

# TrackMe

Version: 1.0

Antonio Urbano  
Enrico Voltan

Release date: 11/11/2018

# Table of Contents

<b>1. Introduction.....</b>	<b>5</b>
1.1 Purpose.....	5
1.1.1 Goals.....	5
1.2 Scope.....	6
1.3 Definitions, Acronyms, Abbreviations.....	8
1.3.1 Definitions.....	8
1.3.2 Acronyms.....	8
1.3.3 Abbreviations.....	8
1.4 Document Structure.....	9
<b>2. Overall Description.....</b>	<b>10</b>
2.1 Product Perspective.....	10
2.2 Product Functions.....	11
2.3 User Characteristics.....	13
2.4 Assumptions, Dependencies and Constraints.....	14
2.4.1 Assumptions.....	14
2.4.2 Dependencies.....	14
<b>3. Specific Requirements.....</b>	<b>15</b>
3.1 External Interface Requirements.....	15
3.1.1 User Interfaces.....	15
3.1.2 Hardware Interfaces.....	21
3.1.3 Software Interfaces.....	21
3.2 Scenarios.....	22
3.2.1 Scenario 1.....	22
3.2.2 Scenario 2.....	22
3.2.3 Scenario 3.....	22
3.2.4 Scenario 4.....	23
3.2.5 Scenario 5.....	23
3.3 Functional Requirements.....	24
3.3.1 Use Case Diagram.....	27
3.3.2 Use Cases.....	27
3.3.3 Sequence Diagrams.....	34
3.3.4 Mapping on requirements.....	37

3.4 Performance Requirements.....	38
3.5 Design Constraints.....	39
3.5.1 Regulatory policies.....	39
3.5.2 Hardware Limitations .....	39
3.6 Software System Attributes.....	40
3.6.1 Reliability.....	40
3.6.2 Availability.....	40
3.6.3 Security.....	40
3.6.4 Maintainability.....	40
3.6.5 Portability.....	40
<b>4. Formal Analysis Using Alloy.....</b>	<b>43</b>
4.1 Alloy constraints.....	43
4.2 Alloy Model.....	44
4.3 Alloy World Generated.....	50
4.4 Alloy Results.....	53
<b>5. Effort Spent.....</b>	<b>54</b>
<b>6. References.....</b>	<b>54</b>

# 1. Introduction

## 1.1 Purpose

This document represents the Requirement Analysis and Specification Document (RASD).

Goals of this documents are to completely describe the system in terms of functional and non-functional requirements, analyse the real needs of the customer in order to model the system, show the constraints and the limit of the software and indicate the typical use cases that will occur after the release.

Furthermore, it provides a description of the two software-based services offered by TrackMe (Data4Help and AutomatedSOS).

This will be done by a detailed presentation of the proposed solution and its purpose, listing its goals, and the requirements and assumptions through which they will be achieved. In particular:

- **Data4Help** is designed as a software application used for allowing third parties to monitor the location and health status of some specific individuals or to access to anonymized data of large groups of individuals.  
Moreover, individuals agree that TrackMe acquires their data (through smartwatches or similar devices) by registering to the service
- **AutomatedSOS** is a service based on Data4Help designed to monitor the health status of the subscribed customers and, when such parameters are below certain thresholds, to send to the location of the customer an ambulance, guaranteeing a reaction time of less than 5 seconds

This document is meant to be used by any person who might be interested in this project, both end users (private customers or third-parties) and developers who have to implement the requirements.

### 1.1.1 Goals

- [G1]: Allow individuals to become registered users of Data4Help
- [G2]: Allow users to sign up for AutomatedSOS service
- [G3]: Provide the registration to third parties who want access to users' data
- [G4]: Give third parties access to users' data
  - [G4.1]: Give third parties access to data of a specific user
  - [G4.2]: Give third parties access to anonymized data of group of users
- [G5]: Allow users to accept or refuse the requests from third-parties to access their own data and their location

- **[G6]:** Allow third parties to subscribe to new data and to receive them as soon they are produced
- **[G7]:** Allow customers to insert or update their own personal data and information about their body measurements (e.g. weight, height, etc.)
- **[G8]:** If some parameters of health status are below certain threshold, send an ambulance to the user location, with a reaction time less than 5 seconds
- **[G9]:** Allow users to keep track of their health status at any time
- **[G10]:** Allow users to withdraw the authorisation to third parties to access their data

## 1.2 Scope

The aim of this section is to analyse the services offered by TrackMe in more detail. As already described in the Purpose part, TrackMe is a software that offers 2 different services:

- **Data4Help:** the service is designed to monitor and store the location and health status of individuals, acquired by smartwatches or similar devices, and share these data with third parties who request them. These stored data can be accessed by third parties only if they are authorized. In particular
  - **Individual request:** Third parties who want to access data of some specific user know his/her individual SSN or Fiscal Code.  
TrackMe passes the request to the specific individuals who can accept or refuse it. Finally, the information required is given to the third party only if the individual has authorized them, accepting previously the request. Otherwise third party is notified about the denial of the request.
  - **Anonymous group data requests:** Third parties can also access to anonymized data of groups of individuals. These requests are handled directly by TrackMe that approves them if it is able to properly anonymize the requested data. TrackMe will accept any request for which the number of individuals whose data satisfy the request is higher than 1000.

Individuals can become registered users after providing credentials, personal data and accepting the terms of use of Data4Help.

Furthermore, every individual can view the stored data at any time to keep track of their health status and can update his own personal data, like weight, height, etc.

Third parties will also be allowed to subscribe to certain data in order to receive them as soon as they are produced.

Finally, at any time users can withdraw the authorisation previously given to third parties to access their data.

- **AutomatedSOS:** is built on top of Data4Help. It monitors the health status of the subscribed users and, when such parameters are below certain thresholds, sends a notification, with all the necessary information, to an external Ambulance Service, guaranteeing a reaction time of less than 5 seconds from the time the parameters are below the threshold

## 1.3 Definitions, Acronyms, Abbreviations

### 1.3.1 Definitions

- **Request:** an entity created by Third-Party to require Users' Personal Data.  
There are "single" or "group" types (*see section "2.3" for definitions of Third-Party and User*)
- **Personal Information:** a set of data (first name, age, gender, height, etc.) concerning User inserted by himself
- **Health Status:** a set of parameters concerning health of the Users (blood pressure and bpm)
- **Threshold values:** limit values for each parameter considered for valuating Users' Health Status
- **Personal data:** the latest update of both Personal Information and Health Status of each User. These are the data provided to Third-Party's requests
- **Authorization:** an action carried out by the Users to give access their data to Third-Party
- **Square Area:** a geographical constraint that a Third-Party can add to a group Request. It indicates two limits by which the Users concerning the Group Request are located
- **Age Range:** it indicates two limits concerning the Users' age involved in the group Request.

### 1.3.2 Acronyms

- **RASD:** Requirement Analysis and Specification Document
- **API:** Application Programming Interface
- **GPS:** Global Positioning System
- **Bpm:** Beats Per Minute
- **SSN:** Social Security Number

### 1.3.3 Abbreviations

- **[Gn]:** n-th goal
- **[Dn]:** n-th domain assumption
- **[Rn]:** n-th functional requirement
- **Tp-a:** Third-Party administrator
- **UC.n:** n-th use case

### 1.3.4 Documents Structure

This document is divided into six sections:

- In the first section it is possible to find the introduction with the scope and the purpose of the TrackMe project.
- In the second section the product is described more in depth, with a definition of its functions, perspective, users and constraints.
- In the third section, interfaces and requirements are described and mock-ups of what the product will be likely to look like are presented.

Some scenarios will help understand better the case in which TrackMe can be used.

Furthermore, the functional requirements are defined by using use case and sequence diagram. The non-functional requirements are defined through performance requirements, design constraints and software system attributes.

- The fourth section contains a formal analysis supported by using “Alloy model” and the discussion of its purpose. Finally, a world generated by it is shown.
- Chapter 5 shows the effort spent by each group member while working on this project.
- Chapter 6 includes the reference documents



## 2. Overall Description

### 2.1 Product Perspective

The software to be developed will offer to Customers a set of APIs used to embed TrackMe services in addition to a mobile and a web application. In particular:

- **Users** can interact only with the mobile application
- **Third-Party** can interact only through the website

The mobile app can track the health status of every registered User by using native health app installed on Users' smartphone which is connected to a device (like a smartwatch or a similar device) that can monitor some specific parameters, like blood pressure and bpm.

The mobile app also deals to keep track the GPS current location of every User.

Once logged into the mobile application, each User can monitor his own measured data, set a range of personal information and activate AutomatedSOS service.

The mobile app finally includes a dedicated section where he can manage the requests from Third-parties.

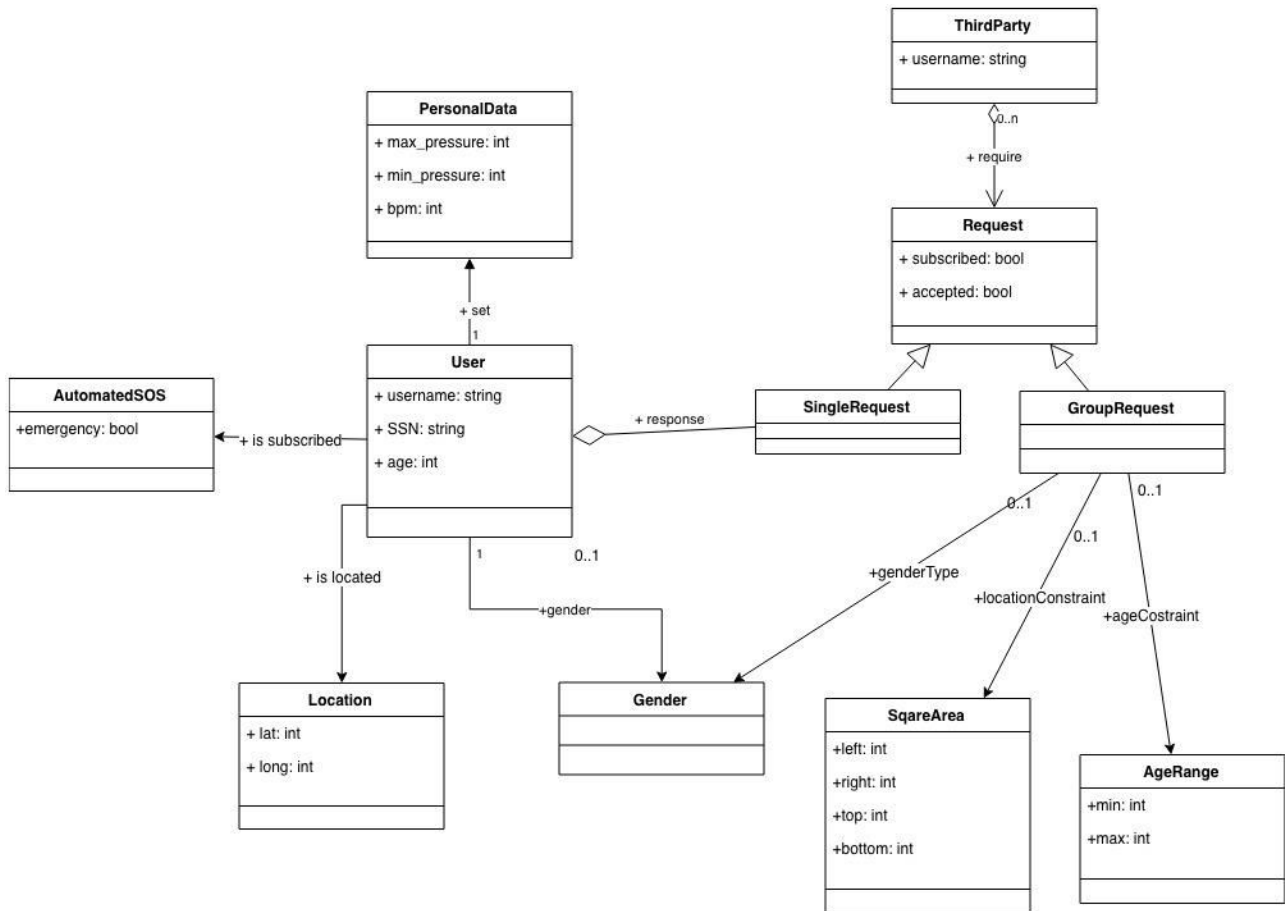
After registration and the authentication on the web app, Third-parties can request data w.r.t. a specific User or about an anonymous group of Users.

They will also be able to see approved and pending requests

Third-Parties can access to the latest Users' data provided by Data4Help using a set of APIs.

Through the website Third-Parties can finally subscribe to Users' data (both individual data and group data) in order to receive them as soon as they are produced.

Regarding the TrackMe system itself, to better describe the domain model used, the following diagram is provided:



## 2.2 Product Functions

In the following section the main functions of the TrackMe system are listed and more precisely specified, w.r.t. the already mentioned goals of the system:

- **Customers registration and authentication**

A possible Customer interested to the services offered by TrackMe must register into the system through the mobile or the web application.

Once inserted a unique username, a unique email and a password the customers can complete their registration by filling out the fields about their personal data (e.g. for Users first name, last name, gender, etc. and for Third-Parties the company name).

To log into the system, both the Users and the Third-Parties will use their own username and the related password

- **Data requests management**

Third Parties can request access to Users' data through the web application.

Third-party can choose which kind of request they want to submit:

- For the single user data requests, Third Parties will need to know the Users' SSN in order to identify the User they want to receive the data from.

Once the request has been submitted TrackMe will send a request notification to the interested User who can accept or refuse to share his data with the Third Party.

Once a request is accepted, the Third-Party can require the corresponding data, until the User withdraws the request.

- Third Parties can also require anonymous data w.r.t. a specific group of anonymous Users.

To specify the group of interest, Third-Parties can specify some fields (age, geographical area, gender).

This type of requests is handled directly by TrackMe that approves them if it is able to properly anonymize the requested data, i.e. TrackMe will accept any request for which the number of individuals whose data satisfy the request is higher than 1000

Third-Parties will have available a subscription option for both types of accepted requests, so they can receive updated data when it's available.

- **Emergency calls**

AutomatedSOS is a service in charge of monitoring constantly the health status of its Users. If data provided by Data4Help cross some specific threshold health parameters,

AutomatedSOS observes an anomaly and forward within 5 seconds a request for help to an external Ambulance Service which will take care of the User

## 2.3 User Characteristics

- **Customer:** either a person or a company who is not logged into the system.  
A customer will only see the registration/login page both on the mobile and web application.
- **User:** a person that is already registered into the system.  
Users' data is collected by TrackMe and each User can always monitor them.  
A User can access to the TrackMe services (Data4Help and AutomatedSOS), but he cannot require accessing data of other Users.
- **Third-Party:** an entity who is registered and logged into the system.  
Third-Parties can access to a specific User's data or to anonymous group data collected by TrackMe. They can also subscribe to some data in order to receive them as soon as they are produced.
- **Third-Party Administrator:** an individual of a Third-Party (e.g. an administrator, a technician, etc.) who deals with carrying out all the main features offered to Third-Party by TrackMe services
- **Ambulance Service:** an external entity which receives emergency notifications from AutomatedSOS and takes in charge of the operations needed to assist the User, according to his health status and anomalies detected.

## **2.4 Assumptions, Dependencies and Constraints**

### **2.4.1 Domain Assumptions**

- [D1]: Accurate individuals' locations are known by GPS
- [D2]: Data gathered from users' devices are correctly measured
- [D3]: Data provided by the users (e.g. their weight, their height, etc.) are assumed to be correct
- [D4]: The users have their own device (a smartwatch or a similar device) that interacts with the health app installed on their mobile.
- [D5]: The ambulance service entirely cover the area where there are AutomatedSOS users
- [D6]: Once the emergency notification is received by the Ambulance Service, it is its responsibility to assist the user that needs help.

### **2.4.2 Dependencies**

TrackMe depends on the native Health app installed on iOS and Android smartphones.

### 3. Specific Requirements

#### 3.1 External Interface Requirements

##### 3.1.1 User Interfaces

This section is meant to provide an idea of how system should look like.

It's not strictly binding, and all the constraints mentioned till now in this document must be met during the development of the system.

A mobile app is provided to Users, while a website is provided to Third Parties

- **Mobile app**

As a first step a login interface will be showed, and the guest will be able to sign in or to sign up.

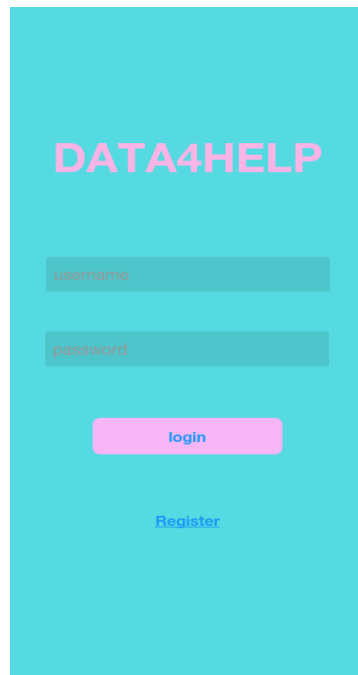
The image shows a mobile app screen for signing in. The background is a solid teal color. At the top, the text "DATA4HELP" is displayed in a bold, pink, sans-serif font. Below the title, there are two input fields: the first is labeled "username" in a small, light blue font, and the second is labeled "password" in the same font. Both labels are positioned to the left of the input fields. Below these fields, there is a pink rectangular button with the word "login" in a small, white, sans-serif font. At the bottom of the screen, the word "Register" is written in a small, light blue, sans-serif font.

FIGURE 1.A: SIGN IN

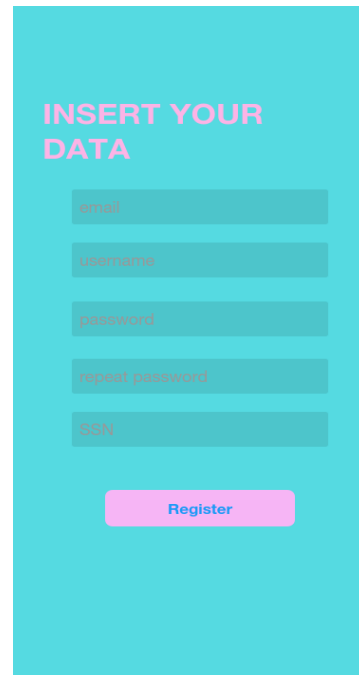
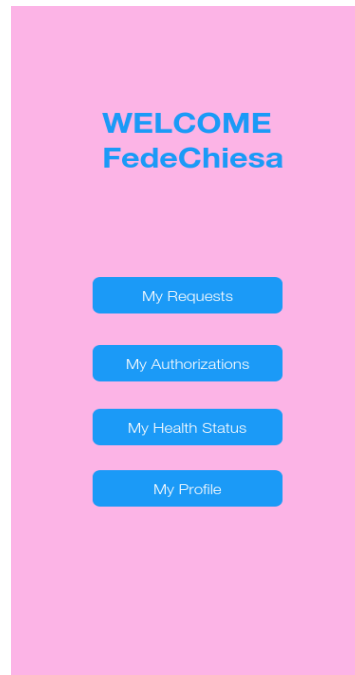
The image shows a mobile app screen for signing up. The background is a solid teal color. At the top, the text "INSERT YOUR DATA" is displayed in a bold, pink, sans-serif font. Below the title, there are five input fields: the first is labeled "email", the second "username", the third "password", the fourth "repeat password", and the fifth "SSN". All labels are in a small, light blue font and are positioned to the left of their respective input fields. Below these fields, there is a pink rectangular button with the word "Register" in a small, white, sans-serif font.

FIGURE 1.B: SIGN UP

Once the User has signed in, the app will directly open the homepage with the buttons to manage the requests, monitor his data or update his profile.



**FIGURE 2: SIGN UP**

If Users want to see their own pending requests, they must click on “My Requests” button. Here they can decide to accept or refuse every pending request.

In the “My Authorizations” section there are all the previously accepted request. Here the users can also decide to withdraw a specific authorization.





FIGURE 3.A: REQUESTS

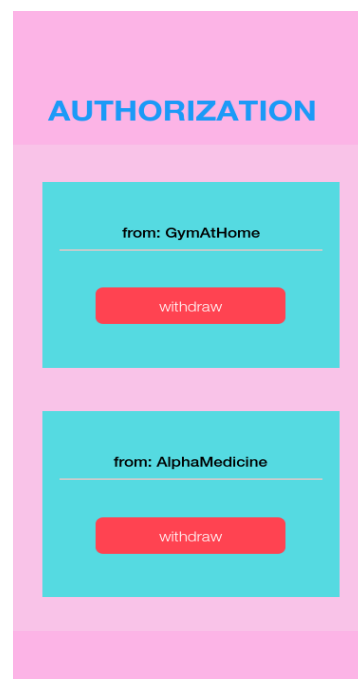
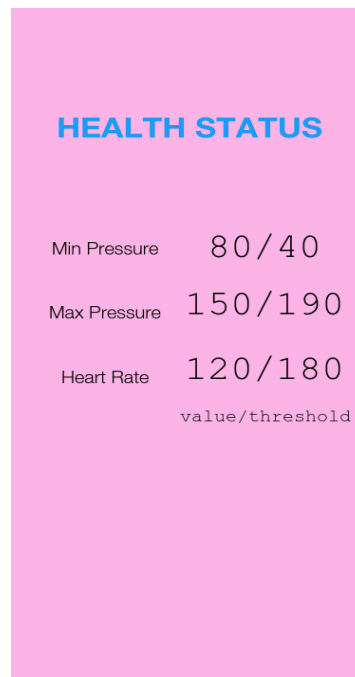


FIGURE 3.B: AUTHORIZATIONS

Users can keep track of their health status comparing them with the threshold values, by accessing on “My Health Status” section.



**FIGURE 4: HEALTH STATUS**

Users that want to subscribe to the AutomatedSOS service have to go in “My profile” section, where there is also a set of personal information that users can update.

If a user decide to register on AutomatedSOS, he must accept terms of condition.

**YOUR DATA**

First Name Federico  
 Last Name Chiesa  
 Height 170 cm  
 Weight 90 kg  
 Age 34 years  
 Gender Male  
 Phone Number 329874927  
 SSN GJHSGHGJ323

Update

AutomatedSOS ☐

REGISTER ON AUTOMATED SOS

**FIGURE 5.A: MY PROFILE SECTION BEFORE REGISTERING ON AUTOMATEDSOS**

**TERMS OF CONDITIONS**

-----  
 -----  
 -----  
 -----  
 -----  
 -----  
 -----  
 -----

ACCEPT

REFUSE

**FIGURE 5.B: TERMS OF CONDITIONS FOR AUTOMATEDSOS**

**YOUR DATA**

First Name Federico  
 Last Name Chiesa  
 Height 170 cm  
 Weight 90 kg  
 Age 34 years  
 Gender Male  
 Phone Number 329874927  
 SSN GJHSGHGJ323

Update

AutomatedSOS ☒

**FIGURE 5.A: MY PROFILE SECTION AFTER REGISTERING ON AUTOMATEDSOS**

## ▪ Web site

As a first step a login interface will be showed.

Third-Parties that aren't registered yet will be able to sign in or to sign up.to the service using the website of Data4Help. Indeed, on the Homepage there will be a login form and a link to the registration form where the Third party can sign up.

**DATA4HELP**

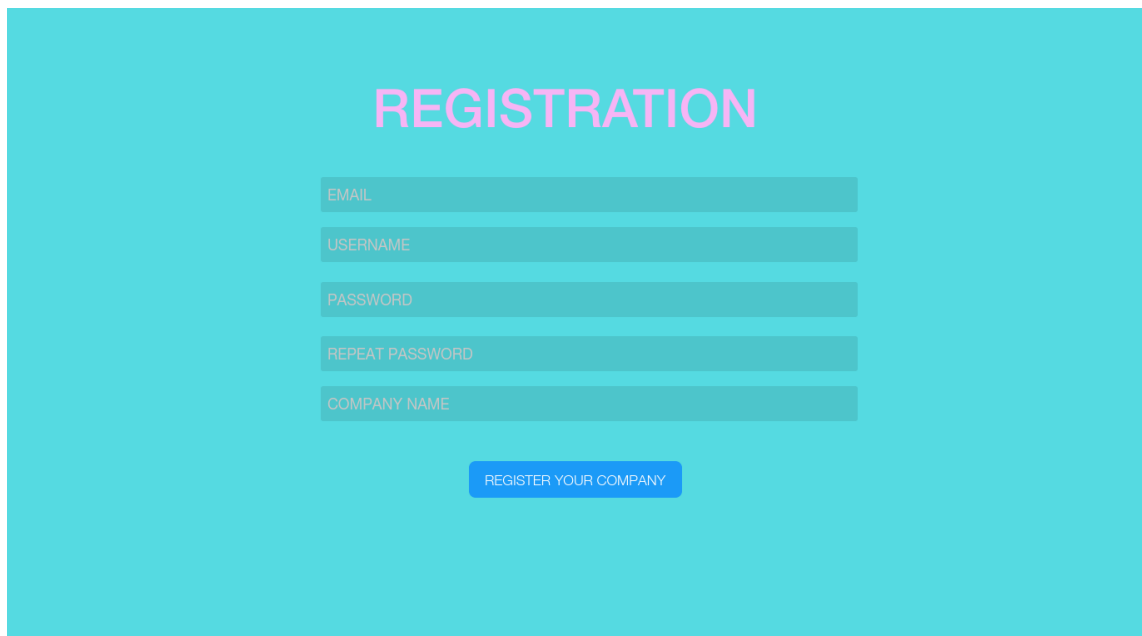
USERNAME

PASSWORD

LOGIN

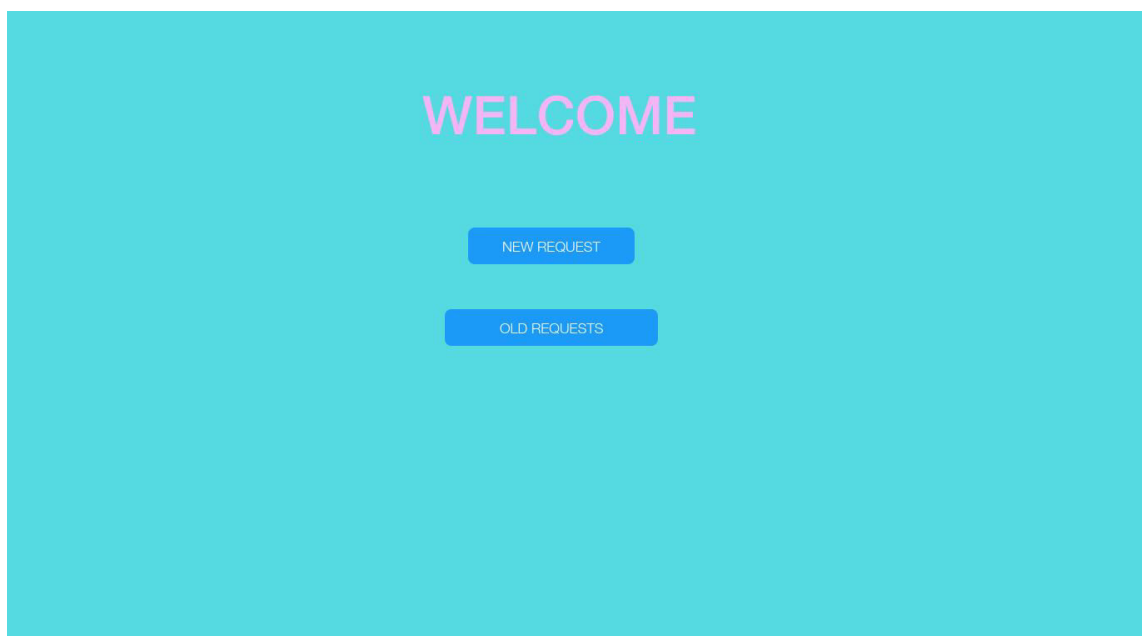
REGISTER YOUR COMPANY

**FIGURE 6.A: WEB SITE HOMEPAGE**

A registration form titled "REGISTRATION" in large, bold, pink capital letters. Below the title are five input fields, each with a light blue border and a light blue label: "EMAIL", "USERNAME", "PASSWORD", "REPEAT PASSWORD", and "COMPANY NAME". Below these fields is a blue button with white text that says "REGISTER YOUR COMPANY". The entire form is centered on a solid light blue background.

**FIGURE 6.B: REGISTRATION FORM FOR THIRD PARTIES**

Once a third party logged in, it will be able to manage the requests and to create new one clicking on the relative button.

A "WELCOME" page in large, bold, pink capital letters. Below the title are two blue buttons with white text: "NEW REQUEST" and "OLD REQUESTS". The page is centered on a solid light blue background.

**FIGURE 7: THIRD PARTY PERSONAL PAGE**

If a Third-Party clicks on “New Request” button it will open a new page where the Tp-a can fill out the form for a new request (he can choose between an individual request or a group request)

## NEW REQUEST

### INDIVIDUAL REQUEST

SSN

DESCRIPTION

SEND

### GROUP REQUEST

GENDER

☒ MALE  
☐ FEMALE

AGE RANGE:

GEOGRAPHIC AREA:

MAP

DESCRIPTION

SEND

FIGURE 8: NEW REQUEST SECTION

Otherwise, if a Tp-a clicks on “Old Requests” button it will open a new page where the Tp-a can manage all the requests

## OLD REQUESTS

### INDIVIDUAL REQUESTS

TO: FedeChiesa

STATUS ● pendant

### GROUP REQUESTS

AGE RANGE: 15 - 40

STATUS ● accepted

SUBSCRIBED ☐

DOWNLOAD DATA

FIGURE 9: OLD REQUESTS SECTION



### **3.1.2 Hardware Interfaces**

Thanks to a wide use of external APIs, the product does not require low-level interaction with the hardware.

The only hardware required to run the software is

- For Users: a mobile phone that can use GPS services and a native Health app
- For Third-Parties a device with a browser and Internet connection

### **3.1.3 Software Interfaces**

When a smartwatch or a similar device is connected to the smartphone, biomedical parameters from a User will be collected in real time by the native health app of the smartphone.

Data4Help will use a set of APIs to collect these collected data from its Users.

The software provides also a set of APIs to Third Parties to facilitate to embed the data provided in their systems. Through these APIs, Third Parties can require access to users' data in the way that has been described in the previously sections.

## **3.2 Scenarios**

### **3.2.1 Scenario 1**

Enrico is an 18-years-old boy, he really likes sports and he is in the school athletic team. Due to the hard training sessions, his teammates tell him to download the Data4Help app to keep track of his health status.

He decides to try this app, so he downloads it and he creates a new account.

After that, he logs in on Data4Help app, inserting his credentials (email and password) and he sets his body measurements, i.e. his weight and his height.

Enrico can now monitor his health status comparing his collected data with the threshold values in the dedicated page of the Data4Help app.

### **3.2.2 Scenario 2**

The private clinic “HealthMyPriority” decides to create a service to support his cardiopathic patients. The clinic wants to notify them when they need to take their meds. HealthMyPriority decides to use the data collected by Data4Help to ensure a good service to its customers.

The HealthMyPriority administrator goes to the Data4Help website and fill out the registration form. Now he creates the data requests for the customers by inserting their SSN. Therefore, Data4Help send them a notification to ask them to answer to the request. When the clinic sees that their customers have accepted the requests, it subscribes to new data in order to receive new data as soon as they are produced and send the notification about the meds to take on their app.

### **3.2.3 Scenario 3**

The Milan city council decides to join Data4Help to keep track of the health status of the citizen and decide to divide them into 3 age groups:

- Under 18
- Between 19 and 59
- Over 60

After a technician has register the city, he sends the 3 requests based on the age range.

The Data4Help system accepts 2 of them and sends the related data. The third one (over 60) didn't match the constraint of the amount of people needed (the number of people over 60 registered to Data4Help is lower than 1000). Once the council receives the notifications, he decides to change the age limits and send 2 new requests:

- Between 19 and 40



- Over 41

Now the system approved the requests and send the related data to the city council.

### **3.2.4 Scenario 4**

Mr. Rossi is a 70 years old man and already has got 2 heart attack and other health problems.

Rosa, his daughter, decides to buy him a smartwatch and to register him to the Data4Help service to monitor his health status.

Once completed the registration, Rosa reads about the AutomatedSOS service and decides to subscribe her father on.

Two nights later the service monitored a health status under the threshold limits.

Automatically AutomatedSOS sent the location and the health status data of Mr. Rossi to the Ambulance Service which arrived at his house only after 5 minutes and saved his life.

### **3.2.5 Scenario 5**

Giacomo has been a Data4Help user for a long time to share his data with his gym service “GymAtHome” that helped him to do gym sessions at home.

However, Giacomo started the courses to a bachelor’s degree in Exercise Science this year. He received a notification on his Data4Help app for a data request from his new university to monitor the health status of his students. He accepted the request and Data4Help can now send his data to the university. At the same time, he decides to withdraw the authorisation with the “GymAtHome” service because he prefers applying what he is learning at the university.

For this reason, he opens the Data4Help app, goes on the authorisations page and revoke the authorisation to “GymAtHome” service.

### 3.3 Functional Requirement

- **[G1]: Allow individuals to become registered users of Data4Help**
  - **[R1]:** The user can create an account for the usage of Data4Help, by selecting a username and a password.
  - **[R2]:** The users have to agree that TrackMe acquires their data
  - **[R3]:** The usernames used in the system are unique to every user
- **[G2]: Allow users to sign up for AutomatedSOS service**
  - **[R4]:** The user can use AutomatedSOS service, accessing with the same account credential of Data4Help
  - **[R5]:** The system must check that every AutomatedSOS user is an user of Data4Help service
- **[G3]: Provide the registration to third parties who want access to users' data**
  - **[R6]:** Third parties can create an account for monitoring data, by selecting an username and a password
- **[G4.1]: Give third parties access to data of a specific user**
  - **[R7]:** The system must check if the user has authorized the third party to get access to his data
  - **[R8]:** The system must notify third party if the user has accepted the request or not
  - **[D1]:** Accurate individuals' locations are known by GPS
  - **[D2]:** Data gathered from users' devices are correctly measured
  - **[D3]:** Data provided by the users (e.g. their weight, their height, etc.) are assumed to be correct
- **[G4.2]: Give access to anonymized data of group**
  - **[R9]:** The system must check that the number of individuals whose data satisfy the request is higher than 1000
  - **[R10]:** The system must notify third party if the request has not been approved
  - **[D1]:** Accurate individuals' locations are known by GPS
  - **[D2]:** Data gathered from users' devices are correctly measured
  - **[D3]:** Data provided by the users (e.g. their weight, their height, etc.) are assumed to be correct
- **[G5]: Allow users to accept or refuse the requests from third-parties to access their own data and their location**

- [R11]: The system must send the request to a specific user from a third party that want access to his individual data
- **[G6]: Allow third parties to subscribe to new data and to receive them as soon they are produced**
  - [R12]: Every time new data are produced, the system must check if exist some third parties subscribed to them
  - [R13]: TrackMe system sends a request to third parties that have access rights to the required data to subscribe to new produced data
- **[G7]: Allow customers to insert personal data and information about their body measurements (e.g. weight, height, etc.)**
  - [R14]: The system must provide a form to the users for inserting or updating data and personal information
  - [D3]: Data provided by the users (e.g. their weight, their height, etc.) are assumed to be correct
- **[G8]: If some specific parameters of health status are below certain threshold, send an ambulance to the user location, with a reaction time less than 5 seconds**
  - [R15]: When devices monitor data below certain threshold, AutomatedSOS system must send the user's health status information to the ambulance service
  - [R16]: When devices monitor data below certain threshold, AutomatedSOS system must send the localization acquired through GPS to the ambulance service
  - [D1]: Accurate individuals' locations are known by GPS
  - [D2]: Data gathered from users' devices are correctly measured
  - [D4]: The users have their own device (a smartwatch or a similar device) that interacts with the health app installed on their mobile.
  - [D5]: The ambulance service entirely covers the area where there are AutomatedSOS users
  - [D6]: Once the emergency notification is received by the Ambulance Service, it is its responsibility to assist the user that needs help.
- **[G9]: Allow users to keep track of their health status at any time**
  - [R17]: The system must provide a view where the users can see their own health status and the specific threshold values

- [D4]: The users have their own device (a smartwatch or a similar device) that interacts with the health app installed on their mobile.
  - [D2]: Data gathered from users' devices are correctly measured
- [G10]: **Allow users to withdraw the authorisation to third parties to access their data**
  - [R18]: If a user decides to withdraw the authorisation to a third party to access his own data, the system must notify this third party
  - [R19]: The system must unsubscribe the access to new data by the third parties whose access rights are revoked

### 3.3.1 Use Case Diagram

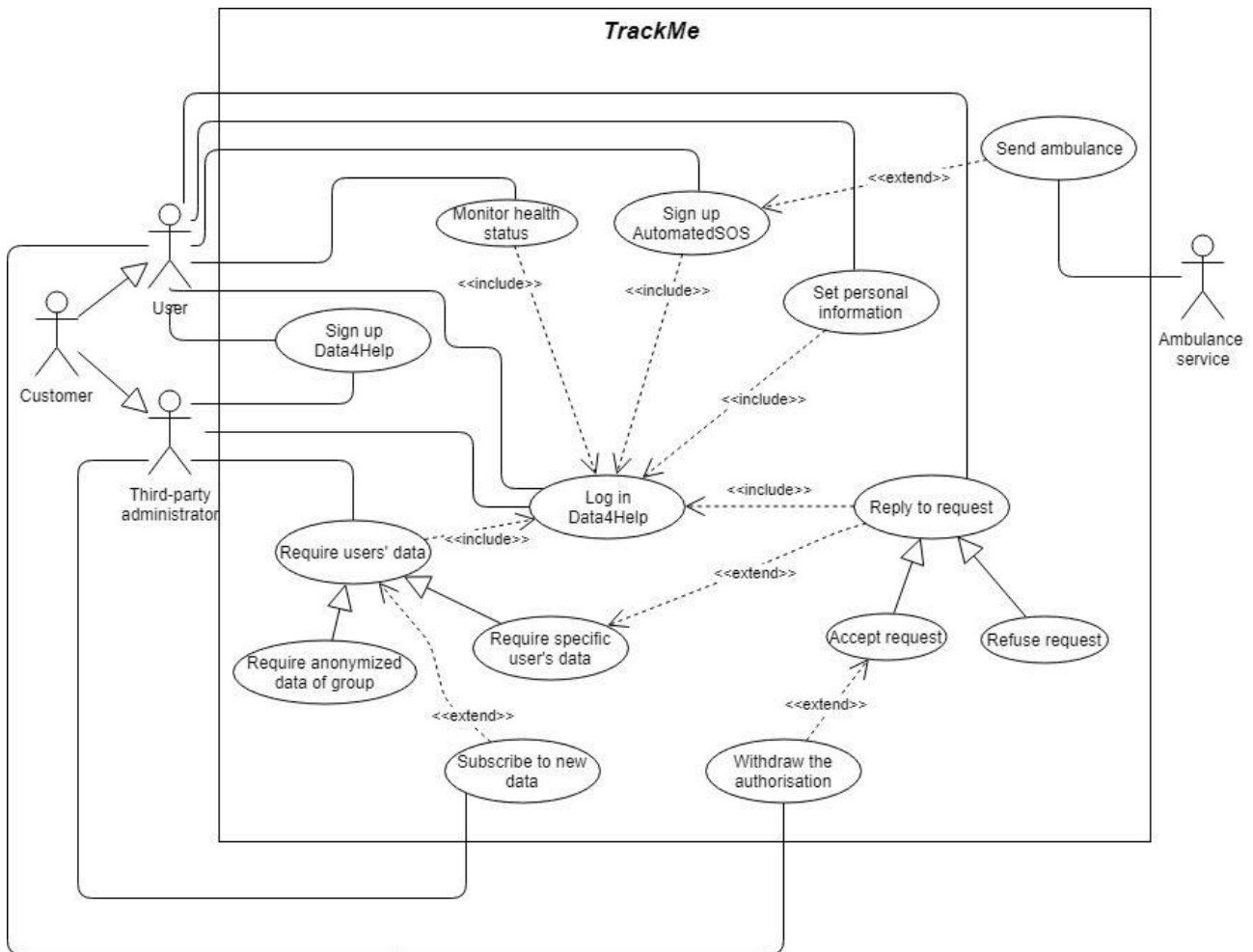


FIGURE 10: USE CASE DIAGRAM

### 3.3.2 Use Cases

Name	Users_Signup_Data4Help
Actor	User
Entry Condition	User has already installed Data4Help app on his mobile
Events Flow	<ul style="list-style-type: none"> <li>- The user opens the Data4Help app</li> <li>- The user clicks on the “Register” button</li> <li>- The user fills out the form with his data</li> <li>- The system saves his data into Database</li> </ul>
Exit Condition	The user has successfully registered

Exceptions	<ul style="list-style-type: none"> <li>- the email is already used</li> <li>- the password is too short</li> <li>- the private user didn't fill all the mandatory fields with valid data</li> </ul> <p>The system reloads the registration page and notifies the user about the wrong fields.</p>
------------	---

*Use Case ID: UC.1*

Name	User_LogIn
Actor	User
Entry Condition	The User has previously successfully signed up
Events Flow	<ul style="list-style-type: none"> <li>- The User open the Data4help app</li> <li>- The User fill out the form with his credentials</li> <li>- The User clicks on the Confirm button</li> <li>- The system opens the user homepage</li> </ul>
Exit Condition	The User is successfully redirected to his homepage
Exceptions	<ul style="list-style-type: none"> <li>- The inserted credentials aren't correct</li> </ul> <p>The system asks the user if he want to try again or open the registration form to sign up.</p>

*Use Case ID: UC.2*

Name	Third-parties_SignUp
Actor	Third-party administrator
Entry Condition	True
Events Flow	<ul style="list-style-type: none"> <li>- The Tp-a open the Data4Help website</li> <li>- The Tp-a opens the registration page</li> <li>- The Tp-a fill out the form with the company data</li> <li>- The system saves his data in the Database</li> </ul>
Exit Condition	The Third-party is successfully registered in the system

Exceptions	<ul style="list-style-type: none"> <li>- The email is already used</li> <li>- The username is already used</li> <li>- The password is too short</li> <li>- The Tp-a didn't fill all the mandatory fields with valid data</li> </ul> <p>The system reloads the registration page and notifies the third-party administrator about the wrong fields.</p>
------------	--

*Use Case ID: UC.3*

Name	Third-Party_LogIn
Actor	Third-Party administrator
Entry Condition	The Third-Party has previously successfully signed up
Events Flow	<ul style="list-style-type: none"> <li>- The Third-Party open the Data4help website homepage</li> <li>- The Third-Party clicks on the Login button</li> <li>- The Third-Party fill out the form with his credentials</li> <li>- The Third-Party clicks on the Confirm button</li> <li>- The system opens the homepage dedicated to Third-Parties</li> </ul>
Exit Condition	The Third-Party is successfully redirected to his personal page
Exceptions	<ul style="list-style-type: none"> <li>- The inserted credentials aren't correct</li> </ul> <p>The system asks the Third-Party if he wants to try again or open the registration form.</p>

*Use Case ID: UC.4*

Name	Users_Signup_AutomatedSOS
Actor	User
Entry Condition	The User is already logged in
Events Flow	<ul style="list-style-type: none"> <li>- The User opens his personal page</li> <li>- The User clicks on the “Register on AutomatedSOS” button</li> <li>- The User accepts the terms of condition</li> <li>- The User clicks on “Confirm registration” button</li> </ul>

Exit Condition	The User has successfully registered to the AutomateSOS service
Exceptions	- The User doesn't accept the terms of condition The system reloads the personal page

*Use Case ID: UC.5*

Name	Require users' data
Actor	- Third-party administrator
Entry Condition	The Tp-a is already logged in
Events Flow	- The tp-a opens the Data4Help website - The tp-a logs into the company account - The tp-a clicks on the "Request Data" Button - The tp-a chooses between "Single user data" or "Group User data" - The tp-a fills out the form and clicks on "Send Request"
Exit Condition	The system notifies the third party with an email if the request has been approved or not.
Exceptions	- The form is filled out with invalid data The system will reload the form page and notify tp-a what fields are incorrect.

*Use Case ID: UC.6*

Name	ReplyRequests
Actor	User Third-party administrator
Entry Condition	- Tp-a has submit a request to the user - The user is already logged in
Events Flow	- The user opens the Data4Help app - The user clicks on the "requests" label - The user can select the requests for his data and click on



	“accept” or “refuse”
Exit Condition	The system will notify the third-party with the user decision
Exceptions	-

*Use Case ID: UC.7*

Name	SubscribeNewData
Actor	Third-party administrator
Entry Condition	- The third-party administrator is logged in
Events Flow	<ul style="list-style-type: none"> <li>- The tp-a clicks on the “requests” label on his personal page</li> <li>- The tp-a selects an accepted request</li> <li>- The tp-a clicks on the “subscribe to new data” button</li> </ul>
Exit Condition	The system registers the tp-a choice and will send data as soon as they are produced
Exceptions	<ul style="list-style-type: none"> <li>- The request is a pendant request, i.e. it hasn’t been accepted yet</li> </ul> <p>The system notifies the tp-a to wait for the acceptance of the request</p>

*Use Case ID: UC.8*

Name	SetPersonalData
Actor	User
Entry Condition	The user is already logged in
Events Flow	<ul style="list-style-type: none"> <li>- The user opens the app and navigates to his personal page</li> <li>- The user clicks on “update data”</li> <li>- The user fills out the form with his data</li> </ul>
Exit Condition	<ul style="list-style-type: none"> <li>- The user clicks on the “Save data” button</li> <li>- The user data are updated.</li> </ul>
Exceptions	<ul style="list-style-type: none"> <li>- The user has inserted invalid data</li> </ul> <p>The system will reload the page and notify the user on what fields are incorrect</p>

*Use Case ID: UC.9*

Name	SendAmbulance
Actor	User Ambulance service
Entry Condition	- The user is registered to AutomatedSOS service - Some specific parameters of user health status are below certain threshold
Events Flow	- The system sends the user personal data to the ambulance service - The system sends the user health status to the ambulance service - The system sends the user location to the ambulance service - The ambulance service notifies the system he has accepted the request
Exit Condition	The ambulance service sends an ambulance to the user location
Exceptions	-

*Use Case ID: UC.10*

Name	MonitorHealthStatus
Actor	User
Entry Condition	The user is already logged in
Events Flow	- The user opens the Data4Help app - The user clicks on the “My health status” button - The user can monitor his health status, comparing it with the threshold values
Exit Condition	The user closes the page or goes on the previous page
Exceptions	-

*Use Case ID: UC.11*

Name	WithdrawAuthorisation
------	-----------------------

Actor	User Third-party administrator
Entry Condition	<ul style="list-style-type: none"> <li>- The user has already given an authorisation to a third party to access his own data</li> <li>- The user is already logged in</li> </ul>
Events Flow	<ul style="list-style-type: none"> <li>- The user opens the Data4Help app</li> <li>- The user clicks on the “authorisation” label</li> <li>- The user can select which third party authorisation he wants to withdraw</li> <li>- The user clicks on the “confirm the withdrawal” button</li> <li>- The system notifies the Tp-a about the withdrawal of the authorisation</li> </ul>
Exit Condition	<ul style="list-style-type: none"> <li>- The user data are now unavailable to the third-party</li> </ul>
Exceptions	-

*Use Case ID: UC.12*

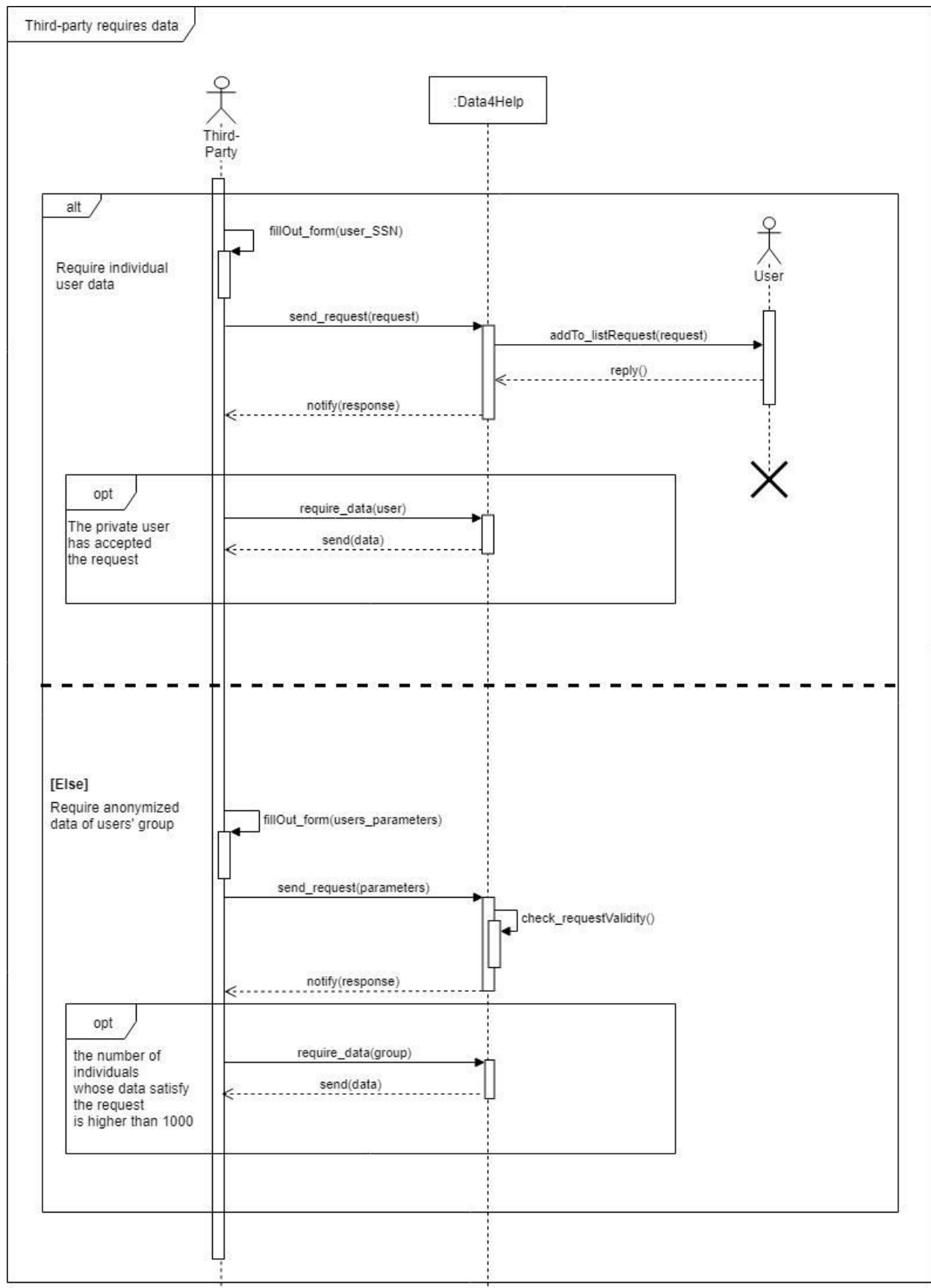
### **3.3.3 Sequence Diagrams**

To better describe the interaction between the actors of the TrackMe system, two Sequence Diagrams will be presented.

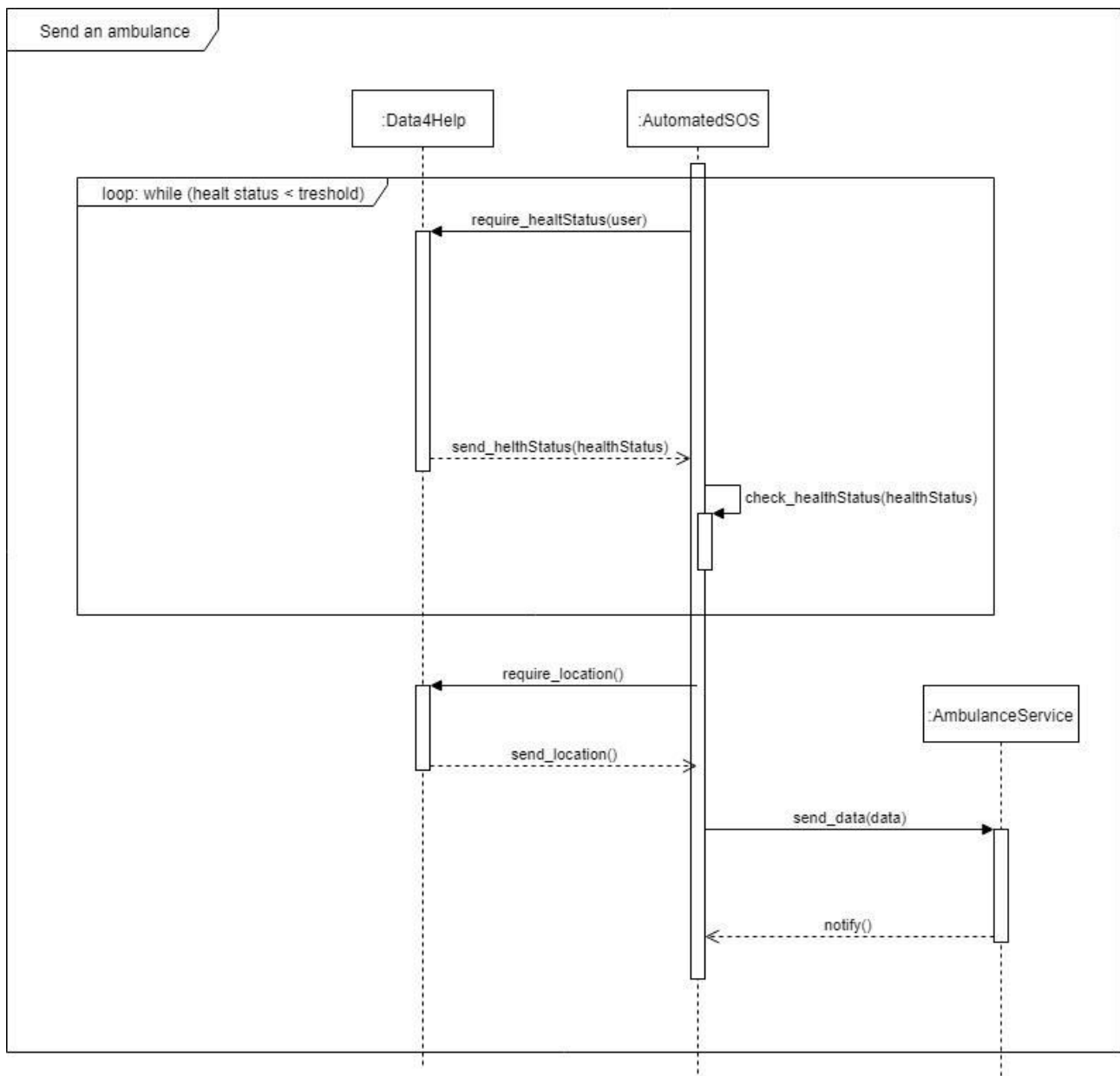
The first one describes how a Third-Party makes a request to give access to Users' data.

The second one describes the management of the AutomatedSOS service.

- **Third-Party requires data Sequence Diagram**



- Send an ambulance Sequence Diagram



### 3.3.4 Mapping on Requirements

Raw ID	Goal ID	Req ID	Use Case ID
1	G1	R1	UC.1
2	G1	R2	UC.1
3	G2	R4	UC.2
4	G3	R6	UC.3
5	G4.1	R8	UC.6
6	G4.2	R10	UC.6
7	G5	R11	UC.7
8	G6	R13	UC.8
9	G7	R14	UC.9
10	G8	R15	UC.10
11	G8	R16	UC.10
12	G9	R17	UC.11
13	G10	R18	UC.12
14	G10	R19	UC.12

*Traceability Matrix*

### **3.4 Performance Requirements**

The system has to be able to handle a big amount of requests simultaneously.

It has to be flexible in order to be easily scalable based on the growth of the number of customers and requests

Furthermore it has to guarantee quick, reactive and correct response both on the mobile and the web application.

Particular attention should be paid to the real time interactions between Data4Help and AutomatedSOS services, i.e. Data4Help needs to acquire data to be monitored by AutomatedSOS.

It is very important to guarantee performance requirements w.r.t. AutomatedSOS, in order to guarantee a reaction time of less than 5 seconds from the time the parameters are below the threshold.



## **3.5 Design Constraints**

### **3.5.1 Regulatory policies**

The system will ask for users' personal information and has to store them safely.

Moreover, the system will have to ask for users' acceptance of Terms and conditions in order to activate the AutomatedSOS service and send his data to the Ambulance Service.

Telephone numbers and email addresses won't be used for commercial uses.

### **3.5.2 Hardware Limitations**

This is a software application with no strict hardware limitations. The only requirements for guaranteeing the proper operation of the whole system are:

- For mobile app an iOS or Android smartphone that supports Internet connection, GPS services and a native health app
- For web app a device with an Internet connection

## **3.6 Software System Attributes**

### **3.6.1 Reliability**

The services provided by our system must be available 24/7.

However, system unavailability for very some small period of time might be tolerated.

### **3.6.2 Availability**

The system is expected to be available 99.9% of the time of the year to offer a good quality of service.

A distributed system with a load balancer has to be taken into consideration to achieve this result.

### **3.6.3 Security**

Security is a fundamental aspect of our services.

Communications between the servers and Data4Help Users will be encrypted in order to guarantee integrity, authentication and confidentiality.

All the data stored on the Data4Help servers will be encrypted as well, including the Users' credentials, personal information and preferences.

Communications between the TrackMe servers and databases or between TrackMe services and APIs will use the latest encryption protocols.

### **3.6.4 Maintainability**

As previously mentioned, the application is going to be flexible and easy to maintain.

Maintenance operation will be easy due to the way the system will be developed.

It should follow design patterns whenever useful and must be well commented.

Of course, complete and detailed documentation will also be provided in order to keep the maintainability on the highest level.

New versions of the mobile applications will be distributed as an update via the native Store both on Android and iOS devices and new versions of the website will be visible to the third parties as soon as they visit the page again.

With respect to APIs, every update will be reported to the documentation found on the website.

### **3.6.5 Portability**

The Data4Help applications will be highly portable since they don't rely on specific hardware requirements.

The mobile application will be distributed for every device that supports an updated iOS or Android OS with a native health app installed on

Portability is also guaranteed for web application by simply using a device with an Internet connection.

## 4. Formal Analysis Using Alloy

### 4.1 Alloy constraints

In this section, the Alloy model is given. Using it, some of the features of the system are specified and explained in more details, with the focus being on the following constraints:

- We assumed that every person can be a user of our system, therefore we have only the constraint that age cannot be negative
- In the Location signature, for simplicity the latitude (lat) can range from -3 to 3 meaning the Earth latitude values from -90 to 90 and the longitude can range from -6 to 6 meaning the Earth longitude values from -180 to 180.
- A Third Party can subscribe to a request only if it is accepted.  
The notification process concerning of the subscribed request is outside the scope of the Alloy model.
- The presence of the automatedSOS in the User signature indicates that the user is subscribed to the AutomatedSOS service.
- The emergency field in the AutomatedSOS signature indicates the bad health status of the user and that the Ambulance service has been notified
- The squareArea is a signature that represents the geographic limits of a group request. Top and Bottom corresponds to the latitude limitations and Left and Right corresponds to the longitude limitations.
- The GroupReq is the signature corresponding to a group request and need to have at least one constraint to select the users to be picked for the set of personal data and the set of Personal data need to have more than 1000 elements (5 only for the alloy model).
- The health threshold values are to be calculated and the emergency thresholds used are only for the alloy model.

## 4.2 Alloy model

```
open util/boolean
open util/integer

-- The username used by a User or a Third-Party --
sig Username{}

-- The SSN of a User --
sig SSN{}

-- For male User, male = TRUE and female = FALSE. Otherwise, for female User, female = TRUE and male = FALSE--
sig Gender{
  male: one Bool,
  female: one Bool
}
{(male.isTrue and female.isfalse) or (female.isTrue and male.isfalse)}

-- The acceptedRequests is a set of Single Requests that a specific user has accepted, while pendantRequests refers to all the pending requests of a specific user --
-- A User can accept a request to give the authorization to a Third-Party to get access to his latest personal data --
-- automatedSos indicates if a User is subscribed to the automatedSOS service --
sig User{
  username: one Username,
  ssn: one SSN,
  acceptedRequests: set SingleReq,
  pendantRequests: set SingleReq,
  location: one Location,
  data: one PersonalData,
  age: one Int,
  gender: one Gender,
  automatedSos: lone AutomatedSOS
}
-- age cannot be negative --
{age >= 0 and (all p: pendantRequests | p.accepted.isfalse) and (all a: acceptedRequests | a.accepted.isTrue)}

sig Location{
  -- latitude --
  lat: one Int,
  -- longitude --
  long: one Int
}
-- { lat >= -90 and lat <= 90 and long >= -180 and long <= 180 }
{lat >= -3 and lat <= 3 and long >= -6 and long <= 6}

-- requests can be accepted or not --
sig ThirdP{
  username: one Username,
  requests: set Request
}

abstract sig Request{
  accepted: one Bool,
  subscribed: one Bool
}
-- a request can have a subscription only if it is an accepted request --
{subscribed.isTrue implies accepted.isTrue}

sig SingleReq extends Request{
  user: one User
}
```

```

-- left, right, top and bottom are the sides of the area concerning a Group request --
sig SquareArea{
    left: one Int,
    right: one Int,
    top: one Int,
    bottom: one Int
}
-- { left<=right and top<=bottom and left>= -90 and right <= 90 and top>= -180 and bottom<= 180 } --
{left<=right and top<=bottom and left>= -6 and right <= 6 and top>= -3 and bottom<= 3}

-- Two limits concerning the Users' age involved in a group Request --
sig AgeRange{
    minAge: one Int,
    maxAge: one Int
}
{minAge <= maxAge and minAge>=0}

-- a group request consists of a set of Personal Data and some constraints used for make the request --
sig GroupReq extends Request{
    data: set PersonalData,
    square: lone SquareArea,
    age: lone AgeRange,
    gender: lone Gender
}
-- a group request exists (exists a set of Personal Data) only if it is accepted
-- A group request is accepted only if there is at least one constraint (the square Area, the age Range or the gender)
{
    (square != none or age!=none or gender!=none) and (accepted.isfalse implies data = none)
}

```

```

-- latest data measured and collected --
sig PersonalData{
    max_pressure: one Int,
    min_pressure: one Int,
    bpm: one Int
}
{min_pressure > 0 and min_pressure < max_pressure and bpm > 0}

-- if emergency = TRUE, the health status of an User of the AutomatedSOS service is critical (at least one of the personal data cross the related treshold value) --
sig AutomatedSOS{
    emergency: one Bool
}

----- FACT -----|

-- Considering two distinct User, they can never have the same username or the same SSN --
fact distinctUser{
    all disj u1,u2: User | u1.username != u2.username and u1.ssn != u2.ssn
}

-- Each request can belong to only one third Party, so there will never be a request concerning two different third-party --
fact distinctReq{
    all disj tp1,tp2: ThirdP | (tp1.requests & tp2.requests) = none
}

-- Each username belong to either a User or a Third Party. There will never be an username which no belong to a Customer --
fact UsernameAreConnected{
    all u: Username | u in ( User.username + ThirdP.username )
}

-- Each SSN belong to one and only one User --
fact ssnConnectionUser{
    all s: SSN | one u: User | u.ssn = s
}

-- Each Location belong to one and only one User --
fact locationConnectionUser{
    all l: Location | one u: User | u.location = l
}

-- Each squareArea belong to one and only one GroupRequest --
fact squareAreaConnectionGroupR{
    all s: SquareArea | one gr: GroupReq | s=gr.square
}

-- Each ageRange belong to one and only one GroupRequest --
fact ageConnectionGroupR{
    all a: AgeRange | one gr: GroupReq | a=gr.age
}

-- A gender constraint can Male or Female --
fact genderConnectionUser{
    (all disj g1,g2: Gender | g1.male!=g2.male) and #Gender = 2
}

-- Considering two distinct Third Parties, they can never have the same username --
fact distinctTp
{all disj tp1,tp2: ThirdP | tp1.username != tp2.username }

-- Considering a User and a Third Party, they can never have the same username --
fact distinctUsername{
    all u:User, tp:ThirdP | u.username != tp.username
}

```

```

-- Each personal data belong to a specific User --
fact PersonalDataUserConnection {
    all p: PersonalData | one u: User | p=u.data
}

-- -- Each request belong to a specific Third Party --
fact RequestThirdPConnection {
    all r: Request | one tp: ThirdP | r in tp.requests
}

-- TrackMe will accept any request for which the number of individuals whose data satisfy the request is higher than 1000
fact validNumGroupRequest{
    all g: GroupReq |
        (#g.data >5) implies g.accepted.isTrue else g.accepted.isFalse
}

-- a groupRequest is valid if each user's location of the request is within the square area specified in the group request OR
-- if the user's age of the request is within the two limits of the age Range OR
-- if The gender types of a Group request and the corresponding Personal data must match
fact validDataGroupReq{
    all g: GroupReq |
        (g.square = none or
            (all d: g.data | one u: User | u.data = d and
                (u.location.lat<=g.square.bottom and u.location.lat>=g.square.top and u.location.long<=g.square.right and u.location.lat>=g.square.left)
            )
        ) and
        (g.age = none or
            (all d: g.data | one u: User | u.data = d and
                (u.age>=g.age.minAge and u.age<=g.age.maxAge)
            )
        ) and
        (g.gender = none or
            (all d: g.data | one u: User | u.data = d and
                (u.gender.male = g.gender.male)))
}

```



---

```

-- If a single request is accepted by a user, it belong to a acceptedRequests list of that specific user --
fact validAcceptedRequest{
  (all s:SingleReq |
    s.accepted.isTrue implies one u: User | s in u.acceptedRequests
  )
}

-- If a single request is NOT accepted by a user, it belong to a pendantRequests list of that specific user --
fact validPendantRequest{
  (all s:SingleReq |
    s.accepted.isFalse implies one u: User | s in u.pendantRequests
  )
}

-- Each AutomatedSOS belong to only one User --
fact SosUserConnection {
  all a: AutomatedSOS | one u: User | a=u.automatedSos
}

-- all the Single request (accepted or not) of a specific user must belong to at least one of the two lists of requests of that specific user (acceptedRequests or pendantRequests) --
fact SingleRequestUserConnection {
  all u: User | all r: (u.acceptedRequests + u.pendantRequests) | r.user = u
}

-- if there is the AutomatedSOS service and at least one of the personal data cross the related threshold value, there will be an emergency notification,
-- otherwise there will not be any emergency notification--
fact emergencyCall{
  (all u: User |
    u.automatedSos = none or
    (u.data.min_pressure>5 or u.data.min_pressure<2 or u.data.max_pressure>6 or u.data.max_pressure<3 or u.data.bpm<2 or u.data.bpm>6) implies
      u.automatedSos.emergency.isTrue
    else u.automatedSos.emergency.isFalse)
  )
}

```

---

---

----- PREDICATES -----

```
-- Health Status of the User u is good (all the Personal Data values DO NOT cross threshold values) --
-- User u has activated AutomatedSos service --
pred healthyUser[u:User]{
    one p:PersonalData | u.data = p and u.automatedSos != none and (p.max_pressure = 3 and p.min_pressure = 2 and p.bpm = 3)
}

-- Health Status of the User u is NOT good (one of the Personal Data values cross the corresponding threshold value) --
-- User u has activated AutomatedSos service --
pred needEmergency[u:User]{
    one p:PersonalData | u.data = p and u.automatedSos != none and p.max_pressure = 2
}

-- User u1 is in healthy condition, so NO emergency call is necessary
-- Otherwise User u2 has bad health conditions, so an emergency call is necessary
pred callAmbulance[u1: User, u2: User]{
    //preconditions
    healthyUser[u1]
    needEmergency[u2]
    //postconditions
    u1.automatedSos.emergency.isFalse
    u2.automatedSos.emergency.isTrue
}

-----

-- The single request s is an accepted request present in the acceptedRequests list of an User u --
pred acceptedSingleReqInAcceptedList[s:SingleReq]{
    one u: User | s in u.acceptedRequests
}

-- The single request s is a pending request present in the pendantRequests list of an User u --
pred pendingSingleReqInPendingList[s:SingleReq]{
    one u: User | s in u.pendantRequests
}

-----

-- If a request has been accepted, it must be in an acceptedRequests list, otherwise it must be in a pendantRequests list--
pred singleReqWellPlaced[s1: SingleReq, s2: SingleReq]{
    //pre-conditions
    s1.accepted.isTrue
    s2.accepted.isFalse
    //post-conditions
    acceptedSingleReqInAcceptedList[s1]
    pendingSingleReqInPendingList[s2]
}

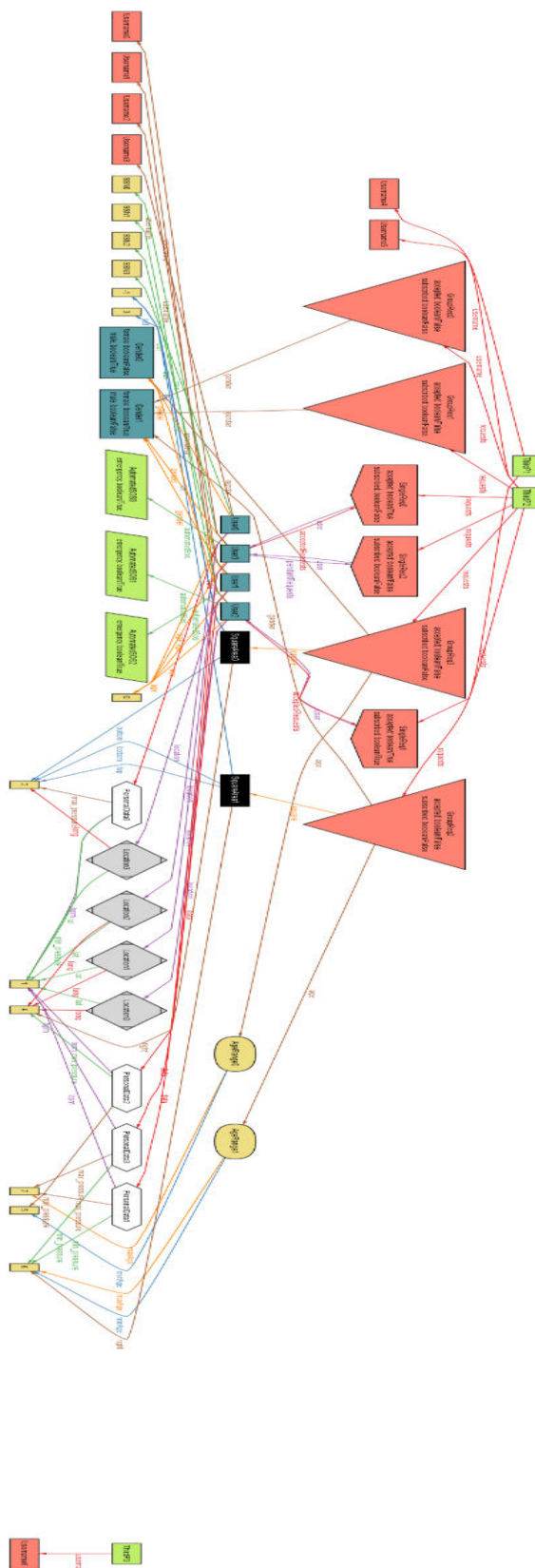
run callAmbulance
run singleReqWellPlaced

pred show{
    #ThirdP > 2
    #User > 2
    #SingleReq > 2
    #GroupReq > 2
}

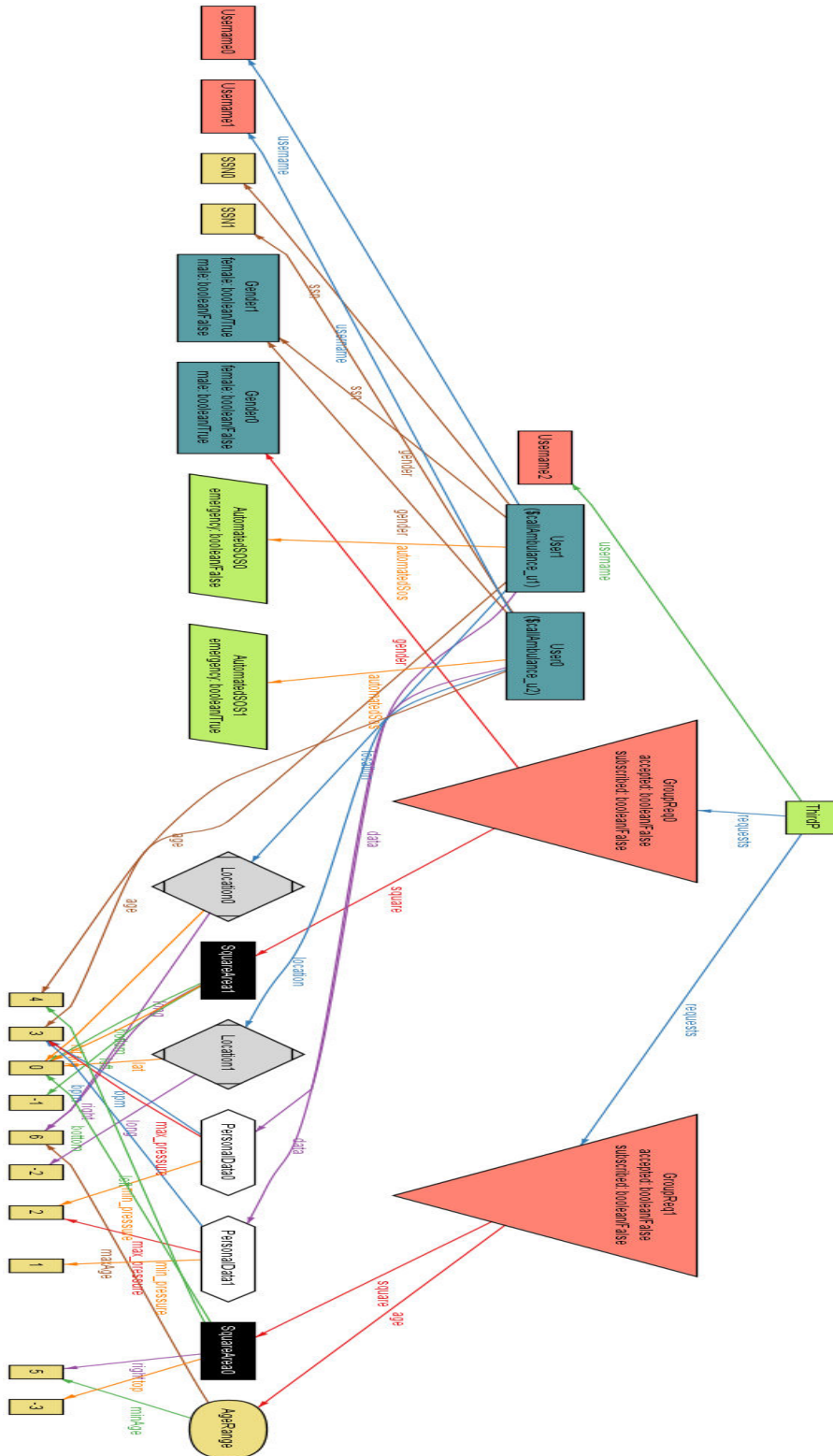
run show for 7
```

### 4.3 Alloy World Generated

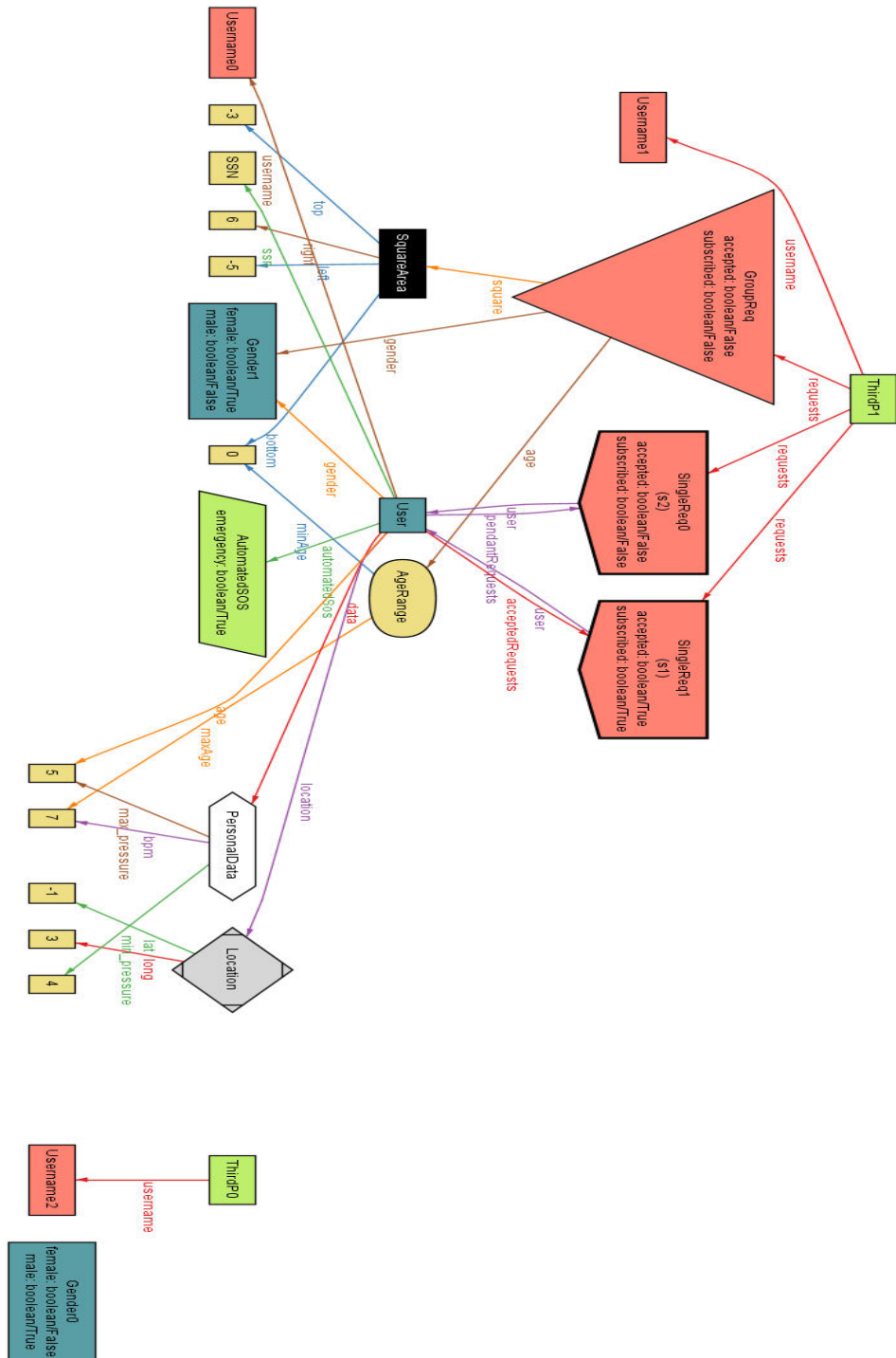
- **World Generated**



- World Generated by “callAmbulance” predicate



- World Generated by “singleReqWellPlaced” predicate



## 4.4 Alloy Results

### Executing "Run show for 7"

Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20

61761 vars. 2233 primary vars. 191192 clauses. 450ms.

**Instance** found. Predicate is consistent. 696ms.

### Executing "Run callAmbulance"

Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20

18432 vars. 783 primary vars. 58704 clauses. 119ms.

**Instance** found. Predicate is consistent. 159ms.

### Executing "Run singleReqWellPlaced"

Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20

18328 vars. 783 primary vars. 58258 clauses. 216ms.

**Instance** found. Predicate is consistent. 179ms.

## 5. Effort Spent

Section	Student	Hours of work
Introduction	Enrico Voltan	3
Introduction	Antonio Urbano	5
Overall description	Enrico Voltan	2
Overall description	Antonio Urbano	3
Specific requirements	Enrico Voltan	7
Specific requirements	Antonio Urbano	4
Alloy model	Enrico Voltan	10
Alloy model	Antonio Urbano	10

## 6. References

- IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications.
- Specification Document: “Mandatory Project Assignment AY 2018-2019.pdf”.
- GPS Performances: “<http://www.gps.gov/systems/gps/performance/accuracy/>”.
- Google Fit APIs: “<https://developers.google.com/fit/>”.
- Apple HealthKit: “<https://developer.apple.com/healthkit/>”.
- Samsung Health SDKs: “<https://developer.samsung.com/health>”.
- Alloy documentation: “<http://alloy.mit.edu/alloy/documentation.html>”
- Slides provided on Beep channel



