ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Ανάπτυξη εφαρμογής απομακρυσμένης διαχείρισης εφαρμογών κινητών συσκευών

Παπαδόπουλος Παναγιώτης

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Υπεύθυνοι

Λουκόπουλος Αθανάσιος
Λέκτορας του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας

Αναγνωστόπουλος Ιωάννης
Επίκουρος Καθηγητής του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας

Λαμία, 7/2015

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΜΕ ΕΦΑΡΜΟΓΕΣ ΣΤΗ ΒΙΟΪΑΤΡΙΚΗ

Ανάπτυξη εφαρμογής απομακρυσμένης διαχείρισης εφαρμογών κινητών συσκευών

Παπαδόπουλος Παναγιώτης

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Επιβλέποντες

Λουκόπουλος Αθανάσιος, Λέκτορας του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας

Αναγνωστόπουλος Ιωάννης, Επίκουρος Καθηγητής του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας

Λαμία, 7/2015

Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

1. *Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

2. *Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. *Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια*

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: 06/07/2015

Ο – Η Δηλ.

(Υπογραφή)

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.

Ανάπτυξη εφαρμογής απομακρυσμένης διαχείρισης εφαρμογών κινητών συσκευών

Παπαδόπουλος Παναγιώτης

Τριμελής Επιτροπή:

Λουκόπουλος Αθανάσιος, Λέκτορας του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας (επιβλέπων)
Αναγνωστόπουλος Ιωάννης, Επίκουρος Καθηγητής του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας (επιβλέπων)
Αδάμ Μαρία, Επίκουρος Καθηγήτρια του Τμήματος Πληροφορικής με Εφαρμογές στη Βιοϊατρική του Πανεπιστημίου Θεσσαλίας

Η πτυχιακή εργασία είναι αφιερωμένη στους γονείς μου για την υποστήριξη και την υπομονή που έκαναν όλα αυτά τα χρόνια!

Και ακόμα δεν αρχίσαμε… Τα καλύτερα έρχονται!!!

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία εστιάζεται στη υλοποίηση μιας εφαρμογής για την απομακρυσμένη διαχείριση εφαρμογών που είναι εγκατεστημένες σε κινητές συσκευές. Το όνομά της είναι **Remote API Caller (RAC) [16]**. Τμήματα αυτή της εργασίας δημοσιεύθηκαν στο PCI Conference που θα γίνει τον Οκτώβριο [17]. Η εφαρμογή στέλνει τις εκάστοτε εντολές για τη διαχείριση των άλλων εφαρμογών και επίσης λαμβάνει και εκτελεί αυτές τις εντολές, δηλαδή λειτουργεί και ως αποστολέας (sender) και ως παραλήπτης (receiver). Όλες οι εντολές όπως και κάποια από τα χαρακτηριστικά της κάθε συσκευής αποθηκεύονται σε μια βάση δεδομένων σε ένα διαδικτυακό διακομιστή (web server). Όταν η εφαρμογή λειτουργεί ως αποστολέας, στέλνει την εντολή που πρακτικά είναι ένα SQL ερώτημα μέσω ενός web service που είναι υλοποιημένο σε PHP στον server και όταν λειτουργεί ως παραλήπτης παίρνει από τον server όλες τις εντολές που αφορούν την συσκευή που διαχειρίζεται και τις εκτελεί. Η εφαρμογή είναι συμβατή με κινητές συσκευές που το λειτουργικό τους σύστημα είναι το Android και ο κώδικας είναι γραμμένος σε Java με χρήση επιπλέον βιβλιοθηκών που δίνονται από την Google για την ανάπτυξη τέτοιου είδους εφαρμογών.

ABSTRACT

This thesis focuses on an implementation for remote management applications installed on mobile devices. It is called **Remote API Caller (RAC) [16]**. Parts of this work were published in the PCI Conference that will take place in October [17]. The application sends commands to manage other applications and also receives and executes these commands, acts as sender and receiver. All commands like some of the features of each device are stored in a database on a web server. When the application runs as the sender, sends the command which is a SQL query via a web service that is implemented in PHP on the server and when operating as a receiver gets from the server all the commands related to the device that manages and execute them. The application is compatible with mobile devices that their operating system is the Android and the code is written in Java using additional libraries provided by Google for the development of such applications.

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ

Η πτυχιακή εργασία χωρίζεται σε εφτά κεφάλαια. Σε αυτό το κεφάλαιο γίνεται η εισαγωγή και η ανάλυση εννοιών που θα μας χρειαστούν αργότερα όπως είναι : το λειτουργικό σύστημα Android, η γλώσσα διαχείρισης βάσεων δεδομένων SQL, το σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων MySQL και μία επέκτασή της η MySQLi, η γλώσσα σεναρίου PHP, τα web services και το JSON format. Επίσης θα δούμε τι είναι το RPC και το Java RMI.

ANDROID

Το **Android** [1] είναι λειτουργικό σύστημα για συσκευές κινητής τηλεφωνίας το οποίο τρέχει τον πυρήνα του λειτουργικού Linux. Αρχικά αναπτύχθηκε από την Google και αργότερα από την Open Handset Alliance. Επιτρέπει στους κατασκευαστές λογισμικού να συνθέτουν κώδικα με την χρήση της γλώσσας προγραμματισμού Java, ελέγχοντας την συσκευή μέσω βιβλιοθηκών λογισμικού ανεπτυγμένων από την Google. Το Android είναι κατά κύριο λόγο σχεδιασμένο για συσκευές με οθόνη αφής, όπως τα έξυπνα τηλέφωνα και τα τάμπλετ, με διαφορετικό περιβάλλον χρήσης για τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Παρόλο που έχει αναπτυχθεί για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί σε κονσόλες παιχνιδιών, ψηφιακές φυτογραφικές μηχανές, συνηθισμένους Η/Υ και σε άλλες ηλεκτρονικές συσκευές.

Η πρώτη παρουσίαση της πλατφόρμας Android έγινε στις 5 Νοεμβρίου 2007, παράλληλα με την ανακοίνωση της ίδρυσης του οργανισμού Open Handset Alliance, μιας κοινοπραξίας 48 τηλεπικοινωνιακών εταιριών, εταιριών λογισμικού καθώς και κατασκευής hardware, οι οποίες είναι αφιερωμένες στην ανάπτυξη και εξέλιξη ανοιχτών προτύπων στις συσκευές

κινητής τηλεφωνίας. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα του Android υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού.

Χαρακτηριστικά

- **Λειτουργίες Οθόνης** : Η πλατφόρμα είναι προσαρμόσιμη σε μεγαλύτερη ανάλυση (VGA), δισδιάστατες ψηφιακές γραφικές βιβλιοθήκες, τρισδιάστατα γραφικά βασισμένα στην OpenGL ES 1.0 έκδοση χαρακτηριστικών, καθώς και παραδοσιακές απεικονίσεις οθόνης "έξυπνων" συσκευών κινητής τηλεφωνίας.
- **Αποθήκευση Δεδομένων** : Χρήση βάσης δεδομένων SQLite για τις ανάγκες αποθήκευσης.
- **Συνδεσιμότητα** : Το Android υποστηρίζει τεχνολογίες συνδεσιμότητας συμπεριλαμβανομένου GSM / EDGE, CDMA, EV-DO, UMTS, Bluetooth, και Wi-Fi.
- **Αποστολή μηνυμάτων** : SMS και MMS είναι οι διαθέσιμοι τρόποι ανταλλαγής μηνυμάτων.
- **Περιήγηση στον Ιστό** : Για την περιήγηση στον ιστό το Android διαθέτει φυλλομετρητή βασισμένο στην ανοιχτή τεχνολογία WebKit.
- **Υποστήριξη Java** : Λογισμικό γραμμένο στην Java είναι δυνατόν να μεταγλωττιστεί και να εκτελεστεί στην εικονική μηχανή Dalvik, η οποία αποτελεί εξειδικευμένη υλοποίηση εικονικής μηχανής, σχεδιασμένης για χρήση σε φορητές συσκευές, παρόλο που δεν είναι πρότυπη εικονική μηχανή Java.
- **Υποστήριξη Πολυμέσων** : Το λειτουργικό Android υποστηρίζει τις ακόλουθες μορφές ήχου, στατικής και κινούμενης εικόνας: H.263, H.264 (σε 3GP ή MP4 container), MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, MP3, MIDI, OGG Vorbis, WAV, JPEG, PNG, GIF, BMP.

- **Επιπλέον υποστήριξη υλικού** : Το λειτουργικό Android μπορεί να συνεργαστεί με κάμερες στατικής ή κινούμενης εικόνας, οθόνες αφής, GPS, αισθητήρες επιτάχυνσης, μαγνητόμετρα, δισδιάστατους καθώς και τρισδιάστατους επιταχυντές γραφικών.
- **Περιβάλλον Ανάπτυξης Λογισμικού** : Περιλαμβάνει ένας προσομοιωτή συσκευής, εργαλεία για διόρθωση σφαλμάτων, μνήμη και εργαλεία ανάλυσης της απόδοσης του εκτελέσιμου λογισμικού καθώς και ένα επιπρόσθετο για το Eclipse IDE.
- **Αγορά και Εγκατάσταση Εφαρμογών** : Παρόμοια με το App Store του iPhone OS, το Android Market είναι ένας κατάλογος εφαρμογών που μπορούν να μεταφορτωθούν και εγκατασταθούν στην συσκευή άμεσα μέσω ασύρματων καναλιών, χωρίς την χρήση υπολογιστή. Αρχικά μόνο δωρεάν εφαρμογές ήταν δυνατόν να εγκατασταθούν. Εφαρμογές επί πληρωμή ήταν μετέπειτα διαθέσιμες στο Android Market στις ΗΠΑ ύστερα από τις 19 Φεβρουαρίου 2009.
- **Οθόνη Αφής Πολλαπλών Σημείων** : Το λειτουργικό Android είχε εξ ορισμού υποστήριξη για οθόνες πολλαπλών σημείων.

ΓΛΩΣΣΑ SQL

Η **SQL** (από το Structured Query Language) [2] είναι μία γλώσσα υπολογιστών στις βάσεις δεδομένων, που σχεδιάστηκε για τη διαχείριση δεδομένων, σε ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων και η οποία αρχικά, βασίστηκε στη σχεσιακή άλγεβρα. Η γλώσσα περιλαμβάνει δυνατότητες ανάκτησης και ενημέρωσης δεδομένων, δημιουργίας και τροποποίησης σχημάτων και σχεσιακών πινάκων, αλλά και ελέγχου πρόσβασης στα δεδομένα. Η SQL ήταν μία από τις πρώτες γλώσσες για το σχεσιακό μοντέλο του Edgar F. Codd, στο σημαντικό άρθρο του το 1970, και έγινε η πιο ευρέως χρησιμοποιούμενη γλώσσα για τις σχεσιακές βάσεις δεδομένων.

Η γλώσσα SQL υποδιαιρείται σε διάφορα γλωσσικά στοιχεία, που περιλαμβάνουν :

- **Clauses**, οι οποίες είναι σε μερικές περιπτώσεις προαιρετικές, αλλά απαραίτητα συστατικά των δηλώσεων και ερωτήσεων.
- **Expressions** που μπορούν να παραγάγουν είτε τις κλιμακωτές τιμές είτε πίνακες που αποτελούνται από στήλες και σειρές στοιχείων.
- **Predicates** που διευκρινίζουν τους όρους που μπορούν να αξιολογηθούν σαν σωστό ή λάθος.
- **Queries** που ανακτούν τα στοιχεία βασισμένες σε ειδικά κριτήρια.
- **Statements** που μπορούν να έχουν μια επίδραση στα σχήματα και τα στοιχεία, ή που μπορούν να ελέγξουν τη ροή του προγράμματος και τις συνδέσεις από άλλα προγράμματα.
- Το κενό αγνοείται γενικά στις Statements και τις Queries SQL. Ένα κενό είναι όμως απαραίτητο για να ξεχωρίζει Statements Όπως και στην κανονική γραφή κειμένων.

Οι πιο σημαντικές SQL εντολές [10] είναι :

- **SELECT** : εξαγωγή δεδομένων από τη βάση
- **UPDATE** : ενημέρωση δεδομένων της βάσης
- **DELETE** : διαγραφή δεδομένων από τη βάση
- **INSERT INTO** : εισαγωγή δεδομένων στη βάση
- **CREATE DATABASE** : δημιουργία βάσης
- **ALTER DATABASE** : τροποποίηση της βάσης
- **CREATE TABLE** : δημιουργία πίνακα
- **ALTER TABLE** : τροποποίηση του πίνακα

- **DROP TABLE** : διαγραφή πίνακα
- **CREATE INDEX** : δημιουργία ευρετηρίου
- **DROP INDEX** : διαγραφή ευρετηρίου

Παράδειγμα

Έστω ο παρακάτω πίνακας με όνομα : Person

id	name	age
1	Panos	24
2	Spiros	20
3	Maria	32
4	Eleni	30

Θέλουμε να βρούμε τα ονόματα όλων των ατόμων που είναι μικρότεροι από 25 χρονών.

SQL Εντολή: **SELECT** name **FROM** Person **WHERE** age < 25;

Θα μας επιστρέψει : Panos, Spiros

ΒΑΣΗ ΔΕΔΟΜΕΝΩΝ MySQL

Η **MySQL** [6] είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων. Τρέχει έναν εξυπηρετητή (server) παρέχοντας πρόσβαση πολλών χρηστών σε ένα σύνολο βάσεων δεδομένων. Η MySQL είναι δημοφιλής βάση δεδομένων για διαδικτυακά προγράμματα και

ιστοσελίδες και χρησιμοποιείται σε κάποιες από τις πιο διαδεδομένες διαδικτυακές υπηρεσίες.

Συμβατότητα

Η MySQL μπορεί να τρέχει σε διάφορα λειτουργικά συστήματα και πλατφόρμες όπως, AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, OS X, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos και Tru64.

Είναι γραμμένη σε C και σε C++. Ο SQL αναλυτής που περιέχει είναι γραμμένος σε yacc. Πολλές γλώσσες προγραμματισμού περιέχουν δικές τους ειδικές βιβλιοθήκες για επικοινωνία και είσοδο σε MySQL βάσεις δεδομένων. Περιέχουν ειδικούς οδηγούς όπως ο JDBC της Java.

Χαρακτηριστικά

Μερικά από τα χαρακτηριστικά [14] της MySQL 5.6 είναι :

- Υποστήριξη πολλαπλών λειτουργικών
- Υποστήριξη SSL
- Triggers
- Cursors
- Information schema

- Performance Schema
- Query caching
- Sub-SELECTs
- Υποστήριξη Unicode
- Πολλαπλές μηχανές αποθήκευσης, επιτρέποντας έτσι να επιλέξετε αυτή που είναι πιο αποτελεσματική για κάθε πίνακα στην εφαρμογή
- Ομαδοποίηση και συγκέντρωση πολλαπλών συναλλαγών από πολλαπλές συνδέσεις για να αυξηθεί η απόδοση του συστήματος
- Ενσωματωμένη βιβλιοθήκη βάσεων δεδομένων
- Ευρετήριο πλήρους κειμένου και αναζήτηση

ΓΛΩΣΣΑ PHP

Η **PHP** είναι μια γλώσσα σεναρίου [3] που τρέχει στη πλευρά του διακομιστή (server-side scripting language). Σχεδιάστηκε για προγραμματισμό στον ιστό αλλά χρησιμοποιείται και ως γλώσσα προγραμματισμού γενικής χρήσης. Ο κώδικας γραμμένος σε PHP μπορεί εύκολα να αναμιχθεί με HTML κώδικα ή και να συνδυαστεί με templating engines και web frameworks. Συνήθως επεξεργάζεται από έναν διερμηνέα PHP ο οποίος παρέχεται ως τοπικό πρόγραμμα (native module) εγκατεστημένο στον web server ή ως ένα CGI (Common Gateway Interface) εκτελέσιμο. Μετά ο PHP κώδικας διερμηνεύεται και εκτελείται, ο web server στέλνει την έξοδο που προκύπτει για τον πελάτη του (client), συνήθως με τη μορφή ενός μέρους της παραγόμενης ιστοσελίδας. Για παράδειγμα, ο κώδικας PHP μπορεί να δημιουργήσει κώδικα HTML μιας ιστοσελίδας, μια εικόνα, ή κάποια άλλα στοιχεία. Η PHP έχει αναπτυχθεί για να περιλαμβάνει επίσης και μια διεπαφή γραμμής εντολών για να μπορεί να χρησιμοποιηθεί σε αυτόνομες εφαρμογές. Μπορεί να εκτελεστεί στους περισσότερους web servers, σε πολλά λειτουργικά συστήματα και πλατφόρμες και μπορεί να χρησιμοποιηθεί με πολλά συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS).

Οι περισσότεροι web hosting πάροχοι υποστηρίζουν PHP για χρήση από τους πελάτες τους. Είναι διαθέσιμη δωρεάν, και ο Όμιλος PHP παρέχει τον πλήρη πηγαίο κώδικα στους χρήστες για να την χτίσουν, να την προσαρμόσουν και να την επεκτείνουν για δική τους χρήση. Η PHP δρα ως φίλτρο, λαμβάνοντας είσοδο από ένα αρχείο ή μια ροή που περιέχει κείμενο ή/και εντολές PHP και εξάγει μια άλλη ροή δεδομένων. Πιο συχνά η έξοδος είναι σε μορφή HTML, αλλά μπορεί να είναι και σε JSON, XML, ή και δυαδική μορφή όπως και σε μορφή εικόνας ή ήχου [11].

Συντακτικό

```
<?php
     εντολή 1
     εντολή 2
     ....
?>
```

Μπορεί να ενσωματωθεί εύκολα μέσα σε HTML κώδικα, για παράδειγμα :

```
<!DOCTYPE html>
<html>
     <head>
               <title>PHP Test</title>
     </head>
     <body>
```

```
        <?php echo '<p>Hello World</p>'; ?>
    </body>
</html>
```

ΕΠΕΚΤΑΣΗ MySQLi

Η **MySQLi** (MySQL Improved) [7] είναι μια επέκταση της MySQL. Είναι ένας οδηγός (driver) που χρησιμοποιείται στη γλώσσα προγραμματισμού PHP για να παρέχει μια διασύνδεση με βάσεις δεδομένων MySQL.

Υπάρχουν τρία κύρια API για να συνδεθούμε σε μια MySQL [12] βάση δεδομένων :

- Η MySQL επέκταση της PHP
- Η MySQLi επέκταση της PHP
- Τα αντικείμενα δεδομένων της PHP (PDO)

Η MySQLi ως μια βελτιωμένη έκδοση του παλιότερου PHP MySQL οδηγού, προσφέρει ποικίλα προνόμια.

Τεχνικά Χαρακτηριστικά

Τα πιο σημαντικά προνόμια που προσφέρει αυτή η επέκταση είναι :

- Αντικειμενοστραφής διεπαφή

- Υποστήριξη για τις προπαρασκευασμένες δηλώσεις (statements)
- Υποστήριξη για πολλαπλές δηλώσεις (statements)
- Υποστήριξη για συναλλαγές (transactions)
- Ενισχυμένη υποστήριξη αποσφαλμάτωσης
- Ενσωματωμένη υποστήριξη διακομιστή
- Πιο ισχυρή λειτουργικότητα

Σημαντικές Εντολές [13]

- mysqli_connect() : Ανοίγει καινούργια σύνδεση στο MySQL Server.
- mysqli_close () : Κλείνει την προηγούμενη σύνδεση.
- mysqli_connect_errno () : Επιστρέφει ένα κωδικό λανθασμένης σύνδεσης.
- mysqli_query () : Στέλνει ένα ερώτημα στη βάση δεδομένων.
- mysqli_select_db () : Επιλέγει βάση δεδομένων.
- mysqli_set_charset () : Αλλάζει τη κωδικοποίηση των χαρακτήρων της βάσης.

WEB SERVICE

Web Service [4] είναι μια μέθοδος επικοινωνίας μεταξύ δύο ηλεκτρονικών συσκευών σε ένα δίκτυο. Είναι μια λειτουργία του λογισμικού παρέχεται σε μια διεύθυνση δικτύου μέσω του web με την υπηρεσία πάντα ενεργοποιημένη. Το W3C (World Wide Web Consortium) ορίζει το web service γενικά ως ένα σύστημα λογισμικού σχεδιασμένο να υποστηρίζει διαλειτουργική αλληλεπίδραση μηχανής προς μηχανή σε ένα δίκτυο.

Πολλοί οργανισμοί χρησιμοποιούν πολλαπλά συστήματα λογισμικού για διαχείριση. Διαφορετικά συστήματα λογισμικού συχνά πρέπει να ανταλλάσσουν δεδομένα μεταξύ τους. Το σύστημα λογισμικού που ζητάει στοιχεία ονομάζεται αιτούντα υπηρεσία (service requester), ενώ το σύστημα λογισμικού που θα επεξεργαστεί το αίτημα και θα παρέχει τα δεδομένα καλείται πάροχος υπηρεσιών (service provider). Διαφορετικό λογισμικό θα μπορούσε να κατασκευαστεί με διαφορετικές γλώσσες προγραμματισμού και ως εκ τούτου υπάρχει ανάγκη για μία μέθοδο ανταλλαγής δεδομένων που δεν εξαρτάται από μία συγκεκριμένη γλώσσα προγραμματισμού. Οι περισσότεροι τύποι λογισμικού μπορούν ωστόσο, να ερμηνεύσουν ετικέτες XML. Έτσι τα web services μπορούν να χρησιμοποιούν αρχεία XML για την ανταλλαγή δεδομένων.

Υπάρχουν κανόνες που πρέπει να καθοριστούν για την επικοινωνία μεταξύ των διαφορετικών συστημάτων όπως :

- Πως ένα σύστημα μπορεί να ζητήσει τα δεδομένα από ένα άλλο σύστημα.
- Ποιες είναι οι συγκεκριμένες παράμετροι που απαιτούνται στην αίτηση στοιχείων.
- Ποια θα είναι η δομή των δεδομένων που παράγονται.
- Τι μηνύματα λάθους θα εμφανιστούν όταν ένας συγκεκριμένος κανόνας για την επικοινωνία δε τηρείται, για να γίνει ευκολότερη η αντιμετώπιση προβλημάτων.

Όλοι αυτοί οι κανόνες επικοινωνίας ορίζονται σε ένα αρχείο που ονομάζεται WSDL (Web Services Description Language).

JSON FORMAT

Το **JSON** (JavaScript Object Notation) [5] είναι ένα ανοιχτό πρότυπο format που χρησιμοποιεί αναγνωρίσιμο από τον άνθρωπο κείμενο για να μεταδώσει αντικείμενα

δεδομένων που αποτελούνται από ζεύγη χαρακτηριστικών-τιμών. Χρησιμοποιείται κυρίως για τη μετάδοση δεδομένων μεταξύ server και εφαρμογής. Παρά το γεγονός ότι προέρχεται αρχικά από τη γλώσσα JavaScript, είναι μια μορφή δεδομένων ανεξάρτητη από τη γλώσσα. Ο κώδικας για την ανάλυση και την παραγωγή δεδομένων JSON είναι εύκολα διαθέσιμος σε πολλές γλώσσες προγραμματισμού.

Βασικοί τύποι δεδομένων

- **Αριθμός** (number) : δεκαδικός αριθμός που μπορεί να περιέχει και κλασματικό μέρος. Δεν επιτρέπονται μη-αριθμοί όπως το NaN.
- **Συμβολοσειρά** (string) : μια ακολουθία από μηδέν ή περισσότερους unicode χαρακτήρες. Η συμβολοσειρά οριοθετείται με διπλά εισαγωγικά.
- **Τύπος δεδομένων αλήθειας** (boolean) : μπορεί να πάρει μόνο δύο τιμές, αλήθεια (true) και ψέμα (false).
- **Πίνακας** (Array) : μια ταξινομημένη λίστα από μηδέν ή περισσότερες τιμές, καθεμία από τις οποίες μπορεί να είναι οποιουδήποτε τύπου. Οι πίνακες χρησιμοποιούν αγκύλες και τα στοιχεία χωρίζονται μεταξύ τους με κόμμα.
- **Αντικείμενο** (Object) : μια μη διατεταγμένη συλλογή από ζεύγη δεδομένων (όνομα-τιμή). Το αντικείμενο οριοθετείται με αγκύλες και τα ζεύγη χωρίζονται μεταξύ τους με κόμμα.
- **null** : κενή τιμή

Παράδειγμα


```
{
    "firstName": "Panos",
    "lastName": "Papadopoulos",
    "isAlive": true,
    "age": 24,
    "address": {
        "streetAddress": "Koila Kozanis",
        "city": "Kozani",
        "postalCode": "50100"
    }
}
```


REMOTE PROCEDURE CALL (RPC)

Στην επιστήμη των υπολογιστών **Remote Procedure Call** (**RPC**) [8] είναι μια δραστηριότητα διαμοιρασμού δεδομένων μεταξύ πολλαπλών και συχνά εξειδικευμένων διεργασιών που χρησιμοποιούν πρωτόκολλα επικοινωνίας. Επιτρέπει σε ένα πρόγραμμα να δημιουργήσει μια υπορουτίνα ή διαδικασία για να εκτελέσει μια ενέργεια σε ένα άλλο υπολογιστή ή σε ένα δίκτυο χωρίς να έχει γράψει ο προγραμματιστής ρητά τις λεπτομέρειες της απομακρυσμένης αυτής αλληλεπίδρασης. Δηλαδή ο προγραμματιστής γράφει ουσιαστικά τον ίδιο κώδικα είτε αν η υπορουτίνα τρέχει τοπικά, είτε απομακρυσμένα. Όταν το εν λόγω λογισμικό χρησιμοποιεί αντικειμενοστραφείς αρχές ονομάζεται **Remote Invocation** ή

Remote Method Invocation (**RMI**). Έχουν υλοποιηθεί πολλές, διαφορετικές και συχνά ασυμβίβαστες τεχνολογίες βασισμένες σε αυτή τη γενική ιδέα.

Το RPC αρχικοποιείται από ένα πελάτη-αποστολέα (client), ο οποίος στέλνει μια αίτηση-κλήση σε ένα γνωστό απομακρυσμένο server για την εκτέλεση μιας συγκεκριμένης διεργασίας με παρεχόμενες παραμέτρους. Ο server στέλνει μια απάντηση στην αίτηση του client και η εφαρμογή συνεχίζει την εργασία της. Όσο ο server επεξεργάζεται την αίτηση, ο client είναι αποκλεισμένος και περιμένει μέχρις ότου ο server ολοκληρώσει την επεξεργασία πριν συνεχίσει την εκτέλεση του, εκτός και αν ο client έχει στείλει μία ασύγχρονη αίτηση στο server, όπως μία XHTTP κλήση. Υπάρχουν πολλές παραλλαγές και διάφορες υλοποιήσεις, με αποτέλεσμα μία ποικιλία διαφορετικών (ασύμβατων) πρωτοκόλλων RPC.

Μια σημαντική διαφορά μεταξύ των κλήσεων απομακρυσμένης διαδικασίας και των τοπικών κλήσεων είναι ότι οι απομακρυσμένες κλήσεις μπορεί να αποτύχουν εξαιτίας απρόβλεπτων προβλημάτων του δικτύου. Επίσης, οι clients γενικά πρέπει να ασχοληθούν με τέτοιου είδους παραλείψεις, χωρίς να γνωρίζουν εάν στην απομακρυσμένη διαδικασία όντως προκλήθηκαν προβλήματα. Διεργασίες οι οποίες δεν έχουν πρόσθετα αποτελέσματα εάν κληθούν περισσότερες από μία φορές αντιμετωπίζονται σχετικά εύκολα, αλλά υπάρχουν αρκετές δυσκολίες στη συγγραφή αποδοτικού κώδικα σε διεργασίες που διαχειρίζονται υποσυστήματα χαμηλού επιπέδου.

Ακολουθία γεγονότων κατά τη διάρκεια μιας RPC

1. Η κλήση από τον client ονομάζεται client stub. Είναι κλήση μιας τοπικής διεργασίας με παραμέτρους που αποθηκεύονται σε μία στοίβα με κανονικό τρόπο.
2. Το client stub δημιουργεί ένα πακέτο με τις παραμέτρους ως ένα μήνυμα και να κάνει μια κλήση συστήματος για να στείλει το μήνυμα. Η δημιουργία του πακέτου των παραμέτρων ονομάζεται marshalling.

3. Το τοπικό λειτουργικό σύστημα του client στέλνει το μήνυμα από το μηχάνημα του client στο μηχάνημα του server.
4. Το τοπικό λειτουργικό σύστημα στο μηχάνημα του server περνάει τα εισερχόμενα πακέτα στο αντίστοιχο server stub.
5. Το server stub εξαγάγει τις παραμέτρους από το μήνυμα. Αυτή η διαδικασία ονομάζεται unmarshalling.
6. Τέλος το server stub καλεί τη διαδικασία του server που θα στείλει την απάντηση στον client.

JAVA REMOTE METHOD INVOCATION (JAVA RMI)

Το **Java Remote Method Invocation** (**Java RMI**) [9] είναι ένα Java API και αντιστοιχεί στο αντικειμενοστραφές ισοδύναμο του **Remote Procedure Call** (**RPC**) με υποστήριξη Java κλάσεων και πρωτοκόλλου DGC - Distributed Garbage Collection.

Η αρχική υλοποίηση εξαρτιόταν από την εικονική μηχανή της Java (JVM) και τους μηχανισμούς της και επιπλέον υποστήριζε κλήσεις μόνο σε άλλη εικονική μηχανή της Java. Η υλοποίηση αυτή ήταν γνωστή ως Java Remote Method Protocol (JRMP). Για να υποστηρίξει κώδικα ο οποίος δε τρέχει μόνο σε JVM, αναπτύχθηκε αργότερα και μια CORBA έκδοση της.

Η χρήση του όρου RMI μπορεί να αναφέρεται μόνο στη διεπαφή προγραμματισμού ή μπορεί να αναφέρεται τόσο στο API, JRMP, IIOP, ή άλλη εφαρμογή. Ενώ ο όρος RMI-IIOP δηλώνει ρητά τη διεπαφή RMI που υποστηρίζει τις λειτουργίες της υλοποίησης CORBA.

Η βασική ιδέα του Java RMI, του DGC πρωτοκόλλου, καθώς και μεγάλο μέρος της αρχιτεκτονικής υπόκειται στην αρχική εφαρμογή της Sun και προέρχεται από το χαρακτηριστικό των "αντικειμένων δικτύου" της γλώσσας προγραμματισμού Modula-3.

ΑΛΛΑ ΣΥΝΑΦΗ ΘΕΜΑΤΑ

Υπάρχουν πολλές εφαρμογές κινητής τηλεφωνίας που επιτρέπουν τη συνεργασία και έχουν ποικίλη πολυπλοκότητα και σκοπό. Το Anywhere Pad [21] και το Keep [28] επιτρέπουν τη συγγραφή σημειώσεων και τη κοινή χρήση τους. Στη διαχείριση απλών καθημερινών εργασιών στοχεύουν τα Any.do [20] και Quip [36]. Το CAL [23] ανήκει στην ίδια κατηγορία αλλά χρησιμοποιεί ημερολόγιο ως κύρια διεπαφή. Τέλος, το IF [29] είναι μία εφαρμογή που επιτρέπει στους χρήστες να δομούν κανόνες για τη διαχείριση γεγονότων όπως η κοινή χρήση αρχείων με απλές συνθήκες if-then, διευκολύνοντας έτσι το φορτίο του χρήστη.

Ο προγραμματισμός συνεδρίων και συναντήσεων επίσης απαιτούν συνεργασία. Οι εφαρμογές σε αυτή τη κατηγορία ποικίλλουν από προγραμματιστές συναντήσεων με δυνατότητα κοινής χρήσης αρχείων και σημειώσεων όπως το join.me [32] μέχρι και βίντεο όπως τα Studiopass [39] και WebEx [41]. Επιπλέον οι εφαρμογές κινητών συσκευών έχουν διεισδύσει σε όλο το εύρος ενός παραδοσιακού συστήματος ERP, με τα Insight [30] και Streamwork [37] να είναι δύο σημαντικά παραδείγματα. Τέλος, μια ενδιαφέρουσα εφαρμογή που χρειάζεται συνεργασία των κινητών συσκευών είναι το πρόβλημα ελαττωματικής ανίχνευσης που μελετήθηκε, για παράδειγμα, στο [19].

Για να μπορούν να λειτουργούν οι παραπάνω εφαρμογές είναι απαραίτητος ο συγχρονισμός μεταξύ των αρχείων που βρίσκονται στη συσκευή με αυτών στο σύννεφο (Cloud). Το Dropbox [25] και το GoogleDrive [27] είναι εμπορικά παραδείγματα συστημάτων αρχείων που επιτρέπουν τη κατανεμημένη κοινή χρήση τους. Τα προβλήματα συνοχής εγείρονται στο συγχρονισμό των αρχείων και έχουν προσελκύσει το ενδιαφέρον των ερευνητών. Το πρόβλημα συχνά εξετάζεται ως πρόβλημα κοινής συναίνεσης με ερευνητές άλλα εμπορικά προγράμματα χρησιμοποιούν το πρωτόκολλο Paxos [33]. Μια ενδιαφέρουσα συζήτηση για τη συνοχή παρέχεται στο [34].

Γενικά κατανεμημένα συστήματα αρχείων με εγγυήσεις για ανοχή στα λάθη είχαν προταθεί στο παρελθόν. Στο [31] οι συγγραφείς χρησιμοποιούν μια προσέγγιση με μηχανή καταστάσεων για την επίτευξη ανοχής σε σφάλματα σε δημοσίευσης/εγγραφής συστήματα υπό βυζαντινά λάθη. Το Wormhole [38] βελτιώνει το [31] υπό την έννοια ότι ακολουθεί μία προσέγγιση που δεν έχει καταστάσεις. Το Thialfi [18] είναι ένα παλαιότερο push based σύστημα ειδοποιήσεων με εγγυήσεις συνέπειας, ενώ το [22] είναι επίσης μία προηγούμενη δουλειά για δημοσίευσης/εγγραφής συστήματα που ερευνά την περίπτωση όπου ένας διακομιστής δεν είναι διαθέσιμος.

Εφαρμογές που στοχεύουν αποκλειστικά περιβάλλον κινητών συσκευών είναι το Mobius [24] και το Simba [26], [35]. Ο σκοπός του Mobius είναι να παρέχει συγχρονισμένες καταχωρήσεις πίνακα, ενώ το Simba προεκτείνει τη προαναφερθείσα ιδέα με ποικίλους τρόπους, κυρίως επιτρέποντας ισορροπία μεταξύ απαιτήσεων συνέπειας και απόδοσης κινητών εφαρμογών [35]. Το πρόβλημα του κατανεμημένου πίνακα κλειδώματος αντιμετωπίστηκε στο παρελθόν στο πλαίσιο των κινητών βάσεων δεδομένων π.χ [40].

ΚΕΦΑΛΑΙΟ 2

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Σε αυτό το κεφάλαιο θα περιγράψουμε την αρχιτεκτονική του συστήματος και τη λειτουργία της εφαρμογής. Θα αναλύσουμε τη βάση δεδομένων, τους πίνακες που περιέχει καθώς και τα web services μέσω των οποίων επικοινωνούν οι συσκευές με τη βάση και κατ' επέκταση μεταξύ τους. Θα δούμε επίσης την αρχιτεκτονική του module (τμήμα ενός προγράμματος) του ξυπνητηριού όπως και αυτή του χάρτη.

Έχουμε υλοποιήσει μια εφαρμογή η οποία δίνει τη δυνατότητα στο χρήστη να διαχειρίζεται απομακρυσμένα εφαρμογές άλλης συσκευής. Για να γίνει αυτό χρειαζόμαστε ένα server και ένα client. Ο server βρίσκεται στο διαδίκτυο ενώ ο client είναι η κάθε συσκευή όπου είναι εγκατεστημένη η εφαρμογή. Έχουμε λοιπόν ένα σύστημα που χρησιμοποιεί διάφορα modules προκειμένου να επικοινωνήσουν όλα τα επιμέρους τμήματά του μεταξύ τους. Αυτά τα modules βρίσκονται στον server και επίσης υπάρχουν και άλλα modules τα οποία εκτελούν τις ενέργειες του client.

Συνοπτικά τα modules μέσω των οποίων επιτυγχάνεται η επικοινωνία του server με τον client είναι δύο και είναι γραμμένα σε γλώσσα PHP. Το ένα είναι το **up_service.php** που επιτρέπει την αποστολή δεδομένων στο server και το άλλο είναι το **down_service.php** το οποίο επιτρέπει τη λήψη δεδομένων από το server.

Στην μεριά του client υπάρχουν modules γραμμένα σε Java μέσω των οποίων στέλνονται τα δεδομένα στο server και τα οποία χρησιμοποιεί ο χρήστης και θα τα αναλύσουμε μετά, αλλά το πιο σημαντικό είναι το module που τρέχει στο παρασκήνιο και το οποίο είναι υπεύθυνο για την λήψη και αποκωδικοποίηση των δεδομένων καθώς και για την πραγμάτωση των

ενεργειών που έχουν ανατεθεί στην εφαρμογή. Είναι η κλάση **ActionsReceiver** που έχει χαρακτηριστικά BroadcastReceiver και θα την αναλύσουμε στη συνέχεια.

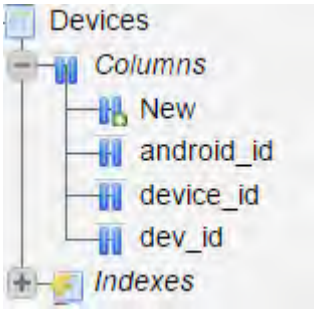
Η γενική ιδέα στην οποία βασίστηκε η αρχιτεκτονική του συστήματος είναι εξαιρετικά απλή και είναι η εξής. Έχουμε δύο συσκευές στις οποίες είναι εγκατεστημένη η εφαρμογή και μία βάση δεδομένων που βρίσκεται σε έναν web server (**ΣΧΗΜΑ 2.1**). Και οι δύο συσκευές λειτουργούν ταυτόχρονα και ως αποστολείς και ως δέκτες, δηλαδή μπορούν να στέλνουν δεδομένα στη βάση και λαμβάνουν δεδομένα από τη βάση.

ΣΧΗΜΑ 2.1 : ΕΠΙΚΟΙΝΩΝΙΑ ΜΕΤΑΞΥ ΣΥΣΚΕΥΩΝ

Να τονίσουμε ότι οι συσκευές δεν επικοινωνούν απευθείας μεταξύ τους, αλλά μέσω του server.

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΒΑΣΗΣ

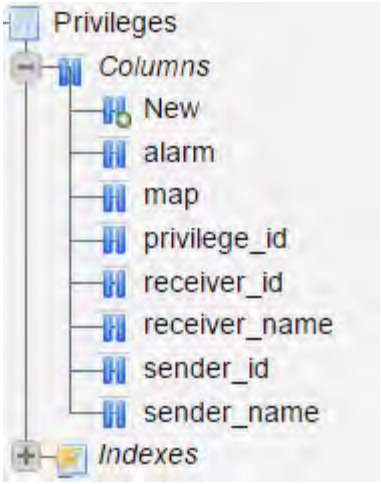
Πάμε να δούμε πρωτίστως τι περιέχει η βάση δεδομένων. Στη βάση λοιπόν υπάρχουν πέντε πίνακες. Ο πρώτος πίνακας στον οποίο θα αναφερθούμε (**ΣΧΗΜΑ 2.2**) ονομάζεται **Devices** και σ αυτόν αποθηκεύονται τα αναγνωριστικά της κάθε συσκευής.



ΣΧΗΜΑ 2.2 : ΠΙΝΑΚΑΣ DEVICES ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Περιέχει τρεις στήλες. Η πρώτη ονομάζεται **android_id** όπου εκεί αποθηκεύεται το αναγνωριστικό που έχει δώσει η εταιρία παραγωγής στη συσκευή και είναι μοναδικό για κάθε συσκευή και η δεύτερη ονομάζεται **device_id** και εκεί αποθηκεύεται το αναγνωριστικό που έχει δημιουργήσει η εφαρμογή για τη συσκευή και είναι και αυτό που γνωστοποιείται στο χρήστη. Αυτή η διαφοροποίηση έχει γίνει για λόγους ασφαλείας. Το **android_id** είναι ένα αναγνωριστικό που πραγματικά χαρακτηρίζει την συσκευή. Δηλαδή μπορεί να χρησιμοποιηθεί για παράδειγμα για να να εντοπίσει που βρίσκεται αυτή η συσκευή, ενώ το **device_id** είναι ένα αναγνωριστικό που ναι μεν διαχωρίζει μοναδικά την κάθε συσκευή, αλλά η χρησιμότητά του είναι μόνο εντός της εμβέλειας διαχείρισης της εφαρμογής. Δεν υφίσταται εκτός αυτής και δεν έχει καμία άλλη χρησιμότητα. Τέλος το **dev_id** περιέχει το αναγνωριστικό της κάθε εγγραφής του πίνακα. Είναι ένας αριθμός που χρησιμοποιείται για να μπορούμε να διαχωρίζουμε μοναδικά την κάθε εγγραφή. Ο αριθμός αυτός δημιουργείται αυτόματα από τη βάση κάθε φορά που προστίθεται μία νέα γραμμή στον πίνακα.

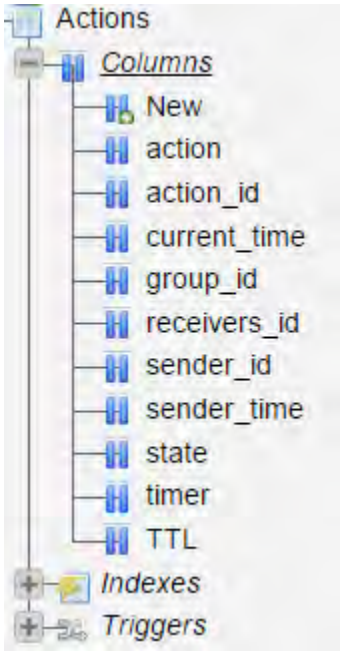
Στον επόμενο πίνακα (**ΣΧΗΜΑ 2.3**) που ονομάζεται **Privileges** αποθηκεύονται τα δικαιώματα μεταξύ δύο συσκευών.



ΣΧΗΜΑ 2.3 : ΠΙΝΑΚΑΣ PRIVILEGES ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

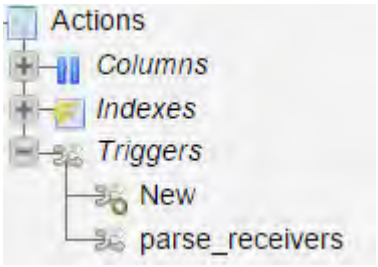
Περιέχει εφτά στήλες. Η πρώτη ονομάζεται **alarm** και αναφέρεται στο δικαίωμα του αποστολέα αναφορικά με τη διαχείριση της εφαρμογής του ξυπνητηριού του δέκτη. Μπορεί να πάρει δύο τιμές οι οποίες είναι το ‘1’ και το ‘0’. Η δεύτερη ονομάζεται **map** και αναφέρεται στο δικαίωμα του αποστολέα αναφορικά με τη διαχείριση της εφαρμογής του χάρτη του δέκτη. Μπορεί να πάρει δύο τιμές οι οποίες είναι το ‘1’ και το ‘0’. Η τρίτη ονομάζεται **privilege_id** περιέχει το αναγνωριστικό της κάθε εγγραφής του πίνακα. Είναι ένας αριθμός που χρησιμοποιείται για να μπορούμε να διαχωρίζουμε μοναδικά την κάθε εγγραφή. Ο αριθμός αυτός δημιουργείται αυτόματα από τη βάση κάθε φορά που προστίθεται μία νέα γραμμή στον πίνακα. Η τέταρτη **receiver_id** και εκεί αποθηκεύεται το **device_id** του δέκτη. Η πέμπτη ονομάζεται **receiver_name** και εκεί αποθηκεύεται το όνομα που θα δώσει ο αποστολέας μέσω της εφαρμογής, στη συσκευή του δέκτη που διαχειρίζεται. Η έκτη ονομάζεται **sender_id** και εκεί αποθηκεύεται το **device_id** του αποστολέα. Και τέλος η έβδομη ονομάζεται **sender_name** και εκεί αποθηκεύεται το όνομα που θα δώσει ο δέκτης μέσω της εφαρμογής, στη συσκευή του αποστολέα την οποία επιτρέπει να διαχειρίζεται της εφαρμογές του.

Ο επόμενος πίνακας (**ΣΧΗΜΑ 2.4**) ονομάζεται **Actions**. Σε αυτόν αποθηκεύονται όλα τα δεδομένα που είναι σχετικά με μία ενέργεια που έχει σταλεί από κάποιο χρήστη και περιέχει δέκα στήλες και ένα trigger ο οποίος ενεργοποιείται όποτε γίνεται εισαγωγή νέας γραμμής.



ΣΧΗΜΑ 2.4 : ΠΙΝΑΚΑΣ ACTIONS ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Η πρώτη ονομάζεται **action** και περιέχει την εντολή που θα εκτελέσει η συσκευή στην οποία αντιστοιχεί ο πίνακας. Η εντολή αυτή είναι μια ακολουθία συμβόλων που και αποτελείται από επιμέρους τμήματα που χωρίζονται από τον χαρακτήρα **'-'**. Η δεύτερη ονομάζεται **action_id** περιέχει το αναγνωριστικό της κάθε εγγραφής του πίνακα. Είναι ένας αριθμός που χρησιμοποιείται για να μπορούμε να διαχωρίζουμε μοναδικά την κάθε εγγραφή. Ο αριθμός αυτός δημιουργείται αυτόματα από τη βάση κάθε φορά που προστίθεται μία νέα γραμμή στον πίνακα. Η τρίτη ονομάζεται **current_time** και περιέχει την χρονοσφραγίδα στην οποία έγινε η εισαγωγή της νέας ενέργειας. Η τέταρτη ονομάζεται **group_id** και περιέχει ένα μοναδικό αναγνωριστικό για μία ομάδα χρηστών. Η πέμπτη ονομάζεται **receivers_id** και περιέχει είτε μόνο ένα αναγνωριστικό που ανήκει στον παραλήπτη, είτε ένα σύνολο από αναγνωριστικά παραληπτών στη περίπτωση που ο αποστολέας θέλει να στείλει ταυτόχρονα την ίδια ενέργεια σε πολλούς χρήστες.



ΣΧΗΜΑ 2.5 : TRIGGER ΤΟΥ ΠΙΝΑΚΑ ACTIONS ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Χρησιμοποιώντας το πεδίο της στήλης αυτής μπορούμε να διαχωρίσουμε μέσω ενός trigger **(ΣΧΗΜΑ 2.5)** που ονομάζεται **parse_receivers** τα αναγνωριστικά των παραληπτών και αποθηκεύσουμε στον πίνακα ActionsTargets (που θα αναλύσουμε παρακάτω) τις ενέργειες που αντιστοιχούν στον κάθε χρήστη. Παραθέτουμε παρακάτω το κώδικα που εκτελεί ο trigger.

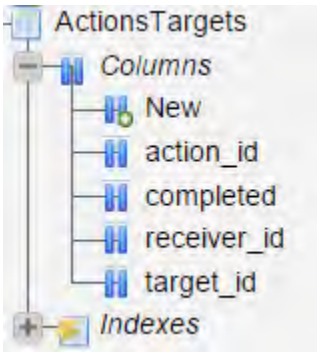
```
BEGIN

    DECLARE aid INT UNSIGNED;

    DECLARE str VARCHAR(50000);

    DECLARE size INT UNSIGNED;

    DECLARE counter INT UNSIGNED;

    DECLARE len INT UNSIGNED;


    SELECT action_id FROM Actions ORDER BY action_id

    DESC LIMIT 1 INTO aid ;

    SELECT receivers_id FROM Actions ORDER BY action_id

    DESC LIMIT 1 INTO str ;


    SET size=SUBSTRING_INDEX(str, ',', 1);
```

```
    SET counter=0;
    SET len=LENGTH(SUBSTRING_INDEX(str, ',', 1))+2;

    WHILE counter < size DO
        SET str=SUBSTRING(str,len);

        INSERT INTO ActionsTargets (action_id, receiver_id)
        VALUES (aid, SUBSTRING_INDEX(str, ',', 1));

        SET counter=counter+1;
        SET len=LENGTH(SUBSTRING_INDEX(str, ',', 1))+2;
    END WHILE;
END
```

Η έκτη ονομάζεται **sender_id** και περιέχει το **device_id** του αποστολέα. Η έβδομη ονομάζεται **sender_time** και περιέχει την χρονοσφραγίδα στην οποία στάλθηκε η ενέργεια. Η όγδοη ονομάζεται **state**. Μπορεί να πάρει δύο τιμές, το ‘1’ και το ‘0’, οι οποίες αντιστοιχούν στο αν έχει ή δεν έχει εκτελεστεί η ενέργεια. Η ένατη ονομάζεται **timer**. Μπορεί να πάρει δύο τιμές, το ‘1’ και το ‘0’, οι οποίες αντιστοιχούν στο αν η ενέργεια έχει ή δεν έχει ημερομηνία ακύρωσης. Και τέλος η δέκατη ονομάζεται **TTL** και περιέχει τον χρόνο (αν υπάρχει, timer=1) στον οποίο ακυρώνεται η ενέργεια.

Ο επόμενος πίνακας (**ΣΧΗΜΑ 2.6**) ονομάζεται **ActionsTargets** και περιέχει 4 στήλες. Η εισαγωγή δεδομένων γίνεται αυτόματα από τον trigger που περιγράψαμε προηγουμένως κάθε

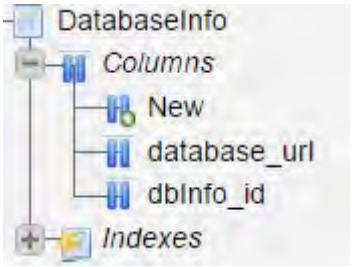
φορά που γίνεται εισαγωγή νέων εγγραφών στον πίνακα Actions. Πάμε να δούμε αναλυτικά τις στήλες.



ΣΧΗΜΑ 2.6 : ΠΙΝΑΚΑΣ ACTIONSTARGETS ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Η πρώτη ονομάζεται **action_id** και περιέχει το αναγνωριστικό της κάθε ενέργειας που έχει σταλεί για λήψη. Η δεύτερη ονομάζεται **completed**. Μπορεί να πάρει δύο τιμές, το '1' και το '0', οι οποίες αντιστοιχούν στο αν έχει ή δεν έχει εκτελεστεί η ενέργεια. Η τρίτη ονομάζεται **receivers_id** και περιέχει το αναγνωριστικό που ανήκει στον παραλήπτη. Η τέταρτη ονομάζεται **target_id** και περιέχει το αναγνωριστικό της κάθε εγγραφής του πίνακα. Είναι ένας αριθμός που χρησιμοποιείται για να μπορούμε να διαχωρίζουμε μοναδικά την κάθε εγγραφή. Ο αριθμός αυτός δημιουργείται αυτόματα από τη βάση κάθε φορά που προστίθεται μία νέα γραμμή στον πίνακα.

Ο τελευταίος πίνακας (**ΣΧΗΜΑ 2.7**) ονομάζεται **DatabaseInfo** και εδώ αποθηκεύονται κάποια στοιχεία σχετικά με τη βάση. Περιέχει δύο στήλες.



ΣΧΗΜΑ 2.7 : ΠΙΝΑΚΑΣ DATABASEINFO ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ

Η πρώτη ονομάζεται **database_url** και περιέχει το url της βάσης δεδομένων. Χρησιμοποιείται για να μπορεί να γίνεται εύκολα η μεταφορά της βάσης σε άλλο server. Τέλος η δεύτερη ονομάζεται **dbInfo_id** και περιέχει το αναγνωριστικό της κάθε εγγραφής του πίνακα. Είναι ένας αριθμός που χρησιμοποιείται για να μπορούμε να διαχωρίζουμε μοναδικά την κάθε εγγραφή. Ο αριθμός αυτός δημιουργείται αυτόματα από τη βάση κάθε φορά που προστίθεται μία νέα γραμμή στον πίνακα.

Η επικοινωνία της κάθε συσκευής με τη βάση δεδομένων επιτυγχάνεται μέσω δύο **web services** που βρίσκονται στον web server. Είναι γραμμένα σε γλώσσα PHP και χρησιμοποιούνται για να μπορούν οι χρήστες να στέλνουν δεδομένα στη βάση και για να λαμβάνουν δεδομένα από τη βάση.

Το web service μέσω του οποίου στέλνονται δεδομένα στη βάση ονομάζεται **up_service.php**. Πάμε να δούμε αναλυτικά πως γίνεται σύνδεση και η αποστολή των δεδομένων. Αρχικά δημιουργούμε τη σύνδεση με τη βάση δεδομένων με την παρακάτω εντολή.

```
$mysqli = new mysqli($host, $username, $password, $dbName);
```

Η μεταβλητή ***$host*** περιέχει τη διεύθυνση της βάσης, η μεταβλητή ***$username*** περιέχει το όνομα του χρήστη που διαχειρίζεται τη βάση, η μεταβλητή ***$password*** περιέχει τον κωδικό σύνδεσης στη βάση και τέλος η μεταβλητή ***$dbName*** περιέχει το όνομα της βάσης. Το αποτέλεσμα αυτής της εντολής αποθηκεύεται στη μεταβλητή ***$mysqli***. Έπειτα γίνεται έλεγχος για να δούμε αν υπάρχει κάποιο πρόβλημα με την σύνδεση. Σε περίπτωση προβλήματος εμφανίζεται αντίστοιχο μήνυμα και γίνεται έξοδος από το web service. Ο παραπάνω έλεγχος γίνεται με τον παρακάτω κώδικα.

```
if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
}
```

Μετά απενεργοποιούμε τις αυτόματες αλλαγές στη βάση δεδομένων με την επόμενη εντολή.

```
$mysqli->autocommit(FALSE);
```

Έπειτα παίρνουμε το SQL ερώτημα από τη συσκευή, το αποθηκεύουμε στη μεταβλητή ***$sql*** και το εκτελούμε.

```
$sql = $_POST['sql'];
$mysqli->query($sql);
```

Ενημερώνουμε τη βάση.

```
$mysqli->commit();
```

Και τέλος κλείνουμε την σύνδεση.

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

Το web service μέσω του οποίου γίνεται η λήψη δεδομένων από την εκάστοτε συσκευή ονομάζεται **down_service.php**. Πάμε να δούμε αναλυτικά πως γίνεται σύνδεση και η κωδικοποίηση και η λήψη των δεδομένων. Αρχικά δημιουργούμε τη σύνδεση με τη βάση δεδομένων με την παρακάτω εντολή.

Η μεταβλητή ***$host*** περιέχει τη διεύθυνση της βάσης, η μεταβλητή ***$username*** περιέχει το όνομα του χρήστη που διαχειρίζεται τη βάση, η μεταβλητή ***$password*** περιέχει τον κωδικό σύνδεσης στη βάση και τέλος η μεταβλητή ***$dbName*** περιέχει το όνομα της βάσης. Το αποτέλεσμα αυτής της εντολής αποθηκεύεται στη μεταβλητή ***$mysqli***. Έπειτα γίνεται έλεγχος για να δούμε αν υπάρχει κάποιο πρόβλημα με την σύνδεση. Σε περίπτωση

προβλήματος εμφανίζεται αντίστοιχο μήνυμα και γίνεται έξοδος από το web service. Ο παραπάνω έλεγχος γίνεται με τον παρακάτω κώδικα.

```
if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
}
```

Μετά απενεργοποιούμε τις αυτόματες αλλαγές στη βάση δεδομένων με την επόμενη εντολή.

```
$mysqli->autocommit(FALSE);
```

Έπειτα παίρνουμε το SQL ερώτημα από τη συσκευή, το αποθηκεύουμε στη μεταβλητή ***$sql*** και το εκτελούμε.

```
$sql = $_POST['sql'];
$mysqli->query($sql);
```

Στη συνέχεια με τις παρακάτω εντολές δημιουργούμε ένα νέο πίνακα με το όνομα ***$dataArray***, εκτελούμε το SQL ερώτημα, ελέγχουμε αν υπάρχουν δεδομένα και όσο υπάρχουν με τη βοήθεια του πίνακα ***$tempArray*** γεμίζουμε τον πίνακα ***$dataArray*** με το αποτέλεσμα του SQL ερωτήματος. Έπειτα κωδικοποιούμε τον πίνακά μας σε JSON format και επιστρέφουμε το αποτέλεσμα στη συσκευή.

```
$dataArray = array();
if ($result = $mysqli->query($sql)) {
        $tempArray = array();

        while($row = $result->fetch_object()) {

                $tempArray = $row;
                array_push($dataArray, $tempArray);
        }
```

```
        echo json_encode($dataArray);
}
```

Ενημερώνουμε τη βάση.

```
$mysqli->commit();
```

Και τέλος κλείνουμε την σύνδεση.

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

ΑΡΧΙΤΕΚΤΟΝΙΚΗ ALARM CLOCK MODULE

Τα βασικά modules με τα οποία διαχειριζόμαστε τις ενέργειες του ξυπνητηριού είναι δύο. Το ένα ονομάζεται **SetAlarmWebServiceTask** και παίρνει τα δεδομένα από τον χρήστη και τα στέλνει στη βάση και το άλλο ονομάζεται **runAlarmAction()**, βρίσκεται στη κλάση **ActionsReceiver**, λαμβάνει δεδομένα από τη βάση και εκτελεί τις αντίστοιχες ενέργειες.

Πάμε θα δούμε αναλυτικά τη κλάση **SetAlarmWebServiceTask** η οποία έχει χαρακτηριστικά AsyncTask. Δηλαδή είναι ένα τμήμα κώδικα που τρέχει ασύγχρονα και υλοποιεί το μέρος του client μέσω του οποίου στέλνονται τα δεδομένα στο μέσω του web service **up_service.php**. Η κλάση λοιπόν αυτή περιέχει τις εξής μεθόδους :

- protected void **onPreExecute() :** σε αυτή τη μέθοδο βρίσκονται οι απαραίτητες εντολές πριν ξεκινήσει η εκτέλεση του module.

- protected String **doInBackground(String... params) :** σε αυτή τη μέθοδο γίνεται η εκτέλεση του module.
- protected void **onPostExecute(String file_url) :** σε αυτή τη μέθοδο βρίσκονται οι εντολές που εκτελούνται μετά την ολοκλήρωση της εκτέλεσης του module.
- String **runWebService() :** είναι η πιο σημαντική μέθοδος της κλάσης, εκτελείται από τη μέθοδο doInBackground(String... params) και υλοποιεί τη σύνδεση με τη βάση και τη λήψη των δεδομένων τα οποία επιστρέφονται ως μια ακολουθία συμβόλων.

Στο τμήμα της εφαρμογής που λειτουργεί ως παραλήπτης αναφορικά με την εφαρμογή του ξυπνητηριού έχουμε τη μέθοδο **runAlarmAction()**. Η μέθοδος αυτή χρησιμοποιεί το **down_service.php** web service για τη λήψη των δεδομένων από τη βάση. Πάμε να την αναλύσουμε. Ξεκινώντας παίρνουμε την ακολουθία συμβολών που περιέχει την την ενέργεια που θέλουμε να εκτελέσουμε. Η ακολουθία αυτή αποτελείται από επιμέρους τμήματα που περιέχουν το είδος της εντολής, εδώ συγκεκριμένα είναι το alarm, το αναγνωριστικό του αποστολέα, το μήνυμα που θα εμφανιστεί και την ώρα που θα μπει το ξυπνητήρι. Παίρνοντας αυτά τα δεδομένα χρησιμοποιούμε το API που μας έχει δώσει η Google για τη διαχείριση του ξυπνητηριού και εισάγουμε καινούριο ξυπνητήρι.

ΑΡΧΙΤΕΚΤΟΝΙΚΗ MAP MODULE

Τα βασικά modules με τα οποία διαχειριζόμαστε τις ενέργειες του χάρτη είναι δύο. Το ένα ονομάζεται **SetMapWebServiceTask** και παίρνει τα δεδομένα από τον χρήστη και τα στέλνει στη βάση και το άλλο ονομάζεται **runMapAction()**, βρίσκεται στη κλάση **ActionsReceiver**, λαμβάνει δεδομένα από τη βάση και εκτελεί τις αντίστοιχες ενέργειες.

Πάμε θα δούμε αναλυτικά τη κλάση **SetMapWebServiceTask** η οποία έχει χαρακτηριστικά AsyncTask. Δηλαδή είναι ένα τμήμα κώδικα που τρέχει ασύγχρονα και υλοποιεί το μέρος

του client μέσω του οποίου στέλνονται τα δεδομένα στο μέσω του web service **up_service.php**. Η κλάση λοιπόν αυτή περιέχει τις εξής μεθόδους :

- protected void **onPreExecute() :** σε αυτή τη μέθοδο βρίσκονται οι απαραίτητες εντολές πριν ξεκινήσει η εκτέλεση του module.
- protected String **doInBackground(String... params) :** σε αυτή τη μέθοδο γίνεται η εκτέλεση του module.
- protected void **onPostExecute(String file_url) :** σε αυτή τη μέθοδο βρίσκονται οι εντολές που εκτελούνται μετά την ολοκλήρωση της εκτέλεσης του module.
- String **runWebService() :** είναι η πιο σημαντική μέθοδος της κλάσης, εκτελείται από τη μέθοδο doInBackground(String... params) και υλοποιεί τη σύνδεση με τη βάση και τη λήψη των δεδομένων τα οποία επιστρέφονται ως μια ακολουθία συμβόλων.

Στο τμήμα της εφαρμογής που λειτουργεί ως παραλήπτης αναφορικά με την εφαρμογή του χάρτη έχουμε τη μέθοδο **runMapAction()**. Η μέθοδος αυτή χρησιμοποιεί το **down_service.php** web service για τη λήψη των δεδομένων από τη βάση. Πάμε να την αναλύσουμε. Ξεκινώντας παίρνουμε την ακολουθία συμβολών που περιέχει την την ενέργεια που θέλουμε να εκτελέσουμε. Η ακολουθία αυτή αποτελείται από επιμέρους τμήματα που περιέχουν το είδος της εντολής, εδώ συγκεκριμένα είναι το map, το αναγνωριστικό του αποστολέα και τις συντεταγμένες του σημείου συνάντησης. Παίρνοντας αυτά τα δεδομένα χρησιμοποιούμε το API που μας έχει δώσει η Google για τη διαχείριση του χάρτη και εισάγουμε καινούριο σημείο συνάντησης στην εφαρμογή του χάρτη και επίσης υπολογίζεται αυτόματα και εμφανίζεται και η διαδρομή από το μέρος που βρίσκεται ο χρήστης έως το σημείο συνάντησης.

ΓΕΝΙΚΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΕΝΟΣ MODULE

Βλέποντας την αρχιτεκτονική των παραπάνω module μπορούμε να παρουσιάσουμε και μια γενική αρχιτεκτονική ενός module για τη διαχείριση δυνητικά οποιασδήποτε εφαρμογής.

Αρχικά θα υπάρχει μια κλάση **_WebServiceTask** η οποία θα παίρνει τα δεδομένα από τον χρήστη και τα στέλνει στη βάση. Η κλάση αυτή θα περιέχει τις εξής μεθόδους :

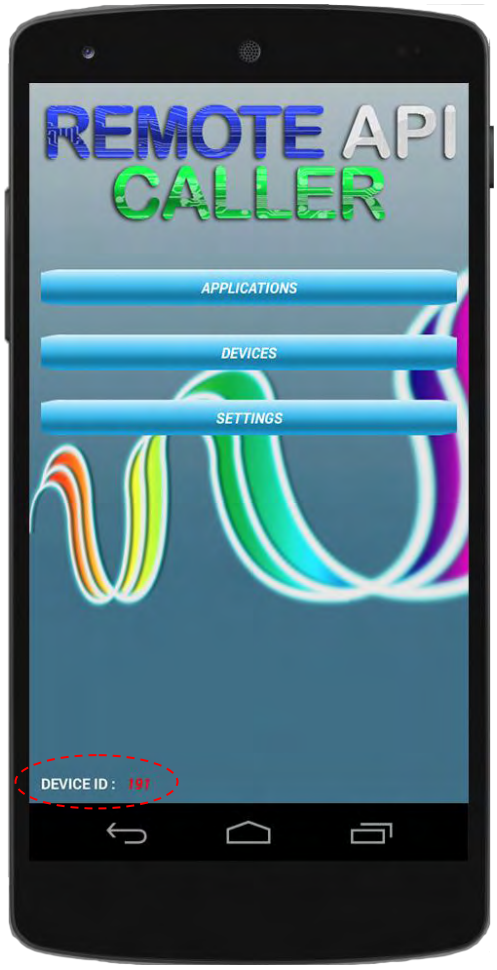
- protected void **onPreExecute() :** σε αυτή τη μέθοδο βρίσκονται οι απαραίτητες εντολές πριν ξεκινήσει η εκτέλεση του module.
- protected String **doInBackground(String... params) :** σε αυτή τη μέθοδο γίνεται η εκτέλεση του module.
- protected void **onPostExecute(String file_url) :** σε αυτή τη μέθοδο βρίσκονται οι εντολές που εκτελούνται μετά την ολοκλήρωση της εκτέλεσης του module.
- String **runWebService() :** είναι η πιο σημαντική μέθοδος της κλάσης, εκτελείται από τη μέθοδο doInBackground(String... params) και υλοποιεί τη σύνδεση με τη βάση και τη λήψη των δεδομένων τα οποία επιστρέφονται ως μια ακολουθία συμβόλων.

Και επίσης θα υπάρχει μέσα στην κλάση **ActionsReceiver** μια μέθοδος που θα χρησιμοποιεί το **down_service.php** web service για τη λήψη των δεδομένων από τη βάση, θα διαχειρίζεται τα δεδομένα και θα εκτελεί τις αντίστοιχες ενέργειες.

ΚΕΦΑΛΑΙΟ 3

ΛΕΙΤΟΥΡΓΙΑ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

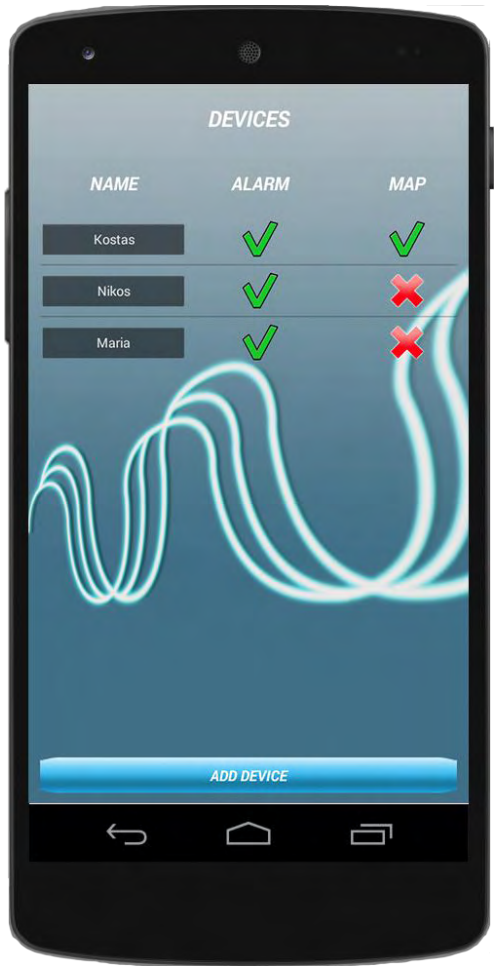
Η εφαρμογή που έχουμε υλοποιήσει λειτουργεί ως εξής. Αρχικά κατά την πρώτη εκτέλεσή της δημιουργείται αυτόματα ένα αναγνωριστικό το **device_id** που είναι μοναδικό για κάθε συσκευή (**ΣΧΗΜΑ 3.1**).



ΣΧΗΜΑ 3.1

Ανοίγοντας λοιπόν την εφαρμογή ο χρήστης μπορεί να λειτουργήσει ως αποστολέας (τη διαδικασία αποστολής ενεργειών σε άλλες συσκευές μέσω παραδειγμάτων εισαγωγής

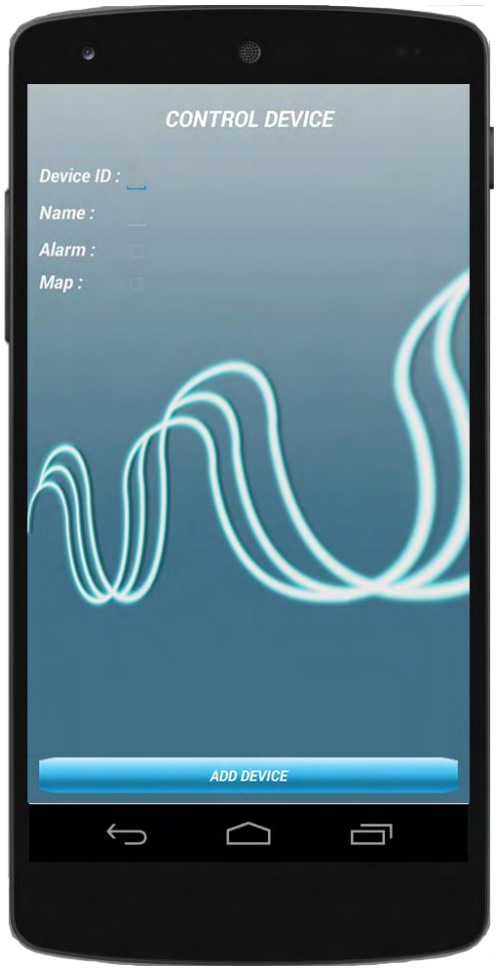
ξυπνητηριού και χάρτη θα δούμε στο επόμενο κεφάλαιο), μπορεί να ορίσει μια νέα συσκευή που θα της επιτρέπει να διαχειρίζεται τη δική του ή και και να ενημερώσει/διαγράψει μία υπάρχουσα συσκευή και τέλος μπορεί να αλλάξει το χρονικό διάστημα μεταξύ της λήψης δεδομένων από τη βάση, μια λειτουργία που γίνεται αυτόματα στο παρασκήνιο από την εφαρμογή.



ΣΧΗΜΑ 3.2

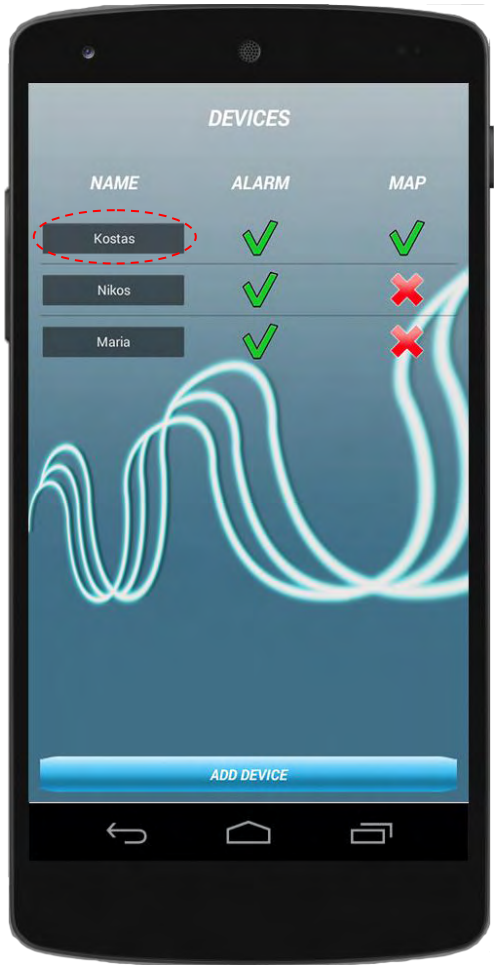
Πατώντας το κουμπί **DEVICES** μεταφέρεται στη επόμενη σελίδα της εφαρμογής όπου θα του εμφανιστεί μια λίστα με όλες τις συσκευές στις οποίες έχει δώσει δικαιώματα διαχείρισης καθώς και ποια είναι αυτά (**ΣΧΗΜΑ 3.2**). Η λίστα αυτή είναι δυναμική ώστε να μπορούν συνεχώς να προστίθενται καινούργιες συσκευές και φυσικά να διαγράφονται οι υπάρχουσες.

Επιλέγοντας το κουμπί **ADD DEVICE** μεταφερόμαστε στην επόμενη σελίδα (**ΣΧΗΜΑ 3.3**) όπου μπορούμε να προσθέσουμε νέα συσκευή εισάγοντας το device_id που μας έχει δοθεί από τον χρήστη-αποστολέα και συμπληρώνοντας τα απαραίτητα στοιχεία που είναι το όνομα και τα δικαιώματα που του δίνουμε αναφορικά με την εφαρμογή του ξυπνητηριού και του χάρτη.



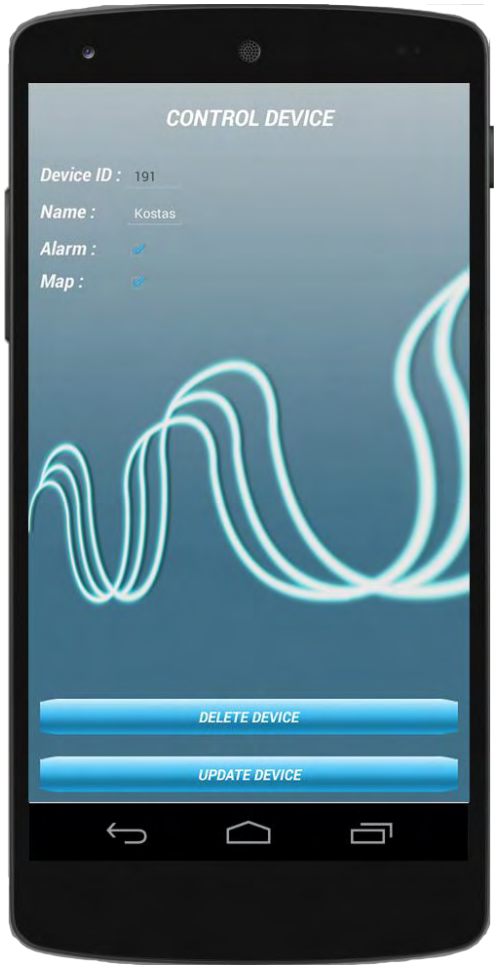
ΣΧΗΜΑ 3.3

Τέλος πατώντας το κουμπί **ADD DEVICE** αυτής της σελίδας προσθέτουμε τη νέα συσκευή ή σε περίπτωση που το **device_id** που έχουμε δώσει υπάρχει ήδη στη λίστα μας, δηλαδή έχουμε ήδη προσθέσει αυτή τη συσκευή, γίνεται ενημέρωση των στοιχείων της. Επιστρέφουμε και πάλι στη σελίδα που περιέχει τη λίστα των συσκευών (**ΣΧΗΜΑ 3.4**) για να δούμε μια επιπλέον δυνατότητα που μας δίνεται.



ΣΧΗΜΑ 3.4

Πατώντας το όνομα της συσκευής μεταφερόμαστε σε μια σελίδα (**ΣΧΗΜΑ 3.5**) που είναι είναι παρόμοια με αυτή του σχήματος 3.3 όπου μπορούμε να ενημερώσουμε τα στοιχεία της συσκευής που πατήσαμε και επίσης μπορούμε και να τη διαγράψουμε. Μπορούμε να αλλάξουμε όλα τα στοιχεία εκτός από το αναγνωριστικό της συσκευής.



ΣΧΗΜΑ 3.5

Τέλος από την αρχική σελίδα της εφαρμογής μπορούμε πατώντας το κουμπί **SETTINGS** να μεταφερθούμε στη σελίδα (**ΣΧΗΜΑ 3.6**) όπου μπορούμε να αλλάξουμε το χρονικό διάστημα μεταξύ της λήψης δεδομένων από τη βάση.

ΣΧΗΜΑ 3.6

Είναι ουσιαστικά ένας θετικός αριθμός που ορίζει το ρυθμό επανάληψης λήψης ενεργειών που θα εκτελέσει η συσκευή. Το διάστημα αυτό έχει οριστεί αρχικά στα 60 δευτερόλεπτα.

ΚΕΦΑΛΑΙΟ 4

ΕΦΑΡΜΟΓΗ ΞΥΠΝΗΤΗΡΙΟΥ ΚΑΙ ΧΑΡΤΗ

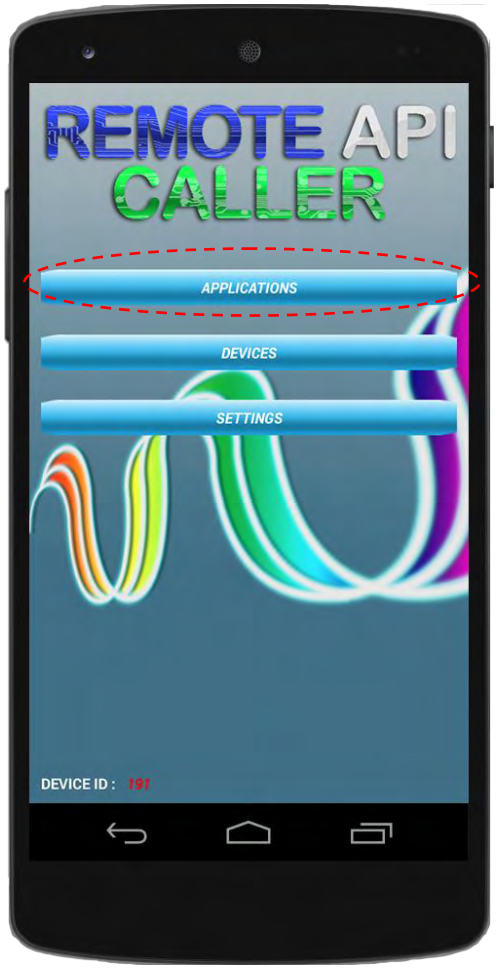
Έχοντας αναλύσει σε προηγούμενο κεφάλαιο το πως εισάγουμε καινούρια συσκευή προς διαχείριση, σε αυτό το κεφάλαιο θα δούμε αναλυτικά το τρόπο με τον οποίο γίνεται η διαχείριση των δύο εφαρμογών που επιλέξαμε. Αυτές είναι, η εφαρμογή του ξυπνητηριού και η εφαρμογή του χάρτη.

ΕΦΑΡΜΟΓΗ ΞΥΠΝΗΤΗΡΙΟΥ

Παρακάτω θα δούμε τη λειτουργία της εφαρμογής ως αποστολέας (sender) για την εφαρμογή του ξυπνητηριού, δηλαδή τα βήματα που κάνουμε για να εισάγουμε καινούριο ξυπνητήρι στη συσκευή που έχουμε επιλέξει καθώς και τον κώδικα που εκτελείται.

Επίσης θα δούμε και τη λειτουργία της ως δέκτης (receiver), δηλαδή όλες τις μεθόδους που λαμβάνουν τα δεδομένα από τη βάση και εκτελούν τις αντίστοιχες ενέργειες που είναι σχετικές με την εφαρμογή του ξυπνητηριού.

Λειτουργώντας λοιπόν ως αποστολέας, ξεκινώντας βλέπουμε την αρχική σελίδα της εφαρμογής (**ΣΧΗΜΑ 4.1**) και επιλέγουμε το κουμπί **APPLICATIONS**.



ΣΧΗΜΑ 4.1

Έτσι μεταφερόμαστε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.2**) στην οποία εμφανίζονται όλες οι εφαρμογές που μπορούμε να διαχειριστούμε. Στα πλαίσια του παραδείγματός μας θα επιλέξουμε το **ALARM**.

ΣΧΗΜΑ 4.2

Έπειτα μεταφερόμαστε στην επόμενη σελίδα για να επιλέξουμε το mode που θέλουμε. Προς το παρόν υπάρχουν δύο διαθέσιμα modes, το **SINGLE RECEIVER** mode (**ΣΧΗΜΑ 4.3**) το οποίο επιλέγουμε για να στείλουμε μια ενέργεια στην εφαρμογή που επιλέξαμε στη προηγούμενη σελίδα προς ένα μόνο χρήστη και το **MULTIPLE RECEIVERS** mode (**ΣΧΗΜΑ 4.4**) το οποίο επιλέγουμε για να στείλουμε την ίδια ενέργεια σε πολλούς χρήστες ταυτόχρονα.

ΣΧΗΜΑ 4.3

ΣΧΗΜΑ 4.4

Επιλέγοντας το κουμπί **CONTINUE** μεταφερόμαστε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.5**) στη οποία εμφανίζεται δυναμικά η λίστα όλων των συσκευών-χρηστών, εκτελώντας τη μέθοδο **`loadReceiversListOnScreen()`**, για τις οποίες μας επιτρέπεται να διαχειριστούμε την εφαρμογή Alarm Clock. Το SQL ερώτημα που στέλνεται για να πάρουμε τα δεδομένα από τη βάση είναι το "**`SELECT`** `sender_id, receiver_id, alarm, map, receiver_name` **`FROM`** `Privileges` **`WHERE`** `sender_id='" + deviceId + "'&&alarm='1'"`.

ΣΧΗΜΑ 4.5

Αν το mode που είχαμε επιλέξει προηγουμένως είναι το single receiver, τότε πατώντας στο όνομα κάποιας από τα τις συσκευές που έχουν εμφανιστεί μεταφερόμαστε απευθείας στην επόμενη σελίδα (**ΣΧΗΜΑ 4.6**), ενώ αν το mode είναι το multiple receivers τότε πατώντας σε κάποιο όνομα επιλέγουμε ή αποεπιλέγουμε μία συσκευή και ύστερα επιλέγουμε το κουμπί **SET ACTION** για να συνεχίσουμε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.6**) για να ορίσουμε την ενέργεια που θέλουμε να στείλουμε.

ΣΧΗΜΑ 4.6

Εδώ εισάγουμε τα δεδομένα του ξυπνητηριού που είναι το μήνυμα που θα εμφανιστεί και το πότε θα χτυπήσει το ξυπνητήρι. Και τέλος επιλέγουμε το κουμπί **ADD ALARM** και εκτελείται ο κώδικας της μεθόδου **`sendActionAddAlarm(View view)`** για να σταλεί το ξυπνητήρι στη συσκευή του δέκτη. Πιο συγκεκριμένα, στέλνεται το παρακάτω SQL ερώτημα "**`INSERT INTO`** `Actions (sender_id, receivers_id, action, state)` **`VALUES`** `('" + deviceId + "','" + receiverId + "','" + action + "','0')`". Στο τέλος εμφανίζεται το μήνυμα "**Alarm Action Completed**" για δείξουμε στο χρήστη ότι η ενέργεια στάλθηκε επιτυχώς. Σε αντίθετη περίπτωση εμφανίζεται κατάλληλο μήνυμα.

Όταν λειτουργεί ως δέκτης, υπάρχει μία κλάση που ονομάζεται **ActionsReceiver** της οποίας οι μέθοδοι τρέχουν συνεχώς στο παρασκήνιο, ακόμα και όταν η εφαρμογή δεν είναι ανοιχτή, και αυτό που κάνει είναι να παίρνει δεδομένα από τη βάση και πιο συγκεκριμένα από τους πίνακες **Actions** και **ActionsTargerts** όπου είναι αποθηκευμένες όλες οι ενέργειες.

ΣΧΗΜΑ 4.7

Το SQL ερώτημα που στέλνεται είναι το `"SELECT DISTINCT Actions.action_id, Actions.sender_id, Actions.action, ActionsTargets.completed " + "FROM Actions "+"JOIN ActionsTargets " + "ON Actions.action_id =ActionsTargets.action_id " + "WHERE ActionsTargets.receiver_id = '" + deviceId + "' && ActionsTargets.completed='0'"`, το οποίο επιστρέφει μεταξύ άλλων και όλες τις ενέργειες που δεν έχουν γίνει ακόμα. Για την ακρίβεια

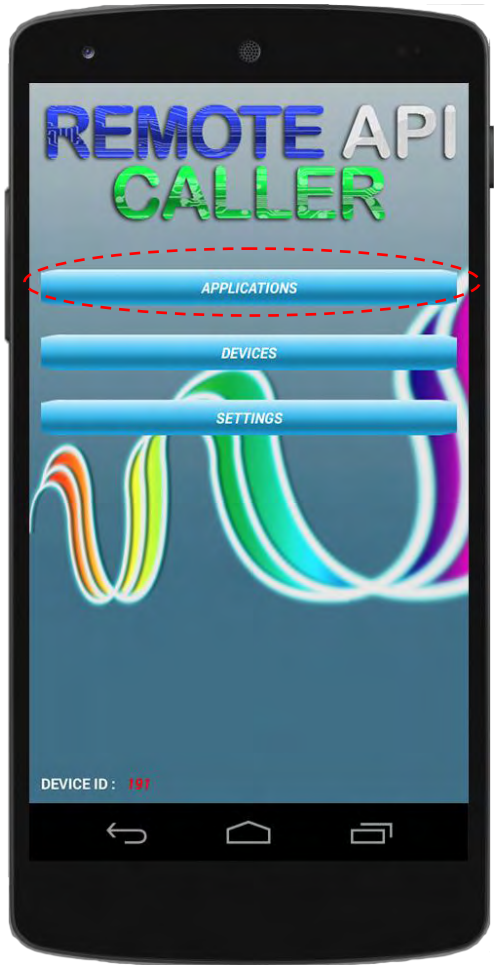
αυτό που επιστρέφει είναι τις ακολουθίες συμβόλων που έστειλε ο αποστολέας. Από το πρώτο μέρος (μέχρι το χαρακτήρα '-') αναγνωρίζει την ενέργεια που έχει να κάνει και την εκτελεί. Εδώ συγκεκριμένα γίνεται η εισαγωγή καινούργιου ξυπνητηριού μέσω της μεθόδου **`runAlarmAction (String actionString)`**. Το αποτέλεσμα το βλέπουμε στο (**ΣΧΗΜΑ 4.7**). Έπειτα για κάθε ενέργεια που έχει εκτελέσει ενημερώνει τον πίνακα **ActionsTargets** ότι η αντίστοιχη ενέργεια έχει ολοκληρωθεί. Αυτό γίνεται στέλνοντας το παρακάτω SQL ερώτημα "**`UPDATE`** `ActionsTargets` **`SET`** `completed='1'` **`WHERE`** `(" + actionsId + ") && receiver_id=" + deviceId`".

ΕΦΑΡΜΟΓΗ ΧΑΡΤΗ

Παρακάτω θα δούμε τη λειτουργία της εφαρμογής ως αποστολέας (sender) για την εφαρμογή του χάρτη, δηλαδή τα βήματα που κάνουμε για να στείλουμε ένα σημείο συνάντησης στην εφαρμογή του χάρτη στη συσκευή που έχουμε επιλέξει εμφανίζοντας αυτόματα και το μονοπάτι που μπορεί να ακολουθήσει ο χρήστης από το σημείο που βρίσκεται μέχρι το σημείο συνάντησης καθώς και τον κώδικα που εκτελείται.

Επίσης θα δούμε και τη λειτουργία της ως δέκτης (receiver), δηλαδή όλες τις μεθόδους που λαμβάνουν τα δεδομένα από τη βάση και εκτελούν τις αντίστοιχες ενέργειες που είναι σχετικές με την εφαρμογή του χάρτη.

Λειτουργώντας λοιπόν ως αποστολέας, ξεκινώντας βλέπουμε την αρχική σελίδα της εφαρμογής (**ΣΧΗΜΑ 4.8**) και επιλέγουμε το κουμπί **APPLICATIONS**.



ΣΧΗΜΑ 4.8

Έτσι μεταφερόμαστε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.9**) στην οποία εμφανίζονται όλες οι εφαρμογές που μπορούμε να διαχειριστούμε. Στα πλαίσια του παραδείγματός μας θα επιλέξουμε το **MAP**.

ΣΧΗΜΑ 4.9

Έπειτα μεταφερόμαστε στην επόμενη σελίδα για να επιλέξουμε το mode που θέλουμε. Προς το παρόν υπάρχουν δύο διαθέσιμα modes, το **SINGLE RECEIVER** mode (**ΣΧΗΜΑ 4.10**) το οποίο επιλέγουμε για να στείλουμε μια ενέργεια στην εφαρμογή που επιλέξαμε στη προηγούμενη σελίδα προς ένα μόνο χρήστη και το **MULTIPLE RECEIVERS** mode (**ΣΧΗΜΑ 4.11**) το οποίο επιλέγουμε για να στείλουμε την ίδια ενέργεια σε πολλούς χρήστες ταυτόχρονα.

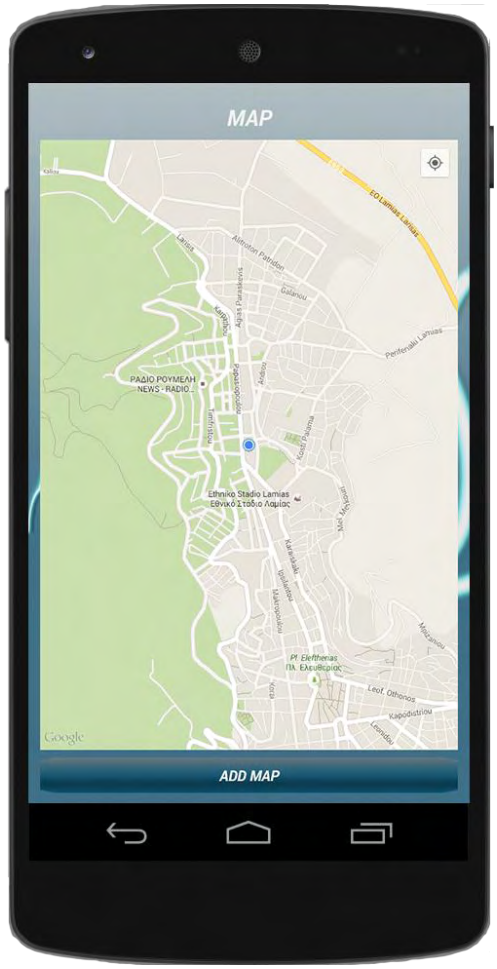
ΣΧΗΜΑ 4.10

ΣΧΗΜΑ 4.11

Επιλέγοντας το κουμπί **CONTINUE** μεταφερόμαστε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.12**) στη οποία εμφανίζεται δυναμικά η λίστα όλων των συσκευών-χρηστών, εκτελώντας τη μέθοδο **`loadReceiversListOnScreen()`**, για τις οποίες μας επιτρέπεται να διαχειριστούμε την εφαρμογή Maps. Το SQL ερώτημα που στέλνεται για να πάρουμε τα δεδομένα από τη βάση είναι το "**`SELECT`** `sender_id, receiver_id, alarm, map, receiver_name` **`FROM`** `Privileges` **`WHERE`** `sender_id='" + deviceId + "'&&map='1'"`.

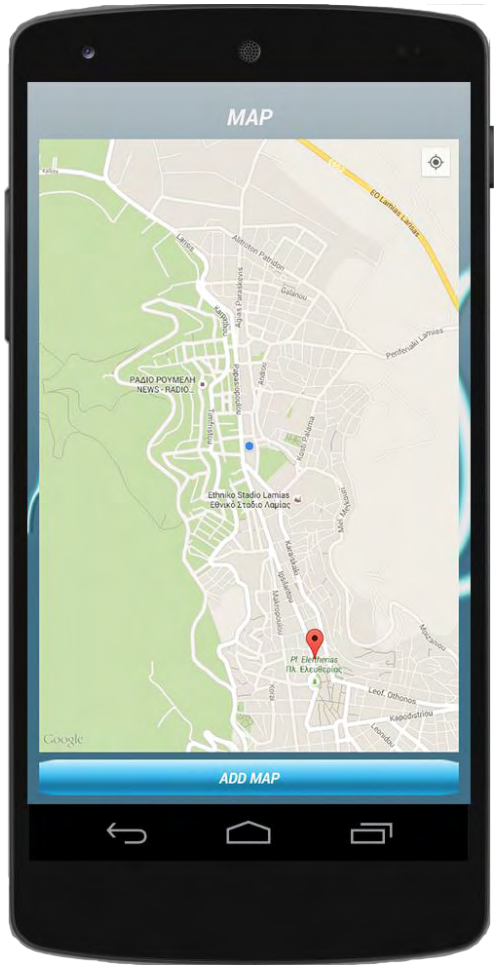
ΣΧΗΜΑ 4.12

Αν το mode που είχαμε επιλέξει, προηγουμένως είναι το single receiver, τότε πατώντας στο όνομα κάποιας από τα τις συσκευές που έχουν εμφανιστεί μεταφερόμαστε απευθείας στην επόμενη σελίδα (**ΣΧΗΜΑ 4.13**), ενώ αν το mode είναι το multiple receivers τότε πατώντας σε κάποιο όνομα επιλέγουμε ή αποεπιλέγουμε μία συσκευή και ύστερα επιλέγουμε το κουμπί **SET ACTION** για να συνεχίσουμε στην επόμενη σελίδα (**ΣΧΗΜΑ 4.13**) για να ορίσουμε την ενέργεια που θέλουμε να στείλουμε.



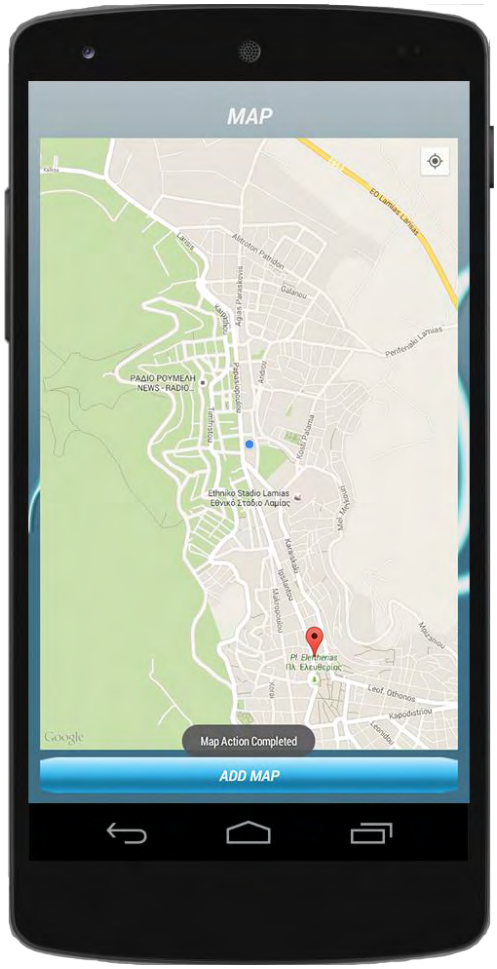
ΣΧΗΜΑ 4.13

Η αρχικοποίηση και εμφάνιση του χάρτη γίνεται από τη μέθοδο **`setUpMapIfNeeded()`** η οποία αρχικοποιεί τον χάρτη μόνο όταν χρειάζεται. Για λόγους ασφαλείας το κουμπί **ADD MAP** που θα στείλει τα δεδομένα στη βάση είναι απενεργοποιημένο γιατί δεν έχουμε ορίσει ακόμα το σημείο που θέλουμε. Μπορούμε να εστιάζουμε ή να απομακρυνθούμε επιλέγοντας τα κουμπιά + και – αντίστοιχα ή χρησιμοποιώντας τα δυο δάχτυλα και πατώντας το κουμπί πάνω δεξιά εστιάζουμε στο σημείο που βρισκόμαστε εμείς και εμφανίζεται ένας μπλε κύκλος. Τα τρία παραπάνω κουμπιά ανάλογα με τη συσκευή και την έκδοση του λογισμικού της μπορεί να βρίσκονται και κάπου αλλού μέσα στο χώρο. Έπειτα για να ορίσουμε σημείο συνάντησης πατάμε οπουδήποτε πάνω στο χάρτη και αυτό το σημείο μαρκάρεται και εμφανίζεται ένα σημάδι (**ΣΧΗΜΑ 4.14**).



ΣΧΗΜΑ 4.14

Πατώντας κάπου αλλού το σημάδι εξαφανίζεται και εμφανίζεται στο καινούριο σημείο. Η παραπάνω διαδικασία υλοποιείται από τη μέθοδο **`onMapClick(LatLng latLng)`**. Για να στείλουμε τελικά την ενέργεια στη συσκευή του δέκτη επιλέγουμε το κουμπί **ADD MAP** και εκτελείται ο κώδικας της μεθόδου **`sendActionSendPosition(View v)`** για να σταλεί το σημείο συνάντησης στη συσκευή του δέκτη.



ΣΧΗΜΑ 4.15

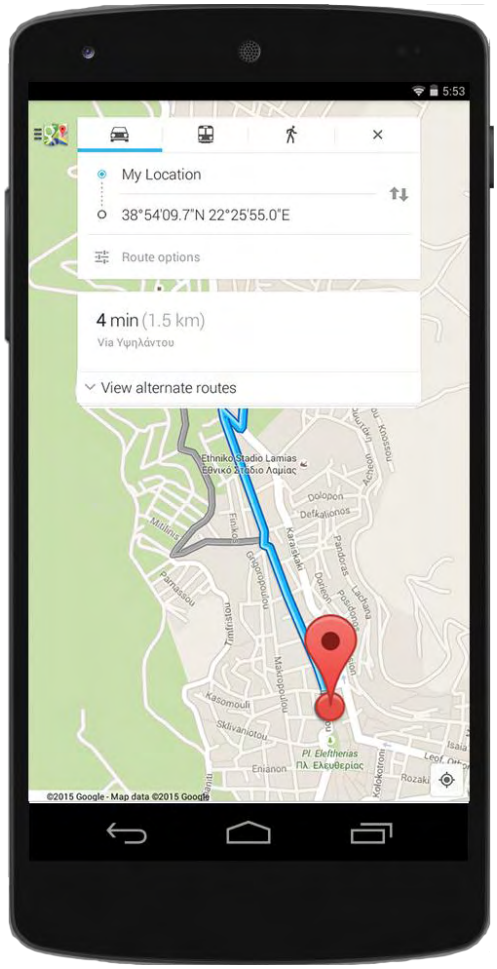
Πιο συγκεκριμένα, στέλνεται το παρακάτω SQL ερώτημα "**INSERT INTO** Actions (sender_id, receivers_id, action, state) **VALUES** ('" + deviceId + "','" + receiverId + "','" + action + "','0')". Στο τέλος εμφανίζεται το μήνυμα "**Map Action Completed**" για δείξουμε στο χρήστη ότι η ενέργεια στάλθηκε επιτυχώς (**ΣΧΗΜΑ 4.15**). Σε αντίθετη περίπτωση εμφανίζεται κατάλληλο μήνυμα.

Όταν λειτουργεί ως δέκτης, υπάρχει μία κλάση που ονομάζεται **ActionsReceiver** της οποίας οι μέθοδοι τρέχουν συνεχώς στο παρασκήνιο, ακόμα και όταν η εφαρμογή δεν είναι ανοιχτή, και αυτό που κάνει είναι να παίρνει δεδομένα από τη βάση και πιο συγκεκριμένα από τους πίνακες **Actions** και **ActionsTargerts** όπου είναι αποθηκευμένες όλες οι ενέργειες.

Το SQL ερώτημα που στέλνεται είναι το "**SELECT DISTINCT** Actions.action_id, Actions.sender_id, Actions.action, ActionsTargets.completed " + "**FROM** Actions "+"**JOIN** ActionsTargets " + "**ON** Actions.action_id =ActionsTargets.action_id " + "**WHERE** ActionsTargets.receiver_id = '" + deviceId + "' && ActionsTargets.completed='0'", το οποίο επιστρέφει μεταξύ άλλων και όλες τις ενέργειες που δεν έχουν γίνει ακόμα. Για την ακρίβεια αυτό που επιστρέφει είναι τις ακολουθίες συμβόλων που έστειλε ο αποστολέας. Από το πρώτο μέρος (μέχρι το χαρακτήρα '-') αναγνωρίζει την ενέργεια που έχει να κάνει και την εκτελεί.

ΣΧΗΜΑ 4.16

Εδώ συγκεκριμένα γίνεται η εμφάνιση σημείου συνάντησης στο χάρτη. Η ενέργεια αυτή αποτελείται από δύο μέρη. Αρχικά εμφανίζεται μία ειδοποίηση μέσω της μεθόδου **`runMapAction (String actionString)`** (**ΣΧΗΜΑ 4.16**) η οποία επίσης δημιουργεί μία εν αναμονή διεργασία (PendingIntent). Μόλις ο χρήστης επιλέξει την ειδοποίηση, εκτελείται και διεργασία που του εμφανίζει στην εφαρμογή του χάρτη το σημείο συνάντησης καθώς και το μονοπάτι που μπορεί να ακολουθήσει από το σημείο που βρίσκεται μέχρι το σημείο συνάντησης (**ΣΧΗΜΑ 4.17**).



ΣΧΗΜΑ 4.17

Έπειτα για κάθε ενέργεια που έχει εκτελέσει ενημερώνει τον πίνακα **ActionsTargets** ότι η αντίστοιχη ενέργεια έχει ολοκληρωθεί. Αυτό γίνεται στέλνοντας το παρακάτω SQL ερώτημα

"**UPDATE** ActionsTargets **SET** completed='1' **WHERE** (" + actionsId + ") && receiver_id=" + deviceId".

ΚΕΦΑΛΑΙΟ 5

ΔΟΜΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ ΣΕ ANDROID

Σε αυτό το κεφάλαιο θα δούμε αναλυτικά τη δομή της εφαρμογής όπως αυτή υλοποιήθηκε σε Android. Αρχικά θα παρουσιάσουμε συνολικά όλες τις βιβλιοθήκες και τα πακέτα που χρησιμοποιήθηκαν και έπειτα θα δούμε όλες τις κλάσεις και τις μεθόδους που περιέχονται στις βιβλιοθήκες αυτές και ήταν απαραίτητες για την υλοποίηση της εφαρμογής [15]. Υπάρχουν τρεις μεγάλες κατηγορίες βιβλιοθηκών.

Αρχικά έχουμε τις βιβλιοθήκες που περιέχουν τις κλάσεις και τις μεθόδους που είναι απαραίτητες για την ανάπτυξη εφαρμογών για συσκευές που έχουν ως λειτουργικό σύστημα το Android και είναι οι παρακάτω :

```
import android.app.Activity;
import android.app.AlarmManager;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.location.Location;
import android.location.LocationManager;
```

```
import android.graphics.Color;
import android.graphics.Typeface;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.provider.AlarmClock;
import android.provider.Settings;
import android.support.v4.app.FragmentActivity;
import android.support.v4.app.NotificationCompat;
import android.util.Log;
import android.util.DisplayMetrics;
import android.view.View;
import android.view.Display;
import android.view.Gravity;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.TimePicker;
```

```
import android.widget.Toast;


import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
```

Έπειτα χρησιμοποιήθηκαν βιβλιοθήκες για την επικοινωνία της εφαρμογής με τον server και για την αποστολή και τη λήψη των δεδομένων. Και είναι οι παρακάτω :

```
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;
```

Τέλος χρησιμοποιήθηκαν και βιβλιοθήκες της Java για τη χρησιμοποίηση μεθόδων που σκοπό είχαν τη δημιουργία και διαχείριση αρχείων, ροών δεδομένων και δομών αποθήκευσης. Και είναι οι παρακάτω:

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.ArrayList;
import java.util.List;
import java.util.Locale;
import java.util.Random;
import java.util.StringTokenizer;
```

Στη συνέχεια πάμε να δούμε πιο αναλυτικά τι περιέχουν οι πιο σημαντικές από τις παραπάνω βιβλιοθήκες.

Activity

Μια **δραστηριότητα** (**Activity**) είναι ένα απλό, συγκεκριμένο πράγμα που μπορεί κάποιος χρήστης να κάνει. Σχεδόν όλες οι δραστηριότητες αλληλεπιδρούν με το χρήστη. Η κλάση Activity φροντίζει για τη δημιουργία ενός παραθύρου όπου μπορούν να τοποθετηθούν αντικείμενα για τη δημιουργία γραφικού περιβάλλοντος χρησιμοποιώντας τη μέθοδο **setContentView(View)**. Ενώ οι δραστηριότητες συχνά παρουσιάζονται στο χρήστη ως παράθυρα σε πλήρη οθόνη, μπορούν επίσης να χρησιμοποιηθούν με άλλους τρόπους, όπως ως αναδυόμενα παράθυρα ή και ενσωματωμένες δραστηριότητες μέσα σε άλλες δραστηριότητες. Υπάρχουν δύο μέθοδοι που σχεδόν όλες οι υποκατηγορίες της δραστηριότητας θα εφαρμόσουν και είναι οι :

- **onCreate(Bundle)** όπου αρχικοποιείται η δραστηριότητα. Το πιο σημαντικό είναι ότι εδώ καλείται συνήθως η μέθοδος setContentView() που ορίζει όλο το γραφικό περιβάλλον χρησιμοποιώντας τη μέθοδο findViewById(int) την ανάκτηση των widgets που βρίσκονται στο γραφικό περιβάλλον και τη δυνατότητα αλληλεπίδρασης με αυτά προγραμματιστικά.
- **onPause()** όπου εκτελούνται οι εντολές όταν ο χρήστης φύγει από τη δραστηριότητα.

Intent

Ένα **Intent** είναι μία αφηρημένη περιγραφή μιας λειτουργίας που μπορεί να πραγματοποιηθεί. Μπορεί να χρησιμοποιηθεί με τη μέθοδο **startActivity** για εκκίνηση μιας νέας δραστηριότητας, ως broadcastIntent για την αποστολή δεδομένων σε ένα **BroadcastReceiver** και επίσης ως **startService(Intent)** ή **bindService(Intent)** για την επικοινωνία με μία υπηρεσία που εκτελείται στο παρασκήνιο. Η σημαντικότερη χρήση του είναι κατά την έναρξη των δραστηριοτήτων και μπορεί να θεωρηθεί ως συνδετικός κρίκος μεταξύ των δραστηριοτήτων. Πρόκειται ουσιαστικά για μια παθητική δομή δεδομένων που περιέχει μια αφηρημένη περιγραφή της δράσης που πρέπει να εκτελεστεί.

PendingIntent

Αντικείμενα αυτής της κλάσης δημιουργούνται με τις μεθόδους g**etActivity(Context, int, Intent, int)**, **getActivities(Context, int, Intent[], int)**, **getBroadcast(Context, int, Intent, int)**, και **getService(Context, int, Intent, int)**. Το αντικείμενο που επιστρέφεται μπορεί να δοθεί σε άλλες εφαρμογές για να εκτελέσουν ενέργειες που θα περιγραφούν παρακάτω.

Δίνοντας ένα **PendingIntent** σε μια άλλη εφαρμογή, της επιτρέπουμε να εκτελέσει την καθορισμένη ενέργεια σαν να την εκτελούσαμε εμείς οι ίδιοι (ίδια δικαιώματα και ταυτότητα). Ως εκ τούτου, θα πρέπει να είμαστε προσεκτικοί σχετικά με το πως θα οικοδομήσουμε ένα **PendingIntent** και πάντα θα πρέπει να περιέχει το αποκλειστικό όνομα του στοιχείου που θέλουμε να επεξεργαστεί για να είμαστε σίγουροι ότι θα πάει σ αυτό και όχι κάπου αλλού.

Το ίδιο το **PendingIntent** είναι ουσιαστικά μία αναφορά σε ένα σύμβολο που διατηρεί το σύστημα και περιγράφει τα αρχικά δεδομένα που θα χρησιμοποιηθούν για να μπορέσουμε να το ανακτήσουμε. Αυτό σημαίνει ότι ακόμα και όταν τερματιστεί η διεργασία της εφαρμογής που το δημιούργησε, αυτό μένει ακόμα ενεργό για χρησιμοποίηση από τις άλλες διεργασίες και εφαρμογές στις οποίες έχει δοθεί. Αν αργότερα η εφαρμογή που το δημιούργησε λάβει ξανά ίδιου είδους **PendingIntent** (ίδια εργασία, ενέργειες, δεδομένα, κατηγορία, συστατικά και σημαίες (flags)), τότε θα λάβει ένα **PendingIntent** με αναφορά στο ίδιο σύμβολο που αναφέραμε πριν και αν είναι ενεργό μπορεί να καλέσει τη μέθοδο **cancel()** για να το τερματίσει.

Εάν χρειαζόμαστε μόνο ένα PendingIntent ενεργό κάθε φορά τότε μπορούμε να χρησιμοποιήσουμε τα flags ***FLAG_CANCEL_CURRENT*** και ***FLAG_UPDATE_CURRENT*** για να ακυρώσουμε ή να ενημερώσουμε το παρόν ενεργοποιημένο Intent.

AlarmManager

Αυτή η κλάση παρέχει πρόσβαση στις υπηρεσίες **alarm** του συστήματος. Αυτό μας επιτρέπει να τρέξουμε διεργασίες της εφαρμογής μας ή και την εφαρμογή αυτή καθαυτή σε κάποιο σημείο στο μέλλον. Όταν το alarm ενεργοποιείται, το intent που έχει καταχωρηθεί μεταδίδεται αυτόματα από το σύστημα και ενεργοποιεί την αντίστοιχη διεργασία ή εφαρμογή αν αυτή δεν εκτελείται ήδη. Τα καταχωρημένα alarm διατηρούνται ακόμα και όταν η συσκευή μπαίνει σε κατάσταση sleep και μπορούν (προαιρετικά) να εκτελεστούν ακόμη και κατά την επανεκκίνηση της συσκευής, αλλά θα διαγραφούν αν είναι απενεργοποιημένοι κατά την απενεργοποίησή της.

NotificationManager

Η κλάση αυτή χρησιμοποιείται για την ενημέρωση του χρήστη αναφορικά με τα γεγονότα που συμβαίνουν. Έτσι ο χρήστης μαθαίνει αυτά που συμβαίνουν στο παρασκήνιο.

Ειδοποιήσεις μπορούν να λάβουν διάφορες μορφές:

- Ένα εικονίδιο που εμφανίζεται στη γραμμή κατάστασης και είναι προσβάσιμο μέσω του launcher (όταν ο χρήστης επιλέγει ένα καθορισμένο intent προς εκτέλεση).
- Ενεργοποίηση των LED που αναβοσβήνουν στη συσκευή.
- Προειδοποίηση του χρήση ανάβοντας το οπίσθιο φως της συσκευής ή παίζοντας κάποιο ήχο ή και μέσω δόνησης.

Κάθε μία από τις μεθόδους που ειδοποιεί τον χρήστη παίρνει μια παράμετρο id (int) και προαιρετικά μια παράμετρο συμβολοσειράς, την ετικέτα, η οποία μπορεί να είναι και null. Αυτές οι παράμετροι χρησιμοποιούνται για να σχηματίσουν ένα ζεύγος (ετικέτα, id), ή (null,

id) αν ετικέτα είναι απροσδιόριστη. Από αυτό το ζεύγος αναγνωρίζεται αυτή η ειδοποίηση της εφαρμογής από το σύστημα, ως εκ τούτου θα πρέπει να είναι μοναδικό στην εφαρμογή. Αν καλέσουμε μια από τις μεθόδους ειδοποίησης με ένα ζεύγος (ετικέτα, id) που είναι ενεργό με ένα νέο σύνολο παραμέτρων, το παρόν ζεύγος θα ενημερώνεται. Για παράδειγμα, αν έχουμε περάσει ένα νέο εικονίδιο στη γραμμή κατάστασης, το παλιό εικονίδιο στη θα αντικατασταθεί με το νέο. Τέλος μπορούμε να ακυρώσουμε την ειδοποίηση μέσω της μεθόδου **cancel(int)**.

BroadcastReceiver

Βασική κλάση που περιέχει τον κώδικα για να μπορούμε να διαχειριζόμαστε intents που αποστέλλονται από τη μέθοδο **sendBroadcast()**.

Μπορούμε είτε να εγγράψουμε δυναμικά ένα αντικείμενο αυτής της κλάσης μέσω της μεθόδου **Context.registerReceiver()** ή στατικά μέσω προσθήκης της ετικέτας **<receiver>** στο ***AndroidManifest.xml*** αρχείο της εφαρμογής. Αν καταχωρήσουμε ένα δέκτη στη **Activity.onResume()** μέθοδο της εφαρμογής, θα πρέπει να τον καταργήσουμε στην μέθοδο **Activity.onPause()**.

Υπάρχουν δύο κύριες κατηγορίες από **broadcasts** που μπορούν να λαμβάνουν :

- **Normal broadcasts** αποστέλλονται με **Context.sendBroadcast** και είναι εντελώς ασύγχρονοι. Όλοι οι δέκτες των broadcasts τρέχουν με απροσδιόριστη σειρά και συχνά κατά τον ίδιο χρόνο. Αυτό είναι αποδοτικό, αλλά σημαίνει ότι οι δέκτες δεν μπορούν να χρησιμοποιήσουν το αποτέλεσμα ή τα APIs που περιλαμβάνονται εδώ.
- **Ordered broadcasts** αποστέλλονται με **Context.sendOrderedBroadcast** και παραδίδονται σε ένα δέκτη τη φορά. Κάθε δέκτης εκτελεί τις ενέργειές του σειριακά και μπορεί να στείλει τα αποτελέσματά του στον επόμενο δέκτη ή να διακόψει την εκπομπή έτσι ώστε να μη συνεχιστεί στους άλλους δέκτες. Η σειρά εκτέλεσης των

δεκτών μπορεί και καθοριστεί και από το χαρακτηριστικό **android:priority** σε συνδυασμό με κάποιο **intent-filter**. Δέκτες που έχουν την ίδια προτεραιότητα εκτελούνται σε τυχαία σειρά.

Ακόμη και στην περίπτωση των κανονικών μεταδόσεων (normal broadcasts), το σύστημα μπορεί, σε ορισμένες περιπτώσεις να μεταδίδει σε ένα δέκτη τη φορά. Ειδικότερα για τους δέκτες αυτούς, μπορεί να αιτήσουμε τη δημιουργία διαδικασιών που όμως μόνο μία από αυτές θα πρέπει να τρέχει τη κάθε φορά για να αποφύγουμε την υπερφόρτωση του συστήματος με νέες διαδικασίες.

LocationManager

Η κλάση αυτή παρέχει πρόσβαση στις υπηρεσίες εντοπισμού του συστήματος. Οι υπηρεσίες αυτές επιτρέπουν στις εφαρμογές να αποκτήσουν περιοδικές ενημερώσεις της γεωγραφικής θέσης της συσκευής ή να εκτελέσουν ένα intent όταν η συσκευή εισέλθει στην εγγύτητα μιας δεδομένης της γεωγραφικής της θέσης.

Όλες οι υπηρεσίες εντοπισμού χρειάζονται δικαιώματα ***ACCESS_COARSE_LOCATION*** και ***ACCESS_FINE_LOCATION*** από τον χρήστη.

AsyncTask

Η **AsyncTask** επιτρέπει την σωστή και εύκολη χρήση UI threads. Η κλάση αυτή μας επιτρέπει να εκτελέσουμε λειτουργίες στο παρασκήνιο και να δημοσιεύσουμε τα αποτελέσματα για το UI thread. Έχει σχεδιαστεί για να είναι μια κλάση βοηθός γύρω από **Threads** και **Handlers** και δεν αποτελεί ένα γενικό threading framework. Ιδανικά θα πρέπει να χρησιμοποιείται για να επιτελεί σύντομες εργασίες (λίγα δευτερόλεπτα το πολύ). Αν θα

πρέπει να κρατήσει thread που τρέχουν για μεγάλο χρονικό διάστημα, συνιστάται ιδιαίτερα να χρησιμοποιήσουμε τα διάφορα APIs που παρέχονται από το πακέτο **java.util.concurrent** όπως ο **Executor**, ο **ThreadPoolExecutor** και η **FutureTask**. Μια ασύγχρονη διεργασία ορίζεται από ένα thread που εκτελείται στο παρασκήνιο και τα αποτελέσματά του δημοσιεύονται στο UI thread.

Γενικοί τύποι AsyncTask

Οι τρεις τύποι που χρησιμοποιούνται από μια ασύγχρονη διεργασία είναι οι ακόλουθοι :

1. **Params** : ο τύπος των παραμέτρων που αποστέλλονται στη διεργασία κατά την εκτέλεση.
2. **Progress** : ο τύπος των δεδομένων που δημιουργούνται κατά τη διάρκεια των υπολογισμών που γίνονται στο παρασκήνιο.
3. **Result** : ο τύπος των αποτελεσμάτων που προκύπτουν από τους υπολογισμούς που γίνονται στο παρασκήνιο.

Τα 4 βήματα

Όταν εκτελείται μια ασύγχρονη εργασία, περνά από 4 βήματα :

1. **onPreExecute()** : εκτελείται στο UI thread πριν εκτελεστεί η εργασία. Αυτό το βήμα συνήθως χρησιμοποιείται για να αρχικοποιήσουμε τις διάφορες ενέργειες της εργασίας, για παράδειγμα δείχνοντας μια γραμμή προόδου στο περιβάλλον εργασίας χρήστη.

2. **doInBackground(Params…)** : εκτελείται στο παρασκήνιο μόλις ολοκληρωθεί η εκτέλεση της onPreExecute(). Αυτό το βήμα χρησιμοποιείται για την εκτέλεση των υπολογισμών που γίνονται στο παρασκήνιο και μπορεί να πάρει πολύ χρόνο. Οι παράμετροι της ασύγχρονης εργασίας χρησιμοποιούνται σε αυτό το βήμα. Τα αποτελέσματα των υπολογισμών επιστρέφονται από αυτό το βήμα και περνούν στο τελευταίο βήμα. Αυτό το βήμα μπορεί επίσης να χρησιμοποιήσει τη μέθοδο publishProgress(Progress ...) για να δημοσιεύσει μια ή περισσότερες τιμές της προόδου. Οι τιμές αυτές δημοσιεύονται στο UI thread από τη μέθοδο onProgressUpdate(Progress ...).

3. **onProgressUpdate(Progress…)** : εκτελείται στο UI thread μετά από την κλήση της μεθόδου publishProgress(Progress …). Η χρονική στιγμή της εκτέλεσής είναι απροσδιόριστη. Αυτή η μέθοδος χρησιμοποιείται για να εμφανίσει οποιαδήποτε μορφή προόδου στο περιβάλλον εργασίας χρήστη, ενώ ο υπολογισμός στο παρασκήνιο συνεχίζει να εκτελείται. Για παράδειγμα, μπορεί να χρησιμοποιηθεί για να εμφανιστεί μια γραμμή προόδου ή χαρακτήρες σε ένα πεδίο κειμένου.

4. **onPostExecute(Result)** : εκτελείται μόλις ολοκληρωθούν οι υπολογισμοί από τη μέθοδο doInBackground(Params…). Το αποτελέσματα περνιούνται ως παράμετροι και επιστρέφονται.

Ακύρωση μιας εργασίας

Μια εργασία μπορεί να ακυρωθεί ανα πάσα στιγμή καλώντας τη μέθοδο **cancel(boolean)**. Η κλήση αυτής της μεθόδου θα προκαλέσει και την αλλαγή της τιμής επιστροφής της μεθόδου **isCancelled()** σε αλήθεια (true). Μετά θα εκτελεστεί η μέθοδος **onCancelled(Object)**, αντί για την **onPostExecute(Object)**. Για να εξασφαλιστεί ότι μια εργασία έχει ακυρωθεί το συντομότερο δυνατό, θα πρέπει να ελέγχουμε πάντα την τιμή επιστροφής της **isCancelled()** σε τακτά χρονικά διαστήματα στη **doInBackground (Object [])**, αν είναι δυνατόν ακόμη και μέσα σε ένα βρόχο.

Κανόνες των threads

Υπάρχουν μερικοί κανόνες των threads που πρέπει να ακολουθηθούν για να λειτουργήσει σωστά η εργασία μας :

- Η **AsyncTask** πρέπει να φορτωθεί στο UI thread.
- Το αντικείμενο της εργασίας πρέπει να φορτωθεί στο UI thread.
- **execute(Params...)** πρέπει να περιλαμβάνονται στο UI thread.
- Δε πρέπει να καλούνται οι μέθοδοι **onPreExecute()**, **onPostExecute(Result)**, **doInBackground(Params…)** και **onProgressUpdate(Progress…)** χειροκίνητα.
- Η εργασία μπορεί να εκτελεστεί μόνο μία φορά.

AlarmClock

Η κλάση αυτή μας δίνει τη δυνατότητα μέσω intents και extras που χρησιμοποιούν οι activities να ορίσουμε ένα καινούργιο ξυπνητήρι ή χρονόμετρο στη εφαρμογή alarm clock της συσκευής. Οι εφαρμογές που θέλουν να λαμβάνουν ***ACTION_SET_ALARM*** και ***ACTION_SET_TIMER*** intents θα πρέπει να δημιουργήσουν μία δραστηριότητα για να χειρίζονται αυτά τα intents και επίσης πρέπει να τους χορηγηθούν τα δικαιώματα ***com.android.alarm.permission.SET_ALARM*** *και* ***com.android.alarm.permission.SET_TIMER***. Οι εφαρμογές που θέλουν να δημιουργήσουν ένα νέο ξυπνητήρι ή χρονόμετρο θα πρέπει να χρησιμοποιήσουν τη μέθοδο **Context.startActivity()** έτσι ώστε ο χρήστης να έχει τη δυνατότητα να επιλέξει το ξυπνητήρι ή χρονόμετρο που επιθυμεί.

ACTION_SET_ALARM

Ενέργεια : Ορισμός ξυπνητηριού

Ενεργοποιεί ένα υπάρχον ξυπνητήρι ή δημιουργεί καινούριο.

Παράμετροι

- **EXTRA_HOUR** : Η τιμή της ώρας που θα οριστεί το ξυπνητήρι.
- **EXTRA_MINUTES** : Η τιμή των λεπτών που θα οριστεί το ξυπνητήρι.
- **EXTRA_DAYS** : Η μέρες που θα επαναλαμβάνεται.
- **EXTRA_MESSAGE** : Το μήνυμα που θα εμφανίζεται όταν θα ενεργοποιηθεί.
- **EXTRA_RINGTONE** : Ο ήχος που θα παίζει.
- **EXTRA_VIBRATE** : Το αν θα δονείται ή όχι η συσκευή.
- **EXTRA_SKIP_UI** : Το αν θα εμφανίζεται γραφικό περιβάλλον κατά την ενεργοποίηση του.

ACTION_SET_TIMER

Ενέργεια : Ορισμός χρονομέτρου

Ενεργοποιεί ένα υπάρχον χρονόμετρο ή δημιουργεί καινούριο.

Παράμετροι

- **EXTRA_LENGTH** : Η τιμή του χρόνου που θα οριστεί το χρονόμετρο.
- **EXTRA_MESSAGE** : Το μήνυμα που θα εμφανίζεται όταν θα ενεργοποιηθεί.

- **EXTRA_SKIP_UI** : Το αν θα εμφανίζεται γραφικό περιβάλλον κατά την ενεργοποίηση του.

GoogleMap

Αυτή είναι η κύρια κλάση του Google Maps Android API και είναι το σημείο εισόδου για όλες τις μεθόδους που σχετίζονται με το χάρτη. Δεν μπορούμε να δημιουργήσουμε ένα αντικείμενο GoogleMap άμεσα, θα πρέπει να αποκτήσουμε ένα από τη μέθοδο **GetMap()** σε **MapFragment** ή **MapView** που θα έχουν προστεθεί στην εφαρμογή μας.

Το παρόν API είναι το Google Maps Android API v2. Πρέπει να έχουμε προσθέσει τις απαραίτητες υπηρεσίες στη εφαρμογή μας **(Google Play services)** για να μπορούμε να χρησιμοποιήσουμε το Google Map.

Επίσης θα πρέπει να χορηγηθούν από τον χρήστη τα παρακάτω δικαιώματα :

- **android.permission.INTERNET** : χρησιμοποιείται για την λήψη των χαρτών από τους servers της Google.
- **android.permission.ACCESS_NETWORK_STATE** : επιτρέπει τον έλεγχο και την κατάστασης της σύνδεσης για να εκτιμήσει πότε μπορούν να κατέβουν τα δεδομένα.
- **android.permission.WRITE_EXTERNAL_STORAGE** : επιτρέπει την αποθήκευση των χαρτών σε εξωτερικό μέσο αποθήκευσης.

Τα παρακάτω δικαιώματα συνίστανται, αλλά μπορούν και να αγνοηθούν αν η εφαρμογή δε χρειάζεται πρόσβαση στη γεωγραφική θέση του χρήστη.

- **android.permission.ACCESS_COARSE_LOCATION** : επιτρέπει τη χρήση WiFi ή Mobile Cell Data για να προσδιορίσει τη γεωγραφική θέση της συσκευής.

- **android.permission.ACCESS_FINE_LOCATION** : επιτρέπει τη χρήση Global Positioning System (GPS) για να προσδιορίσει τη γεωγραφική θέση της συσκευής μέσα σε μικρή περιοχή.

ΚΕΦΑΛΑΙΟ 6

ΑΞΙΟΛΟΓΗΣΗ ΕΠΙΔΟΣΕΩΝ

Έχουμε ανεβάσει την υπηρεσία **RAC** στο σύννεφο χρησιμοποιώντας μία εικονική μηχανή με δύο πυρήνες και 6 GB μνήμης RAM, σύμφωνα με την οποία κάθε πυρήνας είχε τις ακόλουθες προδιαγραφές:

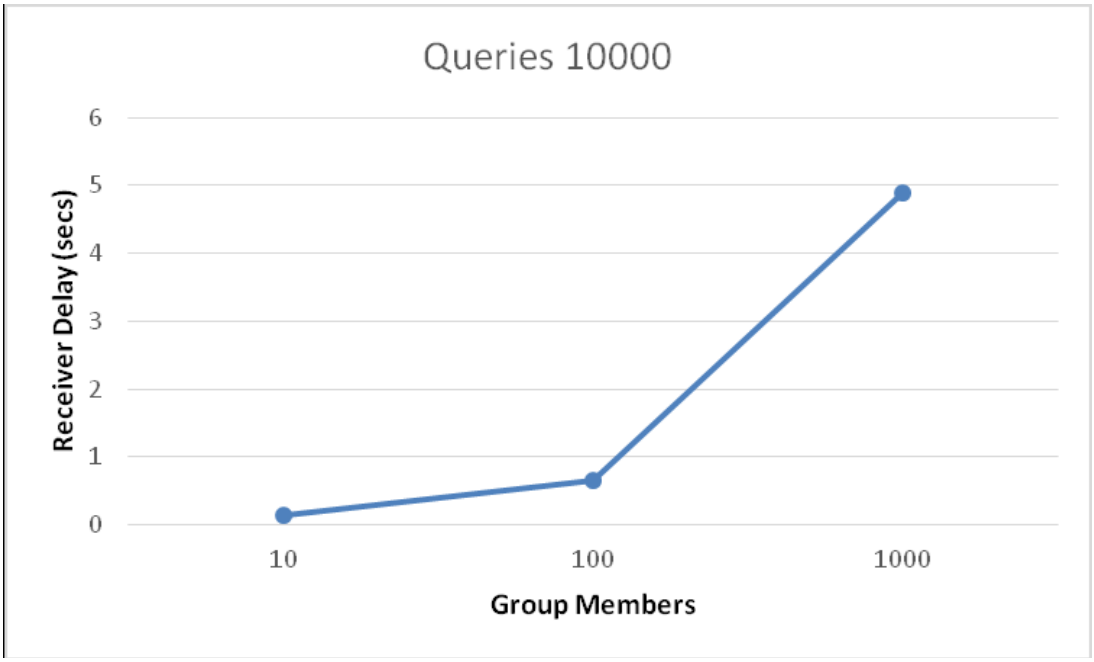
```
vendor_id       : AuthenticAMD
cpu family      : 6
model           : 6
model name      : QEMU Virtual CPU version 2.1.2
stepping        : 3
microcode       : 0x1000065
cpu MHz         : 2100.040
cache size      : 512 KB
```

Στα πειράματα που αποσκοπούσαν στην αξιολόγηση της απόδοσης της αποστολής μηνυμάτων στην υπηρεσία RAC, λόγω της έλλειψης ενός αρκετά μεγάλου συνόλου πραγματικών χρηστών, αποφασίσαμε να προσομοιώσουμε δραστηριότητες χρηστών χρησιμοποιώντας ένα φορητό υπολογιστή με δύο διεργασίες που η μία εκτελούσε την αποστολή των αιτήσεων και η άλλη την ανάκτηση των αιτήσεων από την υπηρεσία. Τα χαρακτηριστικά του laptop συνοψίζονται παρακάτω:

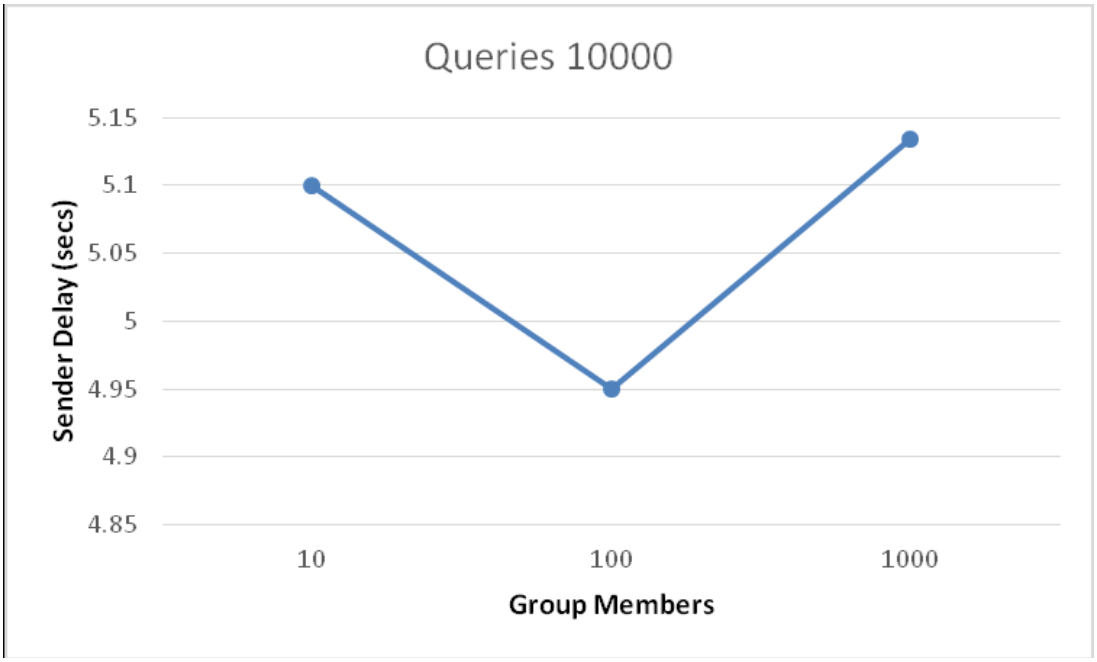
```
System Manufacturer Dell Inc.
System Model Inspiron 5545
System Type x64-based PC
System SKU Inspiron 5545
Processor AMD A10-7300 Radeon R6, 10 Compute Cores 4C+6G, 1900 MHz, 4 Core(s), 4 Logical Processor(s)
Memory (RAM) 8.00 GB DDR3
```

Graphics Card AMD Radeon HD R7 M265 2048MB DDR3

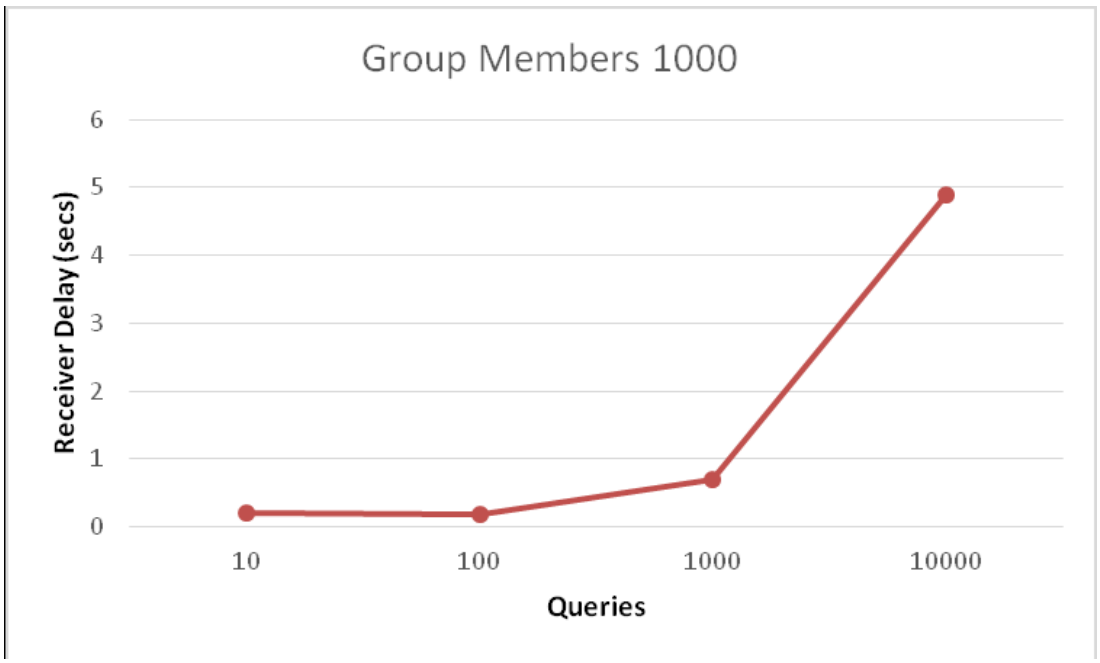
Οργανώσαμε τα πειράματα ως εξής. Ελέγξαμε την επικοινωνία της ομάδας, εξετάζοντας την περίπτωση όπου υπάρχει μια ομάδα και η διαδικασία αποστολής (πιθανώς ένα μέλος της ομάδας, στην πραγματικότητα) στέλνει μήνυμα χρησιμοποιώντας MDBF περιλαμβάνοντας τη συμμετοχή όλων των μελών της ομάδας. Ο χρόνος λήξης τέθηκε έτσι ώστε να είναι επαρκώς μεγάλος για να εξασφαλιστεί η παράδοση των μηνυμάτων και η περίοδος λήψης ορίστηκε σε 1 δευτερόλεπτο για κάθε μέλος της ομάδας.



ΔΙΑΓΡΑΜΜΑ 6.1 : ΜΕΣΗ ΚΑΘΥΣΤΕΡΗΣΗ ΕΡΩΤΗΜΑΤΟΣ ΣΤΟ ΔΕΚΤΗ ΟΣΟ ΑΥΞΑΝΕΤΑΙ Ο ΑΡΙΘΜΟΣ ΤΩΝ ΜΕΛΩΝ ΜΙΑΣ ΟΜΑΔΑΣ ΧΡΗΣΤΩΝ



ΔΙΑΓΡΑΜΜΑ 6.2 : ΜΕΣΗ ΚΑΘΥΣΤΕΡΗΣΗ ΕΡΩΤΗΜΑΤΟΣ ΣΤΟΝ ΑΠΟΣΤΟΛΕΑ ΟΣΟ ΑΥΞΑΝΕΤΑΙ Ο ΑΡΙΘΜΟΣ ΤΩΝ ΜΕΛΩΝ ΜΙΑΣ ΟΜΑΔΑΣ ΧΡΗΣΤΩΝ



ΔΙΑΓΡΑΜΜΑ 6.3 : ΜΕΣΗ ΚΑΘΥΣΤΕΡΗΣΗ ΕΡΩΤΗΜΑΤΟΣ ΣΤΟ ΔΕΚΤΗ ΟΣΟ ΑΥΞΑΝΕΤΑΙ Ο ΑΡΙΘΜΟΣ ΤΩΝ ΕΡΩΤΗΜΑΤΩΝ

Έχουμε πειραματιστεί με διάφορα μεγέθη ομάδων χρηστών. Η διεργασία αποστολέα ορίστηκε για να στέλνει ταυτόχρονα πολλαπλά ερωτήματα για διαφορετικά μεγέθη πλήθους ερωτημάτων. Μετρήσαμε τόσο το χρόνο που χρειάστηκε ο διεργασία αποστολέα για να στείλει τα ερωτήματα, όσο και τον χρόνο λήψης των ερωτημάτων.

Τα παραπάνω διαγράμματα συνοψίζουν τα ευρήματά μας. Από τα αποτελέσματα παρατηρούμε ότι :

i. η διεργασία αποστολέα οριακά επηρεάζεται από το φορτίο των ερωτημάτων που αποστέλλεται

ii. η καθυστέρηση ανάκτησης μηνυμάτων αυξάνει όσο αυξάνει το φορτίο των ερωτημάτων

iii. ακόμη και στην περίπτωση των 10 εκατομμυρίων ταυτόχρονων ερωτημάτων (τρίτο σημείο στα διαγράμματα) η μέση καθυστέρηση δεν υπερέβη ποτέ τα 5-6 δευτερόλεπτα.

ΚΕΦΑΛΑΙΟ 7

ΜΕΛΛΟΝΤΙΚΕΣ ΕΠΕΚΤΑΣΕΙΣ - ΣΥΜΠΕΡΑΣΜΑΤΑ

Στα πλαίσια της παρούσας διπλωματικής εργασία υλοποιήθηκε ένα σύστημα για τη διαχείριση εφαρμογών που είναι εγκατεστημένες σε κινητές συσκευές. Οι εφαρμογές που επιλέχθηκαν ήταν το Alarm Clock και το Maps για την εισαγωγή νέου ξυπνητηριού και σημείου συνάντησης στο χάρτη αντίστοιχα.

Μελλοντικά στοχεύουμε στη βελτίωση αυτού του συστήματος έτσι ώστε να υπάρχει η δυνατότητα δημιουργίας λογαριασμού χρήστη και ομάδων χρηστών καθώς και αποστολή ενεργειών σε αυτές τις ομάδες. Επίσης θα μπορούμε να υποστηρίζουμε πολλαπλές συσκευές για τον ίδιο χρήστη. Σημαντική επέκταση θα είναι η δημιουργία ενός framework που θα δίνει τη δυνατότητα σε μία ομάδα από χρήστες να αποφασίσουν από κοινού για κάτι. Για παράδειγμα τώρα ένας χρήστης μπορεί να στείλει ένα σημείο συνάντησης σε έναν άλλο. Με το παραπάνω framework θα μπορούν να προτείνουν πολλοί διάφορα σημεία συνάντησης και να αποφασίσουν από κοινού για το τελικό σημείο.

Επιπλέον στοχεύουμε στη δημιουργία μιας διαδικτυακής υπηρεσίας για την εκχώρηση του API των εφαρμογών δημιουργώντας έτσι μια βάση δεδομένων για να μπορούν να ελέγχονται αυτόματα όλες αυτές οι εφαρμογές της συσκευής. Η υλοποίηση μιας τέτοιας επέκτασης πιστεύουμε ότι θα είναι εξαιρετικά χρήσιμη σε πολλούς τομείς της καθημερινότητας και όχι μονό. Για παράδειγμα θα μπορούσε να εφαρμοστεί στον τομέα της υγείας ως εφαρμογή απομακρυσμένη υπενθύμισης πρόσληψης φαρμάκων ατόμων προχωρημένης ηλικίας. Και αυτό θα γίνεται διαχειρίζοντας μια άλλη υπάρχουσα εφαρμογή σχετική με αυτό τον τομέα η οποία θα έχει εγγραφεί στην υπηρεσία και θα μας έχει δώσει το API της.

Τέλος στοχεύουμε και στη δημιουργία μιας ξεχωριστής εφαρμογής που θα διευκολύνει τους προγραμματιστές που θέλουν να επιτρέψουν στην εφαρμογής τους να είναι διαχειρίσιμη από απόσταση.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Android. http://el.wikipedia.org/wiki/Android

[2] SQL. http://el.wikipedia.org/wiki/SQL

[3] PHP. http://en.wikipedia.org/wiki/PHP

[4] WebService. http://en.wikipedia.org/wiki/Web_service

[5] JSON. http://en.wikipedia.org/wiki/JSON

[6] MySQL. http://en.wikipedia.org/wiki/MySQL

[7] MySQLi. http://en.wikipedia.org/wiki/MySQLi

[8] RPC. http://en.wikipedia.org/wiki/Remote_procedure_call

[9] JRMI. http://en.wikipedia.org/wiki/Java_remote_method_invocation

[10] SQL.Tutorial. http://www.w3schools.com/sql/default.asp

[11] PHP.Tutorial. http://www.w3schools.com/php/default.asp

[12] MySQLi.Manual. http://php.net/manual/en/mysqli.overview.php

[13] MySQLi.Manual. http://php.net/manual/en/book.mysqli.php

[14] MySQL. http://dev.mysql.com/doc/

[15] Android.Manual. https://developer.android.com/training/index.html

[16] RAC demo available at: https://dl.dropboxusercontent.com/u/65278128/RAC.apk

[17] P. Papadopoulos, Th. Loukopoulos, I. Anagnostopoulos, N. Tziritas, M. Vassilakopoulos. RAC: A Remote Application Calling Framework for Coordination of Mobile Apps. In *Proceedings of PCI Conference*. Oct 2015 (To appear)

[18] Adya, A., Cooper, G., Myers, D., and Piatek, M. 2011. Thialfi: a client notification service for internet-scale applications. In *Proceedings of the 23rd ACM Symposium on Operating Systems Principles* (Cascais, Portugal, Oct. 2011). SOSP'11. 129-142.

[19] Agarwal, S., Mahajan, R., Zheng, A., and Bahl, V. 2010. Diagnosing mobile applications in the wild. *In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks* (Monterey, CA, USA, Oct. 2010). Hotnets-IX. 1-6.

[20] Any.Do. http://www.any.do/anydo

[21] Anywhere Pad. http://www.azeusconvene.com/

[22] Kolbeck, B., Högqvist, M., Stender, J., and Hupfeld, F. 2011. Flease - Lease Coordination without a Lock Server. In *Proceedings of the 25th IEEE International Parallel & Distributed Processing Symposium* (Anchorage, Alaska, USA, May 2011). IPDPS 2011. 978-988.

[23] CAL. http://www.any.do/cal

[24] Chun, B.G., Curino, C., Sears, R., Shraer, A., Madden, S., and Ramakrishnan, R. 2012. Mobius: unified messaging and data serving for mobile apps. In *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services* (Low Wood Bay, Lake District, UK, June 2012). MobiSys'12. 141—154.

[25] Dropbox Inc. https://www.dropbox.com/

[26] Go, Y., Agrawal, N., Aranya, A., and Ungureanu, C. 2015. Reliable, consistent, and efficient data sync for mobile apps. In *Proceedings of the 13th USENIX Conf. on File and Storage Technologies* (Santa Clara, CA, USA, Feb. 2015). FAST'15. 359-372.

[27] Google Drive. https://www.google.com/drive/

[28] Google Keep. http://www.google.com/keep/

[29] IF. https://ifttt.com/

[30] Insight. http://mobileinsight.com/

[31] Jehl, L., and Meling, H. 2013. Towards byzantine fault tolerant publish/subscribe: a state machine approach. In *Proceedings of the 9th Workshop on Hot Topics in Dependable Systems* (Farmington, Pennsylvania, Nov. 2015). HotDep'13. 1-5.

[32] join.me. https://www.join.me/

[33] Lamport, L. (2001). Paxos made simple. *SIGACT News.* 32, 4 (Dec. 2001), 18–25.

[34] Lloyd, W., Freedman, M. J., Kaminsky, M., and Andersen, D. G. A short primer on causal consistency. *USENIX ;login magazine*. 38, 4 (Aug. 2013), 41-43.

[35] Perkins, D., Agrawal, N., Aranya, A., Yu, C., Go, Y., Madhyastha, H., and Ungureanu, C. 2015. Simba: tunable end-to-end data consistency for mobile apps. In *Proceedings of the European Conference on Computer Systems* (Bordeaux, France, April 2015). EuroSys '15.

[36] Quip. https://quip.com/

[37] SAP Streamwork. http://www.sap.com/pc/analytics/business-intelligence/software/streamwork/overview/index.html

[38] Sharma, Y., Ajoux, P., Ang, P., Callies, D., Choudhary, A., Demailly, L., Fersch, T., Guz, L.A., Kotulski, A., Kulkarni, S., Kumar, S., Li, H., Li, J., Makeev, E., Prakasam, K., Renesse, R., Roy, S., Seth, P., Song, Y.J., Wester, B., Veeraraghavan, K., and Xie, P. 2015. Wormhole: reliable pub-sub to support geo-replicated internet services. In *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation* (Oakland, CA, USA, May 2015). NSDI'15. 351-366.

[39] Studiopass. http://studiopass.com/

[40] Tolia, N., Satyanarayanan, M., and Wolbach, A. 2007. Improving mobile database access over wide-area networks without degrading consistency. In *Proceedings of the 5th International Conference on Mobile Systems Applications and Services* (San Juan, Puerto Rico, June 2007). MobiSys'07. 71-84.

[41] WebEx. http://www.webex.com/

ΠΑΡΑΡΤΗΜΑ Α : CLIENT

A.1 AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ppapadop.rac" >

    <permission
        android:name="com.ppapadop.rac.permission.MAP_RECEIVE"
        android:protectionLevel="signature" />

    <uses-permission android:name="com.ppapadop.rac.permission.MAP_RECEIVE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
         android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="com.android.alarm.permission.SET_ALARM" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />

    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/icon"
        android:label="@string/app_name"
        android:theme="@android:style/Theme.Holo.NoActionBar.Fullscreen" >
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="@string/google_maps_key" />

        <receiver android:name=".ActionsReceiver" >
        </receiver>

        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
```

```
        <activity
            android:name=".SetAlarmActivity"
            android:label="@string/title_activity_set_alarm" >
        </activity>
        <activity
            android:name=".TrustedDevicesActivity"
            android:label="@string/title_activity_trusted_devices" >
        </activity>
        <activity
            android:name=".ConnectedDevicesActivity"
            android:label="@string/title_activity_connected_devices" >
        </activity>
        <activity
            android:name=".ActionsActivity"
            android:label="@string/title_activity_actions" >
        </activity>
        <activity
            android:name=".ControlDeviceActivity"
            android:label="@string/title_activity_add_device" >
        </activity>
        <activity
            android:name=".SetMapActivity"
            android:label="@string/title_activity_set_map" >
        </activity>
        <activity
            android:name=".SettingsActivity"
            android:label="@string/title_activity_settings" >
        </activity>
    </application>

</manifest>
```

A.2 MainActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.provider.Settings;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.TextView;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
```

```
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.StringTokenizer;

import static android.os.StrictMode.*;


public class MainActivity extends Activity
{
    public static String deviceId = "";
    public static String dbUrl;

    int randomId;
    int interval;
    String sql = null;
    String methodName;
    boolean isRandomIdExists;

    Button applicationsButton;
    Button trustedDevicesButton;
    Button settingsButton;

    TextView deviceIdTextView;

    private String android_id;

    private AlarmManager manager;
    private PendingIntent pendingIntent;

    private ProgressBar spinner;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        // Get the unique device id
        android_id = Settings.Secure.getString(getContentResolver(),
         Settings.Secure.ANDROID_ID);

        spinner = (ProgressBar)findViewById(R.id.spinner);
```

```
        spinner.setVisibility(View.GONE);

        if (manager!=null)
            manager.cancel(pendingIntent);

        // Set database url
        dbUrl = "http://83.212.106.211/";

        interval = 60000;

        ThreadPolicy policy = new ThreadPolicy.Builder().permitAll().build();
        setThreadPolicy(policy);

        applicationsButton = (Button)findViewById(R.id.applicationsButton);
        trustedDevicesButton = (Button)findViewById(R.id.trustedDevicesButton);
        settingsButton = (Button)findViewById(R.id.settingsButton);

        applicationsButton.setBackgroundResource(R.drawable.button);

        deviceIdTextView = (TextView)findViewById(R.id.deviceIdTextView);
    }

    @Override
    public void onResume() {
        super.onResume();

        spinner.setVisibility(View.GONE);

        //Disable buttons
        applicationsButton.setEnabled(false);
        applicationsButton.setBackgroundResource(R.drawable.button_disabled);

        trustedDevicesButton.setEnabled(false);
        trustedDevicesButton.setBackgroundResource(R.drawable.button_disabled);

        deviceIdTextView.setText("Connecting...");

        ConnectivityManager connManager = (ConnectivityManager)
         getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo mWifi = connManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        NetworkInfo mMobile =
         connManager.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);


        if (mWifi.isConnected() || mMobile.isConnected()) {
            if (manager!=null)
                manager.cancel(pendingIntent);

            setDatabaseUrl();
        }
        else {
            deviceIdTextView.setText("NO INTERNET CONNECTION!!!");
        }
    }

    @Override
    public void onPause() {
        super.onPause();
    }

    void runActionReceiver(String dId)
    {
        Intent actionsIntent = new Intent(this, ActionsReceiver.class);
```

```
    Bundle actionsBundle = new Bundle();

    actionsBundle.putString("deviceId", dId);
    actionsBundle.putString("dbUrl", dbUrl);
    actionsIntent.putExtras(actionsBundle);

    pendingIntent = PendingIntent.getBroadcast(this, 0, actionsIntent, 0);

    startActionsReceiver();
}

void startActionsReceiver()
{
    manager = (AlarmManager) getSystemService(Context.ALARM_SERVICE);

    manager.setRepeating(AlarmManager.RTC_WAKEUP, System.currentTimeMillis(),
     interval, pendingIntent);
}

public void trustedDevicesButtonPressed(View v)
{
    spinner.setEnabled(true);

    Intent intent = new Intent(v.getContext(), TrustedDevicesActivity.class);

    startActivity(intent);
}

public void applicationsButtonPressed(View v) {
    spinner.setEnabled(true);

    Intent intent = new Intent(v.getContext(), ApplicationsActivity.class);

    startActivity(intent);
}

public void settingsButtonPressed(View v)
{
    spinner.setEnabled(true);

    Intent intent = new Intent(v.getContext(), SettingsActivity.class);
    Bundle b = new Bundle();

    b.putInt("interval", interval);
    intent.putExtras(b);

    startActivity(intent);
}

void setDatabaseUrl()
{
    sql = "SELECT database_url FROM DatabaseInfo";
    methodName = "downloadDatabaseUrl";

    new MainWebServiceTask().execute(sql, "down_service.php");
}

void downloadDatabaseUrl(String result)
{
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);
```

```
            if (jsonArray.length() > 0) {
                JSONObject jsonObject = jsonArray.getJSONObject(0);

                // Set database url
                dbUrl = jsonObject.getString("database_url");
            }
        } catch (Exception e) {
            Log.e("Main database url json", "Error parsing data " + e.toString());
        }
    }
}

void downloadDeviceIdFromDatabase(String result)
{
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            if (jsonArray.length() > 0) {
                JSONObject jsonObject = jsonArray.getJSONObject(0);

                // Set device id
                deviceId = jsonObject.getString("device_id");
            }
        } catch (Exception e) {
            Log.e("Main json", "Error parsing data " + e.toString());
        }
    }
}

boolean checkIfRandomIdExists(String result)
{
    int i;

    try {
        // Parse json data
        JSONArray jsonArray = new JSONArray(result);

        i=0;
        while(i < jsonArray.length()) {
            JSONObject jsonObject = jsonArray.getJSONObject(i);

            if(Integer.toString(randomId.equals(
             jsonObject.getString("device_id"))) {
                return true;
            }

            i++;
        }

        return false;
    }
    catch (Exception e) {
        Log.e("Main json", "Error parsing data " + e.toString());
    }

    return true;
}

void setDeviceId()
{
```

```
    isRandomIdExists = true;

    sql = "SELECT device_id FROM Devices WHERE android_id='" + android_id + "'";
    methodName = "setDeviceId";

    new MainWebServiceTask().execute(sql, "down_service.php");
}

void createAppInfoFile(String fileName)
{
    BufferedWriter bufferedWriter = null;

    try {
        FileOutputStream fileOutputStream = openFileOutput(
         fileName,Context.MODE_PRIVATE);
        bufferedWriter = new BufferedWriter(new OutputStreamWriter(
         fileOutputStream));

        bufferedWriter.write("INTERVAL~" + interval + "\n");

    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        try {
            if (bufferedWriter != null) {
                bufferedWriter.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

void loadFromAppInfoFile(String filename)
{
    BufferedReader bufferedReader = null;

    StringTokenizer st;

    try {
        FileInputStream fileInputStream = openFileInput(filename);
        bufferedReader = new BufferedReader(new InputStreamReader(
         fileInputStream));

        String line, name;

        while ((line = bufferedReader.readLine()) != null) {
            st = new StringTokenizer(line, "~");

            name = st.nextToken();

            if(name.equals("INTERVAL"))
                interval = Integer.parseInt(st.nextToken());
        }
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        try {
            if (bufferedReader != null) {
                bufferedReader.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
```

```
        }
    }
}

class MainWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        spinner.setVisibility(View.VISIBLE);
    }

    @Override
    protected String doInBackground(String... params)
    {
        sql = params[0];
        phpFile = params[1];

        result = runWebService();

        return null;
    }

    protected void onProgressUpdate(String... progress) {
    }

    @Override
    protected void onPostExecute(String file_url) {
        if (phpFile.equals("down_service.php")) {
            if (methodName.equals("setDeviceId")) {

                downloadDeviceIdFromDatabase(result);

                // Device id does not exists in database
                if (deviceId.equals("")) {
                    while (isRandomIdExists) {
                        randomId = 100 + Math.abs(new Random().nextInt() % 100);

                        isRandomIdExists = checkIfRandomIdExists(result);
                    }

                    if (!isRandomIdExists) {
                        // Set device id
                        deviceId = Integer.toString(randomId);

                        sql = "INSERT INTO Devices (device_id, android_id) VALUES
                         ('" + randomId + "','" + android_id + "')";
                        new MainWebServiceTask().execute(sql, "up_service.php");
                    }
                }

                if (!deviceId.equals("")) {
                    runActionReceiver(deviceId);

                    applicationsButton.setEnabled(true);
                    trustedDevicesButton.setEnabled(true);
```

```
                applicationsButton.setBackgroundResource(R.drawable.button);
                trustedDevicesButton.setBackgroundResource(R.drawable.button);

                deviceIdTextView.setText(deviceId);
            } else
                deviceIdTextView.setText("CONNECTION ERROR!!!");
        }

        if (methodName.equals("downloadDatabaseUrl")) {

            downloadDatabaseUrl(result);

            Log.v("MAIN DBURL AFT", dbUrl);

            // Create app data file if not exists
            File file = new File(getFilesDir(), "AppInfo");

            if(!file.exists())
                createAppInfoFile("AppInfo");
            else
                loadFromAppInfoFile("AppInfo");

            // Set device id
            setDeviceId();
        }
    }

    spinner.setVisibility(View.GONE);
}

String runWebService()
{
    List<NameValuePair> list = new ArrayList<>(1);

    list.add(new BasicNameValuePair("sql", sql));

    // Setting up the connection
    try {
        HttpClient httpClient = new DefaultHttpClient();
        HttpPost httpPost = new HttpPost(dbUrl + phpFile);

        httpPost.setEntity(new UrlEncodedFormEntity(list));

        HttpResponse response = httpClient.execute(httpPost);

        HttpEntity entity = response.getEntity();
        isr = entity.getContent();
    }
    catch(ClientProtocolException e) {
        Log.e("Main ClientProtocol", e.toString());
        e.printStackTrace();

        result = "0";
    }
    catch (IOException e) {
        Log.e("Main IOException", e.toString());
        e.printStackTrace();

        result = "0";
    }

    // Convert the response to string
    try {
```

```
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e) {
                Log.e("Main Convert", e.toString());
            }

            return result;
        }
    }
}
```

A.3 TrustedDevicesActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.graphics.Color;
import android.graphics.Typeface;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.util.Log;
import android.view.Display;
import android.view.Gravity;
import android.view.View;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;
```

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;


public class TrustedDevicesActivity extends Activity implements View.OnClickListener
{
    String deviceId = "";
    String dbUrl;
    LinearLayout linealLayout, labelsLinealLayout;
    Button addDevice;

    ArrayList<Privileges> privileges = new ArrayList<>();

    private ProgressBar spinner;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_trusted_devices);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        deviceId = MainActivity.deviceId;
        dbUrl = MainActivity.dbUrl;

        linealLayout = (LinearLayout)findViewById(R.id.linealLayout);
        labelsLinealLayout = (LinearLayout)findViewById(R.id.labelsLinealLayout);

        addDevice = (Button)findViewById(R.id.addDevice);

        addDevice.setEnabled(false);
        addDevice.setBackgroundResource(R.drawable.button_disabled);

        spinner = (ProgressBar)findViewById(R.id.spinner);
        spinner.setVisibility(View.GONE);
    }

    @Override
    public void onResume() {
        super.onResume();

        privileges = new ArrayList<>();

        ConnectivityManager connManager = (ConnectivityManager)getSystemService(
         Context.CONNECTIVITY_SERVICE);
        NetworkInfo mWifi = connManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        NetworkInfo mMobile = connManager.getNetworkInfo(
         ConnectivityManager.TYPE_MOBILE);

        if (mWifi.isConnected() || mMobile.isConnected())
            updateTrustedDevicesTable();
        else
            Toast.makeText(getApplicationContext(), "NO INTERNET CONNECTION!!!",
             Toast.LENGTH_LONG).show();

    }

    @Override
    public void onPause() {
```

```
    super.onPause();

    linealLayout.removeAllViews();
    labelsLinealLayout.removeAllViews();
}

public void addDeviceButtonPressed(View v)
{
    if(v.getId() == addDevice.getId()) {
        Intent intent;
        Bundle b;

        Privileges item = new Privileges();

        item.setSenderId(" ");
        item.setReceiverId(deviceId);
        item.setAlarm("0");
        item.setMap("0");
        item.setSenderName("NO NAME");

        intent = new Intent(v.getContext(), ControlDeviceActivity.class);
        b = new Bundle();

        b.putString("senderId", item.getSenderId());
        b.putString("receiverId", item.getReceiverId());
        b.putString("alarm", item.getAlarm());
        b.putString("map", item.getMap());
        b.putString("senderName", item.getSenderName());

        b.putString("deleteButton", "0");

        b.putString("dbUrl", dbUrl);

        b.putInt("from", 0);

        intent.putExtras(b);

        startActivity(intent);
    }
}

void updateTrustedDevicesTable()
{
    String sql = "SELECT sender_id,receiver_id,alarm,map,sender_name FROM
     Privileges WHERE receiver_id='" + deviceId + "'";

    new ConnectedDevicesWebServiceTask().execute(sql, "down_service.php");
}

void downloadTrustedFromDatabase(String result)
{
    int i;
    Privileges item;

    try {
        // Parse json data
        JSONArray jsonArray = new JSONArray(result);

        for (i = 0; i < jsonArray.length(); i++) {
            JSONObject jsonObject = jsonArray.getJSONObject(i);

            item = new Privileges();
```

```
            item.setSenderId(jsonObject.getString("sender_id"));
            item.setReceiverId(jsonObject.getString("receiver_id"));
            item.setAlarm(jsonObject.getString("alarm"));
            item.setMap(jsonObject.getString("map"));
            item.setSenderName(jsonObject.getString("sender_name"));

            privileges.add(item);
        }
    }
    catch (Exception e) {
        Log.e("Trusted json", "Error parsing data " + e.toString());
    }
}

@Override
public void onClick(View view) {
    int index;
    Intent intent;
    Bundle b;
    Button entry = (Button)view;

    Privileges item;

    index = findIndexFromName(entry.getText().toString());

    item = privileges.get(index);

    intent = new Intent(view.getContext(), ControlDeviceActivity.class);
    b = new Bundle();

    b.putString("senderId", item.getSenderId());
    b.putString("receiverId", item.getReceiverId());
    b.putString("alarm", item.getAlarm());
    b.putString("map", item.getMap());
    b.putString("senderName", item.getSenderName());

    b.putString("deleteButton", "1");

    b.putString("dbUrl", dbUrl);

    b.putInt("from", 1);

    intent.putExtras(b);

    startActivity(intent);
}

Integer findIndexFromName(String sName)
{
    int i;

    i = 0;
    while(i<privileges.size()) {
        if (privileges.get(i).getSenderName().equals(sName))
            return i;
        i++;
    }

    return -1;
}

void loadDevicesOnScreen()
{
```

```
int i;

WindowManager wm = (WindowManager) getSystemService(Context.WINDOW_SERVICE);
Display display = wm.getDefaultDisplay();
DisplayMetrics metrics = new DisplayMetrics();
display.getMetrics(metrics);
int width = metrics.widthPixels;

LinearLayout column1Layout, column2Layout, column3Layout;

LinearLayout.LayoutParams params1 = new LinearLayout.LayoutParams(width/3,
 LinearLayout.LayoutParams.WRAP_CONTENT);
params1.setMargins(0,0,0,20);

column1Layout = new LinearLayout(TrustedDevicesActivity.this);
column1Layout.setOrientation(LinearLayout.VERTICAL);
column1Layout.setLayoutParams(params1);
column1Layout.setGravity(Gravity.CENTER);

column2Layout = new LinearLayout(TrustedDevicesActivity.this);
column2Layout.setOrientation(LinearLayout.VERTICAL);
column2Layout.setLayoutParams(params1);
column2Layout.setGravity(Gravity.CENTER);

column3Layout = new LinearLayout(TrustedDevicesActivity.this);
column3Layout.setOrientation(LinearLayout.VERTICAL);
column3Layout.setLayoutParams(params1);
column3Layout.setGravity(Gravity.CENTER);

linealLayout.addView(column1Layout);
linealLayout.addView(column2Layout);
linealLayout.addView(column3Layout);

TextView senderIdText = new TextView(TrustedDevicesActivity.this);
senderIdText.setLayoutParams(params1);
senderIdText.setText("NAME");
senderIdText.setTextSize(25);
senderIdText.setTypeface(Typeface.create("", Typeface.BOLD_ITALIC));
senderIdText.setTextColor(Color.WHITE);
senderIdText.setGravity(Gravity.CENTER);

TextView alarmText = new TextView(TrustedDevicesActivity.this);
alarmText.setLayoutParams(params1);
alarmText.setText("ALARM");
alarmText.setTextSize(25);
alarmText.setTypeface(Typeface.create("", Typeface.BOLD_ITALIC));
alarmText.setTextColor(Color.WHITE);
alarmText.setGravity(Gravity.CENTER);

TextView mapText = new TextView(TrustedDevicesActivity.this);
mapText.setLayoutParams(params1);
mapText.setText("MAP");
mapText.setTextSize(25);
mapText.setTypeface(Typeface.create("", Typeface.BOLD_ITALIC));
mapText.setTextColor(Color.WHITE);
mapText.setGravity(Gravity.CENTER);

labelsLinealLayout.addView(senderIdText);
labelsLinealLayout.addView(alarmText);
labelsLinealLayout.addView(mapText);
```

```
    for (i = 0; i < privileges.size(); i++) {
        LinearLayout.LayoutParams params2 = new LinearLayout.LayoutParams(
         LinearLayout.LayoutParams.MATCH_PARENT,
         LinearLayout.LayoutParams.WRAP_CONTENT);

        Button idButton = new Button(TrustedDevicesActivity.this);

        idButton.setLayoutParams(params2);
        idButton.setText(privileges.get(i).getSenderName());
        idButton.setOnClickListener(TrustedDevicesActivity.this);
        idButton.setGravity(Gravity.CENTER);

        LinearLayout.LayoutParams params3 = new LinearLayout.LayoutParams(
         (int)(48*metrics.density),(int)(48*metrics.density));

        TextView alarmTextView = new TextView(TrustedDevicesActivity.this);
        alarmTextView.setLayoutParams(params3);
        if(privileges.get(i).getAlarm().equals("0"))
            alarmTextView.setBackgroundResource(R.drawable.no);
        else
            alarmTextView.setBackgroundResource(R.drawable.yes);

        TextView mapTextView = new TextView(TrustedDevicesActivity.this);
        mapTextView.setLayoutParams(params3);
        if(privileges.get(i).getMap().equals("0"))
            mapTextView.setBackgroundResource(R.drawable.no);
        else
            mapTextView.setBackgroundResource(R.drawable.yes);


        column1Layout.addView(idButton);
        column2Layout.addView(alarmTextView);
        column3Layout.addView(mapTextView);

        if (i<privileges.size()-1) {
            params1.setMargins(0, 10, 0, 10);

            ImageView line1 = new ImageView(TrustedDevicesActivity.this);
            line1.setLayoutParams(params1);
            line1.setBackgroundResource(R.drawable.hline);

            ImageView line2 = new ImageView(TrustedDevicesActivity.this);
            line2.setLayoutParams(params1);
            line2.setBackgroundResource(R.drawable.hline);

            ImageView line3 = new ImageView(TrustedDevicesActivity.this);
            line3.setLayoutParams(params1);
            line3.setBackgroundResource(R.drawable.hline);

            column1Layout.addView(line1);
            column2Layout.addView(line2);
            column3Layout.addView(line3);
        }
    }
}

class ConnectedDevicesWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;
```

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    spinner.setVisibility(View.VISIBLE);
}

@Override
protected String doInBackground(String... params)
{
    sql = params[0];
    phpFile = params[1];

    result = runWebService();

    return null;
}

protected void onProgressUpdate(String... progress) {
}


@Override
protected void onPostExecute(String file_url) {
    if (phpFile.equals("down_service.php")) {

        downloadTrustedFromDatabase(result);

        loadDevicesOnScreen();
    }

    spinner.setVisibility(View.GONE);
    addDevice.setEnabled(true);
    addDevice.setBackgroundResource(R.drawable.button);
}

String runWebService()
{
    List<NameValuePair> list = new ArrayList<>(1);

    list.add(new BasicNameValuePair("sql", sql ));

    // Setting up the connection
    try {
        HttpClient httpClient = new DefaultHttpClient();

        HttpPost httpPost = new HttpPost(dbUrl + phpFile);

        httpPost.setEntity(new UrlEncodedFormEntity(list));

        HttpResponse response = httpClient.execute(httpPost);

        HttpEntity entity = response.getEntity();
        isr = entity.getContent();
    }
    catch(ClientProtocolException e) {
        Log.e("Trusted ClientProtocol", e.toString());
        e.printStackTrace();

        result = "0";
    }
    catch (IOException e) {
        Log.e("Trusted IOException", e.toString());
```

```
                e.printStackTrace();

                result = "0";
            }

            // Convert the response to string
            try {
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e) {
                Log.e("Trusted Convert", e.toString());
            }

            return result;
        }
    }

}
```

A.4 ConnectedDevicesActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.provider.AlarmClock;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.LinearLayout;
import android.widget.ProgressBar;
import android.widget.Toast;
import android.widget.ToggleButton;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
```

```
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;

public class ConnectedDevicesActivity extends Activity implements
View.OnClickListener, View.OnLongClickListener {

    private String deviceId;
    private String dbUrl;
    private String application;
    private String receiverState;
    private String mode;
    private String timer;
    private boolean isInPopUpView;

    private LinearLayout connectedDevicesLayout;
    private LinearLayout popupViewLayout;
    private Button setActionButton;
    private EditText nameTextView;
    private ProgressBar spinner;

    private ArrayList<Privileges> privileges = new ArrayList<>();
    ArrayList<Privileges> devicesList = new ArrayList<>();
    private Privileges item;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_connected_devices);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        Bundle b = getIntent().getExtras();

        connectedDevicesLayout = (LinearLayout)
         findViewById(R.id.connectedDevicesLayout);
        popupViewLayout= (LinearLayout)findViewById(R.id.popupViewLayout);
        setActionButton = (Button)findViewById(R.id.setActionbutton);
        nameTextView = (EditText)findViewById(R.id.nameTextView);
        spinner = (ProgressBar)findViewById(R.id.spinner);

        deviceId = MainActivity.deviceId;
        dbUrl = MainActivity.dbUrl;

        application = b.getString("application");
        mode = b.getString("mode");
        receiverState = b.getString("receiverState");
        timer = b.getString("timer");
    }

    @Override
    public void onResume()
    {
        super.onResume();
```

```
        Log.v("application", application);
        Log.v("mode", mode);
        Log.v("receiverState", receiverState);
        Log.v("timer", timer);

        spinner.setVisibility(View.GONE);
        popupViewLayout.setVisibility(View.GONE);

        isInPopUpView = false;

        ConnectivityManager connManager = (ConnectivityManager)
         getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo mWifi = connManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
        NetworkInfo mMobile =
         connManager.getNetworkInfo(ConnectivityManager.TYPE_MOBILE);

        if (mWifi.isConnected() || mMobile.isConnected())
            updateConnectedDevicesTable();
        else
            Toast.makeText(getApplicationContext(), "NO INTERNET CONNECTION!!!",
             Toast.LENGTH_LONG).show();
    }

    @Override
    public void onPause()
    {
        super.onPause();

        privileges = new ArrayList<>();
        devicesList = new ArrayList<>();
        connectedDevicesLayout.removeAllViews();
    }

    void updateConnectedDevicesTable()
    {
         String sql = null;

         if(application.equals("AlarmClock"))
             sql = "SELECT sender_id,receiver_id,alarm,map,receiver_name FROM
                  Privileges WHERE sender_id='" + deviceId + "'&&alarm='1'";
         else if(application.equals("Map"))
             sql = "SELECT sender_id,receiver_id,alarm,map,receiver_name FROM
                  Privileges WHERE sender_id='" + deviceId + "'&&map='1'";

         if (sql != null)
             new ConnectedDevicesWebServiceTask().execute(sql,"down_service.php");
    }

    void downloadConnectedDevicesFromDatabase(String result)
    {
        int i;
        Privileges item;

        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            for (i = 0; i < jsonArray.length(); i++) {
                JSONObject jsonObject = jsonArray.getJSONObject(i);

                item = new Privileges();

                item.setSenderId(jsonObject.getString("sender_id"));
```

```
                item.setReceiverId(jsonObject.getString("receiver_id"));
                item.setAlarm(jsonObject.getString("alarm"));
                item.setMap(jsonObject.getString("map"));
                item.setReceiverName(jsonObject.getString("receiver_name"));

                privileges.add(item);
            }
        }
        catch (Exception e) {
            Log.e("Connect json", "Error parsing data " + e.toString());
        }
    }

    @Override
    public void onClick(View view)
    {
        if(!isInPopUpView && view.getId()!=setActionButton.getId()) {

            if (mode.equals("SINGLE")) {
                Intent intent = null;
                Bundle b;

                Button entry = (Button) view;

                spinner.setVisibility(View.GONE);

                item = findEntryFromButtonText((String) entry.getText());

                if (item.getReceiverName().equals("NO NAME")) {
                    nameTextView.setText(item.getReceiverName());
                    popupViewLayout.setVisibility(View.VISIBLE);
                } else {
                    if (application.equals("AlarmClock"))
                        intent = new Intent(view.getContext(),SetAlarmActivity.class);
                    else if (application.equals("Map"))
                        intent = new Intent(view.getContext(), SetMapActivity.class);

                    b = new Bundle();

                    b.putString("senderId", item.getSenderId());
                    b.putString("receiverId", item.getReceiverId());
                    b.putString("mode", mode);
                    b.putString("timer", timer);

                    intent.putExtras(b);

                    if (intent != null)
                        startActivity(intent);
                }
            }
            else {
                ToggleButton entry = (ToggleButton) view;

                spinner.setVisibility(View.GONE);

                item = findEntryFromButtonText((String) entry.getTextOff());

                if (item.getReceiverName().equals("NO NAME")) {
                    nameTextView.setText(item.getReceiverName());
                    popupViewLayout.setVisibility(View.VISIBLE);
                } else {
                    if (entry.isChecked())
                        devicesList.add(item);
```

```
                else
                    devicesList.remove(item);

                Log.v("Devices List Size", devicesList.size()+"");
            }
        }
    }
}

public void setActionButtonPressed(View v)
{
    Intent intent = null;
    Bundle b;

    if (application.equals("AlarmClock"))
        intent = new Intent(v.getContext(), SetAlarmActivity.class);
    else if (application.equals("Map"))
        intent = new Intent(v.getContext(), SetMapActivity.class);

    b = new Bundle();

    b.putStringArrayList("sendersIdList",
     parseSendersIdStringListFromDevicesList());
    b.putStringArrayList("receiversIdList",
     parseReceiversIdStringListFromDevicesList());
    b.putString("mode", mode);
    b.putString("timer", timer);

    intent.putExtras(b);

    if (intent != null)
        startActivity(intent);
}

@Override
public boolean onLongClick(View v)
{
    Button entry = (Button)v;

    isInPopUpView = true;

    item = findEntryFromButtonText((String)entry.getText());
    nameTextView.setText(item.getReceiverName());
    popupViewLayout.setVisibility(View.VISIBLE);

    return false;
}

public void setReceiverName(View v)
{
    String sql;
    String receiverName;

    receiverName = nameTextView.getText().toString().replace('-', '_');

    if (!checkIfEnteredNameExists(receiverName)) {

        sql = "UPDATE Privileges SET receiver_name='" + receiverName + "' WHERE
         receiver_id='" + item.getReceiverId() + "' && sender_id='" +
         item.getSenderId() + "'";

        new ConnectedDevicesWebServiceTask().execute(sql, "up_service.php");
```

```
            popupViewLayout.setVisibility(View.GONE);

            onPause();
            onResume();
        }
        else
            Toast.makeText(getApplicationContext(), "Name already exists!!!",
             Toast.LENGTH_SHORT).show();
    }

    ArrayList<String> parseSendersIdStringListFromDevicesList()
    {
        int i;
        ArrayList<String> list = new ArrayList<>();

        for (i=0;i<devicesList.size();i++)
            list.add(devicesList.get(i).getSenderId());

        return list;
    }

    ArrayList<String> parseReceiversIdStringListFromDevicesList()
    {
        int i;
        ArrayList<String> list = new ArrayList<>();

        for (i=0;i<devicesList.size();i++)
            list.add(devicesList.get(i).getReceiverId());

        return list;
    }

    boolean checkIfEnteredNameExists(String name)
    {
        int i;

        i=0;
        while(i<privileges.size()) {
            if(privileges.get(i).getReceiverName().equals(name))
                return true;
            i++;
        }

        return false;
    }

    Privileges findEntryFromButtonText(String text)
    {
        int i;
        boolean found = false;
        Privileges item = new Privileges();

        i=0;
        while(i<privileges.size() && !found) {

            if (privileges.get(i).getReceiverId().equals(text) ||
             privileges.get(i).getReceiverName().equals(text)) {
                item = privileges.get(i);
                found = true;
            }

            i++;
        }
```

```
    return item;
}

void loadReceiversListOnScreen()
{
    int i;

    LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(
     LinearLayout.LayoutParams.MATCH_PARENT,
     LinearLayout.LayoutParams.WRAP_CONTENT);
    params.setMargins(0,0,0,20);

    for (i=0; i<privileges.size(); i++) {
        if (mode.equals("SINGLE")) {
            Button dButton = new Button(ConnectedDevicesActivity.this);

            dButton.setLayoutParams(params);
            if (privileges.get(i).getReceiverName().equals("NO NAME"))
                dButton.setText(privileges.get(i).getReceiverId());
            else
                dButton.setText(privileges.get(i).getReceiverName());
            dButton.setBackgroundResource(R.drawable.button);
            dButton.setOnClickListener(ConnectedDevicesActivity.this);
            dButton.setLongClickable(true);
            dButton.setOnLongClickListener(ConnectedDevicesActivity.this);

            connectedDevicesLayout.addView(dButton, i);
        }
        else {
            ToggleButton dToggleButton = new ToggleButton(
             ConnectedDevicesActivity.this);

            dToggleButton.setChecked(false);

            dToggleButton.setLayoutParams(params);
            if (privileges.get(i).getReceiverName().equals("NO NAME")) {
                dToggleButton.setText(privileges.get(i).getReceiverId());
                dToggleButton.setTextOn(privileges.get(i).getReceiverId() + "
                 (Selected)");
                dToggleButton.setTextOff(privileges.get(i).getReceiverId());
            }
            else {
                dToggleButton.setText(privileges.get(i).getReceiverName());
                dToggleButton.setTextOn(privileges.get(i).getReceiverName() + "
                 (Selected)");
                dToggleButton.setTextOff(privileges.get(i).getReceiverName());
            }
            dToggleButton.setBackgroundResource(R.drawable.button);
            dToggleButton.setOnClickListener(ConnectedDevicesActivity.this);
            dToggleButton.setLongClickable(true);
            dToggleButton.setOnLongClickListener(ConnectedDevicesActivity.this);

            connectedDevicesLayout.addView(dToggleButton, i);
        }
    }
}

class ConnectedDevicesWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
```

```
String phpFile;

@Override
protected void onPreExecute() {
    super.onPreExecute();

    spinner.setVisibility(View.VISIBLE);
}

@Override
protected String doInBackground(String... params)
{
    sql = params[0];
    phpFile = params[1];

    result = runWebService();

    return null;
}

protected void onProgressUpdate(String... progress) {
}

@Override
protected void onPostExecute(String file_url) {
    if (phpFile.equals("down_service.php")) {
        downloadConnectedDevicesFromDatabase(result);

        loadReceiversListOnScreen();
    }

    spinner.setVisibility(View.GONE);
}

String runWebService()
{
    List<NameValuePair> list = new ArrayList<>(1);

    list.add(new BasicNameValuePair("sql", sql ));

    // Setting up the connection
    try {
        HttpClient httpClient = new DefaultHttpClient();

        HttpPost httpPost = new HttpPost(dbUrl + phpFile);

        httpPost.setEntity(new UrlEncodedFormEntity(list));

        HttpResponse response = httpClient.execute(httpPost);

        HttpEntity entity = response.getEntity();
        isr = entity.getContent();
    }
    catch(ClientProtocolException e) {
        Log.e("Connect ClientProtocol", e.toString());
        e.printStackTrace();

        result = "0";
    }
    catch (IOException e) {
        Log.e("Connect IOException", e.toString());
        e.printStackTrace();
```

```
                result = "0";
            }

            // Convert the response to string
            try {
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e) {
                Log.e("Connect convert", e.toString());
            }

            return result;
        }
    }
}
```

A.5 SettingsActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Context;
import android.content.pm.ActivityInfo;
import android.os.Bundle;
import android.widget.EditText;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.util.StringTokenizer;


public class SettingsActivity extends Activity {

    String dbUrl;
    int interval;

    EditText intervalEditText;
    EditText dbUrlEditText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_settings);
```

```
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        Bundle b;

        intervalEditText = (EditText)findViewById(R.id.intervalEditText);
        dbUrlEditText = (EditText)findViewById(R.id.dbUrlEditText);

        b = getIntent().getExtras();

        interval = b.getInt("interval");
        dbUrl = b.getString("dbUrl");

        intervalEditText.setText(String.valueOf(interval/1000));
        dbUrlEditText.setText(dbUrl);
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();

        interval = Integer.parseInt(intervalEditText.getText().toString()) * 1000;
        dbUrl = dbUrlEditText.getText().toString();

        // Update app info file
        updateAppInfoFile("AppInfo");

        this.finish();
    }

    void updateAppInfoFile(String filename)
    {
        BufferedWriter bufferedWriter = null;
        BufferedReader bufferedReader = null;

        StringTokenizer st;

        StringBuilder str = new StringBuilder();

        try {
            FileInputStream fileInputStream = openFileInput(filename);
            bufferedReader = new BufferedReader(
             new InputStreamReader(fileInputStream));

            FileOutputStream fileOutputStream = openFileOutput(filename+"TMP",
             Context.MODE_PRIVATE);
            bufferedWriter = new BufferedWriter(
             new OutputStreamWriter(fileOutputStream));

            String line, name;

            while ((line = bufferedReader.readLine()) != null) {
                st = new StringTokenizer(line, "~");

                name = st.nextToken();

                if(name.equals("INTERVAL"))
                    line = "INTERVAL~" + interval;

                str.append(line).append("\n");
            }

            bufferedWriter.write(str.toString());
```

```
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            try {
                if (bufferedReader != null) {
                    bufferedReader.close();
                }
                if (bufferedWriter != null) {
                    bufferedWriter.close();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }

        File file = new File(getFilesDir(), filename+"TMP");
        file.renameTo(new File(getFilesDir(), filename));
    }
}
```

A.6 ApplicationsActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;


public class ApplicationsActivity extends Activity {
    Button alarmButton;
    Button mapButton;

    private ProgressBar spinner;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_applications);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        alarmButton = (Button) findViewById(R.id.alarmButton);
        mapButton = (Button) findViewById(R.id.mapButton);

        spinner = (ProgressBar)findViewById(R.id.spinner);
        spinner.setVisibility(View.GONE);
    }

    public void onPause() {
        super.onPause();

        spinner.setVisibility(View.GONE);

        alarmButton.setEnabled(true);
        mapButton.setEnabled(true);
```

```
    }

    public void alarmButtonPressed(View v)
    {
        if(v.getId()==alarmButton.getId()) {

            spinner.setVisibility(View.VISIBLE);

            alarmButton.setEnabled(false);
            mapButton.setEnabled(false);

            Intent intent = new Intent(v.getContext(), ModesActivity.class);
            Bundle b = new Bundle();

            b.putString("application", "AlarmClock");

            intent.putExtras(b);

            startActivity(intent);
        }
    }

    public void mapButtonPressed(View v)
    {
        if(v.getId()==mapButton.getId()) {

            spinner.setVisibility(View.VISIBLE);

            alarmButton.setEnabled(false);
            mapButton.setEnabled(false);

            Intent intent = new Intent(v.getContext(), ModesActivity.class);
            Bundle b = new Bundle();

            b.putString("application", "Map");

            intent.putExtras(b);

            startActivity(intent);
        }
    }
}
```

A.7 ControlDeviceActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Context;
import android.content.pm.ActivityInfo;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.ProgressBar;
```

```
import android.widget.RelativeLayout;
import android.widget.Toast;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;

// ------------------ ADD DEVICE ACTIVITY LAYOUT -------------------

public class ControlDeviceActivity extends Activity {
    String senderId, receiverId, alarm, map, deleteButtonState, senderName;
    String checkId, method, dbUrl, tmpSenderName;
    EditText deviceIdText, nameText;
    CheckBox alarmCheckBox, mapCheckBox;
    Button addDeviceButton, deleteDeviceButton;
    int setMode; // 0 for insert, 1 for update, -1 for searching
    int from;

    ArrayList<String> sendersNamesList, sendersIdList;

    private ProgressBar spinner;
    RelativeLayout mainView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_add_device);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        spinner = (ProgressBar)findViewById(R.id.spinner);
        spinner.setVisibility(View.GONE);

        Bundle b = getIntent().getExtras();

        senderId = b.getString("senderId");
        receiverId = b.getString("receiverId");
        alarm = b.getString("alarm");
        map = b.getString("map");
        dbUrl = b.getString("dbUrl");
        deleteButtonState = b.getString("deleteButton");
        senderName = b.getString("senderName");
        from = b.getInt("from");

        deviceIdText = (EditText) findViewById(R.id.deviceIdText);
        alarmCheckBox = (CheckBox) findViewById(R.id.alarmCheckBox);
        mapCheckBox = (CheckBox) findViewById(R.id.mapCheckBox);
        addDeviceButton = (Button) findViewById(R.id.addDeviceButton);
```

```
        deleteDeviceButton = (Button) findViewById(R.id.deleteDeviceButton);
        nameText = (EditText)findViewById(R.id.nameText);
        mainView = (RelativeLayout)findViewById(R.id.mainView);

        sendersNamesList = new ArrayList<>();
        sendersIdList = new ArrayList<>();

        if (deleteButtonState.equals("1")) {
            deleteDeviceButton.setVisibility(View.VISIBLE);
            addDeviceButton.setText("UPDATE DEVICE");
        }
        else {
            deleteDeviceButton.setVisibility(View.INVISIBLE);
            addDeviceButton.setText("ADD DEVICE");
        }

        if (from == 0) {
            deviceIdText.setText(senderId);
            checkId = "";
            setMode = -1;
        }
        else {
            deviceIdText.setEnabled(false);
            deviceIdText.setText(senderId);
            nameText.setText(senderName);
            checkId = senderId;
            setMode = 1;
        }

        if(alarm.equals("1"))
            alarmCheckBox.setChecked(true);
        else
            alarmCheckBox.setChecked(false);

        if(map.equals("1"))
            mapCheckBox.setChecked(true);
        else
            mapCheckBox.setChecked(false);
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();

        spinner.setVisibility(View.GONE);

        this.finish();
    }

    public void addDeviceButtonPressed(View v)
    {
        if (v.getId() == addDeviceButton.getId()) {

            ConnectivityManager connManager = (ConnectivityManager)getSystemService(
             Context.CONNECTIVITY_SERVICE);
            NetworkInfo mWifi = connManager.getNetworkInfo(
             ConnectivityManager.TYPE_WIFI);
            NetworkInfo mMobile = connManager.getNetworkInfo(
             ConnectivityManager.TYPE_MOBILE);

            if (mWifi.isConnected() || mMobile.isConnected()) {
                // Searching
                if (setMode == -1) {
```

```
                checkId = "";
                senderName = nameText.getText().toString().replace('-','_');

                String sql = "SELECT device_id FROM Devices WHERE device_id='" +
                 deviceIdText.getText().toString().replace('-','_') + "'";
                method = "findSenderId";
                new AddDeviceWebServiceTask().execute(sql, "down_service.php");
            }
            else {
                String sql;

                senderName = nameText.getText().toString().replace('-','_');

                sql = "SELECT sender_id,sender_name FROM Privileges WHERE
                 receiver_id='" + receiverId + "'";
                method = "findInPrivileges";
                new AddDeviceWebServiceTask().execute(sql, "down_service.php");
            }
        }
        else
            Toast.makeText(getApplicationContext(), "NO INTERNET CONNECTION!!!",
             Toast.LENGTH_LONG).show();

    }
}

public void deleteDeviceButtonPressed(View v)
{
    if (v.getId() == deleteDeviceButton.getId()) {
        String sql;

        addDeviceButton.setEnabled(false);
        addDeviceButton.setBackgroundResource(R.drawable.button_disabled);
        deleteDeviceButton.setEnabled(false);
        deleteDeviceButton.setBackgroundResource(R.drawable.button_disabled);

        senderName = nameText.getText().toString().replace('-','_');

        sql = "DELETE FROM Privileges WHERE sender_id='" + senderId + "'";
        method = "deleteEntry";
        new AddDeviceWebServiceTask().execute(sql, "up_service.php");
    }
}

void downloadSenderIdFromDatabase(String result)
{
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            if (jsonArray.length() > 0) {
                JSONObject jsonObject = jsonArray.getJSONObject(0);

                // Set device id
                checkId = jsonObject.getString("device_id");
            }
        } catch (Exception e) {
            Log.e("Control json", "Error parsing data " + e.toString());
        }
    }
}
```

```
void checkInPrivileges(String result)
{
    int i;
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            JSONObject jsonObject;

            for(i=0;i<jsonArray.length();i++) {
                jsonObject = jsonArray.getJSONObject(i);
                sendersIdList.add(jsonObject.getString("sender_id"));
                sendersNamesList.add(jsonObject.getString("sender_name"));
            }

            if (sendersIdList.indexOf(checkId)!=-1) {
                setMode = 1; // Update
                tmpSenderName = sendersNamesList.get(
                 sendersIdList.indexOf(checkId));
            }
            else
                setMode = 0; // Insert
        } catch (Exception e) {
            Log.e("Control json", "Error parsing data " + e.toString());
        }
    }
}

class AddDeviceWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        spinner.setVisibility(View.VISIBLE);

        deviceIdText.setEnabled(false);
        nameText.setEnabled(false);
        alarmCheckBox.setEnabled(false);
        mapCheckBox.setEnabled(false);
        addDeviceButton.setEnabled(false);
        addDeviceButton.setBackgroundResource(R.drawable.button_disabled);
        deleteDeviceButton.setEnabled(false);
        deleteDeviceButton.setBackgroundResource(R.drawable.button_disabled);
        mainView.setEnabled(false);
    }

    @Override
    protected String doInBackground(String... params)
    {
        sql = params[0];
        phpFile = params[1];

        result = runWebService();

        return null;
    }
```

```
protected void onProgressUpdate(String... progress) {
}

@Override
protected void onPostExecute(String file_url) {
    if (phpFile.equals("down_service.php")) {
        if (method.equals("findSenderId")) {
            downloadSenderIdFromDatabase(result);

            if (!checkId.equals("")) {
                String sql;

                senderName = nameText.getText().toString().replace('-','_');

                result = null;

                sql = "SELECT sender_id,sender_name FROM Privileges WHERE
                 receiver_id='" + receiverId + "'";
                method = "findInPrivileges";
                new AddDeviceWebServiceTask().execute(sql,"down_service.php");
            }
            else {
                Toast.makeText(getApplicationContext(), "DEVICE NOT FOUND!!!",
                 Toast.LENGTH_LONG).show();
            }
        }

        if(method.equals("findInPrivileges")) {
            String sql;
            String alarmString, mapString;

            checkInPrivileges(result);

            if (alarmCheckBox.isChecked())
                alarmString = "1";
            else
                alarmString = "0";

            if (mapCheckBox.isChecked())
                mapString = "1";
            else
                mapString = "0";

            // Insert
            if (setMode == 0) {
                if (sendersNamesList.indexOf(senderName) == -1) {
                    sql = "INSERT INTO Privileges(sender_id,receiver_id,alarm,
                    map,sender_name) VALUES ('" + checkId + "','" + receiverId
                    + "','" + alarmString + "','" + mapString + "','" +
                    senderName + "')";

                    new AddDeviceWebServiceTask().execute(sql,
                    "up_service.php");

                    Toast.makeText(getApplicationContext(), "Insert Device
                    Completed", Toast.LENGTH_SHORT).show();
                }
                else
                    Toast.makeText(getApplicationContext(), "Name already
                    exists!!!", Toast.LENGTH_SHORT).show();
            }
```

```
                // Update
                if (setMode == 1) {
                    if (sendersNamesList.indexOf(senderName) == -1 ||
                     senderName.equals(tmpSenderName)) {
                        sql = "UPDATE Privileges SET alarm='" + alarmString + "',
                         map='" + mapString + "', sender_name='" + senderName + "'
                         WHERE receiver_id='" + receiverId + "' && sender_id='" +
                         checkId + "'";

                        new AddDeviceWebServiceTask().execute(sql,
                         "up_service.php");

                        Toast.makeText(getApplicationContext(), "Update Device
                         Completed", Toast.LENGTH_SHORT).show();
                    }
                    else
                        Toast.makeText(getApplicationContext(), "Name already
                         exists!!!", Toast.LENGTH_SHORT).show();
                }
            }
        }
        else {
            if(method.equals("deleteEntry")) {
                Toast.makeText(getApplicationContext(), "Delete Device Completed",
                 Toast.LENGTH_SHORT).show();

                ControlDeviceActivity.this.finish();
            }
        }


        sendersNamesList = new ArrayList<>();
        sendersIdList = new ArrayList<>();

        spinner.setVisibility(View.GONE);

        if (from == 0) {
            setMode = -1;
            deviceIdText.setEnabled(true);
        }
        nameText.setEnabled(true);
        alarmCheckBox.setEnabled(true);
        mapCheckBox.setEnabled(true);
        addDeviceButton.setEnabled(true);
        addDeviceButton.setBackgroundResource(R.drawable.button);
        deleteDeviceButton.setEnabled(true);
        deleteDeviceButton.setBackgroundResource(R.drawable.button);
        mainView.setEnabled(true);
    }

    String runWebService()
    {
        List<NameValuePair> list = new ArrayList<>(1);

        list.add(new BasicNameValuePair("sql", sql ));

        // Setting up the connection
        try {
            HttpClient httpClient = new DefaultHttpClient();

            HttpPost httpPost = new HttpPost(dbUrl + phpFile);
```

```
                httpPost.setEntity(new UrlEncodedFormEntity(list));

                HttpResponse response = httpClient.execute(httpPost);

                HttpEntity entity = response.getEntity();
                isr = entity.getContent();
            }
            catch(ClientProtocolException e) {
                Log.e("Control ClientProtocol", e.toString());
                e.printStackTrace();

                result = "0";
            }
            catch (IOException e) {
                Log.e("Control IOException", e.toString());
                e.printStackTrace();

                result = "0";
            }

            // Convert the response to string
            try {
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e) {
                Log.e("Control Convert", e.toString());
            }

            return result;
        }
    }
}
```

A.8 SetAlarmActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Context;
import android.content.pm.ActivityInfo;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TimePicker;
```

```
import android.widget.Toast;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;


public class SetAlarmActivity extends Activity
{
    private String deviceId;
    private String receiverId;
    private String mode;
    private String timer;
    private String dbUrl;
    private ArrayList<String> devicesIdList;
    private ArrayList<String> receiversIdList;

    private Button addAlarmButton;
    private EditText messageText;
    private TimePicker timePicker;
    private ProgressBar spinner;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_set_alarm);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        devicesIdList = new ArrayList<>();
        receiversIdList = new ArrayList<>();

        Bundle b = getIntent().getExtras();

        dbUrl = MainActivity.dbUrl;
        mode = b.getString("mode");
        timer = b.getString("timer");

        if (mode.equals("SINGLE")) {
            deviceId = b.getString("senderId");
            receiverId = b.getString("receiverId");
        }
        else {
            devicesIdList = b.getStringArrayList("sendersIdList");
            receiversIdList = b.getStringArrayList("receiversIdList");
        }

        addAlarmButton = (Button) findViewById(R.id.addAlarm);
        messageText = (EditText) findViewById(R.id.message);
```

```
    timePicker = (TimePicker) findViewById(R.id.timePicker);

    spinner = (ProgressBar)findViewById(R.id.spinner);
    spinner.setVisibility(View.GONE);
}

public void sendActionAddAlarm(View view)
{
    int i;

    if (view.getId() == addAlarmButton.getId()) {
        ConnectivityManager connManager = (ConnectivityManager)getSystemService(
         Context.CONNECTIVITY_SERVICE);
        NetworkInfo mWifi = connManager.getNetworkInfo(
         ConnectivityManager.TYPE_WIFI);
        NetworkInfo mMobile = connManager.getNetworkInfo(
         ConnectivityManager.TYPE_MOBILE);

        if (mWifi.isConnected() || mMobile.isConnected()) {
            String action;
            String sql;

            if(messageText.getText().toString().equals(""))
                messageText.setText("Empty Message");

            if (mode.equals("SINGLE")) {
                receiverId = "1,".concat(receiverId);
            }
            else {
                receiverId = String.valueOf(receiversIdList.size()).concat(",");

                for(i=0;i<receiversIdList.size()-1;i++) {
                    receiverId = receiverId.concat(
                     receiversIdList.get(i)).concat(",");
                }
                receiverId = receiverId.concat(receiversIdList.get(i));
            }

            action = "alarm-" + messageText.getText().toString().replace('-', '_')
             + "-" + timePicker.getCurrentHour().toString() + "-" +
             timePicker.getCurrentMinute().toString();

            sql = "INSERT INTO Actions (sender_id,receivers_id,action,state)
             VALUES ('" + deviceId + "','" + receiverId + "','" + action +
             "','0')";

            new SetAlarmWebServiceTask().execute(sql, "up_service.php");
        }
        else
            Toast.makeText(getApplicationContext(), "NO INTERNET CONNECTION!!!",
             Toast.LENGTH_LONG).show();
    }
}

class SetAlarmWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;

    @Override
    protected void onPreExecute()
```

```
{
    super.onPreExecute();

    spinner.setVisibility(View.VISIBLE);

    messageText.setEnabled(false);
    timePicker.setEnabled(false);
    addAlarmButton.setEnabled(false);
    addAlarmButton.setBackgroundResource(R.drawable.button_disabled);
}

@Override
protected String doInBackground(String... params)
{
    sql = params[0];
    phpFile = params[1];

    result = runWebService();

    return null;
}

protected void onProgressUpdate(String... progress) {
}

@Override
protected void onPostExecute(String file_url) {
    spinner.setVisibility(View.GONE);

    addAlarmButton.setEnabled(true);
    addAlarmButton.setBackgroundResource(R.drawable.button);
    messageText.setEnabled(true);
    timePicker.setEnabled(true);

    Toast.makeText(getApplicationContext(), "Alarm Action Completed",
     Toast.LENGTH_LONG).show();
}

String runWebService()
{
    List<NameValuePair> list = new ArrayList<>(1);

    list.add(new BasicNameValuePair("sql", sql ));

    // Setting up the connection
    try {

        HttpClient httpClient = new DefaultHttpClient();

        HttpPost httpPost = new HttpPost(dbUrl + phpFile);

        httpPost.setEntity(new UrlEncodedFormEntity(list));

        HttpResponse response = httpClient.execute(httpPost);

        HttpEntity entity = response.getEntity();
        isr = entity.getContent();
    }
    catch(ClientProtocolException e) {
        Log.e("SetAlarm ClientProtocol", e.toString());
        e.printStackTrace();

        result = "0";
```

```
            }
            catch (IOException e) {
                Log.e("SetAlarm IOException", e.toString());
                e.printStackTrace();

                result = "0";
            }

            // Convert the response to string
            try
            {
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e)
            {
                Log.e("SetAlarm Convert", e.toString());
            }

            return result;
        }
    }
}
```

A.9 SetMapActivity.java

```
package com.ppapadop.rac;

import android.content.Context;
import android.content.pm.ActivityInfo;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.support.v4.app.FragmentActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.Toast;

import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
```

```
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;

public class SetMapActivity extends FragmentActivity implements
GoogleMap.OnMapClickListener
{
    private String deviceId;
    private String receiverId;
    private String mode;
    private String timer;
    private String dbUrl;
    private ArrayList<String> devicesIdList;
    private ArrayList<String> receiversIdList;

    private GoogleMap mMap; // Null if Google Play services APK is not available.

    LatLng coord = null;

    Button addMapButton;

    private ProgressBar spinner;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_set_map);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        devicesIdList = new ArrayList<>();
        receiversIdList = new ArrayList<>();

        Bundle b = getIntent().getExtras();

        dbUrl = MainActivity.dbUrl;
        mode = b.getString("mode");
        timer = b.getString("timer");

        if (mode.equals("SINGLE")) {
            deviceId = b.getString("senderId");
            receiverId = b.getString("receiverId");
        }
        else {
            devicesIdList = b.getStringArrayList("sendersIdList");
            receiversIdList = b.getStringArrayList("receiversIdList");
        }

        addMapButton = (Button)findViewById(R.id.addMapButton);
        spinner = (ProgressBar)findViewById(R.id.spinner);
        spinner.setVisibility(View.GONE);

        coord = null;

        setUpMapIfNeeded();
```

```
}

@Override
protected void onResume() {
    super.onResume();

    setUpMapIfNeeded();
}

private void setUpMapIfNeeded() {
    // Do a null check to confirm that we have not already instantiated the map.
    if (mMap == null) {
        // Try to obtain the map from the SupportMapFragment.
        mMap = ((SupportMapFragment) getSupportFragmentManager()
         .findFragmentById(R.id.map)).getMap();
        // Check if we were successful in obtaining the map.
        if (mMap != null) {
            setUpMap();
        }
    }
}

private void setUpMap() {
    mMap.setMyLocationEnabled(true);
    mMap.setOnMapClickListener(SetMapActivity.this);

    addMapButton.setEnabled(false);
    addMapButton.setBackgroundResource(R.drawable.button_disabled);
}

public void sendActionSendPosition(View v)
{
    int i;

    if (v.getId() == addMapButton.getId()) {
        ConnectivityManager connManager = (ConnectivityManager) getSystemService(
         Context.CONNECTIVITY_SERVICE);
        NetworkInfo mWifi = connManager.getNetworkInfo(
         ConnectivityManager.TYPE_WIFI);
        NetworkInfo mMobile = connManager.getNetworkInfo(
         ConnectivityManager.TYPE_MOBILE);

        if (mWifi.isConnected() || mMobile.isConnected()) {
            String action;
            String sql;

            if (mode.equals("SINGLE")) {
                receiverId = "1,".concat(receiverId);
            }
            else {
                receiverId = String.valueOf(receiversIdList.size()).concat(",");

                for(i=0;i<receiversIdList.size()-1;i++) {
                    receiverId = receiverId.concat(
                     receiversIdList.get(i)).concat(",");
                }
                receiverId = receiverId.concat(receiversIdList.get(i));
            }

            action = "map-" + receiverId + "-" + coord.latitude + "-" +
             coord.longitude;

            sql = "INSERT INTO Actions (sender_id,receivers_id,action,state)
```

```
                VALUES ('" + deviceId + "','" + receiverId + "','" + action +
                "','0')";

            new SetMapWebServiceTask().execute(sql, "up_service.php");
        }
        else
            Toast.makeText(getApplicationContext(), "NO INTERNET CONNECTION!!!",
             Toast.LENGTH_LONG).show();
    }
}

@Override
public void onMapClick(LatLng latLng)
{
    coord = latLng;

    mMap.clear();
    mMap.addMarker(new MarkerOptions().position(coord).title("Marker"));

    addMapButton.setEnabled(true);
    addMapButton.setBackgroundResource(R.drawable.button);
}

class SetMapWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;

    @Override
    protected void onPreExecute()
    {
        super.onPreExecute();

        spinner.setVisibility(View.VISIBLE);

        addMapButton.setEnabled(false);
        addMapButton.setBackgroundResource(R.drawable.button_disabled);
    }

    @Override
    protected String doInBackground(String... params)
    {
        sql = params[0];
        phpFile = params[1];

        result = runWebService();

        return null;
    }

    protected void onProgressUpdate(String... progress) {
    }

    @Override
    protected void onPostExecute(String file_url) {
        spinner.setVisibility(View.GONE);

        addMapButton.setEnabled(true);
        addMapButton.setBackgroundResource(R.drawable.button);

        Toast.makeText(getApplicationContext(), "Map Action Completed",
```

```
                Toast.LENGTH_LONG).show();
        }

        String runWebService()
        {
            List<NameValuePair> list = new ArrayList<>(1);

            list.add(new BasicNameValuePair("sql", sql ));

            // Setting up the connection
            try {
                HttpClient httpClient = new DefaultHttpClient();

                HttpPost httpPost = new HttpPost(dbUrl + phpFile);

                httpPost.setEntity(new UrlEncodedFormEntity(list));

                HttpResponse response = httpClient.execute(httpPost);

                HttpEntity entity = response.getEntity();
                isr = entity.getContent();
            }
            catch(ClientProtocolException e) {
                Log.e("SetMap ClientProtocol", e.toString());
                e.printStackTrace();

                result = "0";
            }
            catch (IOException e) {
                Log.e("SetMap IOException", e.toString());
                e.printStackTrace();

                result = "0";
            }

            // Convert the response to string
            try {
                InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
                BufferedReader reader = new BufferedReader(inputReader, 8);
                StringBuilder sb = new StringBuilder();
                String line;

                while ((line = reader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                isr.close();

                result = sb.toString();
            }
            catch(Exception e) {
                Log.e("SetMap Convert", e.toString());
            }

            return result;
        }
    }
}
```

A.10 ModesActivity.java

```
package com.ppapadop.rac;

import android.app.Activity;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.graphics.Color;
import android.opengl.Visibility;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;


public class ModesActivity extends Activity
{
    private String application;
    private String receiverState;
    private String mode;
    private String timer;

    private ToggleButton modesToggleButton;
    private ToggleButton timerToggleButton;
    private Button receiverStateButton;
    private TextView receiverStateTextView;
    private LinearLayout timerLayout;
    private DatePicker timerValue;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_modes);

        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_SENSOR_PORTRAIT);

        Bundle b = getIntent().getExtras();

        application = b.getString("application");
        receiverState = "2";
        mode = "SINGLE";
        timer = "0";

        initGUI();
    }

    public void modesToggleButtonPressed(View v)
    {
        if (modesToggleButton.isChecked()) { // Single mode
            mode = "SINGLE";
            receiverStateTextView.setText("RECEIVER STATE :");
        }
```

```
    else { // Multiple mode
        mode = "MULTIPLE";
        receiverStateTextView.setText("RECEIVERS STATE :");
    }
}

public void receiverStateButtonPressed(View v)
{
    if (receiverState.equals("0")) {
        receiverState = "2";
        receiverStateButton.setText("ALL");
        receiverStateButton.setBackgroundColor(Color.rgb(100, 0, 255));
    }
    else if (receiverState.equals("1")) {
        receiverState = "0";
        receiverStateButton.setText("OFFLINE");
        receiverStateButton.setBackgroundColor(Color.rgb(255,0,0));
    }
    else if (receiverState.equals("2")) {
        receiverState = "1";
        receiverStateButton.setText("ONLINE");
        receiverStateButton.setBackgroundColor(Color.rgb(0, 255, 150));
    }
}

public void timerToggleButtonPressed(View v)
{
    if (timerToggleButton.isChecked()) {
        timer = timerValue.getYear() + "-" + timerValue.getMonth() + "-" +
         timerValue.getDayOfMonth() + " " + "23:59:59";
         timerLayout.setVisibility(View.VISIBLE);
    }
    else {
        timer = "0";
        timerLayout.setVisibility(View.GONE);
    }
}

public void continueButtonPressed(View v)
{
    Intent intent = new Intent(v.getContext(), ConnectedDevicesActivity.class);
    Bundle b = new Bundle();

    b.putString("application", application);
    b.putString("mode", mode);
    b.putString("receiverState", receiverState);
    b.putString("timer", timer);

    intent.putExtras(b);

    startActivity(intent);
}

private void initGUI()
{
    modesToggleButton = (ToggleButton)findViewById(R.id.modesToggleButton);
    timerToggleButton = (ToggleButton)findViewById(R.id.timerToggleButton);
    receiverStateButton = (Button)findViewById(R.id.receiverStateButton);
    receiverStateTextView = (TextView)findViewById(R.id.receiverStateTextView);
    timerLayout = (LinearLayout)findViewById(R.id.timerLayout);
    timerValue = (DatePicker)findViewById(R.id.datePicker);

    modesToggleButton.setChecked(true);
```

```
        modesToggleButton.setText("SINGLE RECEIVER");
        modesToggleButton.setTextOn("SINGLE RECEIVER");
        modesToggleButton.setTextOff("MULTIPLE RECEIVERS");

        receiverStateButton.setText("ALL");
        receiverStateButton.setBackgroundColor(Color.rgb(100, 0, 255));

        timerLayout.setVisibility(View.GONE);
    }
}
```

A.11 Actions.java

```
package com.ppapadop.rac;

public class Actions
{
    private int actionId;
    private String senderId;
    private String receiverId;
    private String action;
    private int state;

    public Actions() {}

    public int getActionId() { return actionId; }
    public String getSenderId() { return senderId; }
    public String getReceiverId() { return receiverId; }
    public String getAction() { return action; }
    public int getState() { return state; }

    public void setActionId(int item) { actionId = item; }
    public void setSenderId(String item) { senderId = item; }
    public void setReceiverId(String item) { receiverId = item; }
    public void setAction(String item) { action = item; }
    public void setState(int item) { state = item; }
}
```

A.12 Privileges.java

```
package com.ppapadop.rac;


public class Privileges
{
    private String senderId;
    private String receiverId;
    private String alarm;
    private String map;
    private String senderName;
    private String receiverName;

    public Privileges() {}

    public String getSenderId() { return senderId; }
    public String getReceiverId() { return receiverId; }
    public String getAlarm() { return alarm; }
```

```
    public String getMap() { return map; }
    public String getSenderName() { return senderName; }
    public String getReceiverName() { return receiverName; }

    public void setSenderId(String item) { senderId = item; }
    public void setReceiverId(String item) { receiverId = item; }
    public void setAlarm(String item) { alarm = item; }
    public void setMap(String item) { map = item; }
    public void setSenderName(String item) { senderName = item; }
    public void setReceiverName(String item) { receiverName = item; }
}
```

A.13 ActionsReceiver.java

```
package com.ppapadop.rac;

import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.location.Location;
import android.location.LocationManager;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.provider.AlarmClock;
import android.provider.Settings;
import android.support.v4.app.NotificationCompat;
import android.util.Log;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.List;
import java.util.Locale;
import java.util.StringTokenizer;


public class ActionsReceiver extends BroadcastReceiver
{
    private String sql = null;
    private String deviceId = "";
```

```
private String dbUrl;
private String methodName;
private String android_id;

private ArrayList<Actions> actions;

private Context gContext;
private NotificationManager nm;

static final int uniqueNotificationID = 341249;

@Override
public void onReceive(Context context, Intent intent)
{
    gContext = context;

    android_id = Settings.Secure.getString(context.getContentResolver(),
     Settings.Secure.ANDROID_ID);

    Bundle b = intent.getExtras();

    deviceId = b.getString("deviceId");
    dbUrl = MainActivity.dbUrl;

    nm = (NotificationManager) gContext.getSystemService(
     Context.NOTIFICATION_SERVICE);

    ConnectivityManager connManager = (ConnectivityManager)
     gContext.getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo mWifi = connManager.getNetworkInfo(ConnectivityManager.TYPE_WIFI);
    NetworkInfo mMobile = connManager.getNetworkInfo(
     ConnectivityManager.TYPE_MOBILE);

    if (mWifi.isConnected() || mMobile.isConnected()) {
        if (deviceId.equals("")) {
            // Get database url
            setDatabaseUrl();
            //setDeviceId();
        } else {
            runDownloadQuery();
        }
    }
    else
        Log.e("ACTION RECEIVER", "CONNECTION ERROR");
}

void setDatabaseUrl()
{
    sql = "SELECT database_url FROM DatabaseInfo";
    methodName = "downloadDatabaseUrl";

    new ActionsReceiverWebServiceTask().execute(sql, "down_service.php");
}

void downloadDatabaseUrl(String result)
{
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            if (jsonArray.length() > 0) {
                JSONObject jsonObject = jsonArray.getJSONObject(0);
```

```
                // Set database url
                dbUrl = jsonObject.getString("database_url");
            }
        } catch (Exception e) {
            Log.e("Main database url json", "Error parsing data " + e.toString());
        }
    }
}

void setDeviceId()
{
    sql = "SELECT device_id FROM Devices WHERE android_id='" + android_id + "'";

    methodName = "setDeviceId";

    new ActionsReceiverWebServiceTask().execute(sql, "down_service.php");
}

void downloadDeviceIdFromDatabase(String result)
{
    if (result != null) {
        try {
            // Parse json data
            JSONArray jsonArray = new JSONArray(result);

            if (jsonArray.length() > 0) {
                JSONObject jsonObject = jsonArray.getJSONObject(0);

                // Set device id
                deviceId = jsonObject.getString("device_id");
            }
        } catch (Exception e) {
            Log.e("ActionR Dev json", "Error parsing data " + e.toString());
        }
    }
}

void downloadActionsDataFromDatabase(String result)
{
    int i;
    Actions item;

    actions = new ArrayList<>();

    try {
        // Parse json data
        JSONArray jsonArray = new JSONArray(result);
        JSONObject jsonObject;

        for (i=0;i<jsonArray.length();i++) {
            jsonObject = jsonArray.getJSONObject(i);

            item = new Actions();

            if (!jsonObject.isNull("action_id")) {
                item.setActionId(jsonObject.getInt("action_id"));
                item.setSenderId(jsonObject.getString("sender_id"));
                item.setAction(jsonObject.getString("action"));
                item.setState(jsonObject.getInt("completed"));

                actions.add(item);
            }
```

```
            }
        }
        catch (Exception e) {
            Log.e("ActionR Act json", "Error parsing data " + e.toString());
        }
    }

    void runDownloadQuery()
    {
        sql = "SELECT DISTINCT Actions.action_id, Actions.sender_id, Actions.action,
                 ActionsTargets.completed " +
                "FROM Actions " +
                "JOIN ActionsTargets " +
                "ON Actions.action_id=ActionsTargets.action_id " +
                "WHERE ActionsTargets.receiver_id = '" + deviceId + "' &&
                 ActionsTargets.completed='0'";

        methodName = "runDownloadQuery";
        new ActionsReceiverWebServiceTask().execute(sql, "down_service.php");
    }

    void updateWithCompletedActions()
    {
        int i;
        String actionsId = "";

        for(i=0;i<actions.size();i++) {
            if (i < actions.size()-1)
                actionsId = actionsId+("action_id=" + actions.get(i).getActionId() +
                 " || ");
            else
                actionsId = actionsId + ("action_id="+actions.get(i).getActionId());
        }

        sql = "UPDATE ActionsTargets SET completed='1' WHERE (" + actionsId + ") &&
         receiver_id=" + deviceId;

        new ActionsReceiverWebServiceTask().execute(sql, "up_service.php");
    }

    void runActions()
    {
        int i;
        String parsedAction;
        StringTokenizer st;

        for(i=0;i<actions.size();i++) {
            st = new StringTokenizer(actions.get(i).getAction(), "-");

            parsedAction = st.nextToken();

            if(parsedAction.equals("alarm")) {
                runAlarmAction(actions.get(i).getAction());
            }
            if(parsedAction.equals("map")) {
                runMapAction(actions.get(i).getAction());
            }
        }
    }

    public void runAlarmAction(String actionString)
    {
        StringTokenizer st = new StringTokenizer(actionString, "-");
```

```
    Intent addAlarm = new Intent(AlarmClock.ACTION_SET_ALARM);

    addAlarm.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

    st.nextToken();

    addAlarm.putExtra(AlarmClock.EXTRA_MESSAGE, st.nextToken());
    addAlarm.putExtra(AlarmClock.EXTRA_HOUR, Integer.parseInt(st.nextToken()));
    addAlarm.putExtra(AlarmClock.EXTRA_MINUTES, Integer.parseInt(st.nextToken()));
    addAlarm.putExtra(AlarmClock.EXTRA_SKIP_UI, true);

    gContext.startActivity(addAlarm);
}

public void runMapAction(String actionString)
{
    StringTokenizer st = new StringTokenizer(actionString, "-");

    String uri;
    LocationManager locationManager = (LocationManager)
     gContext.getSystemService(Context.LOCATION_SERVICE);
    Location location;
    NotificationCompat.Builder builder = new NotificationCompat.Builder(gContext);

    st.nextToken();
    st.nextToken();


    if (locationManager != null) {
        location = locationManager.getLastKnownLocation(
         LocationManager.NETWORK_PROVIDER);

        if (location != null)
            uri = "http://maps.google.com/maps?saddr=" + location.getLatitude() +
             "," + location.getLongitude() + " (My Location)" + "&daddr=" +
             st.nextToken() + "," + st.nextToken() + " (Meeting Location)";
        else
            uri = String.format(Locale.ENGLISH, "geo:0,0?q=" + st.nextToken() +
             "," + st.nextToken() + "(Meeting Location)");
    }
    else
        uri = String.format(Locale.ENGLISH, "geo:0,0?q=" + st.nextToken() + "," +
         st.nextToken() + "(Meeting Location)");


    builder.setSmallIcon(R.drawable.icon);
    builder.setContentTitle("RAC Notification");
    builder.setContentText("Meet me here.");

    Notification notification = builder.build();

    Intent addMap = new Intent(Intent.ACTION_VIEW, Uri.parse(uri));

    addMap.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

    PendingIntent pi = PendingIntent.getActivity(gContext, 0, addMap, 0);

    notification.defaults = Notification.DEFAULT_ALL;
    notification.contentIntent = pi;

    nm.notify(uniqueNotificationID, notification);
}
```

```
class ActionsReceiverWebServiceTask extends AsyncTask<String, String, String>
{
    InputStream isr = null;
    String result = null;
    String sql;
    String phpFile;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
    }

    @Override
    protected String doInBackground(String... params)
    {
        sql = params[0];
        phpFile = params[1];

        result = runWebService();

        return null;
    }

    protected void onProgressUpdate(String... progress) {
    }

    @Override
    protected void onPostExecute(String file_url) {
        if (phpFile.equals("down_service.php")) {
            if (methodName.equals("setDeviceId")) {
                downloadDeviceIdFromDatabase(result);
                runDownloadQuery();
            }

            if (methodName.equals("runDownloadQuery")) {

                downloadActionsDataFromDatabase(result);

                runActions();

                updateWithCompletedActions();
            }

            if (methodName.equals("downloadDatabaseUrl")) {

                downloadDatabaseUrl(result);

                setDeviceId();
            }
        }

    }

    String runWebService()
    {
        List<NameValuePair> list = new ArrayList<>(1);

        list.add(new BasicNameValuePair("sql", sql ));

        // Setting up the connection
        try {
            HttpClient httpClient = new DefaultHttpClient();
```

```
            HttpPost httpPost = new HttpPost(dbUrl + phpFile);

            httpPost.setEntity(new UrlEncodedFormEntity(list));

            HttpResponse response = httpClient.execute(httpPost);

            HttpEntity entity = response.getEntity();
            isr = entity.getContent();
        }
        catch(ClientProtocolException e) {
            Log.e("ActionR ClientProtocol", e.toString());
            e.printStackTrace();

            result = "0";
        }
        catch (IOException e) {
            Log.e("ActionR IOException", e.toString());
            e.printStackTrace();

            result = "0";
        }

        // Convert the response to string
        try {
            InputStreamReader inputReader = new InputStreamReader(isr,"iso-8859-1");
            BufferedReader reader = new BufferedReader(inputReader, 8);
            StringBuilder sb = new StringBuilder();
            String line;

            while ((line = reader.readLine()) != null) {
                sb.append(line).append("\n");
            }
            isr.close();

            result = sb.toString();
        }
        catch(Exception e) {
            Log.e("ActionR Convert", e.toString());
        }

        return result;
    }
  }
}
```

ΠΑΡΑΡΤΗΜΑ Β : CLIENT LAYOUT

B.1 activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    android:background="@drawable/back_fill">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:id="@+id/linearLayout">

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:gravity="center"
            android:id="@+id/textView"
            android:singleLine="false"
            android:phoneNumber="false"
            android:textStyle="bold|italic"
            android:layout_below="@+id/linearLayout"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginTop="10dp"
            android:textColor="#ff000000"
            android:textSize="30dp"
            android:background="@drawable/title" />

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="APPLICATIONS"
            android:id="@+id/applicationsButton"
            android:onClick="applicationsButtonPressed"
            android:textColor="#ffffffff"
            android:textStyle="bold|italic"
            android:background="@drawable/button"
            android:layout_marginTop="60dp"
            android:textSize="20dp" />

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
```

```
            android:text="DEVICES"
            android:id="@+id/trustedDevicesButton"
            android:onClick="trustedDevicesButtonPressed"
            android:textColor="#ffffffff"
            android:background="@drawable/button"
            android:textStyle="bold|italic"
            android:layout_marginTop="40dp"
            android:textSize="20dp" />

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="SETTINGS"
            android:id="@+id/settingsButton"
            android:textColor="#ffffffff"
            android:background="@drawable/button"
            android:textStyle="bold|italic"
            android:layout_marginTop="40dp"
            android:textSize="20dp"
            android:onClick="settingsButtonPressed" />
    </LinearLayout>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="DEVICE ID :   "
        android:id="@+id/textView2"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:textColor="#ffffffff"
        android:textStyle="bold"
        android:textSize="20dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:text="@string/deviceId"
        android:id="@+id/deviceIdTextView"
        android:layout_alignParentBottom="true"
        android:textColor="#ffff0000"
        android:layout_alignTop="@+id/textView2"
        android:layout_toRightOf="@+id/textView2"
        android:layout_toEndOf="@+id/textView2"
        android:textStyle="bold|italic"
        android:textSize="20dp" />

    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

B.2 activity_trusted_devices.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.TrustedDevicesActivity"
    android:background="@drawable/back_white">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_above="@+id/addDevice">

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="DEVICES"
            android:id="@+id/textView7"
            android:layout_gravity="center_horizontal"
            android:layout_alignParentTop="true"
            android:layout_alignParentLeft="false"
            android:layout_marginTop="10dp"
            android:layout_alignParentRight="false"
            android:gravity="center"
            android:textStyle="bold|italic"
            android:textColor="#ffffffff"
            android:textSize="30dp" />

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="40dp"
            android:id="@+id/labelsLinealLayout"></LinearLayout>

        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/scrollView"
            android:layout_below="@+id/textView7"
            android:layout_marginTop="15dp">

            <LinearLayout
                android:orientation="horizontal"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent"
                android:id="@+id/linealLayout">

            </LinearLayout>
        </ScrollView>

    </LinearLayout>

    <Button
        android:layout_width="fill_parent"
```

```
        android:layout_height="wrap_content"
        android:text="ADD DEVICE"
        android:id="@+id/addDevice"
        android:layout_gravity="center_horizontal"
        android:onClick="addDeviceButtonPressed"
        android:background="@drawable/button"
        android:textColor="#ffffffff"
        android:textSize="20dp"
        android:textStyle="bold|italic"
        android:layout_alignParentBottom="true"
        android:layout_marginTop="15dp" />

    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"
        android:layout_weight="0.10"
        android:layout_centerInParent="true" />

</RelativeLayout>
```

B.3 activity_connected_devices.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.ConnectedDevicesActivity"
    android:background="@drawable/back_white">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_above="@+id/setActionbutton">

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="DEVICES"
            android:id="@+id/textView3"
            android:layout_marginTop="10dp"
            android:textStyle="bold|italic"
            android:textSize="30dp"
            android:gravity="center" />

        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/devicesScrollView"
            android:layout_marginTop="40dp">

            <LinearLayout
                android:orientation="vertical"
                android:layout_width="fill_parent"
```

```
            android:layout_height="fill_parent"
            android:id="@+id/connectedDevicesLayout"></LinearLayout>
    </ScrollView>

</LinearLayout>

<ProgressBar
    style="?android:attr/progressBarStyleLarge"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/spinner"
    android:layout_centerInParent="true" />

<LinearLayout
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:id="@+id/popupViewLayout"
    android:background="@color/primary_text_disabled_material_light"
    android:gravity="center"
    android:visibility="gone">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="Enter Device Name"
        android:id="@+id/textView15"
        android:gravity="center"
        android:textStyle="bold|italic"
        android:textSize="25dp" />

    <EditText
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:text="Name"
        android:ems="10"
        android:id="@+id/nameTextView"
        android:gravity="center"
        android:layout_marginTop="15dp"
        android:textStyle="italic"
        android:textSize="20dp" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="OK"
        android:id="@+id/setNameButton"
        android:layout_marginTop="20dp"
        android:textStyle="bold"
        android:textSize="22dp"
        android:background="@drawable/button"
        android:onClick="setReceiverName" />
</LinearLayout>

<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="SET ACTION"
    android:id="@+id/setActionbutton"
    android:onClick="setActionButtonPressed"
    android:textColor="#ffffffff"
```

```
        android:textStyle="bold|italic"
        android:background="@drawable/button"
        android:textSize="20dp"
        android:layout_alignParentEnd="false"
        android:layout_alignParentBottom="true" />

</RelativeLayout>
```

B.4 activity_settings.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.SettingsActivity"
    android:background="@drawable/back_white">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="SETTINGS"
            android:id="@+id/textView16"
            android:layout_alignParentTop="true"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_alignParentRight="true"
            android:layout_alignParentEnd="true"
            android:layout_marginTop="20dp"
            android:textStyle="bold"
            android:gravity="center"
            android:textSize="30dp" />

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="Receiver Time Interval (seconds)"
            android:id="@+id/textView17"
            android:layout_below="@+id/textView16"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginTop="40dp"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <EditText
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:inputType="number"
            android:ems="10"
            android:id="@+id/intervalEditText"
```

```
            android:layout_below="@+id/textView17"
            android:layout_alignRight="@+id/textView17"
            android:layout_marginTop="10dp" />

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="Database URL"
            android:id="@+id/textView18"
            android:layout_below="@+id/textView16"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true"
            android:layout_marginTop="30dp"
            android:textStyle="bold|italic"
            android:textSize="25dp"
            android:layout_gravity="center_horizontal" />

        <EditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:inputType="textUri"
            android:ems="10"
            android:id="@+id/dbUrlEditText"
            android:layout_below="@+id/textView17"
            android:layout_alignRight="@+id/textView17"
            android:layout_marginTop="10dp"
            android:layout_gravity="center_horizontal" />
    </LinearLayout>
</RelativeLayout>
```

B.5 activity_applications.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.ActionsActivity"
    android:background="@drawable/back_white">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="APPLICATIONS"
        android:gravity="center"
        android:id="@+id/textView4"
        android:layout_marginTop="10dp"
        android:textStyle="bold|italic"
        android:textColor="#ffffffff"
        android:textSize="30dp" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ALARM"
        android:id="@+id/alarmButton"
```

```
    android:onClick="alarmButtonPressed"
    android:background="@drawable/button"
    android:textColor="#ffffffff"
    android:textStyle="bold|italic"
    android:textSize="20dp"
    android:layout_below="@+id/textView4"
    android:layout_marginTop="60dp"
    android:layout_toRightOf="@+id/imageView2"
    android:layout_toLeftOf="@+id/imageView3"
    android:layout_toStartOf="@+id/imageView3" />

<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="MAP"
    android:id="@+id/mapButton"
    android:onClick="mapButtonPressed"
    android:background="@drawable/button"
    android:textColor="#ffffffff"
    android:textStyle="bold|italic"
    android:textSize="20dp"
    android:layout_below="@+id/alarmButton"
    android:layout_marginTop="40dp"
    android:layout_alignLeft="@+id/alarmButton"
    android:layout_alignStart="@+id/alarmButton"
    android:layout_toLeftOf="@+id/imageView4"
    android:layout_toStartOf="@+id/imageView4" />

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/imageView2"
    android:background="@drawable/alarm"
    android:layout_alignTop="@+id/alarmButton"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/imageView"
    android:background="@drawable/map"
    android:layout_alignTop="@+id/mapButton"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true" />

<ProgressBar
    style="?android:attr/progressBarStyleLarge"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/spinner"
    android:layout_centerVertical="true"
    android:layout_centerHorizontal="true"
    android:layout_gravity="center_horizontal" />

<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/imageView3"
    android:background="@drawable/alarm"
    android:layout_alignTop="@+id/alarmButton"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true" />
```

```
    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView4"
        android:background="@drawable/map"
        android:layout_alignTop="@+id/mapButton"
        android:layout_alignLeft="@+id/imageView3"
        android:layout_alignStart="@+id/imageView3" />
</RelativeLayout>
```

B.6 activity_modes.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rac.ModesActivity"
    android:background="@drawable/back_white">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_above="@+id/continueButton">

        <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:weightSum="1">

            <TextView
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:textAppearance="?android:attr/textAppearanceLarge"
                android:text="MODE DETAILS"
                android:gravity="center"
                android:id="@+id/textView20"
                android:layout_marginTop="10dp"
                android:textStyle="bold|italic"
                android:textColor="#ffffffff"
                android:textSize="30dp" />

            <LinearLayout
                android:orientation="horizontal"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="40dp" >

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:textAppearance="?android:attr/textAppearanceLarge"
                    android:text="MODE :"
                    android:id="@+id/textView21"
                    android:textStyle="bold" />
```

```
    <ToggleButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New ToggleButton"
        android:id="@+id/modesToggleButton"
        android:layout_marginLeft="10dp"
        android:checked="true"
        android:onClick="modesToggleButtonPressed"
        android:textSize="20dp"
        android:textStyle="bold|italic" />

</LinearLayout>

<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="RECEIVER STATE :"
        android:id="@+id/receiverStateTextView"
        android:textStyle="bold"
        android:visibility="gone" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ALL"
        android:id="@+id/receiverStateButton"
        android:layout_marginLeft="10dp"
        android:checked="true"
        android:onClick="receiverStateButtonPressed"
        android:textSize="20dp"
        android:textStyle="bold|italic"
        android:visibility="gone" />
</LinearLayout>

<LinearLayout
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="TIMER :"
        android:id="@+id/textView19"
        android:textStyle="bold"
        android:visibility="gone" />

    <ToggleButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="New ToggleButton"
        android:id="@+id/timerToggleButton"
        android:layout_marginLeft="10dp"
```

```
                android:checked="false"
                android:onClick="timerToggleButtonPressed"
                android:textSize="20dp"
                android:textStyle="bold|italic"
                android:visibility="gone" />
        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center_horizontal"
            android:layout_marginTop="10dp"
            android:id="@+id/timerLayout">

            <DatePicker
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:id="@+id/datePicker"
                android:visibility="invisible" />

        </LinearLayout>

    </LinearLayout>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="APPLICATIONS"
        android:id="@+id/continueButton"
        android:onClick="continueButtonPressed"
        android:textColor="#ffffffff"
        android:textStyle="bold|italic"
        android:background="@drawable/button"
        android:textSize="20dp"
        android:layout_alignParentEnd="false"
        android:layout_alignParentBottom="true" />

</RelativeLayout>
```

B.7 activity_add_device.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.ControlDeviceActivity"
    android:background="@drawable/back_white"
    android:id="@+id/mainView">

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="CONTROL DEVICE"
```

```
    android:gravity="center"
    android:id="@+id/textView8"
    android:layout_marginTop="10dp"
    android:textStyle="bold|italic"
    android:textSize="30dp"
    android:textColor="#ffffffff" />

<TableLayout
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:id="@+id/tableLayout"
    android:layout_marginTop="80dp">

    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="10dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="Device ID : "
            android:id="@+id/textView9"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/deviceIdText" />

    </TableRow>

    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="10dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="Name :"
            android:id="@+id/textView13"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <EditText
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/nameText" />
    </TableRow>

    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="10dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
```

```
            android:text="Alarm :"
            android:id="@+id/textView10"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/alarmCheckBox" />
    </TableRow>

    <TableRow
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_marginTop="10dp">

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="Map :"
            android:id="@+id/textView11"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <CheckBox
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/mapCheckBox" />
    </TableRow>

</TableLayout>

<LinearLayout
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="false"
    android:layout_alignParentBottom="true">

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="DELETE DEVICE"
        android:id="@+id/deleteDeviceButton"
        android:onClick="deleteDeviceButtonPressed"
        android:background="@drawable/button"
        android:textColor="#ffffffff"
        android:textStyle="bold|italic"
        android:textSize="20dp" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ADD DEVICE"
        android:id="@+id/addDeviceButton"
        android:onClick="addDeviceButtonPressed"
        android:background="@drawable/button"
        android:textColor="#ffffffff"
        android:textStyle="bold|italic"
        android:textSize="20dp"
        android:layout_marginTop="30dp" />
```

```
    </LinearLayout>

    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"
        android:layout_gravity="center_horizontal"
        android:layout_centerInParent="true" />

</RelativeLayout>
```

B.8 activity_set_alarm.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.SetAlarmActivity"
    android:background="@drawable/back_white">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:onClick="sendActionAddAlarm"
        android:layout_above="@+id/addAlarm">

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="SET ALARM"
            android:id="@+id/textView5"
            android:textStyle="bold|italic"
            android:gravity="center"
            android:layout_marginTop="10dp"
            android:textSize="30dp" />

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:maxLength="100"
            android:text="Message"
            android:id="@+id/textView6"
            android:layout_marginTop="50dp"
            android:textStyle="bold|italic"
            android:textSize="25dp" />

        <EditText
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/message"
            android:layout_marginTop="10dp"
```

```
            android:inputType="text"
            android:gravity="center" />

        <TimePicker
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/timePicker"
            android:layout_gravity="center_horizontal"
            android:layout_marginTop="20dp" />

    </LinearLayout>

    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ADD ALARM"
        android:id="@+id/addAlarm"
        android:onClick="sendActionAddAlarm"
        android:background="@drawable/button"
        android:textStyle="bold|italic"
        android:textSize="20dp"
        android:layout_alignParentBottom="true"
        android:textColor="#ffffffff" />
</RelativeLayout>
```

B.9 activity_set_map.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    tools:context="com.ppapadop.rsapp.SetMapActivity"
    android:background="@drawable/back_white">

    <fragment
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_below="@+id/textView14"
        android:layout_marginTop="10dp"
        android:layout_above="@+id/addMapButton" />

    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="MAP"
```

```
        android:id="@+id/textView14"
        android:textStyle="bold|italic"
        android:gravity="center"
        android:layout_marginTop="10dp"
        android:textSize="30dp" />

    <Button
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="ADD MAP"
        android:id="@+id/addMapButton"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:onClick="sendActionSendPosition"
        android:background="@drawable/button"
        android:textSize="20dp"
        android:textStyle="bold|italic"
        android:textColor="#ffffffff" />

    <ProgressBar
        style="?android:attr/progressBarStyleLarge"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/spinner"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

ΠΑΡΑΡΤΗΜΑ Γ : SERVER

Γ.1 up_service.php

```php
<?php

    $host = "localhost";
    $username = "root";
    $password = "...";
    $dbName = "db_rac";

    $mysqli = new mysqli($host, $username, $password, $dbName);

    // Check connection
    if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
    }

    // Set autocommit to off
    $mysqli->autocommit(FALSE);

    // Get a sql query
    $sql = $_POST['sql'];

    // Run query
    $mysqli->query($sql);

    // Commit transaction
    $mysqli->commit();

    // Close connection
    $mysqli->close();

?>
```

Γ.2 down_service.php

```php
<?php

    $host = "localhost";
    $username = "root";
    $password = "...";
    $dbName = "db_rac";
```

```php
    $mysqli = new mysqli($host, $username, $password, $dbName);

    // Check connection
    if (mysqli_connect_errno()) {
        printf("Connect failed: %s\n", mysqli_connect_error());
        exit();
    }

    // Set autocommit to off
    $mysqli->autocommit(FALSE);

    // Get a sql query
    $sql = $_POST['sql'];

    $dataArray = array();

    // Check if there are results
    if ($result = $mysqli->query($sql)) {
        $tempArray = array();

        while($row = $result->fetch_object()) {
            $tempArray = $row;
            array_push($dataArray, $tempArray);
        }


        echo json_encode($dataArray);
    }

    $result->close();

    // Commit transaction
    $mysqli->commit();

    // Close connection
    $mysqli->close();


?>
```