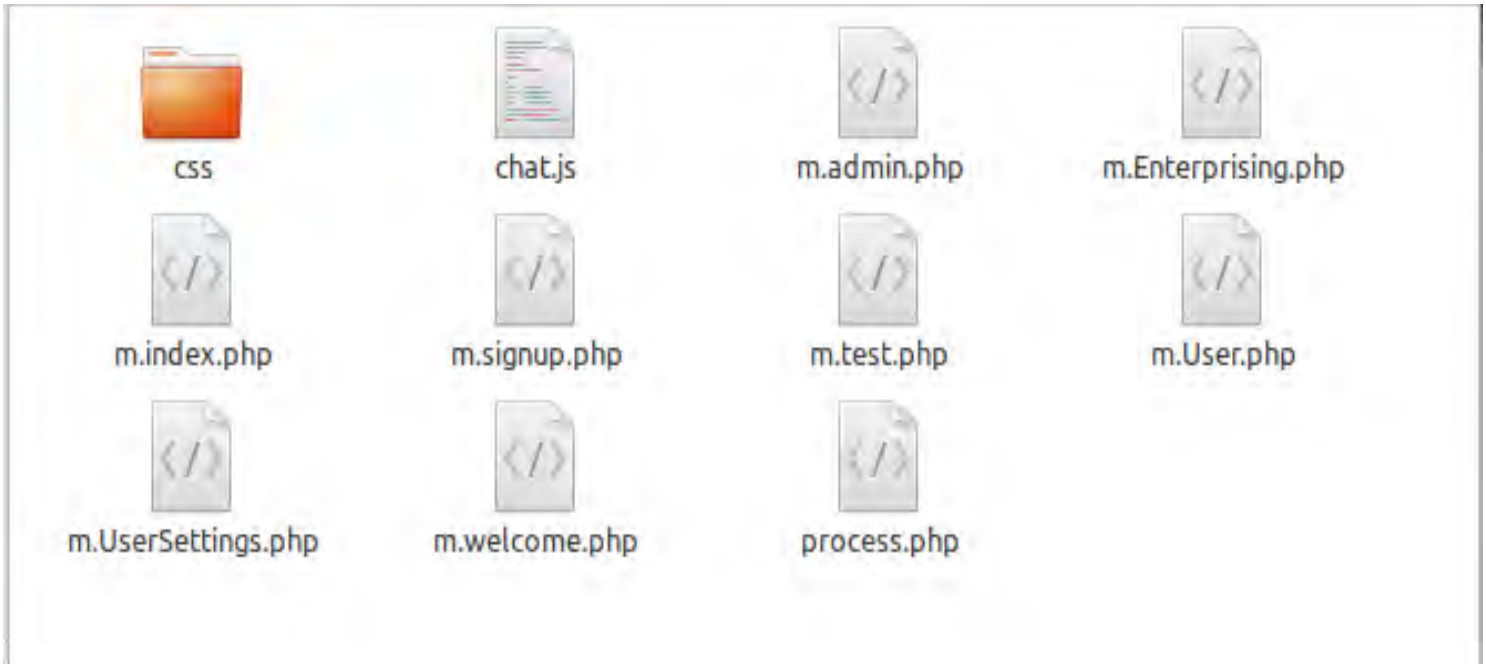
Ο υποκατάλογος css περιέχει τον κώδικα css για κάθε php αρχείο της έκδοσης της εφαρμογής για τον υπολογιστή.

Ο υποκατάλογος arangodb.php περιέχει το πρόγραμμα-πελάτης (client) ArangoDB PHP το οποίο είναι γραμμένο σε PHP και δεν έχει άλλες εξαρτήσεις. Το ArangoDB PHP client είναι ένα API (διεπαφή προγραμματισμού εφαρμογών) που μας επιτρέπει να στέλνουμε και να ανακτούμε έγγραφα από την ArangoDB μέσω της εφαρμογή μας που χρησιμοποιεί γλώσσα προγραμματισμού PHP.



Σχήμα 3.1: Αρχεία της εφαρμογής CoffeeFun.

Πιο συγκεκριμένα ο υποκατάλογος mobile_version περιέχει όλα τα αρχεία κώδικα που τρέχουνε για χρήστες που χρησιμοποιούν κινητό τηλέφωνο ή παρόμοιες συσκευές με μικρή οθόνη.



Σχήμα 3.2: Αρχεία της mobile έκδοσης της εφαρμογής CoffeeFun.

Παρακάτω βλέπουμε τον κώδικα javascript με τον οποίο καταλαβαίνουμε με τι συσκευή έχει συνδεθεί ο χρήστης και αναλόγως να τρέξει η κατάλληλη έκδοση. Στην ουσία κοιτάμε εάν το πλάτος της οθόνης του χρήστη είναι μικρότερο ή ίσο με 800 πίξελς. Εάν ναι τότε θα τρέξει η έκδοση για κινητά τηλέφωνα ή μικρότερες συσκευές, διαφορετικά θα τρέξει η έκδοση για υπολογιστή.

```
<!-- Redirect to Mobile version  -->
  <script type="text/javascript">

      if (screen.width <= 800) {
      window.location = "m.index.php";
      }
      </script>
```

Σχήμα 3.3: Κώδικας Javascript για την αναγνώριση της συσκευής του χρήστη.

3.3 Sessions

Οι συνεδρίες (sessions) επιτρέπουν μέσω της δέσμης ενεργειών (script) php να αποθηκεύσουμε δεδομένα στον απομακρυσμένο διακομιστή (server) που θα μας χρειαστούν αργότερα ακόμα και σε διαφορετικά αιτήματα php σελίδων. Κάθε συνεδρία έχει διαφορετικό αναγνωριστικό το οποίο αποστέλλεται στο πρόγραμμα περιήγησης του πελάτη ως cookie ή ως μεταβλητή $ _GET .Οι περίοδοι σύνδεσης λήγουν όταν ο χρήστης κλείσει το πρόγραμμα περιήγησης ή όταν ο διακομιστής ιστού διαγράψει τις πληροφορίες της περιόδου σύνδεσης ή όταν ο προγραμματιστής καταστρέφει ρητά τη συνεδρία. Οι περίοδοι σύνδεσης είναι πολύ χρήσιμες για την προστασία των δεδομένων που ο χρήστης δεν θα μπορούσε να διαβάσει ή να γράψει, ειδικά όταν ο προγραμματιστής της PHP δεν θέλει να δώσει πληροφορίες στα cookies, καθώς είναι ευανάγνωστα. Οι συνεδρίες μπορούν να ελέγχονται από την καθολική μεταβλητή (superglobal variable) $ _SESSION. Τα δεδομένα που είναι αποθηκευμένα σε αυτόν τον πίνακα είναι σταθερά καθ 'όλη τη διάρκεια της περιόδου σύνδεσης. Οι περίοδοι σύνδεσης είναι πολύ πιο εύχρηστες από τα cookies, κάτι που βοηθά τους προγραμματιστές της PHP πολύ. Συνήθως, οι συνεδρίες χρησιμοποιούνται για τη σύνδεση χρηστών, τα καροτσάκια αγορών και άλλες προσθήκες που απαιτούνται για την ομαλή περιήγηση. Η δέσμη ενεργειών (script) PHP μπορεί εύκολα να ελέγξει το cookie της συνεδρίας που αποστέλλεται και να ελέγχει τα δεδομένα της συνόδου [6].

3.4 HTTP cookies

Τα **cookies** είναι μικρά αρχεία κειμένου τα οποία αποθηκεύονται στον περιηγητή μας κατά την πλοήγησή μας στο διαδίκτυο. Σκοπός τους είναι να ειδοποιούν τον ιστότοπο που επισκέπτεται ο χρήστης, για την προηγούμενη δραστηριότητά του. Συνήθως περιγράφουν στοιχεία μας όπως όνομα χρήστη (user name) και συνθηματικό πρόσβασης (password) με σκοπό κατά την επίσκεψή μας στον ίδιο ιστότοπο αργότερα, να μας "θυμάται" και να μην χρειάζεται να κάνουμε login [7].

3.5 Σύνδεση με την ArangoDB

Ένα απο τα πιο σημαντικά κομμάτια κώδικα υλοποιείται στο αρχείο db.php. Πρόκειται για το αρχείο που περιέχει τον κώδικα για την σύνδεση με την NoSQL βάση δεδομένων (ArangoDB) καθώς και τις πιο σημαντικές συναρτήσεις που υλοποιούν τις λειτουργίες της ιστοσελίδας.

Στο παρακάτω σχήμα βλέπουμε τον κώδικα για την σύνδεση με την ArangoDB του αρχείου db.php.

```php
require __DIR__ . '/arangodb-php/autoload.php';


// set up some aliases for less typing later
use ArangoDBClient\Collection as ArangoCollection;
use ArangoDBClient\CollectionHandler as ArangoCollectionHandler;
use ArangoDBClient\Connection as ArangoConnection;
use ArangoDBClient\ConnectionOptions as ArangoConnectionOptions;
use ArangoDBClient\DocumentHandler as ArangoDocumentHandler;
use ArangoDBClient\Document as ArangoDocument;
use ArangoDBClient\Exception as ArangoException;
use ArangoDBClient\Export as ArangoExport;
use ArangoDBClient\ConnectException as ArangoConnectException;
use ArangoDBClient\ClientException as ArangoClientException;
use ArangoDBClient\ServerException as ArangoServerException;
use ArangoDBClient\Statement as ArangoStatement;
use ArangoDBClient\UpdatePolicy as ArangoUpdatePolicy;

// set up some basic connection options
$connectionOptions = [
    // database name
    ArangoConnectionOptions::OPTION_DATABASE => 'users',
    // server endpoint to connect to
    ArangoConnectionOptions::OPTION_ENDPOINT => 'tcp://127.0.0.1:8529',
    // authorization type to use (currently supported: 'Basic')
    ArangoConnectionOptions::OPTION_AUTH_TYPE => 'Basic',
    // user for basic authorization
    ArangoConnectionOptions::OPTION_AUTH_USER => 'root',
    // password for basic authorization
    ArangoConnectionOptions::OPTION_AUTH_PASSWD => '',
    // connection persistence on server. can use either 'Close' (one-time connections) or 'Keep-Alive' (re-used co
    ArangoConnectionOptions::OPTION_CONNECTION => 'Keep-Alive',
    // connect timeout in seconds
    ArangoConnectionOptions::OPTION_TIMEOUT => 3,
    // whether or not to reconnect when a keep-alive connection has timed out on server
    ArangoConnectionOptions::OPTION_RECONNECT => true,
```

Σχήμα 3.4: Κώδικας PHP για σύνδεση με την ArangoDB.

3.6 Βασικές συναρτήσεις του συστήματος

Οι συναρτήσεις του αρχείου db.php που υλοποιούν τις κύριες λειτουργίες της ιστοσελίδας είναι οι εξής :

Βασική προϋπόθεση είναι ότι το e-mail του χρήστη αποθηκεύεται σε session και έτσι δεν χρειάζεται να το περνάμε σε όλες τις συναρτήσεις ως παράμετρο.

- **register:**

Παίρνει ως παραμέτρους το όνομα,επίθετο, e-mail και τον κωδικό που δημιουργήθηκε τυχαία και στάλθηκε στο e-mail του χρήστη και κάνει την καταχώρηση στην βάση δεδομένων στην συλλογή με όνομα users.

- **RegisterEnter**

Παίρνει ως παραμέτρους το όνομα της καφετέριας, την πόλη,την φωτογραφία,τα σχόλια καθώς και την πλήρη διεύθυνση της καφετέριας και κάνει την καταχώρηση στην βάση δεδομένων στην συλλογή με όνομα Company.

- **Login**

Παίρνει ως παραμέτρους το e-mail χρήστη και τον κωδικό του και ελέγχει εάν ο χρήστης είναι εγγεγραμμένος στην βάση δεδομένων επιστρέφοντας 1 εάν υπάρχει ή 0 αλλιώς.

- **Checkmail**

Παίρνει ως παραμέτρους το e-mail χρήστη και ελέγχει εάν το e-mail χρήστη είναι ήδη εγγεγραμμένο στην βάση δεδομένων επιστρέφοντας 1 εάν υπάρχει ή 0 αλλιώς.

- **Showcompanies**

Παίρνει ως παραμέτρους την πόλη καθώς και τον τύπο συσκευής του χρήστη (desktop,mobile) και αναλόγως εμφανίζει τις διαθέσιμες καφετέριες ή φιλτραρισμένες ανάλογα με την πόλη που επέλεξε ο χρήστης.

- **Vote**

Παίρνει ως παράμετρο τον αριθμό απο αστέρια που αξιολόγησε ο χρήστης την συγκεκριμένη καφετέρια και τον προσθέτει στις υπόλοιπες αξιολογήσεις χρηστών για την συγκεκριμένη καφετέρια.

- **Insertvote**

Παίρνει ως παράμετρο το όνομα της καφετέριας και κάνει καταχώρηση στην συλλογή vote τον χρήστη που έκανε αξιολόγηση την συγκεκριμένη καφετέρια.

- **Checkvote**

Παίρνει ως παραμέτρους το e-mail του χρήστη καθώς και το όνομα της καφετέριας και ελέγχει εάν ο χρήστης έχει ήδη αξιολογήσει την συγκεκριμένη καφετέρια.

- **Dropvote**

Δεν έχει παραμέτρους. Σκοπός της είναι να διαγράφει ολόκληρη την συλλογή vote μία φορά την ημέρα και συγκεκριμένη ώρα. Καλείται μέσω του εργαλείου Cron το οποίο είναι ένας χρονοπρογραμματιστής εργασιών.

- **Returncomment**

Παίρνει ως παράμετρο το μοναδικό id ή αλλιώς key value της καφετέριας και επιστρέφει τα σχόλια που έγραψε ο χρήστης-ιδιοκτήτης.

- **Showsettings**

Παίρνει ως παράμετρο το e-mail του χρήστη και επιστρέφει τα στοιχεία του λογαριασμού του (όνομα, επίθετο, e-mail), καθώς και τα ονόματα των καφετεριών που είναι συνδεδεμένες με τον λογαριασμό του.

- **Editsettings**

Παίρνει ως παράμετρο μία συμβολοσειρά και μία λέξη-κλειδί. Αναλόγως την λέξη- κλειδί αλλάζει το όνομα,επίθετο η τον κωδικό του χρήστη με την καινούργια συμβολοσειρά που έδωσε ο χρήστης.

- **RemoveEnter**

Παίρνει ως παράμετρο το όνομα της καφετέριας και την διαγράφει απο την συλλογή Company.

- **InfoEnter**

Παίρνει ως παραμέτρους το όνομα της καφετέριας και τον τύπο συσκευής του χρήστη (desktop, mobile) και εμφανίζει τις καφετέριες που συνδέονται με τον χρήστη.

- **EditEnter**

Παίρνει ως παραμέτρους μία συμβολοσειρά, μία λέξη κλειδί και το μοναδικό id ή key value της καφετέριας. Αναλόγως την λέξη κλειδί αλλάζει το όνομα,τοποθεσία,φωτογραφία ή τα σχόλια της καφετέριας.

- **Returnaddress**

Παίρνει ως παράμετρο το ο μοναδικό id ή key value της καφετέριας και επιστρέφει την πλήρη διεύθυνση της καφετέριας.

- **Isadmin**

Παίρνει ως παράμετρο το e-mail του χρήστη και επιστρέφει 1 εάν ο χρήστης είναι διαχειριστής ή 0 διαφορετικά.

- **Waitforauthentication**

Παίρνει ως παράμετρο τον τύπο συσκευής του χρήστη (desktop, mobile) και εμφανίζει στον χρήστη-διαχειριστή τις καφετέριες που αναμένουν επιβεβαίωση καταχώρισης στο σύστημα.

- **AcceptEnter**

Παίρνει ως παράμετρο το όνομα της καφετέριας και ανανεώνει το στοιχείο authenticate απο 0 σε 1 της καφετέριας. Με αυτό τον τρόπο η καφετέρια γίνεται δεκτή στο σύστημα και εμφανίσιμη στους υπόλοιπους χρήστες.

Ο κώδικας των συναρτήσεων βρίσκεται στο παράρτημα Α' .

Σημαντικό αρχείο είναι επίσης το editsettings.php. Χρησιμοποιείται για την αντιστοίχιση συναρτήσεων που βρίσκονται στο αρχείο db.php οι οποίες καλούνται απο διάφορα αρχεία μέσω Ajax και έχουν ως παραμέτρους μεταβλητές javascript. Αυτός είναι και ο τρόπος που χρησιμοποιούμε για μεταφορά javascript μεταβλητών σε php συναρτήσεις. Στο παρακάτω σχήμα βλέπουμε τον κώδικα του αρχείου editsettings.php.

```
switch($_POST['function']){

        case('editsettings'):

  editsettings($_POST["txt1"],$_POST["val"]);
  break;


      case('removeenter'):
  RemoveEnter($_POST["companyname"]);
  break;


      case('infoenter'):
  InfoEnter($_POST["companyname"],$_POST["mode"]);
  break;


      case('editenter'):
  EditEnter($_POST["txt1"],$_POST["val"],$_POST["text"],$_POST["companyid"]);
  break;


      case('acceptenter'):
  AcceptEnter($_POST["companyname"]);
  break;

}
```

Σχήμα 3.5: Κώδικας του αρχείου editsettings.php.

Και στο επόμενο σχήμα βλέπουμε ένα δείγμα το πώς καλούμε php συναρτήσεις μέσα απο javascript με την χρήση Ajax.

```
    jQuery.ajax({
      type: "POST",
      url: 'editsettings.php',
      data: {'function': 'editsettings',txt1,val},
      success:function(data) {
      alert('successfully changed !');
      window.location.href = "UserSettings.php";
      }
    });

});
```

Σχήμα 3.6: Παράδειγμα χρήσης Ajax.

Πιο συγκεκριμένα καλούμε με την μέθοδο POST την συνάρτηση editsettings που βρίσκεται στο αρχείο db.php και στέλνουμε δύο μεταβλητές την txt1 και val αφού γίνει πρώτα η αντιστοίχιση απο το αρχείο editsettings.php..

3.6.1 Άλλα αρχεία κώδικα του συστήματος

welcome.php : Περιέχει κώδικα PHP και javascript. Ο κώδικας αυτός τρέχει με το που κάνει είσοδο ο χρήστης. Η λειτουργία του είναι να εμφανίζει όλες ή με βάση το φίλτρο τις εγγεγραμμένες καφετέριες στο σύστημα.

UserSettings.php: Περιέχει κώδικα PHP και javascript. Η βασική του λειτουργία είναι να δίνει την δυνατότητα στον χρήστη να μπορεί να τροποποιήσει τα στοιχεία του λογαριασμού του καθώς και των καφετεριών που είναι συνδεδεμένες με τον λογαριασμό του.

User.php: Περιέχει κώδικα PHP και javascript. Τρέχει με το που ο χρήστης πατήσει πάνω στην φωτογραφία κάποιας καφετέριας και δίνει την δυνατότητα στον χρήστη να δεί την ακριβή τοποθεσία της καφετέριας μέσω των χαρτών της google, να συνομιλεί γραπτώς με άλλους χρήστες που έχουνε επιλέξει την ίδια καφετέρια, να αξιολογήσει την καφετέρια με αστεράκια καθώς και να δεί τα σχόλια του ιδιοκτήτη της καφετέριας.

upload.php: Περιέχει κώδικα PHP. Είναι ο κώδικας που τρέχει κατα το ανέβασμα φωτογραφίας στην διαδικασία δημιουργίας νέας καφετέριας.

test.php: Περιέχει κώδικα javascript. Είναι ο κώδικας που τρέχει όταν πατήσουμε στο κουμπί "click to open the map" κατα την διαδικασία δημιουργίας νέας καφετέριας και μας δίνει την δυνατότητα να επιλέξουμε την ακριβή τοποθεσία της καφετέριας χρησιμοποιώντας τους χάρτες της google.

signup.php: Περιέχει κώδικα PHP και javascript. Είναι ο κώδικας που τρέχει κατα την διάρκεια δημιουργίας νέου λογαριασμού.

process.php: Περιέχει κώδικα PHP. Είναι ο κώδικας που τρέχει όταν ο χρήστης στέλνει γραπτό μήνυμα.

login.php: Περιέχει κώδικα PHP και javascript. Ο σκοπός του είναι να πέρνει τον κωδικο και το e-mail του χρήστη και στην συνέχεια να ελέγχει εάν είναι εγγεγραμμένος στο σύστημα.

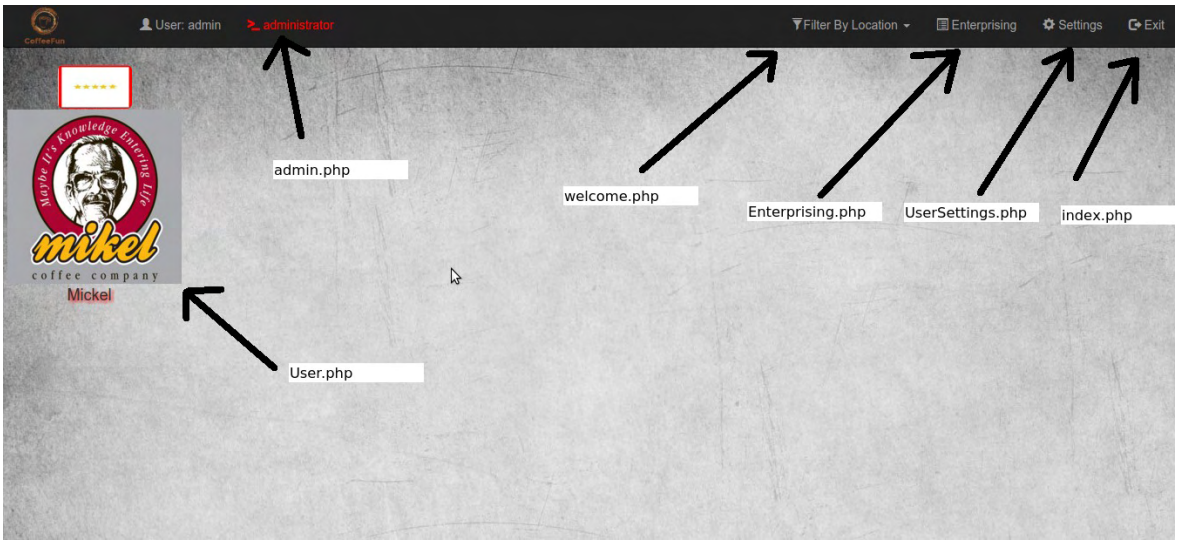
index.php: Περιέχει κώδικα PHP και javascript.Είναι ο κώδικας που τρέχει με το που ο χρήστης μπεί ή κάνει έξοδο στην εφαρμογή .Κύριος σκοπός του είναι η υποδοχή του χρήστη καθώς και η διαγραφή του session κατα την έξοδο του χρήστη.

Enterprising.php: Περιέχει κώδικα PHP και javascript. Είναι ο κώδικας που τρέχει κατα την διάρκεια δημιουργίας νέας καφετέριας. Ζητάει τα στοιχεία της καφετέριας,τα ελέγχει για την ορθότητα τους και στην συνέχεια καταχωρεί την νέα καφετέρια.

dropvote.php: Περιέχει κώδικα PHP. Καλεί την συνάρτηση dropvote που βρίσκεται στο αρχείο db.php.Αυτή η συνάρτηση καταστρέφει και ξανα δημιουργεί εκ` νέου την συλλογή vote. Το αρχείο dropvote.php τρέχει συγκεκριμένη ώρα και κάθε μέρα μέσω του εργαλείου των Linux που ονομάζεται Cron.

admin.php: Περιέχει κώδικα PHP και javascript. Ο κώδικας του τρέχει μόνο απο κάποιον λογαριασμό διαχειριστή και σκοπός του είναι να δίνει την δυνατότητα στους διαχειριστές του συστήματος να εγκρίνουν ή να απορρίπτουν τις αιτήσεις δημιουργίας νέων καφετεριών.

Στο παρακάτω σχήμα τα βελάκια δείχνουν το αρχείο που θα τρέξει αναλόγως την λειτουργία που θα επιλέξει ο χρήστης.



Σχήμα 3.7: Παρουσίαση και επεξήγηση του αρχείου welcome.php

Κεφάλαιο 4: Παρουσίαση Εφαρμογής

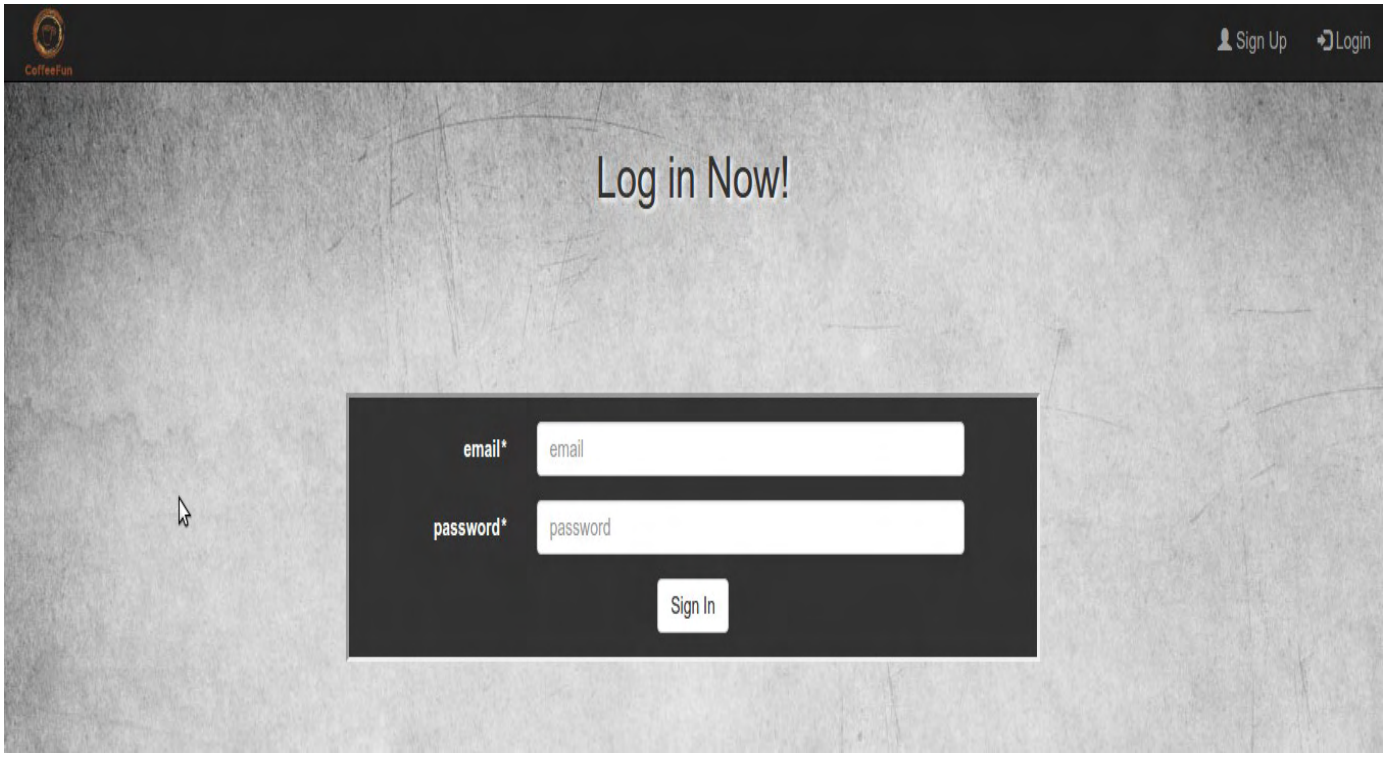
4.1 Περιβάλλον ανάπτυξης

Η εφαρμογή υλοποιήθηκε σε γραφικό περιβάλλον Linux Ubuntu 14.04. Ως πρόγραμμα επεξεργασίας κειμένου χρησιμοποιήθηκε η Kate, ως εξυπηρετητής ο Apache/2.4.26 , έκδοση PHP 7.2.0alpha3, ως βάση δεδομένων η ArangoDB έκδοση 3.0.12 καθώς και ο PHP ArangoDB client έκδοση 3.2.0.

4.2 Παρουσίαση της εφαρμογής

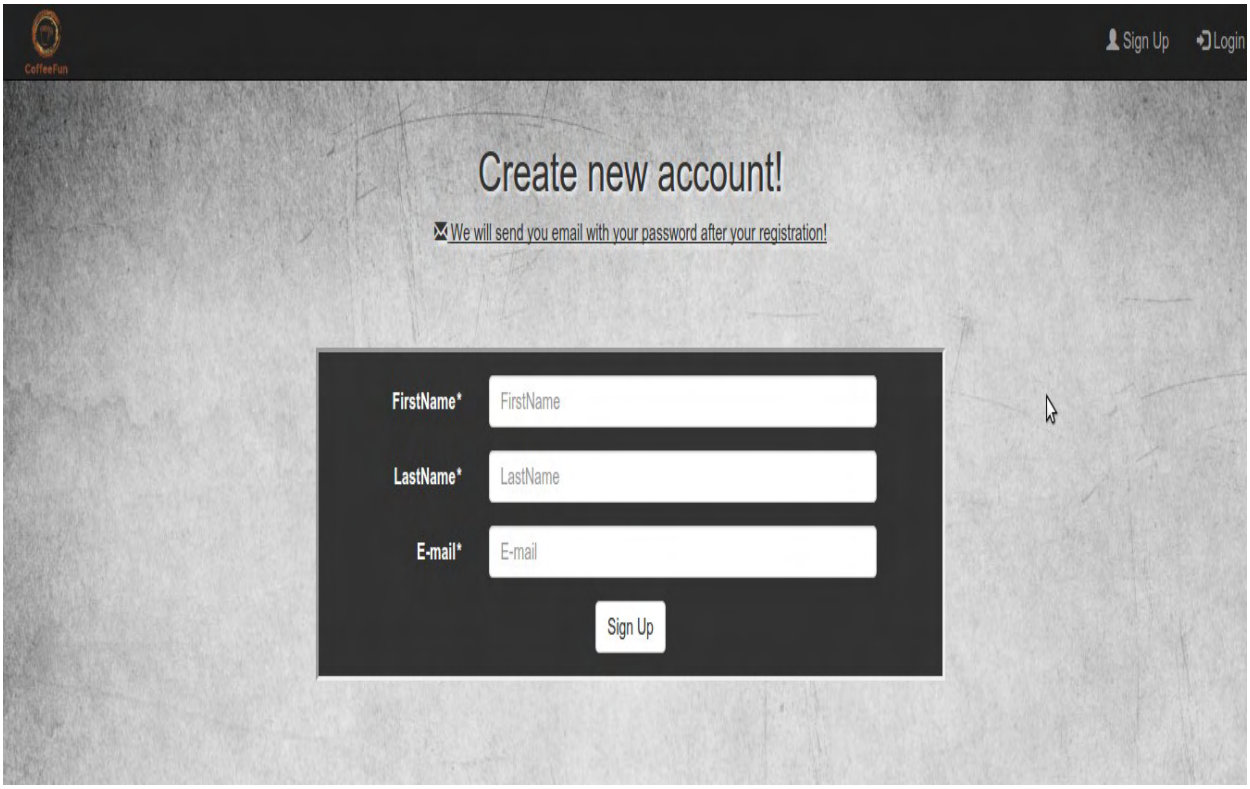
Η εφαρμογή τρέχει στον σύνδεσμο coffeefun.servehttp.com. Κατά την είσοδο του χρήστη στην εφαρμογή του ζητείται να συμπληρώσει το e-mail του και τον κωδικό του για να έχει πρόσβαση στις λειτουργίες της εφαρμογής. Σε περίπτωση που ο χρήστης δεν έχει δημιουργήσει λογαριασμό μπορεί να δημιουργήσει δίνοντας ένα e-mail, όνομα και επίθετο και στην συνέχεια λαμβάνει στο e-mail του τον κωδικό του (ο κωδικός δημιουργείται απο την εφαρμογή) με τον οποίο μπορεί στην συνέχεια να κάνει είσοδο στην εφαρμογή.

Στην παρακάτω εικόνα βλέπουμε την αρχική σελίδα που βλέπει ο χρήστης.



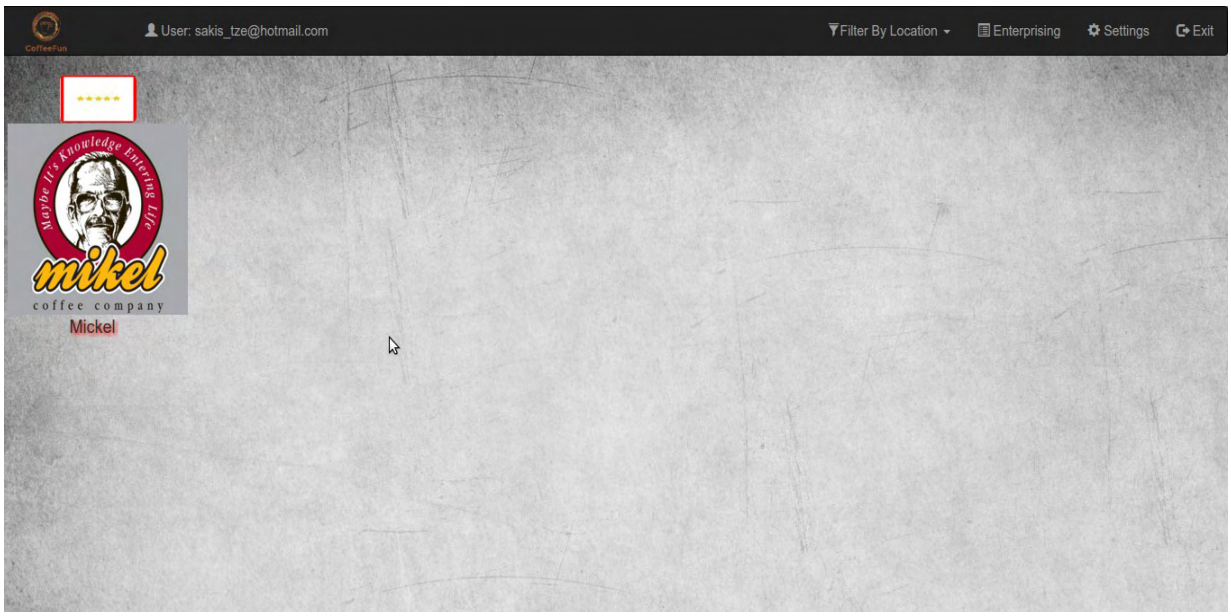
Σχήμα 4.1: Παρουσίαση του αρχείου login.php

Στην παρακάτω εικόνα βλέπουμε την σελίδα που εμφανίζεται για την δημιουργία νέου λογαριασμού.



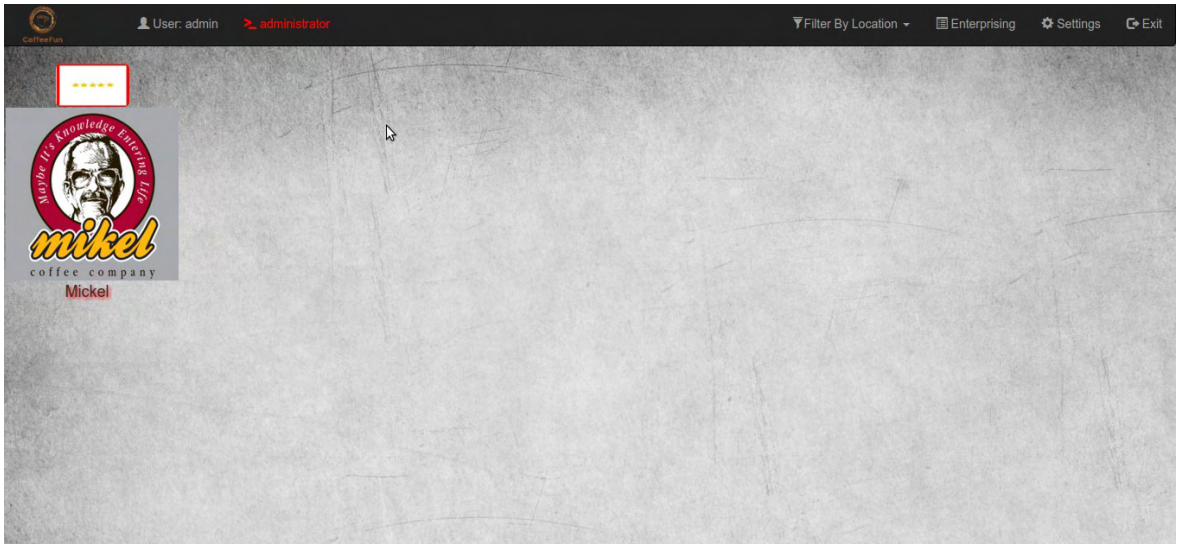
Σχήμα 4.2: Παρουσίαση του αρχείου signup.php.

Αφού ο χρήστης κάνει επιτυχώς είσοδο με τον λογαριασμό του στην εφαρμογή μπορεί πλέον να δεί τις διαθέσιμες καφετέριες που είναι καταχωριμένες στο σύστημα. Ο χρήστης έχει την επιλογή να φιλτράρει τις διαθέσιμες καφετέριες ανάλογα με την πόλη που επιθυμεί ,να δημιουργήσει νέα καφετέρια,να τροποποιήσει τα στοιχεία του λογαριασμού του καθώς και να δει-τροποποιήσει τα στοιχεία των καφετεριών που είναι συνδεδεμένες με τον λογαριασμό του.



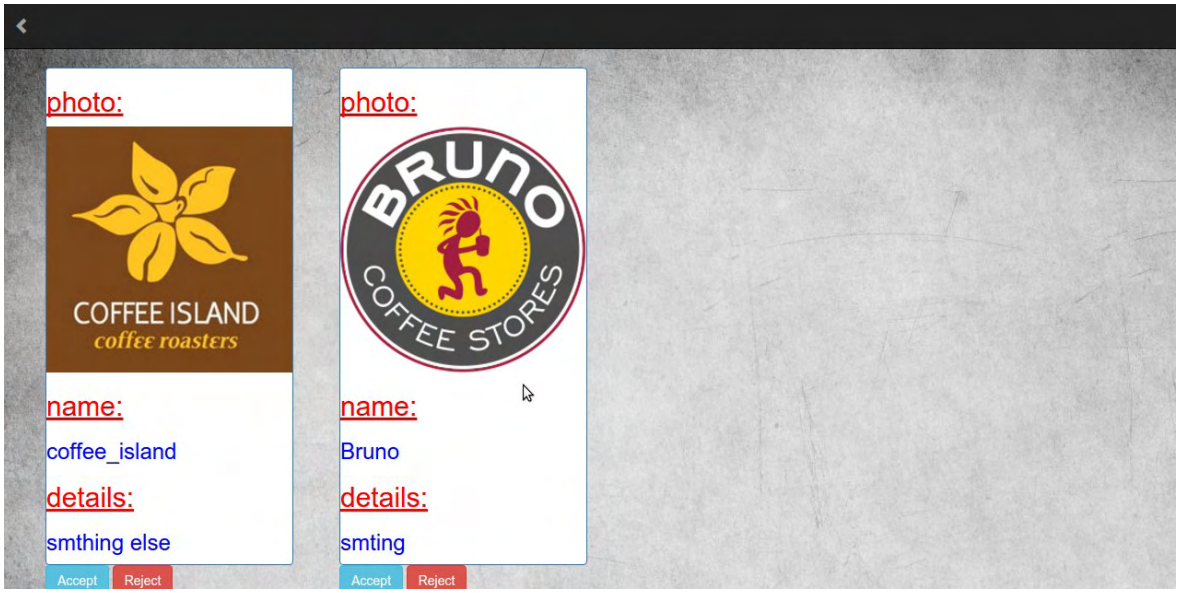
Σχήμα 4.3: Παρουσίαση του αρχείου welcome.php

Οι χρήστες διαχωρίζονται σε διαχειριστές ή μη. Οι διαχειριστές έχουνε μία ακόμα επιπλέον λειτουργία η οποία φαίνεται με τα κόκκινα γράμματα στην παρακάτω φωτογραφία.



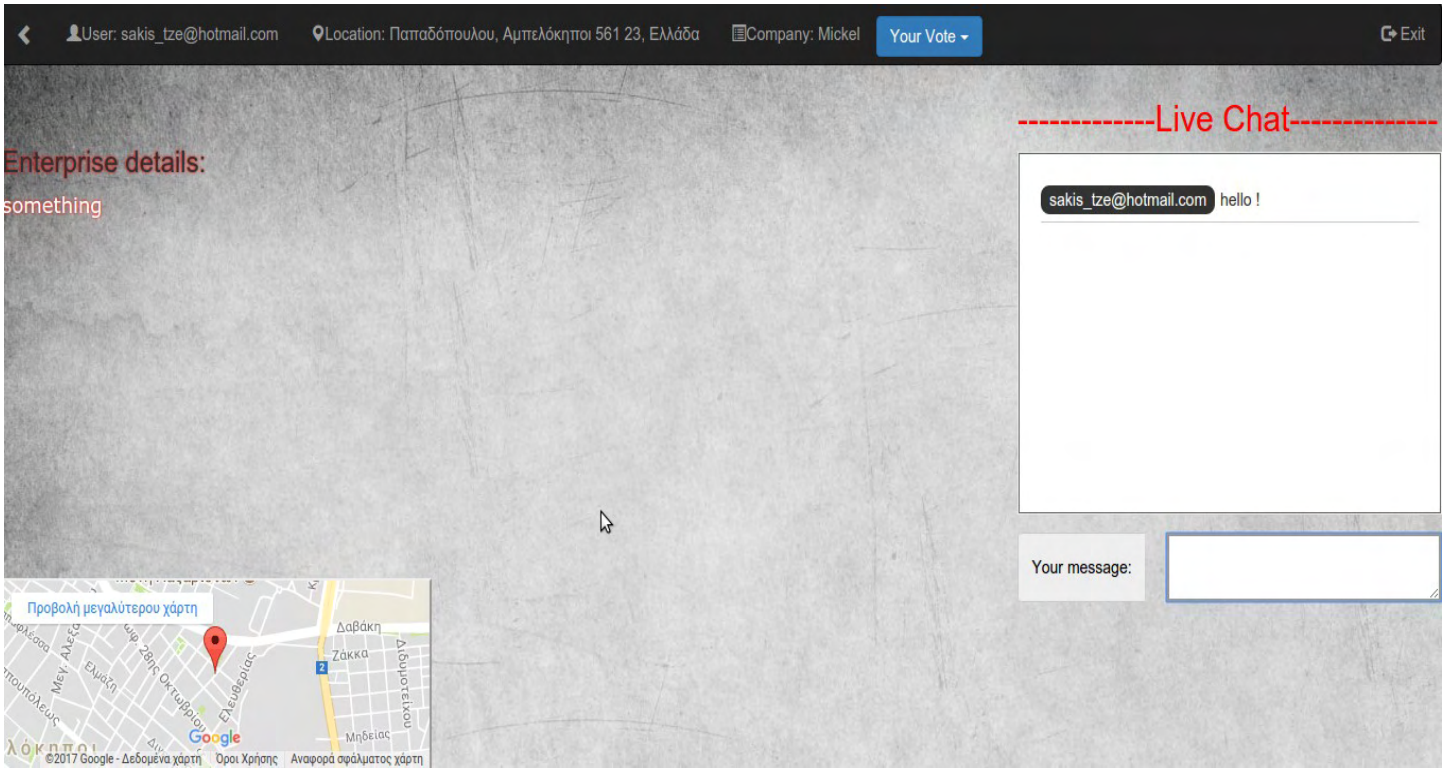
Σχήμα 4.4: Εμφάνισης της σελίδας welcome.php σε διαχειριστή του συστήματος.

Η λειτουργία αυτή τους επιτρέπει να δεχτούν ή να απορρίψουν μία νέα επιχείρηση που έχει κάνει αίτημα εγγραφής στο σύστημα.



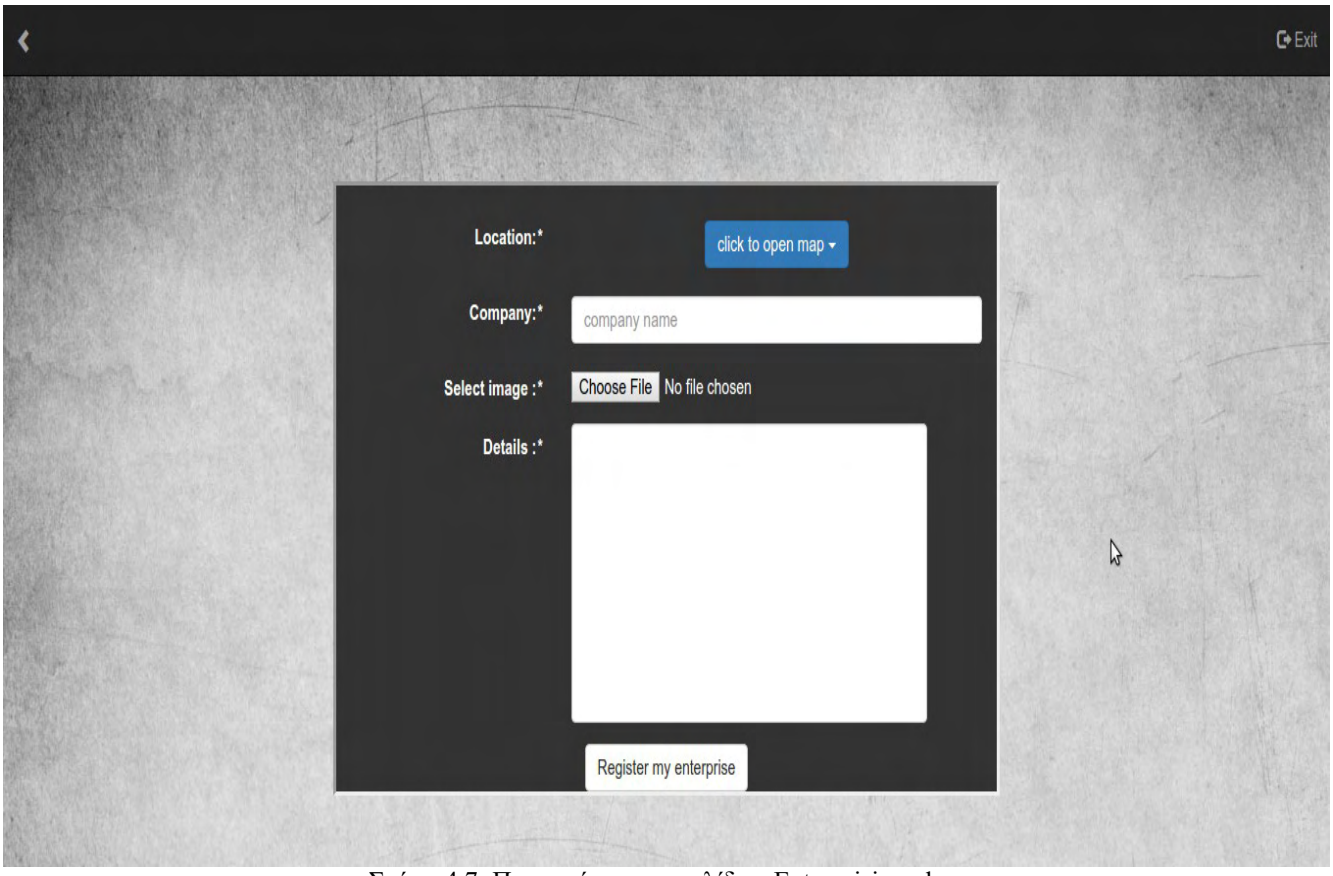
Σχήμα 4.5: Πίνακας του διαχειριστή συστήματος.

Οι χρήστες αφού διαλέξουν την καφετέρια που θέλουνε (πατώντας πάνω στην φωτογραφία της καφετέριας) μεταφέρονται στη σελίδα User.php, όπου μπορούνε να αξιολογήσουν την καφετέρια με αστεράκια απο το ένα έως το πέντε,να δούνε την ακριβή τοποθεσία της καφετέριας στους χάρτες της google,να δούνε τις λεπτομέρειες της καφετέριας που έχει γράψει ο ιδιοκτήτης καθώς και να ανταλλάξουν ζωντανά γραπτά μηνύματα με άλλους χρήστες που έχουνε επιλέξει την ίδια καφετέρια.



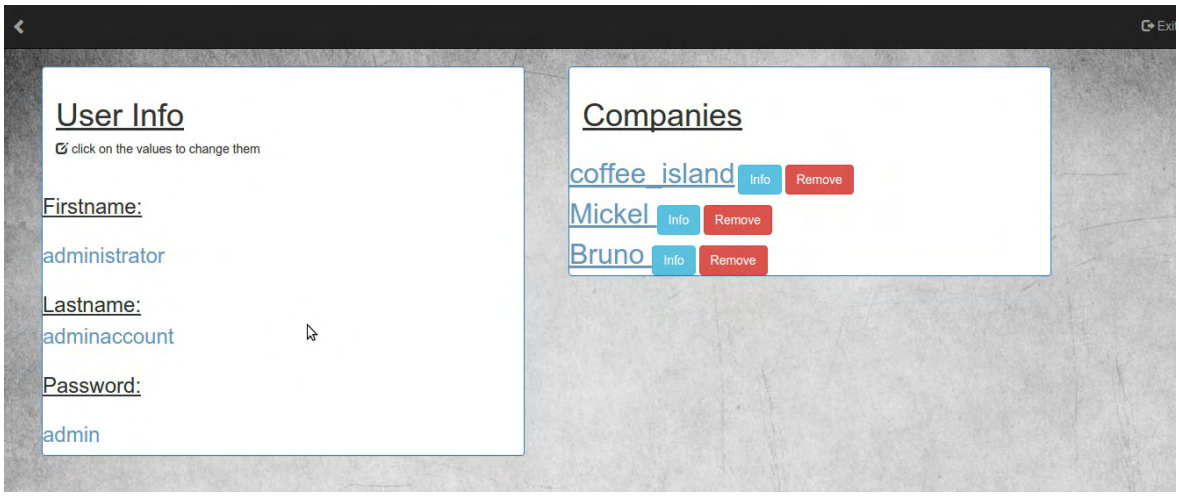
Σχήμα 4.6: Παρουσίαση της σελίδας User.php

Επιλέγοντας την λειτουργία Enterprising ο χρήστης μεταφέρεται στην σελίδα Enterprising.php όπου μπορεί να δημιουργήσει νέα καφετέρια δίνοντας την τοποθεσία,το όνομα,μία φωτογραφία και ένα σχόλιο. Εφόσον ο χρήστης δημιουργήσει την καφετέρια πρέπει κάποιος διαχειριστής του συστήματος να επικυρώσει την ένταξη της νέας καφετέριας στο σύστημα ώστε να γίνει ορατή απο τους υπόλοιπους χρήστες.

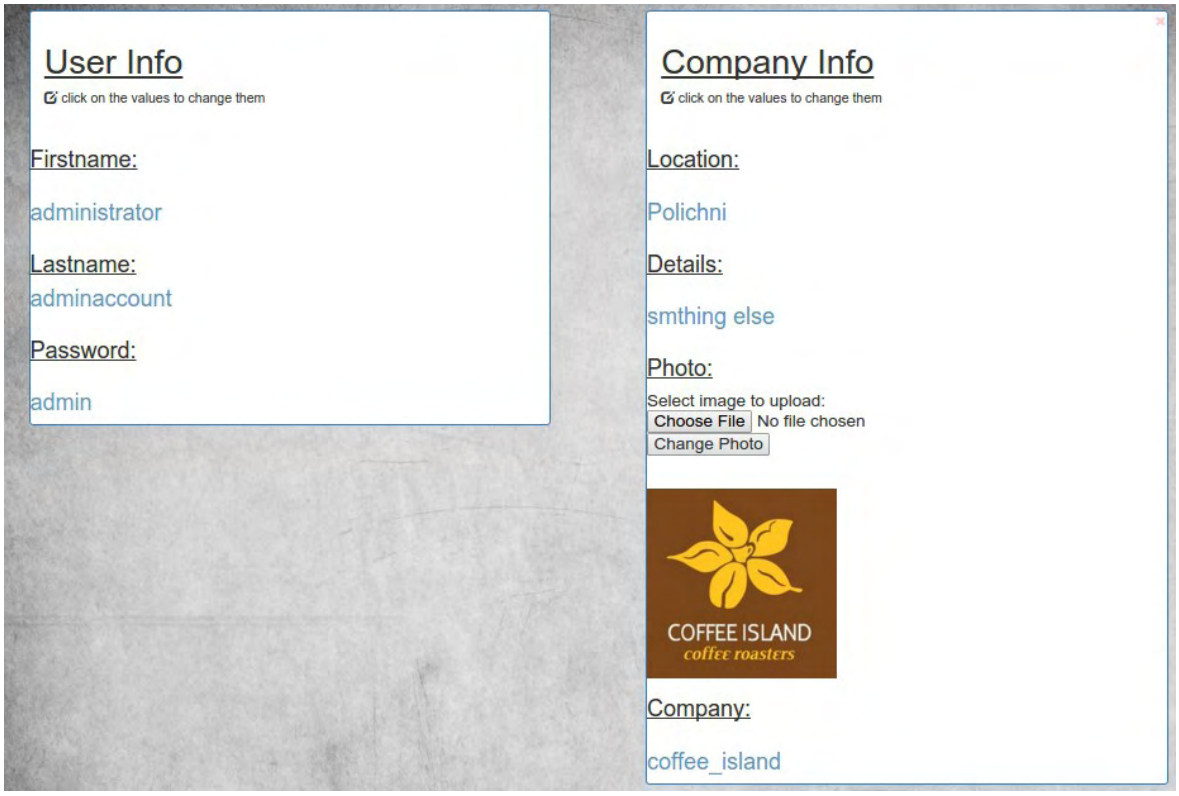


Σχήμα 4.7: Παρουσίαση της σελίδας Enterprising.php.

Επιλέγοντας την λειτουργία Settings ο χρήστης μεταφέρεται στην σελίδα UserSettings.php, όπου μπορεί να δει και να τροποποιήσει τα στοιχεία του λογαριασμού του καθώς και τα στοιχεία των καφετεριών που συνδέονται με τον λογαριασμό του.



Σχήμα 4.8: Παρουσίαση της σελίδας UserSettings.php.



Σχήμα 4.9: Παρουσίαση της σελίδας UserSettings.php.

4.3 Αξιολόγηση

Η εφαρμογή δεν αντιμετωπίζει κάποιο πρόβλημα καθ΄ όλη την διάρκεια της εκτέλεσής της, μπορεί να ανεχτεί ένα ικανοποιητικό φόρτο απο ταυτόχρονους πελάτες και να αποδίδει άψογα, ανάλογα βεβαίως και το hardware και η σύνδεση που χρησιμοποιεί ο διακομιστής.

Παρακάτω θα συγκρίνουμε τον τρόπο που υλοποιούμε την σύνδεση των καφετεριών με τους λογαριασμούς των χρηστών χρησιμοποιώντας NoSQL βάση δεδομένων, σε σχέση με το πώς θα υλοποιούταν σε μία SQL βάση δεδομένων.

Ο τρόπος που χρησιμοποιούμε είναι ο εξής: Αρχικά κοιτάμε για τον κάθε χρήστη, με βάση το μοναδικό e-mail που έχει ο καθένας μετράμε στην συλλογή Company πόσες εταιρείες είναι εγγεγραμμένες με το e-mail του δίνοντας σε κάθε μία έναν αυξανόμενο αριθμό ξεκινώντας απο το 0. Αυτό γίνεται στην συνάρτηση showsettings που βρίσκεται στο αρχείο db.php.

```
foreach ($cursor as $check => $value) {

    if (strcmp($email, json_decode($value)->email) == 0){
       $_SESSION["userid"] = json_decode($value)->id;
    //$email=json_decode($value)->email;
    //$id=json_decode($value)->id;
    //$firstname=json_decode($value)->firstname;
    //$lastname=json_decode($value)->lastname;
       $temp=0;
           foreach ($cursor1 as $check1 => $value1) {
             if(strcmp($email,json_decode($value1)->email) == 0){
               $val = json_encode($value1->company);  //$json={"var1":"value1","var2":"value2"}
           //echo $value;
               $value =substr($value ,0,-1);
               $value .=',"company'.$temp.'":'.$val.'}';
               $temp=$temp+1;
            }
           }


     //echo '<script>window.location.href = "m.index.php";</script>';

   return  $value;
```

Σχήμα 4.10: Κομμάτι κώδικα από την συνάρτηση showsettings.

Στην συνέχεια στο αρχείο ShowSettings.php καλούμε την παραπάνω συνάρτηση και μας δείχνει τις εν λόγω καφετέριες.

```
$item=showsettings($_SESSION['email']);

 $email=json_decode($item)->email;
 //$id=json_decode($item)->id;
 $firstname=json_decode($item)->firstname;
 $lastname=json_decode($item)->lastname;
 $password=json_decode($item)->password;
 //$company=json_decode($item)->company1;



echo '<div id="parent_div_1">';
echo '<div class="panel-group" >';
echo ' <div class="panel panel-primary" >';
echo '  <div id="smt"class="panel-body"><h1><u>User Info</u></h1><h5><span class="glyphicon glyphicon-edit"></span> click on the values to c
echo '<h3 id ="info"  ><u>Firstname: </u></h3><h3 id="info1"  val="fname" style="color:#6897bb;display: inline-block;margin-right: 100px; ">
echo '<h3 id ="info"  ><u>Lastname: </u></h3><h3 id="info2"  val="lname" style="color:#6897bb;margin-right: 100px; ">'.$lastname.'</h3 > ';
//echo '<h3 id ="info" ><u>Email: </u></h3><h3 id="info3" val="email" style="color:#6897bb;display: inline-block;margin-right: 100px; ">'.$e
echo '<h3 id ="info"  ><u>Password: </u></h3><h3 id="info4" val="password" style="color:#6897bb;display: inline-block;margin-right: 100px; '

echo ' </div>';
echo ' </div>';
echo ' </div>';
echo ' <div id="parent_div_2">';
echo '<div class="panel panel-primary" >';
echo '  <div class="panel-body"><h1><u>Companies</u></h1></div>';
    $data = json_decode($item, true);
for($i = 0; $i <= count($data)-6; $i++){
    $str="company".$i;
    $company=json_decode($item)->$str;
echo '<font style="display: inline;color:#6897bb;" size="6" ><u>'.$company.'</u></font> <button type="button" value="'.$company.'" class="bt
echo '<br>';
}
echo '</div>';
echo '</div>';
```

Σχήμα 4.11: Κώδικας που καλούμε την συνάρτηση showsettings..

Σε αντίθεση αν χρησιμοποιούσαμε μια SQL βάση δεδομένων μία υλοποίηση θα ήτανε η εξής: Κατασκευή δύο table, όπου το ένα θα περιέχει τα στοιχεία του χρήστη-ιδιοκτήτη και το άλλο τις καφετέριες. Στην συνέχεια δημιουργούμε μία σχέση μεταξύ τους με την χρήση ενός εξωτερικού κλειδιού (foreign key), το οποίο θα μπορούσε να είναι είτε το μοναδικό id είτε το e-mail του χρήστη-ιδιοκτήτη.

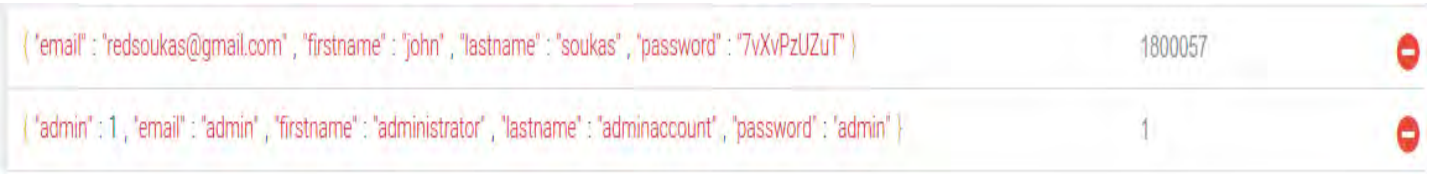
Το συμπέρασμα είναι ότι με την χρήση NoSQL βάσης δεδομένων έχουμε πιο εύχρηστη και επεκτάσιμη την λειτουργία της εφαρμογής, λόγω του ότι μπορούμε να προσθαφαιρέσουμε πεδία και στοιχεία στην βάση μας σε αντίθεση με την δυσκολία που εμφανίζεται με την χρήση μίας σχεσιακής (SQL) βάσης δεδομένων.

Κεφάλαιο 5: Συμπεράσματα και Επεκτάσεις

5.1 Συμπεράσματα

Αναφέρουμε τα τελικά συμπεράσματα που αποκτήθηκαν καθ' όλη τη διάρκεια της μελέτης, έρευνας, σχεδίασης και κατασκευής αυτή της διπλωματικής εργασίας. Υπάρχει αρκετό υλικό στον παγκόσμιο ιστό που εξηγεί την διαφορά ανάμεσα στις σχεσιακές βάσεις δεδομένων και στις μη σχεσιακές βάσεις δεδομένων. Αντιθέτως δεν υπάρχει αρκετό υλικό για την πρακτική υλοποίηση και κατασκευή των μη σχεσιακών βάσεων δεδομένων όπως ούτε μεγάλη κοινότητα χρηστών.

Θα ήθελα να επισημάνω ότι οι μη σχεσιακές βάσεις δεδομένων είναι πολύ πιο εύκολες στην διαχείρισή τους καθώς επιστρέφουν αντικείμενα τύπου Json στα ερωτήματα που τις θέτουμε. Επίσης είναι ευέλικτες και επεκτάσιμες επειδή τα δεδομένα δεν αποθηκεύονται σε tables αλλα σε collections. Η διαφορά τους είναι οτι στα collections μπορούμε να έχουμε εγγραφές με απεριόριστο αριθμό στοιχείων,πράγμα το οποίο δεν συμβαίνει στις σχεσιακές βάσεις δεδομένων. Στο παρακάτω σχήμα βλέπουμε ότι η πρώτη εγγραφή έχει 4 στοιχεία, ενώ η δεύτερη έχει 5 .



Σχήμα 5.1: Παράδειγμα από collection.

5.2 Μελλοντικές επεκτάσεις

Στο μέλλον θα μπορούσαμε να προσθέσουμε τις εξής λειτουργίες:

- Οι χρήστες να μπορούνε να γράφουνε την κριτική τους
- Οι χρήστες να έχουνε την δυνατότητα προσωπικής και ζωντανής γραπτής επικοινωνίας με άλλους χρήστες .
- Οι χρήστες να προσθέτουνε χρήστες "φίλους" στον λογαριασμό τους άλλους χρήστες καθώς και να βλέπουνε πότε είναι ενεργοί αλλά και την δυνατότητα γραπτής συνομιλίας μεταξύ τους.
- Οι διαχειριστές να διακρίνονται σε επίπεδα. Δηλαδή να διαφέρουν τα δικαιώματα απο διαχειριστή σε διαχειριστή.
- Η εφαρμογή να υποστηρίζει εκτός απο καφετέριες και άλλα καταστήματα,όπως για παράδειγμα εστιατόρια,ξενοδοχεία,κλπ.
- Η εφαρμογή να μπεί στον Ωκεανό ώστε να δοκιμαστεί με πολλούς χρήστες και καφετέριες με σκοπό να δούμε την συμπεριφορά της εφαρμογής σε πραγματικές συνθήκες.

Αναφορές και Βιβλιογραφία

[1] https://en.wikipedia.org/wiki/Web_application

[2] http://www.thegeekstuff.com/2014/01/sql-vs-nosql-db/

[3] https://www.javatpoint.com/sql-vs-nosql

[4] https://www.arangodb.com/why-arangodb/sql-aql-comparison/

[5] http://www.thewindowsclub.com/difference-sql-nosql-comparision

[6] https://en.wikibooks.org/wiki/PHP_Programming/Sessions

[7] https://el.wikipedia.org/wiki/HTTP_cookies

[8] https://en.wikipedia.org/wiki/Document_Object_Model

[9] https://github.com/arangodb/arangodb-php#description

[10] https://en.wikipedia.org/wiki/ArangoDB

[11] https://en.wikipedia.org/wiki/JQuery

[12] https://css-tricks.com/jquery-php-chat/

[13] *Adam Fowler (24 February 2015),* NoSQL For Dummies. *John Wiley & Sons*

[14] Πεχλιβανίδης Κωνσταντίνος (Απρίλιος 2013), («Σχεδιασμός και Ανάπτυξη μιας Διαδικτυακής Εφαρμογής για την Παροχή Υπηρεσιών Διαχείρισης και Έκδοσης Κοινοχρήστων σε Οικοδομές») Διπλωματική εργασία

[15] Rebecca M. Riordan (5 Sept 2008),Head First Ajax

[16] Lynn Beighley, Michael Morrison (June 2009),Head First PHP & MySQL

[17] Elisabeth Freeman, Eric Freeman (June 2009),Head First HTMLwith CSS & XHTML

[18] https://www.arangodb.com/why-arangodb/arangodb-web-ui/

Παράρτημα Α'

Κώδικας του αρχείου db.php στο οποίο υλοποιούνται οι βασικές συναρτήσεις που αναφέραμε στην ενότητα 3.6 .

```
<?php

session_start();




// use the following line when using Composer
// require __DIR__ . '/vendor/composer/autoload.php';

// use the following line when using git
require __DIR__ . '/arangodb-php/autoload.php';


// set up some aliases for less typing later
use ArangoDBClient\Collection as ArangoCollection;
use ArangoDBClient\CollectionHandler as ArangoCollectionHandler;
use ArangoDBClient\Connection as ArangoConnection;
use ArangoDBClient\ConnectionOptions as ArangoConnectionOptions;
use ArangoDBClient\DocumentHandler as ArangoDocumentHandler;
use ArangoDBClient\Document as ArangoDocument;
use ArangoDBClient\Exception as ArangoException;
use ArangoDBClient\Export as ArangoExport;
use ArangoDBClient\ConnectException as ArangoConnectException;
use ArangoDBClient\ClientException as ArangoClientException;
```

```
use ArangoDBClient\ServerException as ArangoServerException;
use ArangoDBClient\Statement as ArangoStatement;
use ArangoDBClient\UpdatePolicy as ArangoUpdatePolicy;

// set up some basic connection options
$connectionOptions = [
    // database name
    ArangoConnectionOptions::OPTION_DATABASE => 'users',
    // server endpoint to connect to
    ArangoConnectionOptions::OPTION_ENDPOINT => 'tcp://127.0.0.1:8529',
    // authorization type to use (currently supported: 'Basic')
    ArangoConnectionOptions::OPTION_AUTH_TYPE => 'Basic',
    // user for basic authorization
    ArangoConnectionOptions::OPTION_AUTH_USER => 'root',
    // password for basic authorization
    ArangoConnectionOptions::OPTION_AUTH_PASSWD => '',
    // connection persistence on server. can use either 'Close' (one-time connections) or
'Keep-Alive' (re-used connections)
    ArangoConnectionOptions::OPTION_CONNECTION => 'Keep-Alive',
    // connect timeout in seconds
    ArangoConnectionOptions::OPTION_TIMEOUT => 3,
    // whether or not to reconnect when a keep-alive connection has timed out on server
    ArangoConnectionOptions::OPTION_RECONNECT => true,
    // optionally create new collections when inserting documents
    ArangoConnectionOptions::OPTION_CREATE => true,
    // optionally create new collections when inserting documents
    ArangoConnectionOptions::OPTION_UPDATE_POLICY =>
ArangoUpdatePolicy::LAST,
];


    // turn on exception logging (logs to whatever PHP is configured)
    ArangoException::enableLogging();
```

```
    $connection = new ArangoConnection($connectionOptions);

            // create a new document
 function register($var1_value,$var2_value,$var3_value,$var4_value){
    global $connection;
            try {

   $handler = new ArangoDocumentHandler($connection);
    $user = new ArangoDocument();

// use set method to set document properties
    $user->set('firstname', $var1_value);
    $user->set('lastname', $var2_value);
    $user->set('email', $var3_value);
    $user->set('password', $var4_value);

      // send the document to the server
    $id = $handler->save('users', $user);

      // check if a document exists
    $result = $handler->has('users', $id);
      //var_dump($result);

      // print the document id created by the server
      //var_dump($id);
      //var_dump($user->getId());
} catch (ArangoConnectException $e) {
  print 'Connection error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoClientException $e) {
  print 'Client error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoServerException $e) {
```

```
  print 'Server error: ' . $e->getServerCode() . ':' . $e->getServerMessage() . ' ' . $e->getMessage() . PHP_EOL;
}

  }



 function registerEnter($company,$location,$photo,$comment,$fulladdress){
    global $connection;
            try {

   $handler = new ArangoDocumentHandler($connection);
   $user = new ArangoDocument();

// use set method to set document properties
    $user->set('company', $company);
    $user->set('Location', $location);
    $user->set('email', $_SESSION["email"] );
    $user->set('photo', $photo);
    $user->set('comment',$comment);
    $user->set('address',$fulladdress);
    $user->set('authenticate',0);
    // send the document to the server
    $id = $handler->save('Company', $user);

    // check if a document exists
    $result = $handler->has('Company', $id);
      //echo $id;

      // print the document id created by the server
      //var_dump($id);
      //var_dump($user->getId());
```

```
} catch (ArangoConnectException $e) {
  print 'Connection error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoClientException $e) {
  print 'Client error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoServerException $e) {
  print 'Server error: ' . $e->getServerCode() . ':' . $e->getServerMessage() . ' ' . $e-
>getMessage() . PHP_EOL;
}

  }


  function login($email,$password){
    global $connection;
  //global $email;
  //global $password;
    $query = 'FOR x IN users  RETURN { email: x.email,password: x.password}';

    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();
    $resultingDocuments = array();

//$str ="{\"email\":\"{$email}\",\"password\":\"{$password}\"}";
```

```
  foreach ($cursor as $check => $value) {
     if (strcmp(json_decode($value)->email, $email) == 0 && strcmp(json_decode($value)-
>password, $password) == 0){
       $exists=$exists+1;
    }else{
       $exists=$exists+0;
     }
  }
return $exists;
}



  function checkmail($email){
    global $connection;

    $query = 'FOR x IN users  RETURN x.email';

    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();
    $resultingDocuments = array();

    $sum=0;
```

```
  foreach ($cursor as $check => $value) {
     if (strcmp($email, $value) == 0){
           $sum=1;
     }else{

     }
}

return $sum;

}

  function showcompanies($loc,$mode){
      global $connection;




      if(strcmp($loc, "All") !== 0){
      $stri=" x.Location == '$loc' ";
      $query = 'FOR x IN Company  FILTER '.$stri.'  RETURN {"photo" :
x.photo,"company"
:x.company,"location":x.Location,"res":x.res,"id":x._key,"authenticate":x.authenticate}';
      }else{
      $query = 'FOR x IN Company  RETURN {"photo" : x.photo,"company"
:x.company,"location":x.Location,"res":x.res,"id":x._key,"authenticate":x.authenticate}';
      }
      $statement = new  ArangoStatement(
      $connection,
      array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
```

```
      "sanitize"  => true
          )
    );
```

```
     $cursor = $statement->execute();

    $resultingDocuments = array();

    $a=array();

foreach ($cursor as $check => $value) {
  $pieces = json_decode($value);
if($pieces->authenticate==1){
  if(strcmp($mode,"desktop")==0){
  $temp='uploads/'.$pieces->photo;
}else{
  $temp='../uploads/'.$pieces->photo;
}

  $gk=floor($pieces->res);

  echo '<div class="buttn" style="float: left;width: auto;    " >';

if($gk==0){

  if(strcmp($mode,"desktop")==0){
    echo '<img src="stars/0star.jpg"   class="img-rounded">';
  }else{
```

```
  echo '<img src="../stars/0star.jpg"   class="img-rounded">';
 }

}elseif($gk==1 ){

 if(strcmp($mode,"desktop")==0){
  echo '<img src="stars/1star.jpg"   class="img-rounded">';
 }else{
  echo '<img src="../stars/1star.jpg"   class="img-rounded">';
 }

}elseif($gk==2 ){

        if(strcmp($mode,"desktop")==0){
           echo '<img src="stars/2star.jpg"   class="img-rounded">';
        }else{
           echo '<img src="../stars/2star.jpg"   class="img-rounded">';
        }

 }elseif($gk==3 ){

        if(strcmp($mode,"desktop")==0){
           echo '<img src="stars/3star.jpg"   class="img-rounded">';
        }else{
           echo '<img src="../stars/3star.jpg"   class="img-rounded">';
        }

 }elseif($gk==4 ){

        if(strcmp($mode,"desktop")==0){
           echo '<img src="stars/4star.jpg"   class="img-rounded">';
        }else{
           echo '<img src="../stars/4star.jpg"   class="img-rounded">';
```

```
        }
    }
    else{

        if(strcmp($mode,"desktop")==0){
            echo '<img src="stars/5star.jpg"   class="img-rounded">';
        }else{
            echo '<img src="../stars/5star.jpg"   class="img-rounded">';
        }

    }

    echo '<div>';
  if(strcmp($mode,"desktop")==0){
    echo '<button id="'.$pieces->id.'"   value="'.$pieces->location.'" name="'.$pieces-
>company.'" class="buttn" onclick="myFunction(id,name,value)" ><img src="'.$temp.'"
alt="HTML5 Icon" class="companyimages"></button>';
  }else{
    echo '<button id="'.$pieces->id.'"   value="'.$pieces->location.'" name="'.$pieces-
>company.'" class="buttn" onclick="myFunction(id,name,value)" ><img src="'.$temp.'"
alt="HTML5 Icon" class="companyimages"></button>';
  }
   echo '</div>';

    echo '<p id="buttnimage" class="buttn">'.$pieces->company.'</p>';
    echo '</div>';


    $a[$check] = $pieces->location;
 }
 }
```

```
  $a1 = array_unique($a);

  if(strcmp($loc, "All") == 0){


    foreach ($a1 as $key => $value) {


    echo "<script>";
    echo "$(document).ready(function(){ ";
    echo "$(function() {";
    echo "var change = function( txt ) {";
    echo " $('#locationss').append( '<li><a id =locations  href=#  >' + txt + '</a></li> <li
class=divider></li>' );";
    echo " };";
    echo " change('$value'); ";
    echo " });";
    echo " }); ";
    echo "</script>";
    }

  }


}



function vote($value){
    global $connection;
```

```
    $handler = new ArangoDocumentHandler($connection);

   // update a document
    $stars = $handler->get('Company', $_COOKIE['id']);
    $stars->star =$stars->star + (int) $value;
    $result = $handler->update($stars);



    $counter = $handler->get('Company', $_COOKIE['id']);
    $counter->sum =$counter->sum + 1;
    $result = $handler->update($counter);

    $total = $handler->get('Company', $_COOKIE['id']);
    $total->res =$stars->star / $counter->sum;
    $result = $handler->update($total);

}



function insertvote($company){
    global $connection;
            try {

   $handler = new ArangoDocumentHandler($connection);
   $user = new ArangoDocument();



    // use set method to set document properties
   $user->set('email', $_SESSION["email"] );
```

```
   $user->set('company', $company);









      // send the document to the server
    $id = $handler->save('vote', $user);

      // check if a document exists
    $result = $handler->has('vote', $id);

} catch (ArangoConnectException $e) {
  print 'Connection error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoClientException $e) {
  print 'Client error: ' . $e->getMessage() . PHP_EOL;
} catch (ArangoServerException $e) {
  print 'Server error: ' . $e->getServerCode() . ':' . $e->getServerMessage() . ' ' . $e-
>getMessage() . PHP_EOL;
  }
}

function checkvote($email,$company){
    global $connection;
    $query = 'FOR x IN vote  RETURN {email:x.email,company:x.company}';

    $statement = new  ArangoStatement(
    $connection,
```

```
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();
    $resultingDocuments = array();


    $sum=0;

  foreach ($cursor as $check => $value) {

     if (strcmp($email, json_decode($value)->email) == 0 && strcmp($company,
json_decode($value)->company) == 0){
            $sum=1;




     }else{

     }

  }

return  $sum;

}
```

```
function dropvote(){

  global $connection;
  $collectionHandler = new ArangoCollectionHandler($connection);
  $collectionHandler->drop('vote');

  $voteCollection = new ArangoCollection();
  $voteCollection->setName('vote');
  $collectionHandler->create($voteCollection);

}
```

```
 function returncomment($id){
    global $connection;

    $query = 'FOR x IN Company  RETURN {"id":x._key,"comment":x.comment}';

    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();
```

```
    $resultingDocuments = array();



 foreach ($cursor as $check => $value) {

     if (strcmp($id, json_decode($value)->id) == 0){
            $sum=json_decode($value)->comment;

            return  $sum;

     }else{

     }


 }

return $sum;

}



function showsettings($email){
    global $connection;

    $query = 'FOR x IN users  RETURN
{"email":x.email,"firstname":x.firstname,"lastname":x.lastname,"password":x.password,"i
d":x._key}';
    $query1 = 'FOR y IN Company RETURN {"company":y.company,"email":y.email}';
```

```
$statement = new  ArangoStatement(
$connection,
array(
    "query"     => $query,
    "count"     => true,
    "batchSize" => 10,
    "sanitize"  => true
        )
);

$statement1 = new  ArangoStatement(
$connection,
array(
    "query"     => $query1,
    "count"     => true,
    "batchSize" => 10,
    "sanitize"  => true
        )
);
$cursor = $statement->execute();
$cursor1 = $statement1->execute();
$resultingDocuments = array();
```

```
foreach ($cursor as $check => $value) {

    if (strcmp($email, json_decode($value)->email) == 0){
        $_SESSION["userid"] = json_decode($value)->id;
        $temp=0;
        foreach ($cursor1 as $check1 => $value1) {
```

```
                if(strcmp($email,json_decode($value1)->email) == 0){
                    $val = json_encode($value1->company);
                    $value =substr($value ,0,-1);
                    $value .=',"company'.$temp.'":'.$val.'}';
                    $temp=$temp+1;
                }
               }


    return  $value;

     }else{

     }

}


return $value;

}


function editsettings($txt1,$val){

    global $connection;
    $handler = new ArangoDocumentHandler($connection);

 if(strcmp($val,"fname")==0){

    $id = $handler->get('users', $_SESSION['userid']);
    $id->firstname =$txt1;
```

```
    $result = $handler->update($id);
 }elseif(strcmp($val,"lname")==0){
    $lname= $handler->get('users', $_SESSION['userid']);
    $lname->lastname =$txt1;
    $result = $handler->update($lname);

    }
}
 else{
    $password= $handler->get('users', $_SESSION['userid']);
    $password->password =$txt1;
    $result = $handler->update($password);




    }




}


function RemoveEnter($company){
    global $connection;
    $query = 'FOR y IN Company RETURN
{"company":y.company,"id":y._key,"email":y.email}';


    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
```

```
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();

    $handler = new ArangoDocumentHandler($connection);



foreach ($cursor as $check => $value) {

  if(strcmp($company,json_decode($value)->company)==0 &&
(strcmp($_SESSION['email'],json_decode($value)->email)==0 ||
isadmin($_SESSION["email"]))){
    $theId=json_decode($value)->id;

  }
}

     try {
        $result = $handler->removeById('Company', $theId);
    } catch (\ArangoDBClient\ServerException $e) {
        $e->getMessage();
    }



}
```

```
function InfoEnter($company,$mode){

    global $connection;

    $query = 'FOR y IN Company RETURN
{"Location":y.Location,"comment":y.comment,"company":y.company,"id":y._key,"email"
:y.email,"photo":y.photo}';


    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

  $cursor = $statement->execute();

  $handler = new ArangoDocumentHandler($connection);



foreach ($cursor as $check => $value) {

  if(strcmp($company,json_decode($value)->company)==0 &&
strcmp($_SESSION['email'],json_decode($value)->email)==0){

    $location1=json_decode($value)->Location;
    $companyid=json_decode($value)->id;
    $comment1=json_decode($value)->comment;
```

```
   $company1=json_decode($value)->company;
   $email1=json_decode($value)->email;
   $photo=json_decode($value)->photo;









  }


}



     try {
        $result = $handler->removeById('Company', $theId);
    } catch (\ArangoDBClient\ServerException $e) {
        $e->getMessage();
    }

    echo '<div id="parent_div_3">';
    echo '<div class="panel-group" >';
    echo ' <div class="panel panel-primary" >';
 if(strcmp($mode,"desktop")==0){
    echo '<button type="button"  style="color:red" class="close"  aria-label="Close"
onClick="window.location=\'UserSettings.php\';" ><span aria-
hidden="true">&times;</span></button>';
 }else{
    echo '<button type="button"  style="color:red" class="close"  aria-label="Close"
onClick="window.location=\'m.UserSettings.php\';" ><span aria-
hidden="true">&times;</span></button>';
 }
```

```
    echo '  <div id="smt1" class="panel-body"><h1><u>Company
Info</u></h1><h5><span class="glyphicon glyphicon-edit"></span> click on the values
to change them </h5></div>';
    echo '<h3 id ="infocompany"  ><u>Location: </u></h3><h3 id="infocompany1"
val="location" cid="'.$companyid.'" style="color:#6897bb;display: inline-block;margin-
right: 100px; ">'.$location1.'</h3 >';
    echo '<h3 id ="infocompany"  ><u>Details: </u></h3><h3 id="infocompany2"
val="comment" cid="'.$companyid.'" style="color:#6897bb;display: inline-block;margin-
right: 100px; ">'.$comment1.'</h3 > ';
 if(strcmp($mode,"desktop")==0){
    echo '<h3 id ="infocompany" ><u>Photo: </u><form action="upload.php"
method="post" enctype="multipart/form-data"><h4>';
 }else{
    echo '<h3 id ="infocompany" ><u>Photo: </u><form action="../upload.php"
method="post" enctype="multipart/form-data"><h4>';
 }
    echo 'Select image to upload:
    <input id="infocompany3" type="file" name="fileToUpload" cid="'.$companyid.'">
    <input id="aff" type="submit" value="Change Photo" name="submit"></h4>
</form><br>';
  if(strcmp($mode,"desktop")==0){
    echo '<img  src="uploads/'.$photo.'" height="200px" alt="Image preview...">';
  }else{
    echo '<img  src="../uploads/'.$photo.'" height="100px" alt="Image preview...">';
 }
    echo '<h3 id ="infocompany"  ><u>Company: </u></h3><h3 id="infocompany4"
val="company" cid="'.$companyid.'" style="color:#6897bb;display: inline-block;margin-
right: 100px; ">'.$company1.'</h3 >';



    echo ' </div>';
    echo ' </div>';
```

```
    echo ' </div>';



}









function EditEnter($txt1,$val,$text,$companyid){

    global $connection;
    $handler = new ArangoDocumentHandler($connection);

 if(strcmp($val,"location")==0){

    $location = $handler->get('Company', $companyid);
    $location->Location =$txt1;
    $result = $handler->update($location);
 }elseif(strcmp($val,"comment")==0){
    $comment= $handler->get('Company', $companyid);
    $comment->comment =$txt1;
    $result = $handler->update($comment);

    }
 elseif(strcmp($val,"company")==0){
    $company= $handler->get('Company', $companyid);
    $company->company =$txt1;
    $result = $handler->update($company);
```

```
 }elseif(strcmp($val,"photo")==0){
    $photo= $handler->get('Company', $companyid);
    $photo->photo =$txt1;
    $result = $handler->update($photo);
    }
 else{

    }
}


function returnaddress($companyid){

    global $connection;
    $query = 'FOR x IN Company  RETURN {"id":x._key,"faddress":x.address}';

    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
        "count"     => true,
        "batchSize" => 10,
        "sanitize"  => true
            )
    );

    $cursor = $statement->execute();
    $resultingDocuments = array();
```

```
foreach ($cursor as $check => $value) {

    if (strcmp($companyid, json_decode($value)->id) == 0){
        $sum=json_decode($value)->faddress;

        return  $sum;

    }else{

    }
}

return $sum;



}

function isadmin($email){


   global $connection;

   $query = 'FOR x IN users  RETURN {"id":x._key,"admin":x.admin,"email":x.email}';

   $statement = new  ArangoStatement(
   $connection,
   array(
      "query"     => $query,
      "count"     => true,
      "batchSize" => 10,
      "sanitize"  => true
         )
```

```
    );

    $cursor = $statement->execute();
    $resultingDocuments = array();




foreach ($cursor as $check => $value) {

     if (strcmp($email, json_decode($value)->email) == 0){
          $sum=json_decode($value)->admin;

          return  $sum;

     }
}


}

function  waitforauthentication($mode){

    global $connection;

    $query = 'FOR x IN Company  RETURN
{"id":x._key,"authenticate":x.authenticate,"company":x.company,"photo":x.photo,"comme
nt":x.comment}';

    $statement = new  ArangoStatement(
    $connection,
    array(
        "query"     => $query,
```

```
    "count"     => true,
    "batchSize" => 10,
    "sanitize"  => true
        )
);

$cursor = $statement->execute();
$resultingDocuments = array();



foreach ($cursor as $check => $value) {

   $pieces = json_decode($value);
   if(strcmp($mode,"desktop")==0){
   $temp='uploads/'.$pieces->photo;
   }else{
   $temp='../uploads/'.$pieces->photo;
   }
  if ($pieces->authenticate==0){


   echo '<div id="parent_div_1">';
   echo '<div class="panel-group" >';
   echo ' <div class="panel panel-primary" >';

   echo '<h2><u>photo:</u></h2><img id="authimg" src='.$temp.'  ></img>';
   echo '<h2><u>name:</u></h2><h3 >'.$pieces->company.'</h3>';
   echo '<h2><u>details:</u></h2><h3 >'.$pieces->comment.'</h3>';

   echo ' </div>';
```

```
      echo '<button type="button" value="'.$pieces->company.'" class="btn btn-info"
onclick="AcceptFunction(value)">Accept</button> <button  type="button"
value='.$pieces->company.' class="btn btn-danger"
onclick="RemoveFunction(value)">Reject</button>';

      echo ' </div>';
      echo ' </div>';

    }
  }
}




function AcceptEnter($company){



  global $connection;
   $query = 'FOR y IN Company RETURN
{"company":y.company,"id":y._key,"email":y.email,"authenticate":y.authenticate}';


   $statement = new  ArangoStatement(
   $connection,
   array(
      "query"     => $query,
      "count"     => true,
      "batchSize" => 10,
      "sanitize"  => true
         )
   );
```

```
    $cursor = $statement->execute();

    $handler = new ArangoDocumentHandler($connection);



  foreach ($cursor as $check => $value) {

    if(strcmp($company,json_decode($value)->company)==0 &&
isadmin($_SESSION["email"])){
      $theId=json_decode($value)->id;
    }
  }

   $ok = $handler->get('Company', $theId);
   $ok->authenticate =1;
   $result = $handler->update($ok);

}


?>
```

Παράρτημα Β'

Εγκατάσταση Εφαρμογής

Προαπαιτούμενα:

- **Λειτουργικό σύστημα Linux Ubuntu 14.04**
- **PHP 7.0**
- **Apache HTTP Server**

Εφόσον έχουμε εγκαταστήσει τα παραπάνω ανοιχτού κώδικα εργαλεία στον υπολογιστή μας, μπορούμε να προχωρήσουμε στην εγκατάσταση της βάσης δεδομένων ArangoDB.

Αρχικά ανοίγουμε μία κονσόλα και γράφουμε τις εξής εντολές με την ακόλουθη σειρά :

1. sudo apt-get install cmake make build-essential openssl python2.7 g++ gcc
2. git clone https://github.com/arangodb/arangodb
3. cd arangodb
4. git checkout 3.0
5. git pull
6. mkdir -p build
7. (cd build && cmake -DCMAKE_BUILD_TYPE=RelWithDebInfo ..)
8. git pull
9.(cd build && make -j4)

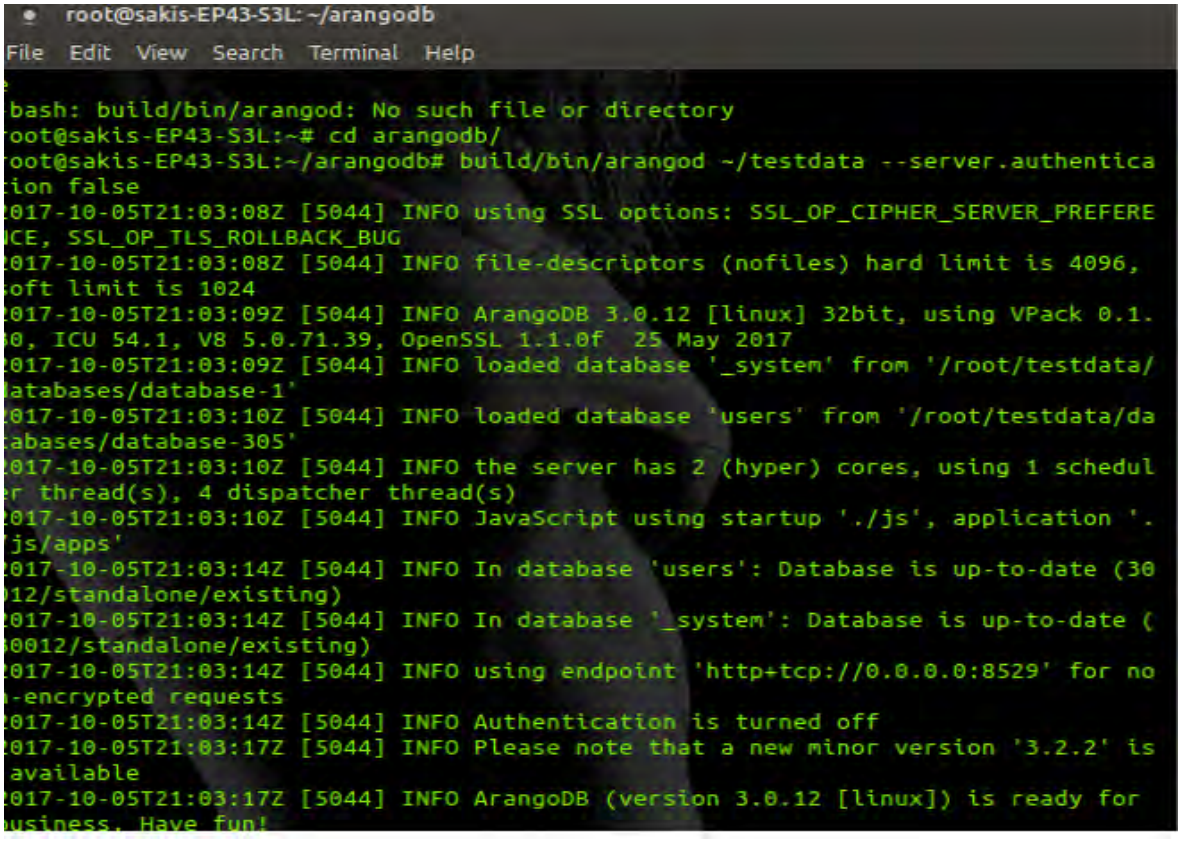
Στην συνέχεια κάθε φορά που θέλουμε να τρέξουμε και να χρησιμοποιήσουμε την βάση δεδομένων θα κάνουμε τα εξής:

1. Ανοίγουμε μία κονσόλα
2. Εκτελούμε την εντολή: sudo -i
3. Συμπληρώνουμε τον κωδικό μας και στην συνέχεια πατάμε Enter

4. Εκτελούμε την εντολή: cd arangodb/

5. Εκτελούμε την εντολή: build/bin/arangod ~/testdata --server.authentication false

Εάν όλα έχουνε πάει καλά, στην κονσόλα θα έχουμε κάτι τέτοιο.



Σχήμα 7: Έξοδος της κονσόλας κατά την έναρξη της ArangoDB.

Το επόμενο βήμα είναι να κατεβάσουμε την εφαρμογή καθώς και τον PHP arangodb-driver από τους συνδέσμους:

https://github.com/AM1228/CoffeeFun

https://github.com/arangodb/arangodb-php

Στην συνέχεια ξεπακετάρουμε τα αρχεία και τα τοποθετούμε στον Apache server δηλαδή στην τοποποθεσία: /var/www/html/

Ένα σημαντικό πράγμα που θα πρέπει να προσέξουμε είναι να συμβαδίζει η έκδοση της ArangoDB με τον PHP arangodb-driver.

Μένει να ρυθμίσουμε τον χρονοπρογραμματιστή διεργασιών Cron καθώς και το πρόγραμμα Postfix το οποίο θα στέλνει τα e-mails στους χρήστες με τους κωδικούς τους.

Για τον Cron: Ανοίγουμε μία κονσόλα και γράφουμε την εντολή: sudo crontab -e , συμπληρώνουμε τον κωδικό μας και Enter.
Στην συνέχεια στο κάτω μέρος του αρχείου γράφουμε: 45 10 * * * php /var/www/html/dropvote.php
, στην συνέχεια cntrl+O και τέλος cntrl+X.

Για το Postfix: Ανοίγουμε μία κονσόλα και γράφουμε τις εντολές :
1. sudo apt-get install postfix mailutils libsasl2-2 ca-certificates libsasl2-modules
2. sudo nano /etc/postfix/sasl_passwd
3. Προσθέτουμε την ακόλουθη γραμμή: [smtp.gmail.com]:587 USERNAME@gmail.com:PASSWORD , στην συνέχεια cntrl+O και τέλος cntrl+X.
4. Εκτελούμε την εντολή: sudo chmod 400 /etc/postfix/sasl_passwd
5. Εκτελούμε την εντολή**:**sudo postmap /etc/postfix/sasl_passwd
6.Εκτελούμε την εντολή**:** cat /etc/ssl/certs/Thawte_Premium_Server_CA.pem | sudo tee -a /etc/postfix/cacert.pem
7. Εκτελούμε την εντολή: sudo /etc/init.d/postfix reload