

UNIVERSITY OF MACEDONIA  
POSTGRADUATE PROGRAM  
DEPARTMENT OF APPLIED INFORMATICS

---

# Smartphone Application for **ACCESSIBLE NAVIGATION**

---

MSc Dissertation

By

Tzanidou Alexandra

Thessaloniki, 06/2018

UNIVERSITY OF MACEDONIA  
POSTGRADUATE PROGRAM  
DEPARTMENT OF APPLIED INFORMATICS

SMARTPHONE APPLICATION FOR ACCESSIBLE NAVIGATION

MSc Dissertation

By

Tzanidou Alexandra

Thessaloniki , 06/2018



## ΕΦΑΡΜΟΓΗ ΠΡΟΣΒΑΣΙΜΗΣ ΕΣΩΤΕΡΙΚΗΣ ΠΛΟΗΓΗΣΗΣ ΜΕ ΧΡΗΣΗ ΚΙΝΗΤΟΥ ΤΗΛΕΦΩΝΟΥ

Τζανίδου Αλεξάνδρα

Πτυχίο Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας, 2016

# Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής  
Μαργαρίτης Κωνσταντίνος

Συνεπιβλέπων Καθηγητής  
Σακελλαρίου Ηλίας

Εγκρίθηκε από την τετραμελή εξεταστική επιτροπή την 26/06/2018

Σακελλαρίου Ηλίας	Μαργαρίτης Κων/νος	Ρεφανίδης Ιωάννης	Κασκάλης Θεόδωρος
-------------------	-----------------------	-------------------	----------------------

.....

Τζανίδου Αλεξάνδρα

.....

## **Abstract**

The main aim of this study is to investigate how the modern smartphone technology can assist people with visual impairments in indoor navigation tasks. We use the free and open indoor navigation service Anyplace, to design an indoor guidance system that is accessible, inexpensive, simple and user-friendly to different user groups disregarding their disabilities. The Android application that Anyplace offers, was extended and modified to serve also the needs of visually impaired users.

The presented system works well with the assistive applications that Android platform offers and provides various ways for interaction between the user and the system. The system is communicating with Anyplace server to inform the user about the information of the surrounding environment and guide him/her to the desired place in the building with accessible messages. The application can process, specific pre-defined user commands and location information from existing QR labels in the building.

This thesis is focusing on assisting the impaired users on indoor navigation tasks, but not on replacing the assistive means that the visually impaired user is already using. (e.g. long cane, guide dog)

Experimental results show the ability of the system to effectively communicate with the user and assist him/her in way-finding tasks in the building of the University of Macedonia. Keywords:

**Keywords:** Accessible Indoor Navigation, Accessible Design, Assistive Technologies

## **Acknowledgements**

My deepest gratitude to my Co-Supervisor Assistant Professor Ilias Sakelariou for giving me the opportunity to work with him, his willingness to share his academic knowledge, encourage me, and to offer his guidance on topics I have not previously encountered and also for his understanding!

I also would like to thank, my supervisor Professor Konstantinos Margaritis for inspiring me to be involved in the topic of Accessible Technologies and for his help, and the examiners Professor Refanidis Ioannis and Deputy Professor Kaskalis Theodoros for dedicating time while examining this thesis.

Furthermore, I would especially like to thank my family and my close ones for their patience and support throughout the project.

Last but not least, many thanks the Anyplace team in University of Cyprus for creating and offering publicly this great service.

# Contents

1	Introduction	1
1.1	Project Contribution	2
1.2	Terminology	3
2	Related Work	4
2.1	Systems For Indoor Navigation	5
2.1.1	Navatar	5
2.1.2	Smart Assistive Navigation System for Blind and Visually Impaired Individuals	6
2.1.3	A virtual blind cane using a Line Laser-Based Vision System and an Inertial Measurement unit	7
2.1.4	CrowdSensing: A crowdsourcing based indoor navigation using RFID based delay tolerant network	8
2.1.5	A Wireless Navigation System For the Visually Impaired	9
2.2	The Anyplace Application	10
2.3	Understanding the most common types of visual impairment	12
2.3.1	Visual Impairments	12
3	Analysis and design of the Accessible Anyplace Application	15
3.1	System Requirements	15
3.1.1	Functional Requirements	15
3.1.2	Non-functional Requirements	17
3.2	Stakeholders	18
3.3	Analysis and Design	18
3.3.1	Application Design Model	23
3.3.1.1	Package Diagram	23
3.3.1.2	Class Diagram	27
3.3.1.3	Sequence Diagram	29
3.3.1.4	The AnyPlace Accessible User Interface Design	32
3.3.1.5	QR Codes use	34
3.3.1.6	Voice Recognition	35
3.3.1.7	Navigation procedure	36
4	Implementation	37
4.1	Extension of Anyplace Application	37
4.1.1	Tools and Libraries used	37
4.1.2	QR Scanner implementation	37
4.1.3	Voice Recognition Activity implementation	38
4.1.4	Voice Feedback implementation	39
4.1.5	Modifications in Navigation Procedure	40
4.1.6	GUI implementation	42
4.1.7	Use of Layout Objects in Activities	44

4.1.8	Communication between Activities . . . . .	45
4.1.9	Minor Modifications . . . . .	46
5	Testing . . . . .	47
5.1	Functional Requirements Testing . . . . .	47
5.1.1	Application usage with Assistive Applications . . . . .	47
5.1.2	Evaluation against the functional requirements . . . . .	49
5.2	Non- Functional Requirements Testing . . . . .	51
5.3	Experimental Evaluation . . . . .	52
6	Conclusions . . . . .	56
6.1	Project summary . . . . .	56
6.2	Final Remarks . . . . .	57
	References . . . . .	58



## List of Figures

1	Cataract . . . . .	12
2	Glaucoma[2] . . . . .	13
3	Macular Degeneration[2] . . . . .	13
4	Diabetic Retinopathy[2] . . . . .	14
5	Color Blindness[1] . . . . .	14
6	The Anyplace Internet-based Indoor Information (IIN) Service Architecture. <a href="https://anyplace.cs.ucy.ac.cy/">https://anyplace.cs.ucy.ac.cy/</a> . . . . .	18
7	Anyplace Logger . . . . .	22
8	Package Diagram . . . . .	24
9	Anyplace package Impact Analysis Diagram . . . . .	26
10	Simplified Class Diagram related to the connections of UnifiedNaviagationActivity . . . . .	28
11	Sequence Diagram handleSelectedPoi method . . . . .	30
12	Sequence Diagram qrLocationHandling method . . . . .	31
13	Main screen layout. . . . .	33
14	PoIs Button is pressed . . . . .	33
15	Location QR label - UoM Main Entrance . . . . .	34
16	ScannerViewActivity Class Diagram . . . . .	38
17	VoiceRecognitionActivity Class Diagram . . . . .	38
18	Transparent Toast message implementation . . . . .	39
19	PoIsModel fields . . . . .	40
20	Difference identification between bearing and heading . . . . .	41
21	Extract of Button elements . . . . .	42
22	Example of Button's description label . . . . .	43
23	XML Declaration of invisible Spinner . . . . .	43
24	PoIsSpinner Creation . . . . .	43
25	Layout after QR Button is pressed . . . . .	44
26	Layout after VOICE Button is pressed . . . . .	44

27	PoIs Spinner and Button loading and handling of Click event. . . . .	44
28	Reaching Voice Recognition Activity with the use of Intent . . . . .	45
29	Start UnifiedNavigationActivity through Intent with putExtra() method.	45
30	Handling of received Intent . . . . .	46
31	Declaration of new Activities in Manifest file . . . . .	46
32	New permissions asked in AndroidManifest.xml . . . . .	46
33	Anyplace Architect - UoM Ground floor with PoIS . . . . .	52
34	Anyplace Architect - UoM Ground floor with Fingerprints . . . . .	53
35	Anyplace Architect - UoM 2nd floor of Applied Informatics tower with Fingerprints . . . . .	55

## List of Tables

1	Table of terms . . . . .	3
2	Functional Requirements . . . . .	16
3	Application's color selections . . . . .	33
4	Functional Requirements Evaluation - Part 1 . . . . .	49
5	Functional Requirements Evaluation - Part 2 . . . . .	50
6	Connectivity Scale . . . . .	53

# 1 Introduction

Although there is a lot of work done to assist navigation in indoor environments, there is still a need for tools that could assist the navigation of every user by considering the physical disabilities, that inevitably cause problems in the interaction between the user and the system. As the most sensitive user groups, in relation with the use of smartphone technology are the visually impaired people and the people with motor disabilities this study is mostly concentrated to them. Visually impaired people, in most times need assistance in the finding their way in unfamiliar spaces, as the Braille markings that are usually placed in the large buildings cannot sufficiently meet the important requirements for navigation. These requirements are the determining the user position and the proposal of a path which should be followed by the user to reach a specific point (target) in the building. In this thesis presents a proposal of an indoor navigation system that will be usable for a wide range of users, however focusing on visually impaired people.

In order to create a system that is user-friendly, accessible and inexpensive, we selected to use an existing indoor navigation service and modify the Android application provided, to make it suitable for use from visually and motor impaired people. On that point, it should be noted that the goal of this project is not to replace the assistive means that are already used by visually impaired individuals (e.g. long cane, guide dog), but the exploitation of already existing technology for the creation of a design that could serve the needs for indoor navigation to almost every user and improve the navigation experience.

The proposed application is using four different location/position sources to identify the user's position, and both manually created and crowdsourced information available in the server to assist the way finding procedure.

The thesis is structured as follows:

**Section 2 - Related Work:** A literature review of the most common patterns used in the development of modern navigation technology and of related projects is presented.

**Section 3 - Analysis and design of the Accessible Anyplace Application:** The requirements that the final product should fulfil and the logic behind their creation is presented. The design of the application is also described in this section.

**Section 4 - Implementation:** This section presents in detail, how the application was implemented.

**Section 5 - Testing:** This section includes the evaluation of the project, which is based in the criteria stated in Section 3 and a real case study that took place in the University of Macedonia with the evaluation of the final results.

**Section 6 - Conclusions:** The main achievements of the project and the proposed future work are discussed.

## 1.1 Project Contribution

In respect to the achieved goals of the project, the project contribution is presented below.

- A study and analysis of the core technologies used for the navigation of visually impaired people was made:
  - Key technologies were presented.
  - Current perspectives and valuable elements were analysed.
  - The benefits and the arguments for/against of each approach were discussed.
- The most common types of visual impairments nowadays were presented and an analysis of the issues that each type of visual impairment could cause to the interaction between a user and an interface was made. This could be a valuable source that could be used during the design phase of software, in order to make the final product approachable from different user groups and possibly make an impact.
- Based on the most common problems that should be solved during the design phase, an analysis of the requirements was done and the logic behind every proposed solution was presented in detail. As for the time, there are no clearly defined design patterns for designing applications that could be the same time usable and user friendly both for sighted and visually impaired users, the analysis of this logic could assumed as a starting point that could lead to final products which can be used from different user groups and possibly make technologies open even to people who cannot be considered as users yet.
- A successful open source project from the University of Cyprus, was modified and made approachable to more target groups. This contribution to an already existed and continuously maintained big project, could be an indication, that

almost every existing software could be modified to serve the needs of people with physical impairments.

- A fully functional and accessible application was delivered, which can support the navigation inside a big building, for almost every individual (except of the total blind people) and is usable for every user group disregarding potential physical disabilities.
  - The application is designed to solve common navigation problems inside big buildings.
  - As an Android application, can be installed for free in any Android smart-phone or tablet.
  - The modified source code is available to developers who want to extend the implemented functionality, add new features or understand better the logic behind the implementation was done.
- A case study for the University of Macedonia was created and is presented in detail in Section 5.3. This case study could be used as a base for providing a solution of existing navigation issues that were reported mainly from sighted students, during the scanning process of university's WiFi coverage.

## 1.2 Terminology

Throughout this report, unless otherwise stated, the terms below are defined according to:

**Table 1:** Table of terms

Built-in Sensors:	Accelerometer, gyroscope, magnetometer.
Digital Compass:	The software method for the orientation computation by the use of accelerometer and magnetometer, provided by Android.
Motor-impaired person:	In the term of this project as a motor-impaired person is considered the user with dexterity, coordination and strength problems.
PoI:	Point of Interest. Specific place inside a building.
QR code:	Quick Response code
QR label:	A label on which a QR code is printed.
dBm:	The power ratio unit expressed in decibels

## 2 Related Work

Since the successful usage of GPS in outdoor navigation brought important benefits in human daily life, significant research effort has invested for indoor localization and navigation solutions. Following that, the last three decades, a significant amount of work has been done on the field of blind and visually impaired navigation systems, both for indoor and outdoor navigation.

A variety of commercialized or free indoor positioning systems, have been proposed and implemented already. This work could be classified in three main categories based on the following approach: sensor based systems, beacon based systems and pedestrian dead reckoning based systems. This section is briefly presenting information about this classification and summarizes accessible positioning projects that are using the related technology.

**Sensor based systems** are using sensors[13, 10] such as cameras[8, 28, 20] and laser scanners[9, 7, 23] that are carried by the user in order to read the surrounding environment. The term read is used in this case to include both the cases that the systems are implemented to recognise pre-existing objects[25] such as stairs and other physical obstacles and inform the user regarding them, but also the systems that are continuously learn to recognise and classify different types of objects following the evolution of Machine Learning[12, 6, 22].

In the **beacon-based systems**, the user is supplied with a device which receives optical or radio frequency (RF) signals from existing beacons in the surrounding environment. In these systems the user position can be estimated by the help of the strength, time-of-flight or phase offered by the WI-FI[26], Bluetooth[27] or infrared signals[8, 16, 14].

**Dead reckoning systems** are the systems that are using dead reckoning technique in order to estimate the position of the user. The dead reckoning technique is calculating the current user's position based on a previously known position, by continuously interpreting readings from sensors such as accelerometers, magnetometers and gyroscopes that are carried by the user. This type of systems are usually used in parallel with, or are integrated in beacon based systems.[18, 19, 15]

In the following we describe in detail, some of the above mentioned systems.

## 2.1 Systems For Indoor Navigation

### 2.1.1 Navatar

The indoor navigation system Navatar[10] is a project created by the University of Nevada, having as main aim to assist students with visual impairments to navigation tasks in the university campus.

Navatar is a project with a unique characteristic, as apart from a set of commonly used techniques and sensors, is also using, as mentioned by its creators, "the user as a sensor". Navatar, which is the Android application part of the system, was implemented in Java programming language. On the first steps, the system was using 3D model maps created with Google Sketchup and Keyhole Markup Language (KML) as a representation of the building with named individual components. In the later steps due to the computational cost-effectiveness of handling in real time a 3D model on a smartphone, a parser for extraction in a 2D model was implemented for the path planning.

This system is capturing the ground truth<sup>1</sup> by the help of the Hagisonic StarGazer beacon based localization system and of passive landmarks with unique identifiers which were installed on the ceiling, as a result, the system is able to measure the accuracy of the estimated location on the fly.

In the first version of Navatar the user was equipped with:

- An infrared camera which was providing information about the ID, the distance and the angle of the closest landmark, but was also allowing the creation of a map with landmarks.
- A battery that was installed in the same belt as the camera.
- A tablet in a backpack which was recording the landmark information received from the camera and was calculating the user's location from the map, based on the relative landmarks.

In the following versions, the exponential growth of the technology allowed the reduction of all the above-mentioned equipment to a pair of smart Google glasses.

The last but the most important, part of the system, is the user as a sensor. In this case, the user has a core role in the use of the system, since after the user will announce the name of a landmark, the system will generate the appropriate directions. The user receives only one direction at a time in a form that makes use of the user's available senses. The user may be requested to follow a wall on his/her right hand, or count doors (i.e., "Follow the wall to your left until you reach a hallway intersection"). Only when the user will confirm the execution of the current task, will be able to receive the following directions. The truth of the confirmation is achieved by the use of pedestrian dead reckoning technique.

---

<sup>1</sup>Ground truth is defined as "The accuracy of remotely sensed or mathematically calculated data based on data actually measured in the field." (<https://support.esri.com/en/other-resources/gis-dictionary/term/ground%20truth>)



### **2.1.2 Smart Assistive Navigation System for Blind and Visually Impaired Individuals**

The main objective of this project[20] was the development of a safe and low-cost navigation system which will help the visually impaired people navigation in both familiar and unfamiliar areas for them.

Having reviewed a number of previous systems, Kinect was chosen as a suitable solution, because of the high accuracy that offers by the use of an IR depth sensor and the low power consumption.

The presented system is based on the Microsoft Kinect sensor which can efficiently help in the transformation of the physical world objects to voice messages that a visually impaired person can hear, with the help of an interface developed by the team, which is the mediator between the Kinect and the Matlab.

After the validation of the interface, the initialization of the processing unit, which in this case is a laptop, and the connection establishment between the Kinect sensor and MATLAB, the entire procedure that is being followed, to inform the user for possible obstacles was divided into 3 main stages.

1. The Kinect sensor detects all the dimensions of the objects in front of the user, by the help of the depth camera, followed by the conversion of all obstacles to black objects with the help of MATLAB.
2. The obstacles that are closer to Kinect sensor have higher pixel number than the other. When the pixels number is greater than a preset number, the system is informing the user with voice commands initiated by the MATLAB, via a Bluetooth headset.
3. The last stage includes the emergency functionality feature, in which, if the emergency button is pressed, a wireless connection between MATLAB and the mobile application is established by the help of a Mega Arduino Microcontroller, which is also responsible for the turn on and turn off procedures. In a situation like the above mentioned, MATLAB will start to send the photos taken by the Kinect to the mobile application, in order to inform a preassigned familiar person, about the user's situation.

Practically, the testing procedure that is being presented on this paper showed satisfactory results. Although, have to be underlined that the system was tested in an inside environment only and the testers were not blind.

The fact that all the above-mentioned equipment has to be carried by the user, accompanied by the importance of light intensity for the good functionality of the system, could be classified as the main drawbacks of the system.

### **2.1.3 A virtual blind cane using a Line Laser-Based Vision System and an Inertial Measurement unit**

The main interest of this paper[7], is the exploration of the environment around the user, through the use of a laser source. The proposed system includes a vertical laser line, which helps the user to investigate the nearby environment by swinging it horizontally, while the movement of user's hand is detected by the help of an inertial sensor unit. This procedure helps the system to make a complete scan of any obstacle in front of the user and provide instant and accurate feedback.

The system is analysing the information of the laser's projection on an obstacle in order to transform them in useful results for further usage in the classification phase that is following.

The classification is feasible, only if the previously mentioned inputs are combined with the coordinate frame of the camera. This procedure can easily give to the system, the 3D coordinates of the detected obstacle. After the classification of an obstacle, the distance between the object and the user is calculated, based on the distance of the closest laser point to the object and the estimated pointed angle, with respect to the navigation coordinate frame. Thereafter, the histogram of the laser pointer with the coordinates of the height is being analysed, in order for the obstacle to be classified in the appropriate object type. The user is informed for the existence of an obstacle or for a possibility of danger, by a single sound feedback, or voice indications.

As the currently described project make use of multiple sensors in fixed positions, the initial but most important procedure that should be followed was the calibration phase, due to the high importance of the estimation of the position and the orientation of each sensor, for the good use of the system. This procedure has to be done only once, taking into consideration that the relations between the sensors were clearly defined since, on that step, the camera and the laser sensor could be considered a unified system. As the basic part of this project, the previously described system is called laser-based system.

Before the first use, the system's height from the ground is also necessary to be determined. After the calibration step, every part of the system is considered as potentially usable. The main role of the Inertial Measurement Unit is the blind user's motion estimation, with the help of an accelerometer, that measures the user's acceleration and the gravity. Furthermore, the motion of the user's hand is estimated by the use of Kalman filter based, motion tracking algorithm. On that point, have to be noted (as mentioned in the paper), that the system, could result in a small error due to the slow walking tempo of the user.

Subsequently, when an obstacle is detected is directly classified with the help of the basic project's algorithm, by the use of the inclination angle and the laser point coordinates, also the distance between the user and the obstacle is being calculated.

The experimental results of the system's usage have shown that the classification procedure can be extremely successful, while the distance estimation could result in a small error. Conclusively, this paper is presenting a simple, low cost and most times accurate system but maybe the light intensity could be a drawback for the functionality.

#### **2.1.4 CrowdSensing: A crowdsourcing based indoor navigation using RFID based delay tolerant network**

This paper[14] proposes a framework which works like a task scheduler and resources manager, in a network which aims for the smooth navigation of a moving target in an indoor environment. All the services of the proposed system are used in the range of an RFID based network.

The system is based on the object's location information which can be obtained from RFID tags, that are already placed on the building's surroundings. The users of the system can easily communicate with the surrounding tags, using tiny devices like smartphones, in order to leave messages on the tags or to read the information that is included on them, by the use of the query and post operations, respectively.

Three types of roles are used for the characterization of the system's subjects, the searcher, the target, and the helper. Every user in the environment can be a searcher, a helper or a target at the same time. That fact, was a big challenge during the development of the system, due to the limited capacity of the RFID tags and the continuous user's movement. For that reason, two types of messages were used (the request message and the event message) and the building was represented as a graph. Both types of messages are carrying three parameters as described below.

Regarding the shape of an event message, that could be like  $E(H_j, V_j, t)$ , this shape states that the user  $H_j$ , passed by the vertex  $V_j$  at the time  $t$ . As for the Request message  $R(S, T, t)$ , states that the user  $S$  is looking for the user  $T$  since the time  $t$ .

In this manner, users have the chance to write on the surrounding tags, using the post-operation, and read from them using the request operations. Initially, all the potential users are in the normal mode, until they make a request to search for something/someone in the building and automatically jump to searcher mode.

After an operation is posted, the navigation procedure is starting with the help of two specially designed algorithms, which are responsible to collect and handle the current information of the nearby tags and the selection of the next vertex.

The performance evaluation of the current system was based on two large-scale experiments, one in simulation mode and one in realistic mode. The performance of the solution was evaluated regarding the average searching time, in four different types of path selection procedures. In both of the experiments, the results denoted that in crowd-sourcing, the number of the users in the environment and the average searching time are inversely proportional amounts.

A big advantage of the system is that is not dependent on a central server. On that point, there are no records of the positions of the active users. Thus, the user's privacy is totally protected. Furthermore, no time is wasted to the communication with a central server node and both the cost and the energy consumption of the system are affordable enough.

### **2.1.5 A Wireless Navigation System For the Visually Impaired**

The authors of this paper[23] focused on the development of a navigation system that could replace a cane stick or a guide dog.

The objectives of the research were clear from the initial phase of the project.

- The cost-effectiveness and the use of interoperable components.
- The integration between the Sonar Obstacle Detection mechanism and the Global Position System (GPS), in real time, with both low failure rate and sound accuracy.
- The selection of a suitable open source library that will allow the use of efficient coding techniques.
- The integration of the different technologies and the synchronisation of them in real time.
- Implementation of an emergency functionality.

In respect with the firstly adopted objectives, the prototype was based on some core ideas, as the incorporating GPS, sonar and wifi technologies, as this interplay could easily increase the accuracy, according to the existing literature. Whereas, at the same time could also decrease the cost-effectiveness comparatively with the cost of other sensor types. For the purposes of the study, the combination of the vibrational feedback with the sound frequency changing feedback could make the human-machine interaction more efficient.

All the previously mentioned functionality was not chosen by accident but with the help of a questionnaire designed in Braille system which was prepared by the team of the project as the first step of their study.

The results of the survey, that are presented in detail are indicative, in order for someone to understand the reason every part of the final product was chosen.

One result has shown that there are visually impaired people, who use to forget where they previously placed their cane. Another experiment showed that there is a huge percentage of visually impaired people, prefer the tactile feedback against the audio feedback and also, that the overhead obstacles are an important difficulty for them during the navigation process.

Another finding showed that a huge percentage of the potential users cannot easily afford the cost of a guide dog or of other blind navigation methods.

Regarding the system's prototype, was divided among three models. The Arduino YUN was selected as the core microcontroller for the Gold and Silver models, because of the wifi capabilities that offers and Arduino UNO was used for the Bronze one.

Furthermore, the wearable system consists of four SONAR sensors (three of them were placed on the trousers of the user and one on the hat), a GPS module which collaborates with Google Maps API, one vibrational motor for each sensor, a buzzer for the auditory feedback, one sensor for the computation of user's pulse rate and one for the measurement of the light intensity. The system offers also a button which is responsible to trigger the emergency functionality.

As a link between the hardware described above and the end user, a web application was developed, in order to allow customization of the needed parameters, such as the permissible limits for the pulse rate and the recipients of a possible emergency call.

The above-mentioned application can also be used by the users who are preassigned from the visually impaired person as familiar.

During the experiments procedure that is presented in this paper, the tester user was a member of the development team, which means that he/she was a trained, non visually impaired person. Although, the core results of the experiments, showed that the equipment is still destruction sensitive and that the location services need to be improved.

Supplementary to the development of the final product, a study about the business plan which includes the improvements mentioned before, resulted an affordable cost, even for a low budget target group that cannot afford to spend more than 200 USD per year for navigation equipment.

## 2.2 The Anyplace Application

In this project, we selected to work with an indoor navigation service which was not designed for blind or visually impaired people, through exploiting advantages offered by technological advances in smartphone services in order to achieve indoor localization.

Anyplace is an open, public service which was created and is continuously maintained till today from a scientific team of the University of Cyprus.

It is a modular, Internet-based Indoor Navigation (IIN) System, that is using crowdsourcing techniques to collect information regarding indoor environments[11].

The Internet-based Indoor Navigation Services make use of building models consisting of sub-models as floor models and landmarks, which are mainly stored in geolocation databases, in order to achieve indoor navigation by the help of signals (e.g. wireless or magnetic signals) that are available in the building[30].

*"Anyplace operates on top of a NoSQL data management back-end service, a JSON Application Protocol Interface (API), mobile clients for Web, Android and Windows Phone and a seamless integration with the Google Maps API for outdoor navigation and search, tiles and satellite view."*[29]

The Anyplace service consists of five modules, which are being described below. **The Anyplace Server** handles the complete application back-end logic as it has a two-way communication with every module of the service. The server delivers the indoor navigation instructions and supports every other feature of the service as the search information in cooperation with the JSON API, the crowdsourcing features in cooperation with Anyplace Logger, the interface of the design views of the data store, the tiling of the images and the user's authentication in the Architect module. **The Datastore**, which is the storage place of WI-Fi heat maps, the Indoor Models and, the RSS logs. **The Anyplace Architect** is a web app that uses the OpenStreetMap API and allows authenticated users to upload building models as image files and design the inner structure of the buildings by placing the Points of Interest (PoIs) that are existing in them.

**The anyplace viewer** is a web app that has a similar logic with the Anyplace Architect app, the main difference, as it is given by the name, is that it works only as a viewer, every user is allowed to use the web app without authentication, all the data are presented on the map and can be processed but they can't be modified by the user.

**The Anyplace Navigator** is a native Android application, that allows users to see their current location on the map or on the top of the floorplan, if they already are in the environment of an authenticated public building and navigate between the available PoIs. The navigator offers superb navigation accuracy, as it makes use of Wi-Fi radio maps constructed by the help of the Anyplace Logger and the built-in smartphone sensors.

**The Anyplace Logger** is integrated into the Anyplace Navigator application and allow users to record Wi-Fi readings from the available Wi-Fi points in the building and upload them to the server. This crowdsourcing contribution supports the construction of the Radio maps for indoor environments. The newly collected Wi-Fi fingerprints, after they are uploaded by the user, are always being compared with the already stored data in the Anyplace Data Store in order for the error of incorrect contributions to be filtered and eliminated.

Although Anyplace service was not implemented targeting accessibility issues, the advantages that offers as an actively maintained, free and open source project are a lot. The service, offers a cost-effective and sophisticated solution for indoor navigation, both for the user and for the infrastructure. The actual cost that the user has to pay, to use the Anyplace Navigator and Logger application is the smartphone price, and the Anyplace viewer is available for free use in any internet browser. Any user has the opportunity to use the Anyplace Architect and create the visual representation of a building at no cost, as soon as he has the floor plan images of the building. On that point, as infrastructure cost could be considered the cost of the WiFi Access Points creation, which as an infrastructure is usually available in every large-scale building nowadays. The Anyplace server, the source code of which is publicly available, makes any parametrization feasible. Lastly, the fact that the project is continuously maintained, ensures the availability and the extensibility of the service.

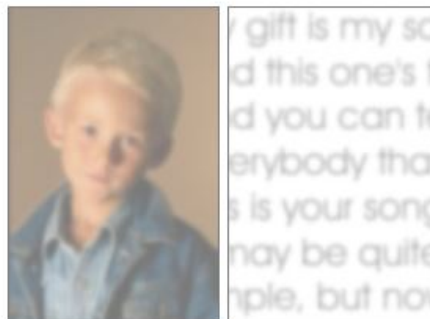
## 2.3 Understanding the most common types of visual impairment

The analysis of the different types of visual impairments[21, 3] is the essential step before the design process of an accessible application. This analysis will assist the understanding of the different limitations that visually impaired users have using their smart phones. This is the main difference between designing an application only for sighted users and designing an application that could be used by a wide range of users disregarding their disabilities. Commonly, a software that is designed only for sighted users has a vertical design in relation with other software available on the user's device, by the mean that, it is created to fulfil different user needs from other smart phone applications or to replace them. On the contrary, a software that is designed for users with different needs due to different disabilities should have a design that allows compatibility and good cooperation with the software that is already used by the user (e.g. screen readers).

### 2.3.1 Visual Impairments

*Blindness* refers to the total lack of vision or partial sight loss cannot be corrected with glasses or contact lenses. Users with blindness rely fully on screen readers, such as Talkback in order to use ordinary smart-phones. However, there are other various degrees of visual impairments, such as cataract, glaucoma, etc. as analysed in the following.

According to World Health Organization Cataract is the most common type of Visual Impairment. The most common symptoms of cataract are double or blurred vision and sensitivity to light and is usually felt like a clouding of the eye lens, as light is being diffused as it enters the eye, having as impact the low clarity of the visual image. *"High contrast is especially important for people with advanced cataract."*[2]



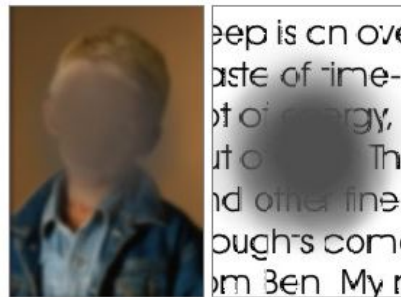
**Figure 1:** Cataract  
[2]

*Glaucoma* is a disease that damages the optic nerve of the eye. Glaucoma can cause, loss of peripheral vision to severe vision loss and blurred vision. "It can be particularly difficult to read text because the text seems faded and blurry." [2]



**Figure 2:** Glaucoma[2]

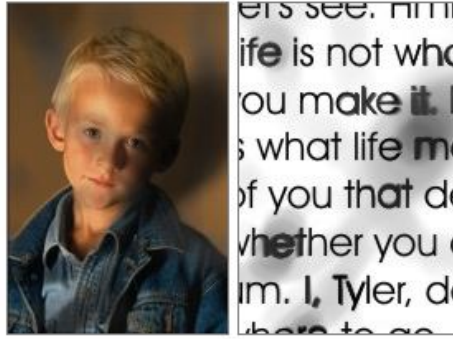
The macular is a tissue near the center of the eye's retina. *Macular degeneration* is age - related and mostly affects people older than 60 years old, as through the years the macular tissue becomes thinner. This disability causes difficulties when the user is trying to focus directly on a specific image. The result is often the opposite of the effect of Glaucoma.



**Figure 3:** Macular Degeneration[2]

*Diabetic Retinopathy* is a secondary disease of diabetes which occurs as retinal blood vessels leak into the retina, causing macular edema (swelling). The most common Symptoms of the disease are: blurred vision, eye floaters and spots, double vision. [4]





**Figure 4:** Diabetic Retinopathy[2]

There different kinds of color blindness, though all the types have as common the factor that the "patient"- user has difficulties in distinguishing the different colors. The different types of color blindness are:

- Protanopia and protanomaly (red deficiencies): Users with Protanopia don't have the ability to distinguish the red color, which is usually confused with the beige or green color. Protanomaly, is a light form of Protanopia as in some cases the users can distinguish the difference between the red and green color with more difficulties than a non-visually impaired user.
- Deutanopia and deuteranomaly (green deficiencies): Users with Deutanopia are not able to distinguish the green color, in this case, the red color tend to look lighter than in the case of Protanopia, but the end result looks very similar. Deuteranomaly is the light form of Deutanopia.
- Tritanopia (blue deficiencies): Tritanopia is rarer than Protanopia and Deutanopia. In this case, the user cannot distinguish the differences between green and blue colors, also yellow tend to show up as lighter shades of red.
- Rod monochromacy or achromacy (no color): This is the rarest type of color blindness. The users who experience this condition are not able to distinguish any color as they can only perceive an image as different shades of black and white.



**Figure 5:** Color Blindness[1]

### **3 Analysis and design of the Accessible Anyplace Application**

#### **3.1 System Requirements**

This project targets an indoor positioning system, that fulfills a set of core requirements, in order to properly assist people with visual impairments, while will still be usable for non-visually impaired people. Those core requirements, include low user cost, affordable cost for the owner of the building, effective accuracy, easiness of use and compatibility with other software that is essential for the use of smartphones by visually impaired people. Hence, emerges that the most important requirement in the framework of accessible design is understanding users with visual impairments and the different issues that they may challenge in correlation with the type of visual impairment.

In this section system requirements are presented. These requirements are divided as usual into functional and non-functional requirements. As the Anyplace Accessible Application was planned to work as an extension of the Anyplace original application, those requirements will determine the necessary modifications and extensions the system needs, in order to serve also people with impairments.

##### **3.1.1 Functional Requirements**

The functional requirements define the functions or the components of the system and express what the system should do.[24]

The system's functional requirements are given in the table below (Table 2), which presents each requirement by giving a description and a short explanation.

**Table 2:** Functional Requirements

Description	Explanation
Any user should be able to use the mobile application, without authentication.	Users must be able to use the application without using any credentials as soon as any internet connection is available.
The layout of the system have to be easily understandable by all the target user groups.	All the components of the main mobile application's layout should be usable and meaningful for the users disregarding their disabilities.
The system should provide accessible messages.	Users must be able to understand any message disregarding their disabilities.
The system should identify the location of the user.	User's location should be identified directly by the system, without any further action by the user, or with minimal actions.
The system must provide a list with the available Points of Interest.	Users should be able to be informed about the available Points of Interest in a building disregarding their disabilities.
The system must provide alternative ways for user input.	Users should be able to provide input to the system disregarding their disabilities. The system have to be able to process both haptic and voice commands.
The system must provide alternative ways for user output	Users should be able to receive any application output disregarding their disabilities. The system have to be able to give both visual and audio feedback.
The functionality of the system should be clear and easily accessible.	The system should provide easy access to application instructions of use.
The system have to be able to use multiple ways to locate the user.	The system should give output to the user and inform him/her about the current location. It should also provide accurate localization via the processing of the scanned information of any available QR labels.
The crowdsourcing component of the system have to be easily accessible to sighted users.	Any sighted user must have the opportunity to contribute data via crowdsourcing without the need of sign in to the system

### 3.1.2 Non-functional Requirements

The non-functional requirements are mainly expressed in the form of how the system shall be and are not related with what the system should do.[24]

- **Financial Cost for the end user:** The price that should be paid by the user is one of the most important factors, which should be taken into account during the design process of a software, that is targeting to a wide audience. This project is targeting the need for indoor navigation of all the visitors of a university building. Hence, the target audience range includes groups that usually cannot afford high costs, as is the group of students and the group of Visually Impaired people.
- **Cost of Infrastructure:** The setup infrastructure cost should be as low as possible both for public or private buildings, as the ability of a cost-effective system installation in an existing building is a factor that could affect the decision to adopt the system.
- **Convenience of use:** There are different factors to be considered for the determination of the user-friendliness of a system. In the frame of this project, the device that is used to "connect" the user with the main system must be easy to carry and allow the use of any other assistance needed by the user, as, for example the long cane or a guide dog. Another contributing factor is the training time required by the user in order to be capable to use effectively the provided software. Moreover, the system has to provide several and clear ways to receive and provide feedback, as every user presents individual limitations during the interaction with a system.
- **Compatibility with other Assistive Software:**As mentioned above, the application should work in parallel with any other assistive application used by the user. Compatibility limitations have to be taken into account from the first stages of the design process, as most times, this "assistant" applications tend to consume most of the system events that could be used to assist people with impairments, since this was the exact reason they were designed. For example Talkback, one of the most popular apps that assist visually impaired Android phone users, is consuming almost every common gesture that a developer could use. The obvious solution to have a procedure to block the use of an "assistive" application, is considered wrong as this could affect the behaviour of any other activity on the phone. A representative case scenario, demonstrating this difficulty is:
  - During the initialization, a process is blocking the behaviour of TalkBack in order to make use of the user gestures, that otherwise would have been consumed from Talkback.
  - The app itself is offering a smooth user experience to the visually impaired user.
  - The user is using the application until he/she receives a phone call.
  - The user is not able to reply because Talkback behaviour is blocked.

### 3.2 Stakeholders

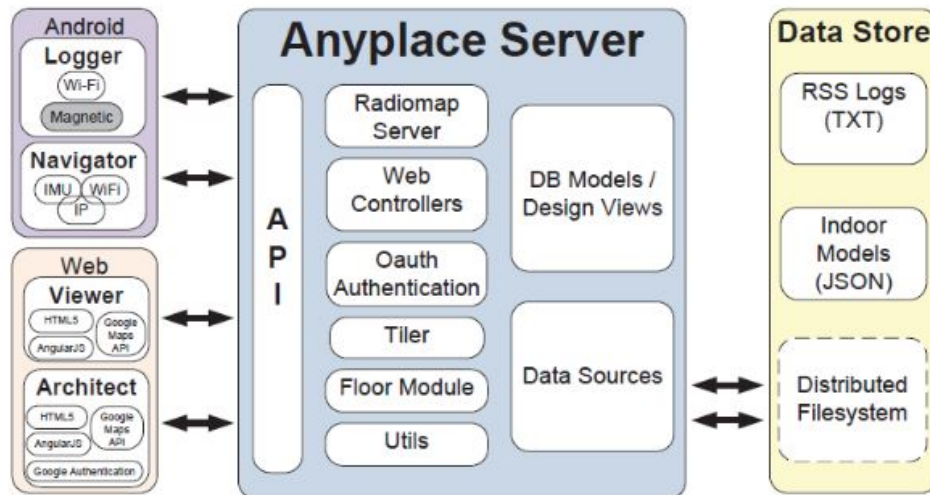
As it turns out from the requirements analysis, the people that will have direct or indirect interest to the service could be categorised in two groups.

- **Primary Users:** The users that are going to use the application, can be categorised in two sub-categories.
  - **Users with disabilities:** In this category is included every user that should be able to use at least the Navigator part of the application.
  - **Users without disabilities:** All the users that can use both the Navigator part of the application provided and the Anyplace logger, in order to provide more data regarding the walkable paths of the building.
- **Administrators:** Authenticated users of the Anyplace Architect web application. The authenticated users, have the right to modify the visual representation of the building by inserting/updating/deleting any component or sub-component of the building (e.g. floors, floor plans, Points of Interest, or even the building).

### 3.3 Analysis and Design

As aforementioned the Accessible Anyplace application is working as an extension of Anyplace official application. Thus, the first part of this chapter gives a detailed presentation of the AnyPlace application and details of various components are provided. This was considered necessary since AnyPlace is the basis on which the system is built.

The architecture of the Anyplace application is shown in the following figure.



**Figure 6:** The Anyplace Internet-based Indoor Information (IIN) Service Architecture.  
<https://anyplace.cs.ucy.ac.cy/>

## 1. Anyplace Server and Datastore

This project requires the communication between the mobile application and the Anyplace Server and NoSql Datastore, exchanging messages to support the delivery of indoor navigation directions, requested by the user, with the help of the stored radio maps. Despite this, the communication between these "modules" is a two way communication, as any user is able to use the logger module of the application and thus can directly contribute data to the server in the form of an RSS (Received Signal Strength) log file to support the crowdsourcing functionality of the mobile application.

The Navigator, Logger and Architect modules are exchanging messages with the server by the use of JSON objects. In order to achieve fast metadata retrieval, the server follows a big data architecture, having Couchbase as its back-end database.

Although, there is not an exact definition for "big-data", in the terms of this project will be used the most quoted definition given by McKinsey Global Institute (MGI), *"Big Data" refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage and analyse.* " [5]

As widely acknowledged, this definition raises a lot of questions, for that reason in the following paragraph will be explained in which way the Anyplace Service make use of the Big Data Technologies.

The users of **Anyplace Logger** have the opportunity to record RSS readings from the Wi-Fi APs (Access Points) around them and upload these readings to Anyplace Server via the Web 2.0 API for crowdsourcing the RSS radio map of the selected building. This is referred as RSS fingerprinting.

The Administrators are able to insert and modify Building's information through the **Anyplace Architect** web Application, these data are also being stored in **Anyplace Server**.

In parallel, the Server is responsible to deliver the stored and processed information both to the users of **Anyplace Viewer** Web application and of **Anyplace Navigator** mobile application.

An important part of the RSS fingerprinting functionality of the service is that the generic radio map is essentially constructed by the help of the RSS module, which does not output the absolute RSS values contributed from the various devices, but the fused data that become available from the Signal Strength differences. As different mobile devices are reporting RSS values in different ways, that factor could potentially compromise the quality of the crowdsourced radio map. Additionally, the filtering module is validating the RSS contributed data, by correlating them with the MAC addresses of the AP's of previously contributed data, for the same user locations available in the DataStore. [17]

## 2. Anyplace Architect

The Architect is a web application accessible with Google authentication procedure. After the successful authentication, the user (in the term of this project - administrator) can insert new buildings or even campuses on the top of Open-Street maps. Every campus can include multiple buildings, as every building can include multiple floors.

After the insertion of the relative denotation of a building with the use of a marker, the administrator is able to upload any floor plans available, in image format and manually place them in the right map location by rotating or scaling them with the help of the floor editor.

Following that, the virtual environment of the building is ready and the placement of Points of Interest can start. The aforementioned procedure can be done relatively easy, as soon as the information regarding the Points of Interest is available (drag-and-drop procedure). Every time that a new Point of Interest is being inserted, the system is asking simple information about the new insertion, as the name, the description and the type of the PoI.

All the PoIs that are available in a building can be connected with each other, directly or indirectly by the use of connector markers, in order to indicate the available walkable paths that would later assist the use of the Navigator part of the mobile application. Thus a graph containing the possible paths is created.

The user-friendly environment of the Architect offers a wide range of functionalities to the administrator, such as.

- Through the Wi-Fi section, the user can view or delete all the available fingerprints, the fingerprints reported in relation with the time they were reported, the WiFi coverage and the estimated AP positions of them. For every WiFi AP, information regarding the MAC address and the manufacturer is also available.
- Through the Buildings/Campuses section, a new Building can be added, or an existing one can be edited, deleted, shared in social media directly from the application or by the use of the generated building/campus related URL. The system also offers the opportunity to backup the whole building/campus in the shape of an exported compressed file in .zip format. On that point, it should be mentioned that the sections of Buildings and Campuses are two different sections that are being analysed together as they are similar.
- Any existing floor plan can be deleted and re-inserted by the help of the floor editor in floor section as at any time a new floor can be inserted.
- All the Points of Interest can be updated/modified, change place or even be deleted, commonly with any connection related to them. Furthermore, the system supports the export and import of any information related to the PoIs in JSON format.

- Additionally, the administrator has always the choice to make the visual representation of a building publicly available or just keep it private and share it through a URL.

As a result of the above-mentioned functionalities, all the export, import, and backup functionalities allow not only the modification and restore of a building but they offer the opportunity of reuse of the already existing work for a different purpose.

### 3. Anyplace Viewer

The Viewer was implemented in the same codebase as the Architect application but with a different purpose. It is a web application, available to every user without the need for authentication. As it is obvious due to the name, nothing related to the structure of the building could be modified with the use of the application, which is designed for viewing and sharing purposes. The layout of the application is clear and simple, the user has the opportunity to view his/her current location in the building, inspect all the available PoIs and their information, take information about the walkable path between two PoIs, see the navigation between his/her position to another PoI and share all the aforementioned information via a URL.

### 4. Anyplace Navigator and Logger

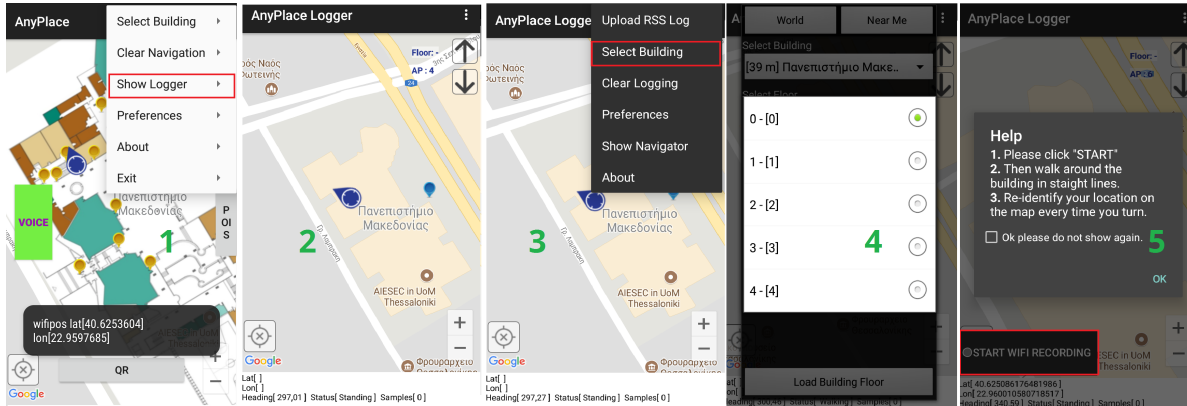
This paragraph presents a closer look to Anyplace application architecture, in combination with the modifications and new enhancements done in order to make Anyplace accessible to people facing disabilities.

Both Anyplace and Accessible Anyplace application are combining the functionalities of the Navigator and the logger. As mentioned above in Stakeholders subsection 3.2, the logger is the part of the app that can be used from sighted volunteers for the collection of RSS data in order to crowdsource data for the creation of the radio map. A basic requirement for the use of the logger is that the user should be inside the building to be scanned, in order for the possibility of wrong signals collection to be minimised. As the crowdsourcer is opening the application can switch to the logger in just few screen taps. After the Anyplace logger screen appears, the user has to follow the instructions displayed on the it to select the building and the floor to be scanned. Afterwards user's choice, the fetching of the selected floor plan is triggered and the user is ready to start the logging procedure.



The user has to follow a simple procedure that is described in help logger's message as described below.

- (a) After the correct positioning of "self marker", press the start indication on the lower left corner of the screen.
- (b) Walk around the building in straight lines.
- (c) Re-identify the location on the map before every turn.



**Figure 7:** Anyplace Logger

During the use of the logger, the crowdsourcer has the opportunity to inspect the collected fingerprints and the samples are being collected on the fly, to keep track of the paths that are not satisfactory covered. The information regarding the collected fingerprints is stored locally on user's smartphone. When the logging procedure is completed, the user can upload the files created during the procedure. After every successful uploading, the results of the logging procedure are available for inspection in Anyplace Architect web application and for transparent use in the Navigator part of the application. The term transparent in this case is used to underline that, although there is no difference in Navigator's layout for the end user, every uploaded RSS log file has an impact to the accuracy related to user's navigation.

The Navigator part of the native Android Accessible Anyplace Application offers to any user the opportunity of navigation in any public building that is already registered in Anyplace Servers by the help of the Anyplace Architect web application. By the use of the navigator, the users can see their current location on the top of the floorplan and navigate between Point of Interest inside a building. The discovery and selection of the available PoIs in a building can be done in multiple ways. This implementation was followed in order to assist a wide range of users to use the app disregarding of any possible disability they have. The

way that the new enhancements could make the application usable to people with different disabilities will be explained later in this chapter.

The application makes use of built-in sensors, available in almost every Smartphone nowadays to enhance the navigation experience. During the launching procedure of the application, the Google Geolocation API is used, in order for the user location to be determined. After the successful estimation of the user's current location, the building map and the associated PoIs are automatically loaded. Following that, the available RSS radio map is being downloaded on user's device, and the estimated position of the user is being displayed on the top of the map.

### **3.3.1 Application Design Model**

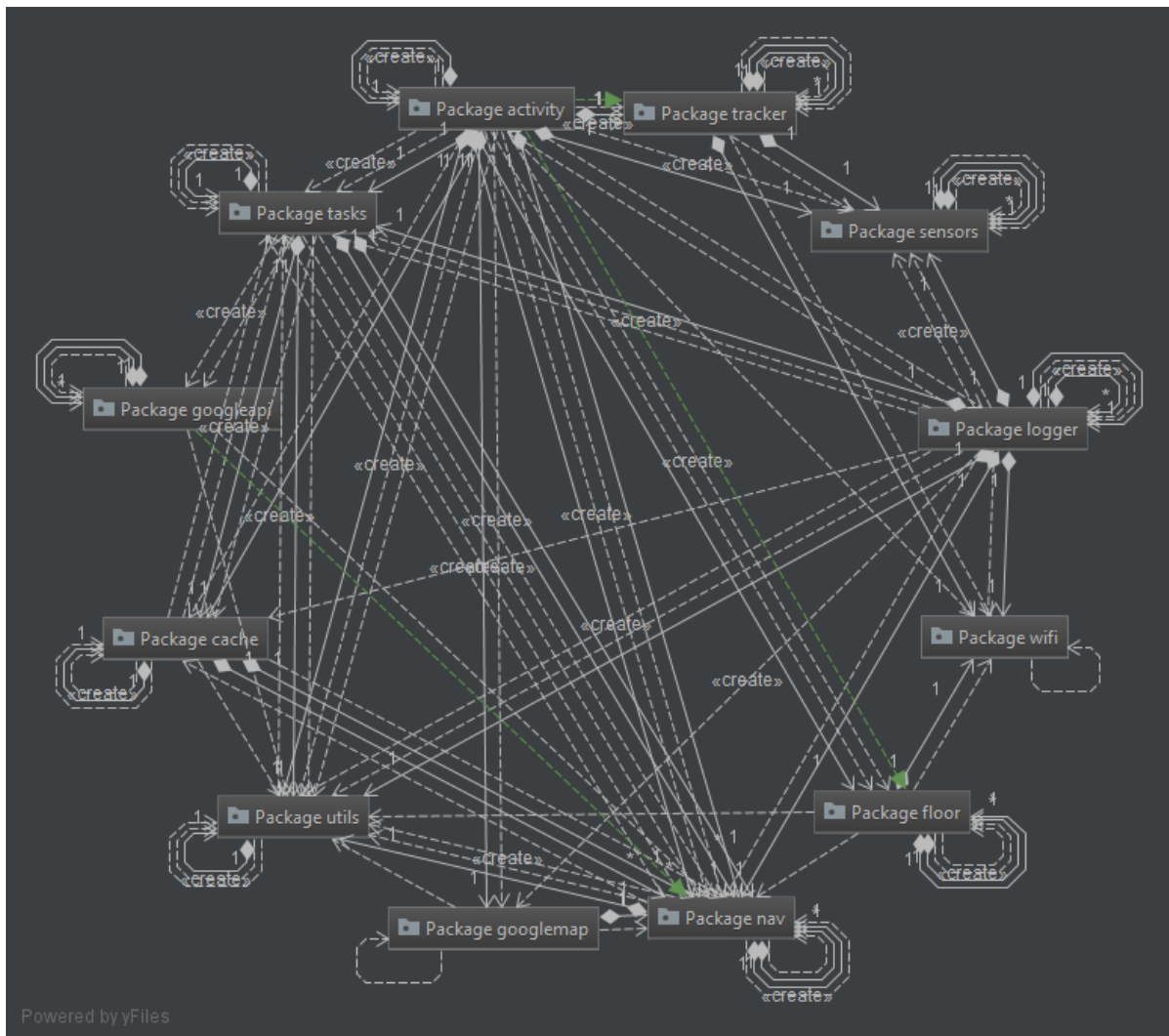
In this section is described in detail the Application structure. For the better understanding of the structure of this application, the analysis is driven from the big picture to the more detailed explanation with the help of diagrams. The basic application components are presented and their appropriate placement and use within the application architecture is determined. The section starts with the presentation of the packages which contain the design elements of the application, followed by the classes of the system and the logic behind them.

#### **3.3.1.1 Package Diagram**

Due to the quite large size of the Service, in the context of this project, only the parts of the application that were used, modified or implemented, in order to make the Anyplace application accessible, and the parts that are directly connected to them will be described. In this respect, a package diagram is presented below, while the packages that are directly related to the current project are being analysed further with the help of Impact analysis diagrams.

The package that was mainly used for the transformation of Anyplace Application to an application accessible to almost every user was the Anyplace package, which is the package that includes the main functionality of the Navigator part of the application. The anyplace package is composed of the following sub-packages:

- activity: activity contains all the classes that are related to application activities (screens) and their operations. This package is responsible for the starting procedure of the application and for the display of all the screens from which the application is composed.
- cache: This package holds the classes needed to store and provide any information related to the last visited places as buildings, floors and PoIs.

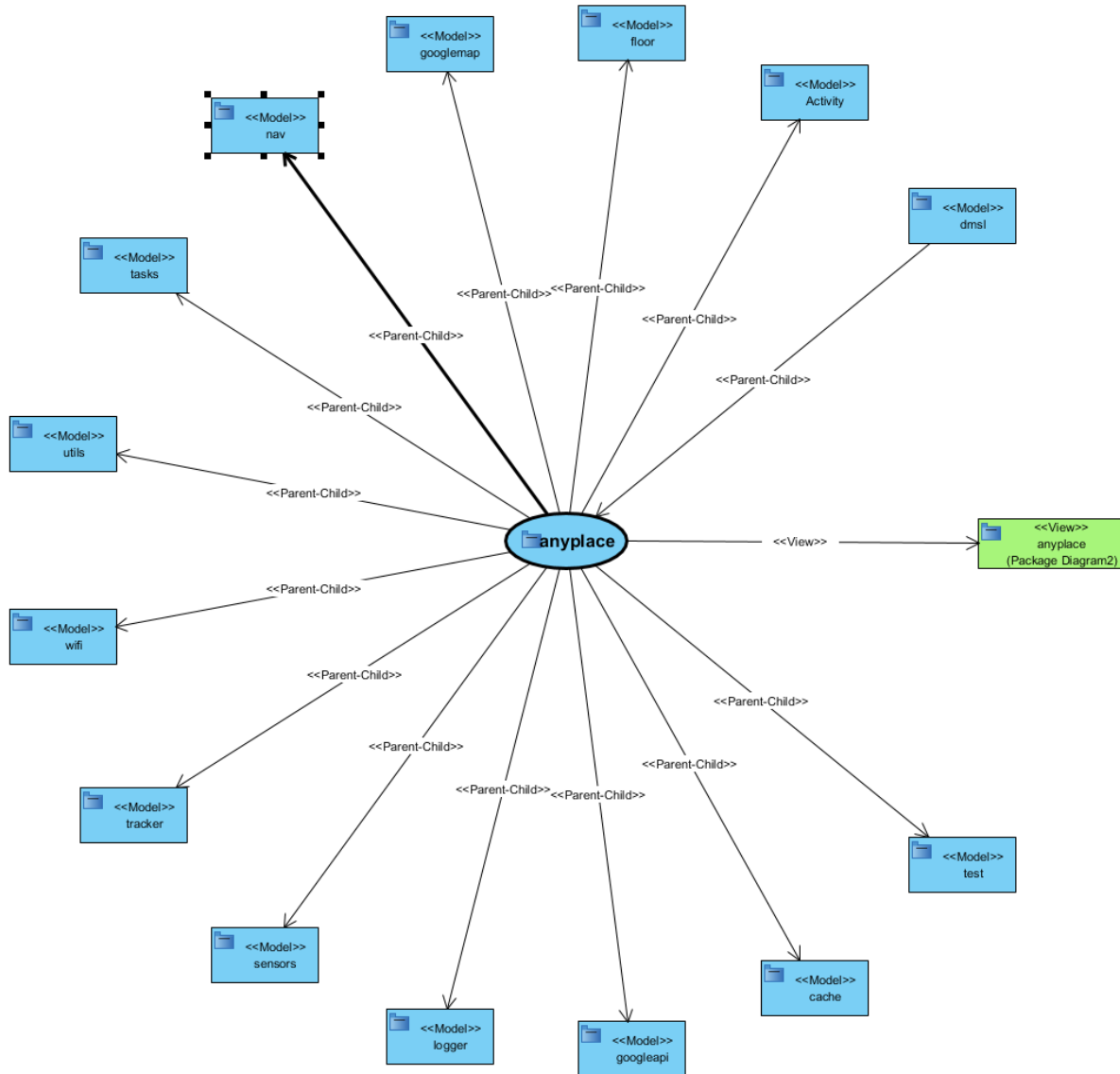


**Figure 8:** Package Diagram

- floor: The package floor contains anything related to the calculation of the current floor that the user is located. In case the user denotes the current floor, the Wifi-radiomap related to the floor is being fetched after a JSON request to the server is triggered from this package. In case that the floor is still unknown the classes that this package contain are responsible to handle the situation and make the current floor estimation.
- Googleapi: The classes contained in this package are responsible to handle any communication with Google API and are mainly connected to outdoors navigation tasks, as route estimation and nearby places retrieval.
- googlemap: concludes all the functionality that is directly related to Google Maps as the customizations needed in Google Maps tile providers and the handling of Google Maps markers and visible objects.
- logger: logger package holds the main functionality of the logger part of Anyplace application.
- nav: nav is the package that holds all the classes that are essential to assist the visual representation of the building, as the classes that determine the objects of the building, the floor and the PoIs models. In this package is also included the which handles class all the available navigation data, as every information that is related of what is currently displayed on the map and the last estimated user positions.
- sensors: Under this package can be found the classes that are responsible for the user's movement detection. The estimation of the current state of the user (e.g. standing/walking) and the handling of the phones pedometer is done with the help of the classes included in this package.
- tasks: All the classes that are related to execution tasks are categorised in this package. The tasks related to the download of available radio maps and the upload of the RSS log files, the fetching of the visual representation of the building and of every element related to it, the classes related to both indoor and outdoor navigation tasks can be found under this package.
- test: This package contains any file which is automatically generated and is related to the build process.
- tracker: The tracker is one of the packages that are directly connected to Anyplace Navigator as it contains the classes that are responsible for the detection of changes in WiFi Scan results and the algorithm that calculates the estimated position according to the provided radio map file.
- utils: utils could be mentioned that it contains the helper classes. By the term helper classes are meant the classes that have a secondary but essential \behaviour", as perform all the appropriate checks to confirm if all the modules

needed (e.g. sensors, memory) for the good use of the application, are enabled and available, or assist the execution of usual actions as file uploads/downloads.

- wifi: This package contains the wifimanager and wifiReceiver classes. As wifi-Manager is used to scan and receive scan results, this package is mainly connected with the logger part of the application.



**Figure 9:** Anyplace package Impact Analysis Diagram

### 3.3.1.2 Class Diagram

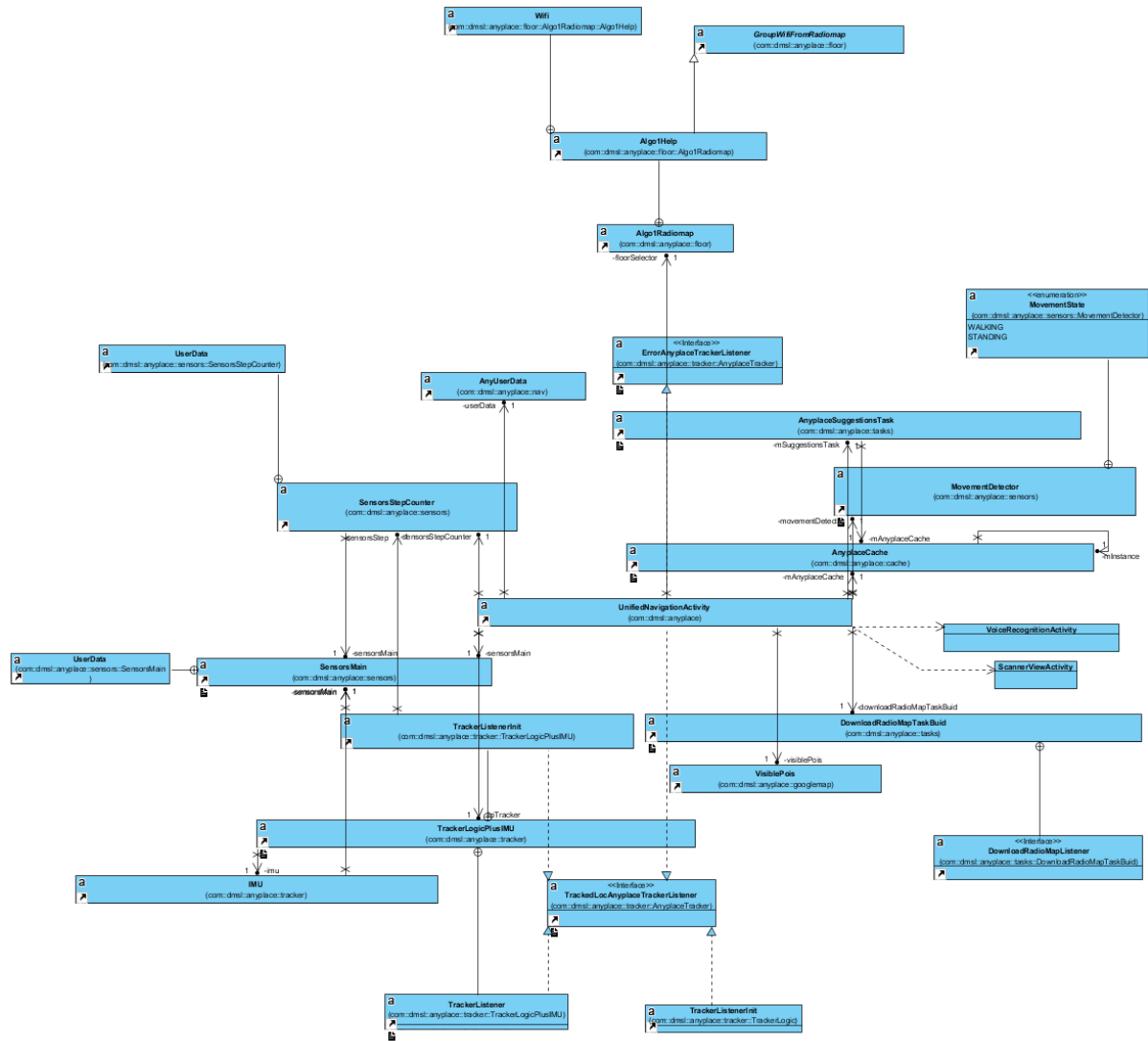
The following simplified diagram presents the most important classes of the application that are directly connected with the class "UnifiedNavigationActivity" which is both the class that handles the main application screen in the Navigator part of the application and the class that was mainly modified during the current project.

In the following Diagram (Figure 10) each dashed line with a triangular arrowhead symbolises a realisation between the classes. A realization is a specialized relationship between the classes, from which the one is representing the supplier (the class that is noted by the arrowhead) and the other represents the client.

Each class that has the suffix Activity is representing a screen of the application. On that diagram are included only six of the eight activity classes of this project, as for simplicity reasons the class that is responsible to display the launching screen (startActivity class) and the class related to the screen that is displaying the description of the project (AnyplaceAboutActivity), are not presenting here. It should also be noted that the VoiceRecognitionActivity class represents a totally transparent screen. The classes that contain the word "task", as the NavIndoortask and AnyplaceSuggestionsTask are asynchronous task classes, that are running for a short period in the background to assist different application's operations.

The Diagram in Figure 10 shows the relationships (dependencies) between the project classes. When there is such a connection between two activity classes, it means that the dependent class calls the independent (supplier) class. More specifically, in this case, if the user wants to call the voiceRecognitionActivity, he/she should first use the UnifiedNavigationActivity (main screen).

Something that could easily be noticed from the following simplified diagram is the advanced level of cooperation that the UnifiedNavigationActivity class displays compared to the other classes of the project. This fact could be characterised as normal, as this class is the main class of the project and is responsible for almost every call that should be done for all the operations related to navigation tasks. Nonetheless, this high occurrence of association between the main class with the rest classes of the project could be a potential problem from the aspect of Software Engineering as the number of associations of one class with other classes is directly proportional with the effort needed for the good maintenance and the expansion of the functionality of the class.



**Figure 10:** Simplified Class Diagram related to the connections of UnifiedNavigationActivity

### 3.3.1.3 Sequence Diagram

The sequence diagrams are representing the interactions between objects, in the sequential order that those interactions occur.

Thus, the use of sequence diagrams could make easier the understanding of complicated project procedures, as the procedure that is triggered when a PoI item is selected (Figure 11) or location information from a QR label are available (Figure 12). The diagrams were automatically generated with the selection of small call depth equals to two, in order to be easily readable and understandable.

From the Sequence Diagram in Figure 11, it is concluded that when the user selects the point of interest he wants, the procedure is triggered from the class `UnifiedNavigationActivity` and asks information from the `PoiModel` class regarding the names of the available PoIs, in order to check if the selected Point of Interest is on the floor that is currently selected or on another floor.

If the PoI is on another floor the system is trying to find the closest PoI of type Elevator that is near the user by asking information regarding the latest user position from the class `AnyUserData`. When the newly requested from the system PoI is known the method `startNavigationTask` is called with given parameter the id of the closest elevator point.

Every time that a new floor is detected, the same procedure is repeated, until the user reaches the floor of the selected PoI. When the user's smartphone is on the same floor as the Point of Interest requested, the method is handling the event as follows.

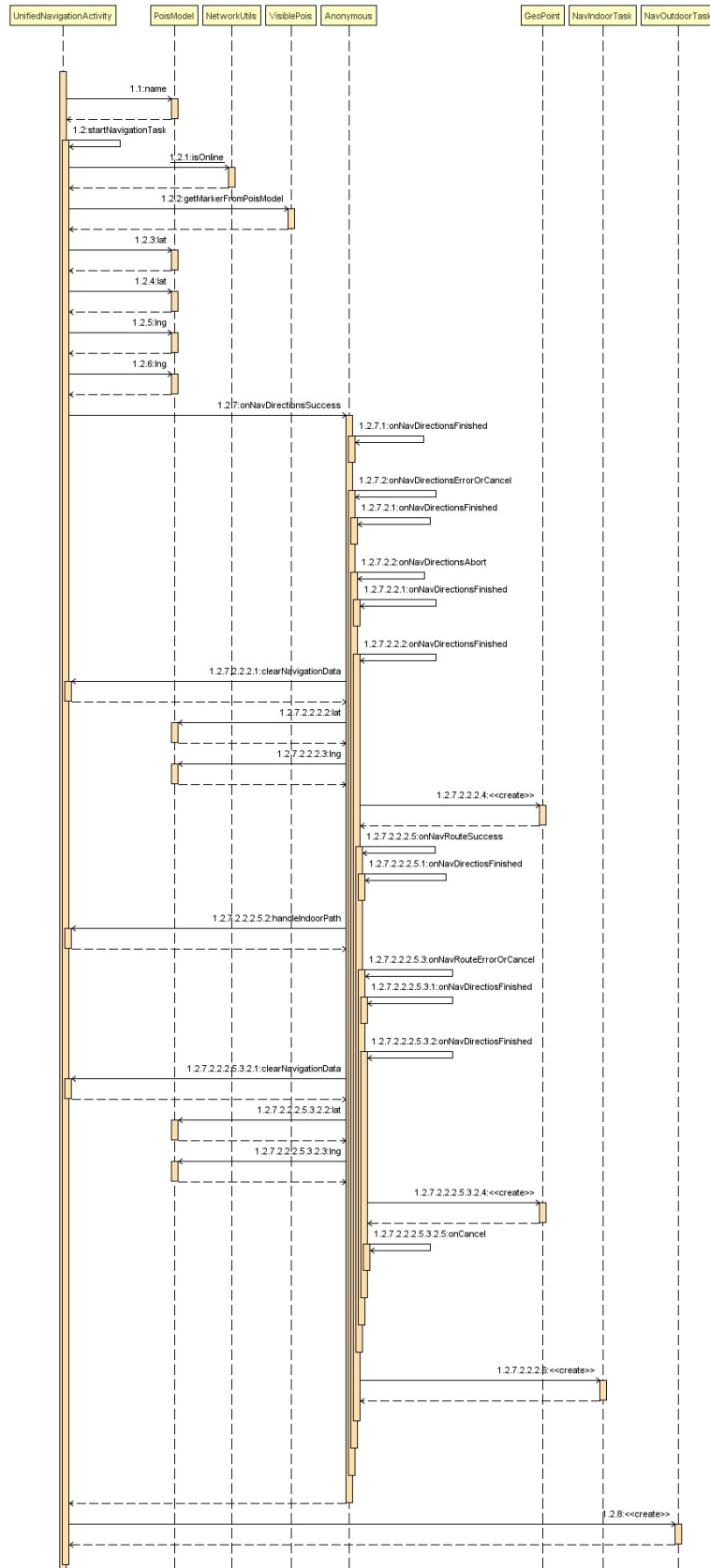
If the PoI requested is on the current floor, the `startNavigationTask` method is called with given parameter the id of the selected PoI.

The Sequence Diagram in Figure 12 describes the procedure that the system follows when the information of a Location QR label is available. The actual event is triggered from the class `ScannerViewActivity` when a label with the indication Any-place is detected, as the Section 3.3.1.5 describes in detail. However, in the following diagram is described the sequence of the actions that take place, after the `UnifiedNavigationActivity` class is informed, that the information on this specific QR label is available.

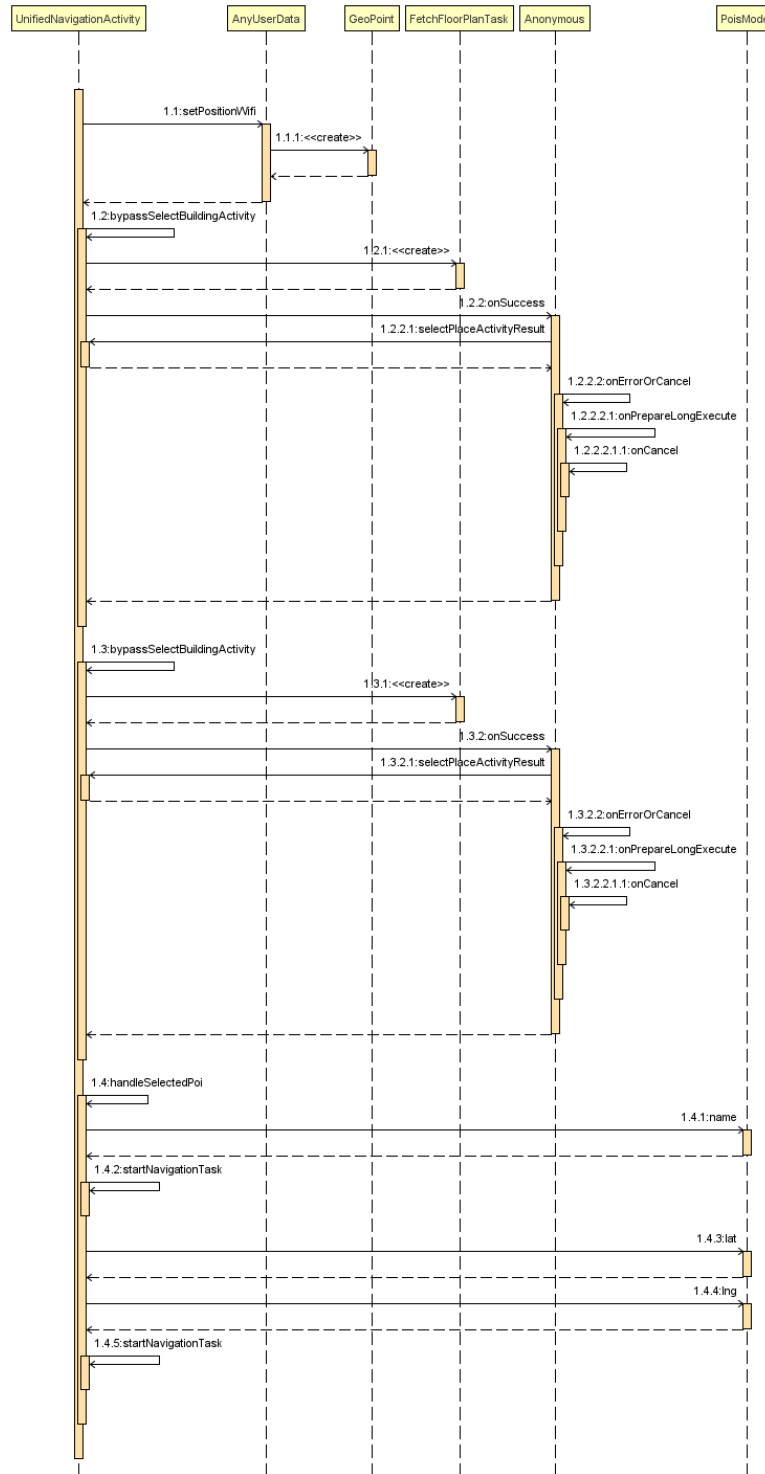
The retrieved information is coming as an input, in the shape of `String`, in the class `UnifiedNavigationActivity` and the method `qrLocationHandling` is called. This method is processing the `String` input in order to retrieve the available information regarding the user's position. When the retrieved information is available the method, informs the class `AnyUserData` about the new location of the user. The class `AnyUserData` is saving the new location with the help of the class `GeoPoint`. In case the retrieved location is on a different floor than the floor that is stored in the system as latest user location, the system loads the new floor plan, in which the QR label is located, with the help of the class `FetchFloorPlanTask`.

In case that a not completed navigation request was previously running, which means that the user is trying to reach a location, the method `qrLocationHandling`, informs the method `handleSelectedPoI` about the PoI which was just visited by the user.





**Figure 11:** Sequence Diagram handleSelectedPoi method



**Figure 12:** Sequence Diagram qrLocationHandling method

### 3.3.1.4 The AnyPlace Accessible User Interface Design

In order to maximise the application's usability and accessibility, much emphasis was given to the modifications that were made in the main Navigator's screen layout. The main design direction followed was that the layout should be easily understandable from the user's that have the impairments mentioned in subsection 2.3 and usable from people with motor disabilities while it will still be user friendly for people with no disabilities. Also, we decided that the existing Anyplace application layout shouldn't be modified but just extended since it is not creating any accessibility issues.

Based on this design direction, three additional buttons were created and they were placed on the three remaining sides of the smartphone screen, as the upper side of the screen was already occupied by the search field of Anyplace original application. In this way, the user can easier understand where every element on the screen is located, as shown in Figure 13 later in this section.

The three buttons created are the following:

- **VoiceButton:** This button can be used when the user wants to communicate with the application using voice commands and is located on the left side of the screen.
- **QRButton:** This button is useful when a user wants to scan a QR label, as when is pressed is launching the QR Scanner module of the application and is located on the bottom of the screen.
- **PolIsButton:** The PolIsButton is responsible for the handling of the list of the available Points of Interest list in a building, in the shape of a Spinner. Only when this button is pressed by the user the Spinner gets visible. This button is located on the right side of the screen.

It was decided that different colors will be used for every button. The combination of colors was selected based on the WCAG Guidelines<sup>2</sup> with the help of colorsafe.co<sup>3</sup> tool, which is available online, with the selection of WCAG Standard : AAA<sup>4</sup>. The logic behind this selection was not to use color indications in order to construct the application instructions based on them, as this wouldn't be compatible with the restrictions of color blindness, as they are mentioned in subsubsection 2.3.1 - Color Blindness, but to help the users that have the ability to make distinctions between colors, in the further use of the application. An explanatory example about that could be a user that confuses the colors red and green but he is still able to distinguish tone differences between the Voice and the QR buttons.

Regarding the background color palette of the buttons and the spinner, the different colors used was a personal selection following the guide of RGB, but the combination of

---

<sup>2</sup><https://webaim.org/blog/wcag-2-0-and-link-colors/>

<sup>3</sup><http://colorsafe.co/>

<sup>4</sup><https://www.w3.org/TR/WCAG20/#conformance-reqs>

the background with the font color, is a result of the tools mentioned in the paragraph above.

In the table 3 are available both the color selections for the application's main layout but also the results generated by the colorsafe.co tool. The rest parameters that were given on the tool but are not mentioned in the following table, were the same for every combination (Font Family : Helvetica , Font Size: 24px, Font weight: 700).

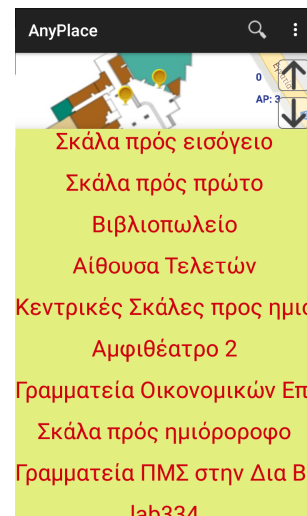
Right after the table, are available two screenshots of the application. The Figure 13 which is the layout of the application's main screen, as it was described above and the Figure 14 which is the visual result when the PoIs Button is pressed and the Spinner which is populated with the Points of Interest of the selected building, is becoming visible.

**Table 3:** Application's color selections

-	WCAG Standard	Background Color (hex)	Font Color (hex)	Goal Contrast Ratio	Contrast Ratio of Selected Combination
Voice Button	AAA	#3acab0	#005051	4.5	4.53
QR Button	AAA	#a71836	#00e0e0	4.5	4.51
PoIs Button	AAA	#2c578a	#E6cc22	4.5	4.59
Spinner	AAA	#e2ee7d	#cf000f	4.5	4.58



**Figure 13:** Main screen layout.



**Figure 14:** PoIs Button is pressed

### 3.3.1.5 QR Codes use

In this project the QR labels have the informative role that they usually have, however they are divided in two main categories.

- **Informative QR labels:** The system is able to scan and process every available QR label and inform the user about the content of the code displayed on the label with accessible messages. This messages could possibly provide the user with information related to an event taking place in a specific point of interest, inform him/her about the next available Points of Interest or give a message that wouldn't be available to a visually impaired user. Examples of the information that an informative QR label can hold are :
  - "4th International Conference on Applied Theory, Macro and Empirical Finance", outside of a conference room
  - "Auditorium 13" on the outer side of the Auditorium 13 doors, or "To Auditorium 13" on a floor tile, which is on the path to the auditorium.
  - "Caution, wet floor"
- **Location QR labels:** This type of QR labels hold a specific message structure, that can be further processed by the system. They provide accurate positioning information, better than that of WiFi positioning.

The structure of the message, in which every words in bold face are a field names (i.e. cannot change), since it should be the same for every location label. Field values, not in bold, hold specific information regarding a location, as shown below:

The QR label of this example is designed to be placed in the main entrance of University of Macedonia building. The content of this QR label is:

Anyplace :

Type: Location :

Place: Main Entrance:

Lat: 40.62530558761516 :

Lon: 22.96012517614372 :

floor: 0

When the user's smartphone camera detects this label, will inform both the user about his current location with the corresponding message in Greek language "You are in the main entrance" and the system in order to locate the user in these exact coordinates of the building.



**Figure 15:** Location QR label  
- UoM Main Entrance

### 3.3.1.6 Voice Recognition

As the system is based on the interaction of the user with the main application's layout, the execution of a wide range of available commands is much easier with the use of voice commands. This functionality allows the execution of some complex commands with just a screen tap to the voice button. So a visually impaired user, who is already familiar with the application or with the building, instead of exploring the screen to find the POIS button, click on it, listen the PoIs list, select the item he wants, can just use the voice button and say the name of the point of interest he wants to go to. The functionality is exactly the same for a motor-impaired person, as he/she can perform almost every action available in the UI by just performing a tap, on a big target of the screen as the VOICE button is.

The voice commands that can be recognised by the system are specific and can be easily extended or translated, by changing the corresponding code of the class `VoiceRecognitionActivity.kt`. The messages that the system supports in Greek language are the following:

- "QR": This message could remain the same for every language. When the user gives this command, the system is calling the `ScannerViewActivity.kt` and is ready to scan any available QR label.
- "Σημεία Ενδιαφέροντος" (Points of Interest): With this command, the system acts as if the user had clicked on the button POIS of the User Interface and an accessible list with the available Points of Interest in the building is automatically generated.
- "Όροφος πάνω" , "όροφος κάτω" ("floor up" , "floor down"): A click is "performed" on the corresponding UI button for floor change, respectively.
- When the system detects any other command than the aforementioned, assumes that the user is asking a specific PoI and the command is compared with the names of all the available PoIs in the building. If the system detects a PoI with the same name, the user is informed that the point was found and the navigation procedure from the user's location to this PoI is starting. If the system is unable to match the given command with one of the available PoIs is informing the user with an accessible message that "The point couldn't be found".

### **3.3.1.7 Navigation procedure**

The Anyplace original application informs the user about the path he/she should follow, by displaying a red polyline on the user screen, there was a need for some changes to the navigation procedure also.

In the Anyplace Accessible Application when a PoI is selected by the user and the polyline is constructed, the user is asked to slowly turn his/her smartphone around, much like performing a "radar scan". During this scanning process, the system detects if the smartphone is aligned with the next point of the polyline, with the help of the smartphone's compass. When the user's smartphone is facing the next spot on the recommended path, the smartphone is giving as output vibrations that are continuously repeated as long as the user is following the correct path. The system is informing the user with the appropriate accessible messages when he/she is close to the next intermediate or final point. When every intermediate point of the path is reached the user should follow again the same procedure. Moreover, when the user is on the selected destination the system informs the user with the appropriate accessible message.

We believe that these changes in the application interface, will allow users with disabilities to successfully use the application, and allow orientation inside the building.

## 4 Implementation

This chapter presents the implementation of Anyplace Accessible Application and the workflow followed with all the parts of Anyplace Service for the creation of a Case study for University of Macedonia (UoM).

### 4.1 Extension of Anyplace Application

In this section are presented the implementation techniques and the code written for the implementation of Anyplace Accessible Application, based on the Analysis and Design presented in section 3.

#### 4.1.1 Tools and Libraries used

As we selected to work with the Android Application of Anyplace Service but without restricting the code development in Java Language, and since the official support of Kotlin programming language in Android was announced in Google I/O 2017<sup>5</sup>, the implementation was done on Android Studio IDE, version 3.0.1, in which the Kotlin language is supported, using the JetBrains kotlin-android plugin. For the implementation of QR Scanner feature, the ZXing<sup>6</sup> open source Library was used, which is an image processing library implemented in Java.

#### 4.1.2 QR Scanner implementation

The QR scanner feature was implemented to serve both the need of locating the user with high accuracy (less than the accuracy of 1.9 meters that is already provided by Anyplace Service) and of providing additional information to the user if needed. As an additional feature, the ScannerViewActivity class was implemented entirely in Kotlin programming language.

When the QR button of the main application's screen is pressed, is calling the ScannerViewActivity.kt. As an activity class, the ScannerViewActivity represents one of the applications screens. This class makes use of ZXing Scanner Library to programmatically initialize the Scanner view of it and get input from Smartphone's camera, in order to use the input of the camera as a source of information. The class is also using the TextToSpeech android library (tts), in order to provide audio instructions to the users.

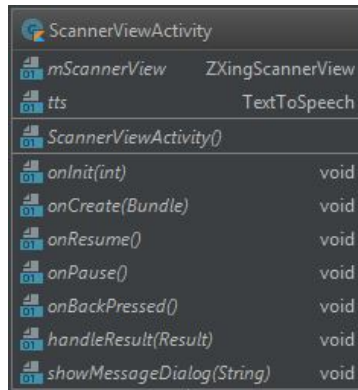
On the fourth and last part of Figure 16, the methods of the class are presented. The onInit() method was implemented only for testing reasons and checks if the text to speech messages are working. In case the audio feedback can be used, is providing instructions of QR scanner use to the user. The methods onCreate, onResume, onPause, onBackPressed are the methods that are handling the main behaviour of the Activity, as the onCreate() is used to start the Activity, the onPause() defines what will happened when the activity is interrupted, the onBackPressed handles a tap on the button that is used to every phone to return in the exact previous state and the onResume() determines the behaviour of the activity after a Pause period. The method

---

<sup>5</sup><https://www.androidcentral.com/google-io-2017-developers-share-their-highlights>

<sup>6</sup><https://github.com/dm77/barcodescanner/tree/master/zxing>





**Figure 16:** ScannerViewActivity Class Diagram

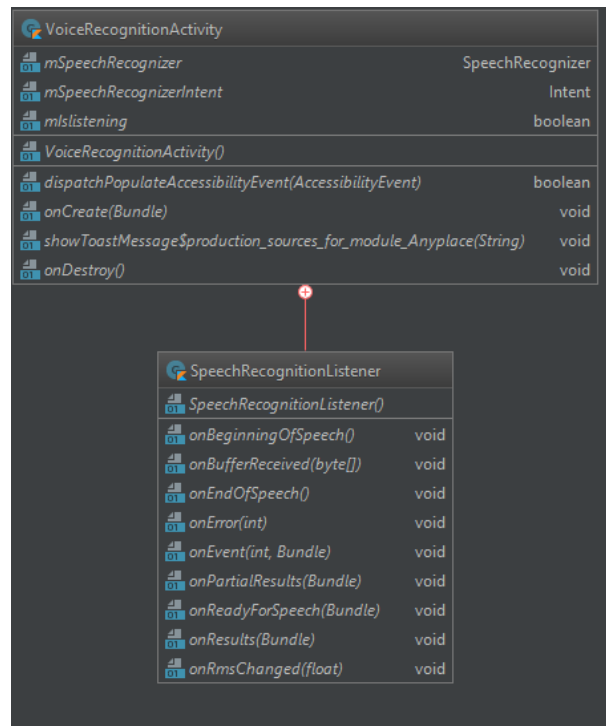
handleResult is getting as parameter the input of the camera and is responsible for every event that will be generated after the result of the scan, as the behaviour of the application and the processing of the result. The ShowMessageDialog method, has an assistant role as it gets a String parameter as input and displays a message dialog with the given message whenever is called.

#### 4.1.3 Voice Recognition Activity implementation

As the application has to provide multiple ways for receiving user input, it was implemented the VoiceRecognition activity, which is the responsible activity for the handling of Voice commands. The voice recognition feature was implemented in Kotlin programming language.

This activity is triggered by the press of the voice button on the main screen. The main difference of this activity is that even though it represents an application's screen, this screen is a transparent one, so from the user's point of view, there is no change in the layout. This activity is using the SpeechRecognizer library class, which is natively provided by Android, that allows access to the Speech Recognizer. It also uses the speech recognizer intent which is the link between the activity and the library as it displays the (transparent) microphone dialog box which receives the speech input.

In the main methods of this activity are included the onCreate method, which as mentioned in the previous section is responsible to start the activity and the onDestroy(), which is responsible to kill any running threads after the completion of the voice recognition procedure.



**Figure 17:** VoiceRecognition-Activity Class Diagram

The `dispatchPopulateAccessibilityEvent` method is also used to prevent the creation of any possible accessibility events as this is a full voicing part of the application.

In the diagram above a dependency with the Inner class, `SpeechRecognitionListener` is also displayed, which is responsible to handle any event related to speech recognition, as it holds the methods that are responsible to recognise when a speech command is starting and when it is completed. Furthermore, it holds the most important method of the class, which is the `onResults` method. The aforementioned method is responsible both for the handling of the speech recognition results, by matching the speech input with pre-defined data and for the communication of the processing results to the main application's activity, the `UnifiedNavigationActivity`.

#### 4.1.4 Voice Feedback implementation

The implementation of the voice feedback can be done in a few different ways, but having in mind that the main target is an application that is the same time, usable for different user groups disregarding any possible disabilities they may have. The approach followed was the use of voice feedback with the use of Text to Speech messages, only in the cases that the message is useful for all the user groups. This approach could be a possible solution for any output of the application but in this way, the sighted users would receive messages that wouldn't add any extra value to them.

For that reason, the approach followed for the audio feedback was slightly different. Currently the only possible way for the visually impaired users to use an Android smartphone is the use of a mobile screen reader. The most commonly used screen reader<sup>7</sup> in Android devices is the native Android screen reader named TalkBack.

The TalkBack application is running in parallel with any other running application, and is providing audio feedback about what is displayed on the screen. Hence, except following common accessibility design methods, as providing alternative text to any visible element in a meaningful way, we decided to use pop-up messages to provide feedback that is generated on the fly. In Android software, this implementation is feasible with the use of Toast messages. The Toast messages are providing information regarding application's operations in small pop-ups, while the current activity remains visible and interactive. The main modification done (the code example of which is available in Figure 18 below), was to keep the messages transparent in order to avoid feeling a part of the application's main screen with messages that are not useful for the sighted users, but are still readable from the Talkback application.

```
Toast toast = Toast.makeText(getApplicationContext(), text: "wifipos "+userData.getPositionWifi(), Toast.LENGTH_LONG);
View view = toast.getView();
TextView text = (TextView) view.findViewById(android.R.id.message);
text.setShadowLayer( radius: 0, dx: 0, dy: 0, Color.TRANSPARENT);
text.setTextColor( Color.TRANSPARENT);
view.setBackgroundColor( Color.TRANSPARENT);
toast.show();
```

**Figure 18:** Transparent Toast message implementation

---

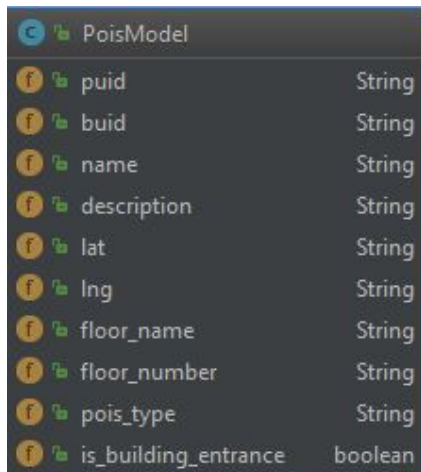
<sup>7</sup><https://webaim.org/projects/screenreadersurvey6/>

#### 4.1.5 Modifications in Navigation Procedure

The main modifications done in the Navigation procedure are directly connected to `UnifiedNavigationActivity` class which is also the responsible class for the handling of the navigation process. Any modifications done in this class are written in Java programming language. The main changes in the class are related to the creation of the Spinner that is being populated with all the Points of Interest available in the Building and, to the way that the navigation procedure is done.

The `PoIsSpinner` is accessible through the `PoIs` button and is mentioned in this section because through this spinner the user can be informed about all the available `PoIs` in the building and not only about the `PoIs` in the selected floor.

When a building is selected by the user, or is detected by the Anyplace service, the spinner is constructing an object of the `AnyplaceCache` class to ask information about the available `PoIs` and is storing the reply in a List that can be populated only with elements of the type `PoIsModel`. The information that an element of the type `PoIsModel` holds is presented in Figure 19.



PoIsModel	
puid	String
buid	String
name	String
description	String
lat	String
lng	String
floor_name	String
floor_number	String
pois_type	String
is_building_entrance	boolean

**Figure 19:** `PoIsModel` fields

After all the necessary information is available in the List mentioned above, a new list of type `String`, in which is stored the name of every `PoIsElement` in the building, is created. Subsequently, the spinner is populated with the `Strings` that are available in the second List.

When the user selects one of the spinner elements, the spinner becomes again transparent and a method called `handleSelectedPoi` is called having as parameters the first List with the `PoIsModel` elements and the `String` that was selected by the user.

The method `handleSelectedPoi`, given the input parameters is matching the input `String` with the `PoIsModel` elements provided in the input list, after the suitable `PoI` is found the method `startNavigationTask` is called. The way that the `handleSelectedPoi` method is working and the related sequence diagram are available in the paragraph 3.3.1.3.

When a navigation route should be drawn for a floor, the method `startNavigationTask` asks information about the selected floor number and the latest user position. When all the required information is available, and all the necessary conditions for the construction of the navigation line are met, the method which is responsible to handle the indoor path is called.

The `handleIndoorpath` method is responsible to draw the navigation route for the selected floor. This method needs as an input parameter, a list with the points of the navigation route, to design the navigation line between the user and the selected point of Interest as a `Polyline`<sup>8</sup>.

<sup>8</sup><https://developers.google.com/android/reference/com/google/android/gms/maps/model/Polyline>

A red line could be a satisfactory solution for sighted users or users with a low level of visual disabilities however, has no meaning for people in higher levels of visual disabilities, the functionalities of some methods should be extended in order for the application to be able to serve more user groups.

On that point, after the construction of an internal navigation path, every time that a change in user's state is detected from the method `onLocationChanged`, another method of the name `followUserPath` is called.

This aforementioned method, as soon as the user is following the provided path, is dividing the path in separate lines, between the different points of interest. After the user is asked to slowly turn his/her smartphone around, the system is calculating if the user has the correct orientation by executing calculations regarding the user's smartphone position and orientation, and the next point of interest that has to be reached.

Firstly, the method is using the method `bearingTo`<sup>9</sup> of Android Location Class in the shape of:

**Bearing to Location = Last known location.bearingTo(target location)**

in order to calculate the angle between the two points in degrees East of true North. Secondly, is reading from the sensors of the device the current unfiltered heading in degrees, which is the angle between North and the direction that the user's device is pointing.

As a third step, the algorithm of the method identifies if the difference between the mentioned calculations and readings. When the difference between the two different amounts is close to zero, the application is calling the Android Vibrator service to inform the user that is in the right direction, as it is displayed in Figure 20.

```
if (((sensorsMain.getRAWHeading() - bearingToLoc) > -7) && ((sensorsMain.getRAWHeading() - bearingToLoc) < 7)) {  
    Vibrator v = (Vibrator) getSystemService(Context.VIBRATOR_SERVICE);  
    v.vibrate( milliseconds: 500);  
}
```

**Figure 20:** Difference identification between bearing and heading

As soon as the user is in the right direction, the smartphone keeps vibrating as a sign that the user is in the correct path. When the user is close to the next intermediate PoI, the application is providing audio feedback related to the distance from the next PoI. When the next intermediate PoI is reached, the user is informed with an audio message and in case, the current position of the user, is not the final destination, the same procedure is starting again.

---

<sup>9</sup>[https://developer.android.com/reference/android/location/Location#bearingTo\(android.location.Location\)](https://developer.android.com/reference/android/location/Location#bearingTo(android.location.Location))

#### 4.1.6 GUI implementation

This section presents the procedure followed for the implementation of the user interface (UI), as it is presented in paragraph 3.3.1.4.

The User Interface layouts and their elements in Android are defined in XML files that are located in /res/layout directory of the Android project. For the modifications done in application's main screen layout, the pre-existing file of the UnifiedActivity-Navigation xml (activity\_unifiednav.xml) file was modified and some other that will be presented in detail below were created.

The Figure 21 shows an extract of the xml file related to main screen layout. Specifically, this is the xml code related to the three buttons described on the paragraph 3.3.1.4. As it can be seen, the three elements are declared as Buttons. Every button carries its own attributes as, a specific width, height, alignment, colors, text style and text size. The label text defines the label of the button that will be displayed on the screen and the label contentDescription determines what the Screen Reader application will read when the button is reached. Every String of the descriptions is declared in the file strings.xml, that is available under the /res/values directory of the project, in order for the code to be more clear and easier extensible in case a translation in another language is needed.

On this step, should be underlined that the Screen Reader application is responsible to inform the user that the screen element is a clickable button, on that point the provided label shouldn't repeat the same information, as it can be seen in Figure 22 below.

```
<Button
    android:id="@+id/qrbutton"
    android:layout_width="200dp"
    android:layout_height="wrap_content"
    android:layout_alignTop="@+id/btnTrackme"
    android:layout_centerHorizontal="true"
    android:background="#a71836"
    android:contentDescription="@string/qrScannerdesc"
    android:text="QR"
    android:textColor="#00e0e0"
    android:textSize="24sp"
    android:textStyle="bold"/>

<Button
    android:id="@+id/voicebutton"
    android:layout_width="50sp"
    android:layout_height="100sp"
    android:layout_centerVertical="true"
    android:layout_marginLeft="16dp"
    android:layout_marginStart="16dp"
    android:background="#3acab0"
    android:contentDescription="@string/VoiceButtondesc"
    android:text="Voice"
    android:textColor="#005051"
    android:textSize="24sp"
    android:textStyle="bold" />

<Button
    android:id="@+id/PoIsbutton"
    android:layout_width="50sp"
    android:layout_height="100sp"
    android:layout_centerVertical="true"
    android:layout_toEndOf="@+id/detectedAPs"
    android:layout_toRightOf="@+id/detectedAPs"
    android:background="#2c578a"
    android:contentDescription="@string/PoIsButtondesc"
    android:text="PoIs"
    android:textAlignment="center"
    android:textColor="#E6cc22"
    android:textSize="24sp"
    android:textStyle="bold"/>
```

**Figure 21:** Extract of Button elements



```

<string name="qrScannerdesc">Σάρωση Κωδικού QR</string>
<string name="PoIsButtondesc">Λίστα σημείων εντός κτηρίου</string>
<string name="VoiceButtondesc">Καταχώρηση Φωνητικής Εντολής</string>

```

**Figure 22:** Example of Button's description label

The rest modifications done in the layout are related to the Spinner that host the PoIs of the building, which is invisible, so except of defining the existence of it in the xml file, extra layout files were needed in order to achieve the desirable behaviour. As is shown in Figure 23, the Spinner element is declared in the main xml file as invisible.

```

<Spinner
    android:id="@+id/poisSpinner"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_alignLeft="@+id/btnTrackme"
    android:layout_alignStart="@+id/btnTrackme"
    android:layout_below="@+id/btnFloorDown"
    android:focusable="true"
    android:prompt="@string/poisSpinner"
    android:visibility="invisible" />

```

**Figure 23:** XML Declaration of invisible Spinner

The visible representation of the Spinner item is being created on the fly, every time that the user is pressing the PoIs Button, with the help of View class as it is presented in Figure 24 below. The custom\_spinner xml file, is mainly providing information regarding the alignment and the text color of Spinner Items and the transparent\_spinner file, is responsible to keep the items of the Spinner transparent after an item is selected.

```

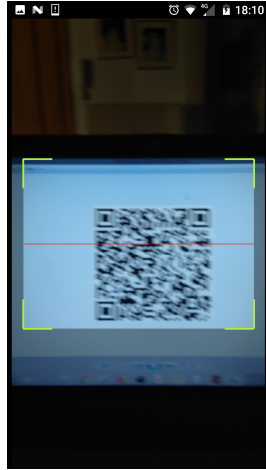
/** Defining the ArrayAdapter to set items to Spinner Widget */
ArrayAdapter<String> adapter = new ArrayAdapter<->( context: this, R.layout.transparent_spinner, ianyplace){

    @Override
    public View getDropDownView(int position, View convertView, ViewGroup parent)
    {
        View v = null;
        v = super.getDropDownView(position, convertView, null, parent);
        v.setBackgroundColor(rgb( red: 226, green: 238, blue: 125));
        v.setPadding( left: 0, top: 0, right: 0, bottom: 3);
        v.setMinimumHeight(100);
        return v;
    }
};

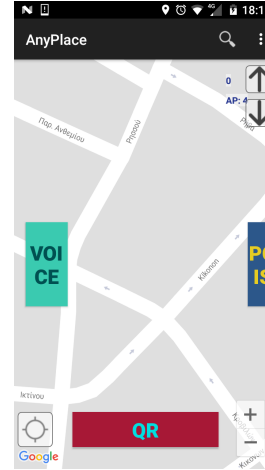
```

**Figure 24:** PoIsSpinner Creation

Regarding the layout of the GUI implementation of the QR Scanner Activity and the Voice Recognition, it should just be noted that the layout files were implemented just to determine the behaviour of Smartphones Screen, as the Voice Recognition Activity is a fully transparent activity (Figure 26) and the QR Scanner Activity, is programmatically initializing the ZXingScanner view as it is provided from the library used (Figure 25).



**Figure 25:** Layout after QR Button is pressed



**Figure 26:** Layout after VOICE Button is pressed

#### 4.1.7 Use of Layout Objects in Activities

This Section presents how the layout objects are used by activities. As the most complicated layout element from the aspect of implementation in this project is the PoI Spinner, this item was chosen to be analysed in this section.

The Spinner creation is being triggered during the initialization phase and is being achieved by the help of the corresponding method `poisSpinnerCreation()`.

As was already shown in Figure 24, an `ArrayAdapter` is used to populate the Spinner Widget with Array elements. Afterwards, the layout object of the Spinner is loaded into the variable `poisSpinner` and the adapter is set. Subsequently, is following the handling of the `PoisButton` layout object, which is loaded into a local variable and the `setOnClickListener` method of the button is used, in order to add a click Listener to the button and programmatically perform a click to the Spinner to make it visible, as is presented in Figure 27.

```
final Spinner poisSpinner = (Spinner) findViewById(R.id.poisSpinner);
poisSpinner.setAdapter(adapter);
adapter.setDropDownViewResource(R.layout.custom_spinner);

/** Adding radio buttons for the spinner items */
Button poisButton = (Button) findViewById(R.id.Poisbutton);
poisButton.setOnClickListener((view) -> {
    poisSpinner.performClick();
    poisSpinner.sendAccessibilityEvent(AccessibilityEvent.TYPE_GESTURE_DETECTION_START);
    poisSpinner.onInitializeAccessibilityEvent(talkBackEvent);
    poisSpinner.setVisibility(View.VISIBLE);
});
```

**Figure 27:** PoIs Spinner and Button loading and handling of Click event.

#### 4.1.8 Communication between Activities

In the previous Sections was described how the implementation of the Activities and the Graphical User Interface was done, in this section is presented how all the implementations mentioned above are being combined and run together as one application.

The main Activity has to communicate with other Activities as the Voice Recognition Activity in order to receive the result of the voice commands that was given by the user and with the QR Scanner to receive any new information that can be provided from a QR label.

This behaviour is achieved with the use of Intents. *"Intents are objects that provide runtime binding between separate components, such as two activities"*<sup>10</sup>

```
voicebutton = (Button) findViewById(R.id.voicebutton);
voicebutton.setOnClickListener((v) -> {
    startActivity(new Intent( packageContext UnifiedNavigationActivity.this, VoiceRecognitionActivity.class));
});
```

**Figure 28:** Reaching Voice Recognition Activity with the use of Intent

As it was presented in Figure 28 above, a click on the Voice Button is triggering the `StartActivity` method, which calls the `VoiceRecognitionActivity`. As a first parameter of the Intent is given the Context which is the current Activity class and as a second parameter the Activity that should be be started.

The Intents can also curry information with them, with the help of the `putExtra()` method as is presented in Figure 29.

```
if (matches.contains("σημεία ενδιαφέροντος")){
    val testString = "poiSpinner"
    val intent = Intent( packageContext this@VoiceRecognitionActivity, UnifiedNavigationActivity::class.java)
    intent.putExtra( name: "methodName",testString)
    startActivity(intent)
}
```

**Figure 29:** Start UnifiedNavigationActivity through Intent with `putExtra()` method.

When a user is giving the voice command ("Σημεία Ενδιαφέροντος"), which is the corresponding command for the call of the PoIs Spinner (as it means Points of Interest in Greek language), an Intent is starting having as target to trigger the `UnifiedNavigationActivity` and pass to it, the message of the Command which is carried in the string `testString` and has the label `methodName`.

When a new Intent is received from the `unifiedNavigationActivity` class the method `onNewIntent` is triggered and calls the method `handleIntent()` with the received Intent as input of the method. Thereafter the `handleIntent` method extracts the message that the Intent is carrying as it shown in the second line of Figure 30. As a result of the nested if statement, if the extracted String equals to `poiSpinner`, a click in the `poiButton` is programmatically performed, the spinner becomes visible and the user has access to the Points of Interest list.

<sup>10</sup>(<https://developer.android.com/training/basics/firstapp/starting-activity>)



```
private void handleIntent(Intent intent) {
    String extras = intent.getExtras().getString( key: "methodName");
    String action = intent.getAction();

    //If extras is not null that means that the intent has string extras, so in this case is a command
    if (extras != null && !extras.isEmpty()){
        if (extras.equals("poisSpinner")) {
            findViewById(R.id.Poisbutton).performClick();
        }
    }
}
```

**Figure 30:** Handling of received Intent

#### 4.1.9 Minor Modifications

At the last part of this Chapter minor modifications done that are not directly connected to the actual target of the project will be explained. The first part is the modifications done in the Manifest file of the project, in which all the project activities and permissions should be declared. In that respect the two new activities, are declared in the manifest file (Figure 31) and three more permissions are asked, in order for the application to be able to use the smartphone's camera for the scan of the QR labels, the vibrator for providing feedback and to record audio to process the user's voice commands (Figure 32). Also, the API\_KEY for google services should be changed.

```
<activity
    android:name="com.dmsl.anyplace.activity.ScannerViewActivity"
    android:screenOrientation="portrait">
</activity>
<activity
    android:name="com.dmsl.anyplace.activity.VoiceRecognitionActivity"
    android:theme="@style/Theme.AppCompat.Transparent.NoActionBar">
</activity>
```

**Figure 31:** Declaration of new Activities in Manifest file

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.VIBRATE"/>
```

**Figure 32:** New permissions asked in AndroidManifest.xml

The Second and last part of this section is related to an issue that was encountered in the early stages of the implementation. As we decided to use the Kotlin programming language in Android Studio, it was selected as preferred IDE the Android Studio of Version > 3.0, in which the Kotlin programming Language is fully supported without the need of external plugin use.

This selection was causing a conflict with the support Library ActionBarSherlock that Anyplace application is using, so it was decided to change it with the ActionBar-Compat of the support library of Google, while making sure, that this wouldn't have any impact to the final result.

## 5 Testing

In this section the evaluation and the testing of the application are presented. In the first part of the chapter, the application is examined against the Functional Requirements and the non-Functional Requirements respectively. In the last part of the section are summarized the results of the testing that has taken place in the building of the University of Macedonia.

### 5.1 Functional Requirements Testing

In order to examine if the application meets the functional requirements presented in subsubsection 3.1.1, we tested the usability of the application when the three most common assistive applications of Android smartphones are enabled and then we examined the satisfaction of the requirements in detail.

#### 5.1.1 Application usage with Assistive Applications

The most common used accessibility features of Android, is the native Android application TalkBack in combination with the Explore by touch feature. This application is using audible and vibration feedback to let the user know what is displayed on the screen and which element is touched. When an executable element is touched by the user, the application allows the user to open or execute it by double tapping on the smartphone's screen.

TalkBalk is most commonly used from people with visual, motor and learning disabilities. The most common types of visual impairments are explained in detail in subsection 2.3, in the terms of Smartphone's accessibility the motor disabilities are concentrated mostly to people that have difficulties to use their hands and, in the terms of this project as people with learning disabilities are defined the people who can't read.

As the most common gestures that can be used in the development of an Android application are consumed by the TalkBack application, developing having this feature in mind, could be a challenge as there are no clear guidelines at the time of how this could be achieved. Having this in mind, the only way to examine the usability of an application against this feature is user testing. On this respect, the testing phase was done by the help of two user stories.

The user scenario that we used for the first example, is the case of the hypothetical user Eve. Eve is a person with low-vision that is using the Anyplace Accessible application for first time but she is already informed about the use of the application as she has already listened the instructions of use with TalkBalk feature enabled, her current position is the entrance of University of Macedonia and she wants to go in the Auditorium 13.

- When Eve opens the application is listening just the name of the it, as a TalkBack user, she is starting exploring the screen by moving her finger on it.
- She is informed from TalkBack for all the elements touched and how they can be used.

- When Eve touches the button VOICE button is hearing the greek corresponding message "Insertion of Voice Command button, double tap for activation"
- Eve is double tapping on the screen and says "13"
- The System is informing the user that the place asked couldn't be found
- At that point Eve has two choices, one of trying again to guess the name of the place needed or to use the PoIs button which includes a list of the spots in the building as she was informed earlier during the screen exploration procedure.
- Eve follows the second option, so she is exploring the screen again until she hears the message "List of points in the building, button, double tap for activation" and she is executing the action
- Eve hears that what is displayed on the screen is a list with x items
- Eve is starting navigating in the list that is displayed on the screen until she listens "selected, Auditorium 13, double tap for activation" and she is executing the double tap action
- Eve is now informed that the Point of Interest she asked is on the first floor while she is on the ground floor, she is also informed that the system will now guide her to the closest elevator and that he should turn around till the device she is currying will vibrate.
- Eve is successfully executing the first instructions and is following the rest of the instructions generated on the fly in order to reach the elevator.
- When Eve is on the first floor is scanning with the help of the QR Scanner button the QR labels that are located on the building's floor, the system is now informed about the floor change and is constructing the new navigation line that is driving to Auditorium 13.
- When Eve is outside the Auditorium, she can use again the QR Scanner to ensure that she is in the right place with the help of the QR label which is on the door.

The user scenario that we used for the second example, is the case of the hypothetical user Bob who is going in Ceremonies Auditorium of UoM building. Bob has motor disabilities as he has difficulties of hands stability. Bob is able to see the screen and follow visual instructions but he has difficulties on selecting small elements of the screen.

- Bob has the choices of pressing the VOICE button and say the name of the point of Interest he wants to go or select it via PoIs menu.
- Bob is not distracted from other messages as the TalkBack feature is muted but he can still receive the vibration feedback that confirms every tap he is executing.

- After the selection of Bob the navigation line is created, so he can follow the available path.

The Screen Magnifier is another native Android feature that people with low vision are using. The feature can be enabled directly from Android accessibility setting menu and it can be generated when the user executes a fast triple tap on the screen. The good behaviour of the UI elements was tested with the screen magnifier enabled and every element of the application was still functional.

### 5.1.2 Evaluation against the functional requirements

The Table 4 below is a modified version of the Table 2 available in subsubsection 3.1.1. In the first Column presents the Requirements presented in the Table 2 and in the second column is presented, if and in which way these requirements are satisfied.

**Table 4:** Functional Requirements Evaluation - Part 1

Description	Evaluation
Any user should be able to use the mobile application, without authentication.	The application is usable from people with visual impairments, motor disabilities but still user friendly to people with no physical disabilities to a Smartphone's usage
The layout of the system have to be easily understandable by all the target user groups.	All the components of the main mobile application's layout are usable for every user group, as all the elements can be reached easily, even with the use of assistive application.
The system should provide accessible messages.	Users must be able to understand any message disregarding their disabilities.
The system should identify the location of the user.	The system can successfully recognise the location of the user, with the use of both of the choices that the original AnyPlace application offers (e.g., location extracted from the GPS module, the smartphone's IP address or the WiFi access points) and the available QR codes. This functionality was implemented during the Accessible Anyplace application project in order for the system to locate the user with high accuracy and be informed the location found.
The system must provide a list with the available Points of Interest.	A list of the available Points of Interest in the building is automatically generated when the user is in an indoor environment, and is available with the use of the button POIS.

**Table 5:** Functional Requirements Evaluation - Part 2

The system must provide alternative ways for user input.	The system can receive input through: <ul style="list-style-type: none"><li>• Voice Commands</li><li>• Haptic Selections</li><li>• Written text</li></ul>
The system must provide alternative ways for user output.	The system can provide output through: <ul style="list-style-type: none"><li>• Audio Feedback</li><li>• Visual Feedback</li><li>• Vibrations</li></ul>
The functionality of the system should be clear and easily accessible.	The Instructions of use are available with the greek voice command of the word "instructions" or via the general menu available on the upper right corner of smartphone's screen.
The system have to be able to use multiple ways to locate the user.	The System can find the user's location via: <ul style="list-style-type: none"><li>• The smartphone's IP address</li><li>• The smartphone's GPS module</li><li>• WiFi available signals</li><li>• Processing of QR labels</li></ul>
The crowdsourcing component of the system have to be easily accessible to sighted users.	As the basic User Interface is still user friendly to non visually impaired , every sighted user can easily contribute data without the need of sign in to the system.

## 5.2 Non- Functional Requirements Testing

In this section, is evaluated if the non-functional requirements presented in subsubsection 3.1.2 are fulfilled.

- **Financial Cost for the end user:** The downloading and the use of the application is free, on that point it could be said that there is no cost for the user. However, it should be noted that the output of the application is directly connected with the quality of the smartphone's built-in sensors. On that point is mentioned, that although the application is available for use in any Android Smartphone, it could provide more accurate results to smartphones with a higher quality of hardware.
- **Cost of Infrastructure:** The infrastructure that a building should provide for the good use of the application is related to the available WiFi sources per floor and the QR labels. On the terms of this project we are not calculating the cost required for the WiFi infrastructure, provided that it is common nowadays the majority of the big buildings to offer numerous WiFi access points. Regarding the QR labels the cost could vary as it is directly connected with the quality of the printed material, but after a very limited financial study, we could assume that would be less than 1 euro per square metre.
- **Convenience:**
  - The device that is necessary for the use of the application and should be carried by the user is an Android Smartphone or Tablet, on that point we assume that the user is able to carry a smartphone device on one hand and handle any other assistance may used, as a long cane or a guide dog with the other.
  - The training time required for the use of the application is the time that the user needs in order to listen or read the instructions of use and possibly some more minutes to explore the User Interface and understand the functionality of the application.
  - The application offers three different ways to receive and provide feedback from and to the user respectively, as it is presented in the Table 5 above.
- **Compatibility with other Software:** The application is fully compatible and tested against the most common assistive applications of Android, except from the applications that are using external hardware as the BrailleBack, which is an add-on accessibility service for the connection of an Android Smartphone with Braille displays via Bluetooth, which was not available.

### 5.3 Experimental Evaluation

For the real-world evaluation of the final product, a case study for the building of the University of Macedonia was created. The procedure of the the creation of this case study and the results obtained, are presented in detail below.

The first step was the creation of the visual representation of the building in Anyplace Architect web service. This step was achieved with screenshots taken from the project Pamak Map<sup>11</sup>

After the successful insertion of the images of the floor plans, the Points of Interest and the connections between them were created, an example of the Ground floor is presented in Figure 33



**Figure 33:** Anyplace Architect - UoM Ground floor with PoIS

For the second step, our physical presence in the University building was essential, as the WiFi coverage of the building had to be mapped with the help of the logger component of the application.

The biggest part of the mapping procedure was done by a Nokia 5 Smartphone and the results of the scanning were available online, some minutes after the upload of the RSS logs. The results of the scan for the ground floor are available in the Figure 34 below.

The Connectivity Scale is presented by the following colors:

As presented in the Figure 34 above, the connectivity levels of the lower level of the building of University of Macedonia are low both due to the lack of WiFi Access points

<sup>11</sup>[http://ilust.uom.gr/~kiki/main\\_index.php#17/40.62514/22.96072](http://ilust.uom.gr/~kiki/main_index.php#17/40.62514/22.96072)

**Table 6:** Connectivity Scale

GREEN:	Excellent connectivity to the WiFi at the highest speeds (0 to 60 dBm).
YELLOW:	Good Connectivity to the WiFi but not always in the highest speeds (-60 to -70 dBm).
ORANGE:	Accepted level of connectivity with occasional interruptions (-70 to -90 dBm).
PURPLE:	Intermittent connectivity level, with low transfer rates and performance issues (-90 to -100 dBm).
RED:	No connectivity to the WiFi (-100 to -110 dBm)



**Figure 34:** Anyplace Architect - UoM Ground floor with Fingerprints



but also because the source of the submitted results was only one device.

The results in Anyplace Accessible Application had a very poor accuracy, which was less than 4 meters when the user was not walking and between 4 to 7 meters while the user was walking. The results were the same for the mezzanine and the first floor of the building.

The results of the upper floors of the building, in which there are WiFi access points, were better as it is shown in the representation of the second floor of the building in the tower of Applied Informatics in the Figure 35. The results are the same also for the third floor of the building. In the cases of the last mentioned floors, the accuracy of the application were much better, as less than two meters for a standing user and between 2 to 4 meters for a walking user. In some cases, specifically outside of the laboratories 234 and 334 the accuracy for a standing user was less than 0.5 meters.

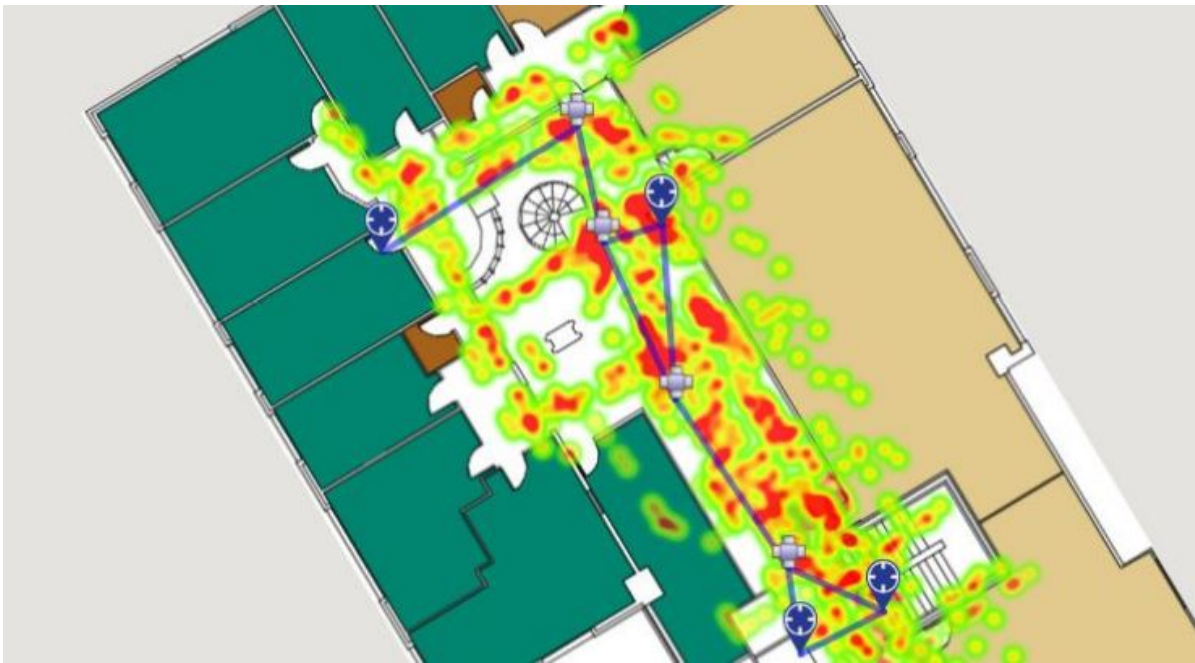
During the procedure, all the available voice commands were tested and successfully recognised by the system. The system was able to:

- Recognise the voice commands related to the available PoIs and start the navigation procedure to the point indicated by the user.
- Process and execute all the voice commands, connected with the buttons of the User Interface and successfully call the related activities
- Process the fault voice commands and provide accessible feedback, that the command was not recognised by the system.

Furthermore, all the QR labels were successfully recognised and processed by the system. When QRs with location information were scanned the system was able to locate correctly the user and inform both the system and the user about the new location reached, with appropriate messages.

For all the other QR labels, which in the term of the project are considered as informative labels, the system was able to produce accessible messages and inform the user about the retrieved content.

In this respect it could be assumed that the use of the Anyplace Accessible Application in the building of the University of Macedonia at the time, is feasible for all the user groups disregarding any possible disabilities, except of the completely blind users. The completely blind users, can still use the application to be informed about their current position, the floor in which a PoI can be found and the available Points of Interest, but before more crowdsourced RSS data and WiFi Access Points are available, the use of the application for completely blind users in the University of Macedonia cannot be considered safe.



**Figure 35:** Anyplace Architect - UoM 2nd floor of Applied Informatics tower with Fingerprints

## 6 Conclusions

In this Section is summarized the work done in this project, the project's contribution and the future work.

### 6.1 Project summary

The goals of this project were to study:

- the benefits that modern technology can bring to the navigation of Visually impaired people, and to investigate the difficulties that people with different kind of physical disabilities have in the use of Android Smartphones,
- the elicitation of the requirements that an application should fulfil in order to be usable for every user disregarding of a wide range of potential disabilities that could be drawbacks in a smartphone usage and lastly,
- the development of a mobile application for Android platform that would serve the need of every user for internal navigation in big buildings.

The most of the goals of the project were accomplished, since a study of how modern technology can assist the navigation of visual impaired people was made in section 2 and the analysis of the requirements is presented in section 3.

The ultimate goal of the project, was achieved as described in detail in section 5, however the requirement for the safe navigation inside the University building for completely blind users was not feasible due to lack of available data and suitable infrastructure in the specific case.

It should be mentioned that the most important goal of this project was to prove that a universal application design is feasible to be achieved, even in the cases that the application was not implemented having the people with disabilities in mind and we believe that this goal was successfully achieved by having the human factor as the center of the design process. So, it could be said that the reply to this challenge is the user-centered design.

### Future Work

In this section are discussed the possible improvements that could be made in the future, in order for the delivered functionality of the application to be improved or extended.

#### Improvements in the provided navigation feedback

At the time, the feedback that the system provides to assist the navigation procedure is only vibrational as no real time route audio guidance could be used in the contexts of Anyplace project as long as it is using the Google Maps API without the violation of Google Maps Term of Service 10.4.c.iii<sup>12</sup>. As the Open Street Maps (OSM) will be implemented in Anyplace service soon (before July 16 2018<sup>13</sup>), this implementation

---

<sup>12</sup><https://developers.google.com/maps/terms#10-license-restrictions>

<sup>13</sup><https://github.com/dmsl/anyplace/issues/238>

of real time audio feedback will be feasible and legal with the help of OSMBonusPack available in GitHub<sup>14</sup>.

#### **Improvements in the provided information with the implementation of IoT**

The positive results that could be achieved with an integration like this would be impressive as, for example an integration of Smart beacons could allow the stairs to inform the user about their existence or the information that a laboratory is available for use in specific hours could be a valuable information for a student. Moreover the localization of the device would be more accurate, as an extra way to track the user's position in the building will be available.

#### **Improvements in the change floor feature.**

The navigation between different floors could be improved by supplying the PoIs of type stair or elevator, with more information regarding the connections that could be achieved with their use in relation to the user's direction, previous location and selected PoI.

#### **Extension of the implemented functionality for Outdoor Spaces**

The implemented functionality could be extended for use in outdoor spaces as well.

## **6.2 Final Remarks**

Having mentioned all the core modern technologies that are used to assist the navigation procedure of visually impaired people and the restrictions and limits that this procedure has, an overall account of this study can be provided. From the early stages of this project, the human factor was taken into account and the limitations that different kind of physical disabilities could cause to the interaction between the software and the user were studied and analysed in detail. As it was noted, the lack of general design patterns for applications that are the same time friendly and usable to a wide range of user groups, make the human factor the most important parameter should be taken into account during the design process.

Considering all these factors, the Anyplace open source service was selected as the starter project that could be extended and adapted to serve the needs for navigation in indoor spaces for every user. In respect to that, the Anyplace Accessible Android Application was implemented, which is suitable for use from sighted or visually impaired users, users with motor disabilities and users with learning disabilities. Thus, the application provides significant benefits to the assistance of navigation in indoor spaces. To conclude, we believe that with further improvements from future developers, with suitable infrastructure available and with the active voluntary participation from sighted users, this already useful application, could be a valuable tool that could enhance the indoor navigation experience and reduce the insecurity feeling of visually impaired users in unknown indoor environments.

---

<sup>14</sup><https://github.com/MKergall/osmbonuspack>

## References

- [1] (2013a). Visual disabilities color-blindness. <https://webaim.org/articles/visual/colorblind>. WebAIM - Web Accessibility In Mind - [Last Accessed: 25/05/2018].
- [2] (2013b). Visual disabilities low vision. <https://webaim.org/articles/visual/lowvision>. WebAIM - Web Accessibility In Mind - [Last Accessed: 25/05/2018].
- [3] (2017). Vision impairment and blindness. <http://www.who.int/en/news-room/fact-sheets/detail/blindness-and-visual-impairment>. World Health Organization - [Last Accessed: 25/05/2018].
- [4] (2017). What is diabetic retinopathy? <https://www.aao.org/eye-health/diseases/what-is-diabetic-retinopathy>. American Academy of Ophthalmology - [Last Accessed: 25/05/2018].
- [5] Akerkar, R. A. (2014). *Big data computing*. CRC Press. p.104.
- [6] Bhowmick, A., Prakash, S., Bhagat, R., Prasad, V., and Hazarika, S. M. (2014). Intellinavi: Navigation for blind based on kinect and machine learning. In Murty, M. N., He, X., Chillarige, R. R., and Weng, P., editors, *Multi-disciplinary Trends in Artificial Intelligence*, pages 172–183, Cham. Springer International Publishing.
- [7] Dang, Q. K., Chee, Y., Pham, D. D., and Suh, Y. S. (2016). A virtual blind cane using a line laser-based vision system and an inertial measurement unit. *Sensors*, 16(1).
- [8] Dong, H. (2014). Indoor navigation system for the visually impaired with user-centric graph representation and vision detection assistance. Master's thesis, University of Massachusetts Amherst.
- [9] Fair, M. and Miller, D. P. (2001). Automated staircase detection, alignment and traversal. School of Aerospace and Mechanical Engineering, University of Oklahoma.
- [10] Fallah, N., Apostolopoulos, I., Bekris, K., and Folmer, E. (2012). The user as a sensor: Navigating users with visual impairments in indoor spaces using tactile landmarks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 425–432, New York, NY, USA. ACM.
- [11] Georgiou, K., Constambeys, T., Laoudias, C., Petrou, L., Chatzimilioudis, G., and Zeinalipour-Yazti, D. (2015). Anyplace: A crowdsourced indoor information service. In *2015 16th IEEE International Conference on Mobile Data Management*, volume 1, pages 291–294.
- [12] GU, T. (2014). Real time obstacle depth perception using stereo vision. Master's thesis, UNIVERSITY OF FLORIDA.

- [13] Jafri, R. and Khan, M. M. (2018). User-centered design of a depth data based obstacle detection and avoidance system for the visually impaired. *Human-centric Computing and Information Sciences*, 8.
- [14] Ji, H., Xie, L., Wang, C., Yin, Y., and Lu, S. (2015). Crowdsensing: A crowdsourcing based indoor navigation using rfid-based delay tolerant network. *J. Network and Computer Applications*, 52:79-89.
- [15] Kannan, B., Meneguzzi, F., Dias, M. B., and Sycara, K. (2013). Predictive indoor navigation using commercial smart-phones. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13*, pages 519-525, New York, NY, USA. ACM.
- [16] Kher Chaitrali, Dabhade Yogita, K. S. D. S. D. A. (2015). An intelligent walking stick for the blind. *International Journal of Engineering Research and General Science*, 3(1).
- [17] Laoudias, C., Zeinalipour-Yazti, D., and Panayiotou, C. G. (2013). Crowdsourced indoor localization for diverse devices through radiomap fusion. In *International Conference on Indoor Positioning and Indoor Navigation*, pages 1-7.
- [18] Meliones, A. and Sampson, D. (2018). Blind museumtourer: A system for self-guided tours in museums and blind indoor navigation. *Technologies*, 6(1).
- [19] Muhammad Al Amin Amali Mazlan, Mohd Haris Md. Khir, N. M. S. S. D. (2017). Wifi fingerprinting indoor positioning with multiple access points in a single base station using probabilistic method. *International Journal of Applied Engineering Research*, 12(6):1102-1113.
- [20] Owayjan, M., Hayek, A., Nassrallah, H., and Eldor, M. (2015). Smart assistive navigation system for blind and visually impaired individuals. In *2015 International Conference on Advances in Biomedical Engineering (ICABME)*, pages 162-165.
- [21] Pascolini, D. and Mariotti, S. P. (2012). Global estimates of visual impairment: 2010. *British Journal of Ophthalmology*, 96(5):614-618.
- [22] Ridwan, M., Choudhury, E., Poon, B., Amin, M. A., and Yan, H. (2014). A navigational aid system for visually impaired using microsoft kinect. 1.
- [23] Sherin Gilson, Sagar Gohil, F. K. V. N. (2015). A wireless navigation system for the visually impaired. Capstone Research Project, Interdisciplinary Telecom Program, University of Colorado.
- [24] Sommerville, I. (2007). *Software engineering*. China Machine Press. p.120-121.
- [25] Sonda Ammar Bouhamed, Imene Khanfir Kallel, D. S. M. (2013). New electronic white cane for stair case detection and recognition using ultrasonic sensor. *International Journal of Advanced Computer Science and Applications*, 4(6).

- [26] Spindler, M., Weber, M., Prescher, D., Miao, M., Weber, G., and Ioannidis, G. (2012). Translating floor plans into directions. In Miesenberger, K., Karshmer, A., Penaz, P., and Zagler, W., editors, *Computers Helping People with Special Needs*, pages 59–66, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [27] Syed Tehzeeb Alam, Sonal Shrivastava, S. T. A. (2015). Smart assistive device for visually impaired people. *International Journal of Engineering Research and Technology* ,IJERT, 4(3).
- [28] Tilch, S. and Mautz, R. (2010). Current investigations at the eth zurich in optical indoor positioning. In *2010 7th Workshop on Positioning, Navigation and Communication*, pages 174–178.
- [29] Zeinalipour-Yazti, D. and Laoudias, C. (2017). The anatomy of the anyplace indoor navigation service. *SIGSPATIAL Special*, 9(2):3–10.
- [30] Zeinalipour-Yazti, D., Laoudias, C., Georgiou, K., and Chatzimilioudis, G. (2017). Internet-based indoor navigation services. *IEEE Internet Computing*, 21(4):54–63.