Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ανίχνευση και Καταγραφή Ποιότητας Οδοστρώματος** **Detection and Registration of the Road Surface**
Ονοματεπώνυμο Φοιτητή	**Ιωάννης Χρυσικόπουλος**
Πατρώνυμο	**Ελευθέριος**
Αριθμός Μητρώου	**ΜΠΣΠ/ 15100**
Επιβλέπων	**Ευθύμιος Αλέπης, Επίκουρος Καθηγητής**

Ημερομηνία Παράδοσης **Σεπτέμβριος 2018**

Τριμελής Εξεταστική Επιτροπή

Αλέπης Ευθύμιος Επίκουρος Καθηγητής	Πατσάκης Κωνσταντίνος Επίκουρος Καθηγητής	Βίρβου Μαρία Καθηγήτρια

Θα ήθελα να ευχαριστήσω τον Κο Ευθύμιο Αλέπη για την βοήθεια και την καθοδήγηση που μου παρείχε σε όλη την διάρκεια της μεταπτυχιακής μου εργασίας όπως επίσης και το Πανεπιστήμιο Πειραιώς και τους καθηγητές του για τις γνώσεις που μου προσέφεραν.

Χρυσικοπουλος Ιωαννής

I would like to thank Mr Eythymi Alepi for his help and guidance through the creation of this project as well as the University of Piraeus and the professors for the knowledge they imparted me.

Chrysikopoulos Ioannis

Περιεχόμενα

Περίληψη

Στην εφαρμογή έχει υλοποιηθεί ένα σύστημα το οποίο ανιχνεύει την ποιότητα του οδοστρώματος-δρόμου. Ανάλογα με το πόσο φθαρμένο είναι το οδόστρωμα, η εφαρμογή ενημερώνει τον οδηγό μέσω του κινητού του τηλεφώνου εάν υπάρχουν λακκούβες ή έντονα φθαρμένη άσφαλτος στην διαδρομή που ακολουθεί, έτσι ώστε να είναι προετοιμασμένος.

Επίσης, η ποιότητα του οδοστρώματος αποτυπώνεται σε μία διαδικτυακή σελίδα στην οποία όλοι έχουν πρόσβαση και μπορούν να δούνε που βρίσκονται οι λακκούβες αλλά και το σύνολο αυτών για κάθε νομό ή δήμο.

Το σύστημα χρησιμοποιεί αρχικά τους αισθητήρες του κινητού για να ανιχνεύει τις λακκούβες αλλά και τις τοποθεσίες τους. Μέσω του wi-fi, αποθηκεύονται σε μια απομακρυσμένη βάση δεδομένων, από την οποία ενημερώνονται και τα υπόλοιπα κινητά που χρησιμοποιούν την εφαρμογή, όπως και η ιστοσελίδα.

Abstract

The application that we have created is a system that detects the road quality. Depending on how worn the road is, the application informs the driver through the mobile device whether there is a worn road ahead of him or pot holes, so he can be aware.

Also, the quality of the road is captured on a web page that everyone has access to and they can see where the pot holes are, as well as the sum of them grouped by each country or municipality.

The system initially uses the cell phone sensors to detect the potholes and their coordinates. Through the wi-fi, they are stored in a remote database, from which all other mobile devices that use the application as well as the website, are updated.

Εισαγωγή – Σύντομη Περιγραφή

Με την αύξηση του πληθυσμού ανά τον κόσμο έχουμε όπως είναι λογικό και την επέκταση και δημιουργία πόλεων αλλά και κατά επέκταση του οδικού δικτύου. Το οδικό δίκτυο όπως όλοι μας βιώνουμε, λόγω της εκτεταμένης χρήσης του ή λόγω της χαμηλής ποιότητας κατασκευής ή των καιρικών φαινομένων, φθείρεται και δημιουργούνται λακκούβες σε αυτό.

Οι λακκούβες εκτός από το ότι μπορεί να είναι ενοχλητικές και να δημιουργούν κίνηση στο δρόμο, μπορεί να είναι επικίνδυνες για τον οδηγό και την πρόκληση ατυχήματος, αλλά και επιζήμιες για το ίδιο το οδικό μέσο. Σύμφωνα με στατιστικές έρευνες, στην Αμερική η μέση τιμή επισκευής φθοράς του οδικού μέσου (αναρτήσεις, λάστιχα, ευθυγράμμιση κτλ.) ανά χρόνο και ανά οδικό μέσο, είναι περίπου 370 δολάρια. Συνολικά, το ποσό για όλους τους οδηγούς ανέρχεται περίπου στα 3 δισεκατομμύρια τον χρόνο και το ποσοστό των δρόμων που χαρακτηρίζεται ως "φτωχό" είναι πολύ μεγάλο ανά περιοχή και συγκεκριμένα από 38% μέχρι και 64%.

Για όλους αυτούς του λόγους είναι πολύ σημαντικό οι οδηγοί αλλά και το κράτος και οι δήμοι να γνωρίζουν την ύπαρξη του κακού έως και επικίνδυνου οδοστρώματος, είτε για την αποφυγή ατυχήματος ή φθοράς του αυτοκινήτου, είτε για την επιδιόρθωσή του.

Η εφαρμογή μας στοχεύει στην ενημέρωση του οδηγού για τις λακκούβες ή για το κακό οδόστρωμα που θα συναντήσει στην διαδρομή του, αλλά και στην καταγραφή και τη δυνατότητα αποτύπωσης και επεξεργασίας αυτών των δεδομένων.

Αρχικά, μέσω κινητής συσκευής και συγκεκριμένα με την χρήση του αξελερομέτρου, καταγράφεται συνεχώς η μεταβολή του οδοστρώματος, ενώ παράλληλα με την χρήση του GPS καταγράφεται η θέση και η πορεία. Ανάλογα με τις μετρήσεις που πραγματοποιούνται και σύμφωνα με συγκεκριμένες προϋποθέσεις αποθηκεύεται το στίγμα της λακκούβας σε μία απομακρυσμένη βάση δεδομένων. Η βάση δεδομένων είναι υπεύθυνη τόσο για την αποθήκευση των δεδομένων, αλλά και για την αποστολή τους στα αντίστοιχα τερματικά που έχουν πρόσβαση σε αυτή .

Η επεξεργασία των δεδομένων για το εάν και πώς θα αποθηκεύουν γίνεται αποκλειστικά από την εφαρμογή της κινητής συσκευής και όχι από την βάση δεδομένων, η οποία εξυπηρετεί την αποθήκευση τους χωρίς περαιτέρω υπολογιστική λειτουργικότητα. Αντίστοιχα, η ιστοσελίδα έχει πρόσβαση στην βάση δεδομένων με τις καταγεγραμμένες λακκούβες και τις αποτυπώνει σε χάρτη.

Η ιστοσελίδα κατά επέκταση επεξεργάζεται τα δεδομένα (διπλογραφίες, κατανόηση δεδομένων) από την βάση και εκτός από την αποτύπωση των δεδομένων στον χάρτη, ενημερώνει τον χρήστη για τον δήμο ή την περιοχή που βρίσκονται οι λακκούβες αλλά και τα σύνολα αυτών.

Introduction – Brief Description

With the increase of the global population, it is logical to extend and create new cities as well as extending the road network. The road network as we all experience, because of its extensive use or because the poor construction or even because of weather phenomena, is damaged and potholes are created on it.

In addition of being annoying and crate traffic at the road, they can be also dangerous to the driver and also harmful for the vehicle. According to statistical surveys, in the United States of America, the average repair cost of road wear (suspensions, tires, alignment, etc.) per year and per vehicle is about $ 370. In total, the amount for all drivers is around 3 billion per year, and the percentage of roads characterized as "poor" is very large per region, namely from 38% to 64%.

For all these reasons, it is very important for drivers, but also for the state and municipalities to know the existence of the bad and dangerous road surface, either to avoid an accident or damage to the car or to repair it.

Our application aims to inform the driver for the pot holes or the bad quality of the road that he will encounter in his path, as well as the recording and the capability of capturing and processing this data.

Initially, through the mobile device and specifically using the accelerometer, the mobile application continuously records the alteration of the pavement, while using the GPS the tracking and the course is recorded. Depending on the measurements made and according to specific conditions, the pitch of the pot hole is stored in a remote database. The database is responsible for both storing the data and sending it to the corresponding terminals that access it.

The processing of the data for whether and how are going to be stored is made exclusively in the application of the mobile device and not in the database, which only serves the storing of the data without further computational functionality. The web page has access to the database of the recorded potholes and maps them accordingly.

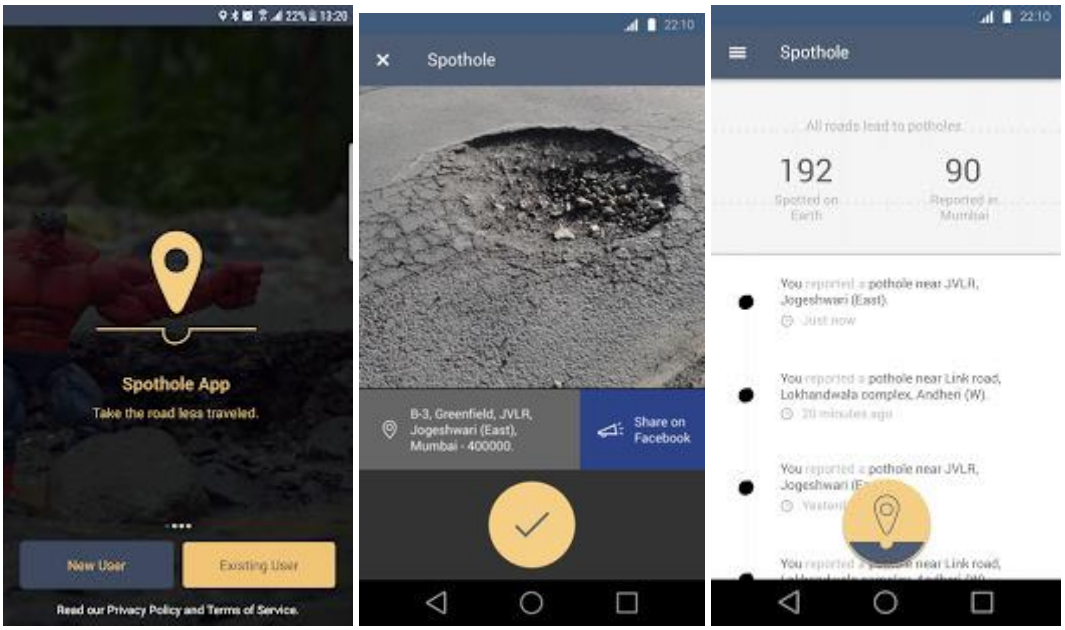
The site processes the data (duplicates, data comprehension) extensively from the database. In addition to mapping the data on the map, it presents to the user about the municipality or the area where the potholes are located, providing the pot holes' sum per area.

Ανασκόπηση Πεδίου

Παρακάτω καταγράφονται ορισμένες εφαρμογές που έχουν υλοποιηθεί παγκοσμίως με παρεμφερές αντικείμενο με τη δικιά μας.

Spothole. Pothole Tracking App

Αυτή η εφαρμογή έχει φτιαχτεί στην Ινδία. Ο χρήστης έχει τη δυνατότητα να βγάλει φωτογραφία τη λακκούβα που εντόπισε, έτσι ώστε αυτή να αποθηκευτεί μαζί με το στίγμα της και κατά συνέπεια, οι αρχές να ενημερωθούν για αυτές.



Street Bump

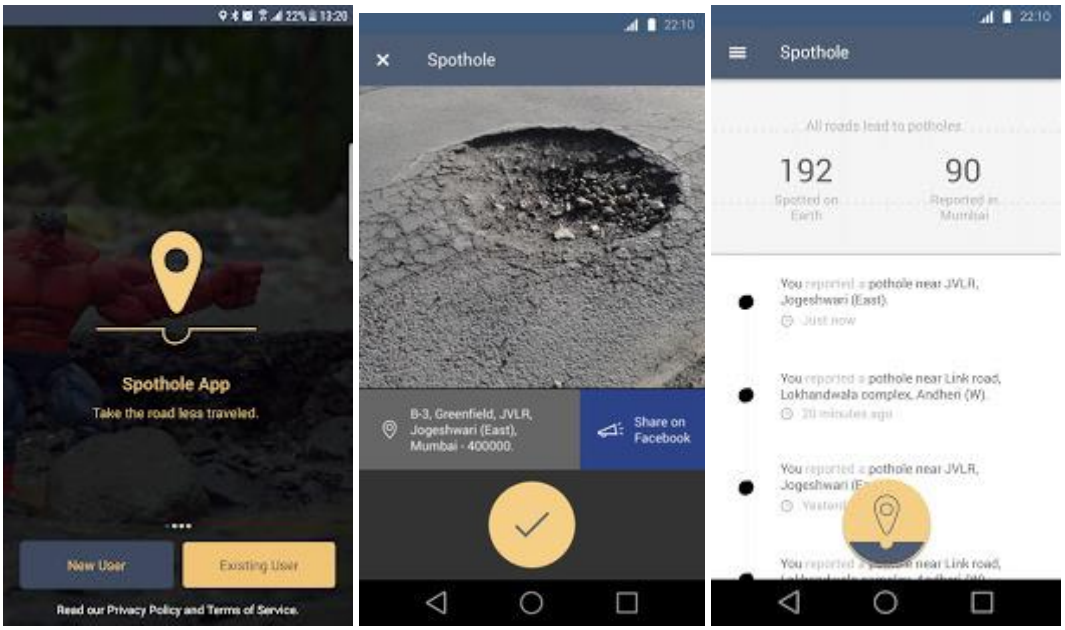
Η εφαρμογή αυτή είναι ένα project του δημαρχείου της Βοστώνης στην οποία ο χρήστης καταγράφει την κάθε διαδρομή που κάνει. Η εφαρμογή καταγραφεί τις λακκούβες της διαδρομής και κρατάει ιστορικό των διαδρομών του χρήστη.

Field Review

Here are some applications that have been implemented, having similar object to ours, globally.

Spothole. Pothole Tracking App

This application is made in India. The user has the ability to take a picture of the pothole he has found so that it is stored, along with his coordinates. The authorities are then informed about them.

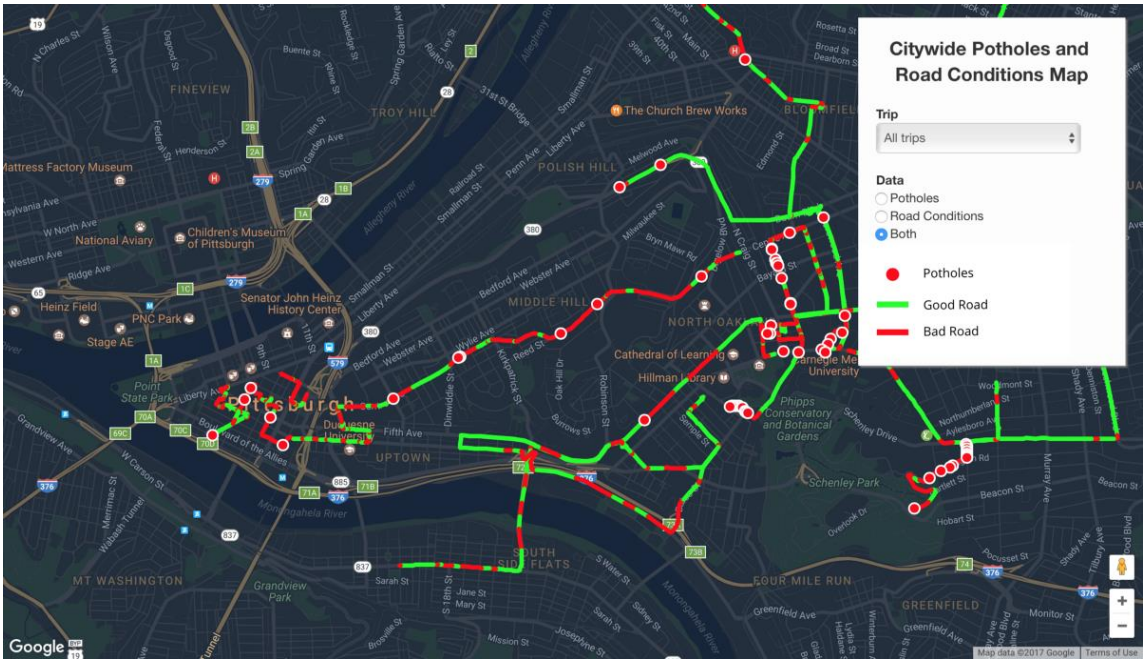


Street Bump

This application is a Boston Town Hall project where the user records each route he or she makes. The application records the path pits and keeps track of the user's paths.

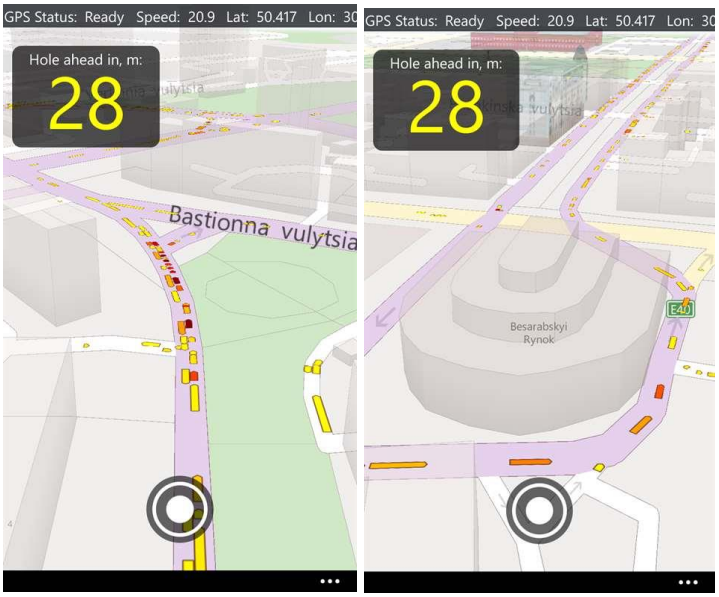
intelligent-pothole-detection

Αυτήν η εφαρμογή υλοποιήθηκε από μια ομάδα του Carnegie Mellon University, η οποία καταγράφει διαδρομές μέσω κινητών συσκευών κατά την κίνηση του οχήματος και δημιουργεί χάρτες, οι οποίοι ενημερώνουν όχι μόνο για λακκούβες, αλλά και λεπτομέρειες για την κατάσταση του οδοστρώματος.



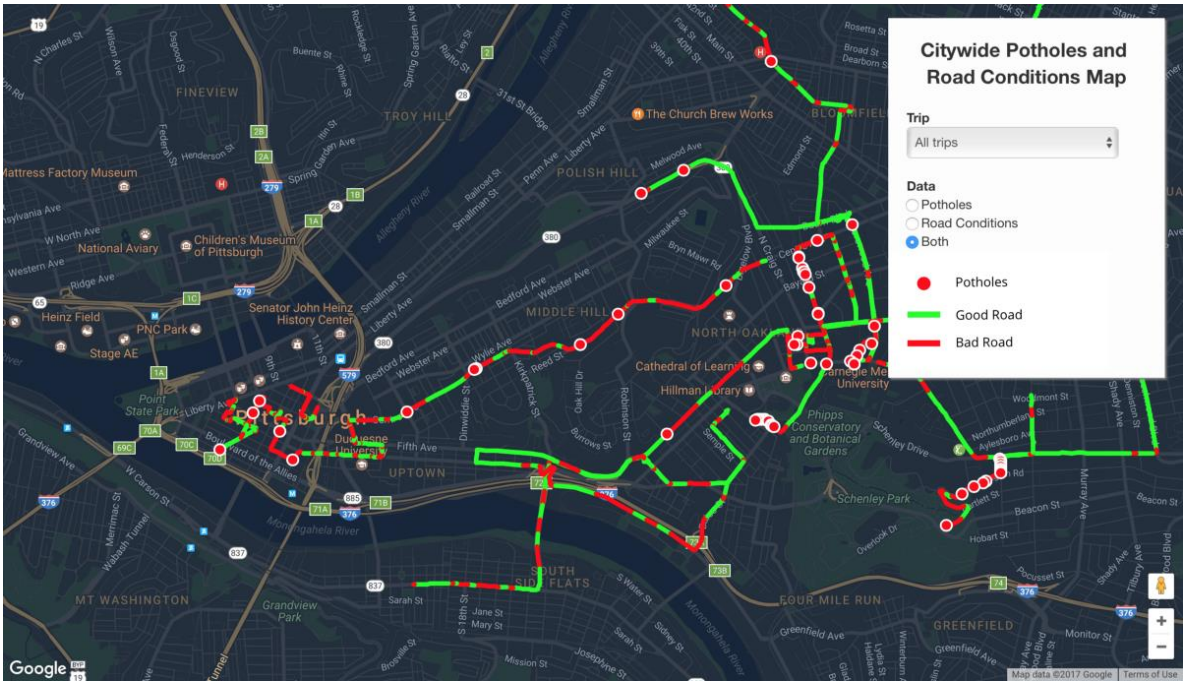
Pothole Tracker

Η συγκεκριμένη εφαρμογή, μέσω των κραδασμών του αυτοκινήτου, ανιχνεύει και αποθηκεύει τις λακκούβες και στην συνέχεια ενημερώνει το χρήστη οπτικά αλλά και με ήχο για τις λακκούβες που μπορεί να συναντήσει στην διαδρομή του.



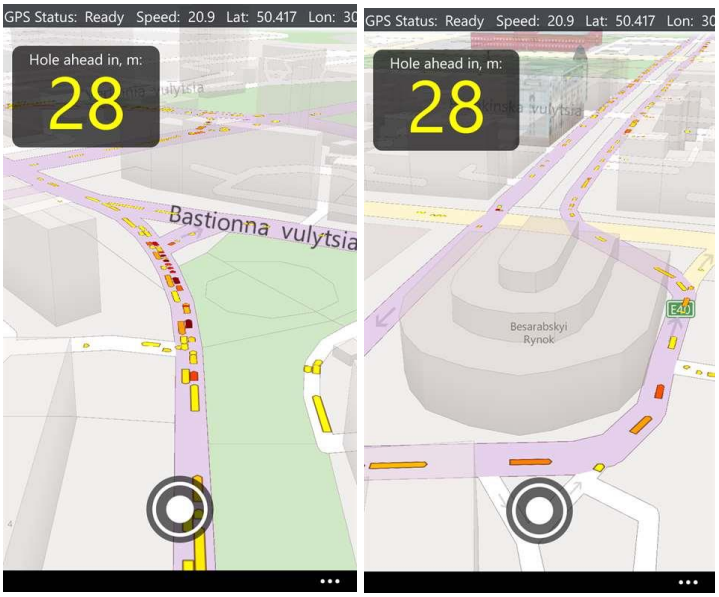
Intelligent-pothole-detection

This application was implemented by a team at Carnegie Mellon University, which records mobile routes during vehicle movement and creates maps that inform not only potholes, but also details of the condition of the road.



Pothole Tracker

This application detects and saves the pot holes, through the vibrations of the car, and then informs the user visually but also with sound for the pots he may encounter in his path.

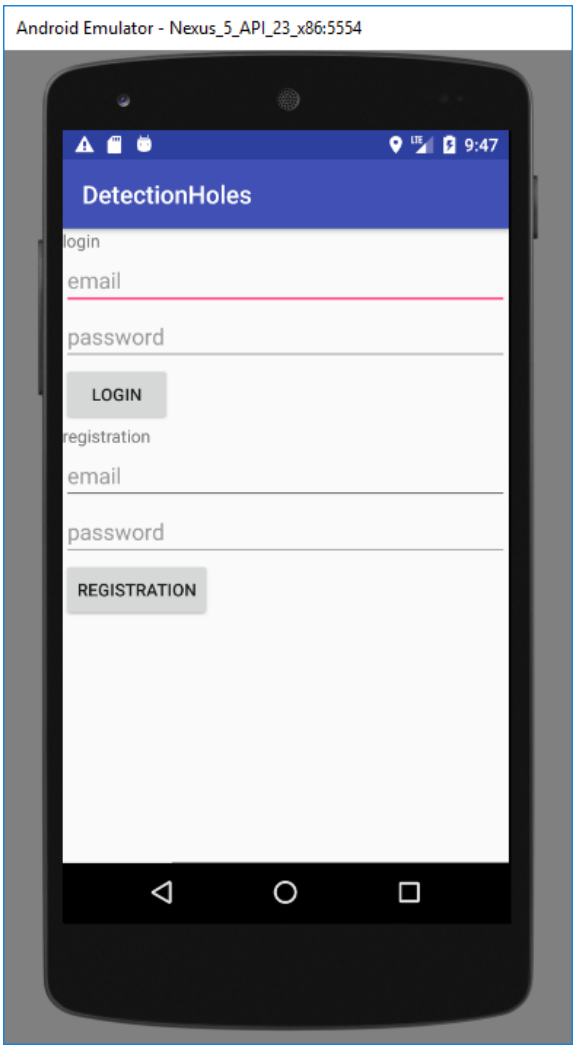


Παρουσίαση και Χρήση Εφαρμογής (User Manual)

Εφαρμογή Κινητής Συσκευής

Η χρήση της εφαρμογής είναι πολύ απλή, χωρίς να απαιτεί ιδιαίτερη αλληλεπίδραση με τον χρήστη, αλλά λειτουργεί αυτόματα.

Όταν ο χρήστης ανοίγει την εφαρμογή για πρώτη φορά θα πρέπει να δημιουργήσει έναν λογαριασμό με το e-mail του και με ένα password.



Log in

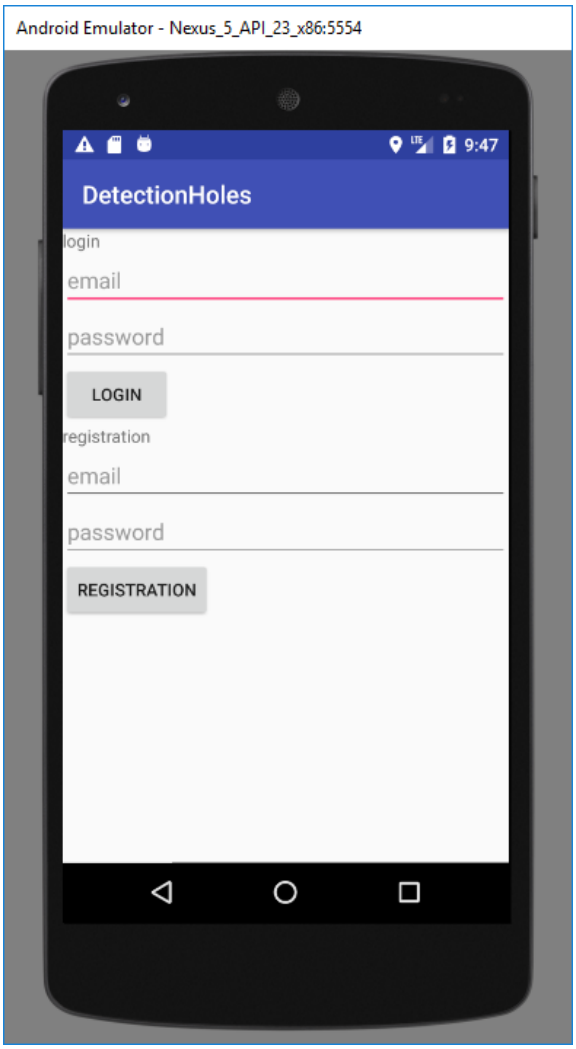
Αφού πραγματοποιηθεί το registration, τότε ο χρήστης είναι έτοιμος να εισάγει τα στοιχεία του στην εφαρμογή. Πριν προχωρήσει η εφαρμογή σε επόμενες λειτουργείες, θα του ζητήσει δικαιώματα για χρήση της τοποθεσίας του. Αυτή η ενέργεια θα γίνει μία φορά και δεν θα χρειαστεί να ξαναδώσει άλλη φορά την άδειά του, αφού η επιλογή του αποθηκεύεται.

Presentation and User (User Manual)

Mobile Application

The use of the application is quite simple, without requiring much interaction from the user, operating independently.

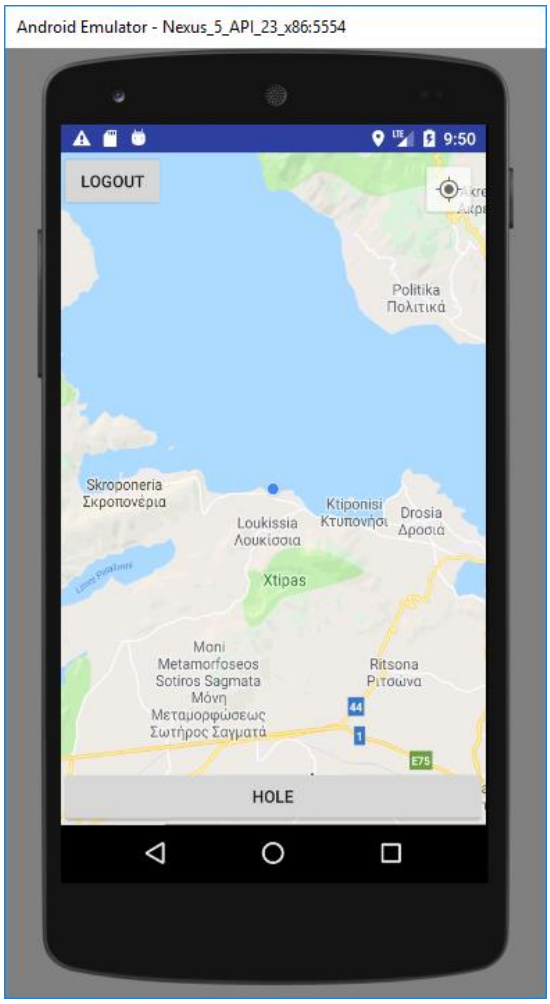
When the user launches the application for the first time, a username and a password account is required.



Log in

Once the registration is complete, the user is ready to fill his credentials. However, before the application displays any further features, it will require permission to location services. This action is only performed once with the selection being saved.

Αφού ο χρήστης εισάγει το e-mail και password και γίνει το confirmation με την βάση, τότε εμφανίζεται στον χρήστη ο χάρτης και το στίγμα του.

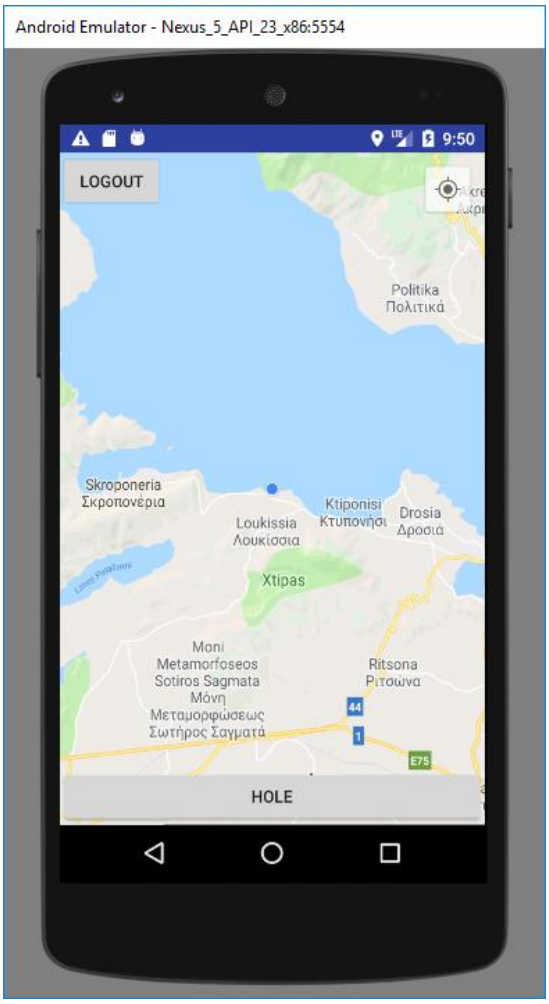


Κεντρικη Οθόνη

Για να λειτουργήσει σωστά η εφαρμογή και να έχουμε σωστή καταγραφή δεδομένων, θα πρέπει ο χρήστης να τοποθετήσει τη συσκευή του σε ένα σταθερό σημείο, έτσι ώστε να μπορεί να το βλέπει και να ενημερώνεται, αλλά και η συσκευή να καταγράφει πραγματικά δεδομένα και όχι λανθασμένα.

Αφού όλα τα παραπάνω βήματα έχουν ολοκληρωθεί σωστά, τότε ο χρήστης μπορεί να ξεκινήσει την διαδρομή του και την καταγραφή αυτής. Κατά την διάρκεια της διαδρομής του στον χάρτη, παρουσιάζονται το στίγμα στον χάρτη, αλλά και οι λακκούβες που υπάρχουν σε αυτόν, με αποτέλεσμα να είναι προετοιμασμένος για τυχόν κραδασμούς. Όσο κινείται το μέσον που βρίσκεται ο χρήστης, το τερματικό, με την χρήση των αισθητήρων του, ελέγχει την ποιότητα του οδοστρώματος. Εάν αναγνωρίσει έναν πολύ μεγάλο κραδασμό ή απότομη αλλαγή πορείας, τότε αμέσως ενημερώνει την βάση για ύπαρξη λακκούβας και κατά επέκταση και τον χάρτη του χρήστη όπου και εμφανίζεται ένα σημάδι.

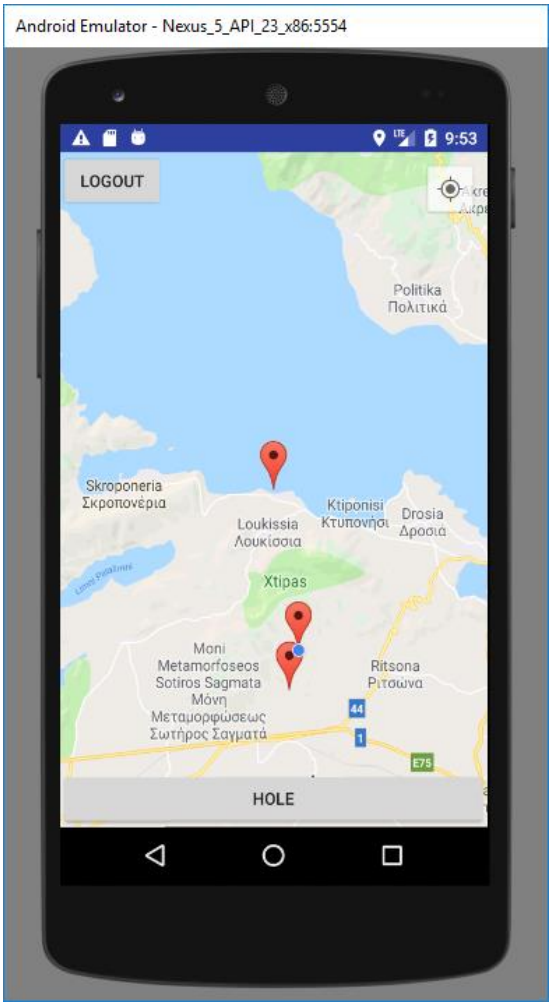
After the email and password confirmation is done with the database, the map and the user's pinpoint location appear.



Main screen

For the application to operate smoothly with proper data recording, the user must place his/her device in a steady place so he can actually see it and to avoid any wrong data recording.

With all the above steps completed, the user can start his trip and the recording. During the trip, the application displays the map and any possible pothole on the road, preparing the driver for possible vibrations. While the user's vehicle is on the move, the terminal, using the system sensors, can track the constructed quality of the road. If it encounters a heavy vibration or a sudden change of course, the database and the map are immediately updated with the existence of a pothole.



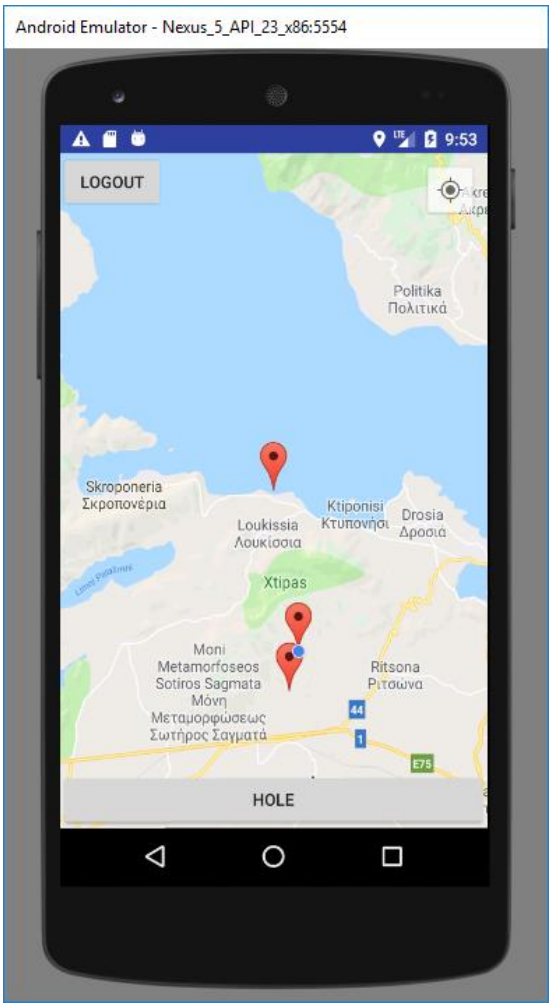
Καταγεραμμένα Στιγματα

Εάν η αλλαγή κατεύθυνσης ή ο κραδασμός που θα ανιχνευτεί από το κινητό είναι μεσαίας κλίμακας τότε δεν εμφανίζει τίποτα στον χάρτη, αλλά καταγράφεται στην βάση σαν λακκούβα υπό εξέταση. Εάν υπάρξει δεύτερη καταγραφή από τον ίδιο ή διαφορετικό χρήστη στο ίδιο σημείο τότε η λακκούβα θα εμφανιστεί στο χάρτη σαν κανονική "επικυρωμένη λακκούβα".

Η εφαρμογή δίνει επιπλέον την δυνατότητα στον χρήστη να καταγράψει χειροκίνητα μία λακκούβα πατώντας το κουμπί "HOLE".

Εφαρμογή Ιστοσελίδας

Η εφαρμογή συνοδεύεται με μία ιστοσελίδα η οποία ενημερώνεται για τις καταγεγραμμένες λακκούβες χωρίς κάποιος να είναι υποχρεωμένος να εισάγει e-mail και password.



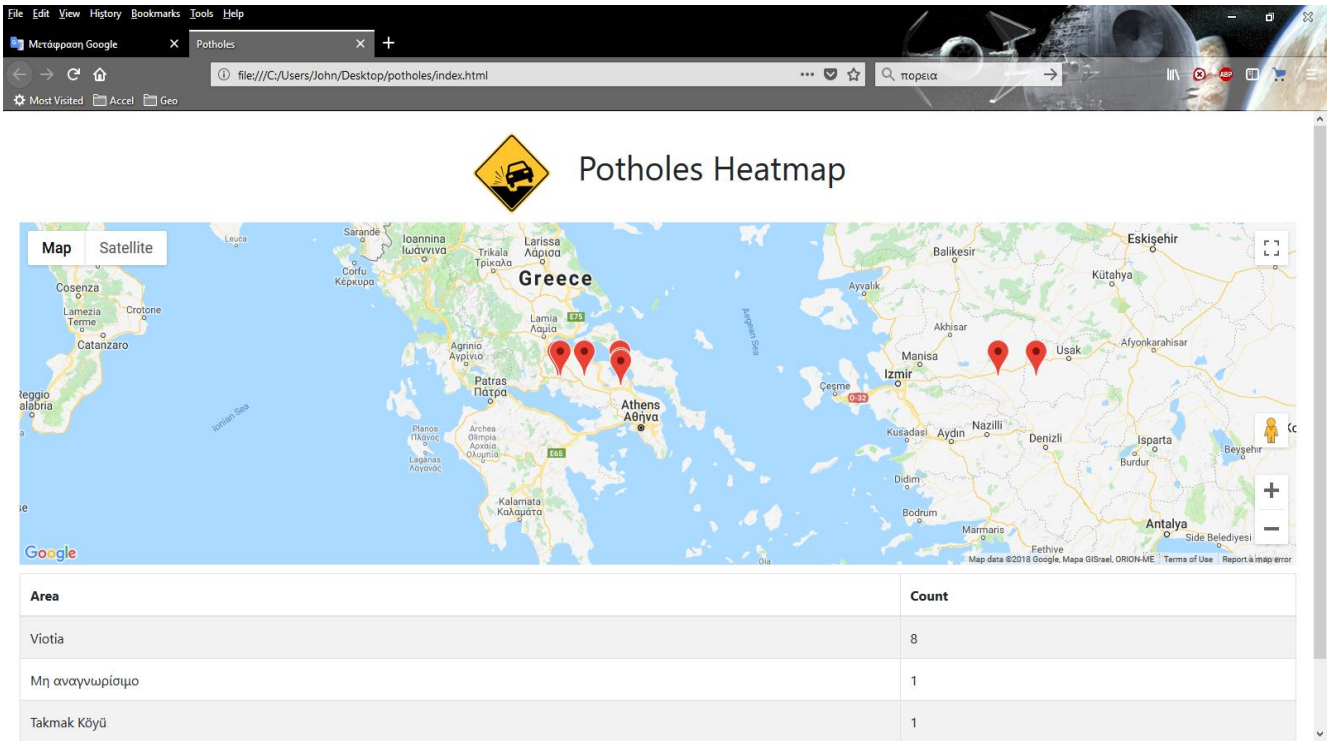
Mapped Potholes' Markers

If the change of course or the vibration is of medium range, the map shows no change but the database classifies it as a possible pothole. If there is a second incident at the same exact spot, then the pothole is valid and is now displayed properly on the map.

Additionally, the application allows the user to manually record an encountered pothole, using the " HOLE" button.

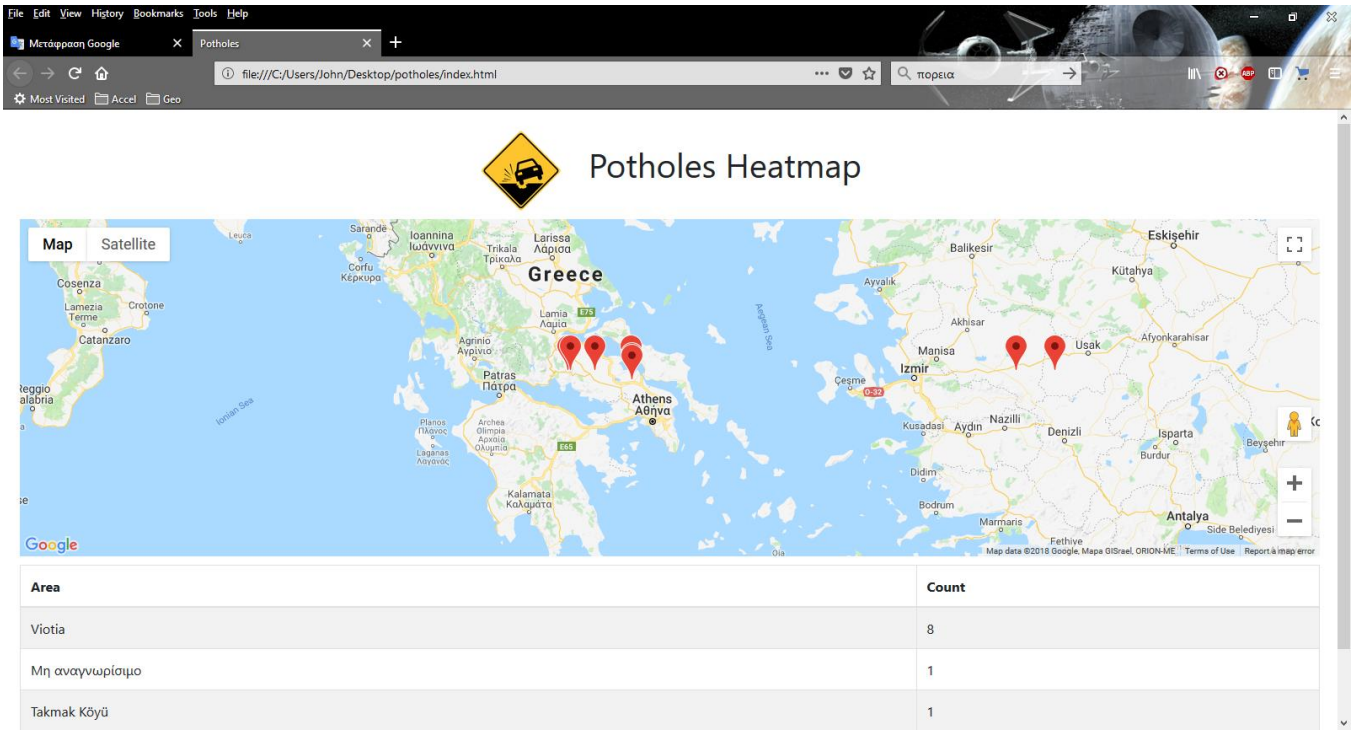
Website Application

There is also a website, recording and saving any possible potholes encountered without the need for a username of password.



Ιστοσελίδα

Στην ιστοσελίδα αυτή, ο χρήστης έχει την δυνατότητα να δει όλες τις λακκούβες που έχουν καταγραφεί από την εφαρμογή ανά τον κόσμο. Συγχρόνως, προβάλλεται πίνακας ο οποίος είναι σχεδιασμένος να παρουσιάζει σε ποια περιοχή υπάρχον λακκούβες αλλά και τα σύνολα τους.

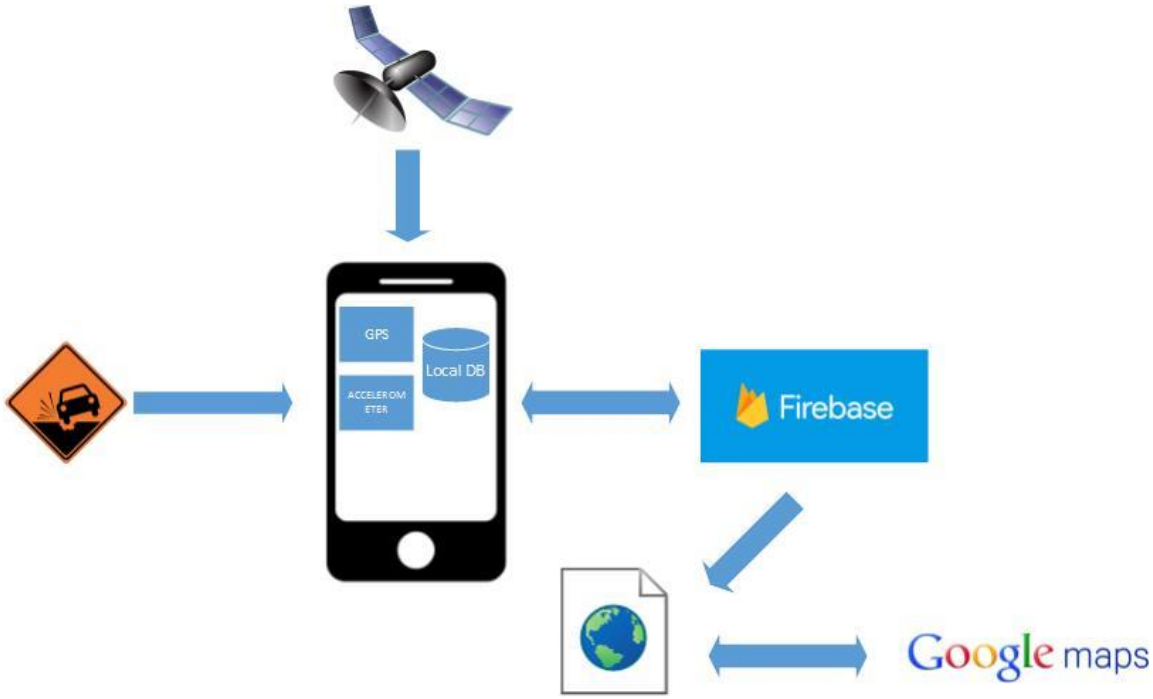


Website

On the website, the user may view all the recorded potholes around the globe. At the same time, a table appears displaying the potholes with their total sums per location.

Αρχιτεκτονική Συστήματος

Όπως εξηγήσαμε προηγούμενος το σύστημα αποτελείτε από τους αισθητήρες της κινητής συσκευής που εντοπίζουν την λακκούβα στον δρόμο και το στίγμα της, την απομακρυσμένη βάση δεδομένων που αποθηκεύονται τα δεδομένα και την ιστοσελίδα που ενημερώνεται από την βάση για την απεικόνιση στον χάρτη των λακκούβων. Επίσης το Google API δίνει την δυνατότητα να βρίσκουμε σε ποιο νομό είναι κάθε λακκούβα.



Απεικόνιση της αρχιτεκτονικής συστήματος.

Βάση Δεδομένων

Για την αποθήκευση των δεδομένων χρησιμοποιήθηκε η Firebase, η οποία είναι μια NoSQL cloud βάση δεδομένων (μη σχεσιακή). Η βάση αυτή συγχρονίζεται μεταξύ όλων των τερματικών συσκευών σε πραγματικό χρόνο και παραμένει διαθέσιμη ακόμα και όταν οι συσκευές μας δεν είναι συνδεδεμένες. Τα δεδομένα αποθηκεύονται σε JSON μορφή. Η δομή αυτή μπορεί και συνεργάζεται χωρίς κανένα πρόβλημα με εφαρμογές που είναι φτιαγμένες σε διαφορετικές πλατφόρμες όπως Android, iOS ή Javascript.

Με την Firebase υπάρχει η δυνατότητα να έχουμε ασφαλή πρόσβαση στα δεδομένα με κώδικα ο οποίος τρέχει στα ίδια τα τερματικά. Τα δεδομένα αποθηκεύονται τοπικά και, ακόμα όταν το τερματικό δεν έχει πρόσβαση στο internet, η εφαρμογή λειτουργεί κανονικά χωρίς κάποιο εμπόδιο. Όταν η συσκευή συνδεθεί πάλι στο ιντερνέτ, τότε αυτόματα έχουμε συγχρονισμό των τοπικών δεδομένων με την cloud βάση, αλλά και το τερματικό δέχεται όλα τα νέα δεδομένα που έχουν αποθηκευτεί στην βάση όσο ήταν εκτός διαδικτύου.

System Architecture

As we explained earlier, the system consists of the mobile device sensors that track the puddle on the road and its position, the remote database that stores the data, and the website updated from the database for presenting the potholes on the map. Google API also enables us to assign each pothole to which prefecture is located.

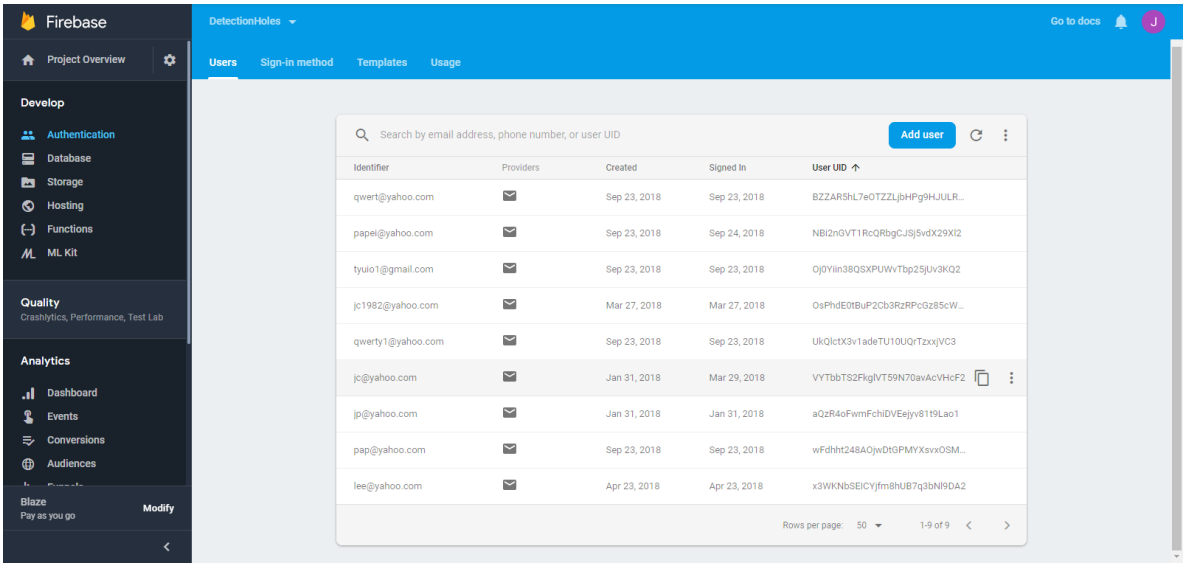
System architecture display.

Database

For the data management we used Firebase, a NoSQL cloud database. The database synchronizes with all the terminal devices in real time and stays active even when the devices have been disconnected. The data are being saved in a JSON extension. This allows for a seamless cross-platform integration, whether it is iOS, Android or Javascript.

Using Firebase, we have secure data access using code running at the terminals. The data may be even saved locally, enabling the application to continue its operation without any internet connection. When the device regains network access, there is an automatic synchronization of the local data with the cloud database and further downloading of any new data the user missed while he was offline.

Η Firebase μας δίνει ακόμα την δυνατότητα ταυτοποίησης ενός χρήστη και, μέσω της χρήσης της Firebase Authentication, έχουμε την δυνατότητα να βάζουμε κανόνες-δικαιώματα για το ποιος θα γράφει στην βάση και ποιος θα διαβάζει.



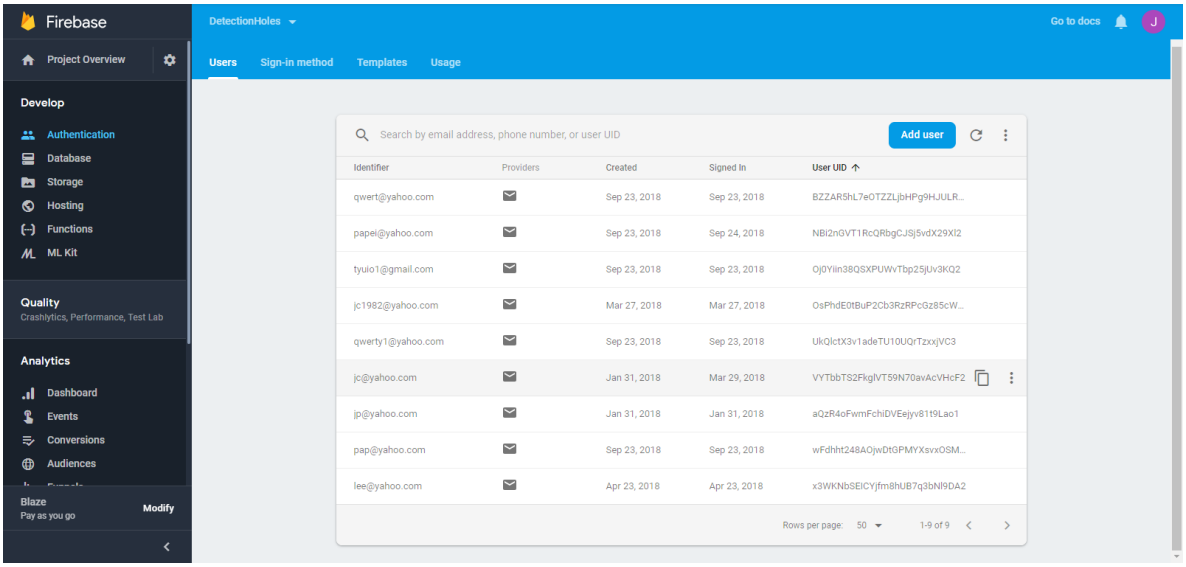
Firebase Authentication εργαλείο

Τέλος, η βάση, λόγω του σχεδιασμού και της δομής της, επιτρέπει να γίνονται μόνο γρήγορες λειτουργίες, το οποίο βοηθάει την εφαρμογή μας να είναι γρήγορη αλλά και να εξυπηρετεί πολλούς χρήστες μαζί χωρίς κανένα πρόβλημα.

Στην βάση έχουμε έναν πίνακα που αποθηκεύονται οι χρήστες μέσω της διαδικασίας του registration που γίνεται από την εφαρμογή κινητής συσκευής. Εκεί μπορούμε να δούμε το email του κάθε χρήστη και, από τις επιλογές που μας δίνει η firebase, να απενεργοποιήσουμε ή και να διαγράφουμε έναν λογαριασμό, ή να κάνουμε reset το password.

Το πρόγραμμά μας για την κύρια λειτουργεία του συνδέεται σε δύο πίνακες. Σε αυτούς τους πίνακες αποθηκεύονται οι συντεταγμένες των λακκούβων. Στο πρώτο πίνακα (Hole) αποθηκεύονται όλες οι αναταραχές που ανιχνεύονται από την εφαρμογή κινητής συσκευής. Αυτές οι αναταραχές δεν είναι απαραίτητο να είναι λακκούβες, και ως εκ τούτου δεν αποτυπώνονται στον χάρτη. Ο δεύτερος πίνακας (Holeconfirmed) περιέχει όλες τις λακκούβες οι οποίες, είτε λόγο της μεγάλης ανατάραξης καταγράφηκαν σαν λακκούβα από το σύστημα, είτε είχαμε δεύτερη αναταραχή στις συντεταγμένες που ήταν στον πρώτο πίνακα άρα όντως είναι λακκούβα ή κακό οδόστρωμα.

Firebase also provides the ability for user authentication and authorization. Taking advantage of the Firebase Authentication, we can actually provide read or write permission permissions to whoever we see fit.

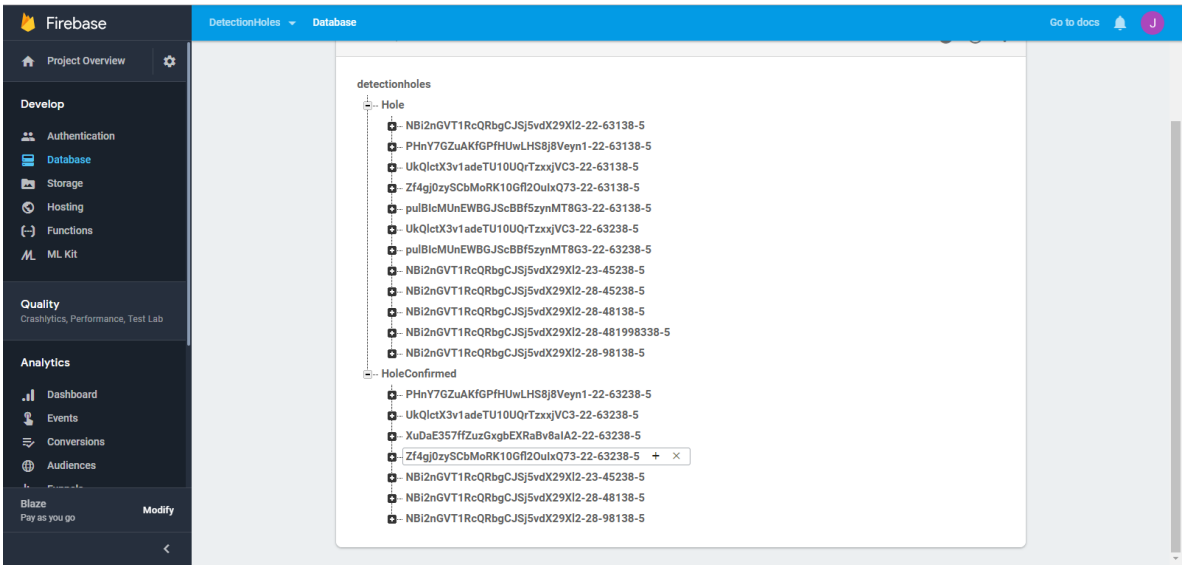


Firebase Authentication tool

Finally, the database, because of its design and architecture, allows for only fast operations, facilitating the application and allowing it to manage many users without any problems.

On a separate array we store each and every user has completed the registration process. That way, we can have access to each user's email address and using the Firebase features, we can delete accounts or reset passwords.

For its main use, the application is connected to two arrays containing the coordinates of the recorded potholes. In the Hole array, any possible encountered vibrations are being registered. There is no need for the vibrations to be actual potholes since they are not displayed on the map. The Holeconfirmed array contains all the confirmed potholes that were recorded from a heavy vibration or a sequential medium one.



Hole και HoleConfirmed πίνακες

Εφαρμογή Κινητού

Για την δημιουργία την εφαρμογής μας χρησιμοποιήσαμε Android για λειτουργικό σύστημα. Η επιλογή αυτή έγινε γιατί το Android σαν λειτουργικό χρησιμοποιείται κατά 76.6% σε συσκευές, αλλά και σε αυτοκίνητα και σε μηχανές, με συνεχώς αυξανόμενη χρήση. Ως εργαλείο ανάπτυξης

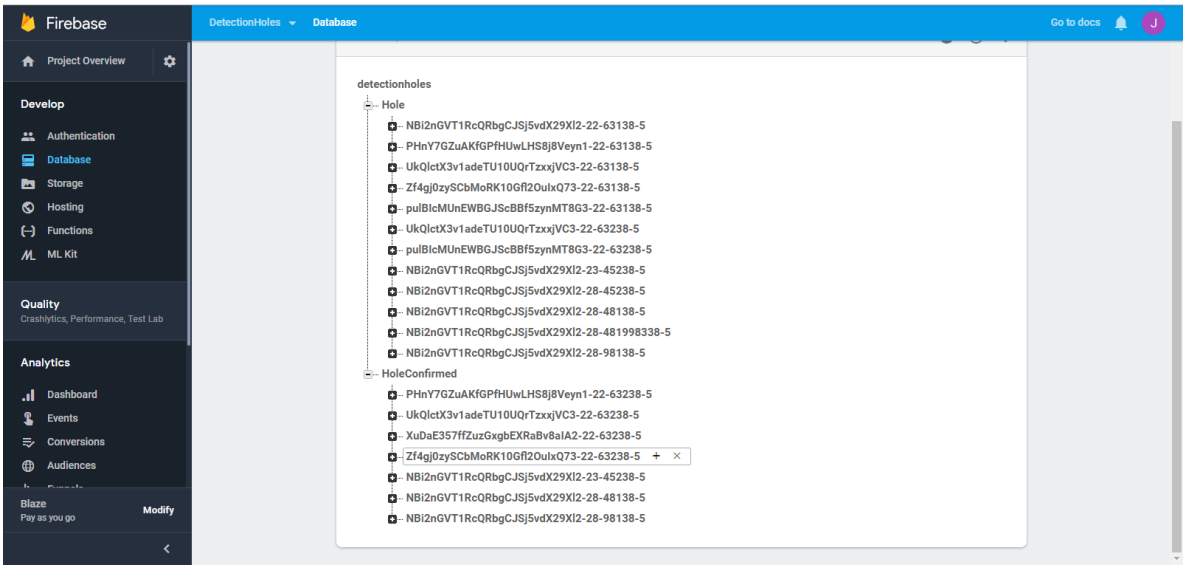
χρησιμοποιήθηκε το Android Studio γιατί είναι φιλικό προς τον χρήστη, με λειτουργίες όπως autocompletion, την οργάνωση του project αλλά και την δυνατότητα χρήσης emulators κάθε συσκευής.

Η εφαρμογή είναι γραμμένη σε java και τα κύρια API που χρησιμοποιεί είναι Geolocation (Google maps platform), Geofire, location service αλλά και το adnroid.hardaware framework.

Με την χρήση του Geolocation API που μας παρέχει η πλατφόρμα της Google Maps, μπορούμε με την χρήση Wi-Fi κόμβων και με την πληροφορία από κεραίες κινητής τηλεφωνίας να έχουμε πληροφορία για την ακριβή τοποθεσία μας. Το Geolocation API μαζί με το GPS μας δίνει με ακρίβεια την θέση μας και μπορούμε να την αποτυπώσουμε στον χάρτη.

Για να μπορέσουμε να ανιχνεύσουμε τις λακκούβες στον δρόμο χρησιμοποιήσαμε το αξελερόμετρο του κινητού. Το αξελερόμετρο είναι μια ηλεκτρομηχανική συσκευή που μετράει δυνάμεις επιτάχυνσης στους άξονες X,Y,Z. Ακόμα, μπορεί να μετρήσει στατικά δυνάμεις, όπως πχ η δύναμη της βαρύτητας. Έτσι, η εφαρμογή μας, μέσω του αξελερόμετρου, δέχεται συνέχεια δεδομένα από τις αναταράξεις του οδοστρώματος. Εάν η ανατάραξη είναι μεγαλύτερη μιας τιμής που έχουμε ορίσει παραμετρικά, τότε η ανατάραξη μεταφράζεται σαν λακκούβα και ενεργοποιείται ένα Geofire query το οποίο αποθηκεύει στην βάση μας στοιχεία της λακκούβας για τον εκάστοτε χρήστη.

Το Geofire 2.0 είναι μια open source βιβλιοθήκη που μας επιτρέπει να αποθηκεύουμε γεωγραφικά δεδομένα. Η Geofire χρησιμοποιεί την Firebase σαν βάση δεδομένων, το οποίο μας επιτρέπει να έχουμε συνέχεια ανανεωμένα δεδομένα. Με την χρήση μεθόδων που μας παρέχει η Geofire μπορούμε να ανιχνεύσουμε λακκούβες που είναι κοντινές μεταξύ τους με αποτέλεσμα να μην έχουμε διπλές εγγραφές λόγω σφάλματος του GPS.



Hole and HoleConfirmed arrays

Smartphone Application

For the creation of the app, we used the Android operating system. The choice was based on the 76.6% user coverage, but also because of its growing use in the automobile manufacturing. The user friendly interface of Android Studio was chosen, boasting features like auto completion, project management and the possible use of emulators.

The application is coded in Java and the main API being used are the Geolocation (Google maps platform), Geofire, location service and the adnroid.hardaware framework.

Using the Geolocation API provided by the Google Maps platform, we can take advantage of the WIFI nodes and using data gained from our ISP provider we can pinpoint our exact location. The Geolocation API along with the GPS provides unparalleled accuracy in the point adjustment.

In order to scan for any possible potholes along the way, we use the device's accelerometer. The accelerometer is an electromechanical device, calculating accelerating forces on the XYZ axis. In addition, it can calculate forces statically, like the gravitational force. Thus, the application using the accelerometer , is continuously being fed data from the road vibrations. If the vibration is of a higher value than a certain parameter, then it translates into a pothole and activated a Geofire query recording the pothole in the database for each user.

Geofire 2.0 is an open source library allowing us to save geodata. Geofire uses Firebase like a database, allowing us to continuously have updated data. Using the Geofire methods, we can track potholes who are close to each other so that we can avoid double entries in the confirmed array because of a GPS induced error.

Δείγμα Κώδικα Αλλαγής στον Sensor

Οι αλλαγές στις μετρήσεις του sensor της κινητής συσκευής διαχειρίζονται στο event onSensorChanged.

```
@Override
public void onSensorChanged(SensorEvent event) {
```

Οι τιμές του sensor από την αμέσως προηγούμενη εκτέλεση του event αποθηκεύονται σε μεταβλητές, έτσι ώστε να γίνει σύγκριση σε αυτές με τις νέες τιμές της τωρινής εκτέλεσης του event.

```
    float xc=x,yc=y,zc=z;
```

Οι τιμές του sensor αποθηκεύονται σε μεταβλητές, έτσι ώστε να υπάρχει πρόσβαση σε αυτές σε επόμενη εκτέλεση του event.

```
    x = event.values[0];
    y = event.values[1];
    z = event.values[2];
```

Γίνεται έλεγχος στους 3 άξονες του sensor. Αν σε οποιονδήποτε έχει υπάρξει αλλαγή μεγαλύτερη από προκαθορισμένη παραμετρική τιμή, τότε η εφαρμογή αναγνωρίζει την ύπαρξη λακκούβας.

```
    if(xc - event.values[0] > sersorChange || yc - event.values[1] > sersorChange
|| zc - event.values[2] > sersorChange ){
```

Γίνεται αρχικοποίηση του αντικειμένου geoQueryListener σε περίπτωση που δεν έχει γίνει ήδη.

```
            if (geoQueryListener == null) {
                geoQueryListener = new GeoQueryListener();
            }
```

Καλείται η μέθοδος foundHoleAt της κλάσης GeoQueryListener, με παραμέτρους συγκεκριμένες συντεταγμένες, προκειμένουν να διαχειριστεί η λακκούβα.

```
            geoQueryListener.foundHoleAt(mLastLocation.getLatitude(),
mLastLocation.getLongitude());
```

Sample Code of Sensor

The Changes in sensor measurements of the mobile device are managed in the onSensorChanged event.

```
@Override
public void onSensorChanged(SensorEvent event) {
```

The sensor values from the previous event are stored in variables to compare them with the new values of the current execution of the event.

```
    float xc=x,yc=y,zc=z;
```

The sensor values are stored to variables, so that we can have access to them on the next event fire.

```
    x = event.values[0];
    y = event.values[1];
    z = event.values[2];
```

Checks on the 3 axes of the sensor. If anyone has changed more than a predefined parametric value, then the application recognizes the existence of a pothole.

```
    if(xc - event.values[0] > sersorChange || yc - event.values[1] > sersorChange
|| zc - event.values[2] > sersorChange ){
```

The object geoQueryListener is initialized if it has not already been done.

```
            if (geoQueryListener == null) {
                geoQueryListener = new GeoQueryListener();
            }
```

The class GeoQueryListener, method foundHoleAt is been called with specific coordinate parameters, to handle the puddle.

```
                    geoQueryListener.foundHoleAt(mLastLocation.getLatitude(),
        mLastLocation.getLongitude());
```

Δείγμα Κώδικα Αποθήκευσης Λακκούβας

Παρακάτω αναλύεται το σημείο της εφαρμογής που εκτελείται για την αποθήκευση της λακκούβας. Συγκεκριμένα, το συγκεκριμένο κομμάτι κώδικα βρίσκεται στην κλάση GeoQueryListener, η οποία κάνει implement το interface GeoQueryEventListener.

Η αποθήκευση της λακκούβας πραγματοποιείται κατά την εκτέλεση του onGeoQueryReady event, το οποίο καλείται όταν όλα τα GeoFire δεδομένα έχουν φορτωθεί και έχουν εκτελεστεί όλα τα σχετικά events.

```
@Override
public void onGeoQueryReady()
{
```

Γίνεται έλεγχος για το αν δεν έχουν αποθηκευτεί λακκούβες στην ίδια περιοχή.

```
    if(queryLocations.size() == 0)
    {
```

Αποθήκευση της πρώτης λακκούβας για τον συγκεκριμένο χρήστη και τις συντεταγμένες που ανιχνεύτηκε η λακκούβα.

```
        //Save to unreliable
        geoFire.setLocation((userId + "-" + currentlocation.longitude + "" +
currentlocation.latitude).replace('.','-'), currentlocation);
```

Εκκαθάριση του object που περιέχει πληροφορίες για τα σημεία της λακκούβας.

```
        queryLocations.clear();
    }
```

Εάν έχει ήδη αποθηκευτεί λακκούβα στο ίδιο σημείο μια φορά τουλάχιστον.

```
    else
    {
```

Πραγματοποιείται loop για κάθε σημείο που βρέθηκε λακκούβα σε συγκεκριμένη περιοχή.

```
for(final GeoLocation gl : queryLocations.values())
        {
```

Γίνεται αναζήτηση του συγκεκριμένου σημείου (gl) και για συγκεκριμένη ακτίνα (radiusThreshold) στον πίνακα με τις επιβεβαιωμένες λακκούβες. Παράλληλα δημιουργείται event listener τύπου GeoQueryEventListener, αλλά και οι απαιτούμενα events που χρειάζεται να γίνουν overridden.

```
            //Save to reliable
            geoFireConfirmed.queryAtLocation(gl,
radiusThreshold).addGeoQueryEventListener(new GeoQueryEventListener() {
```

Αρχικοποιούμε το σύνολο των επεβεβαιωμένων λακκούβων που βρέθηκαν σε 0.

```
                int found = 0;
                @Override
                public void onKeyEntered(String key, GeoLocation location) {
```

Αυξάνουμε το σύνολο των επεβεβαιωμένων λακκούβων για όσες βρέθηκαν.

```
                    found++;
                }

                @Override
                public void onKeyExited(String key) {

                }
```

Sample Code of Storing the Potholes

Below we analyze the point of application that runs for storing the puddle. Specifically, this particular piece of code is in the GeoQueryListener class, which implements the GeoQueryEventListener interface.

Puddle storage is performed when the onGeoQueryReady event is executed, which is called when all GeoFire data has been loaded and all relevant events have been executed.

```
@Override
public void onGeoQueryReady()
{
```

Checks if there is not potholes saved at the same area.

```
    if(queryLocations.size() == 0)
    {
```

Save the first pothole from the user and the coordinates of the spotted pothole.

```
        //Save to unreliable
        geoFire.setLocation((userId + "-" + currentlocation.longitude + "" +
currentlocation.latitude).replace('.','-'), currentlocation);
```

Clears the object that contains the information of the potholes.

```
        queryLocations.clear();
    }
```

If a pothole is saved one more time at the same place.

```
    else
    {
```

A loop is made for each pothole found in a specific region

```
.       for(final GeoLocation gl : queryLocations.values())
        {
```

It searches for the specific point (gl) and a radiusThreshold in the matrix with confirmed pots. At the same time, a GeoQueryEventListener type event listener is created, and the required events need to be overridden.

```
            //Save to reliable
            geoFireConfirmed.queryAtLocation(gl,
radiusThreshold).addGeoQueryEventListener(new GeoQueryEventListener() {
```

We initialize the set of certified pots found at 0.

```
                int found = 0;
                @Override
                public void onKeyEntered(String key, GeoLocation location) {
```

We increment the set of certified pots for those found.

```
                    found++;
                }

                @Override
                public void onKeyExited(String key) {
```

```
                @Override
                public void onKeyMoved(String key, GeoLocation location) {

                }
```

Στην εκτέλεση του event onGeoQueryReady, δηλαδή όταν έχουν φορτωθεί όλα τα GeoFire δεδομένα και έχουν εκτελεστεί όλα τα σχετικά events.

```
                @Override
                public void onGeoQueryReady() {
```

Γίνεται καταγραφή εσωτερικού σχόλιου για έλεγχο του κώδικα που δηλώνει αν βρέθηκε λακκούβα στην περιοχή που ελέγχθηκε.

```
                    Log.i("GeoQ", String.format("Found %d results for location
%s,%s", found, String.valueOf(gl.latitude), String.valueOf(gl.longitude)));
```

Εάν δεν έχει αποθηκευτεί ξανά η λακκούβα.

```
                    if(found == 0)
                    {
```

Πραγματοποιείται η αποθήκευση της λακκούβας στον πίνακα με τις επιβεβαιωμένες λακκούβες, για τον συγκεκριμένο χρήστη και τις συντεταγμένες που ανιχνεύτηκε η λακκούβα.

```
                        geoFireConfirmed.setLocation((userId + "-" +
currentlocation.longitude + "" + currentlocation.latitude).replace('.','-'), gl);
                    }
```

Γίνεται εκκαθάριση του συνόλου με τις επιβεβαιωμένες λακκούβες.

```
                    found = 0;
                }

                @Override
                public void onGeoQueryError(DatabaseError error) {

                }
            });
        }
```

Εκκαθάριση του object που περιέχει πληροφορίες για τα σημεία της λακκούβας

```
        queryLocations.clear();
    }
}
```

Διαδικτυακή Σελίδα

Υλοποιήθηκε μια διαδικτυακή σελίδα στην οποία απεικονίζονται συνολικά οι λακκούβες που έχουν καταγραφεί μέσω της εφαρμογής κινητού και από τον εκάστοτε χρήστη. Στη σελίδα αυτή παρουσιάζεται χάρτης στον οποίο εμφανίζονται pins για την κάθε λακκούβα. Με τη δυνατότητα zoom στο χάρτη, ο χρήστης μπορεί να δει τα σημεία των λακκούβων αναλυτικότερα ή πιο συνοπτικά. Επιπλέον, εμφανίζεται πίνακας που καταγράφει την περιοχή που βρίσκονται οι λακκούβες, δηλαδή σε ποιο νομό αλλά και τον πλήθος σε κάθε νομό.

Η διαδικτυακή σελίδα απευθύνεται σε μεγάλο εύρος χρηστών, τόσο δημογραφικά, όσο και σχετικά με τις γενικότερες γνώσεις και ικανότητες αυτών στη χρήση εφαρμογών, αλλά και των διαφορετικών τεχνολογικών μέσων πρόσβασης στην εφαρμογή. Ως εκ τούτου, απαραίτητα χαρακτηριστικά της σελίδας κρίθηκαν να είναι γρήγορη, ελαφριά και φιλική προς τους χρήστες. Επιπρόσθετη ανάγκη ήταν η σελίδα να ακολουθεί τις σύγχρονες τεχνολογικές απαιτήσεις, δηλαδή να είναι responsive και με δυνατότητες επεκτασιμότητας.

```
                @Override
                public void onKeyMoved(String key, GeoLocation location) {

                }
```

Firing the onGeoQueryReady event, when all GeoFire data has been loaded and all relevant events have been executed.

```
                @Override
                public void onGeoQueryReady() {
```

An internal comment is logged to check the code indicating if a puddle was found in the checked area.

```
                    Log.i("GeoQ", String.format("Found %d results for location
%s,%s", found, String.valueOf(gl.latitude), String.valueOf(gl.longitude)));
```

If the puddle is not saved again.

```
                    if(found == 0)
                    {
```

Storing the pothole in the table with the confirmed potholes for the specific user and the coordinates detected by the puddle.

```
                        geoFireConfirmed.setLocation((userId + "-" +
currentlocation.longitude + "" + currentlocation.latitude).replace('.','-'), gl);
                    }
```

Clear all of the confirmed potholes.

```
                    found = 0;
                }

                @Override
                public void onGeoQueryError(DatabaseError error) {

                }
            });
        }
```

Clear the object containing information about the puddle points

```
        queryLocations.clear();
    }
}
```

Web Page

A web page was created that depicts the puddles recorded through the mobile application from all the users. This page depicts a map presenting pins for each puddle. With zooming on the map, the user can see the pothole points in more detail. In addition, there is a table listing all areas where the potholes are located, that is, in which county as well as the pothole sums in each county.

The website is targeting a wide range of users, both demographically and in terms of their general knowledge and skills in using applications, as well as the different technological means of accessing the application. Therefore, essential features of the page were considered to be fast, light and user-friendly. An additional need was for the page to follow the modern technological requirements, and to be responsive and expandable.

Η διαδικτυακή σελίδα έχει υλοποιηθεί με τεχνολογίες Javascript, GoogleMaps API, GeoFire API, Bootstrap, HTML5 και CSS. Με την χρήση HTML5 CSS και Bootstrap μορφοποιήσαμε την σελίδα, έτσι ώστε να φορτώνει γρήγορα και να είναι resnsposive, δηλαδή να φορτώνει και να είναι ευδιάκριτη σε όλες τις οθόνες που μπορεί κάποιος χρήστης να την φορτώσει . Με την χρήση της Javascript , Geofire και GoogleMaps API μπορέσαμε να εισάγουμε στην οθόνη μας τον χάρτη και να τον συνδέσουμε με την απομακρυσμένη βάση, έτσι ώστε να έχει πρόσβαση στα δεδομένα. Ακόμα, αφού συλλέξαμε τα στοιχεία, μπορέσαμε να χωρίσουμε και να μετρήσουμε τις λακκούβες ανά περιοχή.

Δείγμα Κώδικα Διαχείρισης Συνόλων των Λακκούβων

```
John_UI.UI = {
```

Δημιουργείται array.

```
    summaryArray: [],
```

Δημιουργείται function για βασικές ενέργειες αρχικοποίησης που χρειάζονται.

```
    init: function() {
```

Καταγραφή log για την καλύτερη παρατήρηση και εκτέλεση του κώδικα.

```
        console.log('Initializing UI');
```

Καλείται function που αρχικοποιεί απαραίτητα objects (πχ συνδεσιμότητα με βάση, εκκαθάριση του χάρτη όπου θα απεικονιστούν οι λακκούβες)

```
        John_UI.api.init();
    },
```

Δημιουργείται function για την ενημέρωση του πίνακα όπου καταγράφονται τα σύνολα των λακκούβων. Παίρνει σαν όρισμα την ονομασία της περιοχής της λακκούβας και τις συντεταγμένες της.

```
    updateSummary: function(locationName, latlng) {
```

Τα χαρακτηριστικά της λακκούβας (η ονομασία της περιοχής της λακκούβας και οι συντεταγμένες της) αποθηκεύονται στο array με τις λακκούβες.

```
        John_UI.UI.summaryArray.push({
                name: locationName,
                latlng: latlng
        });
```

Καλείται η function displaySummaryTable με όρισμα το μέγεθος του array όπου καταγράφονται οι λακκούβες.

```
    John_UI.UI.displaySummaryTable(John_UI.UI.calculateArrayCounts(John_UI.UI.summary
Array));
    },
```

Δημιουργείται function για τον υπολογισμό του μεγέθους του array όπου αποθηκεύονται οι λακκούβες. Παίρνει σαν παράμετρο το array αυτό.

```
    calculateArrayCounts: function(summaryArray) {
```

The web site has been implemented with Javascript, GoogleMaps API, GeoFire API, Bootstrap, HTML5 and CSS. Using HTML5, CSS and Bootstrap we formatted the page so it loads fast and is responsive, which means that it loads and is visible on all screens resolutions and devices that a user can load. Using Javascript, Geofire and GoogleMaps API we were able to import the map and connect it to the remote base to access the data. Furthermore, after collecting the data, we were able to divide and calculate potholes by region.

Sample Code of Managint the amount of Potholes

```
John_UI.UI = {

An array is created.
    summaryArray: [],
A function is created for basic initialization actions that are needed.
    init: function() {

Log for better observation and execution of the code.
        console.log('Initializing UI');

A function is called that initializes necessary objects (i.e. connectivity based, clearing the map where the pots will be depicted))
        John_UI.api.init();
    },

A function is created to update the table where potholes are recorded. The arguments it takes are the name of the pothole area and its coordinates.
    updateSummary: function(locationName, latlng) {

The attributes of the potholes (the name of the puddle area and its coordinates) are stored in the array with the puddles.
        John_UI.UI.summaryArray.push({
            name: locationName,
            latlng: latlng
        });

The function displaySummaryTable is called, with the array size argument where potholes are recorded.
    John_UI.UI.displaySummaryTable(John_UI.UI.calculateArrayCounts(John_UI.UI.summary
Array));
    },

A function is created to calculate the size of the array where the pots are stored. It takes this array as a parameter.
    calculateArrayCounts: function(summaryArray) {
```

Δημιουργείται ένα array που θα χρησιμοποιηθεί για την αποθήκευση των συνόλων των λακκούβων ανά ονομασία περιοχής.

```
var counter = [];
```

Αν στο array υπάρχει ήδη καταγεγραμμένη τιμή, τότε η τιμή αυτή αυξάνεται κατά μια μονάδα, αλλιώς αποθηκεύεται δημιουργείται νέο array item με τιμή 1.

```
summaryArray.map(function(item){
    counter[item.name] = counter.hasOwnProperty(item.name) ?
counter[item.name] + 1 : 1;
});
```

Η function τελικά επιστρέφει το ενημερωμένο array.

```
return counter;
},
```

Δημιουργείται function για την απεικόνιση ενός πίνακα που περιέχει τα σύνολα των λακκούβων ανά περιοχή. Παίρνει σαν παράμετρο ένα array που περιέχει τα σύνολα.

```
displaySummaryTable: function(counterArray){
```

Δημιουργείται ένα variable όπου θα καταγραφεί το html κομμάτι που θα απεικονίσει τη λίστα με τις περιοχές των λακκούβων και τα σύνολα αυτών.

```
var tableHtml = '';
```

Εκτελείται ένα loop στο array όπου έχουν καταγραφεί τα σύνολα των λακκούβων ανά περιοχή.

```
for (var areaName in counterArray) {
```

Χτίζεται το html, δημιουργώντας μια σειρά σε html πίνακα για κάθε item του array. Στη σειρά αυτή, δημιουργείται ένα κελί όπου εμφανίζεται η ονομασία της περιοχής της λακκούβας και ένα κελί με το σύνολο των λακκούβων για τη συγκεκριμένη περιοχή.

```
    tableHtml = tableHtml + "<tr><td>" + areaName + "</td><td>" +
counterArray[areaName] + "</td></tr>";
};
```

Η τιμή του html που χτίστηκε στο παραπάνω loop συνδέεται με ήδη δηλωμένο html πίνακα, με id results, ενημερώνοντας το κομμάτι που αφορά το tbody του πίνακα αυτού.

```
$("#results tbody").html(tableHtml);
}
};
```

Δείγμα Κώδικα Διαχείρισης Λακκούβων

Εκτελείται loop για κάθε ένα location στο οποίο έχει βρεθεί λακκούβα.

```
$.each(John_UI.mapConfig.locations, function (index, location) {
    John_UI.mapConfig.markers[index ] = new google.maps.Marker({
        position: new google.maps.LatLng(location.latitude, location.longitude),
        map: John_UI.mapConfig.map
        title: location.key
    });
```

An array is created, used for the storing of the sum total of the potholes per area.

```
var counter = [];
```

If there is already a recorded value in the array, the array item's value increases by one, otherwise a new array item is created having the value of 1.

```
summaryArray.map(function(item){
    counter[item.name] = counter.hasOwnProperty(item.name) ?
counter[item.name] + 1 : 1;
});
```

The function finally returns the updated array.

```
return counter;
},
```

A function is created for the display of an array containing the sum total of potholes per area. The parameter inserted is the sum total.

```
displaySummaryTable: function(counterArray){
```

A variable is created, building the html part displaying the areas list along with their sum total of potholes.

```
var tableHtml = '';
```

There's a loop inside the array recording the sum total of potholes per area.

```
for (var areaName in counterArray) {
```

The html code is built, creating a line in an html array for each of item. On that line, a cell is created displaying the name of the area and another one with the sum total of the area.

```
    tableHtml = tableHtml + "<tr><td>" + areaName + "</td><td>" +
counterArray[areaName] + "</td></tr>";
};
```

The html value that was built upon the loop is attached the the specified html table, using id results, updating the tbody part.

```
$("#results tbody").html(tableHtml);
}
};
```

Code sample – Pothole Management

Running a loop for each location a pothole was found.

```
$.each(John_UI.mapConfig.locations, function (index, location) {
    John_UI.mapConfig.markers[index ] = new google.maps.Marker({
        position: new google.maps.LatLng(location.latitude, location.longitude),
        map: John_UI.mapConfig.map,
        title: location.key
    });
```

Αρχικοποιείται το object του Geocoder, αν είναι η πρώτη φορά που εκτελείται η αποτύπωση των λακκούβων στο χάρτη.

```
if(John_UI.mapConfig.geocoder == null)
{
        John_UI.mapConfig.geocoder = new google.maps.Geocoder;
}
```

Δημιουργείται object που περιέχει τις τιμές των συντεταγμένων του location στο οποίο γίνεται το loop.

```
var latlng = {lat:location.latitude, lng: location.longitude};
```

Γίνεται έλεγχος αν έχουμε ήδη διαχειριστεί το συγκεκριμένο location. Ουσιαστικά, γίνεται σύγκριση των συντεταγμένων του current location του loop τις συντεταγμένες από οποιοδήποτε άλλο location έχει καταγραφεί ήδη.

```
var areaAlreadyExists = John_UI.UI.summaryArray.findIndex(function(obj) {
        return obj.latlng.lat === latlng.lat && obj.latlng.lng === latlng.lng
});
```

Αν το current location του loop δεν υπάρχει ήδη καταγεγραμμένο στη λίστα με τα markers, τότε πραγματοποιείται η διαχείριση του, έτσι ώστε να απεικονιστεί ο marker στο χάρτη.

```
if(areaAlreadyExists < 0){
```

Γίνεται χρήση της geocode μεθόδου της κλάσης google.maps.Geocoder, η οποία παίρνει παράμετρο το συγκεκριμένο location που διαχειριζόμαστε και επιστρέφει αποτελέσματα σχετικά με την γεωγραφική αποτύπωση του εν λόγω location.

```
        John_UI.mapConfig.geocoder.geocode({'location': latlng}, function(results, status) {
```

Γίνεται έλεγχος αν το location που ελέγξαμε είναι πραγματικό/σωστό.

```
        if (status === 'OK') {
                if (results[0]) {
```

Πραγματοποιείται έλεγχος για το αν υπάρχει καταγεγραμμένη τιμή ονομασίας της περιοχής που βρέθηκε το location.

```
                        var filtered_array =
results[0].address_components.filter(function(address_component){
                                return
address_component.types.includes("administrative_area_level_3");
                        });
```

Initializing the object of Geocoder,if it's the first time running the display of the potholes on map.

```
if(John_UI.mapConfig.geocoder == null)
{
        John_UI.mapConfig.geocoder = new google.maps.Geocoder;
}
```

An object is created containing the coordinates of the area that runs the loop.

```
var latlng = {lat:location.latitude, lng: location.longitude};
```

We identify if we've already checked the given location. In specific, there's a comparison between the coordinates of the current location of the loop with the coordinates of any other already registered location.

```
var areaAlreadyExists = John_UI.UI.summaryArray.findIndex(function(obj) {
        return obj.latlng.lat === latlng.lat && obj.latlng.lng === latlng.lng
});
```

If the current location of the loop doesn't exist in the markers list, then we proceed to display the marker on the map.

```
if(areaAlreadyExists < 0){
```

Using the geocode method in the google.maps.Geocoder class, taking as a parameter one specific location, it returns results relevant to that location.

```
        John_UI.mapConfig.geocoder.geocode({'location': latlng}, function(results, status) {
```

We confirm whether the location is given is valid.

```
        if (status === 'OK') {
                if (results[0]) {
```

Checking if there is a registered value of the location.

```
                        var filtered_array =
results[0].address_components.filter(function(address_component){
                                return
address_component.types.includes("administrative_area_level_3");
                        });
```

Αν υπάρχει ονομασία της περιοχής τότε αυτή καταγράφεται, αλλιώς καταγράφεται συγκεκριμένο διευκρινηστικό λεκτικό.

```
var locationName = filtered_array.length ?
filtered_array[0].long_name : "Μη αναγνωρίσιμο";
```

Καταγράφονται logs για τη διευκόλυνση της εκτέλεσης του κώδικα.

```
console.log(areaAlreadyExists);
console.log(locationName);
```

Πραγματοποιείται έλεγχος για το αν η τοποθεσία της λακκούβας έχει ήδη καταγραφεί.

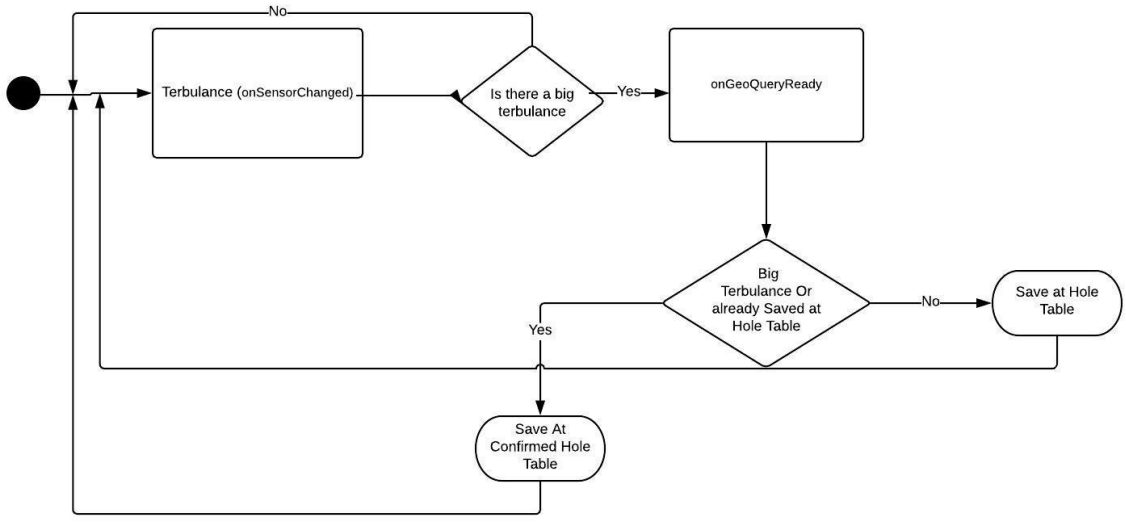
```
if (locationName && locationName !== areaAlreadyExists.name) {
```

Αποθηκεύουμε το location στη λίστα με τα locations που έχουμε διαχειριστεί.

```
                    John_UI.UI.updateSummary(locationName, latlng);
                }
            }
        }
        });
    }
});
```

Σχεδιαγράμματα UML

Στο παρακάτω UML διάγραμμα παρουσιάζεται η βασική λειτουργεία της εφαρμογής



Σχεδιάγαμμα UML της εφαρμογής

If an area location already exists, then it is recorded, otherwise a undefined area message is displayed.

```
                                    var locationName = filtered_array.length ?
filtered_array[0].long_name : "Μη αναγνωρίσιμο";
```

Logs are being recorded for a smoother code run.

```
                                    console.log(areaAlreadyExists);
                                    console.log(locationName);
```

Checking if the location of the pothole has already been recorded.

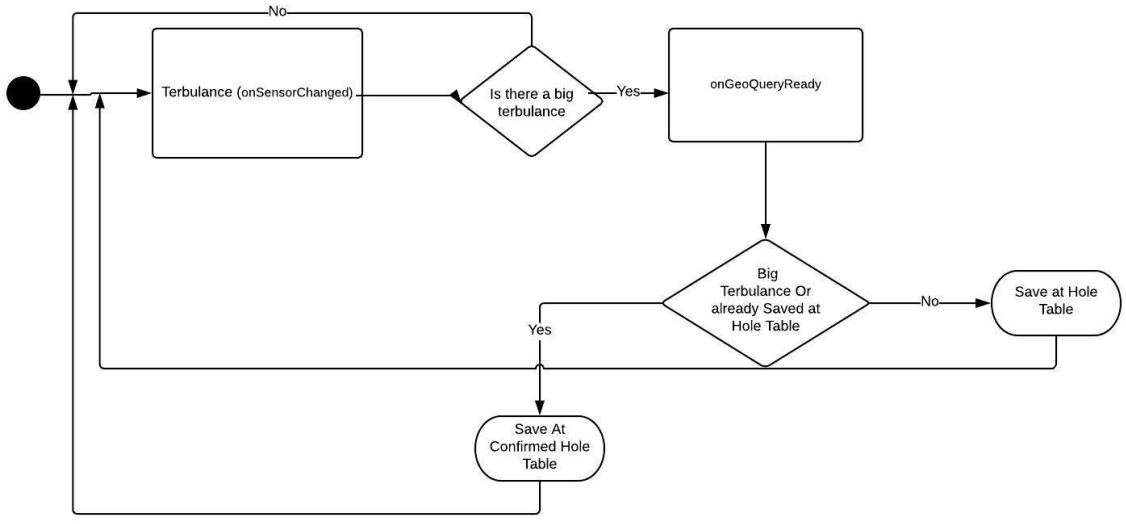
```
                                    if (locationName && locationName !== areaAlreadyExists.name) {
```

Saving the location on the already managed locations list.

```
                                        John_UI.UI.updateSummary(locationName, latlng);
                                    }
                                }
                            }
                            });
                    }
});
```

UML Diagram

The following UML diagram shows the basic configuration of the application.



Application UML Diagram

Συμπεράσματα και Μελλοντικές Επεκτάσεις

Συμπεράσματα

Η εφαρμογή καταγραφής λακκούβων μπορεί να φανεί πολύ χρήσιμη σε έναν οδηγό για την αποφυγή λακκούβων που μπορούν να προκαλέσουν κάποιο ατύχημα, αλλά και φθορά του οδικού μέσου. Ακόμα, με την χρήση της ιστοσελίδας κάποιος μπορεί να προετοιμαστεί για μία διαδρομή που θα ακολουθήσει. Επίσης, ένας κρατικός μηχανισμός μπορεί να ελέγξει και να ενημερωθεί για την ποιότητα του οδοστρώματος σε κάθε δήμο και να προβεί στις αντίστοιχες ενέργειες επιδιόρθωσης ή και να συλλέξει στατιστικά στοιχεία.

Μελλοντικές Επεκτάσεις

Η εφαρμογή αλλά και η ιστοσελίδα είναι μια πρωτότυπη ιδέα και με τις ανάλογες επεκτάσεις μπορεί να γίνει ακόμα πιο χρήσιμη από την τοπική αυτοδιοίκηση - κράτος, τον απλό χρήστη αλλά και ιδιωτικές επιχειρήσεις.

Αρχικά, στην εφαρμογή κινητού μπορεί να προστεθεί ένας ακόμα πιο έξυπνος αλγόριθμος που να ελέγχει πολλές βαθμίδες λακκούβας πχ μικρή, μεσαία, μεγάλη και στη συνέχεια να τις αποτυπώνει ανάλογα στον χάρτη με διαφορετικά χρώματα. Ακόμα, θα βοηθούσε τον χρήστη εάν ακούγονταν ένας ήχος που να τον προειδοποιεί όταν το μέσο πλησιάζει κοντά σε μία λακκούβα. Επίσης, θα μπορούσε να προστεθεί και λειτουργία για εύρεση διαδρομής με τις λιγότερες λακκούβες.

Στην ιστοσελίδα θα μπορούσαν να δημιουργηθούν διάφορα είδη χρηστών με τα αντίστοιχα δικαιώματα και πιθανές ενέργειες. Οι "απλοί" θα μπορούσαν να έχουν πρόσβαση στα δεδομένα τους (διαδρομές, καταγραφές λακκούβας). Οι κρατικοί μηχανισμοί θα μπορούσαν να έχουν την δυνατότητα επεξεργασίας των δεδομένων, δηλαδή να διαγράφουν μια λακκούβα ή να βγάζουν ένα οποιοδήποτε μήνυμα (επιδιόρθωση, μνήματα γενικά προς τους χρήστες). Αυτά τα μνήματα θα μπορούσαν να εμφανίζονται πάνω σε κάθε στίγμα.

Η εφαρμογή καταγραφής λακκούβων θα μπορούσε ακόμα να συμπεριληφθεί σε αυτοκίνητα νέας τεχνολογίας και τα δεδομένα να χρησιμοποιούνται από τα ίδια τα αυτοκίνητα για την ποιοτικότερη και ασφαλέστερη οδήγηση. Πολλές εταιρίες αρχίζουν να εισάγουν αισθητήρες στα αυτοκίνητα τους που να "διαβάζουν" το οδόστρωμα στο οποίο το μέσον να ανταποκρίνεται ανάλογα. Η Land Rover π.χ. χρησιμοποιεί ένα παρόμοιο σύστημα ανίχνευσης λακκούβων και αποθηκεύει το στίγμα τους ένα cloud σύστημα το οποίο με της σειρά του ενημερώνει τα υπόλοιπα αυτοκίνητα. Παρόμοια συστήματα ετοιμάζονται από την Ford και την Mercedes. Άρα, ένα τέτοιο σύστημα που θα μπορούσε να ενσωματωθεί σε όλα τα αυτοκίνητα και στο λογισμικό τους, ώστε να μπορούν όλα ανεξαρτήτως εταιρείας να ενημερώνουν και να ενημερώνονται για την ποιότητα του οδοστρώματος. Αυτό θα ήταν πολύ χρήσιμο για την ασφάλεια των επιβατών, αλλά και την αποφυγή φθορών του αυτοκίνητου.

Conclusion and Future Expansion

Conclusion

The pothole recording application can be extremely useful to a driver in order to avoid a possible accident or any likely vehicle damage. Furthermore, by using the website, the user can even prepare for the course he has opted. In addition, a state apparatus can check or get notified for the road quality in every municipality and proceed to upgrade it or collect analytical data.

Future expansion

The application as well as the webpage, is an original and unconventional idea that with potential future expansion may become even more useful in the local governing body - country, the average user and the business environment.

Initially, there is a possibility for an algorithm integration that can identify many different types of potholes like small, medium and large and use specific colors for map display. In addition, a sound alarming the user for the existence of a nearby pothole would also be beneficial. A function for finding the less pothole heavy course could also be included.

In the webpage, different types of users with different rights could be created having specific actions. "Simple" users could have access in their own data (courses, pothole recordings). State apparatus would edit the data, deleting a pothole or displaying a message of their choice (repair, messages directed to the users). Messages could be displayed on the map.

The pothole recording application could even be included in state of the art automobiles with the data being used by the vehicles, resulting in a better driving experience. Many car manufacturers, have begun integrating vehicle sensors that can “read” the road and act accordingly. Land Rover for example, uses a similar pothole recorder and uploads the data recorded to the cloud, thus sharing the information obtained with all its models. Ford and Mercedes have also begun deploying projects such as this. An application like this one however, could be integrated in all vehicles regardless of brand or manufacturing specifications. It could lead to tremendous accident avoidance and profound mechanical protection

Βιβλιογραφία / Bibliography

- https://www.pothole.info/the-facts/
- https://play.google.com/store/apps/details?id=com.hp.spothole&hl=en
- https://itunes.apple.com/us/app/street-bump/id528964742?mt=8
- http://www.streetbump.org/about
- https://medium.com/@percepsense/intelligent-pothole-detection-879ef635dd38
- https://www.microsoft.com/el-gr/p/pothole-tracker/9nblggh0ghz9?activetab=pivot%3Aoverviewtab
- https://www.landrover.com/experiences/news/pothole-detection.html
- https://developer.android.com/guide/topics/sensors/sensors_overview
- https://developer.android.com/guide/topics/sensors/sensors_motion
- https://developer.android.com/things/sdk/drivers/location
- https://code.tutsplus.com/tutorials/using-the-accelerometer-on-android--mobile-22125
- https://firebase.google.com/docs/
- https://firebase.google.com/docs/auth/
- https://github.com/firebase/geofire-java
- https://firebase.googleblog.com/2014/06/geofire-20.html
- https://github.com/firebase/geofire-js
- https://cloud.google.com/maps-platform/
- https://developers.google.com/maps/documentation/?__utma=236542612.267286862.1537700423.1537719710.1537719710.1&__utmb=236542612.0.10.1537719710&__utmc=236542612&__utmx=-&__utmz=236542612.1537719710.1.1.utmcsr=google|utmccn=(organic)|utmcmd=organic|utmctr=(not%20provided)&__utmv=-&__utmk=63491308&_ga=2.69138834.-267286862.1537700423
- https://developers.google.com/maps/documentation/javascript/tutorial
- https://developers.google.com/maps/documentation/android-sdk/intro
- https://www.w3schools.com/html/
- https://www.w3schools.com/Css/
- http://getbootstrap.com/
- https://www.w3schools.com/js/