

University Degree in Computer Engineering
Academic Year 2019

Bachelor Thesis

“Mobile Application for Emergencies Management”

Author: David Arroyo Martín

Tutor: David Griol Barres

Leganés, June 2019



[Include this code in case you want your Bachelor Thesis published in Open Access University Repository]

This work is licensed under Creative Commons **Attribution – Non Commercial – Non Derivatives**

Acknowledgments

First of all, I would like to thank my tutor David Griol Barres for his help and support during all the time I have been working on this Bachelor project. He gave me the opportunity to work on an initial idea he had and we both, together, developed that idea to make it become the one that is represented in Bachelor project.

To my friends, who they know mean a lot to me. Each of them represents something very important in my life and I know our friendship will always keep this way.

Finally, and for sure the most important, I would like to thank to all my family their unconditional support not only during the development of this project but during all my degree and, most importantly, in my personal life. You have always encouraged me to do my best and to become a better person. I am what I am because of your teaching and the love you give me every single day of my life.

Thank you all,

David.

Summary

The Bachelor Thesis presented in this document is based in the development of an application for Android devices oriented to everyone. The main objective of the application is to notify users about situations of emergency that happen near their location.

The first of the principal functions of the application allows users to notify about situations witnessed considered as emergencies through an interface that allows the creation of voice messages that are sent to a server that will store them to notify other users.

The second principal function of the application is being able to notify any emergency situation happening near the user. To do so, users are able to see a list of all emergencies near them, having the possibility to see in a map the exact location of each of the them.

The server with which the communication is established has the responsibility to check that the emergencies sent by users are trustful by means of a sentimental analysis of the text sent through an external API provided by MeaningCloud. This analysis is carried out with the objective of assuring that the message is not sent with dishonest purposes. In addition, once the message truthfulness is verified, the text is analyzed in search of keywords with which it can be identified with a particular type of emergency.

Moreover, all the data stored in the database is not only obtained from messages sent by users, it also includes emergencies sent directly from the official twitter account of the DGT (Dirección General de Tráfico) that are obtained by means of the Twitter API.

The report includes an analysis of the tools and platforms used to develop the system. Some of these resources are the client-server paradigm, Android Studio, programming languages like Java for Android, Python or MySQL and, external APIs provided by MeaningCloud or Twitter.

Finally, also the different parts in which the system is divided is described in depth, including the way in which they have been developed along with the decisions taken during their development.

Keywords: emergencies notification, voice message, sentiment analysis, Android, Python, Twitter, API, DGT, Android Studio, MySQL, SQLite, MeaningCloud.

Index

<i>Acknowledgments</i>	<i>III</i>
<i>Summary</i>	<i>IV</i>
<i>Index</i>	<i>VI</i>
<i>Figure index</i>	<i>X</i>
<i>Table index</i>	<i>XV</i>
CHAPTER 1. Introduction	1
1.1. Motivation	1
1.2. Objectives	3
1.3. Resources.....	4
1.3.1. Hardware resources	5
1.3.2. Software resources	5
1.4. Development phases.....	6
1.5. Document structure	8
CHAPTER 2. State of the Art	11
2.1. Operating Systems overview	11
2.2. Operating system decision	12
2.3. Android.....	14
2.3.1. Architecture	14
2.3.2. Android components.....	17
2.3.3. Version history.....	18
2.3.3.1. Android KitKat	20
2.3.3.2. Android Lollipop	21
2.3.3.3. Android Marshmallow	22
2.3.3.4. Android Nougat	23
2.3.3.5. Android Oreo	24
2.3.3.6. Android Pie	26
2.4. Resources needed by the system.....	27
2.4.1. Text sentiment analysis and topic extraction.....	27
2.4.2. Google Voice Recognizer	28
2.4.3. Python libraries for Twitter	29
2.5. Similar applications in the market	30
2.5.1. Alertcops	30
2.5.2. My112.....	31

2.5.3.	Waze.....	33
2.5.4.	DGT traffic map.....	34
2.5.5.	Similar applications summary	35
CHAPTER 3. General description of the system		37
3.1.	System overview	37
3.2.	Technologies used.....	38
3.2.1.	Android Studio	38
3.2.1.1.	Setup	39
3.2.1.2.	Java Development Kit.....	40
3.2.1.3.	Software Development Kit	41
3.2.1.4.	Android Virtual Device	41
3.2.1.5.	Application structure	43
3.2.1.6.	Usage.....	44
3.2.2.	MySQL	44
3.2.2.1.	Setup	44
3.2.2.2.	Usage - Database structure	46
3.2.3.	SQLite	48
3.2.3.1.	Usage - database structure.....	49
3.2.4.	MeaningCloud.....	49
3.2.4.1.	Setup	49
3.2.4.2.	Usage.....	51
3.2.5.	Python	53
3.2.5.1.	Setup	53
3.2.5.2.	Usage.....	53
3.2.6.	Twitter API.....	54
3.2.6.1.	Setup	55
3.2.6.2.	Usage.....	56
3.2.7.	Firebase API	57
3.2.7.1.	Setup	57
3.2.7.2.	Usage.....	58
CHAPTER 4. Detailed description of the application modules.		60
4.1.	Splashscreen	60
4.2.	Instructions of use.....	61
4.3.	Login	62
4.3.1.	Use cases	63
4.4.	Register.....	65

4.4.1. Use cases	66
4.5. Password recovery	67
4.5.1. Use cases	68
4.6. Main Screen	69
4.6.1. Use cases	70
4.7. Maps	72
4.7.1. Use cases	72
4.8. Settings	73
4.8.1. Use cases	75
4.9. Emergency update service.....	76
4.10. Application-Server connection	77
4.10.1. Login interaction	77
4.10.2. Register interaction.....	77
4.10.3. Remember password interaction	78
4.10.4. Main screen interaction	78
4.10.5. Settings interaction	79
4.10.6. Emergencies update service interaction	80
CHAPTER 5. Evaluation of the application	82
5.1. Evaluation methodology	82
5.2. Evaluation results.....	87
CHAPTER 6. Project management	95
6.1. Temporal planning	95
6.2. Budget estimation.....	97
6.3. Regulatory framework	101
6.4. Socio-economic factor	103
CHAPTER 7. Conclusions and future work.....	106
7.1. Conclusions	106
7.2. Future work	108
Glossary.....	110
Bibliography	114

Figure index

Figure 1.1: Amount of phone calls made to 1-1-2 each year from 1998

Figure 1.2: Distribution of phone calls made to 1-1-2 by type of emergency

Figure 1.3: Evolution of Smartphone users in the World from 2016

Figure 1.4: Development phases of the system

Figure 2.1: Android vs iOS sales evolution

Figure 2.2: Android vs iOS, world spread differences

Figure 2.3: Android Architecture

Figure 2.4: Android activity life cycle

Figure 2.5: Android version share pie chart [25]

Figure 2.6: Android KitKat home screen

Figure 2.7: Android KitKat “Ok Google” functionality

Figure 2.8: Android KitKat notifications

Figure 2.9: Android Lollipop notifications

Figure 2.10: Android Marshmallow screen scan

Figure 2.11: Android Nougat split screen

Figure 2.12: Android Nougat Google Assistant

Figure 2.13: Android Oreo picture-in-picture

Figure 2.14: Android Oreo Google Play Protect

Figure 2.15: Android Pie main screen without navigation buttons

Figure 2.16: Android Pie applications monitorization

Figure 2.17: Google Voice Recognition

Figure 2.18: Alertcops application

Figure 2.19: My112 application

Figure 2.20: Waze Application

Figure 2.21: DGT traffic map

Figure 3.1: Client-server system abstraction

Figure 3.2: Android studio platform downloads

Figure 3.3: Android Studio installation assistant

Figure 3.4: JDK 8 platform downloads

Figure 3.5: JDK 8 installation assistant

Figure 3.6: AVD Manager

Figure 3.7: AVD creation process

Figure 3.8: Bachelor Thesis application structure

Figure 3.9: MySQL available installation packages

Figure 3.10: MySQL installation

Figure 3.11: SQLite architecture

Figure 3.12: New account creation

Figure 3.13: User account License Key

Figure 3.14: MeaningCloud custom dictionary creation and modification

Figure 3.15: MeaningCloud dictionary entry

Figure 3.16: Python 3 installation

Figure 3.17: Twitter Developer new application creation

Figure 3.18: Twitter API keys

Figure 3.19: New Firebase project creation

Figure 3.20: Build.gradle lines inclusion for Firebase authentication

Figure 4.1: Splashscreen

Figure 4.2: Application presentation slides

Figure 4.3: Login interface

Figure 4.4: Google accounts login

Figure 4.5: Login successful

Figure 4.6: Login unsuccessful

Figure 4.7: Google login successful

Figure 4.8: Recover the password selection

Figure 4.9: Recover the password selection

Figure 4.10: Register interface

Figure 4.11: Successful registration

Figure 4.12: Unsuccessful registration

Figure 4.13: Account password recover

Figure 4.14: Account password recovered successfully

Figure 4.15: Account password not recovered

Figure 4.16: Main screen interface

Figure 4.17: Send emergency

Figure 4.18: Google Speech recognizer

Figure 4.19: Select emergency

Figure 4.20: Show location of specific emergency

Figure 4.21: Show location of all emergencies in list

Figure 4.22: Show application menu

Figure 4.23: Application menu

Figure 4.24: Maps activity

Figure 4.25: Select emergency icon in map

Figure 4.26: User settings

Figure 4.27: Account information interface

Figure 4.28: Change password interface

Figure 4.29: Car mode state update

Figure 4.30: Select option

Figure 4.31: Delete account

Figure 4.32: Update information

Figure 4.33: Update password

Figure 4.34: Login interaction

Figure 4.35: Register interaction

Figure 4.36: Remember password interaction

Figure 4.37: Main screen interaction

Figure 4.38: Settings interaction

Figure 4.39: Emergencies update service interaction

Figure 5.1: EmergencyPal questionnaire

Figure 5.2: Question 1

Figure 5.3: Question 2

Figure 5.4: Question 3

Figure 5.5: Question 4

Figure 5.6: Question 5

Figure 5.7: Question 6

Figure 5.8: Question 7

Figure 5.9: Question 8

Figure 5.10: Question 9

Figure 5.11: Question 10

Figure 5.12: Question 11

Figure 5.13: Question 12

Figure 5.14: Question 13

Figure 5.15: Question 14

Figure 5.16: Question 15

Figure 5.17: Question 16

Figure 5.18: Question 17

Figure 6.1: Gantt diagram tasks

Figure 6.2: Gantt diagram

Figure 6.3: Devices used by people in Spain

Table index

Table 2.1: Android API versions

Table 2.2: Applications features comparison

Table 3.4: User table structure

Table 3.4: Road_PK table structure

Table 3.4: Road_code_name table structure

Table 3.4: Emergency table structure

Table 6.1: Company costs in terms of a worker

Table 6.2: Hardware resources costs

Table 6.3: Software resources costs

Table 6.4: Product linear amortization

Table 6.5: Direct costs

Table 6.6: Bachelor Degree Thesis costs summary

CHAPTER 1. Introduction

The initial chapter of the report has the main objective of presenting a clear and simplified view of the reasons that motivated the development of an Android application that is used to notify emergencies to any other user interested in knowing what is happening near them. Moreover, an explanation of how the process of development is provided along with the different resources, both hardware and software, needed. Finally, there is a brief description of the different parts in which the document is divided.

1.1. Motivation

In big cities such as Madrid and its surroundings the amount of calls made to the emergencies phone numbers is very high. In 2016, even though the number of phone calls is being reduced slightly year by year, there were 4.591.849 phone calls to the emergencies number 112, which is around 12.591 phone calls each day [1]. The reasons why these phone calls are done are very varied, such as health emergencies, robberies, traffic accident and multiple other kind of emergencies.

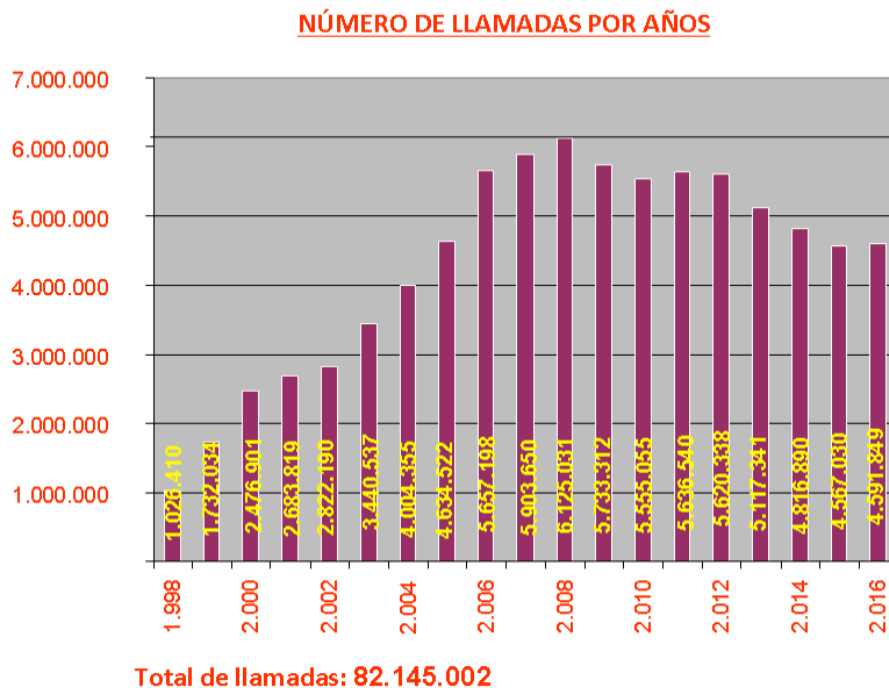


Figure 1.1: Amount of phone calls made to 1-1-2 each year from 1998 [1]

In Figure 1.1, we can see the evolution of phone calls made to emergencies number 112 since 1998, where the number of phone calls was 1.026.410, until 2016, where there were 4.591.849 phone calls. Between 1998 and 2008, the number of calls increased year by year but, after this year, they are being reduced each year.

As we can see in Figure 1.2, the reasons of the emergency calls that happen in cities are very varied and include: health services, public safety, traffic, rescue and fire extinction and others. Almost half of the total amount of emergency calls is related to health services, more than a quarter to public safety, road safety about eleven percent, rescue and fire extinction almost seven percent and other type of reasons almost eight percent.



Figure 1.2: Distribution of phone calls made to 112 by type of emergency [1]

On the other hand, the number of incidences that happen during a day in roads and highways is very big and people is usually not aware of what they may find in their journeys. Moreover, once they start to drive the only information they receive about traffic incidences or traffic accidents is provided by a GPS, which is in many cases not active and, if it is, the information provided is not usually complete.

The main motivation of this project appears with the necessity to keep people informed of real emergencies and situations of danger in an organized and clear way. All the emergencies shown in the app include both emergencies that happen in cities and emergencies that happen in roads or highways.

The emergencies shown in the application will be those that can affect the normal lifestyle of people so, as we can see in Figure 1.2, those emergencies related to health services will not be taken into account as they are not considered as emergencies of public interest.

Nowadays, there exist different programs and applications that allow people to be informed about what is happening near them or to inform the police or health services about personal emergency situations, such as an online map from the DGT that has

information about what is happening in any road of the country, or other smartphone applications that permit people to call the police or the emergency services phone 112 for personal situations that have happened to them. In addition, there exists some applications that can notify about emergencies of public interest, but they only work in some Autonomous Communities and do not include both road and highway emergencies.

The reason why the application is meant to be used on a smartphone is because of the expansion of these devices over the world and the fact that as they are easily portable, they are available at any time. As we can see in Figure 1.4, the number of people using a smartphone has increased by more than five hundred million users from 2016 to 2018 and is expected to grow at a similar rate in the following years.



Figure 1.3: Evolution of Smartphone users in the World from 2016

1.2. Objectives

The main objective of this project is to provide an Android application to manage emergencies sent by other users and to help others know what is happening near them. One of the objectives of developing the application in this way is that as much people uses the application, the more precise the emergencies will be. This happens because if every user participates in the notification of dangerous situations, there won't exist any situation which has not been notified.

Moreover, there are situations in which a user may not be able to notify possible emergencies because it cannot access the phone in that moment. This is likely to happen to those users which are driving and there is no partner in the car. For these situations, the system takes from the official DGT twitter account all the emergencies known in roads and highways in real time.

The application integrates an interface that provides the user the following capabilities:

- **Inform about a new emergency:** Notifications of new emergencies are sent by users by means of the Speech to Text functionality provided by Android.
- **Inform about near emergencies:** A list of all emergencies that are in the range of the user will be shown in the screen.
- **Visualize emergencies in a map:** The user will be able to open a map that shows where the emergencies are located.
- **Voice notifications:** The user will be able to enable / disable the possibility to make the system notify by speech all new emergencies, without the necessity to have the application opened. This capability will make use of the Text to Speech functionality provided by Android.

To achieve these main objectives, a few partial objectives are defined:

- Understand how Android applications are structured.
- Allow the users to receive the notifications without the need to start the application.
- Investigate and discover ways to include emergencies from other sources different than the application users.
- Go deep into data storage mechanisms and socket connections between server and client.
- Put into practice all the knowledge obtained during the time I have studied my degree.

1.3. Resources

During the development of the system, multiple resources, both hardware and software, have been used.

1.3.1. Hardware resources

- Personal computer for programming development.
- Smartphone Oneplus 3T version Android Oreo 8.0.0.
- External screen BenQ.

1.3.2. Software resources

- Android Studio: Android programming environment
- MeaningCloud text features API: Web API used for text analysis.
- Python3: High-level programming language.
- MySQL: Database programming language.
- MySQL Workbench: Environment to manage SQLite databases.
- Software Development Kit (SDK)
- Java Development Kit (JDK)
- Android Virtual Device (AVD): Virtual device that works like a physical Android device.
- Atom: Text editor used to develop the server.
- OneDrive: Web storage platform used to store the project.
- Firebase: Mobile development platform used for Google login authentication.
- Twitter API: Used to read tweets sent by the official twitter account of DGT.
- Microsoft Office Word: Microsoft platform used to develop the memory of the project.

1.4. Development phases

The process of development of the system is divided in four main stages: *planification*, *development*, *evaluation* and *documentation*. The stages are divided in this way because of all the knowledge acquired in the course Software Development Projects Management.

In order to show a visual representation of each of the stages in which the development process is divided, a WBS (Work Breakdown Structure) has been used. A WBS is a hierarchical and incremental decomposition of the project into phases, deliverables and work packages. WBSs are developed by starting with the end objective and successively subdividing it into manageable components in terms of size, duration and responsibility [2].

Figure 1.4 shows a WBS diagram of the system. There, the four main stages stated previously appear along with the subcomponents in which each of them is divided.

All three stages will be described below:

Stage 1: Planification.

- **Requirements Analysis:** In this step the specifications and capabilities that the system must have are defined.
- **Analysis of required technologies:** Study of the different technologies that will be necessary for the correct development of the system.
- **Analysis of developing tools:** Study of all the different tools that are going to be used during the development of the system.
- **Analysis of related applications:** Study all the different applications that are in the market and make a comparison between the capabilities of each system to discover new innovative ideas for **EmergenciesPal** system.

Stage 2: Development.

- **System design:** Recognize the different parts in which the system is divided.

- **Implementation:** Programming process of the different parts of the system.
- **Components integration:** Combine all the different parts of the system to make them work all together.
- **Developer level testing:** Test the system to confirm that performs as expected.

Stage 3: Evaluation.

- **User level testing:** Let users test the application.
- **User evaluation:** Evaluation done by the users that have tested the application.

Stage 4: Documentation.

- **Document writing:** Write the actual report.
- **Final presentation:** Prepare a final presentation.

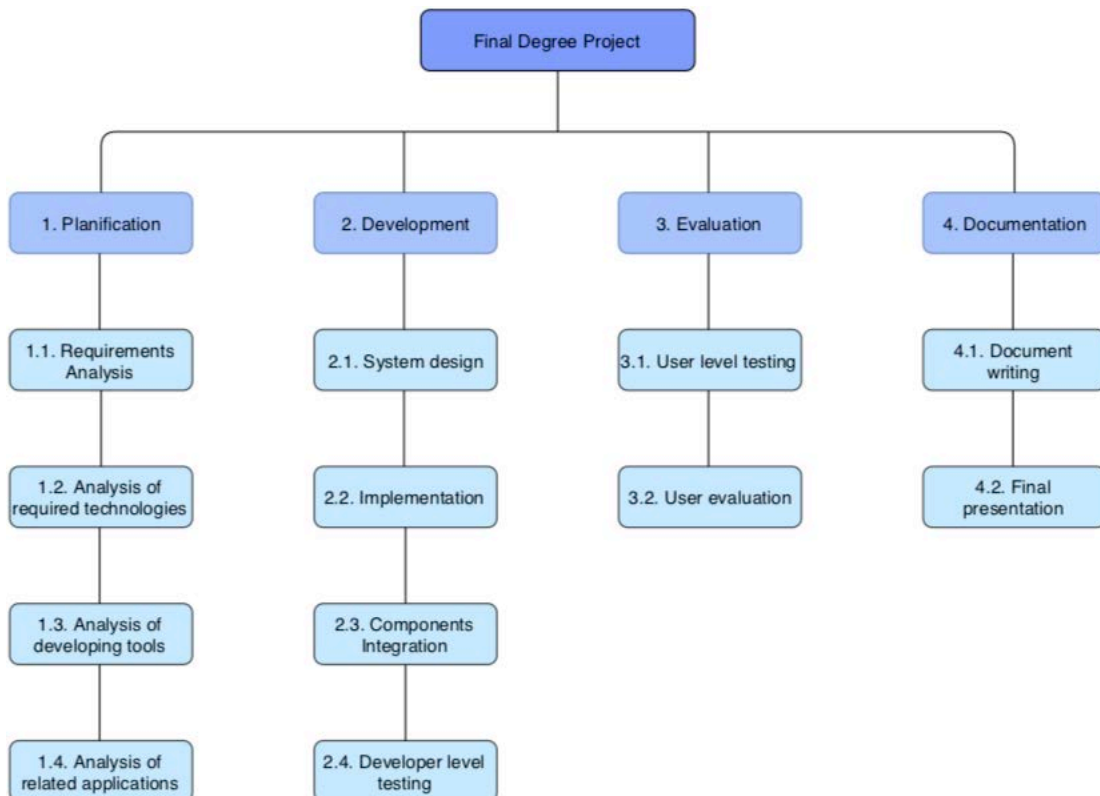


Figure 1.4: Development phases of the system

1.5. Document structure

Summary. This section describes a general overview of the Final Degree Project.

General index. Shows the structure of the system along with the pages in which each of the sections start.

Figure index. This section collects all the figures shown in the report enumerated with the following format: “Figure N.M”, being N the number of the section where it appears and M the number assigned to that figure in the section.

Table index. This section collects all the tables used in the report enumerated with the following format: “Table N.M”, being N the number of the section where it appears and M the number assigned to that table in the section.

Chapter 1: Introduction. In this chapter, a vision of the motivations that inspired the creation of this application is shown along with the capabilities that it should have. In addition, there is an explanation of how the development of the project is structured and, finally, the resources and tools needed to accomplish the task of development.

Chapter 2: State of the Art. This chapter shows an overview of the most important operating systems for mobile devices that are in the market, Android and IOS. Moreover, there is an explanation of why Android was chosen over IOS for the development of this project as well as a detailed description of Android operating system. Finally, there is an explanation and study of other similar apps that are in the market.

Chapter 3: General description of the system. In this chapter, there is an explanation of the system along with a description of all the different technologies used during the development of the system.

Chapter 4: Detailed description of the components of the system. This chapter is focused on making a detailed description of all the different modules that form the system. It is divided in three main parts that are the server side, the application and, finally, the database. For each of the modules, there is a description of how they are implemented, and the decisions taken during their development.

Chapter 5: Evaluation of the application. This chapter shows a study done with the results obtained from different users when using the application. Finally, there is a discussion about the results obtained and which are the conclusions extracted from them.

Chapter 6: Conclusions and future work. This chapter describes the feelings of having finished and evaluated the system, taking into account all the results obtained in

the previous process of evaluation. Moreover, there is a description of how the system could evolve in the future.

Glossary. This section describes all the different technical terms and concepts used in the report that are difficult to understand.

Bibliography. This section collects all the bibliographic references used take information from for both the development of the system and the writing of the report. Each of the references are enumerated using the following format: “[N]”, being N the number of the reference.

CHAPTER 2. State of the Art

This chapter collects all the context in which this Bachelor Degree Project is immersed. Firstly, an overview of the two most important Operating Systems that are now in the market is presented. Below, there is an explanation of why I chose one of them over the other and, after this explanation, there is a more in-depth description of Android Operating System.

Finally, there is a study of other applications related with the developed one in order to show the similitudes and particularities of the Bachelor Degree Project developed system.

It is important to say that only Android and iOS are going to be analyzed in both the operating systems overview and the operating system decision because they are the two most important operating systems nowadays. Other operating systems such as Windows Phone or Blackberry OS are not be considered because they are unused, and their development and maintenance has been finalized.

2.1. Operating Systems overview

Android is an operating system developed by Google and designed for touchscreen mobile devices such as smartphones and tablets. Nowadays, Android has expanded into “Android TV” for televisions, “Android Auto” for cars and “Wear OS” for smartwatches, each of them with a specialized user interface [15].

Android was initially developed by Android Inc. in 2003, which was later bought by Google in 2005. The first Android device was sold in 2008 and since that first version, Android has gone through multiple upgrades until today, with the 9th version, also known as Android Pie [15].

The first companies that started working with Android belonged to a technology consortium called Open Handset Alliance. In this consortium there were companies like Google, manufacturers such as HTC, Motorola and Samsung, chipset makers like Qualcomm and wireless carriers. The main goal of this group of companies was, as they stated, to develop “the first open and comprehensive platform for mobile devices” [15].

iOS is a mobile operating system created and developed by Apple Inc. One particularity of iOS is that it is designed exclusively for its hardware so there is no other manufacturer company that uses this operating system. As Android, iOS is not only used for mobile phones and it is used also in other devices such as iPod Touch and the iPad [18].

iOS was unveiled in January 2007 and released in June of that year. It was named as “iPhone OS” in its release and then renamed to “iOS” in 2010. Each year, Apple releases a new version of iOS and they are now in the 12th version [18].

2.2. Operating system decision

In order to make a decision about choosing the operating system in which the project is going to be developed, it is important to analyze both of them in terms of importance in market, how spread they are in the world, development costs and my own programming knowledge.

As it is seen in Figure 2.1., Android is the operating system with most sales in the world, being iOS the second one. Between both of them, they have almost 100 percent of the market shares, but the difference between them is very big. Of that 99.9 percent of total sales, Android has 85.9 percent, which means that the amount of people using devices with this operating system is much higher than those of iOS [17].

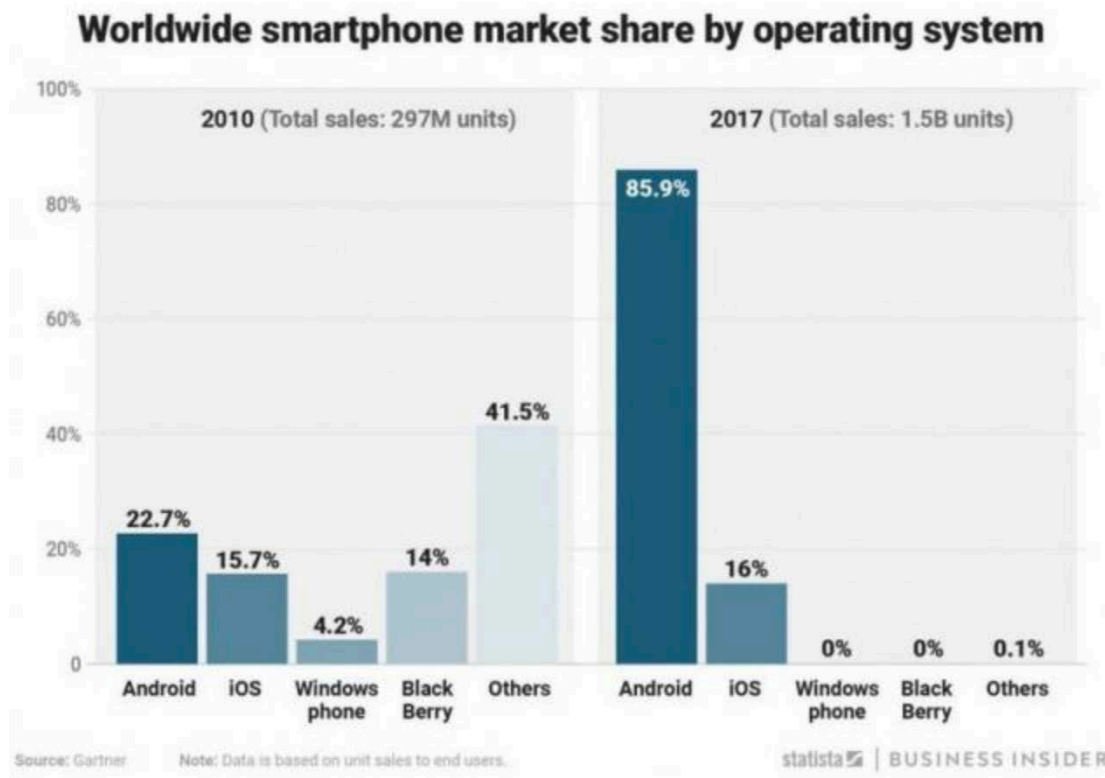


Figure 2.1: Android vs iOS, sales evolution [17]

Moreover, as we can see in Figure 2.2., it is not only that Android is the most used operating system but it is also the most widespread. Apple is the market leader in only some countries such as United States, Canada, Australia, New Zealand and a few European countries. Whereas, on the other hand, Android is the leader in most European countries, Center and South America, and Asia [17].

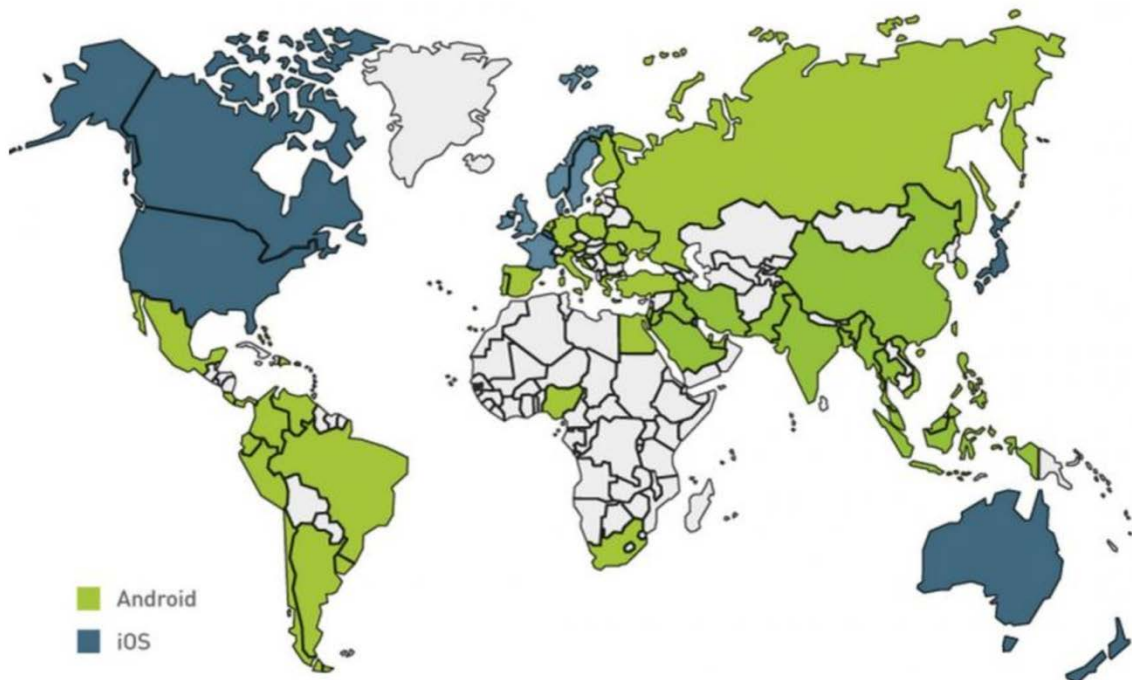


Figure 2.2: Android vs iOS, world spread differences

In relation to development costs, Android and iOS are quite different because, on the one hand, in order to publish an application in the Android market (Play Store) developers only need to pay 25 euros / dollars and, on the other hand, in order to publish an application in the Apple market (App Store) developers must pay 99 euros / dollars each year. Moreover, Android can be developed from computers running Windows, Mac OS or any Linux distribution, whereas iOS applications can only be developed from computers running Mac OS [19].

Finally, the last thing to consider is the previous knowledge in programming Android and iOS applications. In my personal case, before developing this application, I did not develop any iOS application, but I did develop one simple Android application, so my knowledge in developing Android applications was just a bit higher.

In conclusion, after presenting the differences between Android and iOS, the decision taken was to develop the application for Android devices because of how widespread it is, the possibility to develop the app in any kind of computer and, finally, because of the previous knowledge.

2.3. Android

2.3.1. Architecture

Android is presented as an open source operating system with the objective of improving Android by means of new features discovered and implemented by developers. In order to develop any application for Android devices, it is important to understand how Android is structured and how it works.

Android architecture is divided in six main parts (see Figure 2.3): **Linux Kernel**, **Hardware Abstraction Layer**, **Libraries**, **Android Runtime**, **Application Framework** and **Applications**.

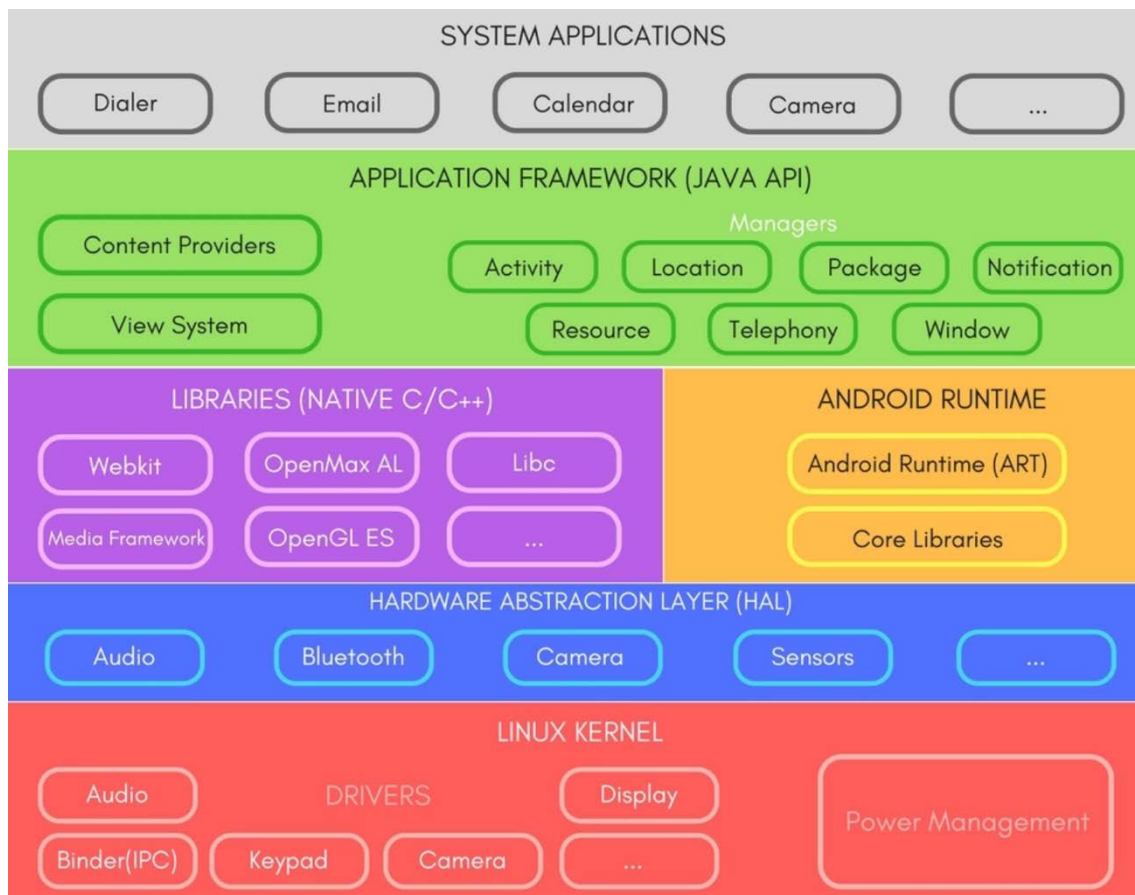


Figure 2.3: Android Architecture

- **Linux Kernel:** It is the base of the operating system. It is involved in very important functions of the system such as security, file system management, process management, network stack or even the drivers model.

Android is built over a kernel based on Linux because of its portability. This means that, as Linux is very easy to compile in very various types of hardware, manufacturers do not need to create specific hardware for the operating system, in reality they just need to modify Android to adapt it for their hardware requirements [20][21][22][23].

- **Hardware Abstraction Layer (HAL):** Bridge between hardware and software. Hall consists on several library modules that implement an interface for each of the hardware components.
When an API makes a call to require access to any hardware component, the system loads the library module for that specified component [21][22][23].
- **Libraries:** Collection of different libraries written in C / C++ that are used by several Android components such as HAL, which was explained before, or ART. The main objective of these libraries is to provide applications a functionality for those tasks that are repeated frequently [21][22][23]. Some of these libraries are:
 - **Libc:** Include all the headers and functions based on C programming language. It is used by all the other libraries.
 - **Surface Manager:** Create the different navigation elements that appear in the screen. It also manages all opened windows from active applications.
 - **OpenGL/S� and SGL:** Represent the graphic libraries. On the one hand, OpenGL/S� manages all 3D graphics, and, on the other hand, SGL manages all 2D graphics.
 - **Media libraries:** Provide all necessary codecs used for multimedia content.
 - **FreeType:** Allows a quick and efficient way to work with different types of fonts.
 - **SSL:** Allows to create secure connections over the network.
 - **SQLite:** Creation and management of databases.
 - **WebKit:** Engine for web navigation applications and forms the core of the by default web browser.

These libraries, in combination with the Kernel, are considered as the core of the system. Moreover, they can be used by developers for their own needs.

- **Android Runtime:** Provides a key component initially called *Dalvik Virtual Machine* which is a JVM optimized for Android. It makes use of Linux Kernel

core features such as memory management or multithreading. Moreover, it also includes libraries that allow developers to write Android applications using standard Java programming language. These Java written programs are then converted into **.dex** files thanks to a tool called **dx**.

Dalvik Virtual Machine has been later replaced by a newer component called **Android Runtime Environment (ART)**, which uses the same bytecode and **.dex** files but that has the objective of improving performance. To do so, it compiles the application only once before it is executed, which was not done in the Dalvik Virtual Machine. This newer version appeared for the first time in Android 4.4 and has been included in Android from version 5.0 until the newest one today [21][22][23][24].

- **Application framework:** Represent the collection of developing tools for any application [21][22][23]. The most important ones are:
 - **Activity Manager:** Group of APIs that manage the life cycle of an application.
 - **Window Manager:** Manage windows belonging to each application.
 - **Location Manager:** Allows applications to get information about the location of the device.
 - **Notification Manager:** Allow the application to communicate to the user events that happen during their execution.
 - **View System:** Provide a huge number of elements that permit the creation of user interfaces to be shown to the user.
- **Application:** Include all the applications installed in the device by default and those that are included afterwards. This is the layer with which most users interact, as it has everything they need for a daily use such as making phone calls or accessing the Web browser. [21][22][23]

2.3.2. Android components

All Android applications can be formed by five different components: *Activities*, *Intents*, *Broadcast Receivers*, *Services* and *Content Providers*.

- **Activity:** It provides the space to the programmer to do anything that is desired. Every “window” that can be seen in the device and with which any users interacts is an activity. Activities are formed by both a Java file that implements the code and an XML file that defines the layout that will be seen once the activity is launched.

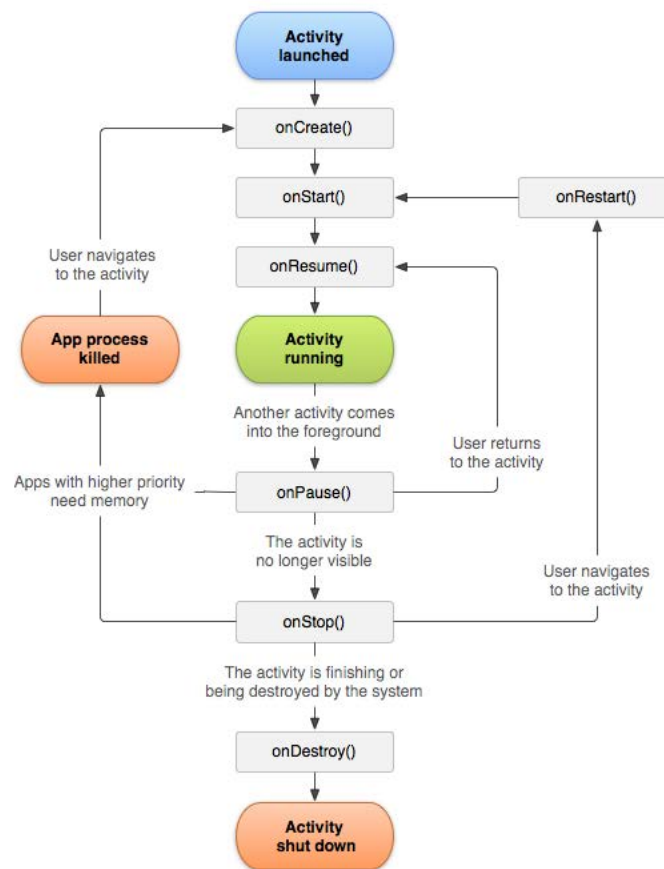


Figure 2.4: Android activity life cycle

In Figure 2.4, the life cycle of an Android activity is shown. Both *onCreate()* and *onDestroy()* methods define the start and end of an activity. States *onStart()* and *onStop()* represent the time for which a user will really notice when the activity begins and ends. *onResume()* and *onPause()* define the time when the user interacts with the application. Finally, *onRestart()* wakes up an application that was not at the forefront.

- **Intent:** It is a messaging object that is used to query an action from another app component, initiate a new activity or get the result of another one.

- **Broadcast Receiver:** Receives messages that have been sent from other applications when an event occurs.
- **Service:** Executes long running operations without the need to have a user interface, which means that they are executed in background.
- **Content Providers:** Provide the possibility to store data in a structured way by means of an object that acts like a client that requests data from any content provider that acts like a server. An example of a content provider is SQLite, which is used to store local data of Android applications inside the memory of the device.

2.3.3. Version history

As an introductory curiosity, all Android versions have a nickname associated to a different dessert, being each of them a name that starts with a letter that goes after the first letter of the previous version nickname.

In Figure 2.5. it is possible to see how each version is being used worldwide. There we can see that the most used version is Android Oreo, including both version 8.0 and 8.1, and that Android Pie, which is posterior to Android Oreo is not very used at this time. This is because this version is very new and is still arriving to many devices by means of an update.

In the “others” part of figure 2.5, all other Android versions that do not appear in the chart are included, but most of the percentage that is dedicated to this part belongs to Android versions KitKat and Jelly Bean. All other left versions are almost or completely unused.

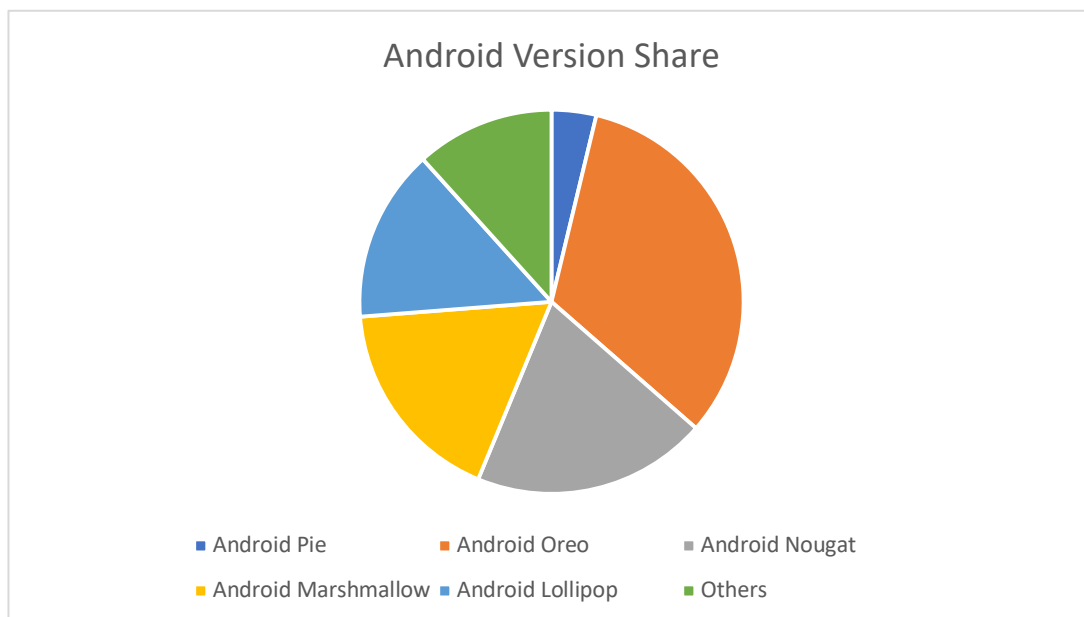


Figure 2.5: Android version share pie chart. Data obtained from [25] the 4th March 2019

In Table 2.1. we can see all the different versions through which Android has been updated. In each of the versions there are new features that were not included in previous ones, not only in the user interface but also in security and usability.

In this section, only versions following Android 4.4 (KitKat) will be described because as it can be seen in Table 2.1., they are the ones that are still being used by most people.

Version [*]	Nickname [*]	API Level [*]	Release date [*]	Market share [**]
Android 1.0	Petit Four	1	September 23, 2008	-
Android 1.1		2	February 9, 2009	
Android 1.5	Cupcake	3	April 27, 2009	-
Android 1.6	Donut	4	September 15, 2009	-
Android 2.0	Eclair	5	October 26, 2009	-
Android 2.0.1		6	December 3, 2009	
Android 2.1		7	January 12, 2010	
Android 2.2	Froyo	8	May 20, 2010	-
Android 2.3	Gingerbread	9	December 6, 2010	0.3 %
Android 2.3.3		10	February 9, 2011	
Android 3.0	Honeycomb	11	February 22, 2011	-
Android 3.1		12	May 10, 2011	
Android 3.2		13	July 15, 2011	
Android 4.0	Ice Cream Sandwich	14	October 19, 2011	0.3 %
Android 4.0.3		15	December 16, 2011	
Android 4.1	Jelly Bean	16	July 9, 2012	2.9 %
Android 4.2		17	November 13, 2012	
Android 4.3		18	July 24, 2013	
Android 4.4	KitKat	19	October 31, 2013	7.3 %
Android 4.4W		20	June 25, 2014	
Android 5.0	Lollipop	21	November 12, 2014	14.7 %
Android 5.1		22	March 9, 2015	
Android 6.0	Marshmallow	23	October 5, 2015	17.7 %
Android 7.0	Nougat	24	August 22, 2016	20 %
Android 7.1		25	October 4, 2016	
Android 8.0	Oreo	26	August 21, 2017	33 %

Android 8.1		27	December 5, 2017	
Android 9	Pie	28	August 6, 2018	3.8 %

Table 2.1: Android API versions

[*] Data was obtained from [26] the 4th March 2019

[**] Data was obtained from [25] the 4th March 2019

2.3.3.1. Android KitKat

Android KitKat was the beginning of a new era for Android in the visual point of view because they abandoned the dark and blue colors that were characteristic of previous versions such as Gingerbread and Honeycomb. With this new version, backgrounds were lighter, and the status bar switched to a transparent version, giving the OS a more contemporary appearance.

In this version there were not very many updates in relation with the previous version in relation to functionalities, but they included a new function that was “**Ok Google**”, with the objective of allowing users to complete tasks by just saying “Ok Google” out loud to the phone in the home screen or in the Google app. After saying these words, the user could ask the smartphone to send a text message, play a song or make voice searches. Another new functionality was the possibility to print documents through printers that were connected in the same Wi-fi network than the smartphone and, finally, it included the possibility to be run in devices with lower memory and it could check which applications could be difficult to run in each device by considering the smartphone specifications.

Also, this version included both Dalvik Virtual Machine and Android Runtime Environment as an experimental testing. In following versions, Dalvik Virtual Machine was definitely substituted by ART.

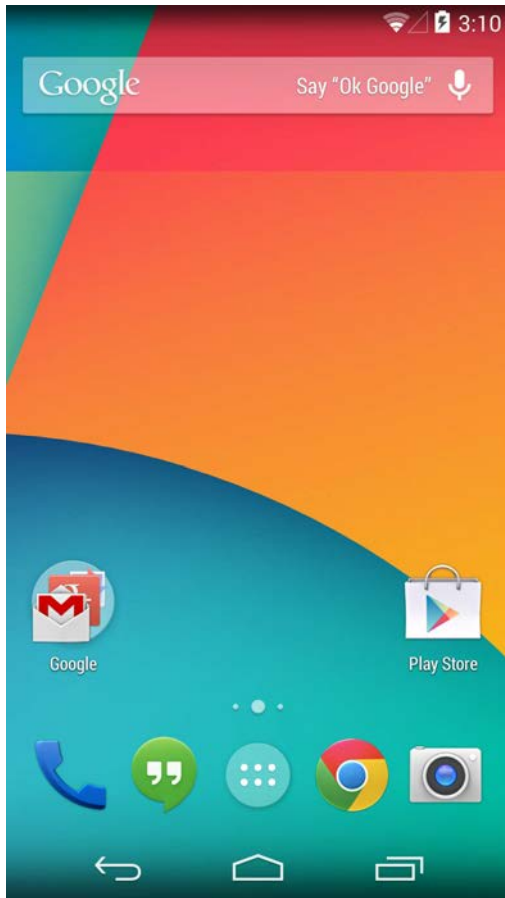


Figure 2.6: Android KitKat home screen

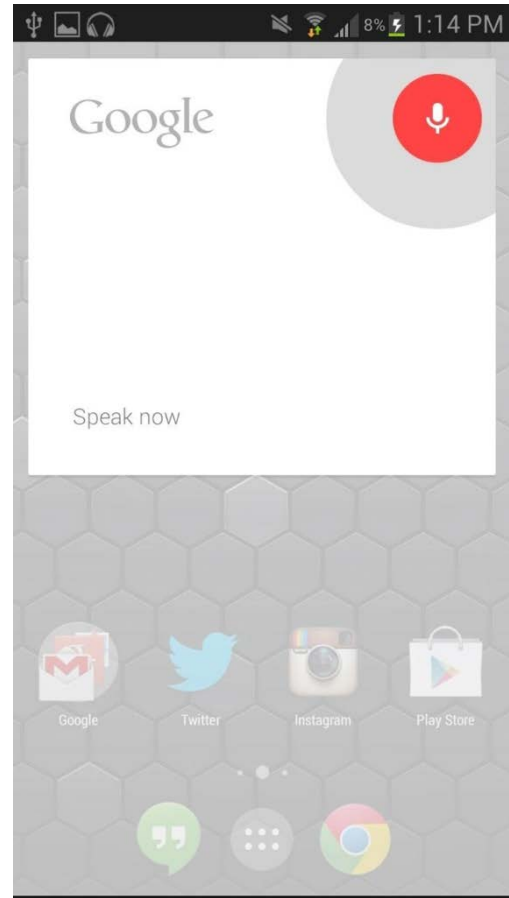


Figure 2.7: Android KitKat "Ok Google" functionality

2.3.3.2. Android Lollipop

Android Lollipop was the first version to introduce a new design language called "Material design" that changed the looking of applications including both stock apps and Play Store installed apps. Material design is based on the Cards metaphor first seen in Google Now.

Multitasking was also updated. With Android Lollipop, rather than simply show users previews of recent applications, they could jump directly to the part of the application they are interested in.

Another major update was the notifications management. In Lollipop, the lock screen is considered as the home of notifications. This means that, as in KitKat, notifications are shown in the lock screen, but they are now organized in independent cards. In the lock screen users could dismiss or deal with notifications directly without the need to unlock the device.

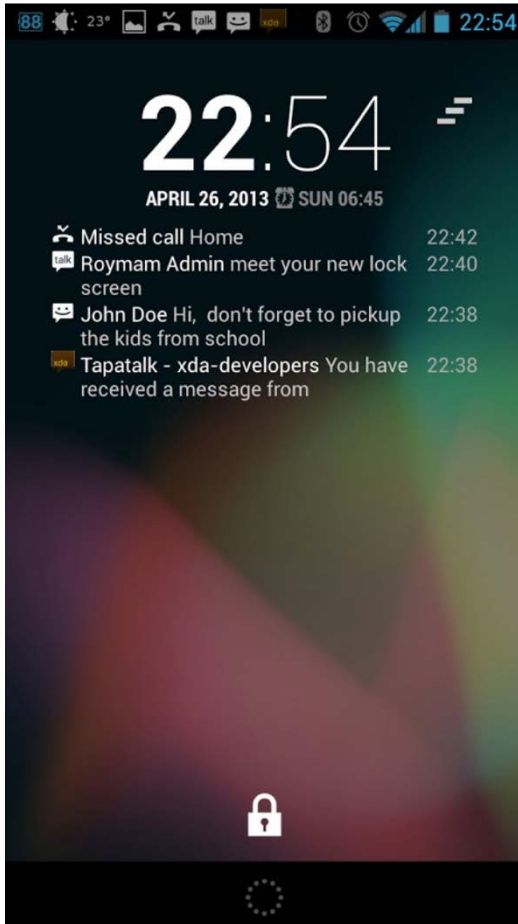


Figure 2.8: Android KitKat notifications

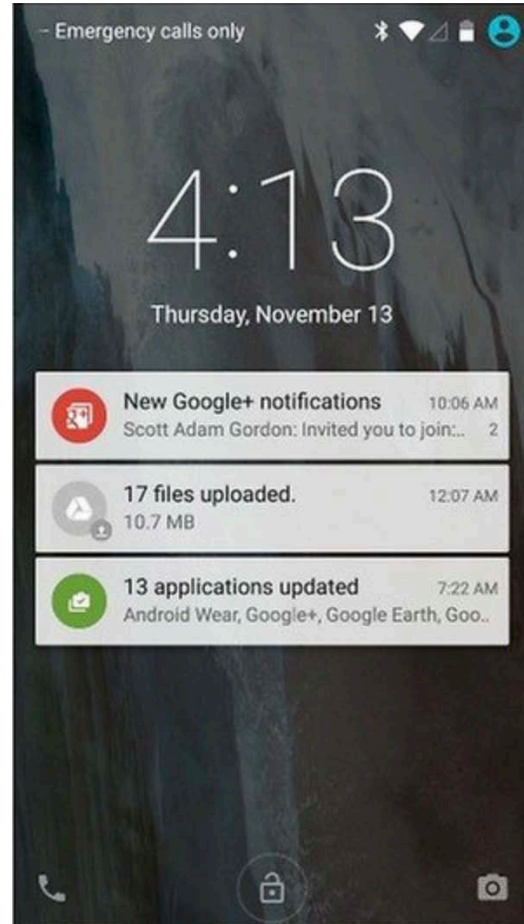


Figure 2.9: Android Lollipop notifications

Moreover, this was the first version to include support for dual-SIM and it also included a new battery management system that would allow users to understand how applications are consuming energy along with a new “battery saver mode” that would turn off all applications but the most vital ones.

2.3.3.3. Android Marshmallow

Android Marshmallow is considered as the starting point in which both Android and iOS move toward each other, Android adopting some iOS security characteristics and iOS to feel as ambitious as Android.

In this version, application permissions were changed in a way that users could install any application without the need to accept any permission and, once they start it,

the application asks individually for each of the permissions need by it. Thanks to this, there was no need to accept all permissions at installation time.

Moreover, Google created a new application that was presented with this version, it was Android Pay. With this application, phones with NFC had the possibility to make payments by linking credit cards to the users account. In order to manage the power consumption of this application and all others that were installed, this version included two new features called *Doze* and *App Standby*. On the one hand, Doze would freeze the applications if the device detected that it was not being used and, on the other hand, App Standby would block background applications of updating if they hadn't been used in a long time.

In Marshmallow, Google continued to improve their “smart” characteristics thanks to the introduction of a new feature that would scan the screen and show relevant information related to what is in it. This feature worked by tapping the home button for a few seconds and was the precursor of Google’s personal assistant, *Google Assistant*.

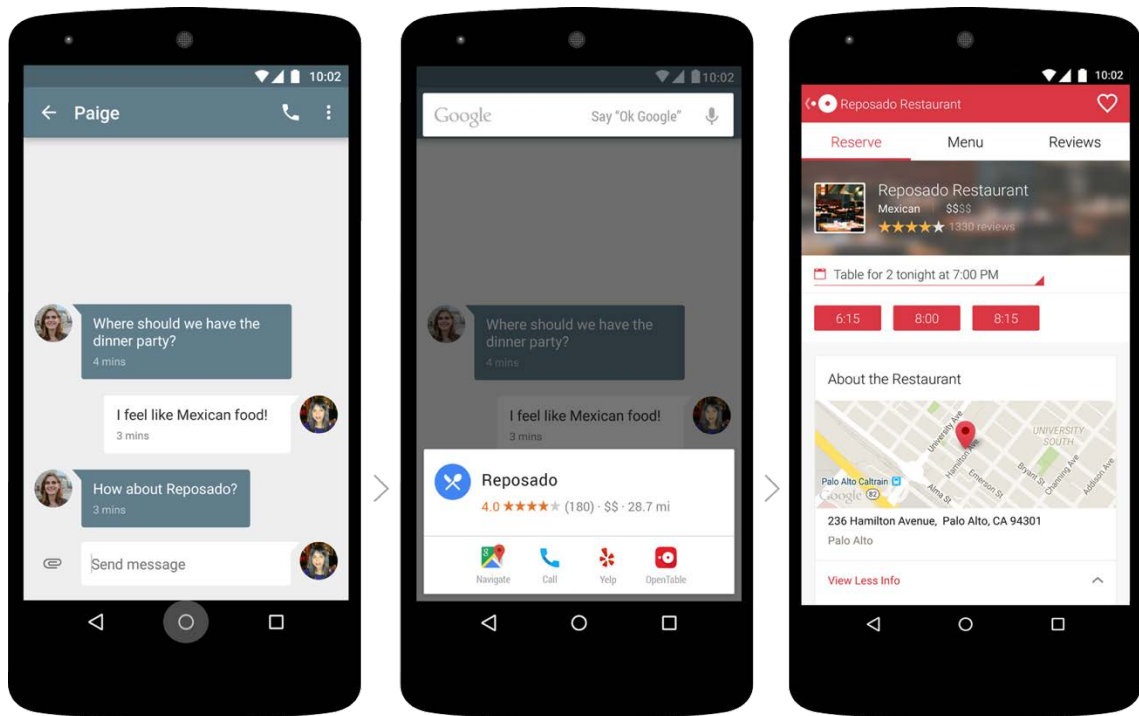


Figure 2.10: Android Marshmallow screen scan

2.3.3.4. Android Nougat

Android Nougat came up with very important updates such as the *split-screen support*, *quick notifications response* or *Google Assistant*. This was also the first version in which Google allowed users to participate in the naming decision.

In relation to the split-screen Nougat implemented a new feature that would allow users to manage two applications at the same time one each of them in one of the two sides in which the screen is divided. Moreover, this functionality included also a double tap in the recent button to switch between the last two used applications.

Nougat also modified the notifications management by including the possibility to answer messages without the need to open the application itself. In addition, each notification appeared one after the other without leaving any space between them as there was in previous versions.

Finally, this was the first version to include Google Assistant, one of the most powerful assistants of the market. It would launch by tapping the home button for a few seconds or even by just saying “Ok Google” and it could open apps, search information and also manage the user account.

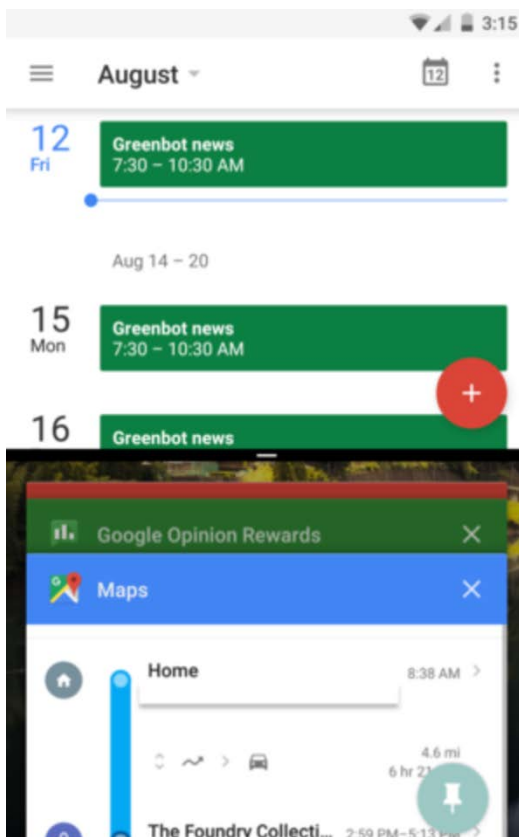


Figure 2.11: Android Nougat split screen

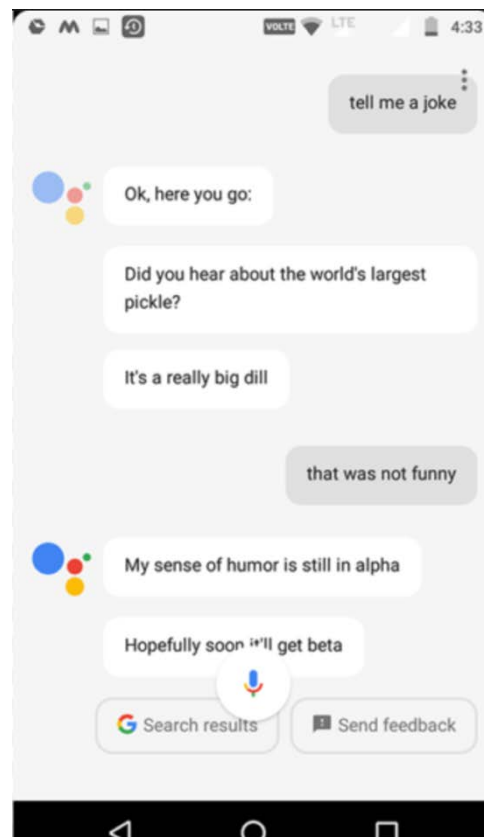


Figure 2.12: Android Nougat Google Assistant

2.3.3.5. Android Oreo

Android Oreo is, nowadays, the most Android version used in the world. It was released in 2017 and it was the second time Google made a partnership to name this version after Android KitKat. An important feature that was released at the same time as Android Oreo was *Google Play Protect*, which is an antivirus that scans the device

regularly in order to find malware that could be dangerous for users. Google Play Protect does not only search for malware in the installed applications, as it also prevents users from dangerous sites in Internet.

Notifications in Android Oreo were updated with new features such as giving the possibility to snooze notifications of a chosen application for a given time. In addition, notifications are ordered by importance for users, which means that notifications coming from applications that are more used by users, come first.

Moreover, to increase multitasking, Google introduced a picture-in-picture feature that would show in a small part of the screen opened applications such as Google Maps or the paid version of YouTube, while they were in other application.

To end up with Android Oreo, it is important to say that also a version for less powerful devices was launched with the name of Android Oreo Go Edition. This version allowed devices with less than 1 Gb of RAM to have a better user experience than with a normal version.

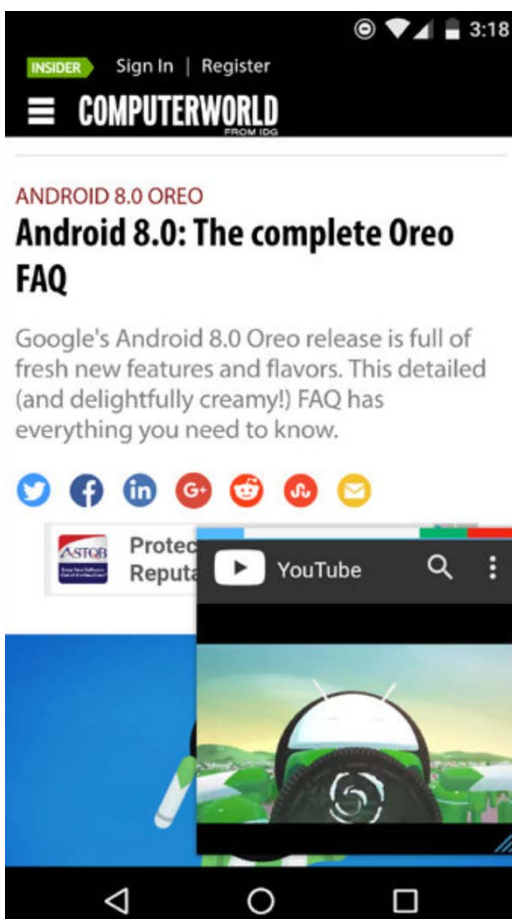


Figure 2.13: Android Oreo picture-in-picture

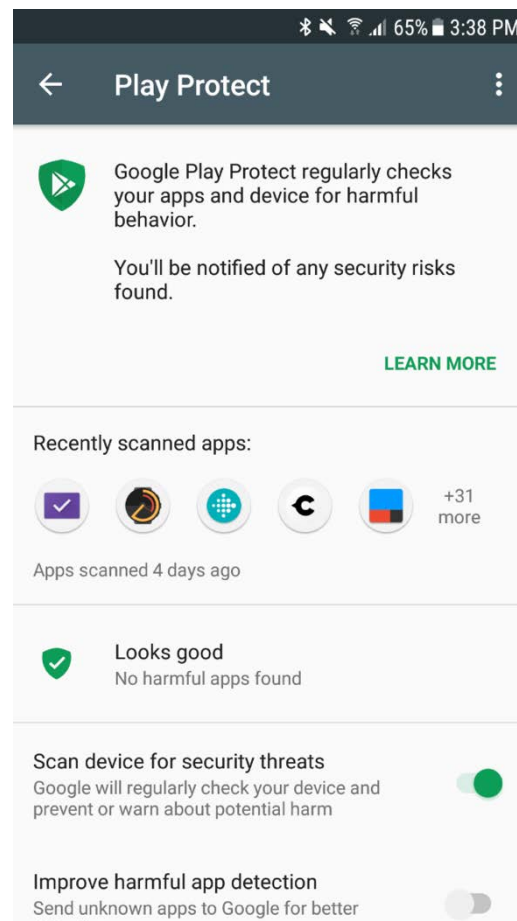


Figure 2.14: Android Oreo Google Play Protect

2.3.3.6. Android Pie

Android Pie is the newest Android version available nowadays but, as it is still very new, it is difficult to find phones running this operating system. In addition, this was the first version that was allowed to be run in non-Google devices.

This version comes with a new gesture-based navigation, which is the most important change in comparison with Android Oreo. This new gesture-based navigation starts with the removal of the typical three-button navigation that was present in all other previous versions and it is changed into a unique “pill” button that stays at the bottom of the screen. From this button, depending on the gesture done, users can open the application drawer, see the recent apps or start Google Assistant.

Another new feature of this version is the arrival of an application that monitors how the phone is being used through the day, the number of messages received, and the time spent in each application. Moreover, the adaptive battery improves the battery manager by prioritizing the power for those applications that are being used the most.

In addition to all these improvements, Google included a new capability to Google Assistant that consists in defining actions. These actions are defined following the “*trigger-action*” definition, which means that the user defines something to be done when some conditions are met.



Figure 2.15: Android Pie main screen without navigation buttons

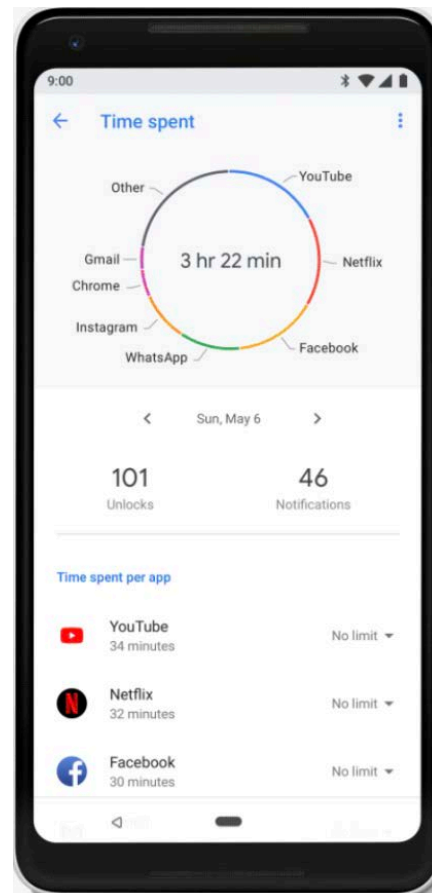


Figure 2.16: Android Pie applications monitorization

[***] Data used to write section Section 2.3.3 was obtained from [25] [26] [27] [28] [29] [30].

2.4. Resources needed by the system

In this part, some of the software resources that have been used for some components of the system will be described.

2.4.1. Text sentiment analysis and topic extraction

Sentiment analysis, also known as Opinion Mining, is a field in *Natural Language Processing* (NLP) that builds systems that try to identify and extract opinions from text. Additionally, to the opinion, they usually extract also the polarity, which expresses if the opinion is positive, neutral or negative, the subject and the opinion holder, which is the person or entity that expresses the opinion.

Moreover, sentiment analysis can be used to analyze both facts or opinions and classify sentences as objective (facts) or subjective (opinions).

The algorithms that can be used to develop sentiment analysis systems can be ruled-based, automatic or hybrid.

- **Ruled-based approach:** Some rules are defined in any scripting language with the objective of recognizing the subjectivity, polarity and the subject of the opinion.
- **Automatic approach:** Based on machine learning techniques. First the system is trained with a set of inputs and finally, after it has been trained, when a new input is tested, it is able to recognize all features thanks to the model obtained during training process.
- **Hybrid approach:** Combines ruled-based approach with the automatic one. These models are usually more precise than the other two.

The API used to perform the sentiment analysis in text messages sent from the Android application is one provided by *MeaningCloud*.

MeaningCloud is a Software as a Service product that enables users to embed text analytics and semantic processing in any application or system. It provides a cloud-based framework that makes the integration of semantic processing into any system something close to a “plug and play” experience. The most important functionalities that it provides are *topic extraction*, *text classification*, *sentiment analysis* and *text clustering*.

The sentiment analysis provided by MeaningCloud works by sending GET or POST messages to the API entry point. The request requires to include a set of parameters to make it valid. These parameters are the user key, the language of the text to be analyzed and the text to be analyzed. In addition, there are more optional parameters that can be included.

The response sent back to the client contains all the information related to the status of the request previously done and has a JSON or XML format. The response includes information such as the polarity, agreement between the sentiments detected, subjectivity or irony.

In addition to the sentiment analysis API, the topic extraction API mentioned above is also needed in order to extract relevant information from the messages sent from the Android devices and to classify the emergencies described in them.

Topic Extraction (Information Extraction) is the task of automatically extracting structured information from unstructured or semi-structured machine-readable documents.

MeaningCloud text extraction API is able to identify elements differenced in categories such as *entities*, *concepts*, *time expressions*, *quantity expressions*, *quotations*, etc. Moreover, it is possible to create custom dictionaries that contain new keywords for any of the entities previously mentioned.

The request done by a client, as in sentiment analysis, is done by means of POST or GET messages. The request requires to introduce the user key, language, the list of topics to be extracted and a text message. In addition, there are more parameters that can be included in the request. The requests made by the server of the system also includes the name of the custom dictionaries included to detect the type of emergency a user is talking about in the message.

The response received by the client contains different lists that contain the detected concepts, entities, time expressions and more. As the sentiment analysis responses, topic extraction responses are sent as JSON or XML files [37] [38] [39] [40].

2.4.2. Google Voice Recognizer

Google Voice Recognizer is a system capable of performing the process of voice recognition and is able to transform it into text (Speech to Text). The first time it appeared on Android devices was in version Android Donut (API 4).

Initially it was called Voice Action and gave the possibility to give commands to the Android device. It does not require Internet access to work and in 2014 it was merged

with Google Now. Nowadays it recognizes more than 60 different languages and can automatically detect up to 5 chosen languages in the same device when used.

As we can see in figure 2.17, the recognizer is activated when a specific button is pressed and ends when the user stops talking. In order to implement the Google Voice Recognizer in an application it is necessary to import *android.speech.RecognizerIntent* [41].

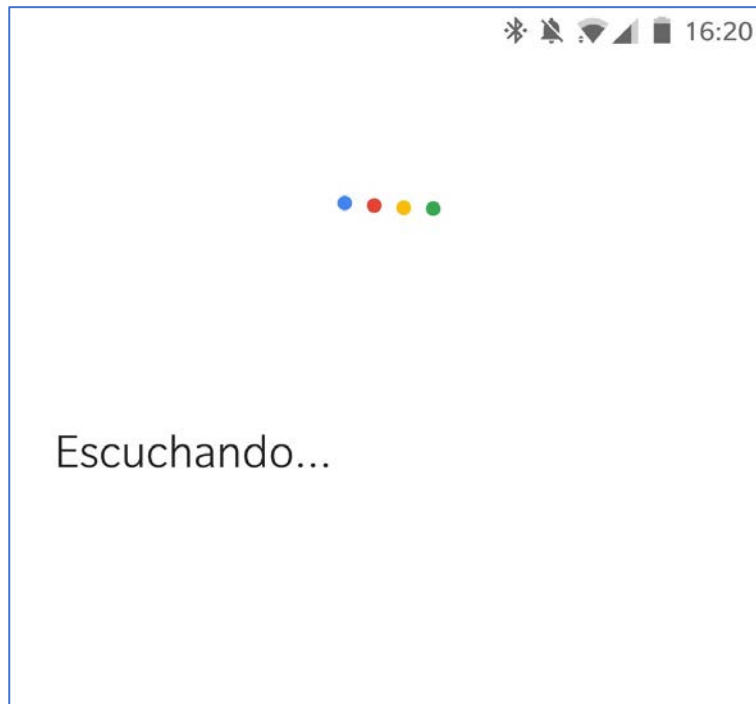


Figure 2.17: Google Voice Recognition

2.4.3. Python libraries for Twitter

There are multiple different libraries that support the standard twitter API and that are written for almost any programming language. In this part, only those that provide a python interface will be considered.

Twitter APIs are supported by nine different python libraries: *Python-twitter*, *Tweepy*, *TweetPony*, *Python Twitter Tools*, *Twitter-gobject*, *TwitterSearch*, *Twython*, *TwitterAPI* and *Birdy*. Any of these libraries permit to make use of any of the five different type of Twitter Web Service APIs:

- **Search Tweets API:** Returns a collection of relevant tweets matching a specified query.

- **Account Activity API:** Gives the possibility to subscribe to realtime activities related to a user account via webhooks. This means that every interaction that affects any account the user is subscribed to will be notified in realtime.
- **Ads API:** Provides programmatic access to advertising accounts. Partners can integrate their solutions to promote tweets or Twitter accounts, manage audiences and more.
- **Direct Message API:** Enables developers to build better-personalized customer experiences at scale as well as other innovative interactions
- **PowerTrack API:** Gives the possibility to filter the full Twitter firehose and only receive the data that is of the interest of the customer.

[] Data obtained from [42].*

2.5. Similar applications in the market

At the beginning of the process of thinking what our application should do, it is important to analyze other applications that are related to the same topic in order to develop something different from the others.

One benefit of making this kind of analysis is that it is possible to find new features that could fit in the application because they may not be implemented or even improve what is already working.

Two of the applications that will be explained are related to the notification of emergencies to emergency corps such as the police or hospitals and, another will be a navigation GPS application. In addition, also the system provided by the DGT in their web page to see the real time state of the roads of the country will be described.

2.5.1. Alertcops

Alertcops is an Android and iOS application with the capability to notify the police about any dangerous situation any user could have witnessed and receive support from them.

When users want to notify the police anything through this application, they start a conversation with a police officer from the nearest police station and they can tell them what has happened. This happens because the application sends the position of the user through GPS.

Having downloaded and tried the application, one of the drawbacks is the fixed types of emergencies a user can notify. If there is another type of emergency different to the ones the application supports, such as road emergencies, users will not be able to notify them correctly. On the other hand, the application developed for this Bachelor Thesis does support any type of emergency happening in both cities and roads, and it will notify every user immediately about them [31].



Figure 2.18: Alertcops application

2.5.2. My112

My112 is an application developed by Telefonica in cooperation with emergencies centers of some autonomous communities of Spain and is available for both Android and iOS. This application allows to communicate with emergencies centers in the same way as if 112 is called but with the singularity that also the location is sent to them. In addition, it isn't necessary to have a SIM card in the phone to establish the connection with the center.

Once the center receives a call from this application, they know exactly where the user is and can send support to them depending on the emergency for what they were called.

Actually, this application is only supported in some autonomous communities such as Castilla y León, Madrid, Islas Baleares, Cataluña, Cantabria and Melilla.

The problem with this application is that it is only supported in some autonomous communities which means that it will not work in many parts of the country. The Bachelor Thesis application implemented does not have any problem on where an emergency is notified so it provides assistance in any part of the country without exception [32] [33].

My112
con geolocalización
la app del CAT112

Trucada d'emergència
amb geolocalització

Emergència
Ajuts

descargate la app

Download on the App Store
ANDROID APP ON Google play

Una vez **descargada**, cuando tengas una emergencia, **llama** al 112 desde la aplicación

Lat: 41,613506
Long: 0,617683

El operador del 112 recibirá tu **llamada geolocalizada**

El 112 avisará a los **cuerpos de emergencias** necesarios

112 emergències

Emergency response vehicles: car, fire truck, helicopter, ambulance.

Figure 2.19: My112 application

2.5.3. Waze

Waze is a Google owned GPS navigation application that works similarly to any other GPS navigation system but with the particularity that users can notify others about events witnessed in real time.

Waze provides a set of possible events to notify such as traffic jams, accidents, police, hazards, gas prices, etc. and, after sending one of these possible events, it will set a marker in the map in the position where it was sent.

Waze is supported in both Android and iOS platforms.

Waze has the problem that, as it is a GPS navigation system, it only works in roads so any situation happening in any other place different from it will not be notified. The application developed, in contrary, provides support for cases happening in roads and outside them [34].

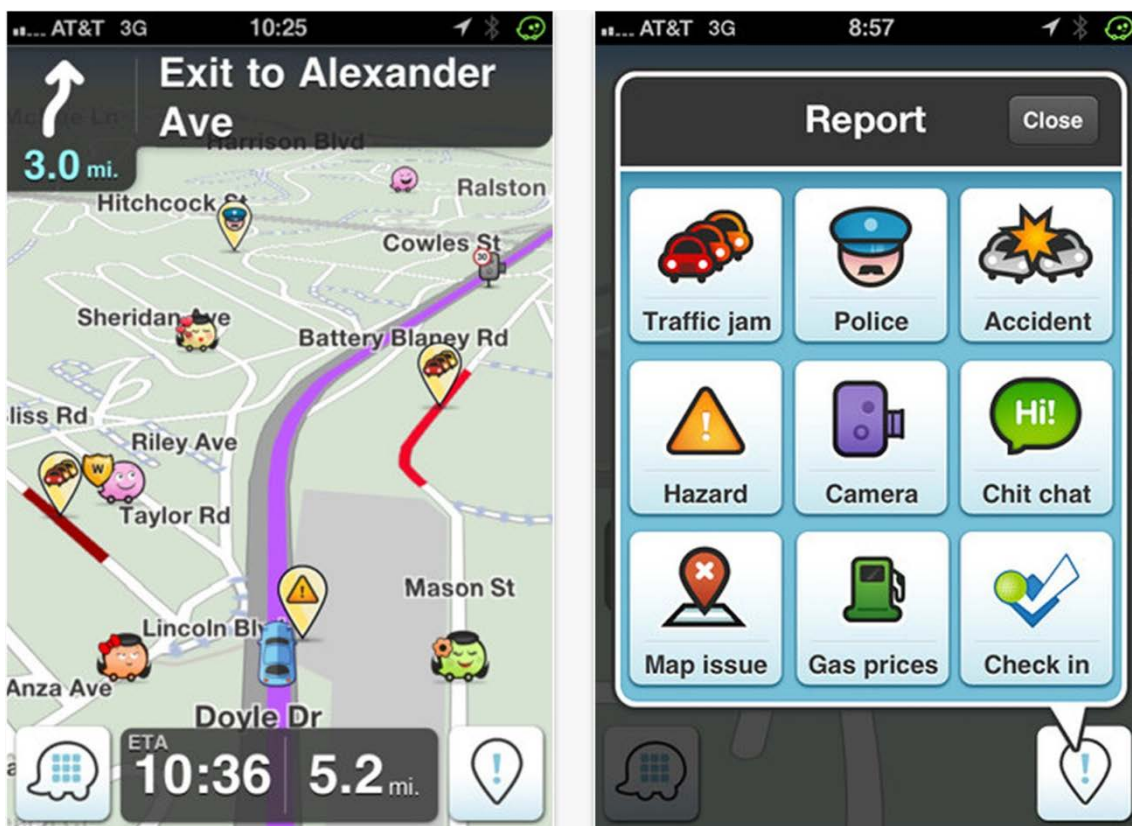


Figure 2.20: Waze Application

2.5.4. DGT traffic map

The DGT traffic map is a web page available in the DGT site that shows the alerts and traffic accidents that have happened in the roads of Spain.

There you can search by provinces or even cities and towns searching for relevant information related to their roads and highways. This information provided includes incidents that have happened there during the day, radars, cameras and more.

In the DGT web application, the situation in relation with the developed application is similar to the one happening with Waze, as they just show emergencies happening in roads. One point in favor of this application is that the emergencies come from a reliable source as they have been validated by the DGT but, as the developed application extracts from Twitter all these emergencies it continues providing a more complete support for users [36].

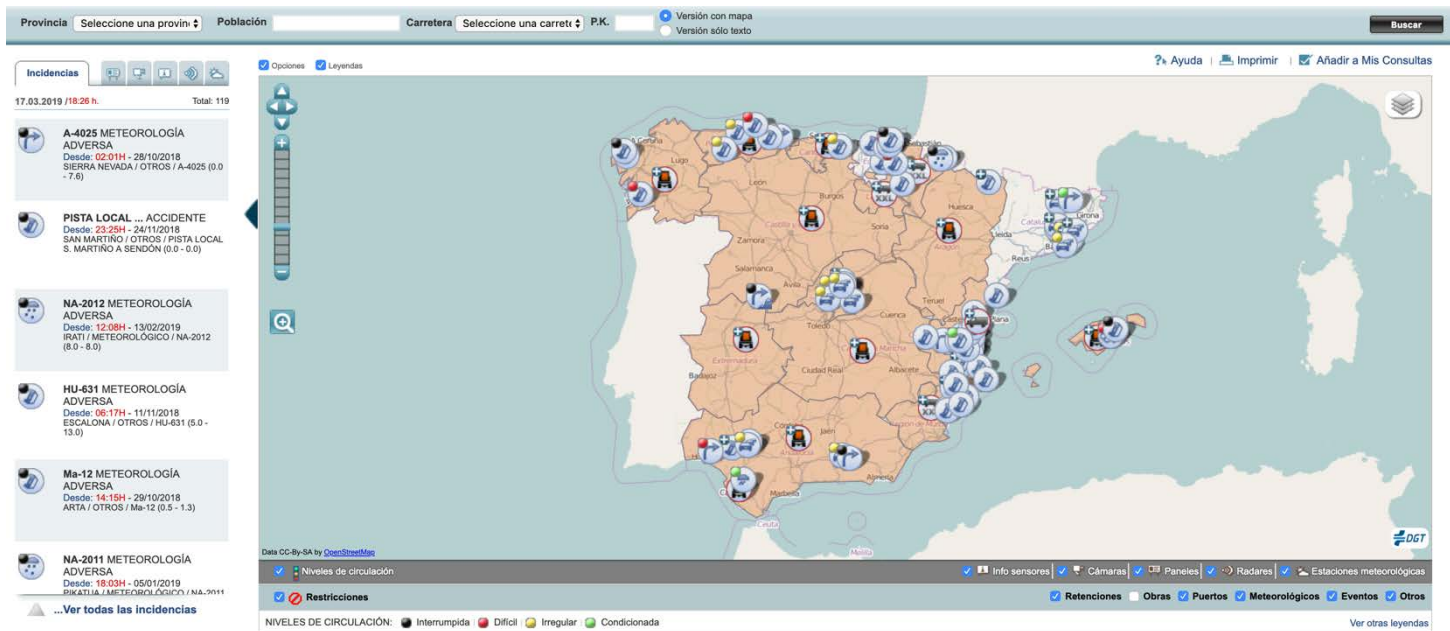


Figure 2.21: DGT traffic map

2.5.5. Similar applications summary

To end up with this chapter, in Table 2.2 a comparison between the capabilities of all the applications described before and the developed one is shown.

	Alertcops	My112	Waze	DGT web service	EmergenciesPal
Voice interaction	✗	✗	✗	✗	✓
Voice notifications	✗	✗	✗	✗	✓
Reaches all areas	✓	✗	✓	✓	✓
Supports road emergencies	✗	✓	✓	✓	✓
Supports urban emergencies	✓	✓	✗	✗	✓
Fixed types of emergencies	✓	✗	✓	✓	✗
Contains verified emergencies	✗	✗	✗	✓	✓

Table 2.2: Applications features comparison

In conclusion, it is possible to say that the developed application, EmergencyPal provides more complete list of features that make it better in comparison to the other applications.

CHAPTER 3. General description of the system

In this chapter, a general description of the system developed for the Final Degree Project will be presented.

Firstly, there is an overview of the whole system explaining how it works and a global schema of the different components. Secondly, all the different tools used to develop the system will be presented explaining what they are, why they have been used and their function in the system.

3.1. System overview

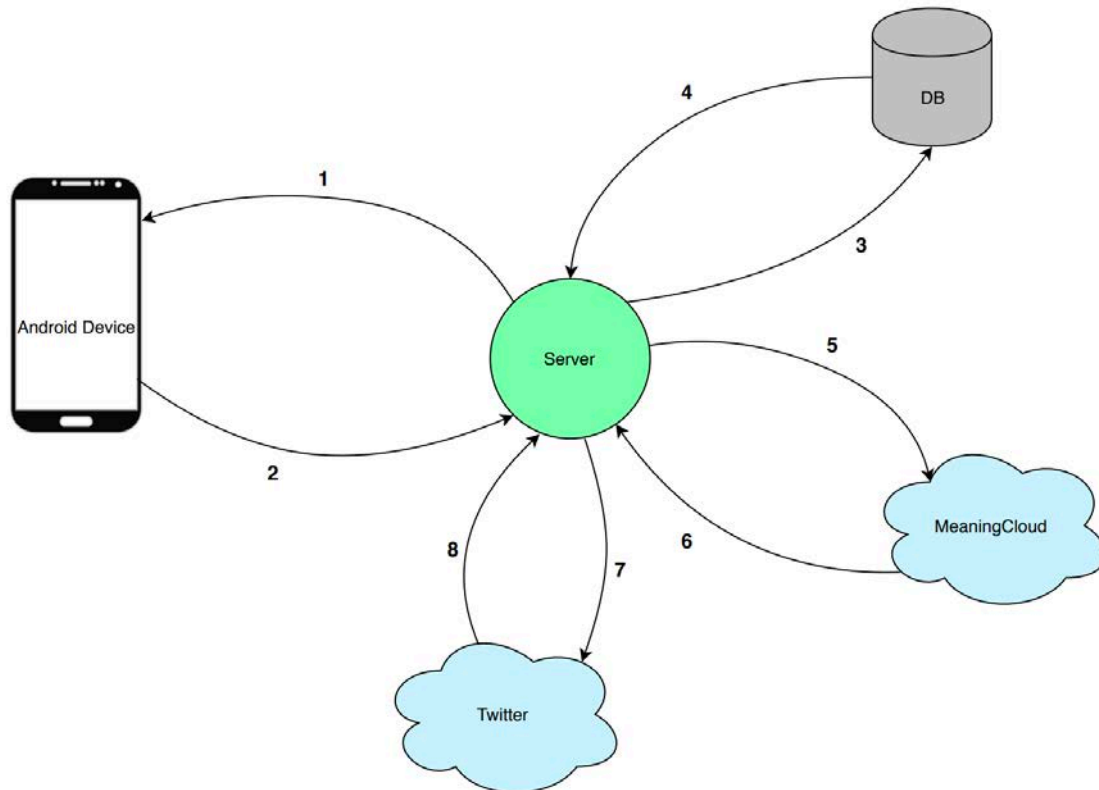
The system consists on an Android application that permits users to send emergencies by means of a dialog system that allows to send by voice a description of the emergency and, be notified of those happening near them thanks to the location provided by the phone GPS.

All emergencies that have been received by a user can be seen in a map, both altogether and individually. In addition, users can choose to activate or deactivate the feature that allows the phone to notify by voice new emergencies received.

Each time the Android application sends an emergency or requests nearby emergencies it connects to a server by means of “sockets”. Moreover, it also connects to the server in the login and register process and also in order to update user account settings and information.

The server has the responsibility to manage all requests done by Android clients, control data stored in the database, make use of the external MeaningCloud API, used to analyze the text with the objective of verifying that the information sent by a user is trustful and extract the type of emergency that has been sent, and make use of the Twitter API, used to get emergencies already verified by the police and include them in the database.

In Figure 3.1, there is a representation of how the system works as a whole. As it can be seen, the server is the center component in the communication because, as explained before, it is connected to every other part of the system. The main scenario would be that the device first sends to the server a request and then the server, depending on the request, accesses the database, makes use of the MeaningCloud API or both.



1. Receive account information / Receive emergencies
2. Login / Register / Send emergency / Update account information
3. Store account information / Update account information / Store emergency / Update emergency
4. Retrieve account information / Retrieve emergency
5. Text sentiment analysis / Text features extraction
6. Receive text sentiment analysis results / Receive text features extraction
7. Request tweets
8. Receive tweets

** The numbers do not mean that the execution is produced in that order. Numbers are just used to identify each call function.

Figure 3.1: Client-server system abstraction [51]

3.2. Technologies used

3.2.1. Android Studio

The development of the Android application has been done with the official Android development environment (IDE) **Android Studio**. Firstly, we will start explaining the installation process of the program and its necessary external tools. We will continue with the description of how an application is structured and, finally, some important tools used to develop some parts of the application.

3.2.1.1. Setup

First, the program must be installed and, to do so, we can download it from the official Android webpage <https://developer.android.com/studio>. In Figure 3.2, we can see the different platforms for which it can be downloaded, Windows (64 bits or 32 bits), Linux or Mac.

Platform	Android Studio package	Size	SHA-256 checksum
Windows (64-bit)	android-studio-ide-182.5314842-windows.exe Recommended	948 MB	a6da479931916e49cc7d077d1e62c3e46658710dfbc9bbc59087aaf8073e1450
	android-studio-ide-182.5314842-windows.zip No .exe installer	1008 MB	375bb4ca287cfef901c28439e51218f34122fca154b1c4025aa68ac636819b39
Windows (32-bit)	android-studio-ide-182.5314842-windows32.zip No .exe installer	1008 MB	aa76ea10b3418ba5d79fc8c51d6bbfdd857d466dc405cc402a2e8b029d80c53b
Mac (64-bit)	android-studio-ide-182.5314842-mac.dmg	997 MB	83efe75e80a9b754947092cbfec50e88e7fe0237926f9e44c34e1459b160f3a
Linux (64-bit)	android-studio-ide-182.5314842-linux.zip	1014 MB	8257d3eab61c3da088e26689888a13e53e210c109a4e775ed71158b4471bb06a

Figure 3.2: Android studio platform downloads

Once Android Studio is downloaded, we can continue to the installing process. First it must be taken into account the amount of space needed to install it and, even though the executable **.dmg** file is less than 1Gb, the real required space is 1.7Gb. This happens because during installation, there are more packages downloaded and installed that are needed by the Android SDK tool.

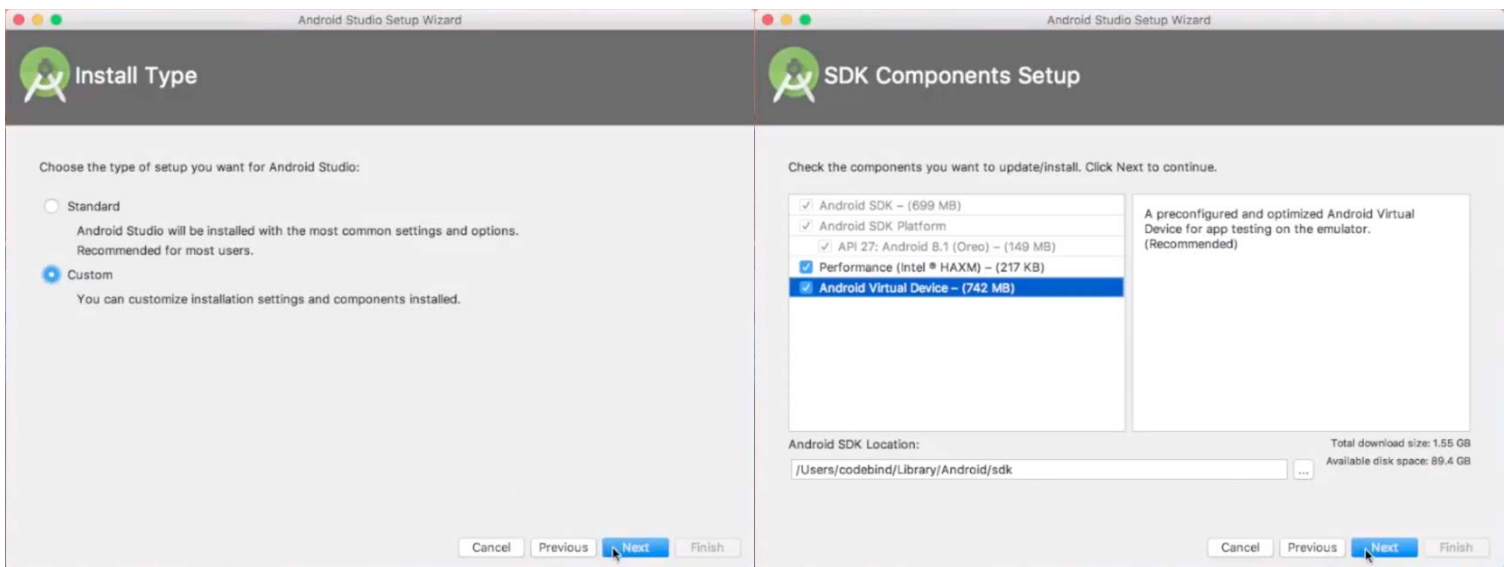


Figure 3.3: Android Studio installation assistant

In Figure 3.3, we can see the Android Studio installation assistant that will help us to configure it by following some steps. In the first half it is possible to see that there are two types of installation, standard and custom. The standard one provides all necessary features needed to start programming android applications in it, and the custom one allows to check the components that will be installed and change them if desired. The second half of figure 3.3 shows the components to be installed. As we chose the custom option, we can decide if we want to install an Android Virtual Device (AVD), which is very useful in order to develop an application.

3.2.1.2. Java Development Kit

Once Android Studio installation has finalized, we must install a Java Development Kit (JDK), which can be downloaded from the official Oracle website: <https://www.oracle.com/technetwork/java/javase/downloads/index.html>.

Figure 3.4 shows the possible platforms for which JDK 8 can be downloaded. The JDK version chosen is JDK 8 for Mac OS.

Java SE Development Kit 8u201		
You must accept the Oracle Binary Code License Agreement for Java SE to download this software.		
Thank you for accepting the Oracle Binary Code License Agreement for Java SE; you may now download this software.		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	72.98 MB	jdk-8u201-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.92 MB	jdk-8u201-linux-arm64-vfp-hflt.tar.gz
Linux x86	170.98 MB	jdk-8u201-linux-i586.rpm
Linux x86	185.77 MB	jdk-8u201-linux-i586.tar.gz
Linux x64	168.05 MB	jdk-8u201-linux-x64.rpm
Linux x64	182.93 MB	jdk-8u201-linux-x64.tar.gz
Mac OS X x64	245.92 MB	jdk-8u201-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	125.33 MB	jdk-8u201-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	88.31 MB	jdk-8u201-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	133.99 MB	jdk-8u201-solaris-x64.tar.Z
Solaris x64	92.16 MB	jdk-8u201-solaris-x64.tar.gz
Windows x86	197.66 MB	jdk-8u201-windows-i586.exe
Windows x64	207.46 MB	jdk-8u201-windows-x64.exe

Figure 3.4: JDK 8 platform downloads

In order to install JDK 8, we must follow the instructions given by the JDK installation assistant that can be seen in Figure 3.5. As when Android Studio was installed, the required space needed for installation is greater than the **.dmg** installation file. In this case, for installing JDK 8 we will need 587 MB of free space.

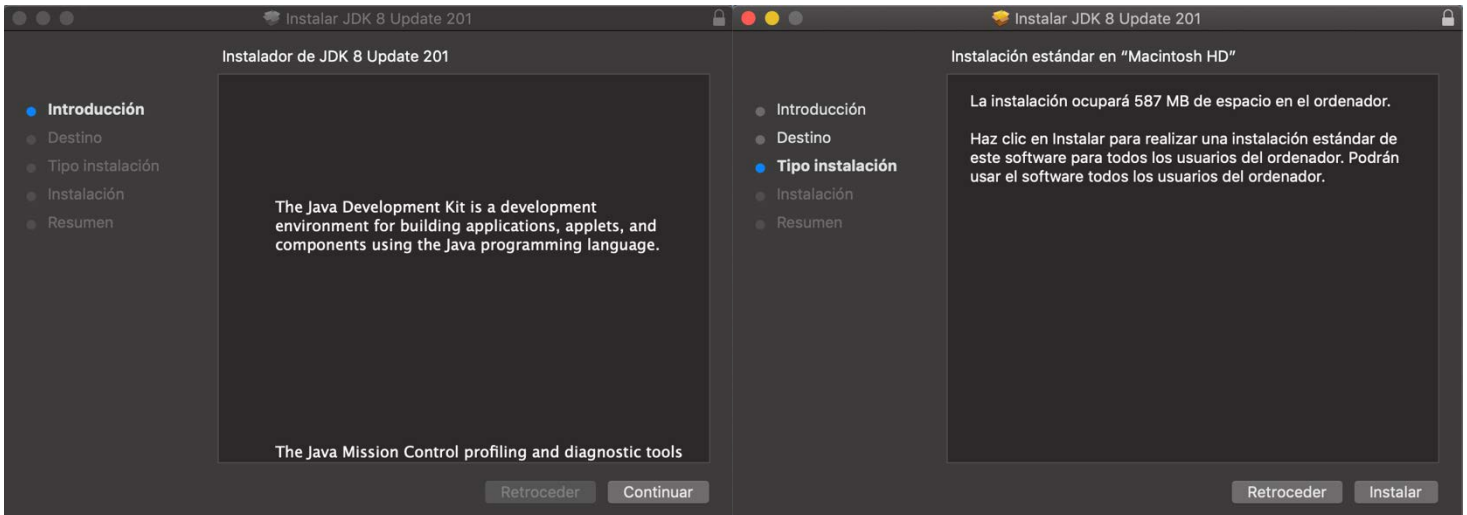


Figure 3.5: JDK 8 installation assistant

3.2.1.3. Software Development Kit

The Software Development Kit (SDK) are the set of tools that allow the creation of an application. Once Android Studio and the JDK have been installed properly, it is necessary to decide the version for which the application will be aimed. For this application, the Android version chosen is Android 8.0 (Oreo) so the necessary tools for this version have been downloaded from SDK Tools, which is already included in Android Studio IDE.

These tools downloaded along with an Android Virtual Device (AVD) provide the perfect environment to test the application.

3.2.1.4. Android Virtual Device

An Android Virtual Device (AVD) is an emulator of an Android device that simulates a real environment for a given Android version.

During the installation, as the chosen path was the custom one, it was possible to install an already designed AVD, but it is possible to install other AVDs with a set of properties desired by the user, such as the type of device (Smartphone, Tablet, Smartwatch or Android TV), dimensions or Android version.

The first step needed to create a new AVD is to open the AVD Manager. There it is possible to see all the already created virtual devices and it gives the possibility to create new ones. In Figure 3.6 there is the AVD Manager showing a virtual device along with its properties.

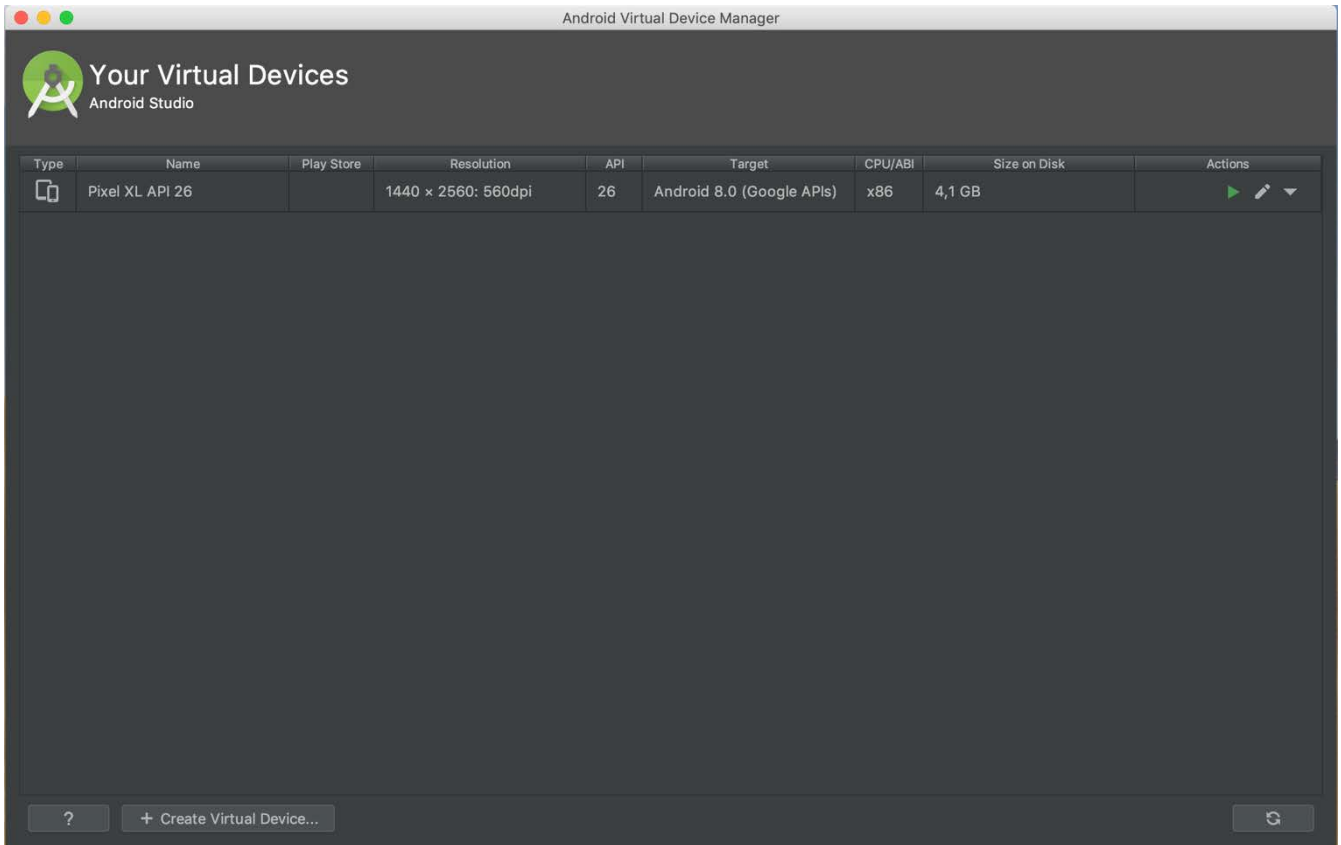


Figure 3.6: AVD Manager

After opening the AVD Manager and selecting the “Create Virtual Device” option, a new window showing a set of existing devices that can be created or giving the possibility to create a custom one. After the device is chosen or designed, the Android version is chosen. After these two steps the virtual device is created and ready to use. In Figure 3.7 all steps needed to create an AVD are shown.

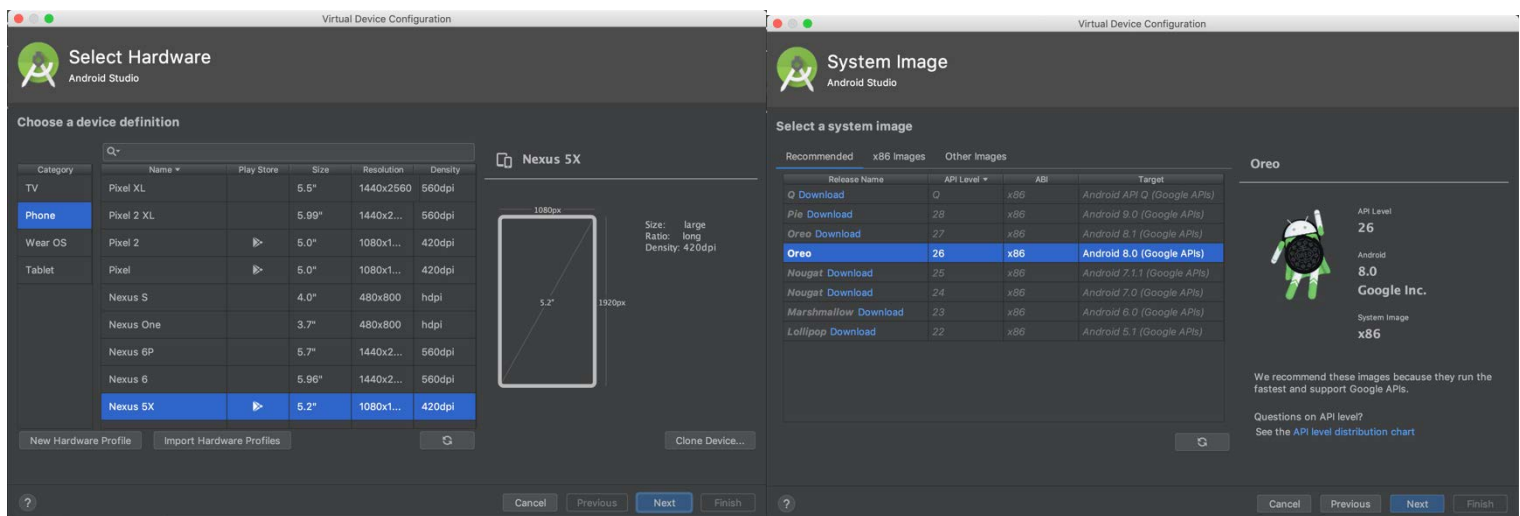


Figure 3.7: AVD creation process

3.2.1.5. Application structure

Android projects are structured in different modules, one of each of the devices for which an application is expected to be used. Each module is formed by three main folders:

- **Manifests:** Stores the Android Manifest, which is an *XML* file that contains information about the application. The information it includes is the structure of the application, the permissions required and metadata.
- **Java:** Contains all the Java files that form the source code of the application.
- **Res:** Contains all resources needed to develop the application such as the layout files for each of the activities of the application, images, strings, etc.

In addition to these folders that form the application, there are also *Gradle Scripts* that contain a set of files that permit the compilation of the application. The most important file we can find in this part is the *build.gradle*, that contains all the options and packages need for compilation [43].

Figure 3.8 shows the structure of the application developed for this Bachelor Thesis.

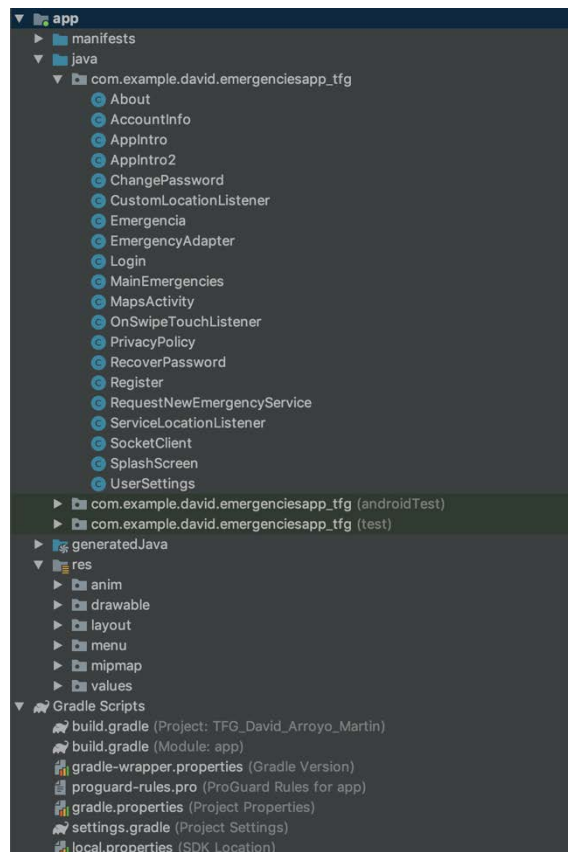


Figure 3.8: Bachelor Thesis application structure

3.2.1.6. Usage

Android Studio and the AVD have been used in order to develop the Android application all the different activities, services, classes, etc. needed to make the application work as expected.

The elements that form the application will be described in detail in chapter four.

3.2.2. MySQL

For the development of the Bachelor Thesis application, emergencies sent by users must be stored somewhere. For this reason, in order to store all this data MySQL was chosen to be the database for the server of the project.

MySQL is an open source relational database that allows to store data using Structured Query Language (SQL) that implements a client-server architecture. This means that there is a database server and an arbitrary number of applications (clients) that communicate with the server to query data, save changes, etc.

All data in a MySQL database is stored in tables and, as it is a relational database, all tables have a set of features such as a primary key, which is used to identify each of the entries of the table, or foreign keys, used to reference data from other tables.

Moreover, MySQL supports the ODBC (Open Database Connectivity) interface Connector/ODBC, which allows MySQL to be addressed by all the usual programming languages such as Python, Visual Basic, Java, etc.

In order to work in a more suitable way with the database created for the Bachelor Thesis application, MySQL Workbench has been used for the creation of the tables and, insertion and management of data [44].

3.2.2.1. Setup

In order to use MySQL, it is necessary to install it. The installation is done completely through a package installer downloaded from the official website of MySQL, <https://dev.mysql.com/downloads/mysql/>. For this project the *.dmg* version available for MacOS will be downloaded. Figure 3.9 shows the different versions available for download.

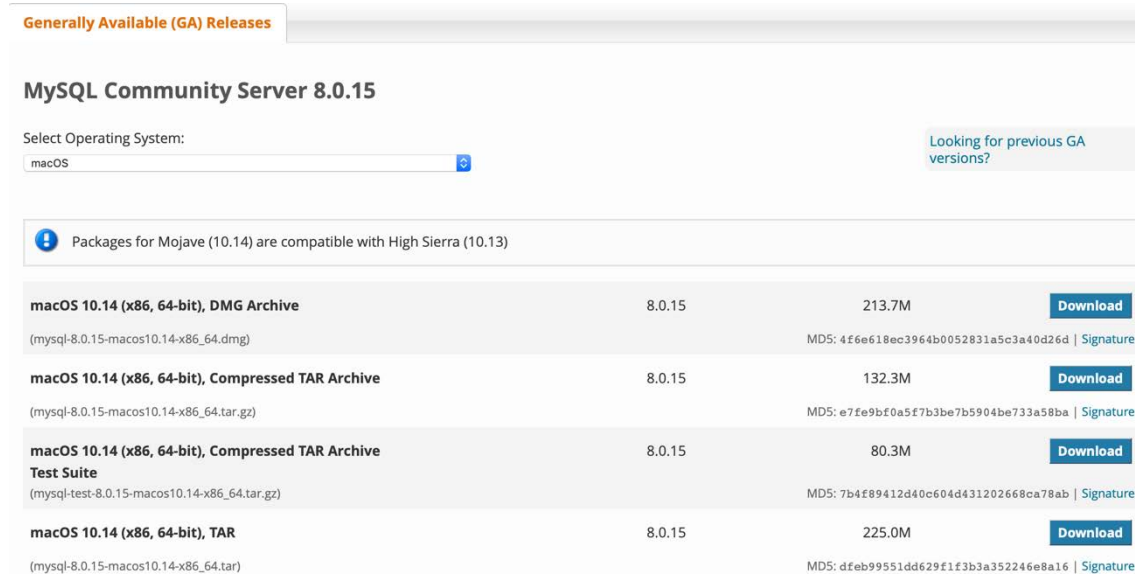


Figure 3.9: MySQL available installation packages

Once the package is downloaded, it must be executed and to perform the installation the steps proposed by the installer must be followed. Figure 3.10 shows the step of installation type. In this step, all options were selected.

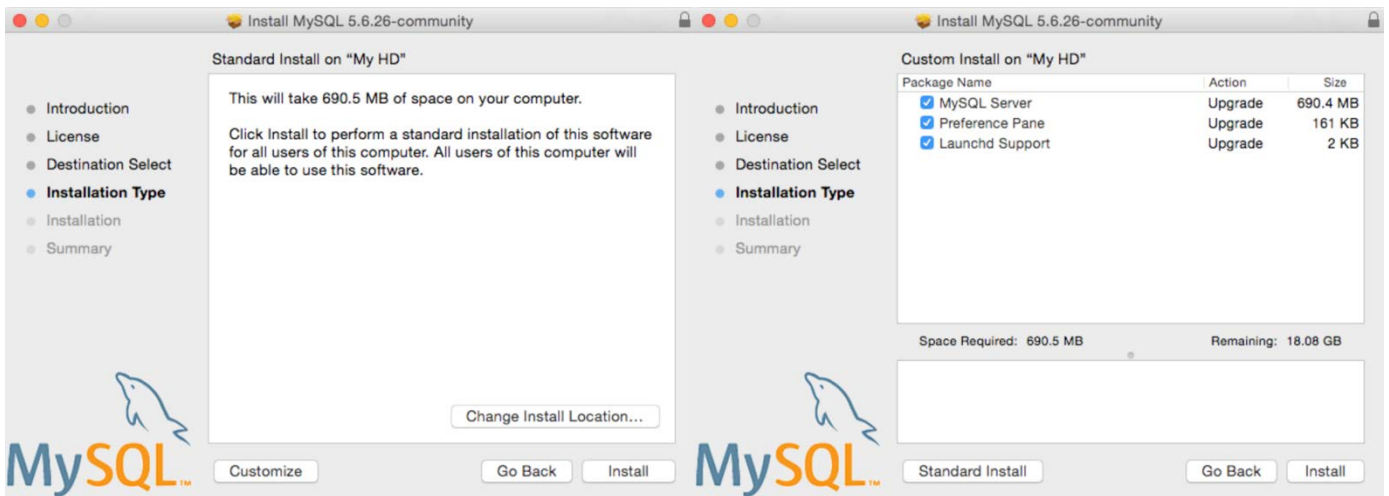


Figure 3.10: MySQL installation

Finally, at the end of the process, a new window appears showing the temporary root password that can be later changed to any other. This message is never shown again so it is important to keep the password provided secure until it is changed.

3.2.2.2. Usage - Database structure

The database created is used not only to store information related to the emergencies sent by users or extracted from the official Twitter account of the DGT, but to store also user accounts and information about all the roads of Spain. For this reason, the database is formed by four different tables: *User*, *Road_PK*, *Road_code_name* and *Emergency*.

- ***User***: This table is used to store all relevant information related to user accounts. The structure of this table is described in Table 3.1.
- ***Road_PK***: This table stores the location of all kilometric points (PK) belonging to each of the Spanish roads. The structure of this table is described in Table 3.2.
- ***Road_code_name***: This table stores the name of all the Spanish roads along with its road code. The structure of this table is described in Table 3.3.
- ***Emergency***: This table stores all information related to each of the emergencies sent by the users and of those extracted from the official Twitter account of the DGT. The structure of this table is described in Table 3.4.

Column name	Description
Username	String type variable that stores the unique username chosen by the user.
Password	String type variable that stores the password chosen by the user to identify its account.
First name	String type variable that stores the name of the user.
Surname	String type variable that stores the complete surname of the user.
Email	String type variable that stores the email chosen to connect the account.
Birthdate	String type variable that stores the birthdate of the user.
TalkEmergency	Boolean type variable that stores the current state of the option that gives the possibility to hear by voice new emergencies received.

Table 3.1: User table structure

Column name	Description
Latitude	Double type variable that stores the latitude of a kilometric point.
Longitude	Double type variable that stores the longitude of a kilometric point.

PK	Integer type variable that stores a kilometric point belonging to a road.
Road name	String type variable that stores the name of a road.

Table 3.2: Road_PK table structure

Column name	Description
Code	String type variable that stores the code associated to a road.
Name	String type variable that stores the name of a road.

Table 3.3: Road_code_name table structure

Column name	Description
Em_ID	Integer type variable that stores the identification number associated to an emergency.
Em_type	String type variable that stores the type of emergency received.
Latitude	Double type variable that stores the latitude of an emergency.
Longitude	Double type variable that stores the longitude of an emergency.
Road	String type variable that stores the road where the emergency has happened in case it is a road emergency. If it is not, its value is NULL.
KM	Integer type variable that stores the kilometer where the emergency has happened in case it is a road emergency. Otherwise, its value is NULL.
Direction	String type variable that stores the direction in which the emergency has happened in case it is a road emergency. Otherwise, its value is NULL.
Location	String type variable that stores the location where the emergency has happened in case it is an urban emergency. Otherwise, its value is NULL.
City	String type variable that stores the city where the emergency has happened in case it is an urban emergency. Otherwise, its value is NULL.
Date_time	String type variable that stores the time when the emergency has happened.
Description	String type variable that stores the description sent by the user.

Table 3.4: Emergency table structure

Data stored inside *Road_PK* and *Road_code_name* tables is used to determine the kilometric point in which a road emergency has happened and has been obtained from two main sources. The first source from which data was obtained was the National Center of Geographic Information (CNIG)¹ of Spain. In this web page, in the downloads center,

¹ <http://centrodedescargas.cnig.es/CentroDescargas/catalogo.do?Serie=REDTR>

it is possible to download all information related to any transport network of the country. Once the data of all the different provinces has been downloaded, they are all combined into one single file that contains each kilometric point of every road along with its latitude, longitude and more information, but for our database only the latitude and longitude of every kilometric point of each road is needed. All this information is stored in the table *Road_PK*.

Moreover, during the development of the application, I realized that when obtaining the position of an emergency, the inverse geolocator returned the roads sometimes with the name they have and other times with their code instead. Therefore, it was necessary to find a way to relate both the name and the code of each road so that if one did not match, the other would. Because of this, the new table *Road_code_name* was created, and its data was obtained from the Spanish “*Ministerio de Fomento*” website². In this web page, in the roads tab, it is possible to find a document with the name of “*Relación de Carreteras que forman la Red de Carreteras del Estado*” that contains both the name and code for each of the existing road in the Spanish territory. As the document is downloaded in a *.pdf* format and it is needed to be compatible with the database, an online tool to transform *PDF* files into *Excel*³ compatible files were used and from this new file, the data was included into the database.

3.2.3. SQLite

The Android application developed for the Bachelor Thesis has the main function of notifying users about emergency situations happening near them. For this reason, emergencies sent from the server to the application must be stored somewhere in the Android device. In order to store these emergencies, SQLite is used as the local database of the application.

SQLite is a software library that provides a Relational Database Management System (RDBMS). It stores data into a text file of the device and Android already has it installed on their devices.

SQLite is serverless, which means that it does not require a server to run. SQLite databases are integrated with the application that accesses the data inside it. Applications interact with the database by means of read and write operations. Figure 3.9 shows an illustration of this serverless architecture.

² <https://www.fomento.gob.es/carreteras/catalogo-y-evolucion-de-la-red-de-carreteras/catalogo-de-la-rce/catalogo-de-anos-anteriores/catalogo-de-la-rce-2015>

³ <https://www.pdfexcel.com>



Figure 3.11: SQLite architecture

Moreover, SQLite is self-contained, as it requires minimal work from the operating system or external libraries. This feature makes it suitable for most environments such as small devices like smartphones. In addition, as it is serverless, it does not require any configuration process similar to others like the required by MySQL [45].

3.2.3.1. Usage - database structure

The database implemented to store the emergencies received by the server is based on just one table that stores all information related to each of them. The structure of the table used to store this information is the same as the one that stores the data in the server database implemented with MySQL, which is represented in Table 3.4.

3.2.4. MeaningCloud

MeaningCloud, as described in point 2.4.1, is a Software as a Service product that enables users to embed text analytics and semantic processing in any application or system. It provides a cloud-based framework that makes the integration of semantic processing into any system something close to a “plug and play” experience. For the development of this application, both *Sentiment Analysis API* and *Topic Extraction API* have been used.

MeaningCloud provide all their APIs for free for every registered user. Free accounts are allowed to make a total 20.000 requests per month and no more than two requests per second. In addition to this free plan there are others with a higher number of requests allowed but that require a monthly subscription.

3.2.4.1. Setup

In order to use MeaningCloud APIs, it is necessary to create an account and make use of a unique key that is used to make all the calls made from any program that needs of any of the APIs MeaningCloud provides. In Figure 3.10, it is possible to see the

3.2.4.2. Usage

Firstly, the Sentiment Analysis API has been used to check the veracity of the messages sent by users from the Android application. The parameters considered to check it are the following ones:

- **Score tag:** Expresses the polarity, if found, of the text analyzed. The possible values returned are *strong positive (P+)*, *positive (P)*, *neutral (NEU)*, *negative (N)*, *strong negative (N+)* and *without sentiment (NONE)*. For our project, as the descriptions of the situations are expected to be negative, only messages with negative and strong negative scores will be considered.
- **Agreement:** Marks the agreement between the different sentiments detected in the text. The possible values returned are *AGREEMENT* or *DISAGREEMENT*. The value expected from the analyzed texts is *AGREEMENT*.
- **Confidence:** Represents the confidence associated with the sentiment analysis performed on the text. Returned value is an integer between 0 and 100 percent. The system will only consider those messages with a confidence higher than 85 percent.
- **Irony:** Represents the irony found when analyzing a text. Returned values can be *NONIRONIC* OR *IRONIC*. Messages with ironic marks will be dropped.

Once the text sent has passed this first analysis, it goes through the Topic Extraction API with the objective of finding the type of emergency that the user has described in its message. This API makes an analysis that finds elements organized in different categories as explained in Section 2.4.1.

As also stated in that part, it is possible to create custom dictionaries that are used to make topic extraction analysis more precise for the user needs. For this project a custom dictionary containing different emergencies was created. In order to create this dictionary, once the user has logged into its account, it must go to the customization tab and select the dictionaries option. In Figure 3.10, it is possible to see the dictionary created for this project with the name of “*Emergencias*”.

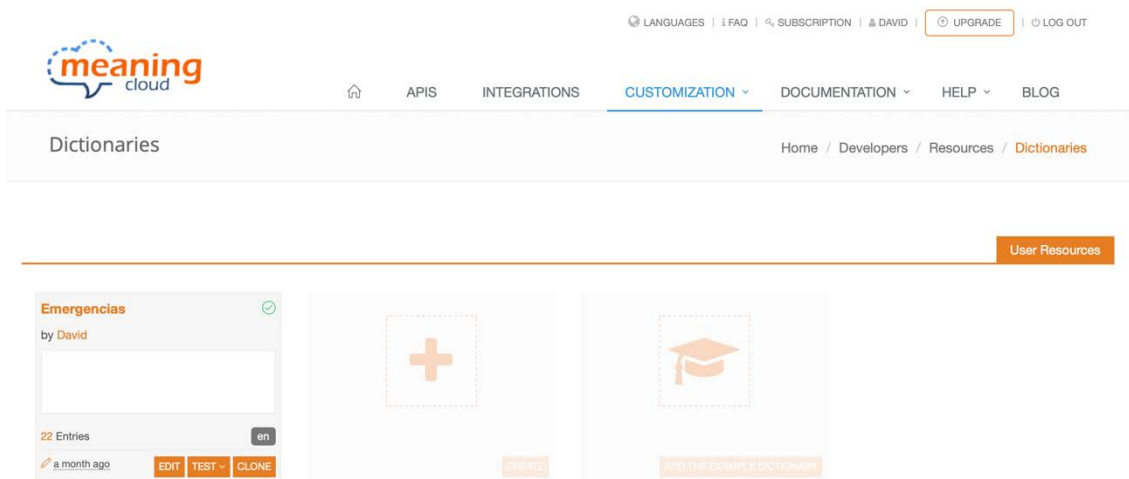


Figure 3.14: MeaningCloud custom dictionary creation and modification

For each of the dictionaries a user has, it is possible to include new entries recognizing them as a *concept* or an *entity* and, for each of those entries, include other aliases with which that entry could be identified. In Figure 3.11, it is possible to see an example of an entry of the “*Emergencias*” dictionary along with its aliases.

Figure 3.15: MeaningCloud dictionary entry

Thanks to this dictionary created, the system is able to identify correctly all emergency situations sent by any user through the Android application.

3.2.5. Python

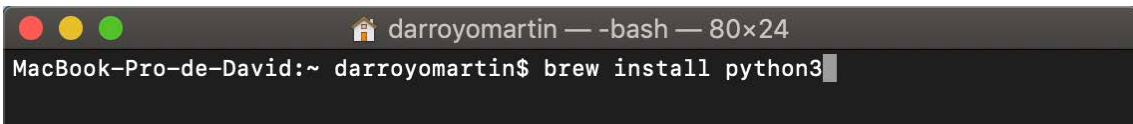
Python is an interpreted, high-level, general purpose programming language. High-level means that it is not necessary to bother about low-level details such as memory management and interpreted refers to the capability of this language to be run without the necessity of a compiler. This means that, unlike other programming languages like C or C++, Python does not need to be compiled into a binary file, it is run directly from the source code. Finally, Python is also an Object-Oriented Programming language (OOP) [46] [47].

3.2.5.1. Setup

Python 3 is the version that will be used to develop the part of the system programmed in Python programming language. The development of the system is done in a Mac computer, which already has a Python version installed by default, but it is version 2.7, which does not fulfil all the required functionalities for the system.

In order to install Python 3 in a Mac OS computer, different paths can be followed. The first one is to download a package installer from the official Python website <https://www.python.org/downloads/mac-osx/>, the second one is to make use of *pip* function and, finally, by means of *Homebrew*.

The method used is by making use of *Homebrew* function. Homebrew is an open-source software package management system that simplifies the installation of software on Apple Mac OS operating system and Linux. In Figure 3.16 the steps needed to install Python 3 are shown in the command line.



```

darroyomartin — -bash — 80x24
MacBook-Pro-de-David:~ darroyomartin$ brew install python3

```

Figure 3.16: Python 3 installation

3.2.5.2. Usage

Python has been for the developing of the server part of the system. The objective of the server developed with Python is to receive all incoming requests of the Android application users. The reason why Python has been used over other programming

languages is the huge variety of external libraries that are available for use. In addition, during the time spent in Computer Engineering I did not have any course in which Python was used, so it was quite of a challenge to be able to learn how to program in Python from zero.

Some of the external libraries used are:

- **Geopy:** Geopy is a python client for several popular geocoding services. The services used by the server side are *geodesic*, which is used to calculate the distances between two points given in coordinates, and *Nominatim*, service that can return the coordinates for a given address (Geocode) or can return the address of a given pair of coordinates (Reverse geocode).

Geopy is installed by writing the command *pip install geopy* in the terminal of the computer.

- **MySQLdb:** MySQLdb provides an interface that allows the communication of a Python program with MySQL Server databases.

MySQLdb is installed by writing the command *pip install MySQL-python* in the terminal of the computer.

- **Python-twitter:** Python-twitter provides an interface that makes possible the usage of the Twitter API.

Python-twitter is installed by writing the command *pip install Python-twitter* in the terminal of the computer

3.2.6. Twitter API

Twitter is an online news and social media networking site where people communicate in short messages called *tweets*. The process of posting tweets is called *tweeting*, and anyone that follows your user account can see those tweets.

In order to use Twitter API is necessary to have a Twitter account from which some credentials will be obtained in the Twitter Developer section. In addition, it is important to know that the API allows a maximum of fifteen calls every fifteen minutes in their free version and a maximum of 180 every fifteen minutes for the paid version.

3.2.6.1. Setup

Firstly, in order to make use of the Twitter API it is necessary to login into the Twitter Developer site of Twitter. The login credentials used must be the same as the ones of the developer Twitter account.

Once the login has been successfully completed, the next step is to go to the “Create an app” section. Here it is possible obtain credentials to be used by other applications developed. Figure 3.17 shows the creation of a new application process.

App details
The following app details will be visible to app users and are required to generate the API keys needed to authenticate Twitter developer products.

App name (required) ?

 Maximum characters: **32**

Application description (required)
Share a description of your app. This description will be visible to users so this is a good place to tell them what your app does.

 Between 10 and 200 characters

Website URL (required) ?

Allow this application to be used to sign in with Twitter [Learn more](#)
 Enable Sign in with Twitter

Callback URLs ?
OAuth 1.0a applications should specify their `oauth_callback` URL on the request token step, which must match the URLs provided here. To restrict your application from using callbacks, leave these blank.
 ×
 + Add another

token step, which must match the URLs provided here. To restrict your application from using callbacks, leave these blank.
 ×
 + Add another

Terms of Service URL ?

Privacy policy URL ?

Organization name ?

Organization website URL

Tell us how this app will be used (required)
This field is only visible to Twitter employees. Help us understand how your app will be used. What will it enable you and your customers to do?

 Minimum characters: **100**

Figure 3.17: Twitter Developer new application creation

Once the new application has been created, it will be included into the user applications and it will have its own credentials that will allow its usage. In Figure 3.18 the tab of the application where the keys and tokens are available is shown. This are the

credentials that will be used in order to initialize the API in the program in which it will be used.

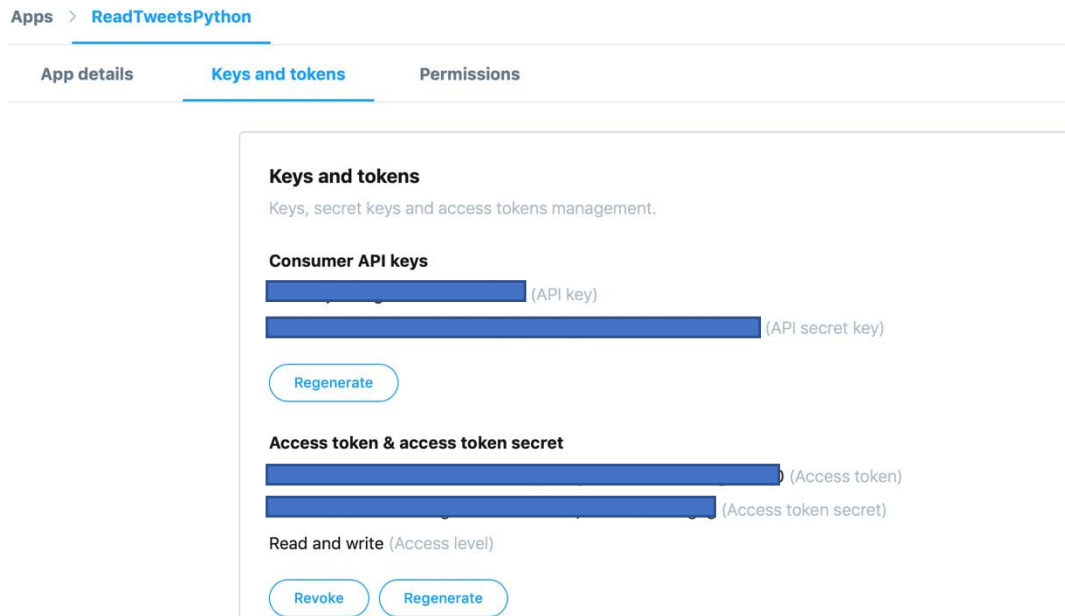


Figure 3.18: Twitter API keys

3.2.6.2. Usage

The Twitter API has been used with the objective of completing with verified information those emergencies that could happen in the roads and highways. The impossibility to send emergencies when being driving alone brought to the idea of using the emergencies sent by the DGT to keep users aware of every emergency situation.

This idea came thanks to the knowledge of the existence of a web application in which the DGT posts every situation happening in roads and highways and that was previously described in Section 2.5. The problem was to find where to extract that data and include it into our database. After some investigating process, we discovered that all those emergencies were also being posted in their Twitter account by means of tweets, so we decided to extract them from there.

In order to extract the emergencies from Twitter, a Python program that makes use of the API `getUserTimeline()` function has been developed. The function returns all the tweets of a given user following some certain parameters introduced in the function call.

The program checks after a certain time for new tweets posted in the DGT Twitter account and, if there is any, they are parsed and included into the database described in Section 3.2.2.

3.2.7. Firebase API

Firebase is technology that allows to make web applications without the need of a server-side and supports web, iOS, OS X and Android clients. Firebase works in a way in which developers does not need to think how information and data is managed and stored, it will do it for them. Moreover, Firebase is not only used as a database, it can also be used for analytics, remote configurations, authentication and more.

3.2.7.1. Setup

Firstly, it is necessary to create a new project in the Firebase Console. In Figure 3.19 it is possible to see the creation of a new project.

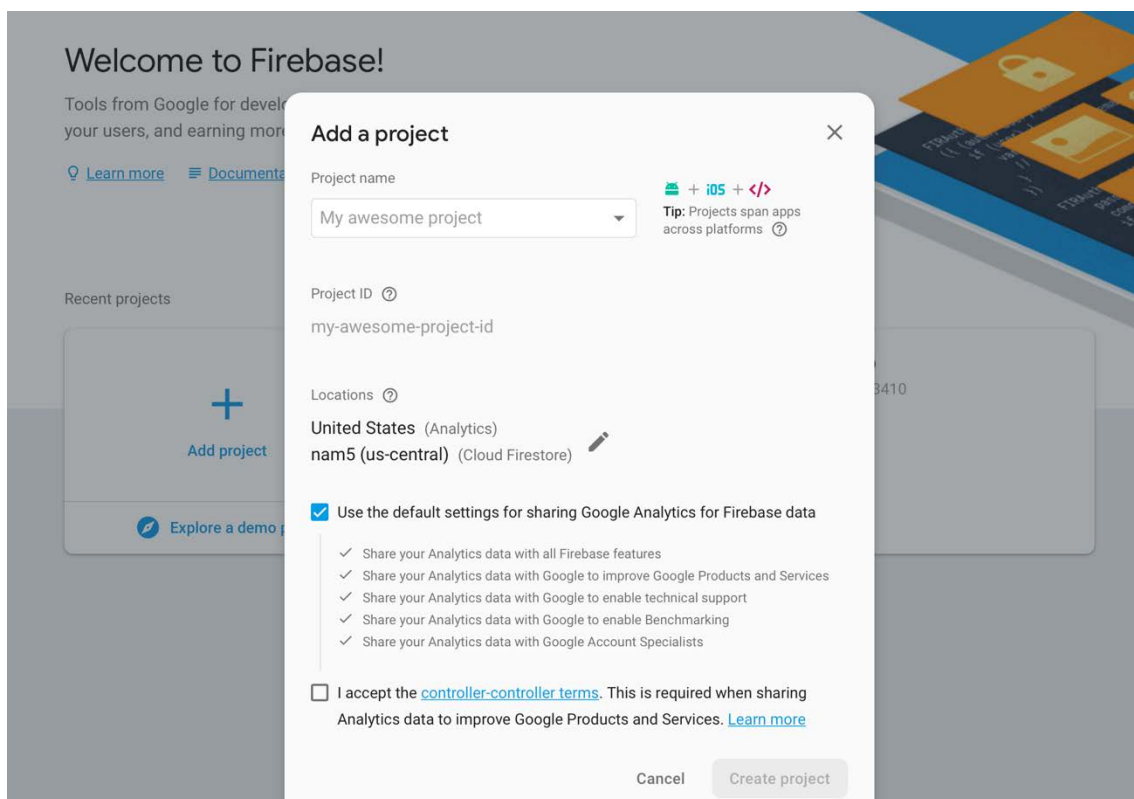


Figure 3.19: New Firebase project creation

Once the project has been created and you are inside it, it is time to create an application. To create the application, it is necessary to include the package name of the

Android application being developed, *com.example.david.emergenciesapp_tfg* in my case, along with a SHA-1 certificate that is obtained by means of *Keytool*.

After having created the application, a **JSON** file is downloaded and it must be included into the Android app folder. In addition, it is necessary to include the dependencies shown in Figure 3.20 in the *build.gradle* file in order to be able to make use of the Firebase application API created in the previous step.

```
classpath 'com.google.gms:google-services:4.2.0'  
implementation 'com.google.firebase:firebase-auth:16.1.0'  
implementation 'com.google.android.gms:play-services-auth:16.0.1'
```

Figure 3.20: Build.gradle lines inclusion for Firebase authentication

3.2.7.2. Usage

The Firebase API is used with the objective of allowing Google sign in authentication. Thanks to the authentication API of Firebase, users can allow to sign in into the Android application with Google instead of with an account created in the application.

CHAPTER 4. Detailed description of the application modules

In this chapter, all the different modules that form the Android application EmergencyPal will be analyzed. For each of them the functionality and use cases, if any, will be described. After all modules have been described, a service used for emergencies updates will be also be explained. The chapter ends with a description of how the communication and exchange of data between the application and the server is done.

4.1. Splashscreen

The Splashscreen is the first interface that the user sees when opening the application. It shows the application logo during 3 seconds before the next interface is shown.

The next interface shown can be the instructions of use, which appears only the first time the application is run, the login screen, if there is not any user already logged into the application, or the main screen, if there is a user already logged.

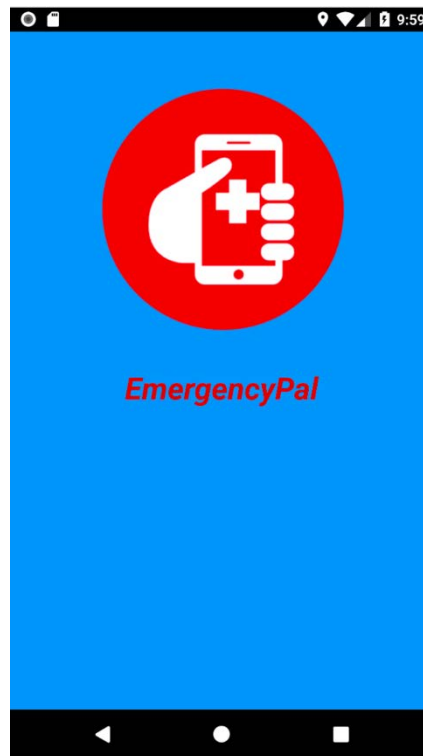


Figure 4.1: Splashscreen

4.2. Instructions of use

This interface shows a collection of slides showing an initial description of the application for new users.

The format followed for these slides come from an already developed library⁴ that simply allows the developer to include a title, description and image to any of them.

Figure 4.2 shows the three slides shown for new users when the application is run for the first time.

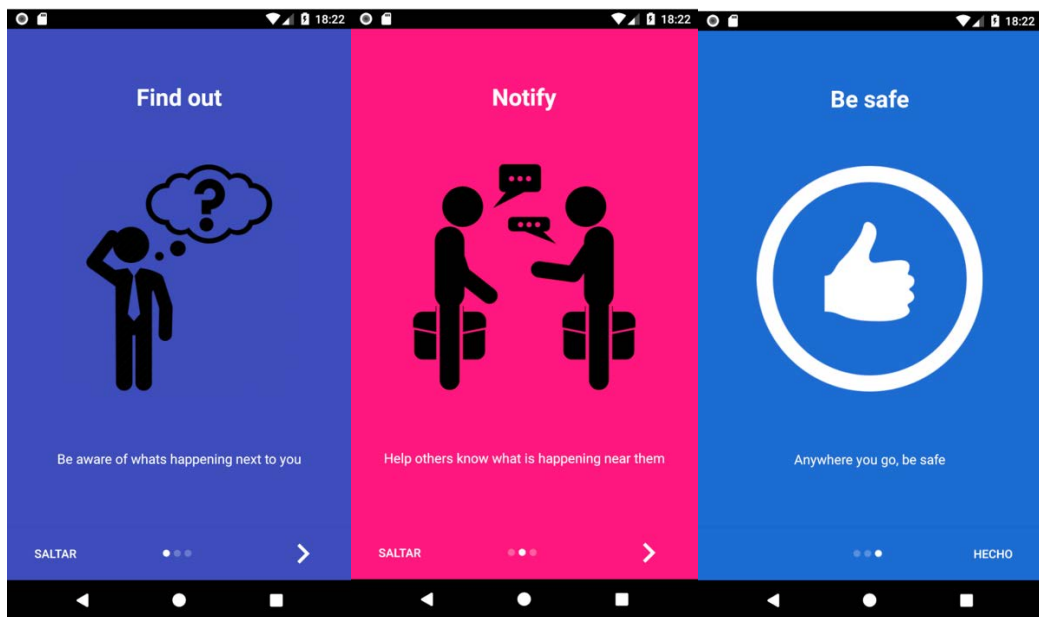


Figure 4.2: Application presentation slides

⁴ <https://github.com/AppIntro/AppIntro>

4.3. Login

The login module gives users the possibility of signing in into their accounts by means of accounts created in the application or by Google accounts. This module is shown only in the case when there is not any user that had previously logged into the application and that has not logged out.

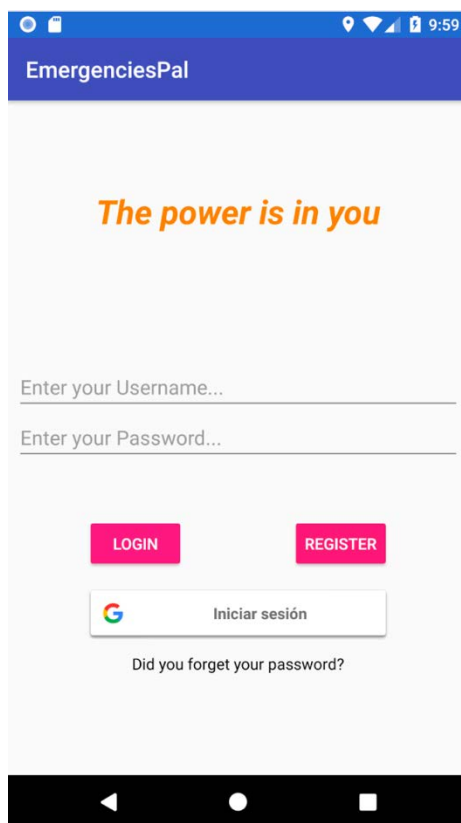


Figure 4.3: Login interface

Both input text spaces are used to introduce the username and password of an account. This account must have been created by means of the registration process that will be described in Section 4.4. On the other hand, it is also possible to sign in using a Google account.

By clicking in the “Login” button, both input text spaces will be checked firstly to not be empty, otherwise a message will be shown to the user telling that they must be filled. If the introduced values are valid, which means that there is a username-password pair existing in the users table of the database, the screen will change, and the main screen interface will be shown. If the username introduced does not match any existing user, the application will show a message notifying the user about this problem, and finally, if the password introduced does not match the given username, another message will be shown to the user.

The “Register” button is used to change from this interface to the one showing the registration steps necessary to create a new account. In addition, by clicking in “Did you forget your password?”, the application will show the password recovery interface.

Finally, when clicking in the Google authentication button, a popup will be shown with all the different Google accounts that have been used in the device and that can be used to authenticate the application. In figure 4.4, the Google accounts popup is shown.

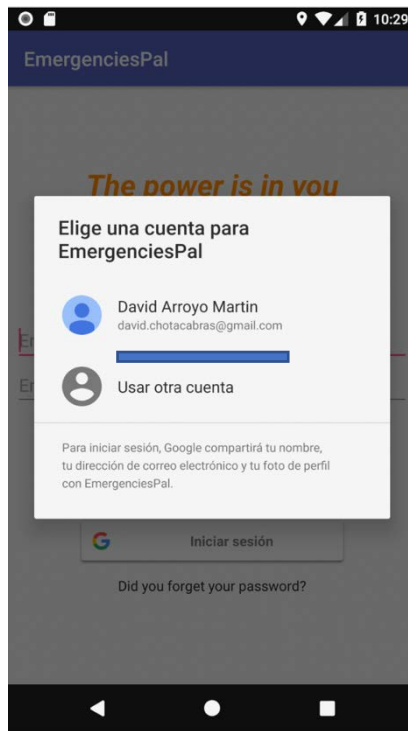


Figure 4.4: Google accounts login

4.3.1. Use cases

In the login module, there are five main use cases, a successful login, an unsuccessful login, caused by any of the previous described cases, successful Google login, opening registration activity and opening the password recovery activity.

App: Load login interface. User: Introduce username and password. User: Select “Login” button. App: Verify introduced username and password. App: Show main screen interface.

Figure 4.5: Login successful

App: Load login interface.
User: Introduce username and password.
User: Select “Login” button.
App: Verify introduced username and password.
App: Show message with the cause of the unsuccessful login.

Figure 4.6: Login unsuccessful

App: Load login interface.
User: Select “Google login” button.
App: Show possible Google accounts to be used.
User: Select a Google account
App: Show next interface

Figure 4.7: Google login successful

App: Load login interface.
User: Select “Register” button.
App: Show new account creation interface.

Figure 4.8: Recover the password selection

App: Load login interface.
User: Select “Did you forge the password?” text.
App: Show password recover interface.

Figure 4.9: Recover the password selection

4.4. Register

The register screen allows new users to create their own account if they don't want to make use of the Google login functionality. In Figure 4.10, the set of fields necessary to include an account are shown.

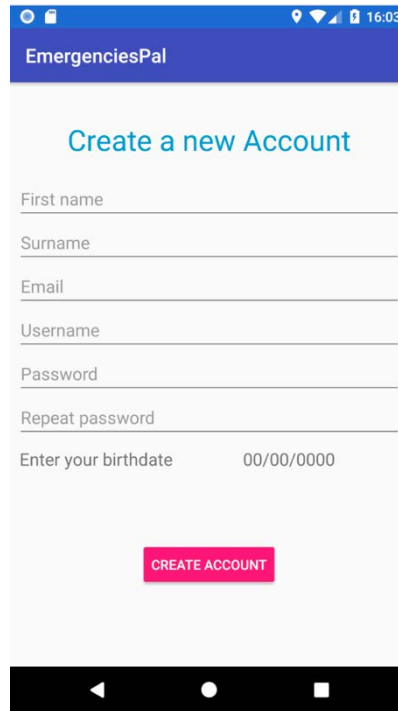


Figure 4.10: Register interface

All shown spaces must be filled with information, otherwise a message will appear in the screen to notify the user that something is left. In the case a user tries to create an account with an already registered username, the application will notify this, and the user will need to select another valid username. In addition, if both passwords are not the same, the application will also notify the user. If the process is successful, the application will directly show the main screen interface and the account information will be stored in the server database.

4.4.1. Use cases

In the registration module there are only to use cases, create a new account or not.

App: Load registration interface.
User: Complete all spaces
User: Press “Create account” button.
App: Verify introduced username.
App: Show main screen interface.

Figure 4.11: Successful registration

App: Load registration interface.
User: Complete all spaces
User: Press “Create account” button.
App: Verify introduced username.
App: Show message with the cause of the unsuccessful registration.

Figure 4.12: Unsuccessful registration

4.5. Password recovery

The account password recover interface permits any user to change their password if they have forgotten their previous one. To ensure that the user changing the password is the correct one, both the username and email address need to be included in order to verify the truthfulness of the user introducing the new password. In figure 4.13, the necessary information to be completed is shown and, as in the login and account creation, all spaces must be completed.

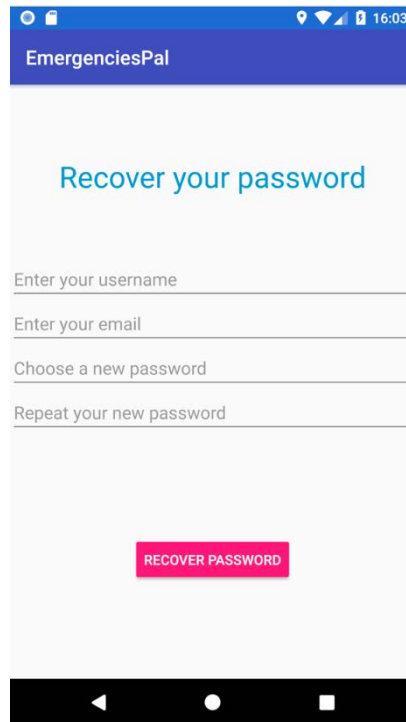
The image is a screenshot of a mobile application interface for password recovery. At the top, there is a blue header bar with the text "EmergenciesPal" in white. Below the header, the main content area has a light gray background. The title "Recover your password" is centered in a blue font. Below the title, there are four input fields, each with a light gray border and a light gray placeholder text: "Enter your username", "Enter your email", "Choose a new password", and "Repeat your new password". At the bottom of the form area, there is a pink rectangular button with the text "RECOVER PASSWORD" in white. The bottom of the screen shows the standard Android navigation bar with a back arrow, a home circle, and a recent apps square.

Figure 4.13: Account password recover

If the pair username-email does not match any existing account in the server database, the application will not update any account password with the new one and will show a message showing that the username or email are not correct. In addition, if the new passwords introduced are not the same, the application will show a message notifying it.

4.5.1. Use cases

In this module, there are two use cases, recover correctly the user password by setting a new one or not.

App: Load password recovery interface.
User: Complete all spaces
User: Press “Recover account” button.
App: Verify introduced pair username-email.
App: Show message notifying successful update.

Figure 4.14: Account password recovered successfully

App: Load password recovery interface.
User: Complete all spaces
User: Press “Recover account” button.
App: Verify introduced pair username-email.
App: Show message notifying unsuccessful update.

Figure 4.15: Account password not recovered

4.6. Main Screen

This module can be considered as the most important visible part of the application, as it is the place where the user will interact most of the time. When the application is run for the first time, in the moment this module is opened, the first thing users will see is a pop-up message asking for permissions to access the device location as it can be seen in the first half of Figure 4.16. This permission is required for the application to work properly as it would be impossible to know where the device is if it is not accepted. Without knowing the location of the device, it would be impossible to notify emergencies near them.

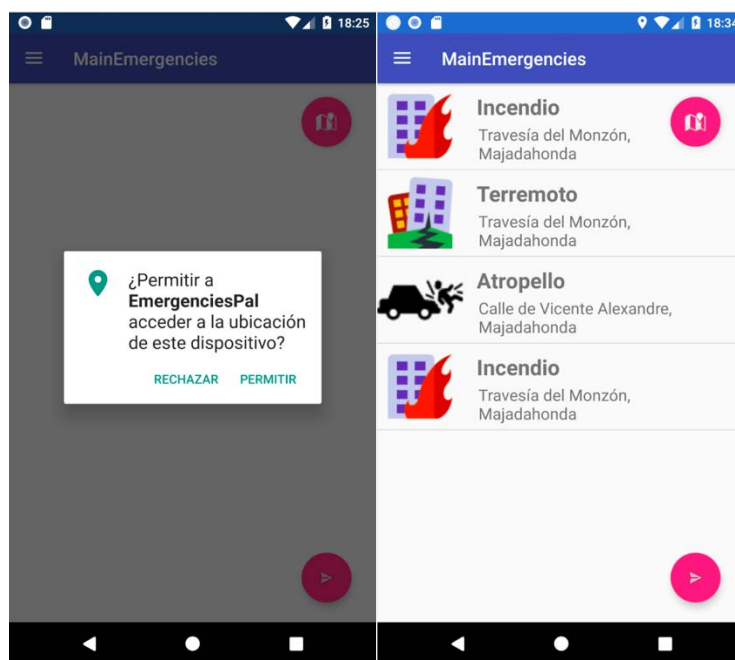


Figure 4.16: Main screen interface

The second half of Figure 4.16 shows a normal view of the interface with a set of emergencies that are near the device. The possible actions allowed by the interface are the following:

- Notify an emergency.
- See emergencies location in a map by pressing over the top-right button.
- Read emergencies descriptions.
- Open a menu with several actions.

When the user presses over any of the emergencies, a dialog box will be shown with information about it. In addition, it is possible to see the location of the emergency in a map when the image of the dialog box is selected. Another possibility the interface provides is to see the position of every emergency that is shown.

The button located in the bottom right of the screen is used to notify any emergency situation as a speech. That message is then sent to the server which will process it and decide how to address it.

Finally, it is also possible to see a tab of settings with some options available such as *Help*, *User Settings*, *About* and *Log Out* in the left part of the screen.

4.6.1. Use cases

This module is the biggest in terms of use cases, is where the interaction is mostly done and that makes a big set of different situations.

App: Load main screen interface.
User: Press bottom-right button.
App: Show pop-up listening for speech.
User: Describe emergency situation.
App: Send the message to the server.

Figure 4.17: Send emergency

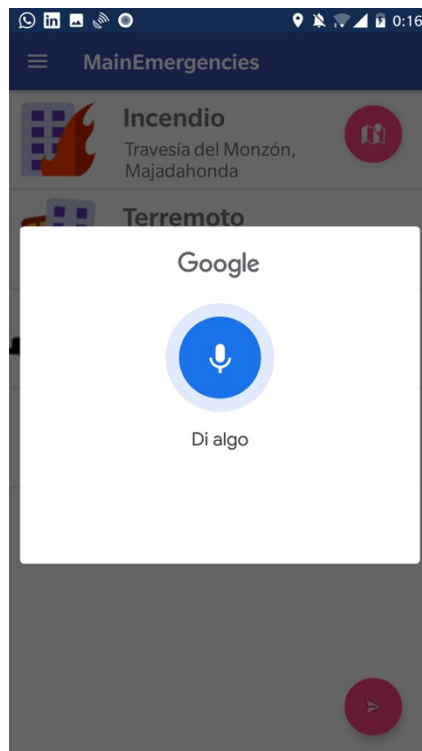


Figure 4.18: Google Speech recognizer

App: Load main screen interface.
User: Press emergency in list.
App: Show pop-up with an image, emergency type, location and description.

Figure 4.19: Select emergency

App: Load main screen interface.
User: Press emergency in list.
App: Show pop-up with an image, emergency type, location and description.
User: Press image.
App: Open maps module to show the location of selected emergency.

Figure 4.20: Show location of specific emergency

App: Load main screen interface.
User: Press top-right button or slide the finger from right to left in the screen.
App: Open maps activity to show current location of all emergencies in the application list.

Figure 4.21: Show location of all emergencies in list

App: Load main screen interface.
User: Press the three lines button at the left of the title or slide the finger from the left to the right in the screen.
App: Show a menu of actions in the left part of the screen.
User: Select any of the possible actions provided.

Figure 4.22: Show application menu

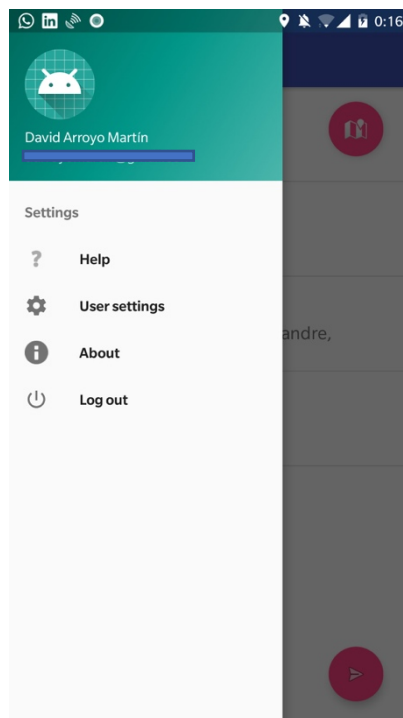


Figure 4.23: Application menu

4.7. Maps

When a user in the main activity module selects the top-right button, the maps module is then started. This module shows a map showing the current location of the user along with every emergency situation happening near them.

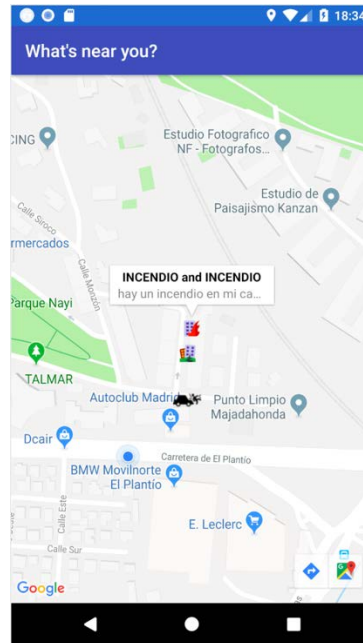


Figure 4.24: Maps activity

Figure 4.24 represents a real situation of a user having some emergencies happening next to it. The user's current location is represented with the blue point whereas each emergency is represented with an icon.

Each emergency is shown as a different image in relation to the type. For example, fires are represented as a building with flames whereas a car accident is represented with a broken car. In addition, it is possible to click over any of these icons to see information about the emergency it represents.

4.7.1. Use cases

In this module there are no real use cases as the main function of it is just showing the exact position where emergencies are happening. The only case in which users interact with this module would be moving the map and select any icon to see information about it. This situation is represented in figure 4.25.

App: Load maps interface.
 App: Load emergencies and current position.
 User: Move the map with the fingers.
 User: Select an icon by pressing over it.

Figure 4.25: Select emergency icon in map

4.8. Settings

This module is loaded when, in the main screen menu the option of “Settings” is selected. Settings interface shows a set of different options used to manage user accounts.

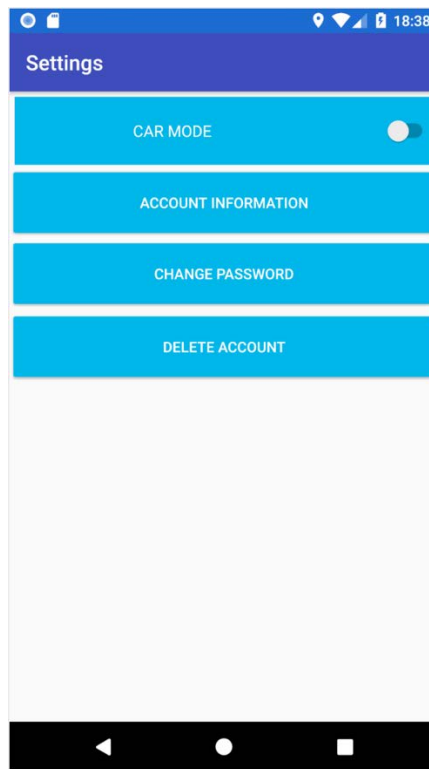


Figure 4.26: User settings

Figure 4.26 shows the four different options available to manage the user account. Firstly, there is an option to activate or deactivate the car mode option. This feature is used to notify by speech new emergencies received by the phone. If it is activated, when a new emergency is detected, the device will reproduce by speech all the information about it as if it was a person. The idea of this feature is to be used in the car because, as it is not possible to check the phone when driving, at least the user will know about the existence of new emergencies near them.

CHAPTER 4. Detailed description of the application modules

The second option opens a new interface, shown in figure 4.27, with some information about the user that was introduced in when the account was created. All information in exception to the username is possible to be changed to any other.

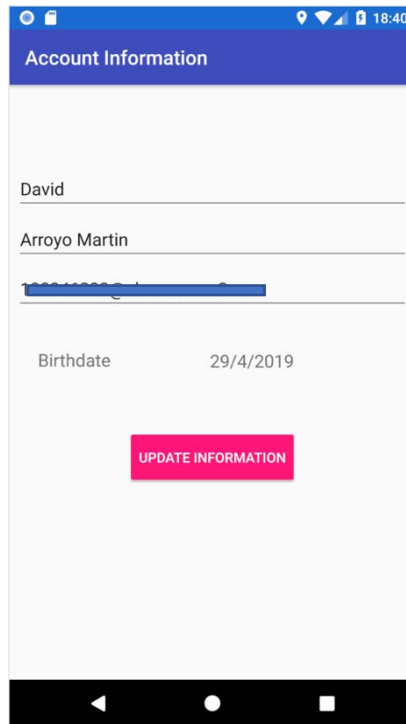


Figure 4.27: Account information interface

The third option is used to change the current password of the account to a new one. When selected, it opens a new interface similar to the one shown in figure 2.28 in which it is necessary to include the previous password and a new one.

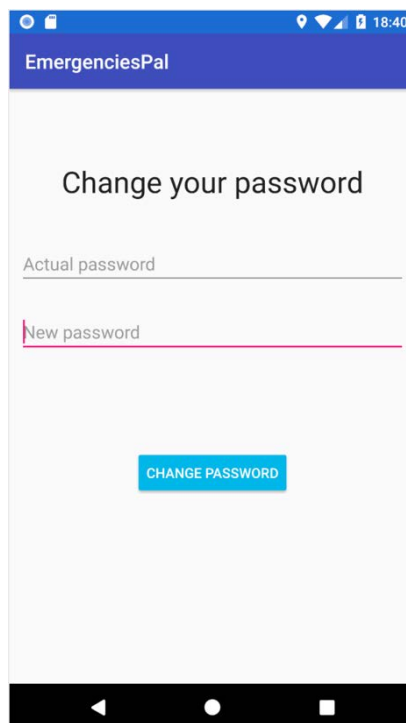


Figure 4.28: Change password interface

Finally, the last option just gives the possibility to delete the account if it is not going to be used anymore and erase all personal data related to the account stored in the server database.

4.8.1. Use cases

There are four main use cases in this module. Change the state of the car mode feature, select any of the three remaining options, update the user account information and change the current password.

App: Load settings interface.
User: Press the switch of the car mode option.
App: Send new state to the server.

Figure 4.29: Car mode state update

App: Load settings interface.
User: Select “account information” or “change password” options.
App: Load “account information update” or “password update” interfaces.

Figure 4.30: Select option

App: Load settings interface.
User: Select delete account option.
App: Send account username to the server to delete all information related to that user.

Figure 4.31: Delete account

App: Load account information interface.
User: Change any field as wanted.
User: Press “Update information” button.
App: Send new information to the server for its update.

Figure 4.32: Update information

App: Load password change interface.
User: Introduce current password in “actual password” field.
User: Introduce new password in “new password” field.
User: Press “Change password” button.
App: Send new password to the server for its update.

Figure 4.33: Update password

4.9. Emergency update service

The final part of the application is a service running in background. A service is an application component that can perform long-running operations in background and that does not provide any type of interface. Services are started and stopped by any other application component and continue running even if another application is opened or if it is closed.

The service running in the application has two main functions:

- Update current emergency situations.
- Send notifications.

The first function has the responsibility to keep an accurate state of the emergency situations happening near the user. This is done by periodically sending requests to the server attaching the latitude and longitude of the device with the objective of getting back all the emergencies occurring near that pair of coordinates. When the service gets back the answer from the server with the set of all emergencies, the service compares them with those ones stored in the local SQLite database of the device. Those emergencies stored in the local database that are not in the server answer are erased from the database because it means that they have finished. On the other hand, those emergencies sent from the server that are not in the local database are included in it.

The second function happens when there are emergencies that have been included in the local database. When this occurs, the service creates a notification that will appear in the device with the objective of notifying users that there are new emergencies near them. On the other hand, if no emergency has been included in the local database, it means that there are no new situations that could be of the user interest. In addition, the service does also check if the user has its “car mode” option active. If it is active, the service, at the same time it sends the notification, will describe by speech all new emergencies included in the database, but not those that were already there.

4.10. Application-Server connection

The communication between the application and the server is done completely by means of TCP sockets. In this section, we will describe the different communication protocols followed in each of the modules and services in which data is exchanged with the server.

Every time a socket is created to establish a connection with the server it needs to be done in a thread, and all data flow between the application and the server will also be done inside the thread. This is done because if something fails, it will not stop and close the application.

4.10.1. Login interaction

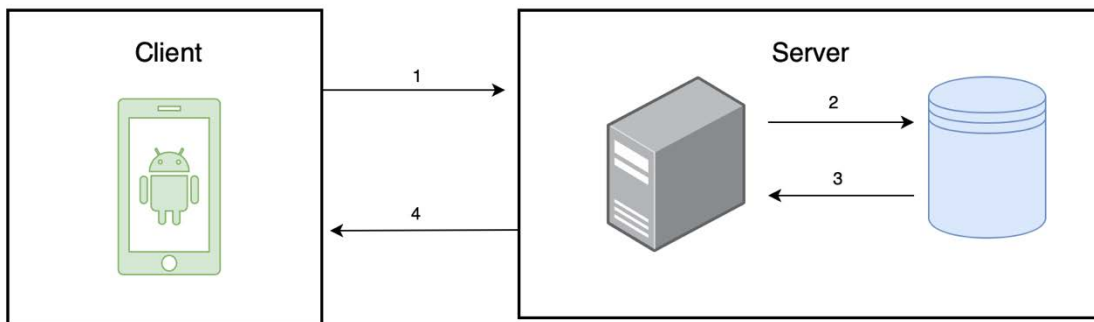


Figure 4.34: Login interaction

1. Send username and password.
2. Retrieve password for the given username in step 1.
3. If password retrieved (user exists), compare both passwords.
4. Send back successful login, username does not exist, or password not correct.

4.10.2. Register interaction

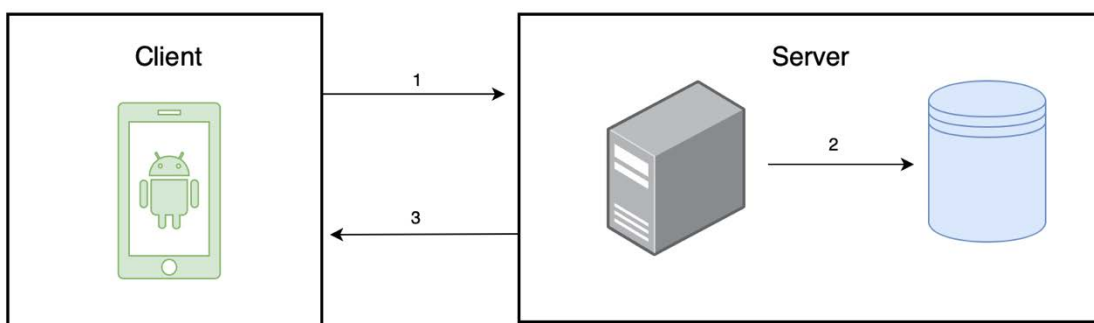


Figure 4.35: Register interaction

1. Send username, password, email, name, surname and birthday.
2. Check no user is already registered with the username sent in step 1.
3. Send back successful registration or not valid username.

4.10.3. Remember password interaction

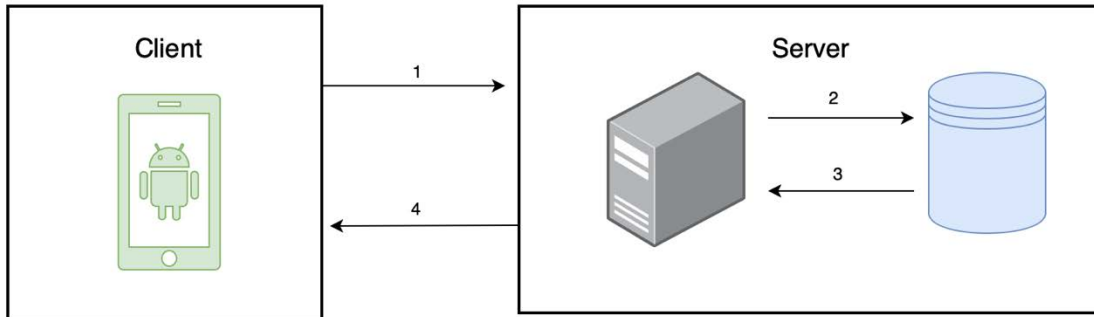


Figure 4.36: Remember password interaction

1. Send username, email and new password.
2. Check both username and email pair belong to the same account.
3. Get result from database query in relation to the condition stated in step 2.
4. Send back successful password recovery or error in case any of the username or email fields are incorrect.

4.10.4. Main screen interaction

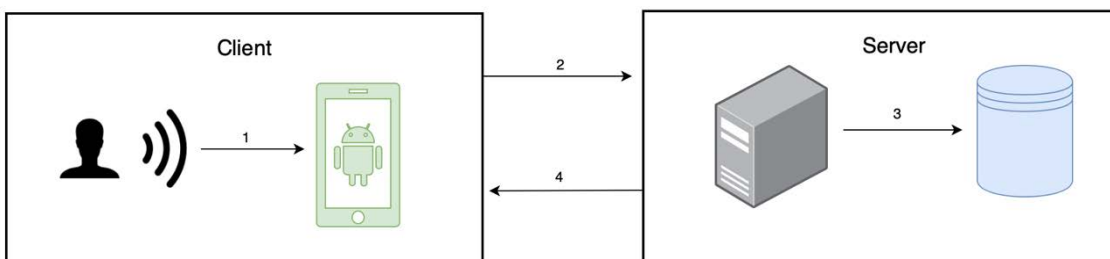


Figure 4.37: Main screen interaction

1. Describe emergency by voice
2. Send emergency described in step 1.
3. Compare existing emergencies with received one to check if it has already been added.
If not already in the server database, store the emergency in the database, otherwise, update datetime.
4. Return successful if emergency added or updated correctly into the database, error if something unexpected happened.

4.10.5. Settings interaction

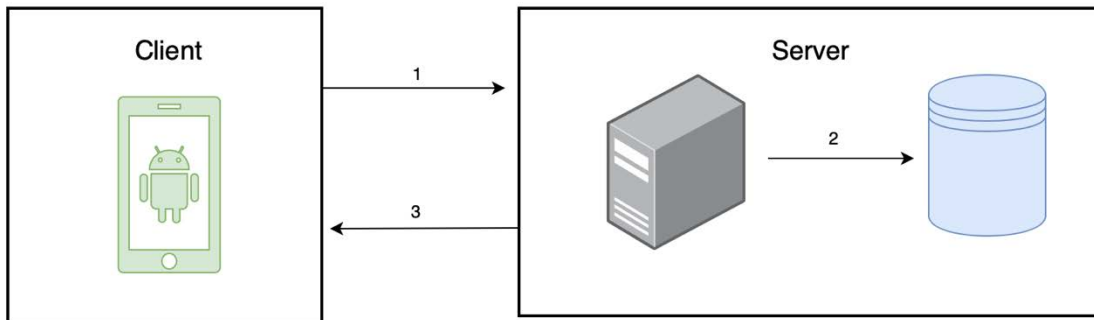


Figure 4.38: Settings interaction

Four different types of interaction can happen in the settings module.

- **Option 1:** Change password.
 - **Option 2:** Update user information.
 - **Option 3:** Text to speech car mode status.
 - **Option 4:** Delete account
1. **Option 1:** Send username, new password and actual password.
Option 2: Send username, email, name, surname and birthday.
Option 3: Send surname and new car mode state.
Option 4: Send username.
 2. **Option 1:** Check if the current password of the user is correct and if so, update the password to the new one.
Option 2: Update account information for the given username account.
Option 3: Update car mode state parameter in the database for the given username.
Option 4: Delete all information related to the given username.
 3. Returns successful if everything went as expected or error if something did not go as expected.

4.10.6. Emergencies update service interaction

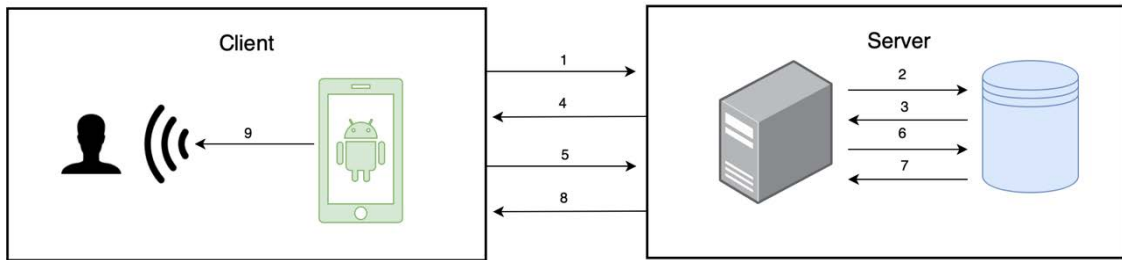


Figure 4.39: Emergencies update service interaction

1. Send latitude and longitude of the device.
2. Request emergencies near the coordinates sent in step 1.
3. Return back the list of emergencies requested in step 2.
4. Send back all emergencies collected in step 3.
5. Send username.
6. Check Text to Speech car-mode state for the given username.
7. Return back the car-mode state requested.
8. Send back the current car-mode status of the user.
9. If new emergencies send notification and if car-mode activated, use Text to Speech to describe all new emergencies.

CHAPTER 5. Evaluation of the application

This chapter describes the satisfaction evaluation done to the application developed in this Bachelor project. In order to make this evaluation, a questionnaire has been created to collect the opinions of different users that had the opportunity to test the application. All results obtained in this process will be shown by means of graphics.

5.1. Evaluation methodology

When an application is developed, the evaluation, performance study and usability analysis are essential processes that help understand what users think of it and how the application can be improved to satisfy them.

The questions asked in the questionnaire created for this Bachelor project focus on the following aspects:

- Previous knowledge and experience.
- Application fluency and appearance.
- Functionalities.
- Utility.

The questionnaire is formed by 16 multiple choice questions that are mandatory to respond and one final optional question in which users can provide a better opinion about the application, describing their thoughts, what they liked and what they would do differently.

The questionnaire has been created with Google Forms [71], which is an online survey administration application included in the Google Drive office suite. The reason why I used this platform was to facilitate users an interface that looked familiar to them as this is one of the mostly used platforms to create forms. In addition, Google Forms provides a results interface that shows the results of all individual questions in pie charts, which I found very useful to save time making the pie charts by myself.

Figure 5.1 shows the questionnaire made for this Bachelor Degree Thesis.

EmergencyPal

Appreciated EmergencyPal user,

Thanks for using the application, the completion of this survey will be very useful for improving it in the future. Please answer all the questions sincerely.

Thanks for your time!

Do you have experience with smartphone interaction? (1 = None, 5 = Every day use) *

Option 1

2

3

4

5

Have you used any other application with similar objectives? *

Yes

No

How well have you understood the application? *

Very bad

Bad

Regular

Good

Very good

When using the application, did it look nice? *

Yes

No

Maybe

How good was the interaction with the application? (1 = Very bad, 5 = Very good) *

1

2

3

4

5

Did you find useful to login with a Google account? *

Yes

No

Maybe

Did you find useful the recover password possibility? *

Yes

No

Maybe

Have you ever used applications with voice interaction? *

Yes

No

Did you find useful the voice interaction? *

Yes

No

How good did the voice recognition work? (1 = Very bad, 5 = Very good) *

1

2

3

4

5

How useful is to see the emergencies as a list and see information about them by clicking over them? (1 = Not useful, 5 = Very useful) *

1

2

3

4

5

How useful is the map to see the location of the emergencies? (1 = Not useful, 5 = Very useful) *

1

2

3

4

5

Were the emergencies shown precisely in the map? *

Yes

No

Maybe

What interface do you prefer to see the emergencies, the map, the list or both? *

List

Map

Both

Did you find the application useful? (1 = Not useful, 5 = Very useful) *

1

2

3

4

5

Would you recommend the application to other people? *

Yes

No

Maybe

Personal comments about the application

Long answer text

Figure 5.1: EmergencyPal questionnaire

5.2. Evaluation results

The evaluation of the application has been done by 15 users that were previously informed about the application objective and functionalities. The results obtained are represented below by means of pie charts and will be analyzed at the end of the chapter.

Do you have experience with smartphone interaction? (1 = None, 5 = Every day use)

15 responses

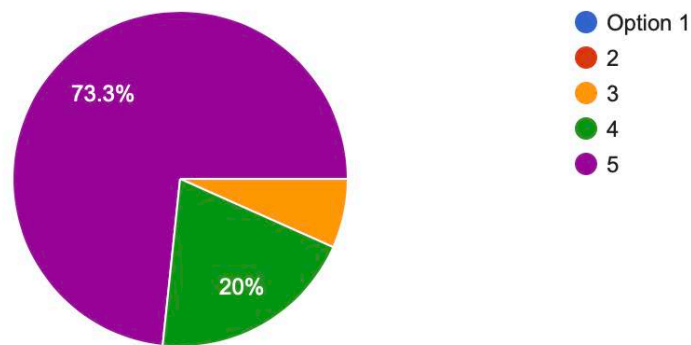


Figure 5.2: Question 1

Have you used any other application with similar objectives?

15 responses

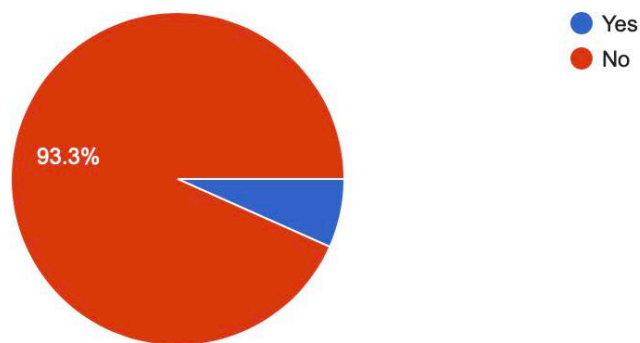


Figure 5.3: Question 2

How well have you understood the application?

15 responses

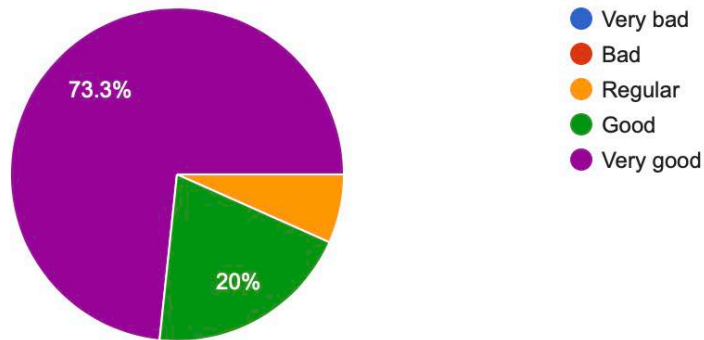


Figure 5.4: Question 3

When using the application, did it look nice?

15 responses

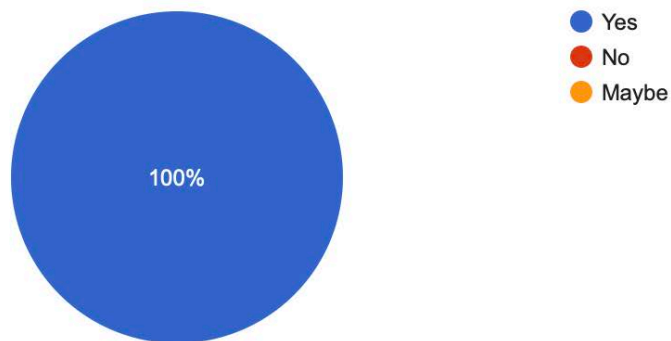


Figure 5.5: Question 4

How good was the interaction with the application? (1 = Very bad, 5 = Very good)

15 responses

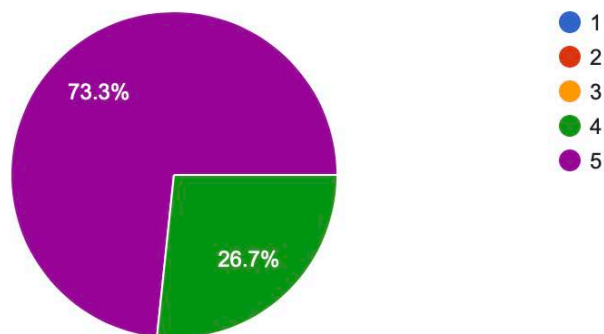


Figure 5.6: Question 5

Did you find useful to login with a Google account?

15 responses

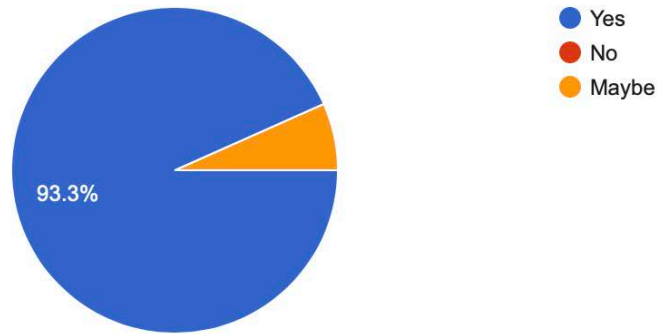


Figure 5.7: Question 6

Did you find useful the recover password possibility?

15 responses

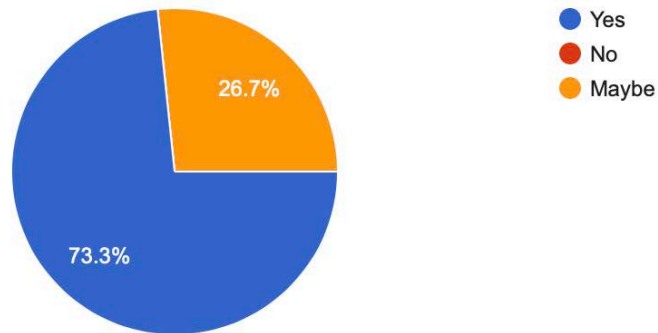


Figure 5.8: Question 7

Have you ever used applications with voice interaction?

15 responses

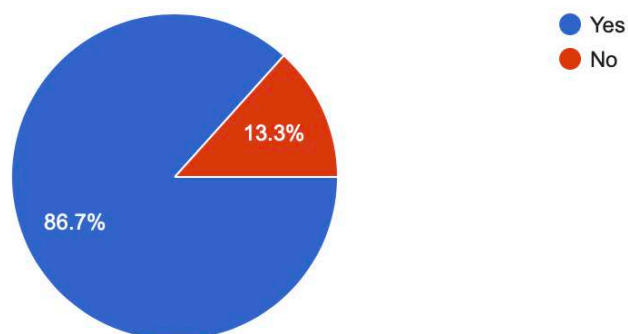


Figure 5.9: Question 8

Did you find useful the voice interaction?

15 responses

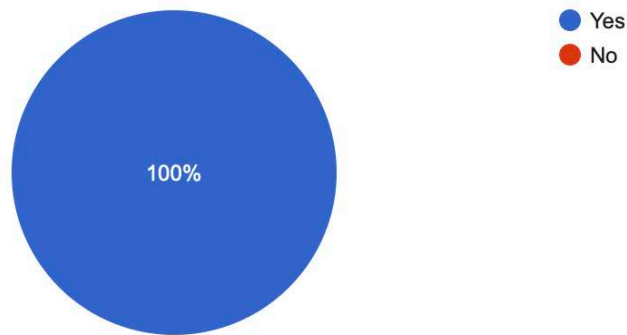


Figure 5.10: Question 9

How good did the voice recognition work? (1 = Very bad, 5 = Very good)

15 responses

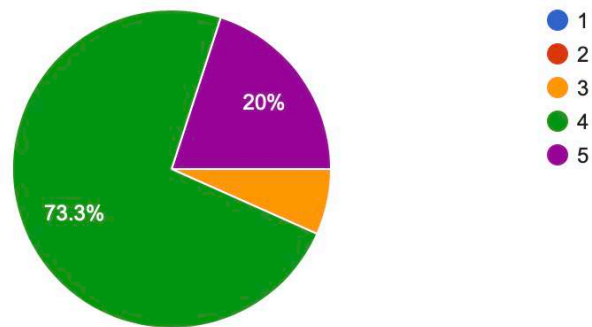


Figure 5.11: Question 10

How useful is to see the emergencies as a list and see information about them by clicking over them? (1 = Not useful, 5 = Very useful)

15 responses

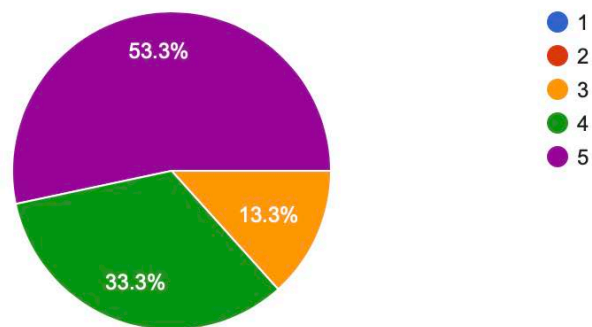


Figure 5.12: Question 11

How useful is the map to see the location of the emergencies? (1 = Not useful, 5 = Very useful)

15 responses

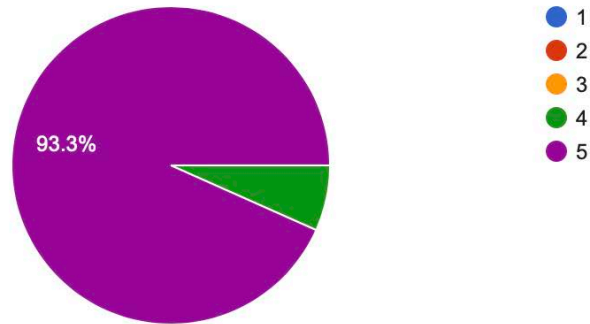


Figure 5.13: Question 12

Were the emergencies shown precisely in the map?

15 responses

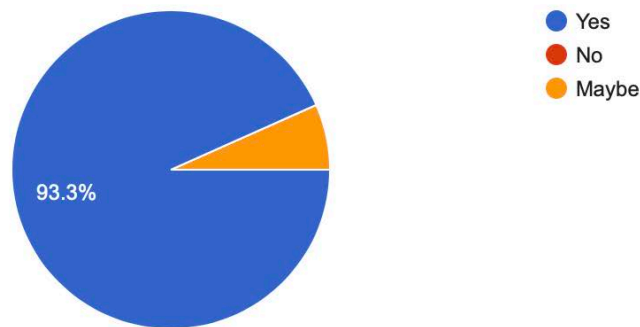


Figure 5.14: Question 13

What interface do you prefer to see the emergencies, the map, the list or both?

15 responses

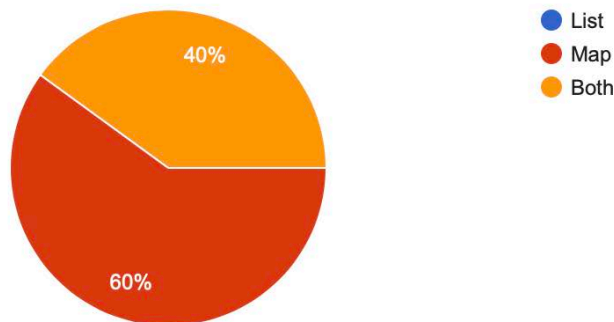


Figure 5.15: Question 14

Did you find the application useful? (1 = Not useful, 5 = Very useful)

15 responses

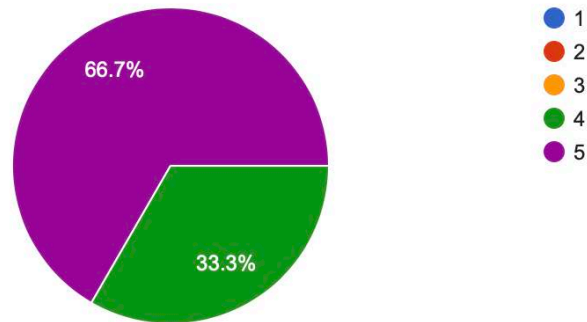


Figure 5.16: Question 15

Would you recommend the application to other people?

15 responses

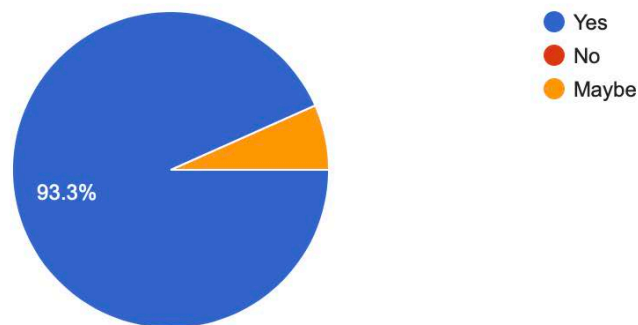


Figure 5.17: Question 16

Personal comments about the application

5 responses

- I found the application really interesting because I have never used an application like this and I consider it can be very useful for people.
- Very nice application, I have never used an application like this and I like the idea.
- The idea of the application is good. In my opinion, having the list is not as useful as the map because in the map you can see directly where it is in an easier way. The interaction was nice and smooth.
- Very nice work, the application looks amazing!
- Its very cool

Figure 5.18: Question 17

In relation to the previous knowledge and experience of the users, the first question shows the expected result, most users consider they use a smartphone almost every day, which is a good starting point because the application is designed for smartphones.

The second question shows a very interesting situation, as the result says that for 14 of the 15 users who tested the application, it is the first application they have used that has the intention of providing a support for emergencies. Moreover, finish with this aspect, most of the users have experience with voice interaction applications, only two of the them have not used any other application with it.

In the questions related to the application fluency and appearance, users consider that the application was easy to understand, as 14 of them think that they understood the application good or very good, just one responded regular. This shows that the intention of the application is well defined though the interfaces. In addition, all users voted that the application looked nice, which is an added value to the application.

To end up with this second aspect, users consider that the interaction with the application is very good or good. In conclusion, we can say that the application is easy to understand and use.

The questions related to the functionalities show that in general, the possibility given to login with a Google account is considerate as useful, as well as the recover password functionality, having in this one a bit more of undecided users. Voice interaction is considered useful by all users, and they think that it worked well.

When comparing the way emergencies are shown in terms of the list or the map, users think that both of them are useful and represent information precisely, but if they had to choose between any of them or both, all of them have voted that they prefer the map or both, but not only the list. This shows a very interesting point of view that will help to think of new ideas for the future of the application.

Finally, all users think that the application is useful or very useful. This is a very important question because, as the results from the second question showed that just one had previous experience with this kind of applications, it means that it may have future in the market.

CHAPTER 6. Project management

In this chapter, the temporal planning will be described showing the different tasks fulfilled and the time spent in each of them approximately. In addition, it includes the budget needed to carry out this project. Moreover, there regulatory framework is exposed making reference to the different laws that must be taken into account when developing an application for any type of device. Finally, the chapter ends with a description about the socio-economic impact that the application EmergencyPal could have.

6.1. Temporal planning

The temporal planning will be represented by means of a Gantt diagram. A Gantt diagram is a graphical tool used to represent the dedicated time to each of the tasks that form a project. The application used to create the Gantt diagram is GanttProject⁵, which is a free project management software that permits the creation of this type of diagrams.

Figure 6.1 shows the different tasks considered to create the Gantt diagram with the start and end dates along with the total number of days spent with them. As described in Section 1.4 four different tasks will be considered, planification, development, evaluation and documentation, along with their corresponding subtasks.

Nombre	Fecha de inicio	Fecha de fin	Duración
▼ ● Planification	16/10/18	3/12/18	35
● Requirement analysis	16/10/18	2/11/18	14
● Analysis of required techn...	5/11/18	27/11/18	17
● Analysis of developing tools	8/11/18	28/11/18	15
● Analysis of related applica...	29/11/18	3/12/18	3
▼ ● Development	4/12/18	9/04/19	91
● Design	4/12/18	20/12/18	13
● Implementation	21/12/18	20/03/19	64
● Components integration	21/03/19	9/04/19	14
● Developer level testing	30/01/19	2/04/19	45
▼ ● Evaluation	22/04/19	14/05/19	17
● User level testing	22/04/19	8/05/19	13
● User evaluation	9/05/19	14/05/19	4
▼ ● Documentation	20/02/19	18/06/19	85
● Document writing	20/02/19	27/05/19	69
● Final presentation	10/06/19	18/06/19	7

Figure 6.1: Gantt diagram tasks

⁵ <https://www.ganttproject.biz>

Figure 6.2 represents the result of the previous Figure. In it, it is possible to see the different tasks over time. It is important to mention some tasks overlap, such as the development of the application with the developer level testing. This overlapping is necessary as in order to be sure that the developing process was going correctly, it should be tested. In addition, the documentation process also overlaps both the developing and evaluation tasks. This has not been a problem as the writing has been done always following the developing steps, which means that once something was finished, I started writing about it.

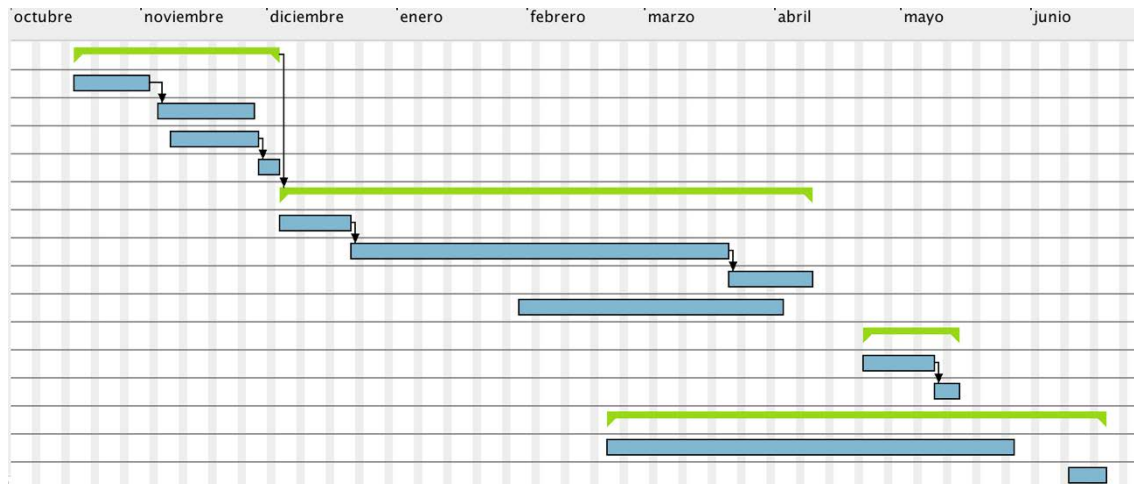


Figure 6.2: Gantt diagram

In conclusion, the project has had a total duration of 245 days (8 months), starting 16th of October of 2018, and finishing 18th of June of 2019. I knew that the project would take a lot of time because I had to enroll in many courses this year and it was going to be impossible to focus only on it, and because of this, I decided to start early. Taking into account this, from the 245 days, excluding rest, vacations and the necessity to center in the course, a total of 130 days have been dedicated for the Bachelor project approximately.

The average time spent by day is of 3 hours during the week and 5.5 hours during the weekends. Taking into account this, and knowing that 130 days were productive, 26 hours were spent during the week approximately, which means 3.71 per day. This implies a total of 482.85 hours dedicated for the Bachelor project.

6.2. Budget estimation

This section describes an analysis of the total costs of the Bachelor project. These costs are divided mainly in two parts: direct costs (human and resources) and indirect costs (20% of the total direct costs). This analysis is done thanks to the course Introduction to business management that I took in the second course of my degree.

- **Direct costs:** Show the total costs of the project in relation to the human and material resources as well as the realization of it.

In relation to the human costs, it is important to mention that it depends on the degree, the qualification or even the geographical location of the worker. In Spain, taking into account a study done by Job & Talent, a recently graduated engineer has an average salary of 1800 Euros gross, which is 21600 Euros gross per year working 40 hours per week. A company does not only pay for this amount, it has to consider also other percentages such as the social security (common contingences) and other concepts such as the unemployment, working accidents, job training or the social guarantee fund (professional contingences). In Table 6.1 there is an analysis of all the different percentages described before along with the quantity they represent [52] [53].

Calculations:

Common contingences = 21600 € x 0.236

Professional contingences = 21600 € x 0.0745

Total company cost = 21600 + Common contingences + Professional contingences

Cost per hour = Total company cost / 2080 hours

Cost = Cost per hour x Total hours

Personal	David Arroyo Martín
Days worked	130
Hours per day	3.71
Total hours	482.85
Common contingences (23.6%)	5097.60 €
Professional contingences (7.45%)	1609.20 €
Total company cost	28306.80 €
Cost per hour	13.61 €/hour
Total cost	6571.59 €

Table 6.1: Company costs in terms of a worker

In relation to the costs of the material resources (hardware and software), in table 6.2, the hardware resources needed for the development of the system are shown and, in Table 6.3 the software resources are shown.

Product	Cost
Laptop	1250 €
Smartphone	170 €
USB cable	2 €
External screen BenQ	125 €
TOTAL	1547 €

Table 6.2: Hardware resources costs

Product	Cost
Microsoft office 2019	149 €
AVD	0 €
JDK	0 €
SDK	0 €
Windows 10	0 €
Atom	0 €

OneDrive	0 €
Firebase	0 €
Twitter API	0 €
MeanningCloud API	0 €
Android Studio	0 €
TOTAL	149 €

Table 6.3: Software resources costs

In order to take into account correctly the cost of these products in terms of the project duration, the lifetime of each of them must be considered. Generally, the lifetime of these objects is of five years, but as the project duration has been of 8 months, only those 8 months should be considered. In order to compute the amortization for each product, linear amortization will be considered, as it is supposed that the usage will be similar for the five years. The linear amortization for each of the products is shown in Table 6.4. The formula used to calculate the linear amortization is the following:

$$\text{Amortization} = \frac{\text{time used}}{\text{lifetime}} \times \text{Value} \times \text{Usage coefficient}$$

Where:

- Time used: Days used.
- Lifetime: As commented before, five years will be considered, which is 1825 days.
- Value: Cost of the product (See Table 6.2 and Table 6.3).
- Usage coefficient: percentage of use dedicated to the project (100% normally).

Product	Cost	Usage coefficient (%)	Project duration (days)	Lifetime (days)	Linear amortization
Laptop	1250 €	100	245	1825	167.80 €
Smartphone	170 €	100	245	1825	22.82€
USB cable	2 €	100	245	1825	0.27 €

External Screen BenQ	125 €	100	245	1825	16.78 €
Microsoft Office 2019	149 €	100	245	1825	20 €
TOTAL	-	-	-	-	227.67

Table 6.4: Product linear amortization

Finally, Table 6.5 shows a summary of the direct cost related to the project.

Description	Cost
Personal costs	6571.59 €
Materials	227.67
Direct costs without IVA	6799.26 €
Direct costs with IVA (21%)	8227.10 €

Table 6.5: Direct costs

- **Indirect costs:** Represent the 20% of the direct costs, therefore, the total indirect costs are 1645.42 €. Table 6.6 shows the total budget required by this Bachelor project.

Description	Cost
Direct costs	8227.10 €
Indirect costs	1645.42 €
TOTAL	9872.52 €

Table 6.6: Bachelor Degree Thesis costs summary

After all this analysis, we can conclude that the total budget required by this Bachelor Degree Thesis is of **NINE THOUSAND EIGHT HUNDRED SEVENTY TWO POINT FIFTY TWO EUROS.**

6.3. Regulatory framework

Creating an application is not only thinking of an idea and implementing it, it is also very important to consider all legal aspects that must be taken into account to develop an application so that it obeys all laws and regulations established [54] [55] [56] [57].

The regulations that every application must obey are:

- **Illicit functionalities:** An application must not allow the existence of activities that encourage to make illegal or immoral actions or attitudes.
- **Own rights and third parties:** Have the licenses required for the resources that are being used. Read conditions of third-party resources carefully because there are some cases in which they exclude the commercial use and cannot be included in the application. Finally, protect own content to avoid plagiarism.
- **Permissions, licenses and conditions of use:** Be precise and clear when asking for permissions. It is mandatory to develop licenses and conditions of use, not to only inform the users but to make them accept them.
- **Minors:** If the application is intended to be used by minors with less than 14 years, it is mandatory to check the corresponding regulations because there are specific laws for these cases such as the data protection or image rights. In these cases, parent's authorization is mandatory and should be asked for it. In Spain, the minimum age for which this is not required is 14 years.
- **Privacy and geolocation:** Information required by the application must be only the necessary one to work as it should, this means that information that is not needed should not be asked for. Moreover, if the application needs the location of the device, this permission should be accepted by the user. In addition, if the user decides to change this permission, it should be allowed to do so.
- **Information and cookies:** It is fundamental to inform users about the regulated aspects in the law and show data related to the developers and who they are. In addition, it is necessary that users accept the cookies by means of an informative notice with precise and basic information about them and the aspects required by law.
- **Markets:** Each application market has its own specific conditions that must be met in order to be able to publish an application on it. In addition, it is possible that these conditions change, and, for some users, the application would not be available.

- **Advertising:** Every time an application monetizes from advertising, it must be clearly specified.

Two very important laws to be taken into account when an application makes use of personal information are the *Organic Law of Data Protection (LOPD)* and the *Law of Services of the Information Society and Electronic Commerce (LSSI-CE)*. The requirements that must be fulfilled to follow these laws are:

- **Terms and conditions of use:** Provide clear and complete information about the developers who are behind the application, third parties with whom data may be shared and information about the application usage. This information must be available before installation and user consent must be provided.
- **User consent before application installation:** When asking about permissions, they must be asked clearly and precisely and the reason why they are required must be explained in every moment. If the application makes use of any permission without asking for it, it is not considered to be working inside legality.
- **Right to uninstall the application:** Facilitate users free, simple and accessible mechanisms to uninstall the application, decline previous conceded permissions or delete any information from the application.

In relation to the developed application in this Bachelor project, no advertising or cookies have been used. The application has not been published in the Google Play Store and the only permission required to be accepted by the users is geolocation, which is asked when the application is run for the first time and that can be revoked in any moment in the permission settings of the smartphone.

6.4. Socio-economic factor

Nowadays there are multiple devices that are used by people daily, such as smartphones, tablets, computers or video consoles. Between all these possibilities, there is one that is used by 92% of the users, the smartphone. Figure 6.3 shows a graphic of the devices used by people in Spain in 2017 along with their percentages.

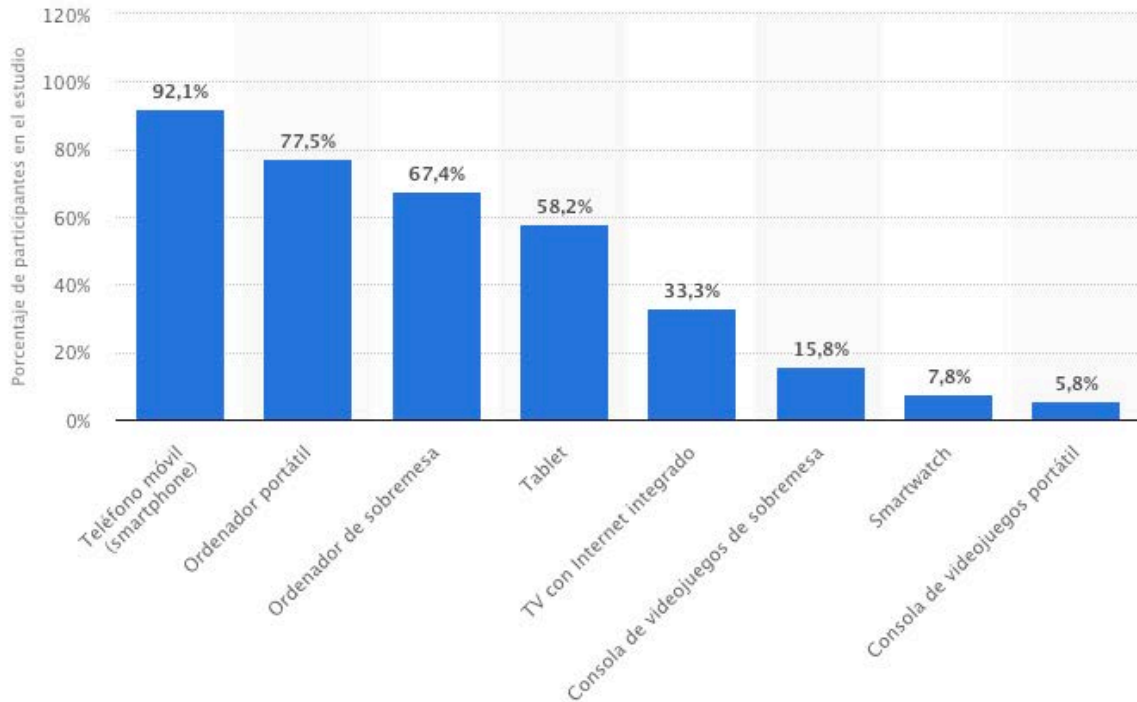


Figure 6.3: Devices used by people in Spain

As mentioned above, smartphones are the most used in comparison with the others shown in Figure 6.3. The reason of this is because they are portable and are used for anything, such as making calls to other people, sending messages over the Internet, see our favorite series or even play games. In addition, most of the activities done with a smartphone are connected, directly or indirectly, with Internet. Because of this, companies consider applications to have a very big socio-economic impact.

In the Google Play Store, it is possible to find plenty different types of applications. Some of them with the objective to help users and others simply to entertain them, but not all of them have the same impact.

The impact an application has depends mostly on what it offers to the users. In addition, the interest of users in the topic an application belongs to is also a determining factor. An example of applications that are having a higher socio-economic impact are those related to IoT, Virtual and Augmented reality, and those that are designed for wearable technology.

In relation to applications which have the objective of providing users a service of support for emergencies, in the Google Play Store it is possible to find multiple different applications, but the impact they have is very low. Taking into account the applications described in Section 2.5, with the exception of Waze, which was not developed for emergencies support, the application with the highest number of downloads is My112 with more than 500.000.

Considering that My112 works only in Spain and that in Spain there are 46.72 million people, only 1% of them has downloaded this application. Knowing this, we can ensure that applications with this objective have a very low socio-economic impact as people do not show a high interest in them [58] [59].

CHAPTER 7. Conclusions and future work

This chapter represents the summary of the work done for this Bachelor project. After analyzing the results obtained in Chapter 5, we can say that the objectives set at the beginning of the project have been successfully completed. Firstly, the conclusions of this project will be exposed and, finally, some ideas and new functionalities will be described to be performed as a future work.

7.1. Conclusions

In this Bachelor project, an application for Android devices has been developed with the objective of providing knowledge about emergency situations happening in every moment.

The first interface a user finds when opening the application is the login screen. In it, it can choose to login by means of an application account or a Google account. In addition, it can choose to create a new account or to recover its password.

The main interface of the application presents a list of emergencies that are happening near the device in real time. Here, it is possible to click over any of them and see information about each of them, such as the type, location and description. In addition, it is possible to see the emergencies on a map, which helps to see the exact location where each emergency is in relation to the current location of the user. Moreover, this interface provides access to the user settings. Finally, emergencies can be notified from here by clicking a button that opens a Google Speech to Text dialog box ready to listen to what the user wants to describe. When the description is ended, the Google Speech to Text recognized will automatically stop and a new dialog box will appear showing the message and a button that must be clicked to send it to the server.

In the user settings, users can access their account information and modify it if they want to. In addition, they can delete their account along with its information in any moment and, also, change their password to a new one.

Users can also activate the car mode option in the settings interface if they want to hear the new emergency situations happening near them when they are updated, by default it is not activated.

In order to develop this application, the first thing done was to make a detailed study about Android operating system in order to understand it and decide to which version this application would be done. The version is very important because it will define the number of users that will have access to the application so, the higher it is, the more users will be able to download it.

A very important part of the application is the communication between the application and the server which manages all the information so, in order to decide how to do this, I made use of what I learned in the Distributed Systems course as well as making an analysis comparing different options. Finally, I chose the best one considering the resources that were going to be needed as well as the functionalities that were intended to be implemented.

In addition, as there are some APIs included in the system, I had to learn how to implement them in the desired part of the system as well as understand how they worked. Examples of this are the inclusion of the Twitter API to get emergencies from the official DGT account, MeaningCloud or Firebase API to implement the Google login functionality.

One very important aspect of the system is how data is treated when it is processed in the server. Here one of the most important parts is how the MeaningCloud API extracts information and analyzes it. The text analysis and sentiment analysis done with this API is crucial, as it differentiates the type of emergency that is received as well as verifies its truthfulness.

As a final conclusion, it is possible to say that the initial objectives stated at the beginning of the project developed in this Bachelor project have been successfully achieved.

The results from the questionnaire obtained in Section 5.2. show that users consider the application useful and easy to understand and use. Even though they think the map is better to show the emergencies than the list, they consider that the list is also useful. In addition, the functionalities implemented such as the Google login or the voice interaction are evaluated as very useful by most of them.

In conclusion, the application works as expected and users are happy with how it works but, nevertheless, there are aspects of the application that can be improved, or new functionalities included. Therefore, in the next section, some improvements will be described to be included in the future.

7.2. Future work

In relation to the future work, the following statements are proposed:

- Include more **languages** to provide an interface that could be understood by a wider number of people. The actual language supported by the application is English, which limits the number of users to English speaking countries and it would be good to increase the number of languages taking into account the ones that are speak the most. In other words, include languages gradually by order of number of speakers.
- Implement a new security measure to allow users to include their **fingerprint** to login into their account. With this measure, the login process will be quicker as nothing has to be introduced in the spaces left to write the username and password.
- **Encrypt** the data if the previous proposal is finally implemented. In the actual version of the application, data is not encrypted as the one used is not sensitive but, if the fingerprint is included, in order to store its data securely, an encryption algorithm should be applied to the database and communications between the Android device and the server.
- Create a new interface used to send **opinions** and new **proposals** to the application. Examples of this could be types of emergencies that may not be included in the application and that users may consider important.
- Agreement with the security forces: It would be a good idea to work together with firemen, policemen or medical centers to provide a better and more precise operation of the application. The agreement would include also new possible updates such as:
 - Creation of new types of emergencies with a more personal intention and not intended to be to other users.
 - Creation of an interface in which the security forces could see the real time state of the emergencies sent by the users, include new ones or modify others.

Glossary

Android: Mobile operating system developed by Google.

Android TV: Android operating system designed for digital media players. It features a user interface designed around content discovery and voice search, surfacing content aggregated from various media apps and services, and integration with other recent Google technologies such as Google Assistant, Cast and Knowledge Graph.

Android Auto: Mobile app developed by Google to mirror features from an Android device to a car's compatible in-dash information and entertainment head unit.

API (Application Programming Interface): Set of subroutine definitions, communication protocols and tools used for building software.

Augmented reality: An enhanced version of reality where live direct or indirect views of physical real-world environments are augmented with superimposed computer-generated images over a user's view of the real-world, thus enhancing one's current perception of reality.

AVD (Android Virtual Device): Device configuration that is run with the Android emulator. Provides a virtual device-specific environment in which to install and run Android apps.

DGT (Dirección General de Tráfico): Government department responsible for the Spanish road transport network.

Gantt Diagram: Is a type of bar chart that illustrates a project schedule.

Google Assistant: Artificial intelligence-powered virtual assistant developed by Google that is primarily available on mobile and smart home devices.

GPS (Global Positioning System): Navigation system that allows land, sea and airborne users to determine their exact location, velocity and time 24 hours a day, anywhere in the world.

IOS: Mobile operating system developed by Apple.

Internet of Things (IoT): Extension of Internet connectivity into physical devices and everyday objects. Embedded with electronics, Internet connectivity, and other forms of hardware, these devices can communicate and interact with others over the Internet, and they can be remotely monitored and controlled.

IPad: Line of tablet computers developed by Apple.

iPod Touch: Line of smart devices developed by Apple with similar characteristics to Apple's iPhone but without cellular network data.

JDK (Java Development Kit): Software development environment used for developing Java applications and applets. Includes the JRE, and interpreter, a compiler and other tools needed for Java development.

JRE (Java Development Environment): Set of software tools for development of Java applications.

MeaningCloud: Software as a Service product that allows users to embed text analytics and semantic processing in any application or system.

MySQL: Open-source relational database management system (RDBMS) that provides support for SQL programming language.

MySQL Workbench: Visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system.

Network socket: One endpoint in a communication flow between two programs running over a network.

Python: High-level programming language.

Random Access Memory (RAM): Form of computer data storage that stores data and machine code currently being used.

System requirement: System specification needed or wanted.

SDK (Software Development Kit): set of software development tools that allow the creation of applications for multiple platforms. Multiple development kits exist, such as Java Development Kit or iOS SDK.

Smartphone: Device that allows access to the Internet, cellular network data and multimedia functionalities.

Splashscreen: Loading interface shown in smartphone applications at the beginning of the opening process.

SQLite: In-process library that implements a self-contained, zero-configuration, serverless, transactional SQL database engine.

Tablet: Device that allows access to the Internet, multimedia functionalities and, sometimes, cellular network just for accessing to the Internet but not for phone calls or text messages.

Tweet: Message sent through Twitter application.

User Interface (UI): Interface shown in software and computerized devices with a focus on looks or style.

Virtual reality: Experience taking place within simulated and immersive environments that can be similar or completely different from the real world.

Wear OS: Version of Android operating system designed for smartwatches and other wearables. It was previously called Android Wear.

112: Assistance number used by people for any kind of emergency situation in the European Union and many other countries of the world.

Bibliography

- [1] 112 Emergencias Comunidad de Madrid <<Información estadística>> [Online]. Available: <http://www.madrid.org/112/index.php/inicio/presupuestos-contratos-gastos/informacion-estadistica>. Last Access: June 2019.
- [2] Wikipedia <<Work breakdown structure>> [Online]. Available: https://en.wikipedia.org/wiki/Work_breakdown_structure. Last Access: June 2019.
- [3] Wikipedia <<Android Programming Interface>> [Online]. Available: https://en.wikipedia.org/wiki/Application_programming_interface. Last Access: June 2019.
- [4] Sitepoint <<Beginning Android: Create an Android Virtual Device>> [Online]. Available: <https://www.sitepoint.com/beginning-android-create-an-android-virtual-device/>. Last Access: June 2019.
- [5] Wikipedia <<Directorate General of Traffic>> [Online]. Available: https://en.wikipedia.org/wiki/Directorate_General_of_Traffic. Last Access: June 2019.
- [6] Gis2gps <<What is GPS?>> [Online]. Available: <https://www.gis2gps.com/GPS/GPSDEF/gpsdef.html>. Last Access: June 2019.
- [7] Techterms <<iOS>> [Online]. Available: <https://techterms.com/definition/ios>. Last Access: June 2019.
- [8] Techterms <<Android>> [Online]. Available: <https://techterms.com/definition/android>. Last Access: June 2019.
- [9] Quora <<What is JDK?>> [Online]. Available: <https://www.quora.com/What-is-JDK>. Last Access: June 2019.
- [10] Techopedia <<Java Runtime Environment>> [Online]. Available: <https://www.techopedia.com/definition/5442/java-runtime-environment-jre>. Last Access: June 2019.
- [11] Wikipedia <<MeaningCloud>> [Online]. Available: <https://en.wikipedia.org/wiki/MeaningCloud>. Last Access: June 2019.
- [12] Wikipedia <<Software Development Kit>> [Online]. Available: https://en.wikipedia.org/wiki/Software_development_kit. Last Access: June 2019.

- [13] WhatIs <<Network socket>> [Online]. Available: <https://whatis.techtarget.com/definition/sockets>. Last Access: June 2019.
- [14] Techopedia <<SQLite>> [Online]. Available: <https://www.techopedia.com/definition/24610/sqlite>. Last Access: June 2019.
- [15] Wikipedia <<Android (operating system)>> [Online]. Available: [https://en.wikipedia.org/wiki/Android_\(operating_system\)](https://en.wikipedia.org/wiki/Android_(operating_system)). Last Access: June 2019.
- [16] AndroidAuthority <<The history of Android OS: its name, origin and more>> [Online]. Available: <https://www.androidauthority.com/history-android-os-name-789433/>. Last Access: June 2019.
- [17] Computer Hoy <<Android vs iPhone: la guerra de los smartphones en cifras>> [Online]. Available: <https://computerhoy.com/reportajes/industria/android-vs-iphone-guerra-smartphones-cifras-271447>. Last Access: June 2019.
- [18] Wikipedia <<IOS>> [Online]. Available: <https://en.wikipedia.org/wiki/IOS#Development>. Last Access: June 2019.
- [19] RubyGarage <<IOS vs Android development>> [Online]. Available: <https://rubygarage.org/blog/ios-vs-android-development>. Last Access: June 2019.
- [20] Unixmen <<Why is Android built on Linux Kernel>> [Online]. Available: <https://www.unixmen.com/why-is-android-built-on-linux-kernel/>. Last Access: June 2019.
- [21] Tutorialspoint <<Android – Architecture>> [Online]. Available: https://www.tutorialspoint.com/android/android_architecture.htm. Last Access: June 2019.
- [22] Developers Android <<Arquitectura de la plataforma>> [Online]. Available: <https://developer.android.com/guide/platform>. Last Access: June 2019.
- [23] Software de comunicaciones <<Arquitectura Android>> [Online]. Available: <https://sites.google.com/site/swcuc3m/home/android/generalidades/2-2-arquitectura-de-android>. Last Access: June 2019.
- [24] Wikipedia <<Dalvik (Software)>> [Online]. Available: [https://en.wikipedia.org/wiki/Dalvik_\(software\)](https://en.wikipedia.org/wiki/Dalvik_(software)). Last Access: June 2019.
- [25] AppBrain <<Top Android OS versions>> [Online]. Available: <https://www.appbrain.com/stats/top-android-sdk-versions>. Last Access: June 2019.

- [26] Wikipedia <<Android version history>> [Online]. Available: https://en.wikipedia.org/wiki/Android_version_history. Last Access: June 2019.
- [27] Computer World <<Android versions: A living history from 1.0 to Pie>> [Online]. Available: <https://www.computerworld.com/article/3235946/android-versions-a-living-history-from-1-0-to-today.html?page=2>. Last Access: June 2019.
- [28] The Verge <<Android: A 10-year visual history>> [Online]. Available: <https://www.theverge.com/2011/12/7/2585779/android-10th-anniversary-google-history-pie-oreo-nougat-cupcake>. Last Access: June 2019.
- [29] Make tech easier <<Google Play Protect>> [Online]. Available: <https://www.maketecheasier.com/google-play-protect-explained/>. Last Access: June 2019.
- [30] Android Authority <<Actions on Google makes it easy to build apps for Google Assistant>> [Online]. Available: <https://www.androidauthority.com/actions-on-google-introduction-807897/>. Last Access: June 2019.
- [31] Ministerio de interior <<Alertcops>> [Online]. Available: <https://alertcops.ses.mir.es/mialertcops/>. Last Access: June 2019.
- [32] Blogs Sony <<My112, la app de emergencias con geolocalización>> [Online]. Available: <https://blogs.sonymobile.com/es/tecnologia/my112-la-app-de-emergencias-con-geolocalizacion/#gref>. Last Access: June 2019.
- [33] Omicrono <<La aplicación que envía tu ubicación e imágenes al 112 cuando tienes una emergencia>> [Online]. Available: <https://omicrono.elespanol.com/2018/02/my112-aplicacion-ubicacion-imagenes-112-emergencia/>. Last Access: June 2019.
- [34] CNBC <<Only amateurs use Google Maps to avoid traffic – here’s how to use Google Waze like the serious commuters>> [Online]. Available: <https://www.cnbc.com/2018/11/13/how-to-use-google-waze-for-directions-and-avoiding-traffic.html>. Last Access: June 2019.
- [35] Xataka Movil <<A finales de 2018 habrá 3.000 millones de usuarios de smartphones en el mundo, según Newzoo>> [Online]. Available: <https://www.xatakamovil.com/mercado/a-finales-2018-habra-3-000-millones-usuarios-smartphones-mundo-newzoo>. Last Access: June 2019.
- [36] DGT <<DGT mapa de tráfico>> [Online]. Available: <http://infocar.dgt.es/etraffic/Buscador?caracter=acontecimiento&Camaras=true&SensoresMeteorologico=true&SensoresTráfico=true&Paneles=true&Radares=true&Incidencia>

[sRETENCION=true&IncidenciasOBRAS=false&IncidenciasMETEOROLOGICA=true&IncidenciasPUERTOS=true&IncidenciasOTROS=true&IncidenciasEVENTOS=true&IncidenciasRESTRICCIONES=true&provincia=77&poblacion=&carretera=&PK=&version=mapa&pagina=null&accion_buscar=Buscar](https://www.monkeylearn.com/sentiment-analysis/). Last Access: June 2019.

[37] MonkeyLearn <<Sentiment Analysis>> [Online]. Available: <https://monkeylearn.com/sentiment-analysis/>. Last Access: June 2019.

[38] MeaningCloud <<What is Sentiment Analysis>> [Online]. Available: <https://www.meaningcloud.com/developer/sentiment-analysis/doc/2.1/what-is-sentiment-analysis>. Last Access: June 2019.

[39] MeaningCloud <<What is Topics Extraction>> [Online]. Available: <https://www.meaningcloud.com/developer/topics-extraction/doc/what-is-topics-extraction>. Last Access: June 2019.

[40] Wikipedia <<Information Extraction>> [Online]. Available: https://en.wikipedia.org/wiki/Information_extraction. Last Access: June 2019.

[41] Dr.fone <<How to use Google Text-to-Speech on Android>> [Online]. Available: <https://drfone.wondershare.com/android/text-to-speech.html>. Last Access: June 2019.

[42] Twitter Developers <<Twitter API>> [Online]. Available: <https://developer.twitter.com/en/docs>. Last Access: June 2019.

[43] WideSkills <<Basic parts of Android application>> [Online]. Available: <https://www.wideskills.com/android/overview-android/principal-ingredients-android>. Last Access: June 2019.

[44] Techtarget <<What are the top MySQL features?>> [Online]. Available: <https://searchitchannel.techtarget.com/feature/What-are-the-top-MySQL-features-What-is-MySQL>. Last Access: June 2019.

[45] SQLite tutorial <<What is SQLite>> [Online]. Available: <http://www.sqlitetutorial.net/what-is-sqlite/>. Last Access: June 2019.

[46] iBiblio.org <<Features of Python>> [Online]. Available: <https://www.ibiblio.org/swaroopch/byteofpython/read/features-of-python.html>. Last Access: June 2019.

[47] Wikipedia <<Python (programming language)>> [Online]. Available: [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language)). Last Access: June 2019.

- [48] Twitter Developer <<Rate limiting>> [Online]. Available: <https://developer.twitter.com/en/docs/basics/rate-limiting.html>. Last Access: June 2019.
- [49] Social Media & Text Analytics <<Twitter API tutorial>> [Online]. Available: <http://socialmedia-class.org/twittertutorial.html>. Last Access: June 2019.
- [50] Firebase <<Como autenticar el Acceso con Google en Android>> [Online]. Available: <https://firebase.google.com/docs/auth/android/google-signin>. Last Access: June 2019.
- [51] Draw.io <<Draw.io tool>> [Online]. Available: <https://www.draw.io>. Last Access: June 2019.
- [52] Blog Job and Talent <<¿Cuáles son los sueldos de los ingenieros en España?>> [Online]. Available: <https://blog.jobandtalent.com/sueldos-de-ingenieros-espana/>. Last Access: June 2019.
- [53] Pymes y autónomos <<¿Cuánto paga tu empresa por ti? Quizá mas de lo que piensas>> [Online]. Available: <https://www.pymesyautonomos.com/fiscalidad-y-contabilidad/cuanto-paga-tu-empresa-por-ti-quiza-mas-de-lo-que-piensas>. Last Access: June 2019.
- [54] Ayuda Protección Datos <<Guía de Protección de Datos para desarrolladores de aplicaciones móviles>> [Online]. Available: <https://ayudaleyprotecciondatos.es/2016/06/06/normativa-lopd-aplicaciones-moviles/>. Last Access: June 2019.
- [55] Agenda de la empresa <<Requisitos legales que debe cumplir una App>> [Online]. Available: <https://www.agendaempresa.com/66355/requisitos-legales-que-debe-cumplir-una-app/>. Last Access: June 2019.
- [56] YeePLY <<Decálogo de buenas prácticas: Aspectos legales de las aplicaciones móviles>> [Online]. Available: <https://www.yeeply.com/blog/decalogo-de-buenas-practicas-aspectos-legales-de-las-aplicaciones-moviles/>. Last Access: June 2019.
- [57] Marina Rocca <<Cómo tener una App legal y segura>> [Online]. Available: <https://marinabrocca.com/app-legal/>. Last Access: June 2019.
- [58] Statista <<Tipos de dispositivos utilizados para conectarse a Internet en España en 2017>> [Online]. Available: <https://es.statista.com/estadisticas/478515/dispositivos-usados-para-acceder-a-internet-en-espana/>. Last Access: June 2019.

- [59] El Boletín <<El crecimiento del mercado de desarrollo de apps y la importancia de los “Marketplace”>> [Online]. Available: <https://www.elboletin.com/noticia/160761/hoy-en-la-red/el-crecimiento-del-mercado-de-desarrollo-de-apps-y-la-importancia-de-los-marketplace.html>. Last Access: June 2019.
- [60] Wikipedia <<Android TV>> [Online]. Available: https://en.wikipedia.org/wiki/Android_TV. Last Access: June 2019.
- [61] Wikipedia <<Android Auto>> [Online]. Available: https://en.wikipedia.org/wiki/Android_Auto. Last Access: June 2019.
- [62] Reality <<The Ultimate Guide to Understanding Augmented Reality (AR) Technology>> [Online]. Available: <https://www.realitytechnologies.com/augmented-reality/>. Last Access: June 2019.
- [63] Wikipedia <<Gantt chart>> [Online]. Available: https://en.wikipedia.org/wiki/Gantt_chart. Last Access: June 2019.
- [64] Wikipedia <<Google Assistant>> [Online]. Available: https://en.wikipedia.org/wiki/Google_Assistant. Last Access: June 2019.
- [65] Wikipedia <<Internet of Things>> [Online]. Available: https://en.wikipedia.org/wiki/Internet_of_things. Last Access: June 2019.
- [66] Wikipedia <<MySQL>> [Online]. Available: <https://en.wikipedia.org/wiki/MySQL>. Last Access: June 2019.
- [67] Wikipedia <<MySQL Workbench>> [Online]. Available: https://en.wikipedia.org/wiki/MySQL_Workbench. Last Access: June 2019.
- [68] Wikipedia <<Random-Access Memory>> [Online]. Available: https://en.wikipedia.org/wiki/Random-access_memory. Last Access: June 2019.
- [69] Interaction Design Foundation <<User Interface (UI) Design>> [Online]. Available: <https://www.interaction-design.org/literature/topics/ui-design>. Last Access: June 2019.
- [70] Wikipedia <<Virtual Reality>> [Online]. Available: https://en.wikipedia.org/wiki/Virtual_reality. Last Access: June 2019.
- [71] Google Forms <<Google Forms>> [Online]. Available: <https://www.google.com/forms/about/>. Last Access: June 2019.

