****

**ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**

**Ανάπτυξη Λογισμικού για Δίκτυα και Τηλεπικοινωνίες**

**ΦΑΣΗ Νο 1**

Σύστημα Παρακολούθησης Οδηγικής Συμπεριφοράς και Προειδοποίησης για Αποφυγή Συγκρούσεων

**Αποστόλογλου Βασίλειος 1115201500225**

**Android Application**

Για την κατασκευή της εφαρμογής χρησιμοποιήθηκε Android Studio και κάποια external dependecies όπως opencsv, Simple JSON.

Η εφαρμογή διαδυνδέεται με τον edge server μέσω mqtt και στέλνει συνεχώς δεδομένα JSON με μια συχνότητα χρησιμποιώντας ένα Service. Υλοποιήθηκε και αλλήλεπίδραση της MainActivity με το Service μέσω τεχνικής Binding.

**Edge Server**

O Edge Server αποτελείται από τον MQTT broker και τον TCP socket client. O mqtt broker υλοποιείται σε ξεχωριστό Thread διότι ταυτόχρονα λαμβάνει data από το Android και το Training\_Set.csv (feature vector) από τον BackHaul Server. To tcp socket δεν τρέχει σε ξεχωριστό thread για τα δεδομένα αυτής της φάσης αλλά μπορεί να αλλάξει στην επόμενη.

Αποθηκεύει το Training\_Set.csv σε μία κλάση μόλις το λαβει από τον BackHaul Server.

Εκτός από το TrainingSet.csv αποθηκεύει προσωρινά και τα JSON data που λαμβάνει από το Android. Υλοποιήθηκε ως μια απόδειξη ότι τα data φτάνουν σωστά από το Android, σβήνονται κατευθείαν για να μην έχουμε σπατάλη μνήμης.

Για τον edge server χρησιμοποιήθηκε Maven project για εύκολη πρόσβαση σε dependencies.

**BackHaul Server**

O BackHaul Server είναι επίσης MAVEN project για εύκολη πρόσβαση σε dependencies. Αποτελείται από ένα tcp client υπέυθυνο για την επικοινωνία με τον edge Server και ένα client που επικοινωνεί με την SQL βάση μέσω JDBC driver. Για τις ανάγκες αυτής της φάσης, o client για την βάση τρέχει σε ξεχωριστό Thread, ο tcp client τρέχει στην Main.

**Mysql Database**

Για τις ανάγκες αυτής της φάσης, η βάση είναι ένας απλός πίνακας με τα χαρακτηριστικά που ζητήθηκαν. Στην επόμενη φάση ενδέχεται αν αλλάξει και η βάση και ο τρόπος με τον οποίο γίνεται το Installation της, o backhaul server θα διαβάζει μια κενή βάση και θα την μετατρέπει κατάλληλα, την πρώτη φορά που θα τρέχει.

**How to run:**

H εκτέλεση των βημάτων πρέπει να πραγματοποιηθεί όπως φαίνεται παρακάτω.

**1o Μέρος: BackHaul and MYSQL**

1) Τοποθετούμε το Training\_Set directory στο Desktop για να μπορέσει ο server να φτιάξει το feature vector csv. Δεν χρειάζεται να αλλάξετε στον κώδικα το directory, το βρίσκει μόνο του.

2) Δημιουργείτε στον MYSQL ένα προφιλ με username: root και pass:8246 και κάνετε import την βάση που παρέχεται στο project. Για την βάση παρέχεται το Workbench αρχείο.

3) Εκτελούμε run από το intellij και αφήνουμε τον BackHaul να τρέχει προχωρώντας στο μέρος 2ο και στον Edge.

**2o Μέρος: Edge Server**

1) Για να επικοινωνήσει ο Edge με τον BackHaul πρέπει να αλλάξετε το hostname του socket client στην main το οποίο φαίνεται παρακάτω:

**myclient = new SocketClient("bill-virtual-machine", 6090);//connect with backhaul**

2) Εκτελούμε run από το intellij για να αρχίσει ο edge Server όπου και θα λάβει το Training\_Set.csv και θα το κάνει save στο Home directory του machine.

Στην συνέχεια τρέχει επ άπειρον λαμβάνοντας data από την android συσκευή. Προφανώς δεν λαμβάνει κάτι ακόμα διότι δεν έχουμε σετάρει την android εφαρμογή.

**3ο Μέρος: Anroid app**

1) Για να συνδεθεί η εφαρμογή με τον Edge Server πρέπει να αλλάξετε το hostname στον constructor του connection object της κλάσης **MqttObjectClass.**

**this.client = new MqttAndroidClient(context, "tcp://VirtualMachine:1883", clientId)**

Αλλάζετε το **VirtualMachine** σε hostname του δικού σας edge Server.

2) Στη συνέχεια εκτελείτε run για να γίνει το installation της εφαρμογής. Με την έναρξη του MainActivity της εφαρμογής κάνετε **Allow για τον GPS Sensor** και μετά η εφαρμογή θέλει restart. Μόλις γίνει το restart, θα αρχίσει το service να στέλνει data στον edge server όπου και θα εμανίζονται τα data στο run terminal του edge στο intellij