POLITECNICO
MILANO 1863

Software Engineering 2

AA 2018/19

Requirements analysis and specification document

Authors:
Federico Ferri 10522586
Ahmad El Bayoumi 10425225

November 11, 2018

1 Introduction

1.1 Purpose

Nowadays it is hard for third parties (such as government or companies) to remotely monitor the location and the health status of a group of individuals, therefore the main purpose TrackMe service is to easily allow this kind of monitoring to provide services to citizens. The TrackMe service will be composed of two different services, Data4Help and AutomatedSOS.

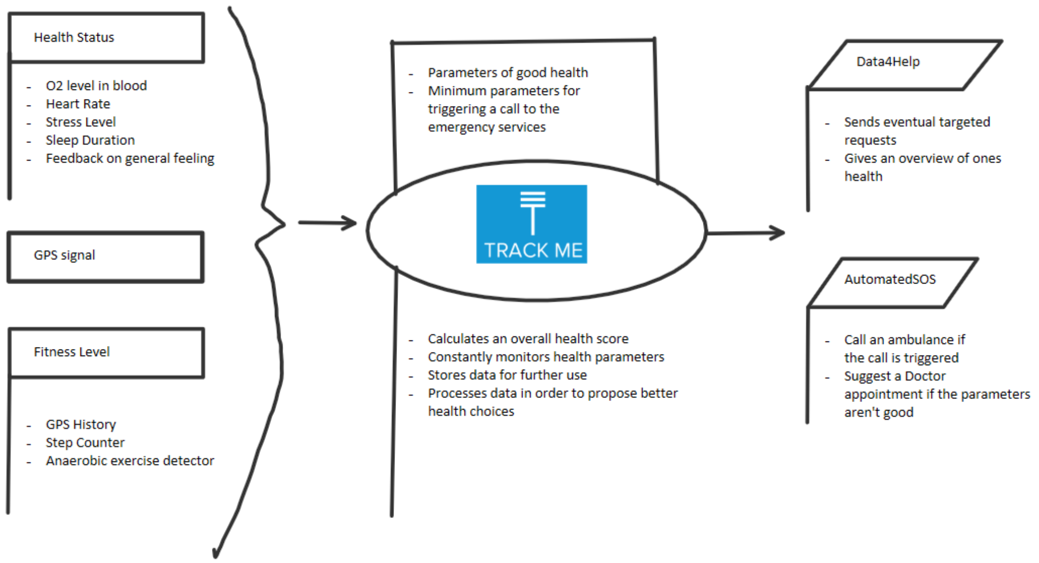
The Data4Help service is meant to be a software application that it it supposed to allow third parties to know the location and health status of individuals. The system will be able to handle two kind of requests:

- For a specific person: the system will accept requests targeted to a specific person provided that the third party knows the fiscal code of the person, in any case for specific requests as an additional security measure the approval of the person will be required.
- For a group of individuals: the third party will make requests targeted for a group of people that fits a certain profile, to guarantee anonymity and to preserve the privacy of the customer the system will automatically discard requests that are overly targeted, therefore the request will produce an output only if there are at least 1000 people that fit the criterias.

AutomatedSOS, as the name suggests, it's a non-intrusive SOS service based on Data4Help meant for senior citizens, AutomatedSOS constantly monitors the health status of the individuals and will eventually trigger an automatic call for an ambulance if the parameters fall below certain thresholds, in the call the GPS acquired position will be sent.

1.2 Scope

In the real world, organizations, whether public or private, want to have as many informations as possible from potential customers, in order to provide a service that will be "optimized" for the specific customer. However there is a big issue with information gathering, privacy, in order to preserve the privacy of the user data will be anonymized for "general" requests and TrackMe's services will provide additional information to the user about it's state of health. The system can be seen as a box that transforms the inputs of the individual, such as GPS position and parameters for the users' health (O2 level in blood, heart rate etc.) and elaborates an output for the user and for the companies that request the data.


Health Status
- O2 level in blood
- Heart Rate
- Stress Level
- Sleep Duration
- Feedback on general feeling
GPS signal
Fitness Level
- GPS History
- Step Counter
- Anaerobic exercise detector
- Parameters of good health
- Minimum parameters for triggering a call to the emergency services
TRACK ME
- Calculates an overall health score
- Constantly monitors health parameters
- Stores data for further use
- Processes data in order to propose better health choices
Data4Help
- Sends eventual targeted requests
- Gives an overview of ones health
AutomatedSOS
- Call an ambulance if the call is triggered
- Suggest a Doctor appointment if the parameters aren't good

Data4Help will, on request, provide third parties with the informations about a group of people or of an individual after approval, an overview of it's state of health will be given to the user

AutomatedSOS will provide senior citizens with their current health state and eventually will be able to suggest when to make an appointment to check their health with a physician. AutomatedSOS will even use their GPS in order to call an ambulance to their location in case the system suggests their life is at risk

1.3 Definitions, Acronyms, Abbreviations

DEFINITIONS

D-1 **Information**

noun - facts about a situation, person, event, etc.

D-2 **Request**

noun - the act of politely or officially asking for something

D-3 **Vital Signs**

noun - signs that show the condition of someone's health, such as body temperature, rate of breathing, and heartbeat

D-4 **Fitness Level**

noun - (Fitness + Level): the condition of being physically strong and healthy + the amount or number of something

D-5 **Position**

noun - the place where something or someone is, often in relation to other things

D-6 **Emergency Call**

noun - an occasion when you use the phone in order to speak with emergency services

D-7 **System**

noun - a set of computer equipment and programs used together for a particular purpose

D-8 **User**

noun - someone who uses a product, machine, or service

D-9 **Feasible request**

adjective + noun - a request that is able to be made, done, or achieved

ACRONYMS

ACR-1 **GPS**: Global Positioning System

ACR-2 **EC**: Emergency Call

ACR-3 **API**: Application rogramming Interface

ACR-4 **GP**: Group of People

ACR-5 **IP**: Individual Person

ACR-6 **DR**: Data Request

ACR-7 **UD**: User Data

ACR-8 **TP**: Third Party

ABBREVIATIONS

ABR-1 **D-n**: Definition number n

ABR-2 **ACR-n**: Acronym number n

ABR-3 **F-n**: Function number n

ABR-4 **AS-n**: Assumption number n

ABR-3 **RE-n**: Functional Requirement number n

1.4 Revision History

Version 1.0 Released on November 10th,2018 - 1 pm

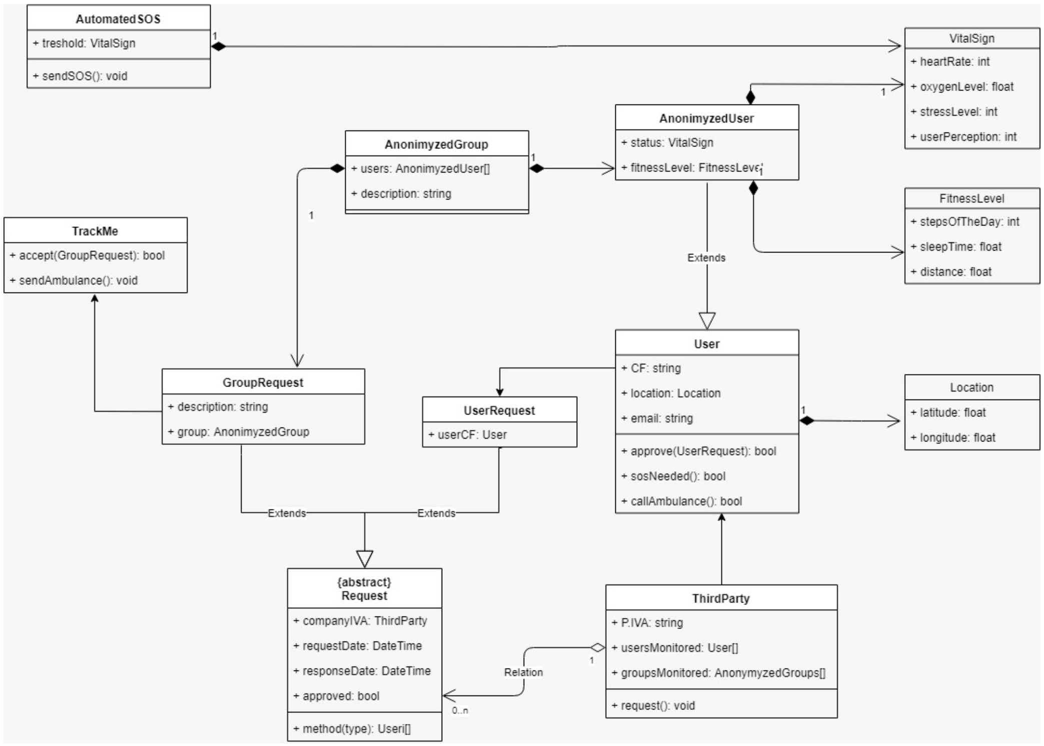
1.5 Reference Documents

Mandatory Project Assignment AY 2018-2019.pdf

2 Overall Description

2.1 Product perspective

The TrackMe system can be analyzed by observing the interactions between the user and everything that happens in it's life.Therefore all the constraints derived from this profile of an average user must be satisfied by the TrackMe system. The system is meant to be used on mobile devices, mainly smartphones and smartwatches. It is designed to use APIs from third party applications (health and GPS) as much as it is possible. To desccribe how the application will work the following UML diagram is provided:



2.2 Product Functions

The main functions of TrackMe will be:

Data4Help:

F-1 : The third parties should be able to make requests for informations

F-2 : The user should be able to accept/refuse individual requests

F-3 : The device should be able to monitor the user's vital signs

F-4 : The user should receive useful informations on how to improve it's general health

F-5 : The third parties should be able to subscribe to informations updates about a specific group that they previously targeted

AutomtedSOS:

F-6 : The devices should be able to call for an ambulance in case the vital signs drop

2.3 User characteristics

The target user of TrackMe is:

- someone who wants to keep his health in a good state
- an elder who wants an optimal solution to call for help in case of an emergency
- someone who would like society by giving data about his life

2.4 Assumptions,dependencies and constraints

On defining the system requirements of the application,we kept in mind that these assumptions are true:

Function 1

AS-1 : The third party association is registered on TrackMe

AS-2 : The third party has inserted the profile of user for which it is going to request informations

Funcition 2

AS-3 : The user is registered on TrackMe

AS-4 : The user connected a valid device to the system

AS-5 : The user received it's username and password to successfully login into the system

AS-6 : There has been at least one targeted request

Function 3

AS-7 : The user's registered device is detecting properly the vital signs

AS-8 : The user is successfully connected to the system

Function 4

AS-9 : The user successfully activated all of the health tracking sensors and the GPS

Function 5

AS-10 : The third party already sent a request about informations for a specific group

AS-11 : The third party subscribed for updates on informations

Function 6

AS-12 : The user is a senior citizen

AS-13 : An ambulance should always be ready to depart when the call comes

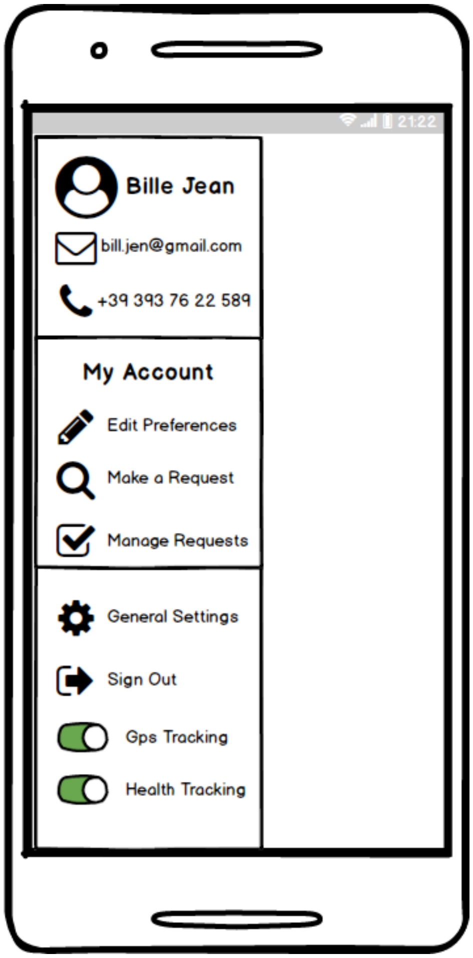
AS-14 : The user location is always available for the TrackMe system

3 Specific requirements

3.1 External Interface Requirements

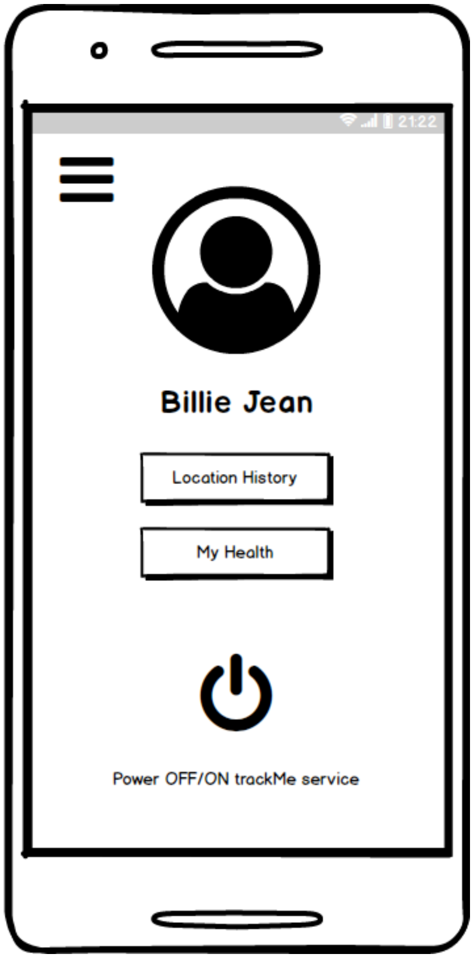
Being TrackMe a multi-device software, different interfaces are therefore needed. We though illustrate only the mobile applications for brevity purposes.

3.1.1 User Interfaces



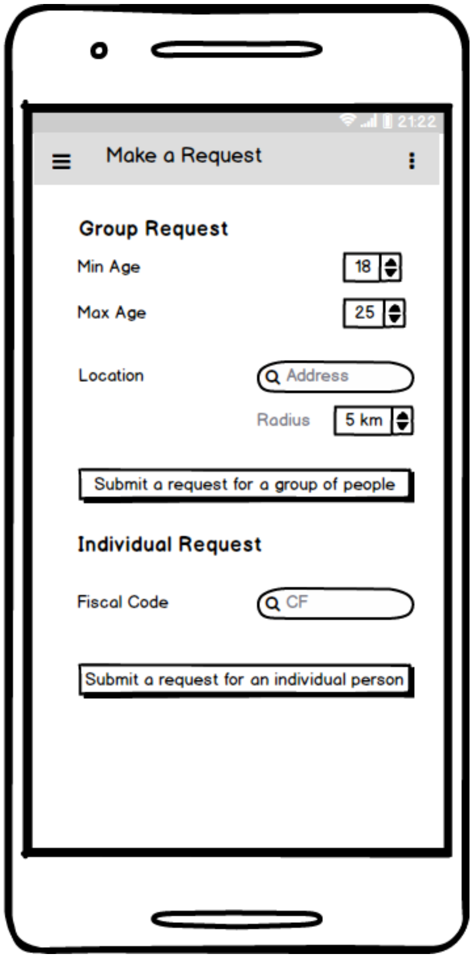
The picture on the left shows the main menu of TrackMe, it will be composed of three sections:

- The first part will be composed of the name of the user plus the email and the phone number all included eventually if the user opted to use one with a profile picture
- the second part will be used to interact with part of the app, here the user will be able to edit his preferences(such as which data to share), make a request for informations and manage his requests.
- the third and final part is composed of general settings for the app (notifications, updates and copyright informations), a log out button and a quick slider to turn OFF/ON gps and health tracking



On the left we can see the main page of TrackMe it can be divided in four parts:

- In the top part we see the profile picture, followed by the full name of the user
- The first element we notice after the profile picture is a button, called "Location History", it will open a page containing a map with an history of all of the places visited by the user
- The second button will be "My Health", it will contain all informations regarding the user's health and tips on how to live a healthier lifestile
- the last button is used to power off all of the services of track me, it will therefore make the user "invisible" to third parties that will try to track the user



21:22

Make a Request

Group Request

Min Age 18

Max Age 25

Location Address

Radius 5 km

Submit a request for a group of people

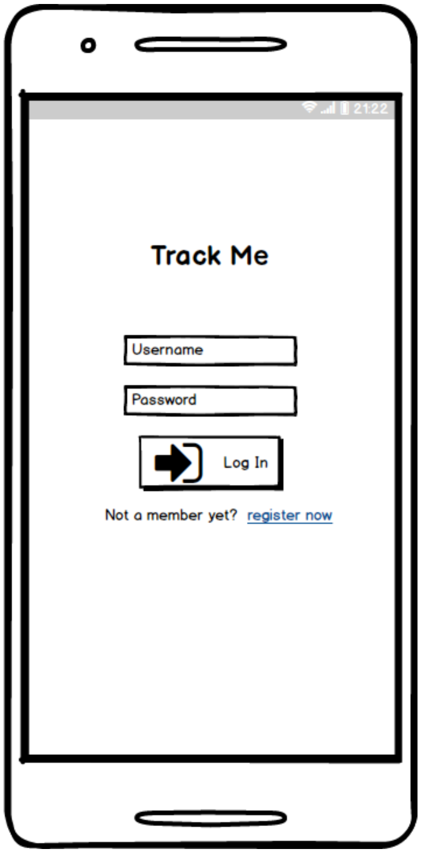
Individual Request

Fiscal Code CF

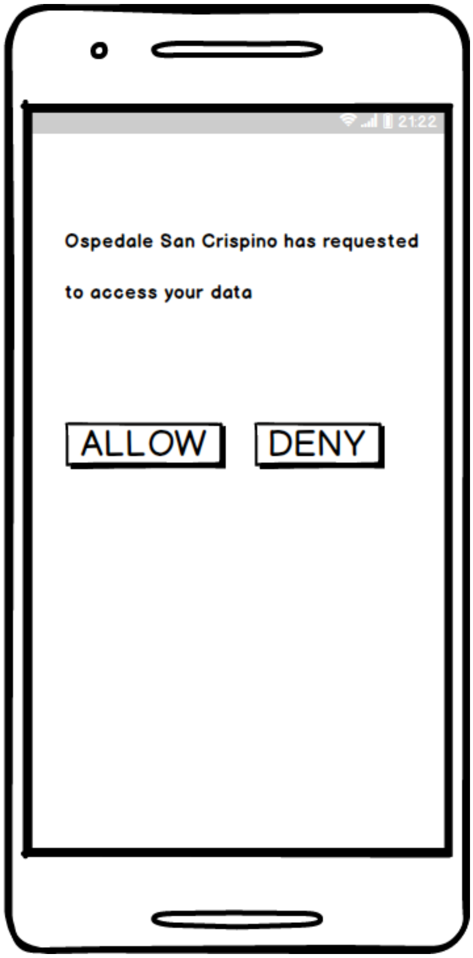
Submit a request for an individual person

The mock-up on the left shows a form to send request to TrackMe, the first part is for Group Request, while the second will be for individual request:

- The group request are made by inputting minimum and maximum age of the group, location of the center of the targeted area and a radius to extend the area
- The individual requests are instead made by filling the form with the Fiscal Code of the desired person,who will receive a notification to accept or deny the request



This is the login page for track me: the functionality is completely basic, it's made so that the user can insert it's email/username and it's password, than the user will have to press the login button. In case the user has not registered itself yet to TrackMe he/she will be able to do so by pressing the "register now" link



Ospedale San Crispino has requested to access your data

This is a simple screen that will be received when clicking on a notification for a request: The user will be therefore required to accept or deny the request for sharing the informations

3.1.2 Hardware Interfaces

The system does not require a specific hardware interface however it is required that the user uses a smartphone for tracking and eventually a smartwatch for health tracking

3.1.3 Software Interfaces

The system will be using external services that will be key to the success of the application in order to slim a bit the whole software appolication, these are:

- Maps: In order to provide a good GPS system and location history the system requires a service for the maps necessary to track the user in the correct way. The best option would be to use Google Maps, since as of right now is one of the most advanced softwares for navigation and positioning, therefore TrackMe will be using Google's API for maps location.

- Health tracking: To provide an efficient health tracking we need to access the sensors of the smartphone/smartwatch, the best way to do so would be to use API given from google fit, an application used for fitness tracking, through this we can monitor the vital signs of the user.

3.2 Functional requirements

3.2.1 Requirements

Function 1

RE-1 : The system must allow the user/third party to login using the correct credentials

RE-2 : The system must allow the third to insert a general profile for his targeted users

Function 2

RE-3 : The system must allow the user to register a new account

RE-4 : The system must allow the user to accept

Function 3

RE-5 : The system should ask the device for the permission to access certain sensors

RE-6 : The system should ensure that it will constantly run in the background

Function 4

RE-7 : The system should periodically analyze the data given by the user

RE-8 : The system should periodically output a report of the analyzed data

Function 5

RE - 9 : The system should allow third parties to subscribe to data from a specific group of people

RE - 10: The system should constantly monitor groups to which a third party is subscribed

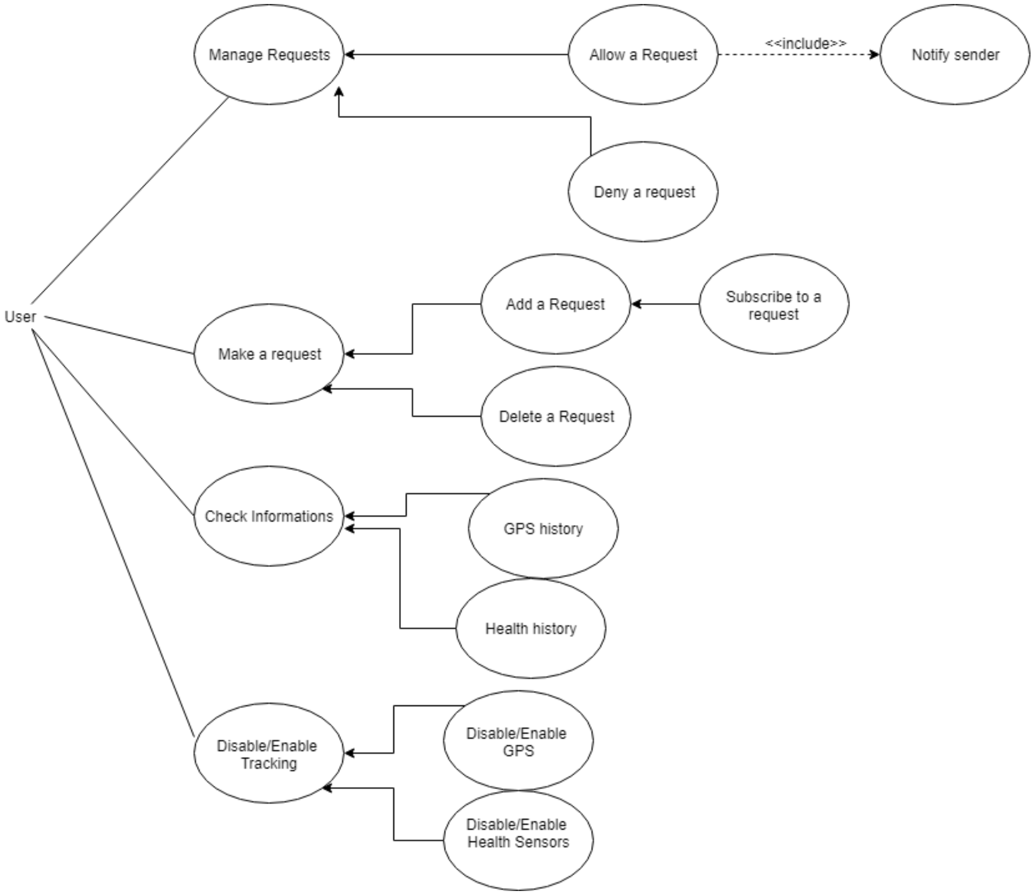
RE - 11: The system should automatically send informations, once updated, to third parties

Function 6

RE - 12 : The system should be constantly connected to the emergency services

RE - 13 : The system should ask for permission to make calls

3.2.2 Use Cases



Case: Sign Up

Actors: User.
Precondition: None

Flow of events:

1. The User enters the required User Account information values and requests that the system saves the entered values.
2. The system validates the entered User Account information.
3. The values for the User Account information are stored in the User's account. The system notifies the User that the account has been created..

Extensions: User Enters Invalid User Account Information: System shows the mistakes

Case: Log In

Actors: User.
Precondition: The user has already signed up

Flow of events:

1. The system requests that the user enter his/her name and password.
2. The user enters his/her name and password.
3. The system validates the entered name and password and logs the user into the
application.

Extensions: Invalid Name / Password: System shows the mistake and clean the inputs.

Case: Make a Request

Actors: User (Third Party).
Precondition: The user has decided the target of his research

Flow of events:

1. The user goes on the ”Make a Request” page.
2. The user enters the search criterias.
3. The system validates the criterias and depending on the type of request it will decide whether to authorize it or not.

Extensions: Invalid Research: System shows the mistake made by the user and cleans all of the inputs.

Case: Accept a request

Actors: User.
Precondition: The user has received a notification to allow/deny a request

Flow of events:

1. The user taps on the notification
2. The user reads the content of the request.
3. The user accepts/denies the request.
4. The system in case of acceptance will transmit the informations to the third party, in case of denial it will send a message the third party stating that the request was denied.

Extensions: No requests: the system will tell the user there are no available request at the moment.

Case: Call an emergency service

Actors: System
Precondition: The system received an emergency signal from the sensors of the user

Flow of events:

1. The systems checks whether the parameters are still not correct
2. The system sends a registered message to the emergency system
3. An ambulance will show up at the persons front door

Extensions: No emergency requests: the system will be on hold till a request is sent.

Case: Monitor own GPS history

Actors: user
Precondition: The user has turned on GPS

Flow of events:

1. The user goes to the "Location history" page
2. The system outputs all the saved locations inserted in the database

Extensions: GPS is off: the system will tell the user that it is not connected to the GPS.

Case: Check own health status

Actors: user
Precondition: The user has successfully connected the needed sensors

Flow of events:

1. The user goes to the "My Health" page
2. The system elaborates the data of the user saved on the database and generates a score for his health level
3. The system outputs the average data to the user and adds the "health level" score with tips on how to improve it's own health

Extensions: Sensors are off: the system will tell the user that there are no informations about him since the sensor were not properly configured.

3.3 Performance Requirements

Data4Help: PR-1 “The system must constantly track the user when possible, the downtime has to be as low as possible, at most 5PR-2 “The information about a request, if it does not need authorization, should be retrieved as quickly as in 10s” PR-3 “The system must guarantee that health services and tracking services operate perfectly together” PR-4 “The system must have an up-time of at least 99.99PR-5 “Notifications should appear as quickly as 10s afterwards an ”individual request” it’s made AutomatedSOS PR-6 “The reaction time of the monitoring device for the vital signs has to be 5 seconds”

3.4 Design Constraints

3.4.1 Standards compliance

- The app supports only vertical orientation of the screen, therefore horizontal will not be used
- The app requests the minimum numbero of permission possible to function properly
- all of the user data are stored both online and on the device fully encrypted
- the app is capable of retrieving previous session data from online database
- the app is capable of remembering previous requests for informations

3.4.2 Hardware limitations

There are no specific hardware limitations but if some elements are missing features will be limited. The recommended requirements are:

- iOS or Android Smartphone or AndroidWear Smartwatch
- Health sensors such as heartbeat and o2 level
- 3G/4G connection
- GPS/Galileo connectivity

3.5 Software system attributes

3.5.1 Reliability

The application must be available 24/7. It is ok for small offline times for maintenance purposes

3.5.2 Availability

In order to keep the system on as much as possible parallel servers will be kept on and the database will be based on RAID 1 arrays to avoid data loss, this should ensure an availability tine as close to 100% as possible

3.5.3 Security

User informations, including passwords,location history and health status are considered as highly confidential and therefore will be stored using encryption in order to prevent theft of the data.

3.5.4 Maintainability

The application will be very easy to maintain and to update, this will be achieved by making the application as ”modular” as possible, in this way it will be very easy too add modules or update previously created ones
Documentation of each module will be provided in order to keep the maintainability at it’s best level

3.5.5 Portability

This application is meant to be used on android and iOS devices, as well as having the possibility to send online requests for informations, because of the multitude of systems the app will have to be compatible with as many systems as possible.

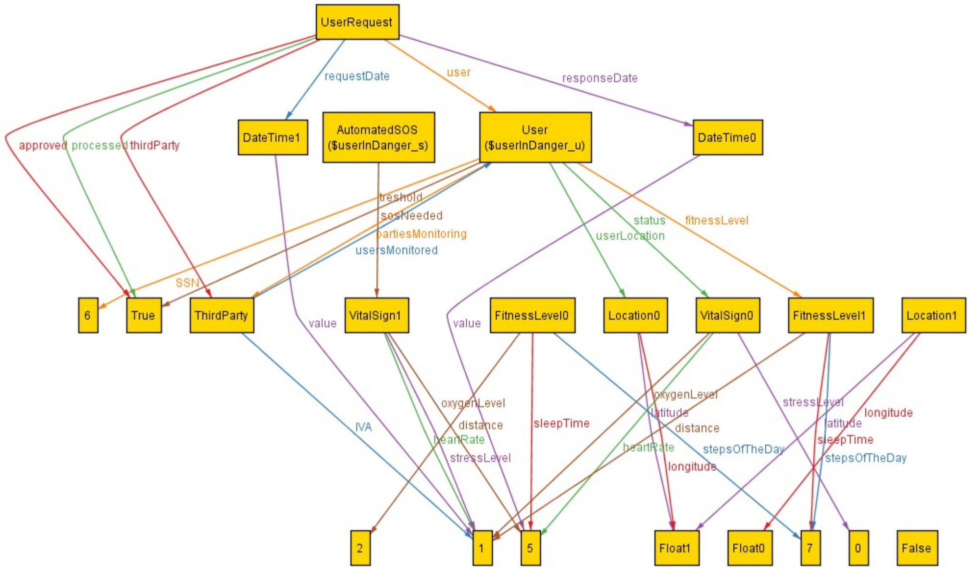
4 Formal analysis using alloy

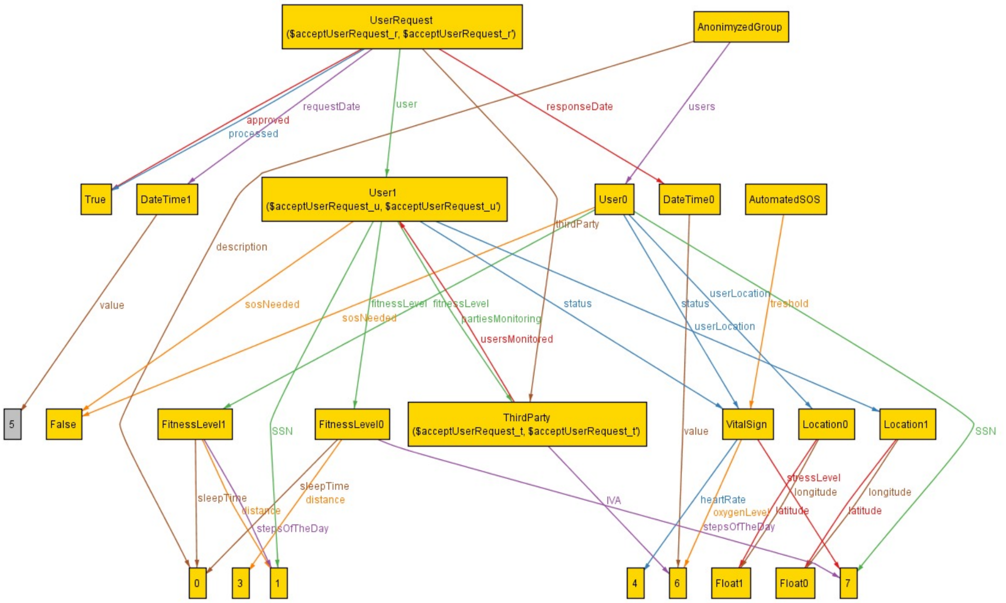
4.1 Alloy code

```
sig Float {}

sig Bool {}

sig True extends Bool {}

sig False  extends Bool {}

sig DateTime {
    value: one Int
} {
    value > 0
}

sig Location {
    latitude: one Float,
    longitude: one Float
}

sig VitalSign {
    heartRate: one Int,
    oxygenLevel: one Int,
    stressLevel: one Int
} {
    // Vital sign parameters must be positive
    heartRate ≥ 0
    oxygenLevel ≥ 0
    stressLevel ≥ 0
}

sig FitnessLevel {
    stepsOfTheDay: one Int,
    sleepTime: one Int,
    distance: one Int
} {
    // Fitness level parameters must be positive
    stepsOfTheDay ≥ 0
    sleepTime ≥ 0
    distance ≥ 0
}

sig AnonimyzedUser {
    status: one VitalSign,
    fitnessLevel: one FitnessLevel
}

one sig AutomatedSOS {
    treshold: VitalSign
} {
    // All the treshold parameters must obviously be greater then 0
    treshold.heartRate > 0
    treshold.oxygenLevel > 0
    treshold.stressLevel > 0
}

sig User extends AnonimyzedUser {
    SSN: one Int,
    userLocation: one Location,
    userPerception: lone Int,
    sosNeeded: one Bool,
    partiesMonitoring: set ThirdParty
} {
    SSN > 0
    userPerception ≥ 0
    // If a ThirdParty is in partiesMonitoring set, then the user must be in
        ↪  the usersMonitored set of that ThirdParty
    all p : partiesMonitoring | this in p.usersMonitored
}
fact {
    // If any vital sign parameter of the user is under the treshold, then
        ↪ user needs SOS (sosNeeded = true) and viceversa
    all u: User, s: AutomatedSOS | u.sosNeeded = True ≤> ((u.status.
        ↪ heartRate < s.treshold.heartRate) ∨ (u.status.oxygenLevel < s.
        ↪ treshold.oxygenLevel) ∨ (u.status.stressLevel < s.treshold.
        ↪ stressLevel))
    all u: User, s: AutomatedSOS | u.sosNeeded = False ≤> ((u.status.
        ↪ heartRate ≥ s.treshold.heartRate) ∧ (u.status.oxygenLevel ≥ s.
        ↪ treshold.oxygenLevel) ∧ (u.status.stressLevel ≥ s.treshold.
        ↪ stressLevel))
}

sig AnonimyzedGroup {
    description: one Int,
    users: set AnonimyzedUser
}

sig ThirdParty {
    IVA: one Int,
    usersMonitored: set User,
    groupsMonitored: set AnonimyzedGroup
} {
    IVA > 0
    // If a User is in usersMonitored set, then the ThirdParty must be in
        ↪ the partiesMonitoring set of that User
    all u : usersMonitored | this in u.partiesMonitoring
}

abstract sig Request {
    thirdParty: one ThirdParty,
    requestDate: one DateTime,
    responseDate: lone DateTime,
    approved: one Bool,
    processed: one Bool
} {
    // requestDate must be before responseDate
    isBeforeDateTime [requestDate, responseDate]
    // A request can not be approved if not processed
    processed = False implies approved = False
}

sig UserRequest extends Request {
    user: one User
} {
    // If request has been approved, then the User is succesfully monitored
        ↪ by the ThirdParty
    (processed = True and approved = True) ≤> thirdParty in user.
        ↪ partiesMonitoring and user in thirdParty.usersMonitored
    // If request has not been approved, then the ThirdParty can not monitor
        ↪  the User
    (processed = True and approved = False) ≤> thirdParty not in user.
        ↪ partiesMonitoring and user not in thirdParty.usersMonitored
    // If request has not been processed, then the ThirdParty can not yet
        ↪ monitor the User
    (processed = False and approved = False) ≤> thirdParty not in user.
        ↪ partiesMonitoring and user not in thirdParty.usersMonitored
}

sig GroupRequest extends Request {
    description: one Int,
    group: lone AnonimyzedGroup
} {
    // A GroupRequest can not be accepted if the target users are less then
        ↪ 1000
    approved = True implies #group.users ≥ 1000 and group in thirdParty.
        ↪ groupsMonitored
}

// There can not be more than 1 User with same SSN
fact userProps {
    no disj u, u' : User | u.SSN = u'.SSN
}

// There can not be more than 1 ThirdParty with same IVA number
fact thirdPartyProps {
    no disj p, p' : ThirdParty | p.IVA = p'.IVA
}

// There can not be more than 1 UserRequest from same ThirdParty to same
    ↪ User
fact userRequestProps {
    no disj r, r' : UserRequest | r.thirdParty = r'.thirdParty ∧ r.user = r
        ↪ '.user
}

pred isBeforeDateTime [dt, dt' : DateTime] {
    dt.value < dt'.value
}

pred acceptUserRequest [r, r' : UserRequest, u, u' : User, t, t' :
    ↪ ThirdParty] {
    t.IVA = t'.IVA
    u.SSN = u'.SSN
    r.user = r'.user
    r.thirdParty = r'.thirdParty
    r.user = u
    r.thirdParty = t
    t'.usersMonitored = t.usersMonitored + u
    u'.partiesMonitoring = u.partiesMonitoring + t
    r'.approved = True
    r'.processed = True
```

```
}

pred acceptGroupRequest [r, r' : GroupRequest, g : AnonimyzedGroup, t, t' :
    ↪ ThirdParty] {
    t.IVA = t'.IVA
    r.thirdParty = r'.thirdParty
    r.thirdParty = t
    t'.groupsMonitored = t.groupsMonitored + g
    r'.approved = True
    r'.processed = True
}


pred userInDanger [s : AutomatedSOS, u : User] {
    (u.status.heartRate < s.treshold.heartRate
    or
    u.status.oxygenLevel < s.treshold.oxygenLevel
    or
    u.status.stressLevel < s.treshold.stressLevel)
    u.sosNeeded = True
}

run acceptUserRequest

run userInDanger
```

4.2 Alloy diagrams





5 Effort Spent

5.1 Federico Ferri

Section 1(Purpose,Scope,Definitions,Acronyms,Abbreviations): 12 hours
Section 2(Product perspective,functions,User characteristics): 10 hours
Section 3(External Interface Requirements,Functional Requirements):9 hours

5.2 Ahmad El Bayoumi

Section 2(Assumptions,dependencies,constraints,UML): 7 hours
Section 3(Performance Requirements,Design Constraints,Software system attributes): 12 Hours
Section 4(Alloy code,Alloy Diagram): 11 hours

6 References

- Specification document “AA 2017-2018 Software Engineering 2—Mandatory Project goal,schedule, and rules”
- IISO/IEC/IEEE 29148 - Standard on requirement engineering
- Google API for android- https://developers.google.com/android/guides/api-client
- Alloy documentation - http://alloy.mit.edu/alloy/documentation.html