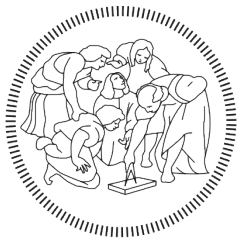

POLITECNICO
MILANO 1863

ATD

Software Engineering 2 Project - TrackMe

v1.0 - 20/01/2019

Authors

- Daniele Montesi - *912980*
- Nicola Fossati - *915244*
- Francesco Sgherzi - *915377*

Contents

1 Introduction

1.1 Purpose

Acceptance Test Document (ATD) is a document whose purpose is to validate and test a project implemented by another group of the Software Engineer 2 course.

1.2 Scope

We are going to test the TrackMe project.
TrackMe is a company that wants to offer the following services to individuals and to third party companies:

- Data4Help, that allows people to register and provide their data to TrackMe through devices such as smartwatches or smartphones and allows companies to get data from a particular group of people;
- AutomatedSOS, that allows individuals, mainly elderly people, to register in order to have their health monitored and to receive an immediate assistance in emergency cases;
- Data4Run, a sport-oriented service that tracks athletes participating in a run. It allows organizers to define the path of the run, participants to enroll to the run, and spectators to see on a map the position of all runners during the run. Track4Run will exploit the features offered by Data4Help

The authors of the project are

- Riccardo Poiani
- Mattia Tibaldi
- Tang-Tang Zhou

A link for their repository on Github if can be found here: TrackMe repository.

1.3 Document structure

The document is divided into the following sections:

- **Installation setup:** in this section the procedures and the steps followed in order to install and test the project will be discussed and explained, focusing on problem and incoherences found in the provided documents.
- **Acceptance test cases:** in this section all the tests run against the project will be discussed, relating them with the requirements / system goals described in the RASD and in the DD.

2 Installation setup

We have followed the guide in the IT Document. The guide was explained in details, even though, we had troubles in testing the system into a unique machine because of RAM limitation. Probably it would have been better to have an instance of the server and of the database running online.

2.1 Dependency installation

We firstly installed the backend. To accomply this, on a Windows machine, you should first install ERLang and then install the RabbitMQ server.

After this, we proceed to install MySQL server and configure it following the guide linked in the IT Document.

The RabbitMQ server and MySQL server started automatically after the installation.

It is not clearly specified in the IDT but is also necessary to install a compatible version of Java and Maven (to run the tests). We installed, respectively, version `1.8.0_191` and version `3.6.0`.

2.2 Backend installation

We simply extracted the precompiled version of the backend in an empty folder. From there we launched each microservice executable, starting from `service-registry` and then all the other jar executables.

Each service was launched using the command `java -jar *servicename*.jar`, except the `api-gateway` which needs an additional command line parameter. We think that this aspect was not stressed enough in the ITD; in fact in the first place we were not able to start the server correctly because we forgot the above mentioned configuration string.

2.3 Frontend installation

We installed the main Android app from the provided apk package in a new Android Emulator instance. We then started the app, configured the IP of the server clicking on the app logo in the main screen.

To test the Bluetooth connection we installed the main app and the Bluetooth client in two different real devices.

The minimum Android version, as specified in the requirements, was 7.0 which made quite difficult to test since some of us had the version 6.0. Moreover the app cannot be fully tested through the emulator since it exploit the Bluetooth connection, which cannot be emulated. However, all the previous remarks were correctly explained in the ITD Document.

3 Acceptance Test Cases

The group has divided the requirements into:

- Core requirements
- Goals and side requirements mapped

Here they are listed as reported in the RASD document:

Core requirements

[R1] Allow a person to register into the system as a user by providing a username, a password, his credentials, his social security number and a consensus on the agreement

[R2] Allow a person or company to register into the system as a third party by providing an e-mail, a password, its credentials and a consensus on the agreement

[R3] A person cannot register as a user by inserting a username that has already been used in another successful registration process

[R4] A person cannot register as user by using the same social security number used in another successful registration process

- DELETED [R5] A person cannot register as user by using the same credential used in another successful registration process
- DELETED [R6] A person or company cannot register as third party into the system by using the same credentials used in another successful process

[R7] A person or company cannot register as third party by using the same e-mail used in another successful registration process

[R8] Allow a person to log into the system as a user only if he has already registered as such

[R9] Allow a person or company to log into the system as a third party only if he has already registered as such

The goals and mapped requirements are listed below:

Goal reaching requirements

[G1] Allow a user to access its own data

- [R10] If a user is logged-in, he is able to view its own data

[G2] Allow a user to contribute to data sharing by providing information about his location and health status

- [R11] Allow a user to send its data to the system automatically when it is generated

[G3 & G4] Once the health parameters of a user have been observed below the threshold for the first time after one hour, an ambulance is sent to the user location. The time experienced between the moment in which the health parameters of a subscribed user are observed below the threshold and the time in which the emergency point is contacted is equal or less than 5 seconds

- [R12] When a user's health parameters has been observed below the threshold, an SOSCall is requested within 5 seconds
- [R13] All the automated SOS call are performed with devices of users whose health parameters are observed below a certain threshold
- [R14] An SOSCall can be requested only every minute
- [R15] An SOSCall is blocked if a previous one has already been accepted within one hour
- [R16] An SOSCall is implemented as an automated call by using an external API
- [R17] During an SOSCall, the GPS is set on high-precision

[G5] Allow a user to participate in a run managed by third parties, as an athlete, if all starting conditions are satisfied.

- [R18] Allow a user to view a list of available runs, i.e. those that are still waiting to start
- [R19] Allow a user to enroll in a run, after choosing it from the list, only before the expiration date by specifying his nickname
- [R20] Allow a user to unsubscribe from an enrolled run only before the expiration date
- [R21] If, after the expiration date, the number of participants is less than the minimum number defined by the organizer, then it is impossible to start the run.

[R22] If the run cannot start due to minimum number of participants unsatisfied, then the enrolled runners are notified.

[G6] Allow spectators (i.e. user) to see on real-time the "correct" positions of all athletes taking part in a run, with at most 15 meters of radius error

[R23] Every athlete participating on a run, only on this specific occasion, shares continuously (i.e. each ten seconds) its position through a device

[R24] Allow a spectator to see on a map the real-time position of every athlete in a specific run

[G7] The maximum time to accept an individual request from any third party is 30 days; after that, the request will expire

[R25] Once the time elapsed from sending request is greater than 30 days, then the request will be deleted from the system

[G8 & G9] Allow a user to accept or refuse a request from third parties. Allow a user to block requests made by a specific third party and all the pending requests will be refused: this action is possible only when the user has already refused one request made by the customer that he is intending to block.

[R26] Allow a user to receive individual requests about data sharing from third parties

[R27] Allow a user to view a list of pending requests

[R28] Allow a user to accept or refuse a request from the list of pending request

[R29] Allow a user to block requests from a defined third party, after having refused a request made by the customer involved in the block.

[R46] When a request it is blocked, the other requests sent by the same Third party to the same user are refused.

[G10] Allow spectators and runners to see the leaderboard, when a run is completed

[R30] Allow an organizer to close the run (when it is terminated)

[R31] After a run is closed, the leaderboard is shown to the spectators and runners

[R32] After a day is elapsed from the date of the race, if the run is not closed the application will automatically close it.

[G11] Allow organizers (i.e. third parties) to set up a run, by defining its name, its path, date, start time, expiration date, and the minimum number of participants

- [R33] Allow a third party to see a map of the world with feasible paths
- [R34] Allow a third party to publish a race by providing an unique name, a feasible and a non-overlapping path (non-overlapping with other races of the same date), a date, a start time, a brief description, an expiration date for subscription and a minimum number of participants
- [R45] Allow a third party to manage its run by giving him a list of its managed race

[G12] Allow a third party to access data specified in a request if the user accepts the request or if he accepted one or more requests from the same third party that provided access to the same data

- [R35] If an individual request is accepted, then the third party who has made the request can access the data specified in the request
- [R36] For each piece of individual data accessible by a third part customer, exists an accepted request regarding it, performed by the same third party
- [R37] Allow a user to accept or refuse request given by third parties
- [R38] Allow a third party to send individual requests to users by providing the user's social security number and a brief motivation

[G13] Allow a third party to access statistical and anonymized data if and only if the number of individual involved is greater than 1000. This is satisfied after the request is approved

- [R39] A group request is accepted if the aggregated data specified in the request is accessible to the third party who performed the demand
- [R40] Group requests are accepted if and only if the number of user involved is greater than 1000
- [R41] Aggregated data is accessible to a third party if an accepted aggregated data that request that data exists
- [R42] Allow a third party to send group request to the system regarding data about many users

[G14] Allow a third party to subscribe to non-existing data. They will have access to them, after the data is generated.

- [R43] Allow a third party to express a data request on future data

[R44] A third party can have access to non-existing aggregated data regarding future information if and only if the number of people involved will be greater than 1000

3.1 Test cases

3.1.1 Core requirements

Requirements	Notes
R1	Ok, tested opening the app and clicking "Register" button and filling up the form in "User".
R2	Ok, tested opening the app and clicking "Register" button and filling up the form in "Third Party".
R3	Ok, but wrong error description.
R4	Ok, by registering an user with the same SSN of a previous one.
R7	Ok, we tried to register a Third Party with the same email of another registered Third Party.
R8	Ok, tested creating new accounts and then trying to log in.
R9	Ok, tested creating new account and then trying to log in.

3.1.2 G1: Access to user's data

Requirements	Notes
R10	Ok, tested after logging in with a user and sending data via the companion bluetooth app with another smarthpone. Then the user can click to "History" tab and see all acquired data.

3.1.3 G2: Load data automatically into the server

Requirements	Notes
R11	Partially, tested logging in using another device with the same username and password of an already existing user and seeing data being in the history. However, location data are not available.

3.1.4 G3 & G4: Ambulance call

Requirements	Notes
R12	Ok, tested inserting data lower to the treshold (1 bpm). The call is performed almost instantaneously.
R13	Ok, identical to R12, tested inserting data lower to the treshold (1 bpm). The call is performed almost instantaneously.
R15	Ok, tested after a call, tried to send twice the same data lower the treshold after 30m, but the call didn't work. Trying after more than one hour results in a correct call being dispatched.
R17	Automatically done by the Android OS when performing an emergency call.

3.1.5 G8 & G9: User manages a request from third party

Requirements	Notes
R26	Ok, tested generating a new individual request.
R27	Ok, tested after having generated a requests and checking the pending request section.
R28	Ok, tested after having generated a requests and then letting the user reject it.
R29	Ok, tested after having generated a requests and then letting the user reject it. After that the Third Party isn't allowed to send individual requests to the user anymore.
R46	Ok, tested like R26 and R29.

3.1.6 G12: Third party access to a specific user data

Requirements	Notes
R35	We couldn't test it as the app stopped responding when tapping on the "Download" icon. However, after contacting the developers they promptly released an update for the app that allowed us to download users' data. Still, the data concerning the position of the user aren't available.
R36	Like R35. However, after the update, we checked that this consistency constrain is maintained

R37	Ok, tested by sending an individual request for data and letting the user accept or refuse it. The Third Party user correctly displays the status of the request
R38	Ok, tested by sending inserting, in the body of the request, the correct SSN of the user and a motivation. User correctly sees the request and the motivation and is able to accept or refuse it.

3.1.7 G13: Company access to group of anonimyzed data

Requirements	Notes
R39	Ok, tested performing a query.
R40	Ok, tested by performing a request on an almost empty database and testing that the query was rejected.
R41	Ok, tested performing a query on an almost empty database and testing that the data weren't accessible by the company (that is, that specific item shows a tag REFUSED and it is not possible to download the data as the "Download" icon is missing)
R42	Ok, tested by performing a query on an almost empty database. Even if the query was rejected (wasn't possible to download users' data) it was still saved.

3.2 Unit and integration tests

3.2.1 Android app

We also run other group's tests against the Android application in emulator, as described in the Installation section of the ITD. We were not able to run all the tests correctly, even after contacting the other group for clarifications, possibly due to some configuration mismatch in the emulator.
Tests in 'com.trackme.trackmeapplication': 24 total, 4 error, 20 passed

localdb.database.AppDatabaseTest

Test	Result	Time
insertEmergencyCall	passed	105 ms
insertPositionData	passed	103 ms

insertPositionDataFail DueToNull-Value	passed	138ms
getNumberOfRecentCalls	passed	127 ms
deleteAllPositionData	passed	153 ms
deleteAllHealthData	passed	103 ms
deleteAllRecentCalls	passed	128 ms
insertHealthData	passed	52 ms
insertHealthDataFail DueToNull-Value	passed	77 ms

service.health.HealthDataCallbackTest

Test	Result	Time
handleMessageCorrect WithRecentE-mergencyCall	passed	180 ms
handleMessageThrow InvalidHealth-DataException DueToLocation	passed	75 ms
handleMessageThrow TimeExcep-tionDueToLocation	passed	154 ms
handleMessageThrow InterruptedEx-ceptionDueToLocation	passed	102 ms
handleMessageCorrect WithSuccess-Call	passed	103 ms
handleMessage ThrowEmergen-cyNumberNotFoundException Due-ToLocation	passed	76 ms
handleMessageGoodHealthData WithoutRecentEmergencyCall	passed	77 ms
handleMessageGoodHealthData WithRecentEmergencyCall	passed	78 ms

HealthServiceHelperImplTest$TestWithPermissionGranted

Test	Result	Time

makeEmergencyCall	failed: java.util.concurrent.TimeoutException: No last location available	3.0 s

HealthServiceTest$IntegrationTesting

Test	Result	Time
testDoubleConsecutiveCall	failed: java.lang.IllegalArgumentException: Provider "gps" unknown	2.75 s
testMakeCallPerformance	failed: java.lang.IllegalArgumentException: Provider "gps" unknown	2.34 s

HealthServiceTest$TestWithPermissionGranted

Test	Result	Time
getUserLocation HasLastLocation	failed: java.util.concurrent.TimeoutException: No last location available	2.79 s
getEmergencyRoom NumberWrongCountryCode	passed	3.48 s
getEmergencyRoom NumberForItaly	passed	2.97 s
getEmergencyRoom NumberForAfghanistan	passed	2.97 s

Tests util in app: 29 total, 29 passed

HealthDataInspectorImplTest

Test	Result	Time
isGraveConditionInvalidData DueToNullValue1	passed	4 ms
isGraveConditionInvalidData DueToNullValue2	passed	1 ms
isGraveConditionInvalidData DueToNullValue3	passed	0 ms
isGraveConditionInvalidData DueToNullValue4	passed	0 ms
isGraveConditionInvalidData DueToNullValue5	passed	1 ms
isGraveConditionInvalidData DueToNullValue6	passed	0 ms
isGraveConditionInvalidData DueToNullValue7	passed	0 ms
isGraveConditionInvalidData DueToNullValue8	passed	0 ms
isGraveConditionInvalidData DueToNullValue9	passed	0 ms
isGraveConditionInvalidData DueToBloodOxygen LevelLess0	passed	0 ms
isGraveConditionInvalid DateSinceTimestamp BeforeBirthDate	passed	1 ms
isGraveConditionInvalid DataDueToPressure	passed	0 ms
isGraveConditionInvalid DataDueToBloodOxygen LevelGreater100	passed	0 ms

HealthDataInspectorImplTest$ParameterTest

Test	Result	Time
isGraveCondition HeartBeatProblem[0]	passed	0 ms
isGraveCondition HeartBeatProblem[1]	passed	0 ms
isGraveCondition HeartBeatProblem[2]	passed	0 ms

isGraveCondition	HeartBeatProblem[3]	passed	0 ms
isGraveCondition	HeartBeatProblem[4]	passed	0 ms
isGraveCondition	HeartBeatProblem[5]	passed	1 ms
isGraveCondition	HeartBeatProblem[6]	passed	0 ms
isGraveCondition	HeartBeatProblem[7]	passed	0 ms
isGraveCondition	HeartBeatProblem[8]	passed	0 ms
isGraveCondition	HeartBeatProblem[9]	passed	0 ms
isGraveCondition	HeartBeatProblem[10]	passed	0 ms
isGraveCondition	HeartBeatProblem[11]	passed	0 ms
isGraveCondition	HeartBeatProblem[12]	passed	0 ms

ThresholdIntegerTest

Test	Result	Time
ThresholdIntegerTest.contains CheckBounds	passed	0 ms
ThresholdIntegerTest.construct ThresholdIntegerMaxLessThanMin	passed	0 ms
ThresholdIntegerTest.construct ThresholdIntegerMaxEqualMin	passed	0 ms

3.2.2 Backend

All test ran successfully, however, due to the sheer amount of them, being 307 in total, we decided to present their results grouped by microservices.

Test	Result	Time
`api-gateway`	89/89 passed	5.01min
`grouprequestservice`	38/38 passed	2.33min

`individualrequestservice`	86/86 passed	5.36min
`sharedataservice`	94/94 passed	5.07min

4 Additional Notes

We would like to point out the following remarks:

- The code (of the backend and the app) looks well commented and explained.
- The immediate solution to the problem we faced concerning the download of individual data showed the great responsability and will to collaborate of the tested group.
- The group showed great disposability in answering our questions promptly and detailedly.

5 Effort spent

Date	Nicola Fossati	Daniele Montesi	Francesco Sgherzi
14/01/2019	0	0	1
17/01/2019	0	3	3
18/01/2019	6	5	3
19/01/2019	3	1	5
20/01/2019	1	1	3
Total	9	10	15