POLITECNICO
MILANO 1863

Software Engineering 2

AA 2018/19

Implementation and Testing Document

Authors:
Federico Ferri 10522586
Ahmad El Bayoumi 10425225

January 13, 2019

Contents

.

1 Introduction

1.1 Scope

In this document we present the implementation for the software "TrackMe". The main purpose of the application was already touched in the REQUIREMENT ANALYSIS AND SPECIFICATION DOCUMENT and into the DESIGN DOCUMENT (see Reference documents), this document will be therefore the used in order to show the implementation and to explain why such an implementation was chosen as well as showing the algorithms that were specifically designed for the application.

1.2 Document Contents

The document will be composed of five main sections:

- Requirements and Functions: in this section we will discuss what requirements are needed in order to run the software and the functions that were implemented.
- Adopted frameworks: inside this section we will include the information about all of the choices that we made.
- Code Structure: here we will include the structure of our source code and the logic behind the structure.
- Testing: here we will write how the testing was performed and all of the tools that were used to perform them.
- Installation Instructions: instructions needed in order to run the software

1.3 Revision History

Version 1.0 Released on January 13th,2018 - 9 pm

1.4 Reference Documents

- Mandatory Project Assignment AY 2018-2019.pdf

- Implementation and Testing Project Assignment
- https://firebase.google.com/docs/guides/?authuser=0

.

2 Requirements and Functions

2.1 Requirements

The system is built in order to run onto Android powered devices, therefore it was built with the use of Android Studio, in order to run an android application there are four requirements

- An APK (Android Package) of the TrackMe application, this is a file with the .apk format which is used for the distribution and the installation of components found onto the Android platform, it is a variant of the .jar (used for Java Applications)
- A device running an Android version with the API level 19 or above (Android version 4.4 KitKat)
- Access to Fine Location and Fitness History (Google FIT), it is required that the user allows the application to access GPS and network location and Fitness History, in case these informations are not allowed the users location and steps won't be tracked
- An account, in order to fully use the application there is a need for an account (it is possible to create it inside of the app)

2.2 Functions

2.2.1 Introduction

TrackMe is a software composed of the Data4Help service, the purpose of Data4Help is to provide consumers and businesses with the ability for the first (consumers) to track their health status,provide useful informations and accept/deny requests made by businesses, and for the last(businesses) to make request for information about health status, gender and age for group of individuals or single individuals(these will need to be allowed access by the target consumer).

2.2.2 Consumer's Functions

The consumer will be provided with the following functions

- Insert/Edit informations: the user will be able to insert informations and to change the information in case of a mistake, these information are: Name,Last Name,Date of Birth,CF(Fiscal Code),Phone Number
- Check Fitness Level: By inputting the Weight and the Height the user will be provided with his/her current BMI(Body Mass Index) and his current status, moreover by accessing the Fitness API the user will be presente with the steps that were taken on that particular day and with a Fitness Level provided with an algorythm create inside of the TrackMe application.
- Accept/Deny Requests: The user will be able to accept if his personal informations are shared with third parties once they are requested, moreover in case the third party wants to subscribe to the request the user will be able to know that that request is different from a "single use" one
- Share: The user will need to be able to share the application with frineds with a share button provided inside of the app
- Login/Logout: The user may Log In or Out of the app without any problem

2.2.3 Businesses' Functions

The Businesses will be provided with the following functions

- Insert/Edit informations: the business will be able to insert informations and to change the information in case of a mistake, these information are: Name,CF(Business Fiscal Code),Phone Number
- Make a Group request: The business will be able to access the informations of a group of people by filtering them by age and location, in order to do so the system will check if the informations come from a group sized enough to respect the consumers' privacy (in the Testing implementation this size is equal to 1 but on the final version it should be raised to 1000 people)
- Make a Private request: The business will be able to ask for access to informations regarding a single individual provided that they know

his Fiscal Code, moreover the business may select to subscribe to the informations and it will be able to permanently receive that specific user's informations.

- Check past requests: In case the request was approved the businesses will need to be able to access the data for one single time unless they asked to subscribe (in this case the business may ask for data an unlimited number of times)
- Log-in/Log-out: The business may Log In or Out of the app without any problem

3 Adopted Frameworks

3.1 Introduction

In this section we will discuss all of the frameworks that we used in order to develop the final version of our application, this section will be divided in three parts:

- **Programming Languages**: here we will discuss which programming languages where used in the development process and why according to us they where the best possible choices.
- **External APIs**: in this part we are going to discuss about all of the APIs that were used inside of the application and why they were used

3.2 Programming Languages

In order to develop the final version of **TrackMe** we used three programming languages for the development:

- **Java** : In order to program all of the "logic" part of the software we used JAVA 8 since, in order to properly develop an Android application, it is regarded as the best programming language for this purpose since it is fully compatible with Google's APIs (our main source for external APIs, this will be discussed in the following section) and it is the main language of choice for developing Android powered applications being fully integrated into the Android Studio IDE. Java and Android Studio

make sure that our application is native to Android and therefore it is much more robust and has a grater performance than any other kind of application (for example a Web App).

- **XML**: XML (eXtensible Markup Language) is a markup language that can be used for many purposes but it's main use is to build Graphical User Interfaces (**GUI**). In our case we decided to use XML in order to build the entire GUI of our application since it is the best language for this purpose because it is fully integrated with the Android platform and it is well integrated with JAVA . In particular we used two kinds of layouts that are native of Android: **Constraint Layout** (This kind of layout is used to make sure that the application will run smoothly with different devices and resolutions since each element is "constrained" to the others and distances between them are flexible in order to guarantee the same layout in each device), **Linear Layout** (This layout is used to separate each element in a table like environment, it is very useful to create a clean look in particular cases)

- **JSON**: JSON (JavaScript Object Notation) is a notation language and we used it for three main purposes:

  - To edit auto-generated files by Android Studio

  - For Testing purposes, in order to provide Google's robotic script with inputs to go through our custom made log-in page and Test our application for eventual problems related with bugs and performances

  - To design our database, we decided to use a NoSQL database since it gives us a greater flexibility to work with objects and to interact with the database, which is something that totally fitted what we were looking for, moreover since the database was provided by Google it was perfectly implemented with the Android platform.

3.3 External APIs

In order to smoothly run the project and save lots of time we used external APIs to perform actions that were already created in a separate environment we used external APIs, as a provider for those APIs we selected Google since it is one of the companies that offers a great variety of services that fitted our demands in the following lines we will explain one by one the APIs that we used:

- **Google Maps API**: The Google Maps API is part of the Google Cloud Platform and it is the leading company for maps related informations, we decided to use this API in order to locate a particular point in the map, when the user makes a request and looks for a particular address the API will analyze that address and will return the coordinates for that specific point
- **Fitness API**: the Fitness API is provided by Google Fit in particular we used the Fitness History API to retrieve data about steps that were taken by the user on a specific time period, we use this data in order to understand the fitness level of the user
- **Firebase Auth**: Firebase Auth is the API used to manage all user logins and registrations it encrypts the passwords and keeps them hashed, moreover it is extremely safe since it is kept separate from the database and therefore adds an extra layer of security to the system.
- **Firebase Firestore**: Firebase Firestore is the API used to access and make requests to the database, the database is NoSQL and is based on JSON we use the API through the "DAO" class inside of our application to update, remove and add elements to our database

4 Code Structure

4.1 Introduction

In this part we will discuss the structure of the code and why it was structured in such a way. Moreover in the final part we will discuss the differences with what was written in the Design Document and why it was structured in a different way. The source code is available for download at the following link: ***Source Code***

4.2 Classes

All of the Java Classes found inside the application can be found under the following path: " /app/src/main/java/com/trackme/trackme", this is because the Android App is developed to be all underneath one single package, in the package there are 12 classes composed of:

- 1 Database Handling Class
- 3 Activities
- 6 Fragments
- 2 Classes for implementing Database with the Activities and Fragments

Now we will discuss all of the classes and the xml files linked to them, the rest of the project is generated automatically by the Android Studio IDE in order to succesfully build the app

4.2.1 DAO

This is the core class that handles the interaction with the database, in fact all the queries to the DB are done and handled by this Data Access Object. Here we have two main fields: FirebaseAuth and FirebaseFirestore. The first one purpose is to deal with the authentication service of the Firebase system, such as Login, Signup etc. The FirebaseFirestore field instead, is the only object that has really to deal with the database data: its function is doing all of the query necessary to the application. The purpose of this class is separating the tasks of the application by delegating to it all the tasks of the data access layer.

4.2.2 BaseActivity an BaseFragment

These are the two super classes that all of our Android Activities and Fragments extend. Both are simple classes and similar to each other, in fact they only provide to activities and fragments the DAO object for approaching the db data.

4.2.3 Fitness Level

This fragment's layout is described in the fragmentfitnesslevel.xml file. This view provides two EditText controls that ask for weight and height, and, provided these two values, it calculates the BMI index (Body Mass Index) with a short feedback of the meaning of this value and hints to improve the health level of the user. In addition to this, thanks to the Google Fit API's, the application retrieves the steps of the day made by the user and, with this value and the BMI value as inputs, it calculates with an algorithm the Fitness Level of the user.

4.2.4 Group Request

This fragment, described in the fragmentgrouprequest.xml file, is available only for the business users. Its function is to allow the user to ask for the average information of an anonymized group of users. It consists in a form that takes in input four filter values: minimum age, maximum age, an address and a radius from the address. Once the user provides the filters and clicks the SEARCH button, the application query the database, and if the number of users returned by the query is greater than a setted threshold (the purpose of this threshold is to ensure that the users information will be anonymous), it shows to the user the average of: BMI, WEIGHT, HEIGHT, STEPS AND AGE.

4.2.5 Login Activity

If in the MainActivity you click on the link to login, you will be redirected to this LoginActivity that asks only for email and password needed for the authentication. This activity is bound with the activitylogin.xml layout file.

4.2.6 Main Activity

This is the first activity that our app launches and its bound with the activitymain.xml file . Here we find the subscription form and a link to the LoginActivity if already have an account. As for every activity, the main method is the onCreate. It function is to set up the view and bind every widget and control of the XML view to its corresponding Java object.

4.2.7 Past Requests

In this final fragment the business user has the possibility to show the information of the users that have accepted his request by inserting their fiscal code. The layout file is called fragmentsettings.xml

4.2.8 Requests

In this last basic user Fragment, we have the list of the pending request made to the user and waiting for a response. The user can accept or deny them. The graphic of this fragment is provided by the fragmentrequests.xml file.

4.2.9 Single Request

In this fragment the business user has the possibility to search for a user by his fiscal code ("Codice fiscale") and to send a request to him. This request can be a "once-time" request or a subscription to the user. In the first case, if the user accepts the request then the business user can view the user's information only one time, the subscription instead allows to view the information every time he wants. The layout of this section is very simple as described in its layout file (fragmentgrouprequest.xml), since it has only and EditText control to insert the fiscal code and an Android Toast that gives a feedback of the operation.

4.2.10 User

Once you have successfully authenticated, you will see the User navigation activity with the graphic described by the activityuser2.xml file. From here you can access to all of the links of the navigation bar of the activity. These links, excluded the "Log out", the "My informations" and the "Share" ones that are available to all users, depend on the type of your user profile: if business or basic user. If you are a basic user than you have other 2 choices: My Informations, My Fitness Level, My Requests. If instead you're a business user you find: New Public Request, New Private Request, My Private Requests. Every choice of these is an Android Fragment that inherits from BaseFragment.

4.2.11 User Info

This fragment is bound to the fragmentuserinfo.xml layout file. It shows the logged in user's information by retrieving them from the database and allows to the user the possibility to edit them.

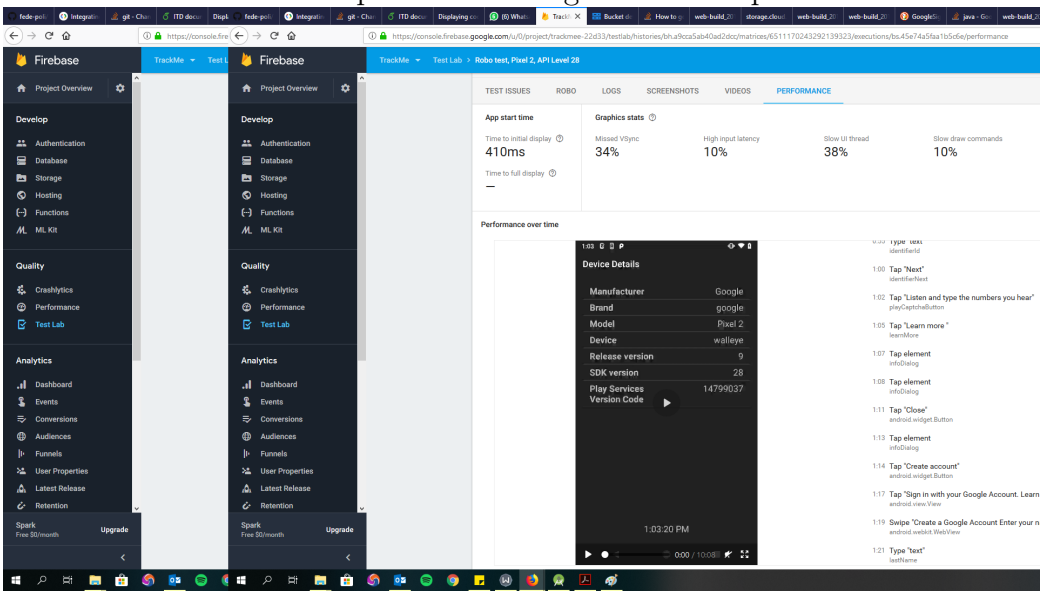
5 Testing

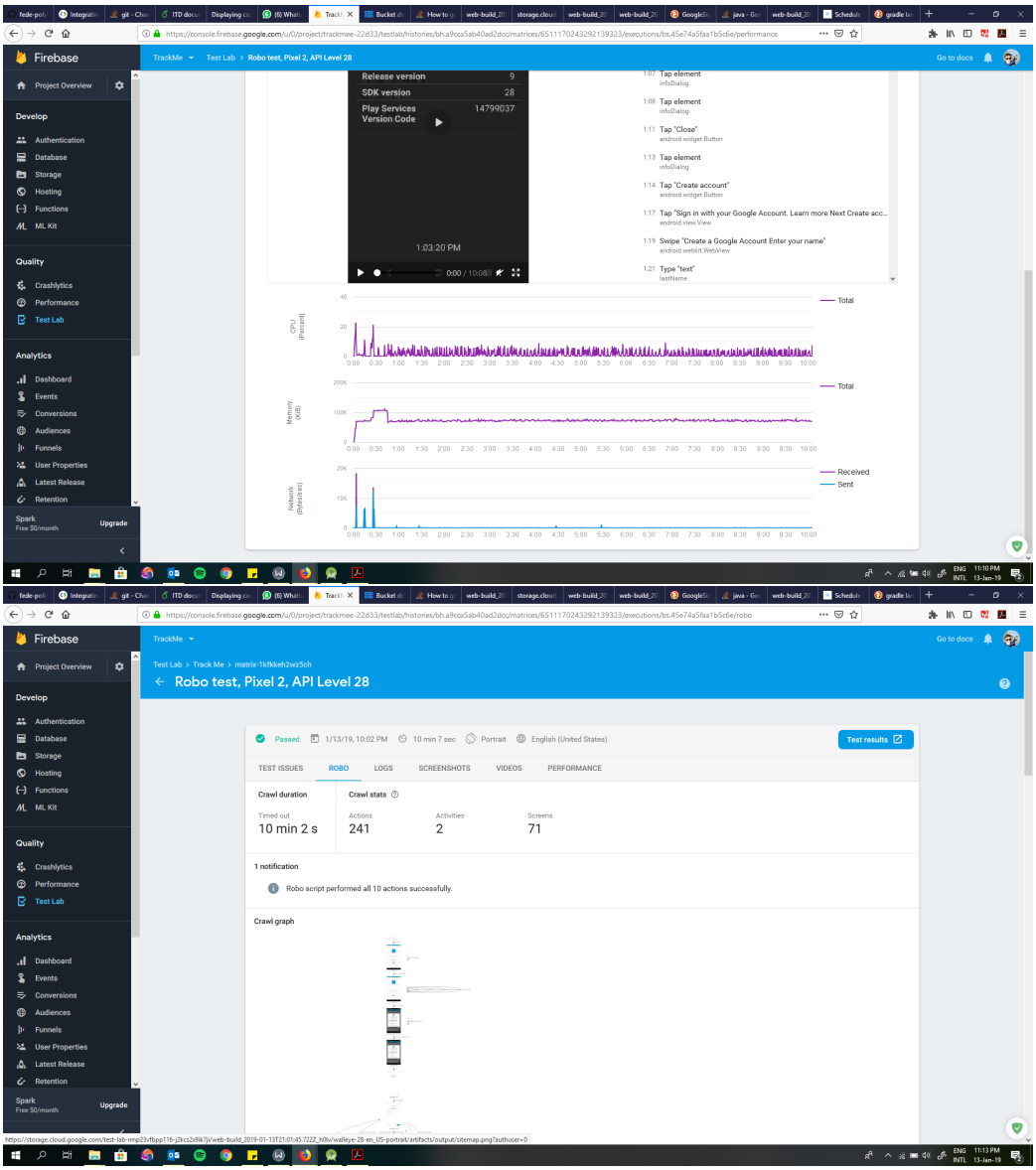
5.1 Introduction

This part will focus on the testing part of the implementation for which we used Google's Robo Testing which is am automated testing tool that tests the performance, the usability and that detects eventual issues and errors.

5.1.1 Google's Robo Testing

Our software passed the Google's Robo test both with physical and virtual devices and uses an Artificial Intelligence to mimic all of the possible combinations that can be performed on the app, all the results (the app passed them all) can be found here:
Robo Output
Moreover we added some pictures of the generated output:





6 Installation

The installation of the app is extremely easy, the user just need to download the .apk file from the following link (apk is a variant of the most known .jar

file, the executable file of JAVA, the apk is nothing else than the built source code) and tap on it on his/her Android device the system will then take him on the extremely easy process of installation.
Click here to go to the .apk file

7 References

Tools that were used to create this document:

- overleaf (online LaTeXeditor)

Other references:

- https://developers.google.com/fit/
- https://firebase.google.com/docs/guides/?authuser=0
- https://cloud.google.com/maps-platform/ (Google Maps API)
- https://developer.android.com/studio/ (Android Studio)