



HELLENIC REPUBLIC

**National and Kapodistrian  
University of Athens**

EST. 1837



DEPARTMENT OF  
INFORMATICS &  
TELECOMMUNICATIONS

## README

Ευστρατία Ευαγγελινού 1115201500038

Σοφία Καζαντζίδου 1115201500051

Ηλίας Καλαμάτας 1115201400053

Ακαδημαϊκό Έτος 2020-2021





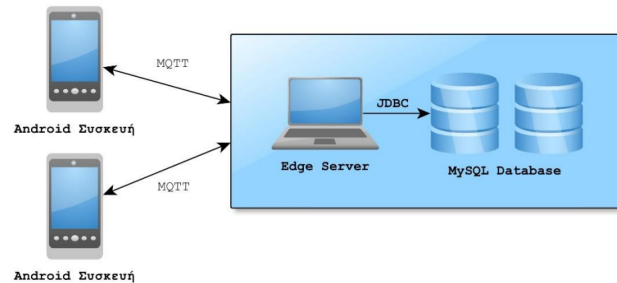
## Περιεχόμενα

<b>1</b>	<b>Εισαγωγή</b>	<b>3</b>
1.1	Σχετικά με το Project	3
1.2	External Libraries	3
<b>2</b>	<b>Edge Server</b>	<b>3</b>
2.1	Heatmaps	3
2.1.1	Διαχωρισμός Αρχείων	3
2.1.2	Υλοποίηση	4
2.1.3	Αποτέλεσμα Εκτέλεσης	4
2.2	MQTT Connection	4
2.2.1	Διαχωρισμός Αρχείων	4
2.3	Διαδικασία Επεξεργασίας Δεδομένων	5
2.3.1	Κλάσεις	5
2.3.2	Υλοποίηση	5
2.4	Οπτικοποίηση Δεδομένων	6
<b>3</b>	<b>Android Terminal</b>	<b>6</b>
3.1	Διαχωρισμός Αρχείων	6
3.2	Στοιχεία Υλοποίησης	6
3.3	Περιήγηση στην Εφαρμογή	7
3.4	Στιγμιότυπα Εφαρμογής	7
<b>4</b>	<b>MySQL Base</b>	<b>9</b>



# 1 Εισαγωγή

Σκοπός της παρούσας εργασίας είναι η ανάπτυξη ενός συστήματος παρακολούθησης κίνησης οχημάτων και πρόβλεψης της πορείας τους καθώς και της απόδοσης του δικτύου καθόλη τη διαδρομή τους. Τα οχήματα θα κινούνται σε ένα τμήμα του δρόμου της Πανεπιστημιούπολης και η κίνησή τους θα προσομοιωθεί με τη χρήση του εργαλείου SUMO (Simulation of Urban Mobility).



Εικόνα 1: Αρχιτεκτονική Δομικών Στοιχείων

## 1.1 Σχετικά με το Project

- Η εκτέλεση των εφαρμογών δεν απαιτεί την προσθήκη ορισμάτων.
- Η συνεργασία στα πλαίσια της ομάδας έγινε μέσω share screen sessions. Η συγγραφή της πλειοψηφίας του κώδικα έχει γίνει από όλα τα μέλη της ομάδας.
- Στο GitLab έχουμε δημιουργήσει ένα project στο οποίο υπάρχουν δύο φάκελοι, ένας για τον Edge Server κι ένας για το Android Terminal.

## 1.2 External Libraries

1. **jheatmap**: χρησιμοποιείται για τη δημιουργία των Heat Maps.
2. **paho client mqttv**: είναι η MQTT client βιβλιοθήκη, που χρειαζόμαστε για την υλοποίηση της εφαρμογής.
3. **mysql-connector-java**: απαραίτητη για τη σύνδεση με τη βάση δεδομένων.

# 2 Edge Server

Σε αυτό το κομμάτι του Project περιέχονται πακέτα Java, που αφορούν το κομμάτι της σύνδεσης και λειτουργίας του Edge Server, της δημιουργίας των Heat Maps και της δημιουργίας προβλέψεων πορείας και ποιότητας δικτύου για τα τεμαχικά.

## 2.1 Heatmaps

### 2.1.1 Διαχωρισμός Αρχείων

Έχουμε 3 βασικούς φακέλους:

- **InputData**: όπου βρίσκονται όλα τα δεδομένα, που χρειαζόμαστε για την λειτουργία του εκτελεστή
- **OutputData**: στον οποίο αποθηκεύουμε τα αρχεία, που παράγουμε
- **src**: όπου υπάρχουν όλες οι κλάσεις, που δημιουργήσαμε για τη παραγωγή του Heat Map και χωρίζονται ως εξής:
  - **CreateCSV**: Εδώ υπάρχουν 3 συνάρτησεις, μία συνάρτηση για την παραγωγή των RSSI τιμών και μία αντίστοιχα για τις τιμές του Throughput καθώς και μία συνάρτηση για την μετατροπή των xml αρχείων σε αρχεία csv.
  - **HeatMap**: Σε αυτήν την κλάση υπάρχουν πολλά χρήσιμα πεδία, όπως οι μέγιστες και ελάχιστες επιτρεπτές τιμές του χάρτη, πίνακες που κρατάνε απαραίτητα δεδομένα, καθώς και διάφορες μέθοδοι, όπως για παράδειγμα η παραγωγή των μέσων τιμών RSSI και Throughput και η δημιουργία του Heat Map από αυτές.
  - **HeatMapUtil**: Εδώ υπάρχουν συναρτήσεις γενικού τύπου, όπως για παράδειγμα συναρτήσεις επεξεργασίας εικόνας, αρχικοποίησης κλάσεων, συνδυασμού της εικόνας του χάρτη με το Heat Map κτλ.

### 2.1.2 Υλοποίηση

Η λογική που ακολουθήσαμε για την υλοποίηση του HeatMap είναι η εξής:

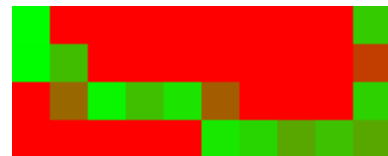
1. Διατρέχουμε τα 3 xml αρχεία και αποθηκεύουμε τις τιμές, που μας ενδιαφέρουν στα αντίστοιχα csv αρχεία, παράγοντας παράλληλα τις τιμές RSSI και Throughput με βάση την κανονική κατανομή. (Θεωρήσαμε ως μέση τιμή 60 και ως διασπορά 13)
2. Αρχικοποιούμε δύο κλάσεις Heat Map. (Μία για το RSSI και μία για το Throughput)
3. Διαβάζουμε το παραγόμενο csv αρχείο γραμμή προς γραμμή και:  
(α') Ελέγχουμε αν το σημείο βρίσκεται μέσα στο χάρτη.  
(β') Αν ικανοποιείται το παραπάνω, κάνουμε ενημέρωση του πλήθους και των τιμών RSSI και Throughput στα αντίστοιχα κελιά του grid.
4. Παράγουμε τις μέσες τιμές των RSSI και Throughput, που υπολογίσαμε στο βήμα 3.
5. Δημιουργούμε μία εικόνα HeatMap για το καθένα. (Χρήση της βιβλιοθήκης *jheatmap*)
6. Συνδυάζουμε την εικόνα του χάρτη της Πανεπιστημιούπολης μαζί με τα παραγόμενα Heat Maps, ώστε να δημιουργήσουμε τις τελικές εικόνες, που χρειαζόμαστε.

### 2.1.3 Αποτέλεσμα Εκτέλεσης

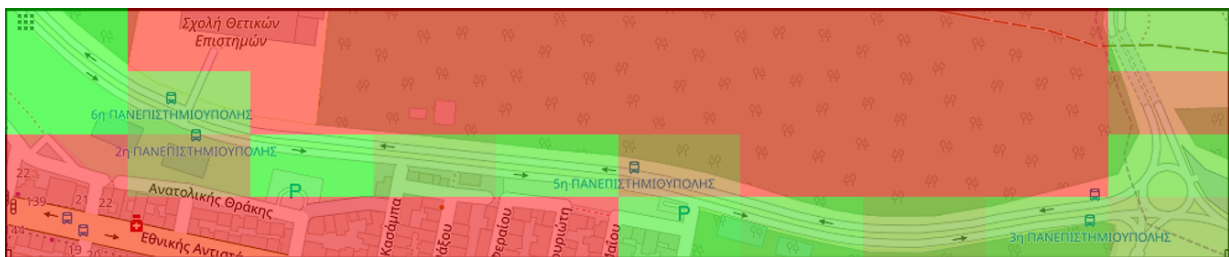
Παρακάτω παρουσιάζουμε τα παραγόμενα αρχεία μιας ενδεικτικής εκτέλεσης του προγράμματος. Η εικόνα 3 είναι το αποτέλεσμα του βήματος 5 και η εικόνα 4 του βήματος 6.



Εικόνα 2: Χάρτης Πανεπιστημιούπολης



Εικόνα 3: Παραγόμενο Heat Map



Εικόνα 4: Συνδυασμός των δύο παραπάνω εικόνων

## 2.2 MQTT Connection

### 2.2.1 Διαχωρισμός Αρχείων

Σε αυτό το σημείο του Project περιλαμβάνονται οι 3 παρακάτω κλάσεις:

- **MQTTInfo:** πραγματοποιεί αρχικοποιήσεις (π.χ. URL, Topic Names)
- **MQTTPublisher:** χρησιμοποιείται για την αποστολή των προβλέψεων στα Android τερματικά
- **MQTTSubscriber:** διαβάζει τα δεδομένα, που λαμβάνονται από τα Android τερματικά

## 2.3 Διαδικασία Επεξεργασίας Δεδομένων

### 2.3.1 Κλάσεις

Για τη διαδικασία επεξεργασίας δεδομένων έχουν δημιουργηθεί στο `src/Predictions` οι εξής κλάσεις:

- **PredictionData**: Σε αυτήν την κλάση έχουμε πεδία, που αντιστοιχούν στις στήλες του πίνακα της βάσης δεδομένων. Επίσης, έχουμε κάποιες βοηθητικές τιμές, που χρησιμεύουν στον υπολογισμό του μέσου λάθους απόκλισης πραγματικών και προβλεπόμενων δεδομένων.
- **PredictionDataUtil**: Εδώ έχουν υλοποιηθεί βοηθητικές συναρτήσεις οι οποίες αφορούν τους υπολογισμούς των προβλέψεων, τον υπολογισμό του mean error και τον έλεγχο αν όλα τα οχήματα έχουν ολοκληρώσει την αποστολή δεδομένων.
- **DatabaseUtil**: Σε αυτήν την κλάση υλοποιείται η σύνδεση με τη βάση δεδομένων. Περισσότερες λεπτομέρειες παρουσιάζονται στην [ενότητα 4](#).
- **BarChartCreator**: Εδώ δημιουργείται ένα bar chart το οποίο χρησιμοποιείται για την οπτική παρουσίαση των mean errors κάθε οχήματος, καθώς και ενός γενικού.

### 2.3.2 Υλοποίηση

Για την διαδικασία της δημιουργίας των προβλέψεων ακολουθείται η παρακάτω διαδικασία:

1. Αρχικοποίηση μιας κενής λίστας τύπου Prediction Data.
2. Για κάθε μήνυμα, που λαμβάνεται από κάποιο τερματικό, κάνουμε extract το id και υπολογίζουμε βάσει της φόρμουλας, που μας δόθηκε, τις προβλεπόμενες τιμές σχετικά με τη θέση του οχήματος και χρησιμοποιούμε τα παραγόμενα heatmaps, για τη λήψη των τιμών του throughput και RSSI.

Formula

$$\delta = \frac{t * \text{speed}}{R}, (\text{για την άσκηση } t = 1)$$

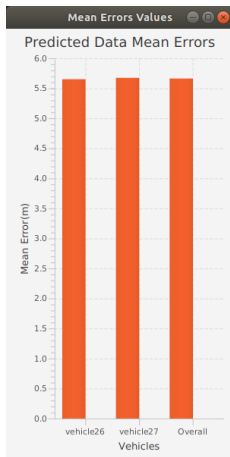
$$\text{lat}_{\text{end}} = \arcsin(\sin(\text{lat}_{\text{start}}) * \cos(\delta) + \cos(\text{lat}_{\text{start}}) * \sin(\delta) * \cos(\text{angle}))$$

$$\text{long}_{\text{end}} = \text{long}_{\text{start}} + \arctan2(\sin(\text{angle}) * \sin(\delta) * \cos(\text{lat}_{\text{start}}), \cos(\delta) - \sin(\text{lat}_{\text{start}}) * \sin(\text{lat}_{\text{end}}))$$

Εικόνα 5: Φόρμουλες πρόβλεψης συντεταγμένων

3. Αν το id δεν υπάρχει στη λίστα τότε:
  - (α') Δημιουργούμε ένα αντικείμενο της κλάσης Prediction Data.
  - (β') Το προσθέτουμε στη λίστα.
  - (γ') Ενημερώνουμε τις προβλεπόμενες τιμές για το επόμενο στιγμιότυπο.
4. Αλλιώς:
  - (α') Ενημερώνουμε τις πραγματικές τιμές για το τρέχον στιγμιότυπο.
  - (β') Ενημερώνουμε τις σχετικές τιμές για τον υπολογισμό του mean error.
  - (γ') Εισάγουμε τα δεδομένα στη βάση.
  - (δ') Κάνουμε publish τα δεδομένα στο αντίστοιχο topic.
  - (ε') Ενημερώνουμε τις προβλεπόμενες τιμές για το επόμενο στιγμιότυπο.

## 2.4 Οπτικοποίηση Δεδομένων



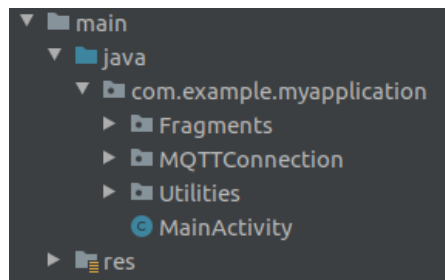
Εικόνα 6: Bar Chart

- Όταν ένα τερματικό στείλει ένα μήνυμα τύπου 'END', τότε γνωρίζει ο Edge Server πως έχει τερματίσει η αποστολή δεδομένων (είτε λόγω ολοκλήρωσης του αρχείου είτε λόγω τερματισμού από τον χρήστη).
- Σε αυτήν την περίπτωση εκτυπώνει το μέσο λάθος προβλέψεων για το συγκεκριμένο τερματικό.
- Όταν τελειώσει η αποστολή όλων των τερματικών, τότε εμφανίζεται ένα Bar Chart, που οπτικοποιεί τόσο τα mean errors κάθε τερματικού όσο και ένα γενικό μέσο όρο τους.

## 3 Android Terminal

### 3.1 Διαχωρισμός Αρχείων

- Fragments: Περιέχει τα αρχεία HomeFragment και SettingsFragment. Υπάρχουν επιπλέον τρία αρχεία. Ένα για την αναπαράσταση του χάρτη με τις πραγματικές θέσεις, ένα για την αναπαράσταση του χάρτη με τις προβλεπόμενες θέσεις και ένα για το χάρτη και με τις δύο θέσεις. (αντίστοιχα RealFragment, PredictionsFragment, MapsFragment)
- MQTTConnection: Ανάλογη λογική με την [υποενότητα 2.2](#)
- Utilities: Περιέχει γενικού τύπου κλάσεις, όπως εκείνη για τον έλεγχο σύνδεσης στο διαδίκτυο και εκείνη για την ανάγνωση των δεδομένων του csv αρχείου.
- res: Περιέχει αρχεία τύπου png και xml.



Εικόνα 7: Διαχωρισμός Αρχείων Android

### 3.2 Στοιχεία Υλοποίησης

- Στο HomeFragment περιέχεται ο listener, που είναι συνδεδεμένος με το κουμπί που υλοποιείται στο fragment\_home.xml. Ανάλογα με την κατάσταση στην οποία βρισκόμαστε γίνεται και η αντίστοιχη ενέργεια. Δηλαδή, είτε διακόπτεται εντελώς η διαδικασία από το κουμπί Stop ή μπορεί να διακοπεί προσωρινά η διαδικασία (Pause Process) ή αν είναι σε Pause να συνεχιστεί (Continue Process). Όταν η διαδικασία τερματιστεί, είτε από το κουμπί Stop είτε επειδή στάλθηκαν όλα τα δεδομένα, δεν μπορεί να ξεκινήσει πάλι από την αρχή. (Επειδή πρέπει να σταλθεί ένα κατάλληλο μήνυμα στον Edge Server, για να γνωρίζει τότε να εμφανίσει το BarChart. Δηλαδή όταν λάβει κατάλληλο μήνυμα από όλα τα τερματικά.)
- Για την υλοποίηση του μενού χρησιμοποιήσαμε το Navigation Bar του Android Studio κάνοντας μερικές τροποποιήσεις. Επιπλέον η έξοδος από την εφαρμογή γίνεται στο μενού και όχι στις ρυθμίσεις.
- Στο SettingsFragment γίνεται extend η κλάση PreferenceFragmentCompat και γίνεται implement το SharedPreferences.OnSharedPreferenceChangeListener. Πιο συγκεκριμένα στην συνάρτηση onSharedPreferenceChanged ανάλογα με την αλλαγή, που γίνεται στο αντίστοιχο πεδίο (το οποίο είναι ένα EditTextPreference, που βρίσκεται στο αρχείο res/xml/settings.xml) επηρεάζεται η αντίστοιχη ρύθμιση.



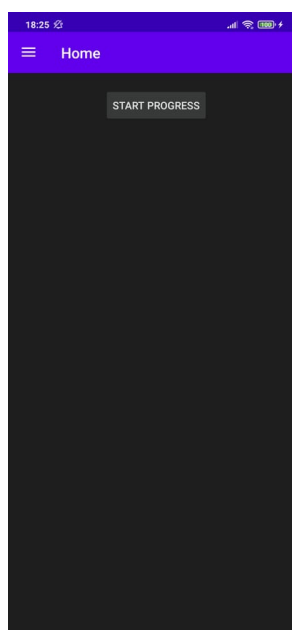
- Έχουμε δύο static arrays. Ένα για τις πραγματικές θέσεις και ένα για τις προβλεπόμενες. Κάθε φορά που θέλουμε να δημιουργηθεί ένα νέο marker στο χάρτη δημιουργούμε ένα MarkerOptions και το προσθέτουμε στον αντίστοιχο πίνακα. Υπάρχει ένας έλεγχος σε κάθε χάρτη, ο οποίος κάθε δευτερόλεπτο ανανεώνει τα σημεία με βάσει τα δεδομένα στα arrays.

### 3.3 Περιήγηση στην Εφαρμογή

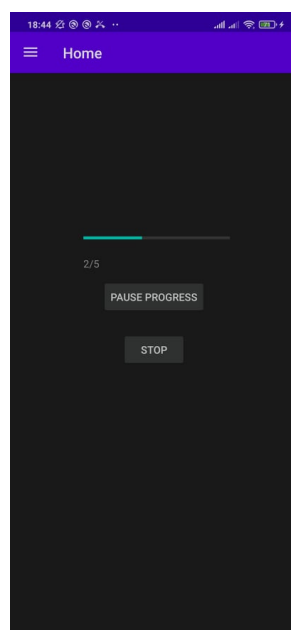
- Ανοίγοντας την εφαρμογή εμφανίζεται η κύρια οθόνη. Σε αυτήν υπάρχει ένα κουμπί, με το οποίο ξεκινάει η διαδικασία αποστολής δεδομένων. (Εικ.8)
- Ξεκινώντας τη διαδικασία της αποστολής εμφανίζεται ένα progress bar. Υπάρχει η δυνατότητα παύσης της αποστολής και τερματισμό της αποστολής. (Εικ.9)
- Στην περίπτωση παύσης της αποστολής δίνεται η δυνατότητα να συνεχίσει ο χρήστης την αποστολή ή να πάλι να την τερματίσει (Εικ. 18)
- Πατώντας τις 3 γραμμές πάνω αριστερά ανοίγει το μενού. Σε αυτό υπάρχουν έξι επιλογές. (Εικ.10)
  - Home: Μετάβαση στην κύρια οθόνη
  - Settings: Μετάβαση στις ρυθμίσεις
  - Overall Map: Γενικός Χάρτης (Εικ. 16)
  - Real route Map: Χάρτης Πραγματικής Θέσης (Εικ.14)
  - Predicted route Map: Χάρτης Προβλεπόμενης Θέσης (Εικ.15)
  - Exit: Έξοδος από την εφαρμογή
- Στις ρυθμίσεις δίνεται η δυνατότητα αλλαγής IP Address, Port, Sending Time. Υπάρχει έλεγχος για την σωστή καταχώρηση των δεδομένων. (Το Port και το Sending Time να είναι αριθμοί και το IP Address να έχει σωστή δομή.) (Εικ.11)
- Αν χαθεί η σύνδεση με το διαδίκτυο, τότε υπάρχει ανάλογη ειδοποίηση στην οθόνη με pop-up window. (Εικ.12)
- Κατά την έξοδο από την εφαρμογή, γίνεται ερώτηση στο χρήστη αν θέλει σίγουρα να προχωρήσει με την επιλογή του αυτή. (Εικ.13)

### 3.4 Στιγμιότυπα Εφαρμογής

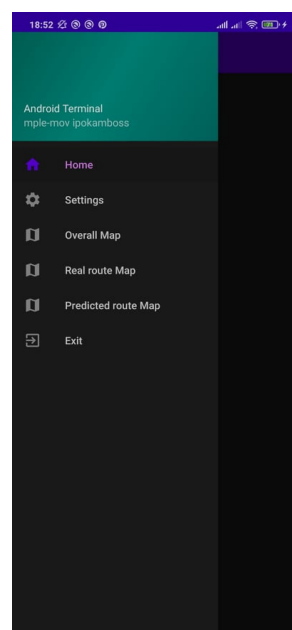
Παρακάτω παρατίθενται βοηθητικά στιγμιότυπα οθόνης για την ανάδειξη των λειτουργιών της εφαρμογής.



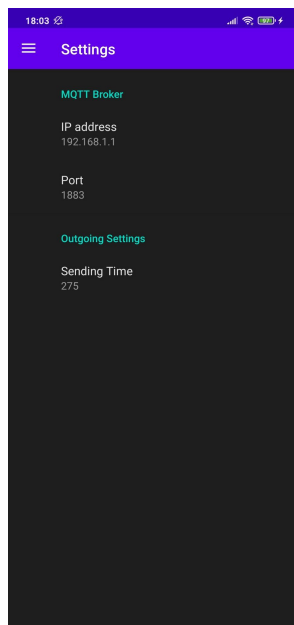
Εικόνα 8



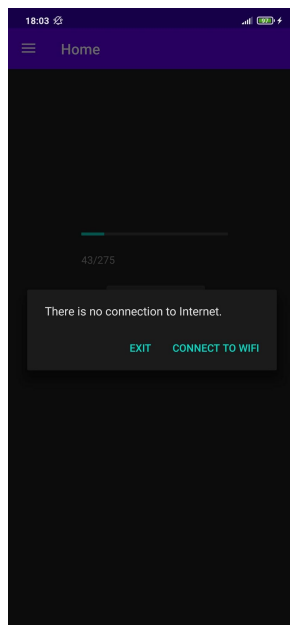
Εικόνα 9



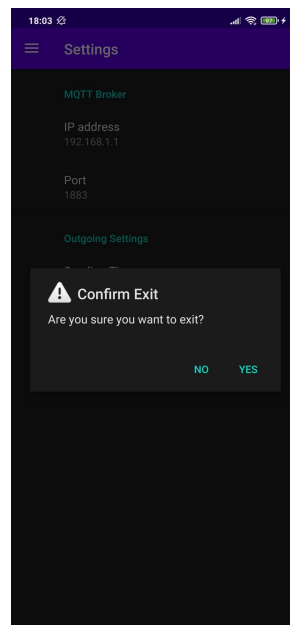
Εικόνα 10



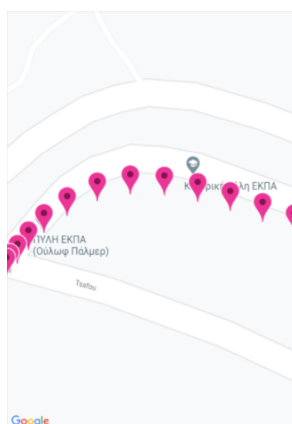
Εικόνα 11



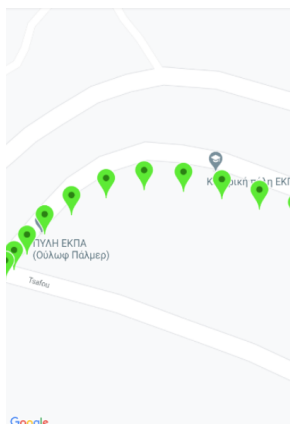
Εικόνα 12



Εικόνα 13



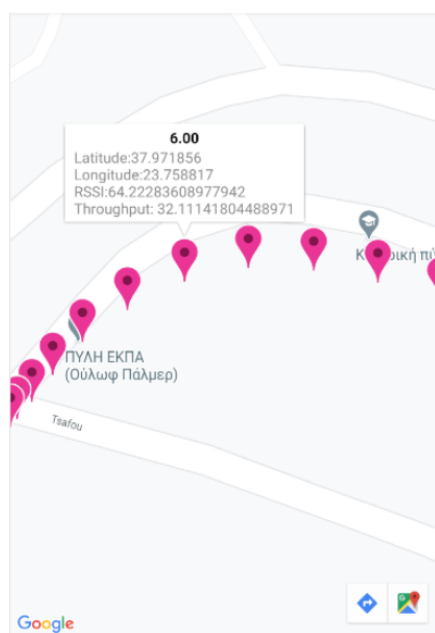
Εικόνα 14



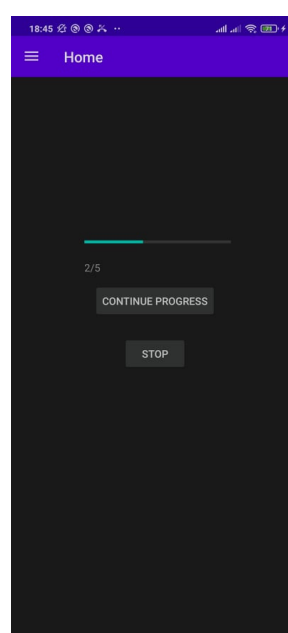
Εικόνα 15



Εικόνα 16



Εικόνα 17: Παράθυρο Πληροφοριών



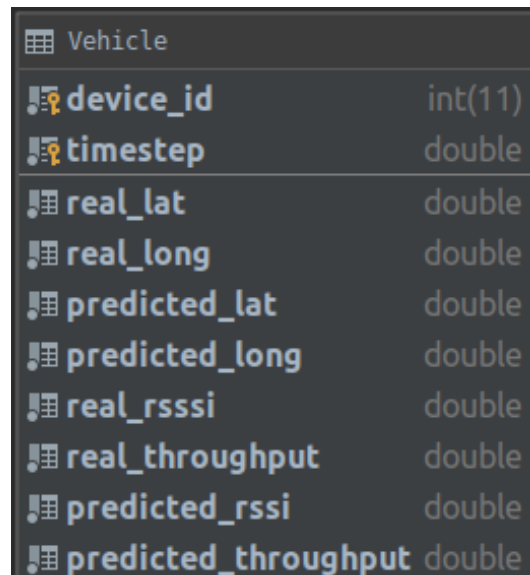
Εικόνα 18



## 4 MySQL Base

Η βάση βρίσκεται στον Edge Server στο φάκελο Database. Αρχικά είναι άδεια και στο πίνακα vehicle υπάρχουν όλα τα υποχρεωτικά πεδία, που ζητούνται στην εκφώνηση.

- Στον constructor της κλάσης DatabaseUtil κάνουμε connect με τη βάση δεδομένων χρησιμοποιώντας τον DriverManager.
- Κάθε φορά που εισάγουμε δεδομένα στη βάση δημιουργούμε ένα insert statement, αρχικοποιούμε τις απαραίτητες μεταβλητές και το κάνουμε execute για να αποθηκευτούν τα δεδομένα στη βάση.
- Με τον τερματισμό του Edge Server κλείνουμε και το connection με τη βάση.



The image shows a screenshot of a database table structure for a table named 'Vehicle'. The table has 10 columns, each with a primary key icon (a small key) and a data type. The columns are: device\_id (int(11)), timestep (double), real\_lat (double), real\_long (double), predicted\_lat (double), predicted\_long (double), real\_rssi (double), real\_throughput (double), predicted\_rssi (double), and predicted\_throughput (double).

Vehicle	
device_id	int(11)
timestep	double
real_lat	double
real_long	double
predicted_lat	double
predicted_long	double
real_rssi	double
real_throughput	double
predicted_rssi	double
predicted_throughput	double

Εικόνα 19: Πίνακας vehicle