ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΊΔΡΥΜΑ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΠΟΛΥΜΕΣΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΤΙΤΛΟΣ:

Συστημα online ψηφοφορίας σε περιβαλλον android

ΒΑΣΙΛΕΙΟΣ ΓΕΩΡΓΙΟΣ ΜΠΑΚΟΠΟΥΛΟΣ(ΑΜ:3167)

Επιβλέπων καθηγητής :
Επιτροπή Αξιολόγησης :
Ημερομηνία Παρουσίασης:

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον Κύριο Νικόλαο Παπαδάκη για την καθοδήγηση της παρούσας πτυχιακής εργασίας και την οικογένεια μου για την πολύτιμη βοήθεια κατά τη διάρκεια της φοίτησης μου στη σχολή.

Abstract

In this thesis, the objective is to study and create a system through which you can place an online vote by using android powered devices.The system consists of two parts one part is the management area and the other one consists of the android application. The administration section, once the administrator has been authenticate, will be able to create and edit queries, add or remove new users and monitor the statistical results for each question that is available for e-voting. In the section which relates to the Android application, the user, by entering username and password, will have the opportunity to vote one of the questions that are available.

Σύνοψη

Στην παρούσα πτυχιακή εργασία στόχος είναι η μελέτη και δημιουργία ενός συστήματος μέσο του οποίου θα μπορεί να πραγματοποιηθεί μια ηλεκτρονική ψηφοφορία με τη χρήση έξυπνων κινητών συσκευών που διαθέτουν λειτουργικό σύστημα android .Το σύστημα αποτελείται από δύο μέρη το ένα μέρος αποτελείται από την περιοχή διαχείρισης και το άλλο μέρος αποτελείται από την εφαρμογή android. Στο τμήμα διαχείρισης αφού συνδεθεί ο διαχειριστής θα έχει τη δυνατότητα να δημιουργήσει και να επεξεργαστεί ερωτήματα προς ψηφοφορία ,επίσης θα μπορεί να προσθέσει η να αφαιρέσει νέους χρήστες και να παρακολουθήσει στατιστικά αποτελέσματα για το κάθε ερώτημα που έχει θέσει προς ψηφοφορία. Στο τμήμα το οποίο αφορά την εφαρμογή android ο χρήστης εισάγωντας username και password θα εχει τη δυνατότητα να ψηφίσει ένα από τα ερωτήματα που έχουν υποβληθεί προς ψηφοφορία.

Πίνακας Περιεχομένων

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

ΠΙΝΑΚΑΣ ΠΙΝΑΚΩΝ

1. Εισαγωγή

1.1 Περίληψη

Σκοπός της πτυχιακής εργασίας ήταν η μελέτη ,σχεδίαση και ανάπτυξη ενός συστήματος μέσο του οποίου θα μπορεί να πραγματοποιηθεί μια διαδικασία ψηφοφορίας χρησιμοποιώντας έξυπνες κινητές συσκευές (smartphones) .

Η εφαρμογή αποτελείται από δυο τμήματα το ένα τμήμα είναι το περιβάλλον διαχείρισης (back-end) και το άλλο η εφαρμογή χρήστη(front-end) .Η εφαρμογή χρήστη είναι υλοποιημένη σε λειτουργικό σύστημα Android και το περιβάλλον διαχείρισης είναι φτιαγμένο χρησιμοποιώντας τεχνολογίες διαδικτύου . Για την λειτουργία του συστήματος χρειάζεται ένας server και μια βάση δεδομένων όπου θα αποθηκεύονται τα δεδομένα .

1.2 Κίνητρο για την Διεξαγωγή της Εργασίας

Η ανάγκη ανάπτυξης συστημάτων για καταμέτρηση των ψήφων γρηγορότερα και ποιο εύκολα , η μείωση του κόστους της εκλογικής διαδικασίας αλλά και η δυσκολία μετακίνησης ατόμων με ειδικές ανάγκες καθως και η προσωπική μου επιθυμία για ενασχόληση με τις συγκεκριμένες τεχνολογίες με οδήγησαν στην μελέτη και ανάπτυξη ενός συστήματος ηλεκτρονικής ψηφοφορίας το οποίο χρησιμοποιεί φορητές συσκευές με λειτουργικό σύστημα Android.

1.3 Σκοπός και Στόχοι Εργασίας

Σκοπός και στόχος της εργασίας αυτής είναι η μελέτη ανάλυση και κατασκέυη ενός συστήματος το οποίο θα είναι σε θέση να πραγματοποιήσει μια διαδικασία ψηφοφορίας κάνοντας χρήση φορητών συσκευών με λειτουργικό σύστημα Android .

1.4 Δομη Εργασιας

Η πτυχιακή εργασία αποτελείται από 5 κεφάλαια.Το πρώτο κεφάλαιο έχει να κάνει με την εισαγωγή όπου γίνεται μια περίληψη ,και αναλύονται το κίνητρο και οι στόχοι .Στο δεύτερο κεφάλαιο παρουσιάζονται κάποιες θεωρίες οι οποίες έχουν να κάνουν με τη βασική δομή της εργασίας . Στο τρίτο κεφάλαιο αναλύονται όλα τα εργαλεία τα οποία χρησιμοποιήθηκαν για να δημιουργηθεί το σύστημα . Στο τέταρτο κεφάλαιο το οποίο είναι και το ποιο σημαντικό παρουσιάζεται η υλοποίηση του συστήματος ξεκινώντας από την ανάλυση της βάσης δεδομένων στη συνέχεια γίνεται ανάλυση της web εφαρμογής και τέλος παρουσιάζεται η ανάπτυξη της εφαρμογής για λειτουργικό σύστημα Android.Στο πέμπτο και τελευταίο κεφάλαιο παρουσιάζονται τα συμπεράσματα και οι μελλοντικές επεκτάσεις της εφαρμογής.

2.Μεθοδολογία Υλοποίησης

2.1 Ηλεκτρονική Ψηφοφορια (e-voting).

Όταν για την διεξαγωγή μιας ψηφοφορίας γίνεται χρήση ηλεκτρονικών μέσων όπως ηλεκτρονικοί υπολογιστές και έξυπνα κινητά τηλέφωνα τα οποία διαθέτουν σύνδεση στο διαδίκτυο τότε η διαδικασία αυτή αποκαλείται ηλεκτρονική ψηφοφορία (e-voting).
Η ανάγκη για καταμέτρηση των ψήφων γρηγορότερα και ποιο εύκολα , η μείωση του κόστους της εκλογικής διαδικασίας αλλά και η δυσκολία μετακίνησης ατόμων με ειδικές ανάγκες προκάλεσε την ανάπτυξη συστημάτων ηλεκτρονικής ψηφοφορίας.Τα συστήματα ηλεκτρονικής ψηφοφορίας χωρίζονται σε δυο κατηγορίες:

- Ηλεκτρονική ψηφοφορία σε εκλογικά κέντρα
- Ηλεκτρονική ψηφοφορία με απομακρυσμένη πρόσβαση

Ηλεκτρονική ψηφοφορία σε εκλογικά κέντρα (poll site e-voting):Ο ψηφοφόρος θα πρέπει να μεταβεί σε ειδικά διαμορφωμένα σημεία για να μπορέσει να ασκήσει το εκλογικό του δικαίωμα.Στα σημεία αυτά υπάρχουν ειδικές ηλεκτρονικές συσκευές συνδεδεμένες με το διαδίκτυο .
Ηλεκτρονική Ψηφοφορία με απομακρυσμένη πρόσβαση (Remote e-voting):Η ηλεκτρονική ψηφοφορία από απόσταση μπορεί να γίνει με την χρήση ηλεκτρονικών μέσων όπως: Ηλεκτρονικοί Υπολογιστές, Tablets,Smartphones τα οποία διαθέτουν σύνδεση στο διαδίκτυο .Στην περίπτωση αυτή δεν χρειάζεται ο ψηφοφόρος να παρευρεθεί σε ένα συγκεκριμένο σημείο για να ασκήσει το εκλογικό του δικαίωμα αλλά μπορεί να είναι οπουδήποτε έχοντας απλά στην κατοχή του ένα ηλεκτρονικό μέσο απο αυτά που αναφέρθηκαν παραπάνω συνδεδεμένο στο διαδίκτυο.

Πλεονεκτήματα :

- Ευκολότερη πρόσβαση
- Μειωμένο κόστος
- Πρόληψη Λαθών
- Πιθανή αύξηση συμμέτοχης

Μειονεκτήματα:

- Μη άρτια τεχνολογία
- Απώλεια εμπιστοσύνης Ψηφοφόρων

2.2 Διαδικτυο -Ασφαλεια

Εικονα 1 : Διαδικτυο

Το διαδίκτυο είναι το μεγαλύτερο δίκτυο που υπάρχει το οποιο ενώνει εκατομμύρια συσκευές σε όλο τον κόσμο. Στο παρελθόν πλειοψηφία αυτών των συσκευών αποτελούσαν οι κλασικοί ηλεκτρονικοί υπολογιστές και οι εξυπηρετητές (Servers) .Σήμερα στο διαδίκτυο συνδέονται ολοένα και περισσότερα είδη όπως για παράδειγμα :

- έξυπνα κινητά τηλεφωνά (smartphones),
- είδη οικιακών συσκευών(πλυντήρια ,κουζίνες)
- έξυπνες τηλεοράσεις (smart-TV),
- φορητοί υπολογιστές.

Οι συσκευές που είναι συνδεδεμένες σήμερα στον παγκόσμιο δίκτυο αγγίζουν τα 22,9 δισεκατομμύρια και ο αριθμός συνεχίζει να αυξάνεται εκθετικά.

Ασφάλεια Απένταντι στους κινδύνους του διαδικτύου

Ένας σημαντικός παράγοντας ο οποίος πρέπει όπωσδηποτε να αναφερθεί είναι ο τομέας της ασφάλειας στο διαδίκτυο. Τα επίπεδα της Ασφάλειας είναι :

- Ασφαλεια των Web Sever (εξυπηρετητών)
- Ασφάλεια στο δίαυλο επικοινωνίας Web-Server(εξυπηρετητή) και Client(Πελάτη)

- Ασφάλεια προσωπικού υπολογιστή χρήστη.

Ασφάλεια Ηλεκτρονικής Ψηφοφορίας

Για είναι εφικτή η προστασία της ιδιωτικότητας των πληροφοριών που έχουν να κάνουν με την ψηφοφορία μέσο διαδικτύου αναπτύχθηκαν οι εξής τεχνικές :
Ομορφικες Συναρτησεις Κρυπτογραφησης . Κάθε ψηφοφόρος συνδυάζει μαζί με την ψήφο του ένα κρυπτογραφημένο πιστοποιητικό από τις εκλογικές αρχές . Κάθε πιστοποιητικό αποτελείται από τυχαίους αριθμούς οι οποίοι δίνονται από την επιτροπή των εκλογών .
Κανάλια ανώνυμης επικοινωνίας . Δίνει τη δυνατότητα σε αυτόν που στέλνει ένα μήνυμα να μπορέσει να αποκρύψει την ταυτότητα του από αυτόν που παραλαμβάνει το μήνυμα . Μια υλοποίηση ενός ανωνύμου καναλιού είναι τα Mix-networks .Στα Mix-nets μια ομάδα από εξυπηρετητές (mix-servers) αποκρυπτογραφούν και προωθούν τα ψηφοδέλτια σε ένα τελικό server έτσι ώστε να μην είναι δυνατόν να πραγματοποιηθεί συσχετισμός ενός ψηφοφόρου με μια ψήφο.

2.3 ANDROID

Το Android είναι ένα λειτουργικό σύστημα που έχει αναπτύξει η Google και είναι βασισμένο στον πυρήνα του Linux. Το λειτουργικό έχει σχεδιαστεί κυρίως για συσκευές με οθόνη αφής όπως τα smartphone, τα tablet, υπολογιστές με ειδικά διαμορφωμένο περιβάλλον διεπαφής για τηλεοράσεις (Android TV), ενσωματωμένα συστήματα σε αυτοκίνητα (Android Auto), και ακόμα και σε ρολόγια χειρός (Android Wear).

Εικόνα 2 :Το λογότυπο του Android

Τα χαρακτηριστικά του Android

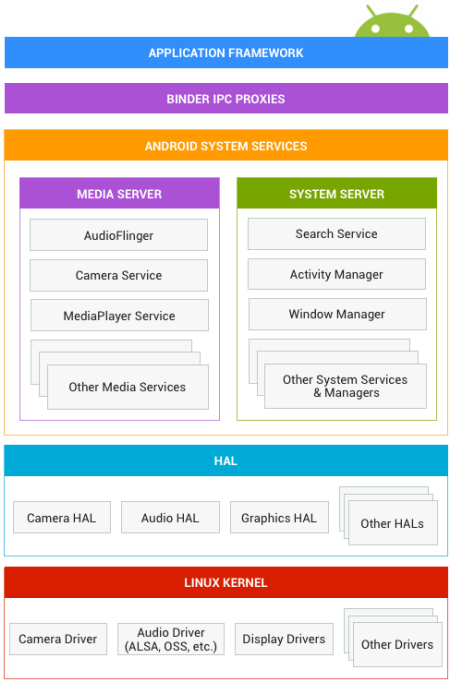
Το Android όπως και οι αντίπαλοι του έδωσε την δυνατότητα στα κινητά τηλεφώνα αλλά και σε άλλες κινητές συσκευές (ρολόγια, φωτογραφικές) μετέπειτα να εξελιχθούν από πολύ απλές συσκευές που πραγματοποιούσαν τηλεφωνικές κλήσεις ή έστελναν απλά SMS για παράδειγμα σε υπερσύγχρονα υπολογιστικά συστήματα που δεν έχουν τίποτα να ζηλέψουν από έναν τυπικό υπολογιστή αλλά και πολύ περισσότερα που δύσκολα θα τα βρεις οπουδήποτε αλλού. Ας δούμε περιληπτικά μερικά από τα πιο βασικά χαρακτηριστικά.

- Η πλατφόρμα είναι προσαρμόσιμη σε πολλές αναλύσεις από απλή VGA μέχρι και 4K στις πιο high-end συσκευές. Παρέχει δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες και τρισδιάστατα γραφικά βασισμένα στην βιβλιοθήκη γραφικών OpenGL ES 3.0+ . Ενώ παρέχει επίσης και υποστήριξη αφής (multitouching).
- Υποστηρίζει πληθώρα τεχνολογιών συνδεσιμότητας όπως για παράδειγμα GSM/EDGE, 3G, 4G,4G+, CDMA, EV-DO, UMTS, Bluetooth, NFC, και Wi-Fi.
- Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογία WebKit.
- Παρέχει υποστήριξη για πλήθος τύπων αρχείων ήχου, εικόνας και βίντεο όπως H.263, H.264(3GP, ή MP4 container), MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OGG Vorbis, WAV, JPEG, PNG, GIF, BMP
- Διαθέτει υποστήριξη για υψηλής ανάλυσης φωτογραφικές μηχανές, για οθόνες αφής, για GPS αλλά και πλήθος αισθητήρων όπως αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.
- Παρέχει στους προγραμματιστές ένα πλήρες περιβάλλον ανάπτυξης λογισμικού που διαθέτει προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων και εργαλεία ανάλυσης της απόδοσης του λογισμικού.
- Τέλος έχει το δικό του App Store υπό την ονομασία Google Play που προσφέρει πλήθος εφαρμογών τόσο δωρεάν όσο και επί πληρωμής.

Αρχιτεκτονική του Android

Σε αυτή την ενότητα θα αναλύσουμε την αρχιτεκτονική του Android. Όπως φαίνεται στην παρακάτω εικόνα το Android αποτελείται από τα εξής 5 επίπεδα:

- Linux Kernel
- Hardware Abstraction Layer (HAL)
- Android System Services
    - Media Server
    - System Server
- Binder IPC Proxies
- Application Framework

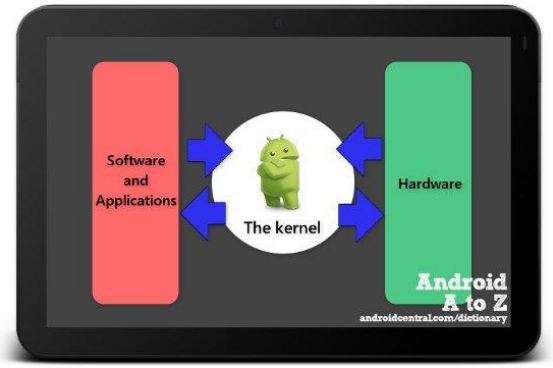


Εικόνα 3 :Αρχιτεκτονική του Android

Ξεκινώντας από το πιο χαμηλό επίπεδο στις επόμενες υποενότητες θα κάνουμε μια σύντομη περιγραφή του κάθε επιπέδου.

Linux Kernel

Το Android είναι κυρίως βασισμένο στον πυρήνα του Linux. Πιο συγκεκριμένα χρησιμοποιεί μια έκδοση του Linux Kernel με κάποιες επιπλέον ιδιαίτερες προσθήκες όπως είναι τα λεγόμενα wakelocks, ένα σύστημα διαχείρισης μνήμης που κυρίως ειδικεύεται στην διατήρηση της μνήμης, ο driver Binder IPC, και διάφορα άλλα χαρακτηριστικά σημαντικά για φορητές πλατφόρμες. Αυτές οι προσθήκες αφορούν κυρίως την λειτουργικότητα του συστήματος και δεν επηρεάζουν την ανάπτυξη οδηγών.
Ακριβώς επειδή το Android είναι ανοιχτού προτύπου, οι προγραμματιστές μπορούν να τροποποιήσουν τον πυρήνα έτσι ώστε να καλύπτει τις ανάγκες τους. Ωστόσο ενώ θα μπορούσαν να φτιάξουν τον δικό τους πυρήνα, αυτό δεν χρειάζεται καθώς η Linux τους παρέχει ήδη έναν προκατασκευασμένο, σταθερό πυρήνα λειτουργικού συστήματος.
Στην σελίδα του Android είναι διαθέσιμες όλες οι μέχρι τώρα εκδόσεις του Android Kernel και έτσι μπορεί κανείς να χρησιμοποιήσει οποιαδήποτε από αυτές αρκεί βέβαια να χρησιμοποιεί τα απαραίτητα χαρακτηριστικά (για παράδειγμα ο driven Binder). Παρ’ όλα αυτά η ομάδα του Android συστήνει να χρησιμοποιούμε την πιο τελευταία έκδοση του Android Kernel.



Εικόνα 4: Android Kernel

Επίπεδο αφαίρεσης υλικού (HAL)

Το επίπεδο αφαίρεσης υλικού είναι στην ουσία ένα πρότυπο διασύνδεσης για τους προμηθευτές συσκευών, που επιτρέπει στο Android να γνωρίζει τους πιο βασικούς drivers (π.χ. driver ήχου, φωτογραφικής μηχανής, Bluetooth κ.α.). Το HAL σου επιτρέπει να χρησιμοποιήσεις οποιονδήποτε από αυτούς τους οδηγούς χωρίς να επηρεάζει ή να τροποποιεί το υψηλότερο επίπεδο συστήματος. Όλοι αυτοί οι οδηγοί χωρισμένοι σε πακέτα της μορφής .so και φορτώνονται από το σύστημα την κατάλληλη στιγμή.

Υπηρεσίες συστήματος

Οι υπηρεσίες συστήματος Android είναι μέρη του συστήματος που αφορούν συγκεκριμένες λειτουργίες του κορμού του λειτουργικού όπως για παράδειγμα ο διαχειριστής παραθύρων (Windows Manager) ή ο διαχειριστής ειδοποιήσεων (Notification Manager). Το Android περιλαμβάνει δύο κατηγορίες υπηρεσιών, της υπηρεσίες συστήματος και τις υπηρεσίες πολυμέσων.

Binder IPC Proxies

Ο μηχανισμός Binder Inter-Process Communication δίνει την δυνατότητα στο application framework να επικοινωνεί απευθείας και να καλεί κώδικα από το επίπεδο υπηρεσίες συστήματος. Έτσι δίνεται η δυνατότητα σε υψηλού επιπέδου API (Application Programming Interface) να αλληλεπιδρά με τις υπηρεσίες συστήματος. Ο προγραμματιστής δεν έχει την δυνατότητα να δει αυτή την επικοινωνία από το Application Framework καθώς είναι κρυμμένη και έτσι φαίνεται απλά ότι όλα δουλεύουν σωστά.

Application Framework

Το Application Framework ή πλαίσιο εφαρμογής είναι αυτό που χρησιμοποιούν πιο συχνά οι προγραμματιστές για τις εφαρμογές τους καθώς τους παρέχει όλα τα διαθέσιμα API. Κατά συνέπεια ένας προγραμματιστής θα πρέπει πάντα να είναι ενήμερος για τα API (προσθήκες ή αλλαγές), διότι τα περισσότερα δείχνουν απευθείας στο επίπεδο αφαίρεσης υλικού και παρέχουν χρήσιμες πληροφορίες για την εισαγωγή οδηγών.

Εκδόσεις του λειτουργικού

Android 1.0 (API LEVEL 1)

Η πρώτη έκδοση του λειτουργικού κυκλοφόρησε στις 23 Σεπτεμβρίου του 2008

Android 1.1 (API Level 2)

Η δεύτερη έκδοση του λειτουργικού κυκλοφορήσε στις 9 Φεβρουαρίου του 2009

Android 1.5 – Cupcake (API Level 3)

Στις 27 Απριλίου του 2009 έρχεται η τρίτη αναβάθμιση του λειτουργικού την 1.5 βασισμένη στην έκδοση 2.6.27 του Linux Kernel, ενώ επισήμως τότε αρχίζουν και οι γλυκές ονομασίες του λειτουργικού αφού αυτή ονομάστηκε Cupcake.

Android 1.6 – Donut (API Level 4)

Η επόμενη αναβάθμιση κυκλοφορεί στις 15 Σεπτέμβριο του 2009 με επίσημη ονομασία Donut, βασισμένη στην έκδοση 2.6.29 του Linux Kernel,

Android 2.0 – 2.1 Éclair (API Level 5, 6, 7)

Τον Οκτώβριο του 2009, κυκλοφορεί το SDK της επόμενης έκδοσης. Η έκδοση αυτή είναι βασισμένη στο Linux Kernel 2.6.29 και έχει την κωδική ονομασία Eclair

Android 2.2 -2.2.3 Froyo (API Level 8)

Τον Μάιο του 2010 κυκλοφορεί το SDK της επόμενης έκδοσης με επίσημη ονομασία Froyo (συντομογραφία της φράσης Frozen Yogurt), η οποία βασίστηκε στο Linux Kernel 2.6.32

Android 2.3 – 2.3.7 Gingerbread (API Level 9 - 10)

Στις 6 Δεκεμβρίου του 2010 κυκλοφορεί το SDK για την έκδοση 2.3 του Android, βασισμένη στο Linux Kernel 2.6.35

Android 3.0 – 3.2.6 Honeycomb (API Level 11-13)

Η επόμενη μεγάλη αναβάθμιση για το λειτουργικό έρχεται τον Φεβρουάριο του 2011 που όμως κυκλοφόρησε μόνο για tablet. Ήταν βασισμένη στον Linux Kernel 2.6.36 και η πρώτη συσκευή που το φόρεσε ήταν το Motorola Xoom που κυκλοφόρησε τον ίδιο μήνα.

Android 4.0 – 4.0.4 Ice Cream Sandwich (API Level 14-15)

Στις 19 Οκτωβρίου του 2011 κυκλοφορεί το SDK για την επόμενη σημαντική αναβάθμιση του λειτουργικού της Google η υπ' αριθμόν 4.0, βασισμένη στο Linux Kernel 3.0.1. Όπως ανακοίνωσε ο Gabe Gohen θα ήταν συμβατή με όλες τις συσκευές που κυκλοφορούσαν εκείνη την περίοδο και έφεραν την έκδοση 2.3. Ο πηγαίος κώδικας για αυτή την έκδοση έγινε διαθέσιμος τον Νοέμβριο του 2011. Αξίζει να σημειωθεί ότι ήταν και η τελευταία έκδοση που υποστήριζε επίσημα τον flash player της Adobe.

Android 4.1 – 4.3.1 Jelly Bean (API Level 16 - 17)

Στην διάσκεψη Google I/O τον Ιούνιο του 2012 ανακοινώνεται από την εταιρεία η έκδοση 4.1 του λειτουργικού, η επονομαζόμενη και ως Jelly Bean βασισμένη στον Linux Kernel 3.0.31. Το Jelly Bean ήταν μια σταδιακή αναβάθμιση με κύριο σκοπό την βελτίωση της λειτουργικότητας και της απόδοσης του user interface. Η πρώτη συσκευή που έφερε αυτή την έκδοση του λειτουργικού ήταν το tablet Nexus 7 που κυκλοφόρησε στις 13 Ιουλίου του 2012.

Android 4.4 - 4.4.4 KitKat (API Level 19-20)

Στις 3 Σεπτεμβρίου του 2013 η Google ανακοινώνει το Android 4.4. Αν και αρχικά είχε την κωδική ονομασία "Key Lime Pie", τελικά το όνομα άλλαξε επειδή, «πολύ λίγοι γνωρίζουν την γεύση μιας Key Lime Pie», όπως είπαν. Αρκετοί bloggers περίμεναν ότι το Key Lime Pie θα ήταν το Android 5.

Το KitKat έκανε το ντεμπούτο του στην συσκευή Nexus 5 της Google και ήταν φτιαγμένο ώστε να λειτουργεί σε μεγαλύτερο εύρος συσκευών από ότι οι προηγούμενες εκδόσεις, αφού οι ελάχιστες απαιτήσεις στη μνήμη RAM ήταν στα 512 MB

Android 5.0-5.1.1 Lolipop(Api Level 21-22)

Η πέμπτη έκδοση του λειτουργικού αποκαλύφθηκε κατά την διάρκεια της Google I/O τον Ιούνιο του 2014, με κωδική ονομασία Android L. Επίσημα διαθέσιμη έγινε τον Νοέμβριο του ίδιου έτους μέσω over-the-air updates, για επιλεγμένες συσκευές που τρέχουν εκδόσεις του λειτουργικού κατευθείαν από την Google, όπως οι συσκευές της σειράς Nexus ή οι συσκευές Google Play Edition. Ο πηγαίος κώδικας του Lollipop έγινε διαθέσιμος στις 3 Νοεμβρίου του 2014.

Android 6.0 Marshmallow (API Level 23)

Όπως αναμενόταν, η Google παρουσίασε επίσημα στις 28 Μαΐου στο πλαίσιο του Google I/O 2015 το Android M όπως είχε ονομαστεί τότε, ενώ αργότερα ανακοινώθηκε το επίσημο όνομα της έκδοσης που δεν είναι άλλο από το Marshmallow.

3. Τεχνολογιες αναπτυξης της εφαρμογης

3.1 PHP

Εικονα 5 : php

Η **PHP** ειναι γλώσσα προγραμματισμού για τη δημιουργία σελίδων web με δυναμικό περιεχόμενο. Μια σελίδα PHP περνά από επεξεργασία από συμβατό web server (**Apache**),ώστε να παραχθεί σε πραγματικό χρόνο το τελικό περιεχόμενο, που θα σταλεί στο πρόγραμμα περιήγησης σε μορφή κώδικα HTML .
Ο κωδικας της PHP χρησιμοποιει τα συμβολα <?> </?> στην αρχη και στο τελος . Ένα παράδειγμα σύνταξης κώδικα PHP στην περίπτωση που ενσωματώνεται σε ένα HTML κώδικα

```
<!DOCTYPE html>
<html>
<body>

<?php
echo "Hello World!";
?>

</body>
</html>
```

3.2 JQUERY

Εικονα 6 : jquery

Η jquery είναι μια βιβλιοθήκη της γλώσσας προγραμματισμού JavaScript η οποία είναι σχεδιασμένη να δίνει τη δυνατότητα στο προγραμματιστή χωρίς ιδιαίτερο κόπο να μπορεί να υλοποιήσει της παρακάτω λειτουργίες :

- Εκτέλεση Ajax requests με μεγάλη ευελιξία
- Δυναμική αλλαγή ιδιοτήτων του css και του html κώδικα
- Δημιουργία διάφορων εφέ –animation σε έγγραφα html
- Χειρισμός events

3.3 SQL

Εικονα 7 : sql

Η SQL (Structured Query Language) είναι μια δομημένη γλώσσα ερωτημάτων , η οποία έχει σχεδιαστεί για την διαχείριση δεδομένων τα οποια βρίσκονται αποθηκευμένα σε σχεσιακές βάσεις. Η sql παρέχει τη δυνατότητα στο προγραμματιστή να μπορέσει να εκτελέσει μέσο ερωτημάτων στη βάση δεδομένων λειτουργίες όπως ανάκτηση και ενημέρωση δεδομένων, δημιουργία και τροποποίηση σχημάτων και σχεσιακών πινάκων.

3.4 HTML

Εικονα 8 : html5

Η HTML (Hyper Markup Language) είναι μια γλώσσα περιγραφής που χρησιμοποιείται για την παρουσίαση του περιεχομένου μιας ιστοσελίδας.Η HTML χρησιμοποιεί στοιχεία τα οποία αποτελούνται από ετικέτες (tags).Οι ετικέτες περιέχονται ανάμεσα σε ειδικά σύμβολα (<>,</>).Οι ετικέτες συνήθως τοποθετουνται σε ζευγαρια ,δηλαδή μετα το άνοιγμα μιας ετικέτας (<h1>) ακολουθεί το κλείσιμο (</h2>).Αναμεσά στις ετικέτες υπάρχουν διάφορα στοιχεία τα οποία επηρεάζονται ανάλογα με το τι έχει η ετικέτα. Η HTML δινει τη δυνατότητα της εισαγωγής εικόνων και άλλων αντικειμένων σε μια web σελίδα

Παραδειγμα :

```
<!DOCTYPE html>
<html>
<title>HTML Example</title>
<body>

<h1> Heading</h1>
<p> paragraph.</p>

</body>
</html>
```

3.5 CSS (Cascading Style Sheets)

Εικονα 9 : css

Είναι ένα είδος γλώσσας το οποίο ανήκει στη κατηγορία των γλωσσών μορφοποίησης του στυλ εμφάνισης μιας σελίδας (style –sheet). Χρησιμοποιείται για να ρυθμίσει την εμφάνιση για ένα HTML έγγραφο. Η CSS είναι μια γλώσσα κατασκευασμένη ώστε μέσο αυτης να μπορει ο προγραμματιστής να διαμορφώσει χαρακτηριστικά τα οποία αφορούν την εμφάνιση μιας web εφαρμογής όπως στοίχιση, χρώματα κ.α.

3.6 Javascript

Εικονα 10 :javascript

Η javascript είναι μια γλώσσα προγραμματισμού η οποία χρησιμοποιείται για την δημιουργία μιας ιστοσελίδας δυναμικού περιεχομένου. Εκτελείται στη μεριά του πελάτη (Client Side). Όπως και η PHP , η Javascript έχει βασιστεί στη γλώσσα προγραμματισμού C, με την οποία παρουσιάζει πολλές ομοιότητες. Όμως ενώ η PHP είναι μια server side γλώσσα προγραμματισμού, η Javascript είναι client side. Αυτό σημαινει οτι η εκτέλεση και επεξεργασία του κώδικα δεν συμβαίνει στον Εξυπηρετητη(Server) αλλά στον περιηγητή ιστού(browser).Η javascript εξαιτίας της ιδιαιτερότητας της αυτής έχει κάποια πλεονεκτήματα και μειονεκτήματα.

Πλεονεκτήματα:

- βασίζεται στις δυνατότητες του browser κα όχι του server.
- Μπορεί να ενσωματωθεί και σε στατικές ιστοσελίδες HTML

Μειονεκτήματα

- Δεν παρέχει δυνατότητα σύνδεσης με βάση δεδομένων

Πολλες φορες υπαρχει ένα μπερδεμα αναμεσα στην jacascript και τη java .Οι δυο αυτές γλωσσες δεν εχουν καμια σχεση είναι εντελως διαφορετικες και εχουν διαφορετικες εφαρμογες.
Η Javascript δεν θα πρέπει να συγχέεται με τη Java, που είναι διαφορετική γλώσσα προγραμματισμού και με διαφορετικές εφαρμογές.

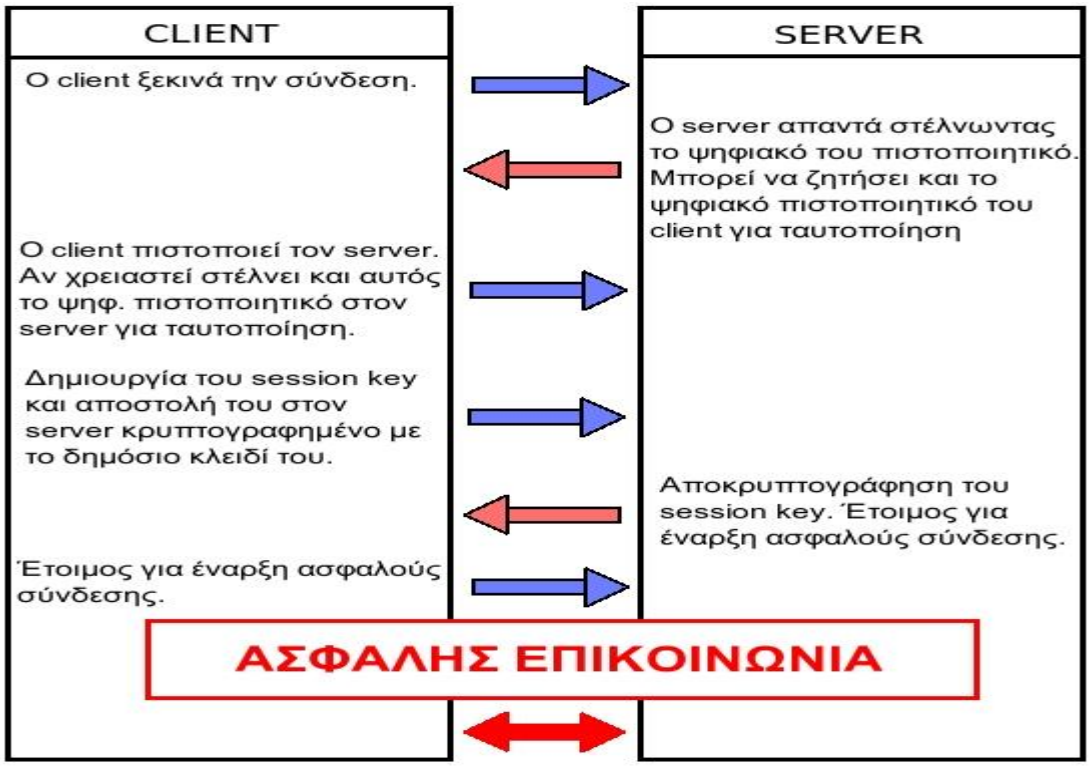
3.7 JAVA

Εικονα 11 : java

Η java είναι μια αντικειμενοστραφής γλώσσα προγραμματισμού που σχεδιάστηκε από την εταιρία Sun Microsystems.Βασικό πλεονέκτημα της java είναι ότι είναι ανεξάρτητη πλατφόρμας και λειτουργικού συστήματος .Όλα δηλαδή τα προγράμματα java εκτελούνται με όμοια συμπεριφορά, είτε το λειτουργικό είναι Windows,Linux, είτε Macintosh.Για να μπορέσει αυτό να είναι εφικτό υπάρχει μια εικονική μηχανή (JVM java Virtual Machine) η οποία μεταγλωττίζει τον κώδικα σε γλώσσα μηχανής .

3.8 SSL

Το SSL (Secure Sockets Layer) είναι ένα πρωτόκολλο το οποίο έχει δημιουργηθεί για να παρέχει ασφάλεια κατά την μεταφορά σημαντικών πληροφοριών μέσο διαδικτύου .Το πρωτόκολλο SSL δημιουργήθηκε από την εταιρία Netscape .Το SSL δημιουργεί μια σύνδεση χρησιμοποιώντας μεθόδους κρυπτογράφησης στις πληροφορίες που ανταλλάσσονται αναμεσά σε ένα Server και ένα Client.Για την μεταφορά των δεδομένων μεταξύ server και client γίνεται χρήση του πρωτοκόλλου TCP/IP .Εξαιτίας του ότι δεν υπάρχει εξάρτηση από την εφαρμογή που χρησιμοποίει ο χρήστης υπάρχει η δυνατότητα παροχής υπηρεσιών ασφάλειας στα πρωτόκολλα HTTP και FTP τα οποια ανήκουν σε ποιο πάνω επίπεδα.



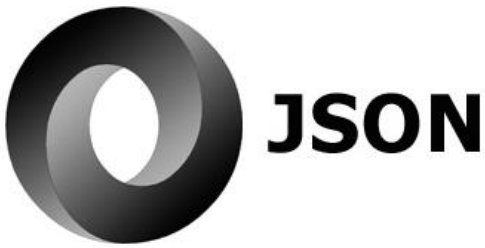
Εικονα 12 :ssl

3.9 Ajax

Εικονα 13 :ajax

AJAX (Asynchronous JavaScript and XML) είναι μια τεχνική η οποία δίνει τη δυνατότητα σε ιστοσελίδες να ενημερώνονται ασύγχρονα χωρίς να χρειάζεται δηλαδή η ανανέωση ολόκληρης της ιστοσελίδας.

3.10 JSON



Εικονα 14 : json

JSON (Javascript Object Notation) είναι ένα ανοιχτό πρότυπο ανταλλαγής πληροφοριών το οποιο ειναι βασισμένο στην γλώσσα προγραμματισμού JavaScript.Το JSON είναι ένα πρότυπο που μπορεί να χρησιμοποιηθεί με αρκετές γλώσσες προγραμματισμού, και χαρακτηρίζεται από απλούστερη δομή σχετικά με το πρότυπο XML.

Το json αποτελείται από δύο μέρη

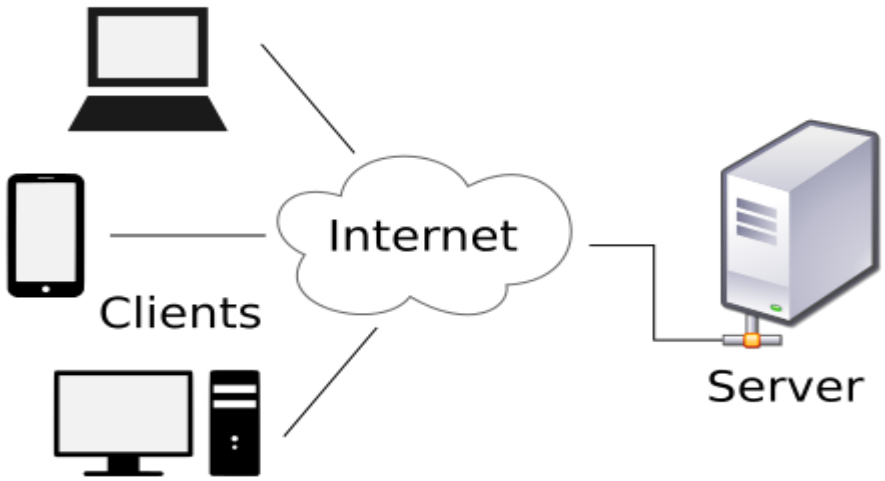
- Μια λίστα από τιμές με κάθε στοιχείο να είναι ταξινομημένο. Αυτό μπορεί να είναι ένας πίνακας ,λίστα, η διάνυσμα
- Μια συλλογή από ζευγάρια ονομάτων/ τιμών (object)

Παράδειγμα σύνταξης json

```
"employees":[
    {"firstName":"Vasilis", "lastName":"Alfa"},
    {"firstName":"Giwrgos", "lastName":"Beta"},
    {"firstName":"Peter","lastName":"Delta"}
]
```

3.11 Μοντελο Πελατη-Εξυπηρετητη (Client –Server)

Είναι μια αρχιτεκτονική αναπτυξης λογισμικού όπου ο Πελάτης(Client/Front-End) ζητάει κάτι από έναν εξυπηρετητή(Server/Back-End).Αυτό το οποίο ζητάει ο πελάτης από τον εξυπηρετητή είναι μια υπηρεσία.Στοχος του Server είναι να παρέχει πληροφορίες η υπολογιστική ισχύ μετά από ένα αίτημα του Client.



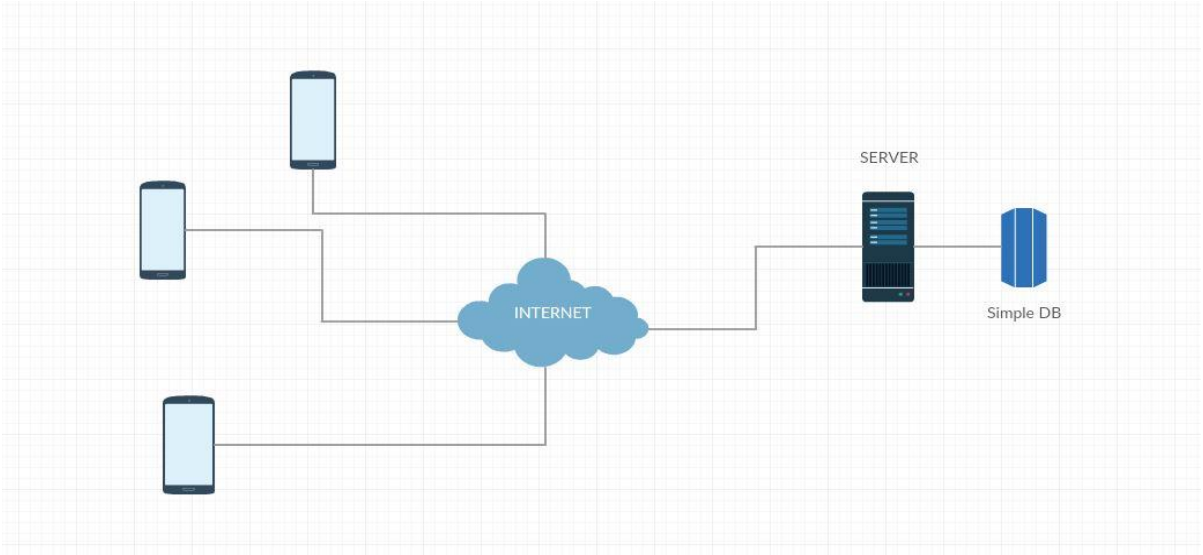
Εικονα 15 :client -server

3.12 Eclipse-Android SDk

Το eclipse είναι ένα περιβάλλον ανάπτυξης λογισμικού (Intergrated development environment IDE) το οποίο χρησιμοποιείται κυρίως για την ανάπτυξη κώδικα σε Java.Το eclipse υποστηρίζει τη δυνατότητα προθήκης επεκτάσεων (plug-ins) με τη χρήση των οποίων είναι δυνατό να γραφούν γλώσσες προγραμματισμού όπως C,C++,JavaScript,PHP,python,R και διάφορες άλλες. Με τις κατάλληλες προθήκες είναι δυνατή και η ανάπτυξη κώδικα για λειτουργικό σύστημα Android (Android-SDK).

4. Υλοποιηση Συστηματος

Σε αυτό το κεφάλαιο θα γίνει αναλυτική περιγραφή της υλοποίησης του συστήματος .Η περιγραφή της υλοποίησης είναι χωρισμένη σε δυο τμήματα .Το ένα τμήμα αφορά την web εφαρμογή η οποία έχει να κάνει με το κομμάτι διαχείρισης του συστήματος.Το άλλο τμήμα αφορά την εφαρμογή χρήστη που είναι κατασκευασμένη για έξυπνες κινητές συσκευές οι οποίες τρέχουν σε λειτουργικό σύστημα android.



Εικονα 16 :application diagram

Server

Για την δημιουργία της βάσης δεδομένων αλλά και της web εφαρμογής χρειάστηκε να χρησιμοποιήσουμε ένα εξυπηρετητή διαδικτύου (Web Server).Ο server ο οποίος χρησιμοποιήσαμε παρέχει δωρεάν φιλοξενία και υποστηρίζει τεχνολογίες php και mysql.

4.1 DATABASE (ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ)

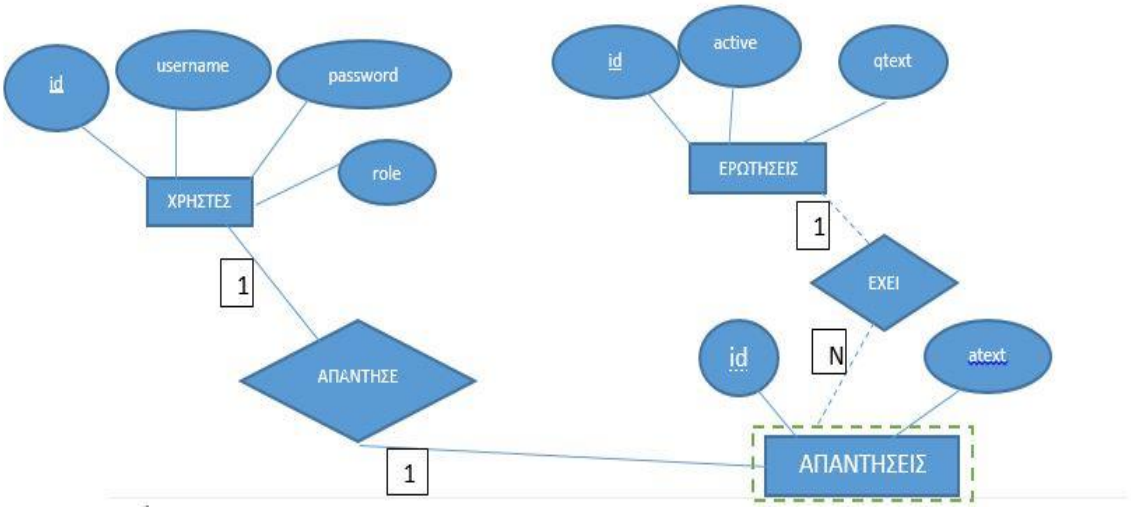
Αρχικά για να μπορέσει το σύστημα να λειτουργήσει σωστά είναι απαραίτητη η δημιουργία μιας βάσης δεδομένων οπού θα αποθηκεύονται διάφορα στοιχεία .

Οι πίνακες που θα χρειαστεί να δημιουργηθούν είναι οι εξής:

- **Χρήστες(Users):**ο πίνακας αυτός περιλαμβάνει τα id ,username,password,role .Το id είναι ένα αναγνωριστικό το οποίο θα χρησιμοποιηθεί και ως πρωτεύον κλειδί για να μπορεί να προσδιοριστεί μοναδικά κάθε εγγραφή ενός χρήστη **.** Όνομα Χρήστη (username):το όνομα χρήστη είναι ένα σύνολο χαρακτήρων (συμβολοσειρά)το οποίο χρησιμεύει στο προσδιορισμό του ονόματος κάθε χρήστη . Κωδικός Χρήστη (password) :το πεδίο αυτό χρησιμεύει στην αποθήκευση ενός μοναδικού κωδικού πρόσβασης στο σύστημα.

Ρόλος(role) : προσδιορίζει το ρόλο του κάθε χρήστη ,είτε είναι απλός χρήστης είτε διαχειριστής.

- **Ερωτήσεις (Questions)**:Σε αυτόν το πίνακα θα καταχωρούνται οι ερωτήσεις οι οποίες έχουν δημιουργηθεί στο σύστημα είτε είναι ενεργές είτε όχι .Όπως και στον παραπάνω πίνακα έτσι και εδώ υπάρχει το id για προσδιορίσει μοναδικά της ερωτήσεις υπάρχει επίσης το question_text όπου αποθηκεύεται ως συμβολοσειρά το κείμενο το οποίο έχει καταχωρηθεί στο σύστημα σαν ερώτηση . Τελευταίο πεδίο είναι το active που δηλώνει το εάν μια ερώτηση είναι ενεργή η όχι ,εάν δηλαδή είναι η ερώτηση η οποία εμφανίζεται ως κύριο ερώτημα προς ψηφοφορία στο σύστημα .
- **Απαντήσεις(Answers):**πίνακας ο οποίος αποθηκεύει ένα μοναδικό id για κάθε εγγραφή μια συμβολοσειρά για το κείμενο που έχει καταχωρηθεί ως απάντηση (answer_text) και ένα id της ερώτησης που αντιστοιχεί με το answer_text (question_id)
- **Απαντήσεις Χρηστών(User_Answered):**συσχετίζει τα μοναδικά αναγνωριστικά id για answer,question,και user .



Εικονα 17 : database diagram

Η βάση μας αποτελείται από 4 πίνακες.

USERS

id	username	password	role

QUESTIONS

id	question_text	active

ANSWERS

id	answer_text	question_id

USERS_ANSWERED

answers.id	questions.id	user.id

ΑΝΑΛΥΣΗ ΠΕΔΙΩΝ ΠΙΝΑΚΩΝ

USERS

id: ΑΚΕΡΑΙΟΣ, AUTO INCREMENT, PRIMARY KEY
username: ΑΛΦΑΡΙΘΜΗΤΙΚΟ (VARCHAR (100)), NOT NULL
password: ΑΛΦΑΡΙΘΜΗΤΙΚΟ (TEXT), NOT NULL
role: ΑΛΦΑΡΙΘΜΗΤΙΚΟ (enum), NOT NULL

QUESTIONS

id: ΑΚΕΡΑΙΟΣ, AUTO INCREMENT, PRIMARY KEY
question_text: ΑΛΦΑΡΙΘΜΗΤΙΚΟ (TEXT), NOT NULL
active: ΑΚΕΡΑΙΟΣ (TINYINT)

ANSWERS

id: ΑΚΕΡΑΙΟΣ, AUTO INCREMENT, PRIMARY KEY
answer_text: ΑΛΦΑΡΙΘΜΗΤΙΚΟ (TEXT), NOT NULL
question_id: ΑΚΕΡΑΙΟΣ , FOREIGN KEY REFERENCING QUESTIONS TABLE

USERS_ANSWERED

id: ΑΚΕΡΑΙΟΣ, AUTO INCREMENT, PRIMARY KEY
answer_id: ΑΚΕΡΑΙΟΣ, FOREIGN KEY REFERENCING ANSWERS TABLE
question_id: ΑΚΕΡΑΙΟΣ , FOREIGN KEY REFERENCING QUESTIONS TABLE
user_id: ΑΚΕΡΑΙΟΣ , FOREIGN KEY REFERENCING USERS TABLE

Παρακάτω φαίνεται ο κώδικας σε SQL για τη δημιουργία της βάσης.

```
CREATE TABLE IF NOT EXISTS `answers` (
 `id` int(11) NOT NULL,
  `answer` text NOT NULL,
  `question_id` int(11) NOT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `questions` (
 `id` int(11) unsigned NOT NULL,
  `question_text` text NOT NULL,
  `active` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (id)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `users` (
 `id` int(10) unsigned NOT NULL ,
  `username` varchar(100) NOT NULL,
  `password` text NOT NULL,
  `role` enum('admin','user') NOT NULL DEFAULT 'user',
 PRIMARY KEY (id)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

CREATE TABLE IF NOT EXISTS `users_answered` (
 `id` int(10) unsigned NOT NULL,
  `user_id` int(10) unsigned DEFAULT NULL,
  `question_id` int(10) unsigned DEFAULT NULL,
  `answer_id` int(11) NOT NULL,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 AUTO_INCREMENT=1 ;

ALTER TABLE `users` ADD UNIQUE KEY `username` (`username`);
ALTER TABLE `users_answered` ADD KEY `user_id` (`user_id`), ADD KEY `question_id`
(`question_id`);
ALTER TABLE `users_answered` ADD KEY `answer_id` (`answer_id`);
ALTER TABLE `users_answered` ADD CONSTRAINT `users_answered_ibfk_1` FOREIGN KEY
(`user_id`) REFERENCES `users` (`id`) ON DELETE SET NULL ON UPDATE NO ACTION,
ADD CONSTRAINT `users_answered_ibfk_2` FOREIGN KEY (`question_id`) REFERENCES
`questions` (`id`) ON DELETE SET NULL ON UPDATE NO ACTION,
ADD CONSTRAINT `users_answered_ibfk_3` FOREIGN KEY (`answer_id`) REFERENCES `answers`
(`id`) ON DELETE SET NULL ON UPDATE NO ACTION;
```

Η διαχείριση της Βάσης Δεδομένων που χρησιμοποιεί το σύστημα, γίνεται από το αρχείο *includes/database.php* . Το σύστημα χρησιμοποιεί τη MySQLi βιβλιοθήκη, για να κάνει τις ερωτήσεις στη βάση όπως και για να συνδεθεί.

Η χρήσης της MySQLi μας δίνει αρκετά πλεονεκτήματα εκ των οποίων μερικά από αυτά είναι τα παρακάτω:

- τη δυνατότητα να χρησιμοποιήσουμε prepared statements,
- να έχουμε καλύτερα μηνύματα αποσφαλμάτωσης,

- όπως επίσης να έχουμε κάποια μορφή αντικειμενοστραφή προγραμματισμού για την εκτέλεση των ερωτημάτων.

Σύνδεση στη Βάση

Για να συνδεθούμε στη βάση δεδομένων χρησιμοποιούμε το παρακάτω κώδικα:

```
$sqli=new mysqli(HOST, USER, PASSWORD, NAME);
if ($sqli->connect_errno){
   return false;
}
$sqli->set_charset('utf8');
```

Μετά από κάθε σύνδεση στη βάση, θα πρέπει να κλείνουμε την επικοινωνία. Αυτό επιτυχάνεται με το παρακάτω κώδικα:

```
$sqli->close();
```

Πραγματοποίηση Ερωτήματος στη Βάση

Κάθε φορά που θέλουμε να κάνουμε κάποιο ερώτημα προς τη βάση δεδομένων, δημιουργούμε πρώτα ένα *statement.* Αυτό γίνεται ορίζοντας πρώτα το ερώτημα που θέλουμε και καλώντας μετά την μέθοδο *prepare* πάνω στο sqli αντικείμενο που έχουμε από την σύνδεση στη βάση.

π.χ

```
$sql = 'update questions set question_text=?, active=? where id=?';
$stmt = $sqli->prepare($sql);
```

Στο παραπάνω παράδειγμα η μεταβλητή $stmt έχει ένα αντικείμενο τύπου Statement. Επίσης όπως βλέπουμε η SQL περιέχει των χαρακτήρα '?', αυτό μας δίνει τη δυνατότητα να ορίσουμε το τύπο δεδομένων που δεχόμαστε για το συγκεκριμένο πεδίο, όπως επίσης και να τρέξουμε το ερώτημα πολλές φορές με διαφορετικά δεδομένα χωρίς να χρειάζεται να αλλάζουμε συνεχώς το ερώτημα, καλώντας τη μέθοδο bind_param στο αντικείμενο Statement.
Μετά τη δημιουργία του ερωτήματος (prepare, bind_param), καλώντας τη μέθοδο *execute* στο αντικείμενο *Statement,* εκτελείτε το ερώτημα που έχουμε ετοιμάσει. Αν θέλουμε να πάρουμε κάποιο αποτέλεσμα (π.χ να βρούμε το id ενός συγκεκριμένου χρήστη), μετά την εκτέλεση του ερωτήματος πρέπει να καλέσουμε τη μέθοδο *fetch* για να μας φέρει τα αποτελέσματα.

π.χ

```
$sql = 'select id,username,role from users';
$stmt = $sqli->prepare($sql);
$stmt->execute();
$stmt->bind_result($userId, $username, $role);
$result = array();
$i=0;
while ($row = $stmt->fetch()){
   $result[$i]['id'] = $userId;
   $result[$i]['username'] = $username;
   $result[$i]['role'] = $role;
   $i++;
}
```

```
$stmt->close();
$sqli->close();
```

Στο παραπάνω παράδειγμα θέλουμε να βρούμε όλους τους χρήστες που έχει το σύστημα μας.

Επιστροφή Αποτελέσματος

Σε όλα τα ερωτήματα που θέλουμε κάποια απάντηση (π.χ να βρούμε το αναγνωριστικό ενός συγκεκριμένου χρήστη), οι συναρτήσεις που χρησιμοποιεί το σύστημα επιστρέφουν ένα πίνακα με τα δεδομένα που θέλουμε αν υπάρχει αποτέλεσμα στο ερώτημα αλλιώς επιστρέφουμε boolean false τιμή.

```
function getQuestions(){
   $sqli = new mysqli(HOST, USER, PASSWORD, NAME);
   if ($sqli->connect_errno){
      return false;
   }
   $sqli->set_charset('utf8');

   $sql = 'select id,question_text,answers_text,active from questions';
   $stmt = $sqli->prepare($sql);
   $stmt->execute();
   $stmt->bind_result($questionId, $question_text, $answer_text, $active);
   $result = array();
   $i=0;
   while ($row = $stmt->fetch()){
      $result[$i]['id'] = $questionId;
      $result[$i]['question_text'] = $question_text;
      if ($answer_text != null){
         $test = json_decode($answer_text);

         if (count($test)>1){
            $test = implode("||", $test);
         }else {
            $test = $test[0];
         }
         $result[$i]['answer_text'] = $test;

      }else {
         $result[$i]['answer_text'] = $answer_text;
      }
      $result[$i]['active'] = $active;
      $i++;
   }
   $stmt->close();
   $sqli->close();
   if (empty($result)){
      return false;
   }
   return $result;
}
```

Η παραπάνω μέθοδος επιστρέφει τις ερωτήσεις που έχουμε καταχωρημένες στη βάση δεδομένων, εκτελώντας ένα ερώτημα πάνω στο πίνακα **questions** της βάσης μας.
Όπως φαίνεται στο παραπάνω κώδικα, δημιουργούμε ένα πίνακα *$result* όπου και κρατάμε τα αποτελέσματα του ερωτήματος που κάναμε στη βάση **"select id,question_text,answers_text,active**

from questions". Πριν η μέθοδος επιστρέψει ελέγχουμε αν ο πίνακας είναι άδειος χρησιμοποιώντας τη μέθοδο *empty* της php. Αν ο πίνακας δεν έχει στοιχεία μέσα τότε επιστρέφουμε false (*return false;)* αλλιώς επιστρέφουμε το πίνακα που έχουμε δημιουργήσει (*return $result)*.

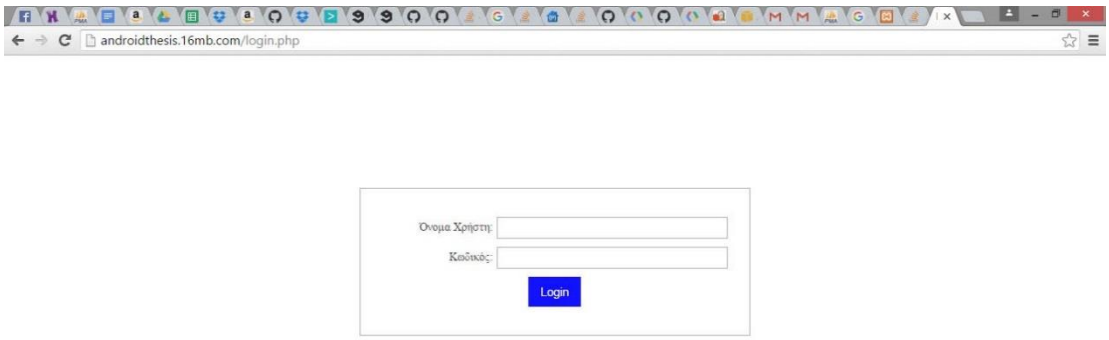
4.2 Web Εφαρμογή

Το σύστημα ειναι βασισμένο σε WEB τεχνολογίες για τη καλύτερη διαχείριση των Ερωτήσεων, Χρηστών, και εμφάνιση στατιστικών αποτελεσμάτων των απαντήσεων. Στο σύστημα πρόσβαση έχουν μόνο οι χρήστες που έχουν ρόλο διαχειριστή (admin). Μέσω του συστήματος ένας διαχειριστής μπορεί να επεξεργαστεί τους Χρήστες, τις Ερωτήσεις με τις Απαντήσεις τους, και να δει τα αποτελέσματα των ψηφοφοριών. Το σύστημα απλοποιεί τις βασικές λειτουργίες μιας βάσης, χωρίς να χρειάζεται ο συνδεδεμένος χρήστης να έχει γνώσεις βάσεων δεδομένων, προσφέροντας μια σειρά από επιλογές όπως επεξεργασία, διαγραφή, εισαγωγή. Τέλος ο συνδεδεμένος χρήστης διαλέγοντας μια ερώτηση μπορεί να δει τις απαντήσεις που έχουν δοθεί στη συγκεκριμένη ερώτηση.

Για την υλοποίηση του συστήματος διαχείρισης χρησιμοποιήθηκαν οι εξής τεχνολογίες: *PHP, Javascript, HTML5, JQuery, AJAX.*

ΕΙΣΟΔΟΣ (LOGIN)

Κάθε φορά που υπάρχει ένα ερώτημα προς το σύστημα μας, το σύστημα ελέγχει αν ο χρήστης που ξεκίνησε το αίτημα είναι συνδεδεμένος και αν έχει το δικαίωμα να ξεκινήσει αυτό το αίτημα. Αν ο χρήστης δεν έχει το δικαίωμα να ξεκινήσει το ερώτημα, ή αν ο χρήστης δεν είναι συνδεδεμένος, το σύστημα των ανακατευθύνει στη σελίδα εισόδου, για να βάλει τα στοιχεία του και να συνδεθεί.



Εικονα 18 : web-app login

Ο χρήστης βάζοντας τα σωστά στοιχεία του στη παραπάνω φόρμα, συνδέεται στο σύστημα. Μόλις γίνει υποβολή της φόρμας, το σύστημα ελέγχει τα στοιχεία αν είναι σωστά. Αν είναι σωστά τότε αποθηκεύουμε στο SESSION το αναγνωριστικό, και το ρόλο του χρήστη, και τον ανακατευθύνουμε στην αρχική σελίδα. Αλλιώς εμφανίζεται στο χρήστη μήνυμα ότι τα στοιχεία που μας έδωσε δεν είναι σωστά. (Αρχείο login.php).

Χρήστες

Η εφαρμογή υποστηρίζει την εμφάνιση,δημιουργία,επεξεργασία,και διαγραφή ενος χρήστη .

Χρήστες: Εμφάνιση

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή χρήστες, το σύστημα του επιστρέφει όλους τους χρήστες που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα.

DB ID	Όνομα Χρήστη	Ρόλος	Δράσεις
1	bill	admin	Επεξεργασία Διαγραφή
2	Manolis	user	Επεξεργασία Διαγραφή
4	test1	admin	Επεξεργασία Διαγραφή
5	test22	user	Επεξεργασία Διαγραφή
6	user1	user	Επεξεργασία Διαγραφή
7	ablaoubla	admin	Επεξεργασία Διαγραφή
8	ablaoubla1	admin	Επεξεργασία Διαγραφή
10	aaaaa1	user	Επεξεργασία Διαγραφή

Εικονα 19 : web-app users

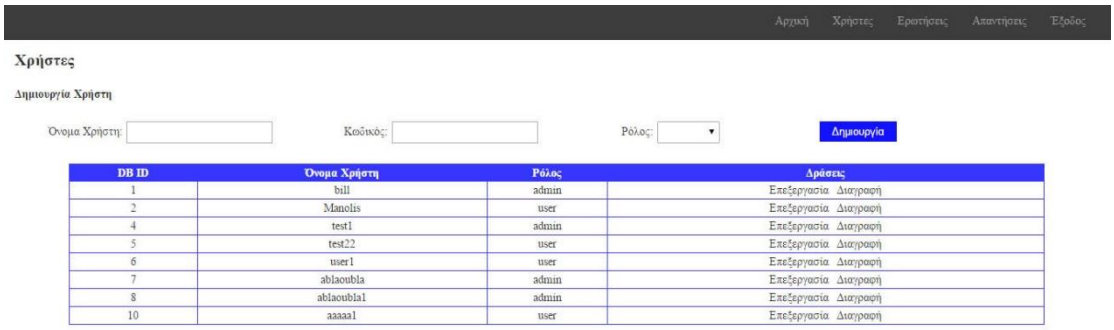
Στο παραπάνω πίνακα ο συνδεδεμένος χρήστης, βλέπει όλους τους χρήστες με το ID που έχουν στη βάση μας, το όνομα χρήστη τους και το ρόλο τους. Στη τελευταία στήλη υπάρχουν οι δράσεις που μπορούν να εφαρμοστούν στο κάθε χρήστη όπως διαγραφή, και επεξεργασία.

Δημιουργία Χρηστών

Για την εισαγωγή ενος χρήστη στο σύστημα θα πρέπει ο διαχειριστής ο οποιος έχει κάνει είσοδο στο σύστημα να εισάγει τα στοιχεία :

- **Όνομα Χρήστη**
- **Κωδικός**
- **Ρόλος(User/Admin)**

και να πατήσει το κουμπι Δημιουργία.Μετά από αυτή τη διαδικασία γίνεται η καταχώρηση των στοιχείων στη βάση δεδομένων .



Εικονα 20 : web-app create user

Πατώντας το κουμπί δημιουργία, το σύστημα μέσω της τεχνολογίας AJAX ελέγχει αν είναι αποδεκτά τα στοιχεία που θέλουμε να εισάγουμε (αν τα πεδία δεν είναι κενά, αν το όνομα χρήστη είναι μοναδικό και αν το όνομα χρήστη δεν ξεπερνάει τους 100 χαρακτήρες), και εφόσον τα στοιχεία είναι σωστά γίνεται η εισαγωγή του χρήστη στη βάση μας.

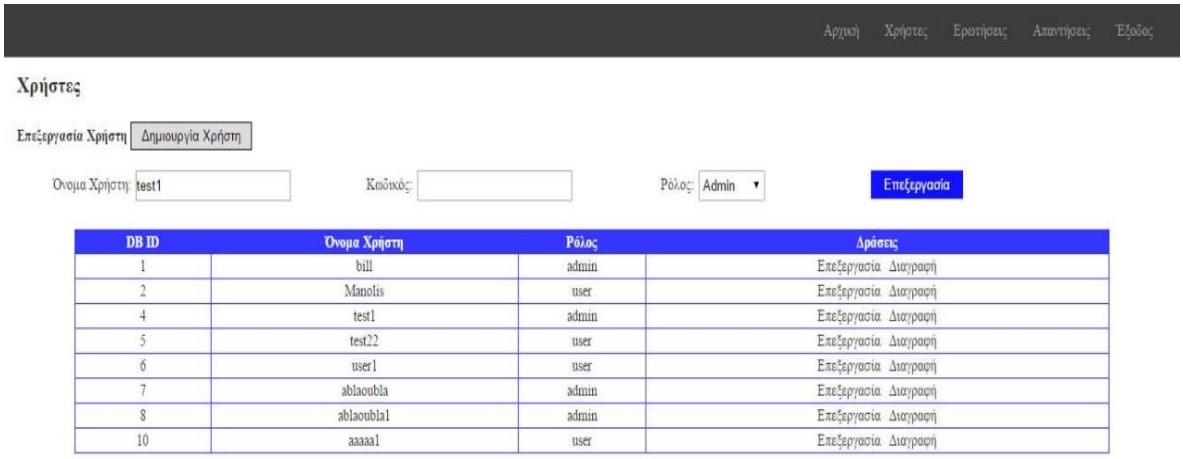
Επεξεργασία Χρηστών

Το σύστημα δίνει τη δυνατότητα για επεξεργασία των στοιχείων ενός χρήστη όπως :

- αλλαγή username η password .
- αλλαγή ρόλου (admin /User) .

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή χρήστες, το σύστημα του εμφανίζει όλους τους χρήστες σαν ένα HTML πίνακα. Πατώντας τον σύνδεσμο επεξεργασία, η φόρμα αλλάζει και από δημιουργία χρήστη, μπορούμε πλέον να επεξεργαστούμε το χρήστη που διαλέξαμε. Η φόρμα έχει συμπληρωμένα τα πεδία του ονόματος χρήστη, και του ρόλου,

όπως επίσης συμπληρώνει το κρυφό πεδίο της φόρμας που αντιστοιχεί στο αναγνωριστικό που έχει ο συγκεκριμένος χρήστης στη βάση.



Εικονα 21: web-app user processing

Διαγραφη Χρηστών

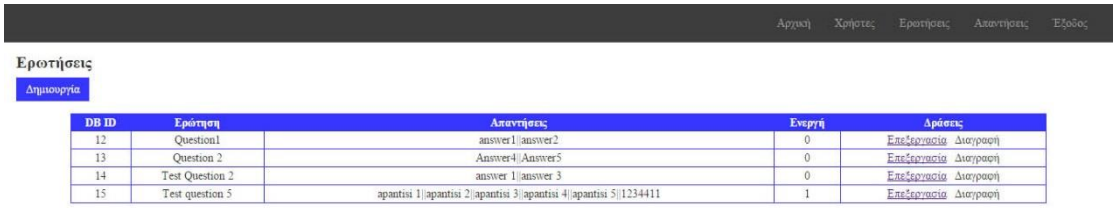
Ο διαχειριστής του συστήματος έχει τη δυνατότητα να διαγράψει ένα χρήστη .Η διαγραφή σημαίνει πλήρη αφαίρεση των στοιχείων από τη βάση δεδομένων.
Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή χρήστες, το σύστημα του εμφανίζει όλους τους χρήστες σαν ένα HTML πίνακα. Πατώντας τον σύνδεσμο Διαγραφή, ο χρήστης διαγράφεται από τη βάση μας μέσω της τεχνολογίας AJAX. Η διαγραφή γίνεται βάση το αναγνωριστικού που έχει ο χρήστης στη βάση. Μετά το πέρας της διαγραφής και αν η διαγραφή γίνει επιτυχώς, ο πίνακας που δείχνει όλους τους χρήστες ανανεώνεται αφαιρώντας τον διαγραμμένο χρήστη.

Ερωτήσεις

Στην ενότητα ερωτήσεις δίνεται η δυνατότητα στον διαχειριστή για τη δημιουργία ,προβολή και επεξεργασία ενός ερωτήματος.

Εμφάνιση Ερωτήσεων

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Ερωτήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα.

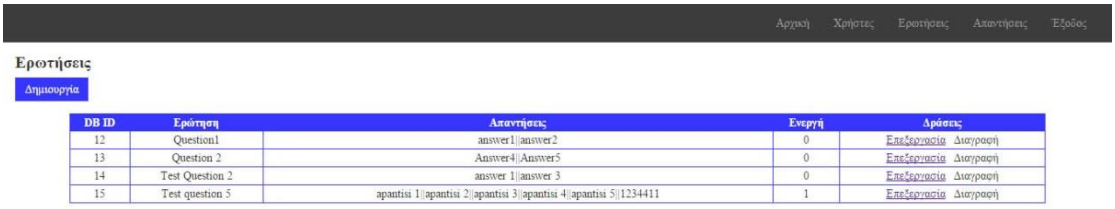


Εικονα 22 : web-app question appearance

Στο παραπάνω πίνακα ο συνδεδεμένος χρήστης, βλέπει όλες τις ερωτήσεις με το ID που έχουν στη βάση μας, τις απαντήσεις τους και το αν είναι ενεργή μία ερώτηση. Στη τελευταία στήλη υπάρχουν οι δράσεις που μπορούν να εφαρμοστούν στο κάθε χρήστη όπως διαγραφή, και επεξεργασία.

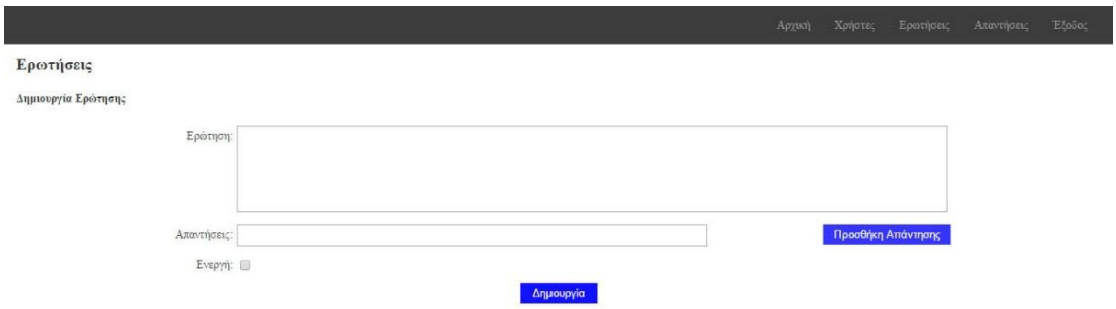
Δημιουργία Ερωτήσεων

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Ερωτήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα. Πάνω από το πίνακα υπάρχει ένα κουμπί, το οποίο πατώντας το μεταφέρει το χρήστη στη σελίδα της δημιουργίας ερώτησης.



Εικονα 23 : web-app question creation 1

Στη φόρμα που εμφανίζεται ο χρήστης μπορεί να συμπληρώσει τα πεδία της ερώτησης και πατώντας το κουμπί δημιουργία, το σύστημα ελέγχει αν τα πεδία είναι αποδεκτά, και αν είναι γίνεται η εισαγωγή της ερώτησης στη βάση.



Εικονα 24 : web-app question creation 2

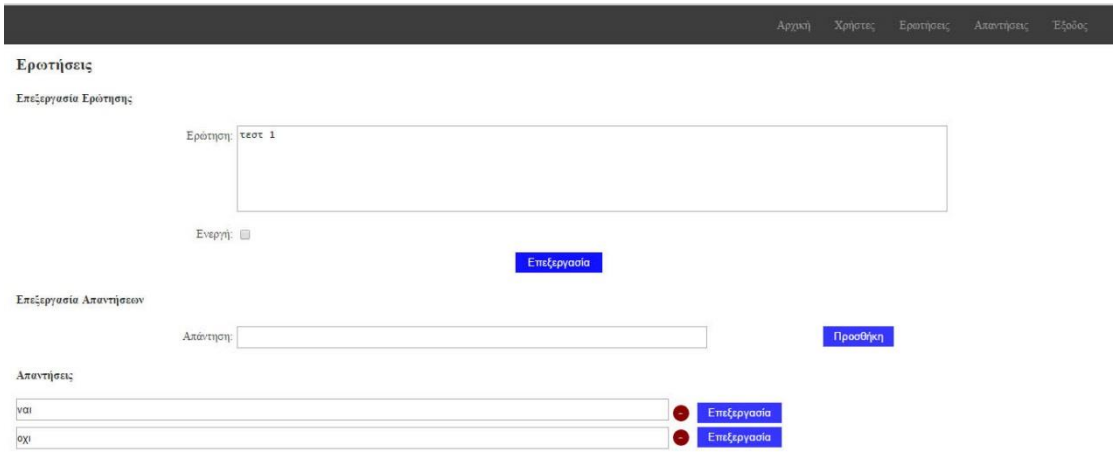
Αν ο χρήστης δεν συμπληρώσει κάποια απάντηση, τότε το σύστημα δημιουργεί αυτόματα δύο απαντήσεις (ΝΑΙ/ΟΧΙ) για τη συγκεκριμένη ερώτηση. Ο χρήστης κάθε φορά που πατάει το κουμπί Προσθήκη Απάντησης δημιουργεί άλλο ένα πεδίο για να συμπληρώσει μια απάντηση που θα έχει η ερώτηση.
Αν ο χρήστης επιλέξει να κάνει ενεργή την ερώτηση που θέλει να δημιουργήσει τότε το σύστημα απενεργοποιεί τις υπόλοιπες ενεργές ερωτήσεις.
Πατώντας το κουμπί δημιουργία το σύστημα μέσω τεχνολογίας AJAX, ελέγχει τα πεδία και αν είναι σωστά γίνεται η εισαγωγή στη βάση, και εμφανίζει το αντίστοιχο μήνυμα στο χρήστη.

Επεξεργασία Ερωτήσεων

Ο διαχειριστής μπορεί να επεξεργαστεί ένα ερώτημα .
Η δυνατότητες επεξεργασίας είναι :

- τροποποίηση της ερώτησης
- ενεργή η όχι
- προσθήκη η διαγραφή απαντήσεων
- αλλαγή η τροποποίηση απαντήσεων

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Ερωτήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα. Πατώντας στο σύνδεσμο Επεξεργασία, ο χρήστης μεταφέρεται στη σελίδα επεξεργασίας της ερώτησης



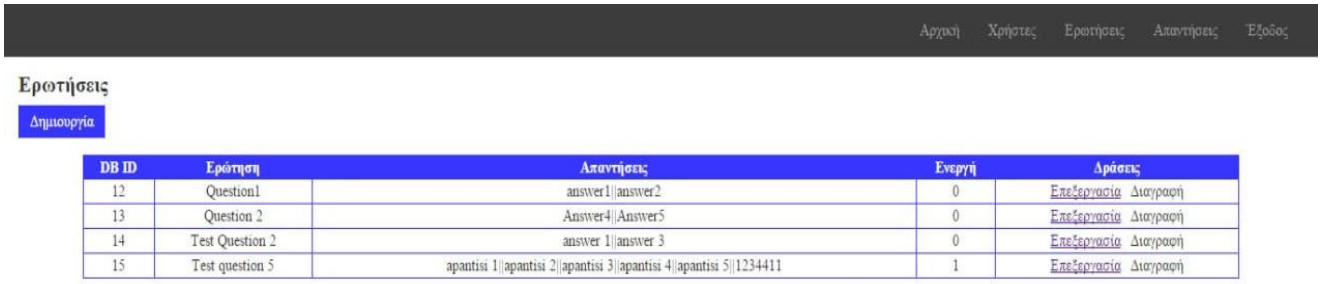
Εικονα 25 : web-app question processing

Εδώ ο χρήστης μπορεί να αλλάξει την ερώτηση, να την ενεργοποιήσει, και να προσθέσει ή να αφαιρέσει απαντήσεις από την ερώτηση. Ουσιαστικά η σελίδα χωρίζεται σε 2 τομείς. Ο ένας έχει να κάνει με την ερώτηση και ο άλλος με τις απαντήσεις.

Στη φόρμα των απαντήσεων, ο χρήστης μπορεί αλλάζοντας το κείμενο σε μία απάντηση και πατώντας το κουμπί Επεξεργασία που βρίσκεται δίπλα από κάθε απάντηση να αλλάξει το κείμενο της. Πατώντας το κουμπί "-", τότε η απάντηση αφαιρείτε από την ερώτηση.

Διαγραφή Ερωτήσεων

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Ερωτήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα. Πατώντας το σύνδεσμο Διαγραφή που υπάρχει σε κάθε γραμμή του πίνακα, έχει ως αποτέλεσμα τη διαγραφή της συγκεκριμένης ερώτησης από το σύστημα. Το σύστημα αυτόματα ανανεώνει το πίνακα, έτσι ώστε να μην φαίνεται η διαγραμμένη ερώτηση.



Εικονα 26 : web-app question delete

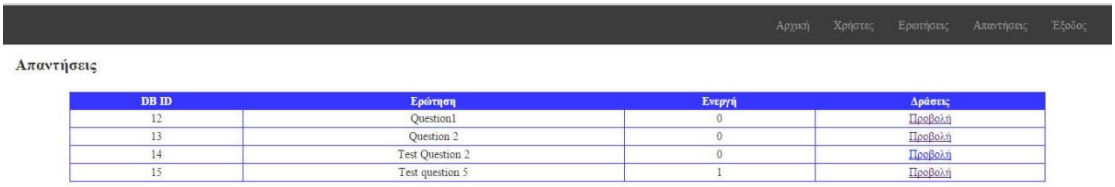
Απαντήσεις

Απαντήσεις Εμφάνιση

Το σύστημα δίνει την δυνατότητα για :

- συσχέτιση ερωτήσεων ,απαντήσεων
- εμφάνιση αποτελεσμάτων

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Απαντήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα.

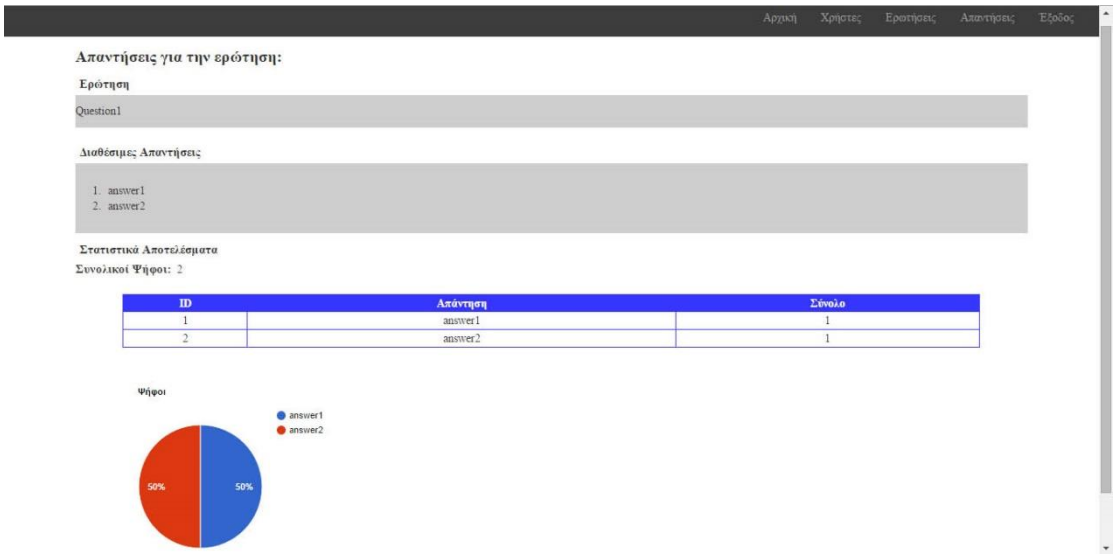


Εικονα 27 : web-app answers

Ο χρήστης πατώντας το σύνδεσμο Προβολή μπορεί να δει τις απαντήσεις που έχουν δώσει οι χρήστες στη συγκεκριμένη ερώτηση.

Απαντήσεις Αποτελέσματα

Ο χρήστης αφού έχει εισέλθει στο σύστημα, επιλέγοντας από το μενού μας την επιλογή Απαντήσεις, το σύστημα του επιστρέφει όλες τις ερωτήσεις που υπάρχουν, και τους εμφανίζει σε ένα HTML πίνακα, όπως φαίνεται στη παρακάτω εικόνα. Αφού ο χρήστης επιλέξει έναν σύνδεσμο Προβολή, τότε ο χρήστης μεταφέρεται στη σελίδα των αποτελεσμάτων τις επιλεγμένης ερώτησης.



Εικονα 28 : web-app answer pie chart

Όπως φαίνεται στη παραπάνω εικόνα, το σύστημα του εμφανίζει την ερώτηση, τις διαθέσιμες απαντήσεις και τα σύνολα των ψήφων που υπάρχουν. Στο τέλος το σύστημα εμφανίζει μέσω ενός "διαγράμματος πίττας" (pie chart), το σύνολο ψήφων που έχει κάθε απάντηση.

Έλεγχοι εγκυρότητας δεδομένων

Όταν υπάρχει κάποια εισαγωγή, ενημέρωση ή διαγραφή το σύστημα ελέγχει μέσω php κώδικα αν η τιμή που θέλουμε να εισάγουμε, ενημερώσουμε, διαγράψουμε είναι έγκυρη και δεν αντιτίθεται στον ορισμό της βάσης μας.
π.χ. Όταν θέλουμε να αλλάξουμε το όνομα χρήστη ενός χρήστη, το σύστημα ελέγχει αν η τιμή είναι αλφαριθμητικό όχι κενό και μικρότερο από 100 χαρακτήρες αφού το πεδίο όνομα χρήστη δέχεται αλφαριθμητικά μεγέθους εώς 100 χαρακτήρες.

User Interface

Για την εμφάνιση κάποιας σελίδας στο χρήστη το σύστημα χρησιμοποιεί ένα απλό template system. Όλος ο HTML κώδικας που χρειαζόμαστε υπάρχει μέσα στο φάκελο *templates/* .
Κάθε φορά που θέλουμε να εμφανίσουμε μια σελίδα στο χρήστη, το σύστημα φορτώνει το βασικό μας layout που βρίσκεται στο *templates/layout.php* και μέσα σε αυτό φορτώνει τη σελίδα και τα δεδομένα που θέλει ο χρήστης να δει. Για να πετύχει το σύστημα την σωστή εμφάνιση χρησιμοποιεί μέσα στα αρχεία των templates κάποια απλά tags τύπου {%content%}. Παρακάτω δίνεται ένα παράδειγμα πως επιτυγχάνεται η εμφάνιση όλων των χρηστών του συστήματος.

```
$script = '<script type="text/javascript" src="js/users.js"></script>';
//Get main layout


$layout = file_get_contents('templates/layout.php');
//Get users template
$template = file_get_contents('templates/users.php');

$page = str_replace("{%content%}", $template, $layout);
$page = str_replace("{%scripts%}", $script, $page);
$page = str_replace("{%googlescripts%}", "", $page);
//Retrieve users from database
$users = getUsers();

if (is_array($users)){
   $table = '<table class="table table-bordered table-alternating user-table">';
   $table .=   '<tr><th>DB ID</th><th>Όνομα
Χρήστη</th><th>Ρόλος</th><th>Δράσεις</th></tr>';
   foreach ($users as $user){
      $table .= '<tr
id="row_'.$user['id'].'"><td>'.$user['id'].'</td><td>'.$user['username'].'</td><td>'.$user['role'
].'</td><td><span data-id="'.$user['id'].'" class="js-link edit-link">Επεξεργασία</span><span
data-id="'.$user['id'].'" class="js-link delete-link">Διαγραφή</span></td>';
   }
   $table .= '</table>';
}else {
   $table = '<table class="table table-bordered table-alternating user-table">
            <tr ><th>DB ID</th><th>Όνομα
Χρήστη</th><th>Ρολος</th><th>Δράσεις</th></tr>
            <tr><td colspan="4" class="empty-table">Δεν βρέθηκαν χρήστες.</td></tr>
            </table>';
}

$page = str_replace("{%userTable%}", $table, $page);

echo $page;
```

Αν δούμε αναλυτικά το παραπάνω κώδικα θα παρατηρήσουμε ότι το σύστημα φορτώνει στη μεταβλητή *$layout* το βασικό μας layout και κάνει κάποιες αντικαταστάσεις στα tags που αναφέραμε παραπάνω.
Τα tags *{%script%}, {%googlescript%}* χρησιμοποιούνται για να φορτώσουμε στη σελίδα τα javascript αρχεία που χρειαζόμαστε. Το tag *{%googlescript%}* συνήθως είναι κενό, εκτός στην σελίδα με τα στατιστικά αποτελέσματα μιας ερώτησης όπου το χρησιμοποιούμε για να δείξουμε το διάγραμμά μας.

Javascript/AJAX

Για την εκτέλεση ασύγχρονων ερωτημάτων προς το σύστημα μας, όπως επίσης και κάποιων απλών λειτουργιών το σύστημα χρησιμοποιεί μηχανισμούς γραμμένους στη γλώσσα javascript. Σε όλες τις σελίδες χρησιμοποιούμε την εξωτερική βιβλιοθήκη JQuery για πιο εύκολη διαχείριση των ασύγχρονων κλήσεων και τη διαχείριση της σελίδας. Όλα τα αρχεία javascript που χρησιμοποιεί το σύστημα βρίσκονται στο φάκελο *js/* και φορτώνονται όπως φαίνεται από το όνομα τους στις αντίστοιχες σελίδες (π.χ το login.js φορτώνεται όταν ο χρήστης βλέπει τη σελίδα σύνδεσης).

Εγκυρότητα Δεδομένων

Μέσω javascript κάνουμε ένα μικρό validation στα δεδομένα που μας έχει δώσει ο χρήστης. Βάζοντας την κλάση required στο input μιας φόρμας, πριν η φόρμα υποβληθεί το σύστημα θα διατρέξει το πεδίο και θα δει αν είναι άδειο. Αν είναι άδειο, θα σταματήσει την υποβολή της φόρμας και θα ενημερώσει το χρήστη ότι πρέπει να βάλει δεδομένα στο συγκεκριμένο πεδίο.

Λειτουργίες στο Πάτημα HTML Component

Σε αρκετά σημεία το σύστημα προσφέρει στο χρήστη λειτουργίες που δεν χρειάζεται να ξαναφορτωθεί η σελίδα για να ολοκληρωθούν (π.χ. η εναλλαγή από Δημιουργία Χρήστη σε Επεξεργασία Χρήστη και το ανάποδο, ή κάποια Διαγραφή). Για αυτό το σκοπό, όπου χρειάζεται κάνουμε "attach" κάποιους listeners, οι οποίοι συνήθως ανταποκρίνονται στο πάτημα του συγκεκριμένου component, με τη χρήση του παρακάτω κώδικα:

```
$('.edit-user-button').on('click', function(e){
   $('.create-user-row').removeClass('hidden');
   $('.edit-user-row').addClass('hidden');
});
```

Στο παραπάνω κώδικα λέμε στο σύστημα ότι όταν ο χρήστης πατήσει το component με κλάση "*edit-user-button*" θα πρέπει να του εμφανίσουμε τη φόρμα Δημιουργίας χρήστη, και να εξαφανίσουμε τη φόρμα Επεξεργασίας Χρήστη.
Το ίδιο συμβαίνει και όταν θέλουμε να κάνουμε κάποια ασύγχρονη υποβολή φόρμας στο σύστημα μας. Παρακάτω φαίνεται πως επιτυγχάνεται αυτό για την Επεξεργασία ενός χρήστη. (Εδώ ο listener "ακούει" τα γεγονότα τύπου submit).

```
$('.edit-user-form').on('submit', function(e){
   var valFlag = true;
   $('.edit-user-form input.required').each(function(){
      if ($.trim($(this).val()).length == 0){
         $(this).addClass('has-error');
         valFlag = valFlag && false;
      }else {
         $(this).removeClass('has-error');
         valFlag = valFlag && true;
      }
   });
   $('.edit-user-form select.required').each(function(){
      if ($.trim($(this).val()).length == 0){
         $(this).addClass('has-error');
         valFlag = valFlag && false;
      }else {
         $(this).removeClass('has-error');
```

```
            valFlag = valFlag && true;
        }
    });
    if (valFlag){
        $.ajax({
            url:'ajax/editUser.php',
            type: 'POST',
            dataType: 'JSON',
            data: {userId:$('#user_id').val(),username: $('#edit_username').val(), password:
$('#edit_password').val(), role: $('#edit_role').val()},
            success: function(data){
                if (data.success){
                    $('.edit-user-row .alert').remove();
                    tr = $('#row_'+data.user.id);
                    $(tr.children()[1]).html(data.user.username);
                    //console.log($(tr.children()));
                    $(tr.children()[2]).html(data.user.role);
                }else {
                    if (data.redirect=="login.php"){
                        window.location.href="logout.php";
                    }
                    $('.edit-user-row .alert').remove();
                    $('.edit-user-row').prepend('<div class="alert alert-error"><p class="text-error text-
center">'+data.error+'</p></div>');
                }
            },
            error: function(){
                alert('Υπήρξε κάποιο λάθος. Παρακαλώ δοκιμάστε ξανά.');
            }
        })
    }
    return false;
});
```

Ασύγχρονα Ερωτήματα (AJAX requests)

Για την υλοποίηση των ασύγχρονων κλήσεων στο σύστημά μας, χρησιμοποιούμε τη μέθοδο *$.ajax()* της βιβλιοθήκης jquery. Έτσι έχουμε πιο εύκολη διαχείριση των δεδομένων που δέχεται, και στέλνει η ασύγχρονη κλήση. Όλες οι δράσεις που γίνονται μέσω ajax υπάρχουν στο φάκελο *ajax/* , και επιστρέφουν αντικείμενα της μορφής JSON. Αύτο μας δίνει τη δυνατότητα να διαχειριστούμε το αποτέλεσμα της κλήσης καλύτερα και να επιστρέψουμε πολλαπλά δεδομένα χωρίς να χρειάζεται κάποια ξεχωριστή μέθοδος για να κάνουμε κάποιο parsing του αποτελέσματος. Όλες οι κλήσεις μέσω ajax επιστρέφουν τουλάχιστον ένα πεδίο success τύπου boolean που μας δείχνει αν η κλήση τελείωσε επιτυχώς ή όχι. Ένα παράδειγμα JSON που μας επιστρέφει η κλήση για τη δημιουργία ενός χρήστη είναι το παρακάτω: *{"success":true,"user":{"username":"ablaoubla3","role":"admin","id":11}}*

```
$.ajax({
            url:'ajax/editUser.php',
            type: 'POST',
            dataType: 'JSON',
            data: {userId:$('#user_id').val(),username: $('#edit_username').val(), password:
$('#edit_password').val(), role: $('#edit_role').val()},
            success: function(data){
                if (data.success){
```

```
            $('.edit-user-row .alert').remove();
            tr = $('#row_'+data.user.id);
            $(tr.children()[1]).html(data.user.username);
            //console.log($(tr.children()));
            $(tr.children()[2]).html(data.user.role);
          }else {
            if (data.redirect=="login.php"){
              window.location.href="logout.php";
            }
            $('.edit-user-row .alert').remove();
            $('.edit-user-row').prepend('<div class="alert alert-error"><p class="text-error text-
center">'+data.error+'</p></div>');
          }
        },
        error: function(){
          alert('Υπήρξε κάποιο λάθος. Παρακαλώ δοκιμάστε ξανά.');
        }
      })
```

Ο παραπάνω κώδικας εκτελεί μία ασύγχρονη κλήση στο σύστημα, συγκεκριμένα επεξεργάζεται τα στοιχεία ενός χρήστη. Η μέθοδος *$.ajax* της jquery βιβλιοθήκης παίρνει ως όρισμα ένα αντικείμενο *PlainObject* (π.χ {data: "Vasilis"}), το οποίο παραμετροποιεί τη κλήση της. Όπως φαίνεται παραπάνω για να πετύχουμε την ασύγχρονη κλήση, χρειαζόμαστε τα εξής κλειδιά/μέλη μέσα στο αντικείμενο που δίνουμε σαν όρισμα:

- url: τη τοποθεσία του αρχείου που θέλουμε να στείλουμε την ασύγχρονη κλήση
- type: ο τύπος του request (πχ POST, GET, HEAD)
- dataType: ο τύπος του αποτελέσματος που περιμένουμε από το request.
- data: ένα PlainObject που περιέχει τα query/post variable που θέλουμε να περάσουμε στην ασύγχρονη κλήση (πχ data:{user_id: 4})
- success: Ένα callback function, που εκτελείτε στη περίπτωση που η κλήση επιτύχει. Όταν δηλαδή η κλήση μας επιστρέψει ένα αποδεκτό αποτέλεσμα. Στο παράδειγμα μας, βλέπουμε ότι στο callback function υπάρχει ένα όρισμα με όνομα *data.* Αυτό το όρισμα περιέχει την απάντηση της κλήσης και είναι τύπου JSON Object.
- error: Callback function που εκτελείτε όταν υπάρξει κάποιο λάθος στη κλήση. (π.χ. Όταν πάρουμε response code 500, ή στη παράδειγμα μας όταν η απάντηση δεν μπορεί να διαβαστεί σαν JSON)

ΦΥΣΙΚΗ ΥΠΟΣΤΑΣΗ ΣΤΟ ΔΙΑΚΟΜΙΣΤΗ

Ακολουθεί μια μικρή επεξήγηση για τη δομή των αρχείων στο διακομιστή.

Στο root φάκελο του web-server μας έχουμε τα αρχεία που κάνουν τη διαχείριση του συστηματος μας. Κάθε αρχείο αντιστοιχεί και σε ένα τομέα του συστήματος.

Λίστα Αρχείων

answers.php	Επεξεργάζεται τη προβολή των απαντήσεων
calls.php	Επεξεργάζεται τις κλήσεις από το android app
index.php	Επεξεργάζεται τη προβολή της αρχικής σελίδας
login.php	Επεξεργάζεται τη προβολή της σελίδας σύνδεσης
logout.php	Επεξεργάζεται την αποσύνδεση του χρήστη
questions.php	Επεξεργάζεται τη προβολή των ερωτήσεων
users.php	Επεξεργάζεται τη προβολή των χρηστών

Πίνακας 1 : λιστα αρχειων server

Τα αρχεία στο φάκελο ajax επεξεργάζονται τις ασύγχρονες κλήσεις που γίνονται στο σύστημα μας.

Λίστα Αρχείων

addAnswer.php	Προσθέτει μια απάντηση σε μια ερώτηση
createQuestion.php	Προσθέτει μια ερώτηση
createUser.php	Προσθέτει ένα χρήστη
deleteAnswer.php	Διαγράφει μια απάντηση
deleteQuestion.php	Διαγράφει μια ερώτηση
deleteUser.php	Διαγράφει ένα χρήστη
editAnswer.php	Επεξεργάζεται μια απάντηση
editQuestion.php	Επεξεργάζεται μια ερώτηση
editUser.php	Επεξεργάζεται ένα χρήστη

Πίνακας 2 : λιστα αρχειων ajax server

Στο φάκελο assets έχουμε το css του συστήματος.

Τα αρχεία στο φάκελο includes περιέχουν κάποιες κοινές μεθόδους για τη διαχείριση του συστήματος μας, ή ξεχωρίζουν επειδή κάνουν συγκεκριμένες λειτουργίες.

Λίστα Αρχείων

database.php	Διαχειρίζεται τη βάση δεδομένων
functions.php	Περιέχει τις μεθόδους ελέγχου εγκυρότητας, αποσύνδεσης
session.php	Διαχειρίζεται το php session

Πίνακας 3 : λιστα αρχειων includes server

Τα αρχεία στο φάκελο js περιέχουν τη javascript που χρειάζεται το σύστημα

Λίστα Αρχείων

jquery-2.1.3.min.php	Εξωτερική βιβλιοθήκη jquery
login.js	Έλεγχος εγκυρότητας τιμών για τη σύνδεση
questions.js	Διαχειρίζεται το ui των σελίδων για τις ερωτήσεις, έλεγχος εγκυρότητας τιμών ερωτήσεων, ασύγχρονες κλήσεις
users.js	Διαχειρίζεται το ui των σελίδων για τους χρήστες, έλεγχος εγκυρότητας τιμών χρηστών, ασύγχρονες κλήσεις

Πίνακας 4 : λιστα αρχειων js server

Τα αρχεία στο φάκελο templates περιέχουν τον HTML κώδικα του συστήματος μας.

Λίστα Αρχείων

answers.php	Το layout για τη σελίδα προβολής ερωτήσεων για τη προβολή απαντήσεων
answerView.php	Το layout για τη σελίδα προβολής των στατιστικών αποτελεσμάτων των απαντήσεων σε μια ερώτηση
index.php	Το layout της αρχικής σελίδας.
layout.php	Αρχικό και κύριο layout.
login.php	Το layout για την σελίδα σύνδεσης
questions.php	Το layout για τη προβολή της λίστας των ερωτήσεων
questionsCreate.php	Το layout για τη δημιουργία ερώτησης
questionsEdit.php	Το layout για την επεξεργασία ερώτησης
users.php	Το layout για τη προβολή των χρηστών

Πίνακας 5 : λιστα αρχειων templates server

4.3 ΕΦΑΡΜΟΓΗ ANDROID

ΣΥΜΒΑΤΟΤΗΤΑ

Η εφαρμογή για το android που μπορεί να χρησιμοποιήσει ο χρήστης είναι συμβατή με 4.1 android sdk και πάνω (SDK VERSION: 16). Η εφαρμογή λειτουργεί το ίδιο και σε οριζόντια και σε κάθετη οθόνη. Η συσκευή θα πρέπει να είναι συνδεδεμένη στο internet έτσι ώστε να μπορούν να γίνουν οι κλήσεις στο σύστημα μας.

//permissions required

Η εφαρμογή για να μπορέσει να διαχειριστεί τις κλήσεις στο διαδίκτυο χρειάζεται τα 3 παρακάτω permissions(άδειες) να είναι ορισμένα στο AndroidManifest.xml.

- android.permission.INTERNET
- android.permission.ACCESS_NETWORK_STATE
- android.permission.ACCESS_WIFI_STATE

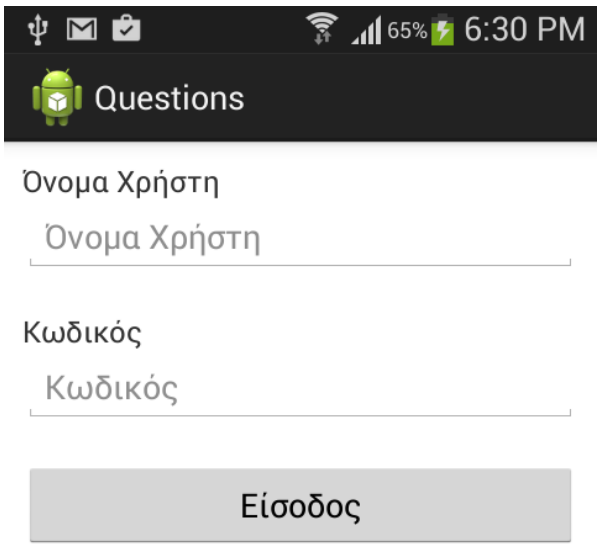
USER INTERFACE (UI)

Για την υλοποίηση του user interface, χρησιμοποιήσαμε τη τεχνολογία των *Fragments* που μας προσφέρει το android sdk. Έχοντας ένα hosting *activity*, η εφαρμογή χρησιμοποιεί δύο *fragments*: LoginFragment.java, QuestionFragment.java.

Το καθένα από τα παραπάνω αναφερόμενα fragments, χρησιμοποιείται για διαφορετικούς λόγους και για να δείξει η εφαρμογή διαφορετικά δεδομένα στο χρήστη.

LoginFragment (Οθόνη Σύνδεσης)

Το Login Fragment, είναι υπεύθυνο για τη παρουσίαση στο χρήστη μιας φόρμας, έτσι ώστε ο χρήστης να μπορεί να εισάγει το Username και το κωδικό του και πατώντας το κουμπί Είσοδος, να γίνει η κλήση στο σύστημα μας για να μπορέσει ο χρήστης να συνδεθεί και να πάρει την ενεργή ερώτηση. Το Fragment αυτό αποτελείτε από δύο textview, δύο edittext και ένα button.



Εικονα 29 : android-app login fragmnet

Αν για κάποιο λόγο ο χρήστης πατήσει το κουμπί Είσοδος, χωρίς να έχει συμπληρώσει τα στοιχεία του τότε θα του εμφανιστεί ένα μήνυμα της μορφής *Toast* ενημερώνοντας τον για το λάθος που έχει συμβεί, χωρίς να έχει γίνει η κλήση στο σύστημα. Αλλιώς το fragment "κάνει" τη κλήση, αρχικοποιώντας ένα ασύγχρονο task (AsyncTask) για να λάβει αποτέλεσμα. Αν ο χρήστης έχει βάλει σωστά τα στοιχεία του τότε το fragment, αφού λάβει το τέλος του παραπάνω task θα σώσει τοπικά τα στοιχεία που χρειαζόμαστε από την απάντηση και θα ενημερώσει το hosting activity ότι πρέπει να δείξει το QuestionFragment.
Αν ο χρήστης δώσει λάθος στοιχεία, η εφαρμογή θα τον ενημερώσει αλλά θα παραμείνει στην οθόνη της σύνδεσης.
Ακολουθεί το xml της οθόνης του LoginFragment

```
<ScrollView xmlns:android= "http://schemas.android.com/apk/res/android"
   android:layout_width= "match_parent"
   android:layout_height= "match_parent">
   <LinearLayout
      android:layout_width= "match_parent"
```

```
android:layout_height="wrap_content"
android:orientation="vertical">
<RelativeLayout
   android:id="@+id/form_username_container"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:padding="10dp">
   <TextView
      android:id="@+id/username_label"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textSize="16dp"
      android:layout_alignParentTop="true"
      android:text="@string/label_username"|>
   <EditText
      android:layout_below="@id/username_label"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:maxLines="1"
      android:inputType="text"
      android:hint="@string/label_username"
      android:id="@+id/username_field"|>
</RelativeLayout>
<RelativeLayout
   android:id="@+id/form_password_container"
    android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:padding="10dp">
   <TextView
      android:id="@+id/password_label"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textSize="16dp"
      android:layout_alignParentTop="true"
```

```
            android:text="@string/label_password"/>
        <EditText
            android:layout_below="@id/password_label"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:maxLines="1"
            android:inputType="textPassword"
            android:hint="@string/label_password"
            android:id="@+id/password_field"/>
    </RelativeLayout>
    <RelativeLayout
        android:id="@+id/form_button_container"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:padding="10dp">
        <Button
            android:id="@+id/login_button"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_login"/>
    </RelativeLayout>
  </LinearLayout>
</ScrollView>
```

Το LoginFragment χρησιμοποιεί τα εξής στοιχεία για τη δημιουργία των γραφικών:

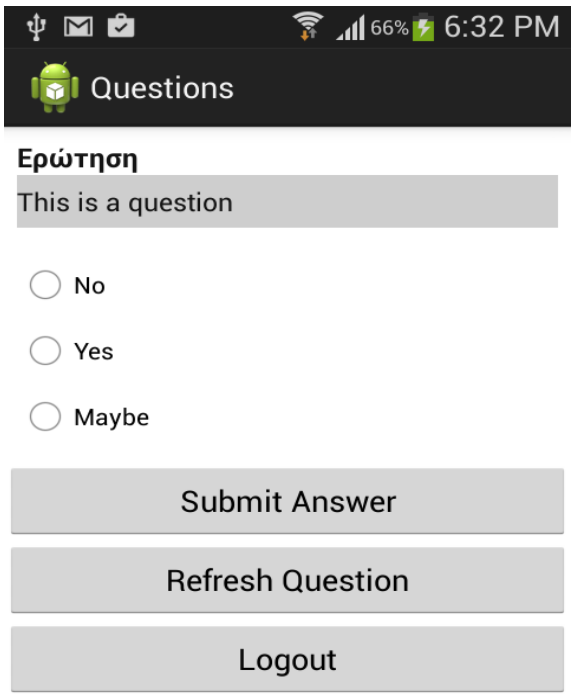
- **ScrollView** το οποίο είναι το root view μας και περιέχει την οθόνη του LoginFragment. Η εφαρμογή χρησιμοποιεί το ScrollView για να μπορεί να είναι συμβατή με διαφόρου τύπου οθόνες, και με τους 2 προσανατολισμούς που έχει ένα android σύστημα (portrait, landscape)
- Αμέσως μετά υπάρχει ένα **LinearLayout** με κάθετο προσανατολισμό. Αυτό γίνεται υποχρεωτικά γιατί το ScrollView δεν μπορεί να περιέχει παραπάνω από 1 άμεσα παιδιά. Το LinearLayout περιέχει όλα τα υπόλοιπα στοιχεία που φαίνονται στην οθόνη του χρήστη.
- **RelativeLayouts** που εσωκλείουν τα στοιχεία της λίστας. Μέσα στο LinearLayout υπάρχουν 3 RelativeLayouts που το καθένα περιέχει ένα στοιχείο της φόρμας μας. Για παράδειγμα έχουμε ένα RelativeLayout που περιέχει το label (την ετικέτα) του πεδίου "Κωδικός" και το στοιχείο που ο χρήστης γράφει το κωδικό του.
- Τα πεδία της φόρμας δημιουργούνται στην οθόνη χρησιμοποιώντας ένα ζευγάρι από textview και edittext. Το textview, η εφαρμογή το χρησιμοποιεί για να δείξει κάποιο κείμενο

στη περίπτωση μας την ετικέτα του πεδίου (Ονομα Χρήστη, Κωδικός). Το στοιχείο edittext χρησιμοποιείται για να μπορεί ο χρήστης να εισάγει (γράψει) κάποιο κείμενο, στη περίπτωση της εφαρμογής, να γράψει τον κωδικό του και το όνομα χρήστη που έχει στο backend σύστημα μας.

- Τέλος, η εφαρμογή χρησιμοποιεί ένα στοιχείο τύπου button, το οποίο όταν ο χρήστης το πατήσει, η εφαρμογή στέλνει το όνομα χρήστη και το κωδικό που έχει πληκτρολογήσει ο χρήστης στο backend για να συνδέσει το χρήστη στο σύστημα μας.

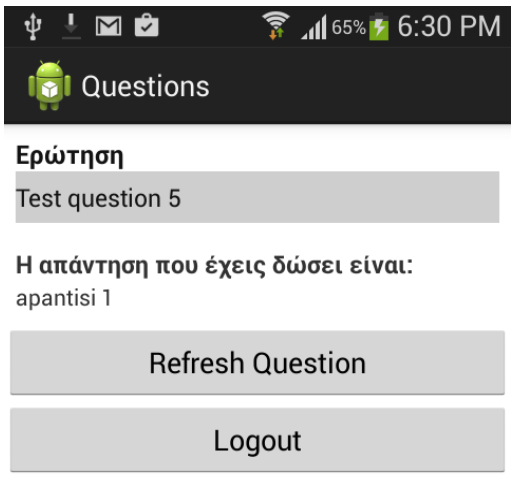
QuestionFragment (Οθόνη Ερώτησης)

Το question fragment είναι υπεύθυνο για τη παρουσίαση της ερώτησης στο χρήστη. Αναλόγως αν υπάρχει ενεργή ερώτηση, αν ο χρήστης έχει απαντήσει στην ερώτηση ή όχι, το fragment μεταβάλλεται για να αντικατοπτρίζει τα σωστά δεδομένα. Παρακάτω φαίνονται οι 3 καταστάσεις που μπορεί να εμφανίζει το fragment.



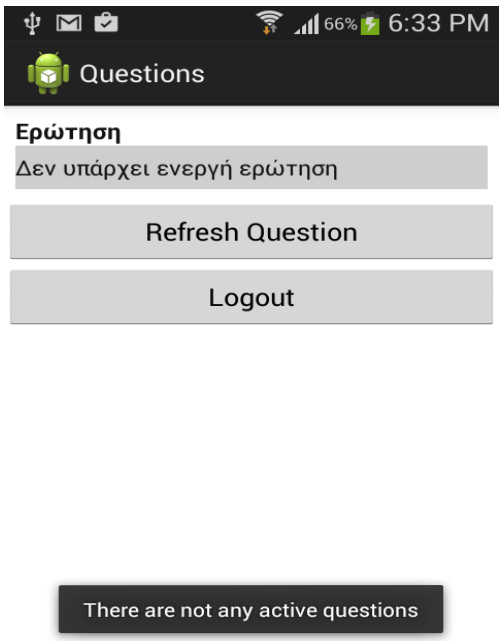
Εικονα 30: android-app question fragment 1

Στη πρώτη οθόνη φαίνεται πως θα είναι το fragment όταν ο χρήστης δεν έχει απαντήσει στην ενεργή ερώτηση.



Εικονα 31: android-app question fragment 2

Στη δεύτερη οθόνη φαίνεται πως θα είναι το fragment όταν ο χρήστης έχει απαντήσει στην ενεργή ερώτηση.



Εικονα 32: android-app question fragment 3

Στη τρίτη οθόνη φαίνεται πως θα είναι το fragment όταν στο σύστημα δεν υπάρχει ενεργή ερώτηση.

Το Question Fragment επιτρέπει στο χρήστη να απαντήσει μια ερώτηση επιλέγοντας την απάντηση που θέλει. Εκτός από τη παραπάνω ενέργεια, το fragment επιτρέπει στο χρήστη μέσω του κουμπιού "Logout" να αποσυνδεθεί από το σύστημα ή να ανανεώσει την ερώτηση.
Για να απαντήσει ο χρήστης σε μία ερώτηση, θα πρέπει να διαλέξει μία απάντηση και να πατήσει το κουμπί "Submit Answer". Πατώντας το κουμπί, το fragment "κάνει" τη κλήση στο σύστημα, αρχικοποιώντας ένα ασύγχρονο task (AsyncTask) για να λάβει αποτέλεσμα. Αν το σύστημα δεχτεί την απάντηση του, τότε το fragment θα ανανεωθεί δείχνοντας την ερώτηση και την απάντηση που επέλεξε ο χρήστης.

Ακολουθεί το xml της οθόνης του QuestionFragment

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
```

```
<RelativeLayout
   android:id="@+id/questioncontainer"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:padding="7dp"
   >
   <TextView
      android:id="@+id/questionlabel"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textColor="#FF131313"
      android:layout_alignParentTop="true"
      android:textSize="16dp"
      android:textStyle="bold"
      android:text="@string/label_question"/>
   <TextView
      android:id="@+id/questiontext"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:textColor="#FF131313"
      android:layout_below="@id/questionlabel"
      android:paddingTop="5dp"
      android:paddingBottom="5dp"
      android:background="#FFCECECE"
      android:textSize="15dp"
      />
</RelativeLayout>
<RelativeLayout
   android:id="@+id/alreadyansweredcontainer"
   android:layout_width="match_parent"
   android:layout_height="wrap_content"
   android:padding="7dp"
   >
   <TextView android:id="@+id/answeredlabel"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="15dp"
        android:textStyle="bold"
        android:layout_alignParentTop="true"
        android:text="Η απάντηση που έχεις δώσει είναι:"/>
    <TextView android:id="@+id/myanswer"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="14dp"
        android:layout_below="@id/answeredlabel"/>
</RelativeLayout>
 <RadioGroup
    android:id="@+id/answerscontainer"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="7dp"
    android:orientation="vertical"
    >
</RadioGroup>
<Button
    android:id="@+id/submit_answer"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Submit Answer"/>
 <Button
    android:id="@+id/refresh_question"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Refresh Question"/>
<Button
    android:id="@+id/logout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
```

android:text="Logout"/>

</LinearLayout>

</ScrollView>

Το QuestionFragment χρησιμοποιεί τα παρακάτω στοιχεία για να δείξει στο χρήστη αυτά που χρειάζεται:

- Όπως και στο LoginFragment τα δύο πρώτα ui στοιχεία που έχει είναι ένα ScrollView και ένα LinearLayout με κάθετο προσανατολισμό.
- Μέσα στο LinearLayout, το πρώτο ui στοιχείο που υπάρχει είναι ένα RelativeLayout το οποίο περιέχει δύο textviews για την εμφάνιση της ενεργής ερώτησης στο χρήστη. Το ένα textview δείχνει την ετικέτα "Ερώτηση" ενώ το άλλο δείχνει την ερώτηση όπως την έχουμε λάβει από το backend σύστημα.
- Μετά το RelativeLayout που περιέχει την ερώτηση, υπάρχει άλλο ένα RelativeLayout (θα το ονομάσουμε **A**) το οποίο δείχνει την απάντηση που έχει επιλέξει ο χρήστης (αν ο χρήστης έχει ήδη απαντήσει στην ενεργή ερώτηση). Αυτό επιτυγχάνεται με τον ίδιο τρόπο, που δείχνουμε την ερώτηση στο χρήστη (δηλαδή δύο textviews το ένα δείχνει την ετικέτα "Η απάντηση που έχεις δώσει είναι:" και το άλλο το κείμενο της απάντησης που έχει ο χρήστης επιλέξει).
- Το παραπάνω RelativeLayout ακολουθείτε από ένα RadioGroup στοιχείο. Το οποίο στον ορισμό της οθόνης μας φαίνεται να είναι κενό. Όμως μέσα σε αύτο το RadioGroup δημιουργούνται και εισάγονται στοιχεία τύπου RadioButton τα οποία το καθένα αντιστοιχεί σε μία από τις πιθανές απαντήσεις που μπορεί να επιλέξει ο χρήστης. Η εφαρμογή δημιουργεί δυναμικά αυτά τα RadioButtons καθώς δεν είναι εφικτό να ξέρουμε από πριν τον αριθμό και τα στοιχεία των απαντήσεων. Το RadioGroup που περιέχει τις απαντήσεις και το RelativeLayout **A** δεν εμφανίζονται και τα δύο ταυτόχρονα στο χρήστη*.
- Έπειτα υπάρχουν 3 ui στοιχεία τύπου Button τα οποία είναι υπεύθυνα για τις ενέργειες που μπορεί να κάνει ο χρήστης.
    1. Κουμπί "Υποβολής Απάντησης". Εμφανίζεται στο χρήστη όταν δεν έχει απαντήσει στην ενεργή ερώτηση και στέλνει την απάντηση που έχει επιλέξει για αποθηκεύσει στο backend σύστημα. (Ξεκινάει ένα AnswerRequest)
    2. Κουμπί "Refresh Question". Ρωτάει το backend σύστημα αν υπάρχει καινούρια ερώτηση προς απάντηση, και αν υπάρχει ανανεώνει το ui (ξεκινάει ένα QuestionRequest).
    3. Κουμπί "Logout". Αποσυνδέει το χρήστη από το σύστημα.

Για τη δυναμική δημιουργία των RadioButton αναλόγως των απαντήσεων που υπάρχουν στο σύστημα το QuestionFragment κάνει μια κλήση στη private μέθοδο fillAnswers.

```
private void fillAnswers(){

        LayoutInflater inflater =
(LayoutInflater)getActivity().getSystemService(Context.LAYOUT_INFLATER_SERVICE);

        answers = sp.getString("QUESTION_ANSWERS", "");
        buttons = new ArrayList<RadioButton>();
```

```
//If the question has answers
if (!answers.equals("")){
	try{
		JSONArray obj = new JSONArray(answers);
		RelativeLayout temp1;
		for (int i=0; i<obj.length(); i++){
			//Get the answer layout
			temp1 =
(RelativeLayout)inflater.inflate(R.layout.answer_layout, answersContainer, false);
			JSONObject jsonString = obj.getJSONObject(i);
			//Fill the texts in answer layout


((RadioButton)temp1.findViewById(R.id.answer_checkbox)).setText(jsonString.getString("text"));

	((RadioButton)temp1.findViewById(R.id.answer_checkbox)).setTag(jsonString.getInt("id")
+"");

	buttons.add((RadioButton)temp1.findViewById(R.id.answer_checkbox));
			//Add it to our view.
			answersContainer.addView(temp1);
		}
		for (int i=0;i<buttons.size();i++){
			RadioButton rb = buttons.get(i);
			//set the listener in each radio button, to get the id of the
answer the user has chosen
			rb.setOnCheckedChangeListener(new
OnCheckedChangeListener (){
				@Override
				public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked){
					if (isChecked){

	processRadioButtonClick(buttonView);
					}
				}
			});
		}
	}catch(Exception e){
		e.printStackTrace();
	}
}else {
	//if the question does not have answer show two buttons with Yes or No
	RelativeLayout temp;
	temp = (RelativeLayout)inflater.inflate(R.layout.answer_layout,
answersContainer, false);
	((RadioButton)temp.findViewById(R.id.answer_checkbox)).setText("NAI");
	((RadioButton)temp.findViewById(R.id.answer_checkbox)).setTag("0");
	buttons.add((RadioButton)temp.findViewById(R.id.answer_checkbox));
	answersContainer.addView(temp);
```

```
                    temp = (RelativeLayout)inflater.inflate(R.layout.answer_layout,
answersContainer, false);
                    ((RadioButton)temp.findViewById(R.id.answer_checkbox)).setText("OXI");
                    ((RadioButton)temp.findViewById(R.id.answer_checkbox)).setTag("1");
                    buttons.add((RadioButton)temp.findViewById(R.id.answer_checkbox));
                    answersContainer.addView(temp);
                    for (int i=0;i<buttons.size();i++){
                        RadioButton rb = buttons.get(i);
                        rb.setOnCheckedChangeListener(new OnCheckedChangeListener
(){
                            @Override
                            public void onCheckedChanged(CompoundButton
buttonView, boolean isChecked){
                                if (isChecked){
                                    processRadioButtonClick(buttonView);
                                }
                            }
                        });
                    }
                }
            }
```

Όπως βλέπουμε παραπάνω η μέθοδος fillAnswer διαβάζει τις απαντήσεις της ερώτησης και σε ένα βρόγχο δημιουργεί τα RadioButtons.
Το κάθε RadioButton έχει ένα κείμενο (το κείμενο της απάντησης) και ένα tag που στη περίπτωση μας είναι το αναγνωριστικό της απάντησης. Μόλις δημιουργήσει το κάθε κουμπί το προσθέτει στο RadioGroup και σε ένα ArrayList όπου έχουμε ορίσει για να κρατάμε τα RadioButtons. Η λίστα αυτή χρησιμεύει στην σωστή λειτουργία του RadioGroup καθώς επειδή τα RadioButtons δεν είναι ορισμένα από την αρχή το android δεν αποκλείει το ένα από το άλλο (δηλαδή θα μπορούσε ο χρήστης να διαλέξει δύο απαντήσεις). Επιτυγχάνουμε τη σωστή λειτουργία των RadioButtons με το να κάνουμε attach ένα listener (*setOnCheckedChangeListener*) ο οποίος κοιτάζει πότε θα αλλάξει το status του κάθε RadioButton. Μόλις ο χρήστης επιλέξει κάποιο RadioButton τότε διατρέχουμε το πίνακα με τα RadioButtons που έχουμε αποθηκεύσει και απενεργοποιούμε όλα τα RadioButtons εκτός από αυτό που επέλεξε ο χρήστης.

```
private void processRadioButtonClick(CompoundButton buttonView){
        for(int i=0; i<buttons.size();i++){
            RadioButton rb = buttons.get(i);
            if (rb != buttonView){
                rb.setChecked(false);
            }
        }
    }
```

* Όταν το QuestionFragment γίνεται update(onResume ή λάβει ότι τελείωσε μία από τις κλήσεις) τότε ελέγχουμε αν υπάρχει ενεργή ερώτηση. Αν δεν υπάρχει ενεργή ερώτηση τότε με το παρακάτω κώδικα "κρύβουμε" από το χρήστη το RelativeLayout A και το RadioGroup

answersContainer.setVisibility(View.*GONE*); (κρύβει το RadioGroup)
submit.setVisibility(View.*GONE*); (κρύβει το κουμπί "Υποβολής Ερώτησης")
alreadyContainer.setVisibility(View.*GONE*); (κρύβει το RelativeLayout A)

Αν υπάρχει ερώτηση αλλά ο χρήστης έχει ήδη απαντήσει τότε πρέπει να κρύψουμε το RadioGroup, και το κουμπί "Υποβολή Ερώτησης".

answersContainer.setVisibility(View.*GONE*); (κρύβει το RadioGroup)
submit.setVisibility(View.*GONE*); (κρύβει το κουμπί "Υποβολής Ερώτησης")
alreadyContainer.setVisibility(View.*VISIBLE*); (εμφανίζει το RelativeLayout A)

Αν υπάρχει ερώτηση και ο χρήστης δεν έχει απαντήσει τότε πρέπει να εμφανίσουμε το RadioGroup μαζί με τα RadioButtons που περιέχει και το κουμπί "Υποβολή Ερώτησης" ενώ θα πρέπει να εξαφανίσουμε το RelativeLayout A.

answersContainer.setVisibility(View.*VISIBLE*); (εμφανίζει το RadioGroup)
submit.setVisibility(View.*VISIBLE*); (εμφανίζει το κουμπί "Υποβολής Ερώτησης")
alreadyContainer.setVisibility(View.*GONE*); (κρύβει το RelativeLayout A)

ΚΛΗΣΕΙΣ ΣΤΟ BACKEND ΣΥΣΤΗΜΑ

Επειδή το android sdk δεν επιτρέπει συνδέσεις δικτύου στο κύριο thread μιας εφαρμογής, για να κάνουμε τις κλήσεις στο σύστημα μας, η εφαρμογή χρησιμοποιεί την κλάση AsyncTask που μας προσφέρει το android sdk. Η κλάση AsyncTask αναλαμβάνει την εκτέλεση της κλήσης, δημιουργώντας ένα νέο background thread για να τρέξει.
Όλα τα request περιμένουν την επιστροφή ενός έγκυρου αλφαριθμητικού τύπου JSON.
Όλες οι κλήσεις στο σύστημα γίνονται στο αρχείο calls.php και διαχωρίζονται βάση του *action* query variable. Τα επιτρεπτά actions είναι τα παρακάτω:

- login
- answer
- fetch

Η εφαρμογή περιλαμβάνει 3 AsyncTask για την επικοινωνία με το σύστημα μας.

LoginRequest

Είναι υπεύθυνη για την είσοδο του χρήστη στο σύστημα. Κάνει την σύνδεση στο σύστημα καλώντας τη μέθοδος *getStreamLogin* από τη κλάση Utilities. Μόλις γίνει η κλήση και λάβουμε κάποια απάντηση από το σύστημα τότε καλείτε η μέθοδος *onPostExecute* της AsyncTask, που είναι υπεύθυνη για να ενημερώσει το αντίστοιχο τμήμα της εφαρμογής ότι η κλήση της τελείωσε. Στη περίπτωση της εφαρμογής μας ενημερώνεται το LoginFragment, έτσι ώστε να αποθηκεύσει τα δεδομένα που χρειαζόμαστε και αναλόγως αν ο χρήστης συνδέθηκε επιτυχώς να ενημερώσει την εφαρμογή ότι πρέπει να δείξει το question fragment.
Η κλήση της login request είναι τύπου POST και χρειάζεται 3 πεδία: action, username, password
Ακολουθεί ο κώδικας της κλάσης που είναι υπεύθυνη για να κάνει τη κλήση και να λάβει μια απάντηση από το backend σύστημα.

```
public class LoginRequest extends AsyncTask<Void, Void, JSONObject>{

	private Context _context;
	private ProgressDialog progressDialog;
	private String username, password;
	private LoginCallback callback;

	public LoginRequest(Context context, ProgressDialog dialog, String username, String
password, LoginCallback callback){
		this._context = context;
		this.progressDialog = dialog;
		this.username = username;
		this.password = password;
		this.callback = callback;
	}

	protected void onPreExecute(){
		//show progress dialog, so the user knows that a request is being made
		progressDialog.setMessage("Πραγματοποιείται η είσοδος ...");
		progressDialog.show();
		super.onPreExecute();
	}

	@Override
	protected JSONObject doInBackground(Void... params){
		JSONObject json = null;
		//Check to see if device is connected to network.
		if (Utilities.hasNetwork(_context)){
			json = Utilities.getStreamLogin(username, password);
			Log.e("JSON", json.toString());
		}
		return json;
	}

	@Override
	protected void onPostExecute(JSONObject result){
		try{
			if (null != progressDialog && progressDialog.isShowing()) {
				progressDialog.dismiss();
			}
		}catch(Exception e){
			e.printStackTrace();
		}

		if (result == null){
			Toast.makeText(this._context, "Login Failed",
Toast.LENGTH_LONG).show();
			return;
```

```
            }
            try{
                //Unsuccessful login attempt.
                if (result.has("success") && result.getBoolean("success") == false){
                    Toast.makeText(this._context, result.getString("error"),
Toast.LENGTH_LONG).show();
                    return;
                }
                if (result.has("success")){
                    boolean success = result.getBoolean("success");
                    if (success){
                        //successfull login attempt, but the user has already
answered the active question
                        if (result.has("msg")){
                            Toast.makeText(this._context, "Login success, but
question already answered", Toast.LENGTH_LONG).show();
                            if (callback != null){
                                //update ui, save question and user locally
                                this.callback.loginSucceed(username,
password, result.getString("question"), result.getInt("qid"), result.getString("answers"),
result.getInt("uid"),result.getString("answer_num"));
                            }
                        }else if (result.has("error")){
                            //successfull attempt but there is not an active
question
                            Toast.makeText(this._context, "Login success, but
there is not an active question", Toast.LENGTH_LONG).show();
                            if (callback != null){
                                //update ui, save user locally
                                this.callback.loginFailed(username,
password, result.getInt("uid"),result.getString("error"));
                            }
                        }else {
                            //fallback - successfull attempt
                            Toast.makeText(this._context, "Login success",
Toast.LENGTH_LONG).show();
                            if (callback != null){
                                //update ui - save user and question locally
                                this.callback.loginSucceed(username,
password, result.getString("question"), result.getInt("qid"), result.getString("answers"),
result.getInt("uid"),null);
                            }
                        }
                    }else{
                        Toast.makeText(this._context, "Login Failed",
Toast.LENGTH_LONG).show();
                    }
                }else {
                    Toast.makeText(this._context, "Login Failed",
Toast.LENGTH_LONG).show();
                }
```

```
            }catch(Exception e){
                e.printStackTrace();
            }
        }

}
```

Κάθε AsyncTask στην εφαρμογή μας εκτελεί τις παρακάτω μεθόδους σειριακά

1. **onPreExecute**: εκτελείτε πριν την doInBackground, τρέχει στο main thread της εφαρμογής
2. **doInBackround**: είναι η μέθοδος που ουσιαστικά ξεκινάει την σύνδεση με το σύστημα και περιμένει την απάντηση της κλήσης, τρέχει σε ξεχωριστό thread.
3. **onPostExecute**: Εκτελείτε μετά το πέρας της doInBackground, τρέχει στο main thread της εφαρμογής.

Στη παραπάνω κλάση βλέπουμε ότι και οι 3 προαναφερθείσες μέθοδοι είναι ορισμένες.

Το **LoginCallback** που χρειάζεται η κλάση για να δημιουργηθεί είναι ένα interface που κάνει implement το LoginFragment, για να μπορεί να ενημερωθεί για το τέλος του AsyncTask. Μόλις η εφαρμογή δημιουργήσει ένα αντικείμενο τύπου LoginRequest και καλέσει την execute τότε συμβαίνουν τα παρακάτω:

- Η onPreExecute εμφανίζει ένα progress dialog που ενημερώνει το χρήστη ότι γίνεται μία κλήση στο σύστημα μας.
- Μόλις το dialog εμφανιστεί και έχει τελειώσει η κλήση της onPreExecute, τότε ξεκινάει η doInBackround σε ξεχωριστό thread.
- Η doInBackground ελέγχει αν υπάρχει συνδεσιμότητα με το διαδίκτυο. Αν η εφαρμογή μπορεί να συνδεθεί στο σύστημα τότε πραγματοποιεί μια κλήση στη μέθοδο Utilities.getStreamLogin που είναι υπεύθυνη για τη δημιουργία της σύνδεσης και την παραλαβή του αποτελέσματος. Αν η εφαρμογή δεν μπορεί να συνδεθεί επιστρέφουμε ένα null JSONObject.
- Μόλις τελειώσει η κλήση της doInBackground, τότε ξεκινάει να εκτελείτε η onPostExecute.
- Η onPostExecute έχοντας λάβει σαν όρισμα το JSONObject που επιστρέφει η doInBackground, πρώτα εξαφανίζει το progress dialog που φαίνεται στην οθόνη, για να καταλάβει ο χρήστης ότι η κλήση τελείωσε και έπειτα επεξεργάζεται το JSONObject όρισμα.
- Αν το όρισμα (στη περίπτωση μας η μεταβλητή result) είναι null τότε πάει να πει ότι ή κάποιο λάθος έγινε κατά τη διάρκεια της κλήσης (δεν λάβαμε αλφαριθμητικό τύπου JSON ή δεν μπορέσαμε να συνδεθούμε στο σύστημα) και ενημερώνει με κάποιο *TOAST* μήνυμα το χρήστη.
- Αν το JSONObject έχει μέλος με κλειδί success και η τιμή του είναι false, τότε πάλι ενημερώνετε ο χρήστης με κάποιο TOAST μήνυμα που έχει σαν κείμενο το λάθος που συνέβει (μέλος error στο JSONObject).
- Αν η τιμή του κλειδιού success έχει τη τιμή true, τότε η onPostExecute βλέπει αν υπάρχει πρώτα κάποιο μέλος στο JSONObject μας με κλειδί msg, και μετά με κλειδί error. Στη πρώτη

περίπτωση ξέρουμε ότι ο χρήστης έχει απαντήσει στην ενεργή ερώτηση, τον ενημερώνουμε με ένα *TOAST* μήνυμα, και αν το callback δεν είναι null, ενημερώνουμε το user interface, σώνουμε τα στοιχεία του χρήστη και τα στοιχεία της ενεργής ερώτηση τοπικά. Αν υπάρχει το κλειδί error, τότε το σύστημα δεν έχει κάποια ενεργή ερώτηση αυτή τη στιγμή οπότε ενημερώνουμε το χρήστη όπως και παραπάνω, και αν το callback δεν είναι null, ενημερώνουμε το user interface και σώνουμε τα στοιχεία χρήστη τοπικά.

Αν δεν υπάρχουν τίποτα από τα δύο τότε ο χρήστης έχει συνδεθεί επιτυχώς και αν το callback δεν είναι null, ενημερώνουμε το user interface και σώνουμε τοπικά τα στοιχεία του χρήστη και τα στοιχεία της ερώτησης.

*Με τον ίδιο τρόπο λειτουργούν και τα υπόλοιπα AsyncTasks που έχει η εφαρμογή.

QuestionRequest

Είναι υπεύθυνη για την ανανέωση της ερώτησης που δείχνει η εφαρμογή. Κάνει την σύνδεση στο σύστημα καλώντας τη μέθοδο *getStreamQuestion* από τη κλάση Utilities. Μόλις γίνει η κλήση και λάβουμε κάποια απάντηση από το σύστημα τότε καλείτε η μέθοδος *onPostExecute* της AsyncTask, που είναι υπεύθυνη για να ενημερώσει το αντίστοιχο τμήμα της εφαρμογής ότι η κλήση της τελείωσε. Στη περίπτωση της εφαρμογής μας ενημερώνεται το QuestionFragment, έτσι ώστε να αποθηκεύσει τα δεδομένα που χρειαζόμαστε και αναλόγως να ενημερώσει το user interface, δείχνοντας τη σωστή ερώτηση αν αυτή έχει αλλάξει, και τις απαντήσεις αν ο χρήστης δεν έχει απαντήσει στην ενεργή ερώτηση.
Η κλήση της question request είναι τύπου POST και χρειάζεται 3 πεδία: action, username, password

AnswerRequest

Είναι υπεύθυνη για την αποστολή της απάντησης που έχει επιλέξει ο χρήστης στο σύστημα. Κάνει την σύνδεση στο σύστημα καλώντας τη μέθοδο *getStreamAnswer* από τη κλάση Utilities. Μόλις γίνει η κλήση και λάβουμε κάποια απάντηση από το σύστημα τότε καλείτε η μέθοδος *onPostExecute* της AsyncTask, που είναι υπεύθυνη για να ενημερώσει το αντίστοιχο τμήμα της εφαρμογής ότι η κλήση της τελείωσε. Στη περίπτωση της εφαρμογής μας ενημερώνεται το QuestionFragment, έτσι ώστε να αποθηκεύσει τα δεδομένα που χρειαζόμαστε και αναλόγως να ενημερώσει το user interface, δείχνοντας αφαιρώντας την επιλογή απαντήσεων και δείχνοντας την απάντηση που έχει καταχωρηθεί στο σύστημα.
Η κλήση της answer request είναι τύπου POST και χρειάζεται 5 πεδία: action, uid(αναγνωριστικό χρήστη), qid(αναγνωριστικό ερώτησης), aid(αναγνωριστικό απάντησης), text(κείμενο απάντησης).
Η κλάση Utilities της εφαρμογής περιέχει τις μεθόδους όπου κάνουν την σύνδεση στο backend σύστημα. Και οι 3 μέθοδοι που χρησιμοποιούνται *getStreamLogin, getStreamAnswer, getStreamQuestion* λειτουργούν με παρόμοιο τρόπο.
Χρησιμοποιώντας τη κλάση HttpURLConnection που μας προσφέρει η java, ανοίγουν μία σύνδεση στο σύστημα μας και γράφουν στο outputStream της σύνδεσης μας το query που θέλουμε.

π.χ.
action=answer&uid="+userId+"&qid="+questionId+"&aid="+answerId+"&text="+text

Μόλις η εγγραφή τελειώσει, τότε πρώτα καλώντας την *getResponseCode* για να βεβαιωθεί ότι η σύνδεση έγινε επιτυχώς, και μετά καλώντας την *getInputStream* πάνω στη σύνδεση η εκάστοτε μέθοδος μπορεί και διαβάζει το αλφαριθμητικό που έχει στείλει το σύστημα και το μετατρέπει σε ένα αντικείμενο τύπου JSON που το επιστρέφει στην μέθοδο *doInBackground* της AsyncTask που την κάλεσε.
Ακολουθεί ο κώδικας μίας εκ των τριών μεθόδων που χρησιμοποιεί η εφαρμογή για να συνδεθεί (να κάνει μια κλήση στο σύστημα μας) με το backend συστημα.
Συγκεκριμένα θα δείξουμε την **getStreamAnswer** καθώς οι υπόλοιπες δύο λειτουργούν με τον ίδιο τρόπο.

```
public static JSONObject getStreamAnswer(String questionId, String answerId, String text, String userId){

		JSONObject jsonObj = null;
		try{
			//request fields
			String query = "action=answer&uid="+userId+"&qid="+questionId+"&aid="+answerId+"&text="+text;

			URL url = new URL("http://androidthesis.16mb.com/calls.php");
			HttpURLConnection connection = (HttpURLConnection)url.openConnection();
			//Set the response request property.
			connection.setRequestProperty("Accept", "application/json, text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8");
			connection.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");


			connection.setDoOutput(true);
			//set the request method to POST
			connection.setRequestMethod("POST");
			connection.setReadTimeout(10000);
			//send the request fields to backend
			Writer writer = new OutputStreamWriter(connection.getOutputStream());
		writer.write(query);
		writer.flush();
		writer.close();
		int con = connection.getResponseCode();
		//if response code is within the below switch an error has happened.
		switch (con){

			case 500:
			case 502:
			case 404:
			case 405:
			case 408:
			case 409:
			case 403:
```

```
                    case 504:
                    case 406:
                    case -1:
                            connection.disconnect();
                            throw new Exception(con+"");
                    default:break;
                    }
            InputStream is = null;
                    try{
                            //read and store the response stream as a string.
                            is = connection.getInputStream();
                            BufferedReader reader = new BufferedReader(new
InputStreamReader(is,"utf-8"),8);
                            StringBuilder sb = new StringBuilder();
                            String line=null;
                            while((line=reader.readLine())!=null){
                                    sb.append(line);
                            }
                            is.close();
                            connection.disconnect();
                            Log.e("STREAM", sb.toString());
                            //convert response string to jsonobject
                            jsonObj = new JSONObject(sb.toString());
                    }catch(Exception e){
                            e.printStackTrace();
                    }
            }catch(Exception e){
                    e.printStackTrace();
            }
            return jsonObj;
    }
```

Η παραπάνω μέθοδος στέλνει στο σύστημα μας την απάντηση που έχει επιλέξει ο χρήστης σε μία συγκεκριμένη ερώτηση. Όπως θα δούμε και στο παραπάνω κομμάτι κώδικα, στην αρχή ορίζουμε τις POST μεταβλητές που θέλουμε να στείλουμε. Αυτές είναι οι εξής:

- **action**: ορίζει ποια δράση θέλουμε να πραγματοποιήσουμε
- **uid**: το αναγνωριστικό του χρήστη που στέλνει την απάντηση
- **qid**: το αναγνωριστικο της ερώτησης που απάντησε ο χρήστης
- **aid**: το αναγνωριστικό της απάντησης που επέλεξε ο χρήστης
- **text**: το κείμενο της απάντησης που επέλεξε ο χρήστης.

Έπειτα χρησιμοποιώντας τη κλάση **HttpURLConnection** που προσφέρει η JAVA ανοίγουμε μία σύνδεση με το σύστημά μας. Μετά ορίζουμε ότι η σύνδεση αυτή θα πρέπει να λάβει σαν απάντηση ένα application/json αλφαριθμητικό

***(connection.setRequestProperty("Accept", "application/json)*;.**

Μετά επειδή ο τύπος του request είναι POST και η εφαρμογή πρέπει να στείλει τις παραπάνω μεταβλητές γράφουμε στη σύνδεση τις μεταβλητές σαν αλφαριθμητικό. Αυτό επιτυγχάνεται με το παρακάτω κώδικα

```
Writer writer = new OutputStreamWriter(connection.getOutputStream());
writer.write(query);
 writer.flush();
 writer.close();
```

Έπειτα καλώντας την getResponseCode , κάνουμε το request και βλέπουμε το HTTP Response Code. Σε περίπτωση λάθους (responsecode = 500, 404 κτλ) ενεργοποιούμε ένα exception.
Αν περάσει ο έλεγχος του response code τότε προσπαθούμε να διαβάσουμε το αποτέλεσμα της κλήσης χρησιμοποιώντας τη κλάση BufferedReader της JAVA που μπορεί και διαβάζει ένα inputstream. Αποθηκεύουμε την απάντηση σε ένα αντικείμενο τύπου StringBuilder και το μετατρέπουμε σε JSON. Σε περίπτωση που η απάντηση δεν μπορεί να μετατραπεί σε JSON τότε ενεργοποιούμε ένα exception και η μέθοδος μας επιστρέφει null.

Αυτά φαίνονται στο παρακάτων κομμάτι κώδικα.

```
InputStream is = null;
                        try{
                                //read and store the response stream as a string.
                                is = connection.getInputStream();
                                BufferedReader reader = new BufferedReader(new
InputStreamReader(is,"utf-8"),8);
                                StringBuilder sb = new StringBuilder();
                                String line=null;
                                while((line=reader.readLine())!=null){
                                        sb.append(line);
                                }
                                is.close();
                                connection.disconnect();
                                Log.e("STREAM", sb.toString());
                                //convert response string to jsonobject
                                jsonObj = new JSONObject(sb.toString());
                        }catch(Exception e){
                                e.printStackTrace();    }
```

LOCAL STORAGE (τοπική αποθήκευση δεδομένων)

Η εφαρμογή για να λειτουργήσει σωστά πρέπει να αποθηκεύει κάποια δεδομένα τοπικά. Αυτό γίνετε με τη χρησιμοποίηση της κλάσης SharedPreferences του android sdk. Η κλάση αυτή δημιουργεί ένα αρχείο που μέσα περιέχει ζευγάρια από κλειδιά και τιμές. Το αρχείο αυτό είναι προσπελάσιμο μόνο από τη συγκεκριμένη εφαρμογή.

Η εφαρμογή χρειάζεται να αποθηκεύει τις παρακάτω τιμές για την σωστή λειτουργία της:

- Αναγνωριστικό Χρήστη
- Όνομα Χρήστη
- Κωδικός Χρήστη
- Αναγνωριστικό Ερώτησης
- Κείμενο Ερώτησης
- Αναγνωριστικό Απάντησης
- Κείμενο Απάντησης/Απαντήσεων
- Δείχτης για την ανανέωση της ερώτησης

Κρατώντας τα 3 πρώτα στοιχεία η εφαρμογή ξέρει ότι ο χρήστης έχει συνδεθεί επιτυχώς με το σύστημα. Τα υπόλοιπα (εκτός από το "δείχτη ανανέωσης ερώτησης) στοιχεία τα χρησιμοποιεί για να δείξει την question fragment σε περίπτωση που η εφαρμογή δεν έχει πρόσβαση στο διαδίκτυο.
Ο δείχτης ανανέωσης της ερώτησης χρησιμοποιείται όταν η εφαρμογή πάει στο background της συσκευής ή κλείσει ενώ είμαστε στο question fragment.

Ακολουθεί ο πίνακας κλειδιών - τύπου για τη τοπική αποθήκευση δεδομένων.

QUESTIONS_USERNAME	Αλφαριθμητικό (string)
QUESTIONS_PASSWORD	Αλφαριθμητικό (string)
QUESTIONS_USER_ID	Ακέραιος (integer)
QUESTIONS_TEXT	Αλφαριθμητικό (string)
QUESTIONS_ANSWER_VALUE	Αλφαριθμητικό (string)
QUESTIONS_ANSWERS	Αλφαριθμητικό (string)
QUESTIONS_QUESTION_ID	Ακέραιος (integer)
SHOULD_MAKE_REQUEST	Ακέραιος (integer)

Πίνακας 6 : πίνακας κλειδιών local storage

Ακολουθεί πίνακας επεξήγησης κλειδιών.

QUESTIONS_USERNAME	Περιέχει το όνομα χρήστη του συνδεδεμένου χρήστη
QUESTIONS_PASSWORD	Περιέχει το κωδικό του συνδεδεμένου χρήστη
QUESTIONS_USER_ID	Περιέχει το αναγνωριστικό του συνδεδεμένου χρήστη
QUESTIONS_TEXT	Περιέχει το κείμενο της ερώτησης

QUESTIONS_ANSWER_VALUE	Περιέχει την απάντηση που έχει δώσει ο χρήστης
QUESTIONS_ANSWERS	Περιέχει τις διαθέσιμες απαντήσεις στην ενεργή ερώτηση
QUESTIONS_QUESTION_ID	Περιέχει το αναγνωριστικό Ερώτησης
SHOULD_MAKE_REQUEST	Δείχνει αν πρέπει να γίνει κλήση στο σύστημα για να ανανεώσει η εφαρμογή την ερώτηση που δείχνει

Πίνακας 7 : πίνακας επεξήγησης κλειδιών

5.Αποτελέσματα

Το τελικό αποτέλεσμα το οποίο δημιουργήθηκε είναι ένα σύστημα το οποίο λειτουργεί αποτελεσματικά και μπορεί να χρησιμοποιηθεί με ελάχιστες παραμετροποιήσεις από διάφορους οργανισμούς όπως πανεπιστήμια , εταιρίες άλλα και κυβερνήσεις.

Μελλοντική Εργασία και Επεκτάσεις

Σαν μελλοντική επέκταση του συστήματος θα μπορούσαν να προστεθούν κάποιες τροποιήσεις στο θέμα της ασφάλειας.

[1] **Android JSON Parsing - Retrieve From MySQL Database**
https://www.simplifiedcoding.net/android-json-parsing-retrieve-from-mysql-database

[2] **SQL**
https://el.wikipedia.org/wiki/SQL
[3] **PHP**
https://el.wikipedia.org/wiki/PHP
[4] **Javascript**
https://el.wikipedia.org/wiki/JavaScript
[5] **Ajax**
https://en.wikipedia.org/wiki/Ajax_(programming)
[6] **Css**
https://en.wikipedia.org/wiki/Cascading_Style_Sheets
[7] **Html**
https://el.wikipedia.org/wiki/HTML
[8] **Java**
https://en.wikipedia.org/wiki/Java_(programming_language)
[9] Json
https://en.wikipedia.org/wiki/JSON
[10] **Μοντέλο πελάτη-διακομιστή**
https://el.wikipedia.org/wiki/%CE%9C%CE%BF%CE%BD%CF%84%CE%AD%CE%BB%CE%BF_%CF%80%CE%B5%CE%BB%CE%AC%CF%84%CE%B7-%CE%B4%CE%B9%CE%B1%CE%BA%CE%BF%CE%BC%CE%B9%CF%83%CF%84%CE%AE
[11] **electronic voting**
https://en.wikipedia.org/wiki/Electronic_voting
[12] **SSL**
https://el.wikipedia.org/wiki/SSL
[13] **Android**
https://el.wikipedia.org/wiki/Android
[14] **w3schools**
http://www.w3schools.com/
[15] **StackOverFlow**
http://stackoverflow.com/
[16] **php tutorial**
http://www.php.net/
[17] **Android**
https://www.quora.com/Who-founded-Android-Inc