ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Διαχείριση Δικτύων

Τελική εργασία μαθήματος

Σέργιος Αυτσίδης	1115200900052
Αναστάσιος Γιαννακόπουλος	1115200900053

ΑΘΗΝΑ

ΙΟΥΛΙΟΣ 2015

Επεξήγηση Εργασίας

Εισαγωγικά - Περιγραφή

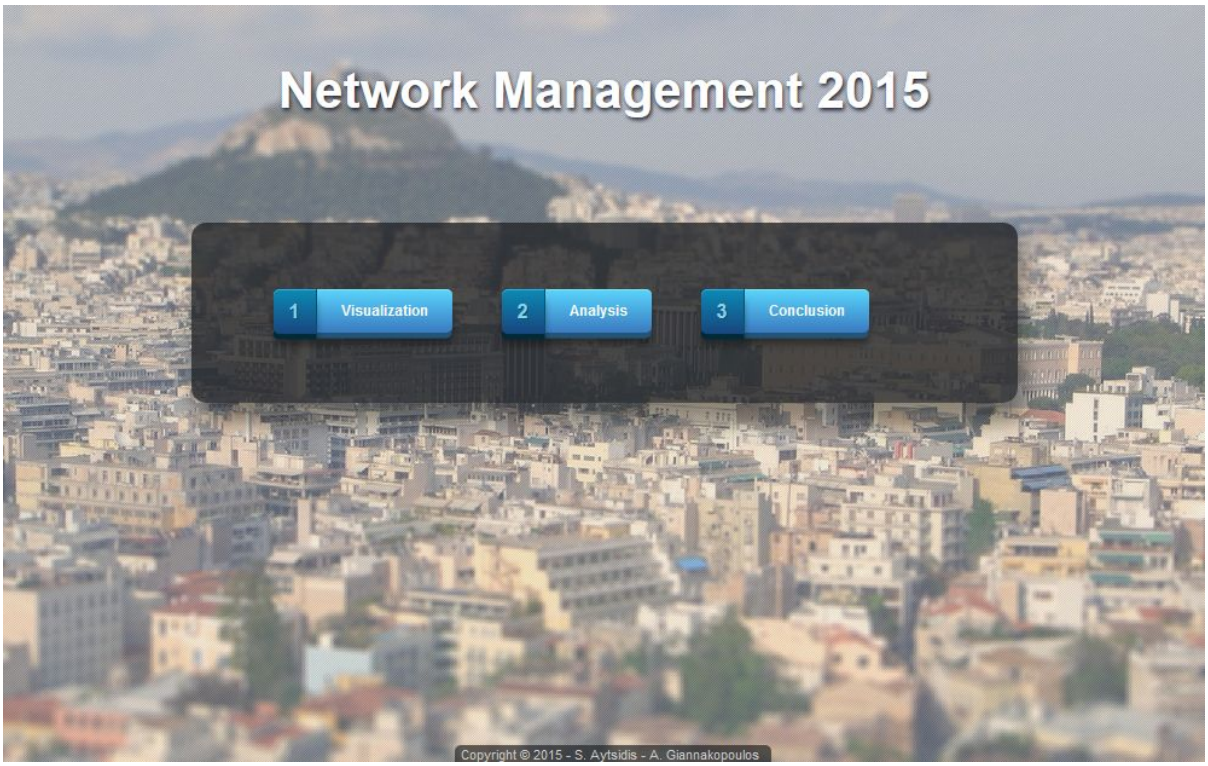
Η εργασία αναπτύχθηκε ως Web Application ακολουθώντας το μοντέλο Server-Client. Αναπτύχθηκε σε περιβάλλον Linux, με την χρήση του Netbeans 8, και ο server που χρησιμοποιήθηκε ηταν ο Apache Tomcat 8.0.15. Η επικοινωνία κώδικα μεταξύ των μελών της ομάδας έγινε με την χρήση Git (Account δημιουργήθηκε στο BitBucket). Χρησιμοποιήθηκαν Servlets και jsps για την επικοινωνία με τον server, ενώ το backend υλοποιήθηκε σε Java.

Περιγραφή διεπαφής (Frontend)

Πέρα απο το html/css κομμάτι της διεπάφης, το κυριότερο τμήμα του, που χειριζόταν την λειτουργία όλης της διεπαφής είναι γραμμένο σε javascript.
Οι javascript βιβλιοθήκες που χρησιμοποιήθηκαν είναι:

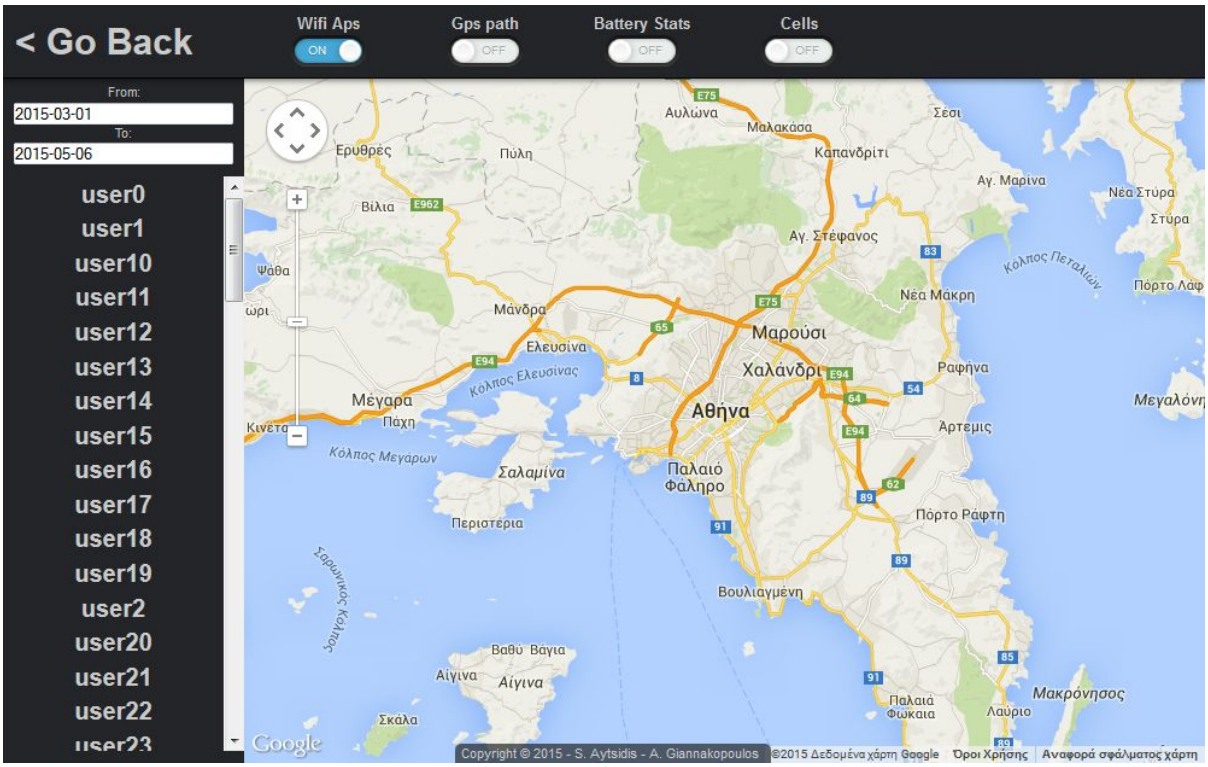
- Jquery
- Jquery-ui
- google - jsapi
- markerclusterer

Η επικοινωνία μεταξύ των servlets και του client γινόταν με post αιτήματα .getJSON όπου τα δεδομένα στέλνονταν πίσω στον client απο το servlet με json.

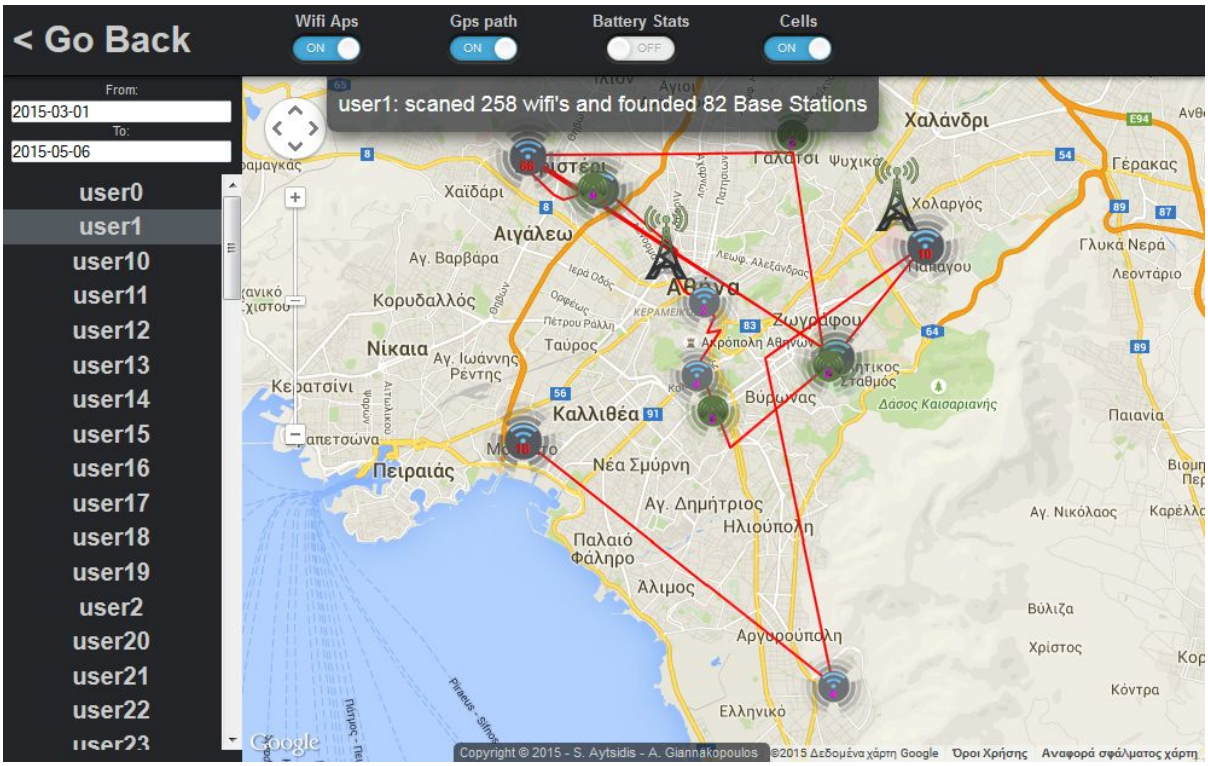


Η αρχική εικόνα, όταν ο χρήστης έρχεται σε επαφή με την εφαρμογή. Του δίνονται οι επιλογές να πλοηγηθεί στα ερωτήματα της άσκησης πατώντας τα αριθμημένα κουμπιά.
Το πρώτο κουμπί σε οδηγεί στην υλοποίηση του ερωτήματος 2.
Το δεύτερο κουμπί σε οδηγεί στην υλοποίηση του ερωτήματος 3.
Το τρίτο κουμπί σε οδηγεί στην υλοποίηση του ερωτήματος 4.

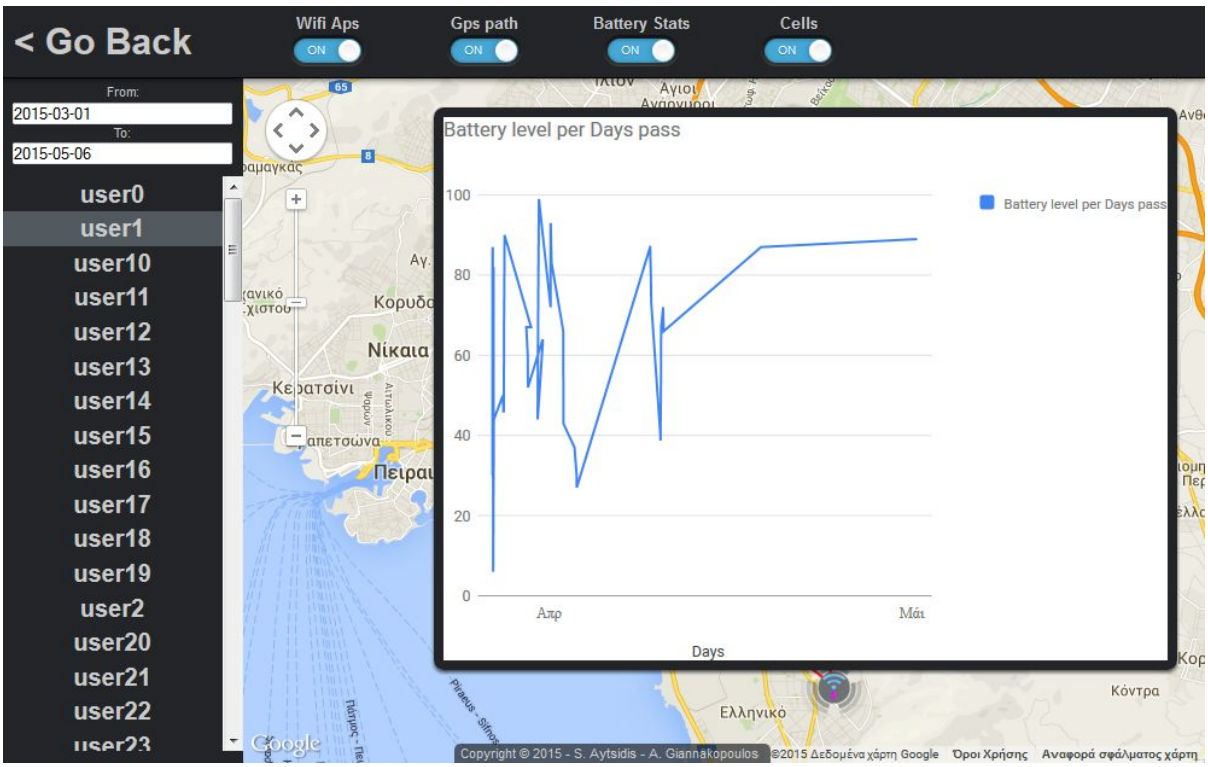
Ερώτημα 2



Αυτή είναι η εικόνα υλοποίησης του ερωτήματος 2. Δίνεται στον χρήστη η δυνατότητα να επιλέξει χρονικό περιθώριο, χρήστη, καθώς και τα υποερωτήματα 1, 2, 3 και 4 που επιθυμεί να δει ταυτόχρονα.

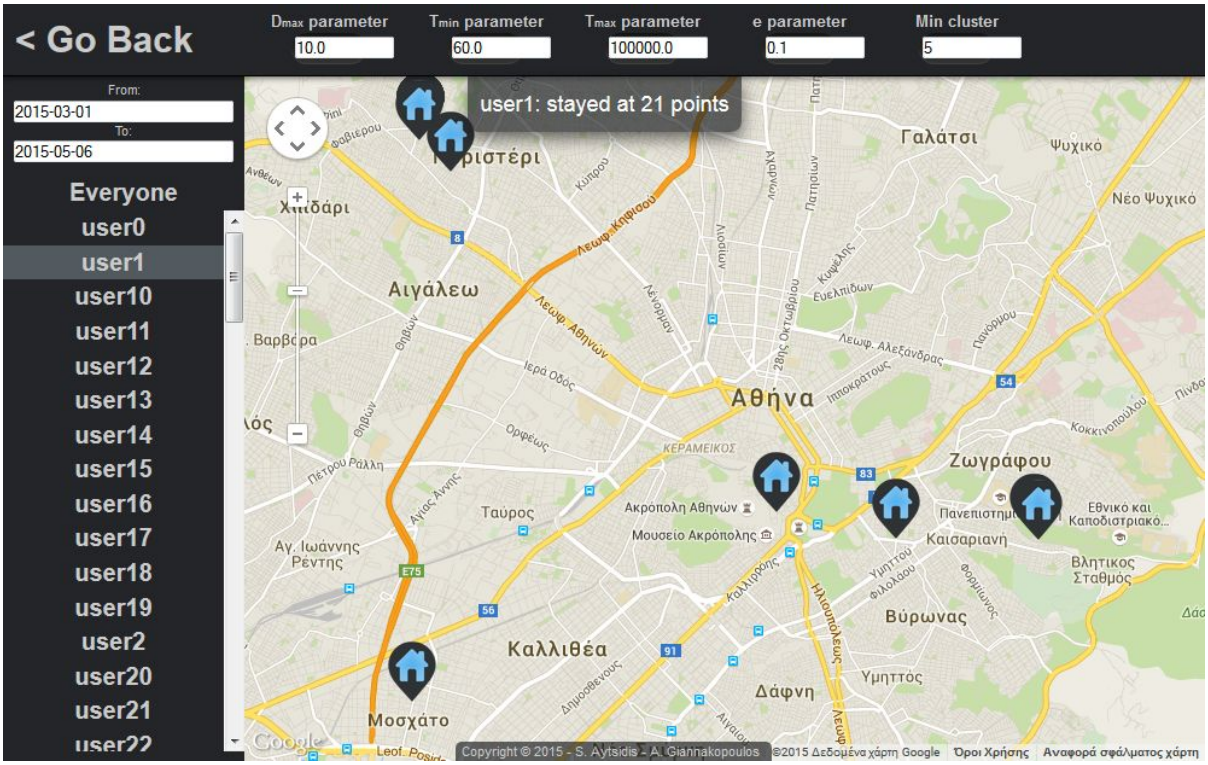


Υποερωτήματα 1, 2 και 4 για τον user1 στο συγκεκριμένο περιθώριο χρόνου που επιλέχθηκε από τον χρήστη. Αν ο χρήστης κατευθύνει το ποντίκι του πάνω σε κάποιο marker στον χάρτη θα εμφανιστούν οι πληροφορίες του marker αυτού.

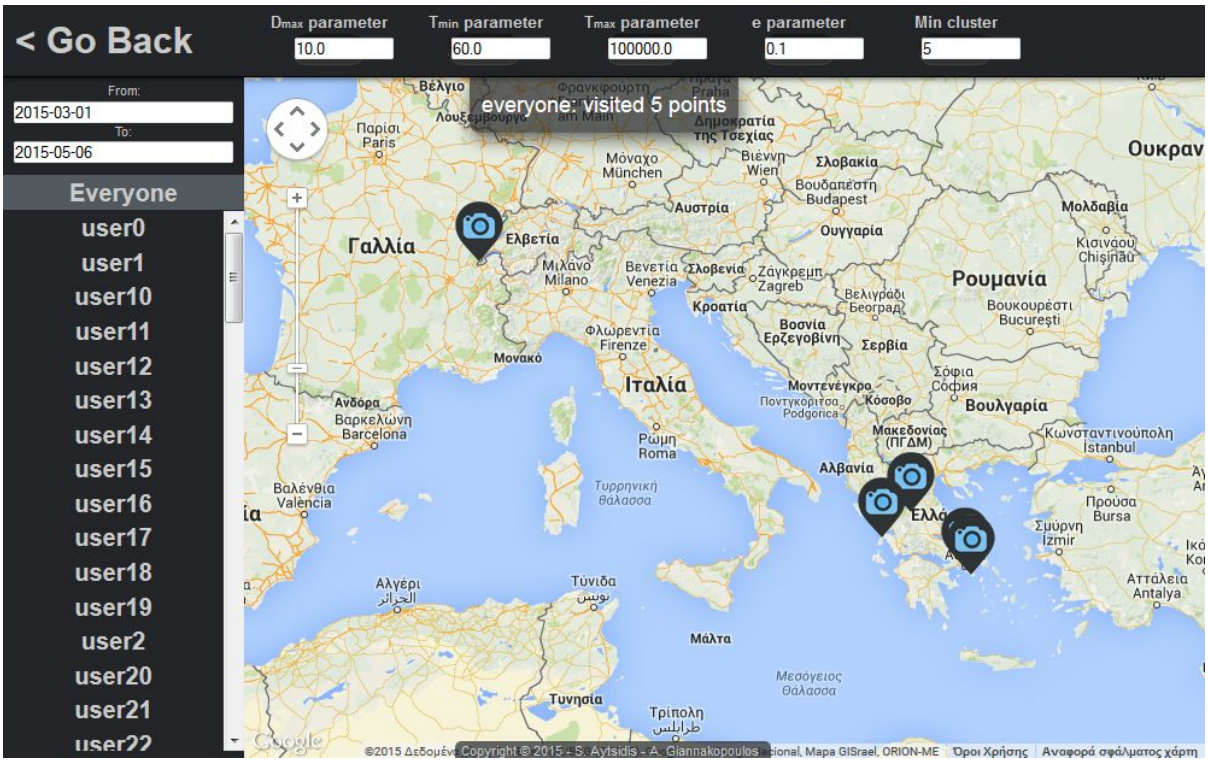


Το υποερώτημα 3. Όπου φαίνεται το διάγραμμα του επιπέδου της μπαταρίας του χρήστη user1 για το επιλεγμένο περιθώριο χρόνου.

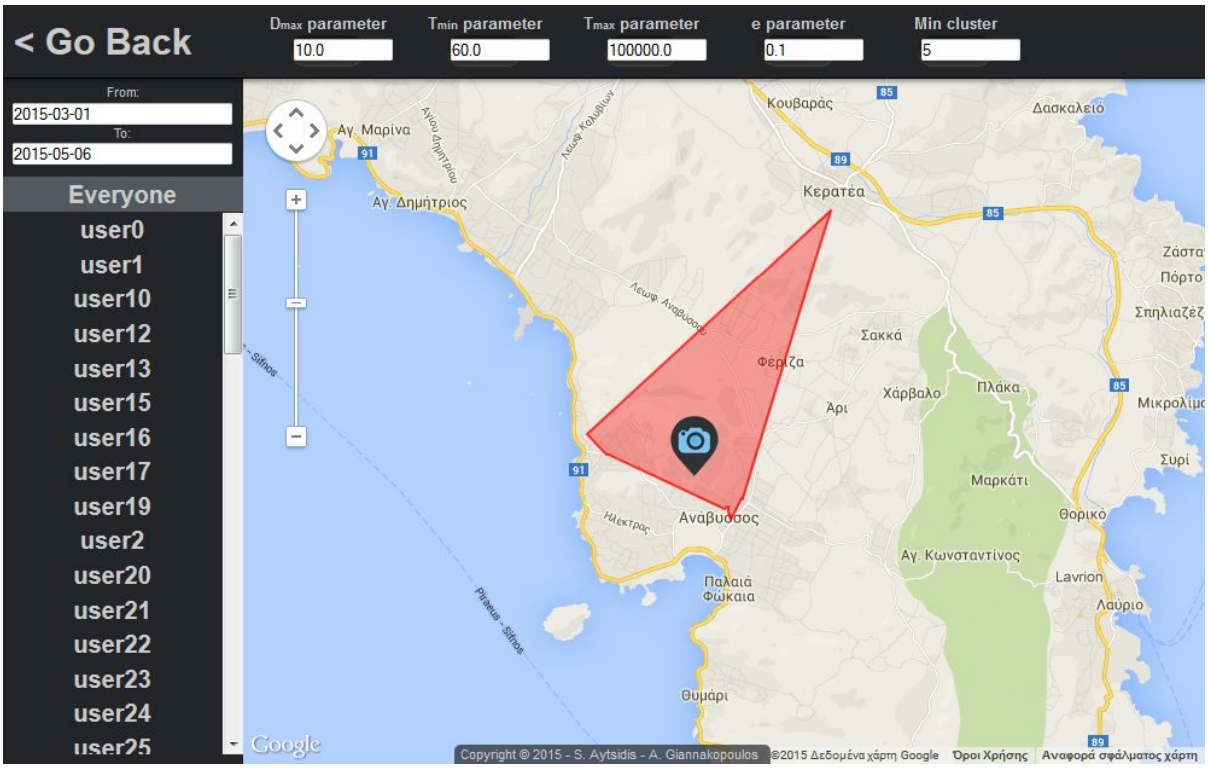
Ερώτημα 3



Υποερώτημα 1, όπου έχει επιλεγεί το χρονικό περιθώριο, και ο χρήστης user1 με τις συγκεκριμένες παραμέτρους Dmax, Tmin και Tmax. Εμφανίζονται τα staypoints του χρήστη αυτού με τις συγκεκριμένες επιλογές.

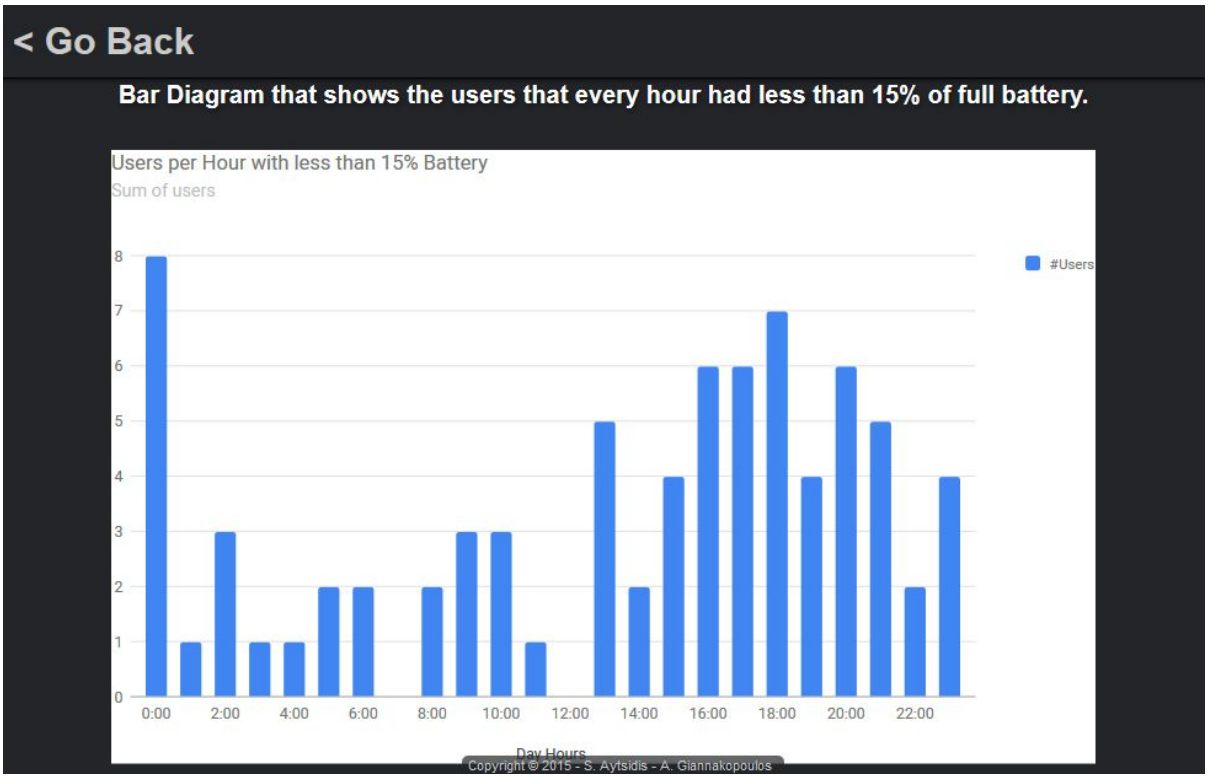


Ο χρήστης μπορεί να επιλέξει να δει το υποερώτημα 2, επιλέγοντας το Everyone που εμφανίζεται στο πάνω μέρος της λίστας. Εδώ χρησιμοποιούνται και οι παράμετροι e και cluster min.

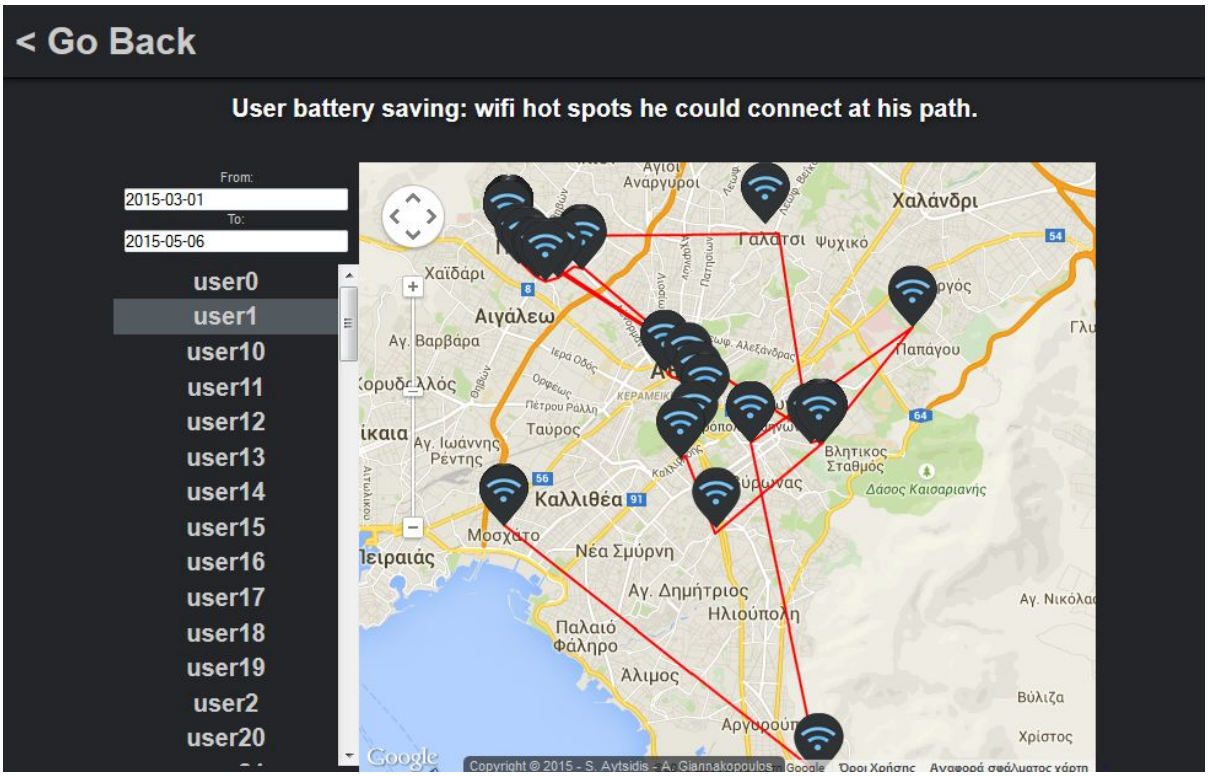


Στο υποερώτημα 2 παρουσιάζονται επίσης και πο πολύγωνοι χώροι που ανήκει το point of interest που βρέθηκε.

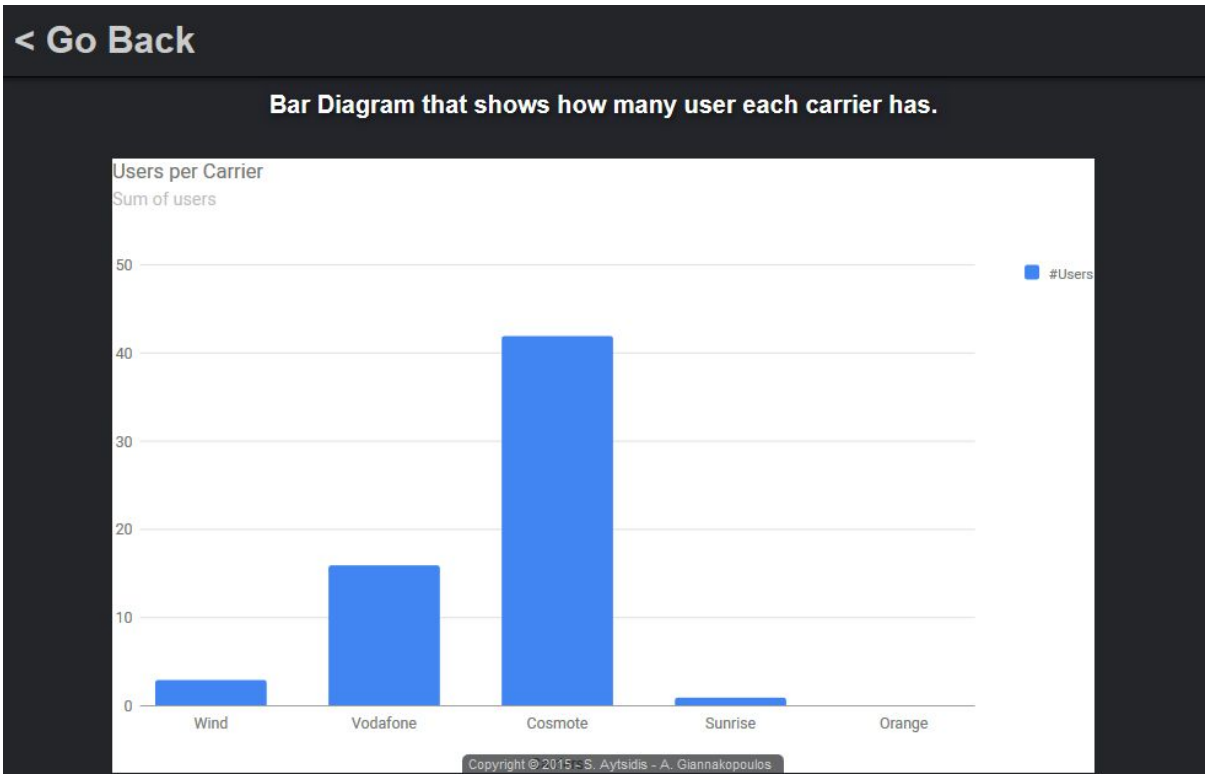
Ερώτημα 4



Υποερώτημα 1. Εμφανίζονται οι συνολικοί χρήστες ανά ώρα της ημέρας όπου το επίπεδο της μπαταρίας τους είναι μικρότερο του 15%.



Υποερώτημα 2. Επιλέγοντας χρονικό περιθώριο και χρήστη, σου εμφανίζεται η πορεία του καθώς και τα access points που μπορεί να συνδεθεί ώστε να ελαχιστοποιήσει την χρήση μπαταρίας.



Υποερώτημα 3. Εμφανίζονται οι συνολικοί χρήστες της κάθε κινητής τηλεφωνίας.

Ο κώδικας javascript περιλαμβάνει την καταχώρηση cached δεδομένων ώστε να ελαχιστοποιηθεί η επικοινωνία με τον server στις περιπτώσεις όπου δεν χρειάζεται κάτι τέτοιο. Όσο τα δεδομένα για την εμφάνιση των αποτελεσμάτων είναι διαθέσιμα δεν ζητάτε από τον server κάποια απάντηση ή λήψη δεδομένων.

Επίσης χρησιμοποιήθηκε marker clusterer για τα markers που εμφανίζονται στον χάρτη στα ερωτήματα όπου τα δεδομένα είναι πάρα πολλά. Αυτό ελαχιστοποίησε τον χρόνο επεξεργασίας και αυξήθηκαν τα fps κατα πολύ κατα την πλήγηση του χάρτη απο τον χρήστη.

Περιγραφή υλοποίησης (Backend)

Τα πηγαία αρχεία κώδικα είναι χωρισμένα κατ αρχήν σε 3 packages:

- **Data:** Περιέχει όλες τις κλάσσεις που είτε αναπαριστούν δεδομένα από τα αρχεία εισόδου είτε τα χειρίζονται. Επίσης περιέχει τις κλάσεις CustomComparator, που είναι μια κλάση σύγκρισης μεταξύ instances της κλάσης User, καθώς και την κλάση Utilities που περιέχει υλοποιήσεις χρήσιμων συναρτήσεων που χρησιμοποιούνται σε όλο το package (αλλά και έξω από αυτό).

- **DataBase**:
    - Η κλαση DataBase είναι η κύρια αποθήκη δεδομένων, καθώς περιέχει λίστες με τους Users, τα AccessPoints και γενικότερα ότι input έχει δωθεί στα αρχεία εισόδου. Όλα βρίσκονται σε static δομές δεδομένων, και έτσι δεν χρειάζεται instance της κλάσης για να χρησιμοποιηθούν.
    - Η κλάση BuildDB είναι μια κλάση που χρησιμεύει για το parse των αρχείων εισόδου και την εισαγωγή των δεδομένων στην κλάση DataBase.
    - Η κλαση ConnectionInit είναι ενα servlet που εκτελείτα μια φορά όταν γίνεται deploy και το project. Εκεί αρχικοποιείται η DataBase.

- **WebServices:**
    - Εδώ υπάρχουν τα Servlets που υλοποιούν την επικοινωνία με το Frontend της εφαρμογής (1 για κάθε σελίδα).

Αντιμετώπιση των ζητουμένων

Ερώτημα 1

Για την αντιμετώπιση αυτού του ερωτήματος θα χρειαστεί μια προεπεξεργασία πάνω στο αρχείο εισόδου που περιέχει τα scanned wifi access points από τους χρήστες.
Συγκεκριμένα θα πρέπει να εισαχθούν στις δομές μας όλοι οι χρήστες ώστε να τους χρησιμοποιήσουμε αργότερα, όλα τα access points που υπάρχουν στο αρχείο και αντιστοίχηση αυτών προς τους χρήστες.

Στην συνέχεια αρκεί να χρησιμοποιήσουμε τον αλγόριθμο που προτείνεται ώστε να βρεθούν οι μέσες γεωγραφικές θέσεις όλων των access points. Τα εξαγόμενα αποτελέσματα του αλγόριθμου αποθηκεύονται και αυτά σε δομές για μετέπειτα χρήση.

Ερώτημα 2

Υποερώτημα 1:

Δεδομένου κάποιου χρήστη από την βάση, χρειάζεται να πάρουμε τα ήδη αποθηκευμένα και αντιστοιχισμένα σε αυτόν wifi access points, και για κάθε ένα από αυτά να χρησιμοποιήσουμε τα στοιχεία που έχουν εξαχθεί για το συγκεκριμένο access point από όλους τους χρήστες από την επεξεργασία του ερωτήματος 1. Στην συνέχεια επιστρέφεται στον client η λίστα με αυτά τα access points. Στον client τα δεδομένα εμφανίζονται με markers πάνω στον χάρτη google maps όπου στο infowindows του κάθε marker περιέχονται η πληροφορίες του κάθε access point.

Υποερώτημα 2:

Σε αυτό το σημείο χρειάζεται μια προεπεξεργασία ώστε να ληφθούν τα δεδομένα των gps στιγμάτων των χρηστών από το αντίστοιχο αρχείο. Τα στίγματα στην συνέχεια αντιστοιχίζονται στους χρήστες που τα έχουν παράγει.
Στον client επιστρέφεται απλά ή λίστα των στιγμάτων για τον συγκεκριμένο χρήστη για το συγκεκριμένο χρονικό περιθώριο. Στον client τα δεδομένα εμφανίζονται μέσω javascript googlemaps API πάνω στον αντίστοιχο χάρτη και ενώνονται με googlemaps - polylines.

Υποερώτημα 3:

Αντίστοιχα με το ερώτημα 2, χρειάζονται τα δεδομένα μπαταρίας του κάθε χρήστη. Επομένως γίνεται προεπεξεργασία ώστε να εισαχθούν τα δεδομένα αυτά στις δομές μας, και να αντιστοιχηθούν στον κάθε χρήστη.
Στον client επιστρέφεται απλά ή λίστα των δεδομένων μπαταρίας για τον συγκεκριμένο χρήστη για το συγκεκριμένο χρονικό περιθώριο. Στον client τα δεδομένα εμφανίζονται μέσω javascript google JSAPI πάνω σε αντίστοιχο line graph.

Υποερώτημα4:

Αντίστοιχα με το ερώτημα 2 και 3, χρειάζονται τα δεδομένα των καταγεγραμμένων cells του κάθε χρήστη. Επομένως γίνεται προεπεξεργασία ώστε να εισαχθούν τα δεδομένα αυτά στις δομές μας, και να αντιστοιχηθούν στον κάθε χρήστη.
Στον client επιστρέφεται απλά ή λίστα των δεδομένων των καταγεγραμένων cells για τον συγκεκριμένο χρήστη για το συγκεκριμένο χρονικό περιθώριο. Στον client τα δεδομένα εμφανίζονται με markers πάνω στον χάρτη google maps όπου στο infowindows του κάθε marker περιέχονται η πληροφορίες του κάθε base station.

Ερώτημα 3

Υποερώτημα 1

Σε αυτό το ερώτημα δεν χρειάζεται κάποια προεπεξεργασία δεδομένων. Για να παραχθούν τα stay points ενός χρήστη για κάποιο συγκεκριμένο χρονικό περιθώριο χρειάζεται η χρήση του αλγόριθμου που μας δίνεται με είσοδο τα στίγματα gps που παράγει και το ερώτημα 2.2. Ο αλγόριθμος υλοποιήθηκε σε Java με βάση τον ψευδοκώδικα της εκφώνησης. Οι παράμετροι δίνονται από τον client μέσω της διεπαφής του.
Τα δεδομένα που εξάγονται από τον αλγόριθμο είναι τα stay points όπου στέλνονται στον client και απεικονίζονται στον χάρτη google maps με μορφή markers.

Υποερώτημα 2

Για να παραχθούν τα points of interest όλων των χρηστών για το συγκεκριμένο χρονικό περιθώριο χρειάζεται η εύρεση των stay points όλων το χρηστών όπως στο υποερώτημα 1, και έπειτα η εισαγωγή στην DBScan Clusterer, (δες σχόλιο 3), για την εύρεση των points of interest.
Οι παράμετροι δίνονται από τον client μέσω της διεπαφής του.
Τα δεδομένα που εξάγονται από τον DBScan Clusterer είναι τα points of interest όπου στέλνονται στον client και απεικονίζονται στον χάρτη google maps με μορφή markers.

Ερώτημα 4

Υποερώτημα 1

Τα δεδομένα που χρειάζονται υπάρχουν ήδη στις δομές μας, αρκεί να γίνει η επεξεργασία τους, ώστε να βρεθεί το ζητούμενο ερώτημα. Τα δεδομένα στέλνονται στον client και απεικονίζονται σε bar diagram μέσω της βιβλιοθήκης JSAPI της google.

Υποερώτημα 2

Τα δεδομένα που χρειάζονται υπάρχουν ήδη στις δομές μας, αρκεί να γίνει η επεξεργασία τους, ώστε να βρεθεί το ζητούμενο ερώτημα. Επομένως, για τον συγκεκριμένο χρήστη την συγκεκριμένη χρονική στιγμή, βρίσκεται το gps path του, όπως και στο ερώτημα 2.2.
Για την υλοποίηση του ερωτήματος αυτού, χρειάζεται να ακολουθήσουμε αυτό το μονοπάτι, και για κάθε σημείο του να βρίσκουμε το κοντινότερο από τα Access Points που έχουν βρεθεί στο ερώτημα 1. Στην συνέχεια τα εξαγόμενα σημεία στέλνονται στον client και απεικονίζονται στον χάρτη google maps σε μορφή markers.

Υποερώτημα 3

Τα δεδομένα που χρειάζονται υπάρχουν ήδη στις δομές μας, αρκεί να γίνει η επεξεργασία τους, ώστε να βρεθεί το ζητούμενο ερώτημα. Συγκεκριμένα αθροίζονται οι χρήστες των εταιριών κινητής τηλεφωνίας. Τα δεδομένα στέλνονται στον client και απεικονίζονται σε bar diagram μέσω της βιβλιοθήκης JSAPI της google.

Υποερώτημα 4

Στα 2.4Ghz κάθε συχνότητα έχει 22Mhz παράθυρο έτσι το εύρος αυτό χωρίζεται σε 14 κανάλια. Συνεπώς η κάθε συχνότητα απέχει με τις γειτονικές 5Mhz. Γιαυτό τον λόγο τα κανάλια που συνήθως επιλέγονται είναι τα 1-6-11.

Συνεπώς για να διαλέξουμε ένα κανάλι με την ελάχιστη παρεμβολή, αρκεί σε κάθε κανάλι να αθροίσουμε την ισχύ που έχει μετρηθεί από όλα τα access points που είναι μέσα στο cluster του point of interest (δηλαδή εκπέμπουν σε αυτό το κανάλι), και να επιλέξουμε το κανάλι του οποίου ελαχιστοποιείται η συνολική παρεμβαλλόμενη ισχύς.
Ως παραδοχή παίρνουμε ότι λόγω της ύπαρξης του cluster του point of interest οι μετρήσεις έχουν γίνει από το κεντροειδές του, συνεπώς οι ισχύς του κάθε access point είναι κατά προσέγγιση σωστή σε σχέση με το να μετράμε από οπουδήποτε μέσα στο cluster.

Όσον αφορά την τοποθεσία των δύο νέων access points:

1. Θεωρούμε ότι κοντά στο κεντροειδές του cluster η πυκνότητα των access points είναι μεγαλύτερη.
2. Όσο απομακρυνόμαστε από ένα point of interest τα access points είναι πιο διάσπαρτα.

Συνεπώς θα διαλέξουμε να τοποθετήσουμε τα δύο νέα access points στο κυρτό περίβλημα του cluster και σε σημεία μεγιστοποιούν την απόσταση από τα υπόλοιπα access points. Καθώς ταυτόχρονα να μεγιστοποιούν και την απόσταση μεταξύ τους.

Σχόλια - Παρατηρήσεις:

1. Το διάβασμα της εισόδου γίνεται σταδιακά. Με την εκκίνηση της εφαρμογής διαβάζεται το αρχείο με τα Wifi records καθώς είναι το αρχείο που χρησιμοποιείται περισσότερο αλλά και για να δημιουργήσουμε μια λίστα με τους users. Τα υπόλοιπα αρχεία διαβάζονται οταν και αν είναι απαραίτητο.
2. Για μεγαλύτερη αποδοτικότητα, όπου ήταν εφικτό έχουν χρησιμοποιηθεί HashMap. Επίσης, τα δεδομένα υπάρχουν μια φορά στην μνήμη αλλά σε κάποιες περιπτώσεις υπάρχουν και άλλα Instances σε δευτερεύουσες δομές για την επιτάχυνση ερωτημάτων. Πχ, τα GPS records υπάρχουν συγκεντρωμένα σε ένα HashMap, αλλά κάθε User έχει μια λίστα με Instances αυτών, για να είναι εφικτό να βρούμε τα GPS records του πιο γρήγορα.
3. Όσον αφορά το DBSCAN clustering, χρησιμοποιήθηκε η κλάση DBSCANClusterer, που περιλαμβάνεται στο πακέτο *org.apache.commons.math3.ml.clustering*.