

Software Engineering 2: TrackMe

Requirements Analysis and Specification  
Document

Haritha Harikumar, Mohini Gupta, Saloni Kyal  
Politecnico di Milano

November 11, 2018

# Contents

<b>Contents</b>	<b>1</b>
<b>1 Introduction</b>	<b>3</b>
1.1 Purpose . . . . .	3
1.2 Scope . . . . .	3
1.3 Goals . . . . .	5
1.4 Definitions, Acronyms, and Abbreviations . . . . .	6
1.4.1 Definitions . . . . .	6
1.4.2 Acronyms . . . . .	6
1.4.3 Abbreviations . . . . .	6
1.5 Document Structure . . . . .	7
<b>2 Overall description</b>	<b>8</b>
2.1 Product perspective . . . . .	8
2.1.1 Class Diagram . . . . .	8
2.1.2 State Charts . . . . .	9
2.2 Product functions . . . . .	10
2.2.1 Vital Status . . . . .	10
2.2.2 Location Tracking . . . . .	10
2.2.3 Third Party Validation . . . . .	10
2.2.4 Organizing a Race . . . . .	11
2.2.5 Expeditious Data Convey . . . . .	11
2.2.6 Real Time Visualization of Race . . . . .	11
2.3 User characteristics . . . . .	12
2.3.1 Admin . . . . .	12
2.3.2 Individual . . . . .	12
2.3.3 Third Party . . . . .	12
2.3.4 Organizers . . . . .	12
2.3.5 Athletes . . . . .	12
2.3.6 Spectators . . . . .	12
2.3.7 Ambulance Drivers . . . . .	13

2.4	Assumptions, Dependencies and Constraints . . . . .	13
2.4.1	Assumptions . . . . .	13
2.4.2	Dependencies . . . . .	14
2.4.3	Constraints . . . . .	14
<b>3</b>	<b>Specific requirements</b>	<b>15</b>
3.1	External interface requirements . . . . .	15
3.1.1	User Interfaces . . . . .	15
3.1.2	Software Interfaces . . . . .	22
3.1.3	Communication Interface . . . . .	23
3.2	Functional requirements . . . . .	23
3.3	UML Modeling . . . . .	27
3.3.1	Use Case Diagrams . . . . .	27
3.3.2	Sequence Diagram . . . . .	39
3.4	Performance Requirements . . . . .	45
3.5	Design Constraints . . . . .	46
3.5.1	Standards Compliance . . . . .	46
3.5.2	Hardware Limitations . . . . .	46
3.6	Software System Attributes . . . . .	46
3.6.1	Reliability . . . . .	46
3.6.2	Availability . . . . .	46
3.6.3	Security . . . . .	46
3.6.4	Maintainability . . . . .	46
3.6.5	Portability . . . . .	47
<b>4</b>	<b>Alloy Model</b>	<b>48</b>
4.0.1	Description . . . . .	48
4.0.2	Alloy code . . . . .	49
4.0.3	Result . . . . .	54
4.0.4	Alloy Model . . . . .	55
<b>5</b>	<b>Effort</b>	<b>56</b>
5.0.1	Haritha Harikumar . . . . .	56
5.0.2	Saloni Kyal . . . . .	57
5.0.3	Mohini Gupta . . . . .	58
<b>6</b>	<b>References</b>	<b>59</b>

# Chapter 1

## Introduction

### 1.1 Purpose

TrackMe is an android application developed by TrackMe. The application helps to monitor health vitals and location of individuals. TrackMe stands apart from other health monitoring application with its features that enables 3rd party services to monitor the data of individuals for a valid cause. TrackMe also provides other useful services like emergency alerts to ambulances in case an individual's vitals are in critical condition. Apart from that TrackMe comes with a service which offers people or a group of people to organize and visualize races or marathons.

This following document focuses on describing TrackMe system with respect to its goals and requirements and then formulating the functional and non-functional requirements. The document also provides an ample idea of the process flow and a model of the application with the help of diagrams and mock-ups.

Further, this document will provide formal specification of some features of the applications, by means of the Alloy language [?].

### 1.2 Scope

Track4Me is a company which provides three software based services. The services are limited to the citizens of Milan. The given problem is to design and develop a software service called "Data4Help", which acts as an intermediary between the individuals and the other companies. It targets two set of users, individuals and third party. The location and health of the user is extracted from a wearable device. The data from the wearable device is extracted with the help of the API of the wearable device. The software

tracks the location and health status of individuals or group of individuals and send the information to the third party on their request to monitor the health of individuals by validating the type of requests. The validation of the requests for the information of specific individuals is done by the individual's self acceptance or rejection. The requests for the information of group of individuals is decided by Track4Me and generalized by allowing the access to the third party, if the number of individuals whose information is to be accessed is greater than 1000.

It builds a new service called "AutomatedSOS" over the top of "Data4Help" after monitoring the demand of requests from third parties for giving non-intrusive SOS service to elderly people, which sends the ambulance to individual's location when the health parameters are below the thresholds. The nearest ambulance is tracked and sent to ensure individual's quick and efficient recovery. A separate mobile application of Track4Me is to be exclusively developed for the ambulance drivers who are informed by push notifications in the mobile application along with the location and other details of the patient in less than 5 seconds to ensure spontaneity in reaction to emergencies. All the hospitals in Milan are informed to ensure all the ambulance drivers installs TrackMe application.

A new service called "Track4Run" is built as a source of revenue which allows organizer to organize an event, athletes to participate in the event and spectators to see the athletes position live in a Map. It exploits the service of "Data4Help" which tracks the location of the athletes to be visible to everyone and helps the organizer to monitor the health status of athletes as well. The organizer will be registered as an existing or new third-party. The athletes will be registered as existing or new individual and the spectators can be registered as new/existing individual/third party.

The users can either register as an individual or as an organization/ company (third party). Data4Help is a base service which all the individuals and third party will have by default even if the user registers for a different service other than "Data4Help". The third party can register for two services: "Data4Help" and "Track4Run". It uses "Data4Help" for receiving information about targeted individuals and "Track4Run" to organize events for a particular cause/occasion. The third party is a source of revenue for both cases, as Track4Me provides information about individuals and thereby, provides customers to their organization which brings them profit. Individuals can register for any of the three services. The first service is a free service with a motive to help them to receive specific input from respective third party for their health issues or other factors. The individual can also upgrade to other services if they are an existing a member using one service by making payment. Payment can be made using PayPal or a credit card.

Track4Me guarantees the security of the individuals data and only associates with trusted third party companies. Ultimately, it helps both individuals and third party to manage and monitor their personal data and aims at being a robust and efficient software.

## 1.3 Goals

- [G1] Providing a list of the wearable devices that an individual can choose from.
- [G2] Expedite the request made by the 3rd party to an individual to access their details.
- [G3] Provides an interface for an individual to analyze the request made by the 3rd party.
- [G4] Validating the request made by the 3rd party to provide the details of a group of individuals.
- [G5] Quick access of the data to the 3rd party once the request is approved.
- [G6] Immediate update of the data once an individual updates their own details.
- [G7] Provides an up-gradation to the new services, Automated SOS or Track4Run.
- [G8] Monitoring the health status of the subscribed individual to the Automated SOS service.
  - [G8.1] Alerts the nearby ambulance, when the health status of the individual is low, within 5 seconds.
- [G9] Develops an interface to organize a race for the subscribed 3rd party to the Track4Run service.
- [G10] Give access to the details of the race once an individual participate in it.
- [G11] Live visualization of race.

## **1.4 Definitions, Acronyms, and Abbreviations**

### **1.4.1 Definitions**

1. Services: Functionalities offered by the application
2. Validation: Verifying if requests provided by the 3rd party are genuine or not. If it is genuine it is accepted else rejected.
3. Upgrade: Enhancing the functionalities.
4. Update: Adding new information or changing the previous one.
5. Threshold: A limit. Threshold on age has been kept for Automated SOS service.
6. Push Notification: The message that pops up on the Mobile.

### **1.4.2 Acronyms**

1. RASD: Requirement Analysis and Specification Document
2. API: Application Program Interface
3. GPS: Global Positioning System
4. SOS: Save our Souls

### **1.4.3 Abbreviations**

1. Gn: n-Goals
2. An: n-Assumptions
3. Dn: n-Dependencies
4. Cn: n-Constraints
5. Rn: n-Functional Requirement s

## 1.5 Document Structure

The RASD is composed of 6 parts including references

1. Chapter 1 is an introduction to the problem which identifies the scope and goals of the application to be developed. Apart from this different definitions, acronyms and abbreviations used in the document are also noted down here.
2. Chapter 2 comprises of the overall description of the application which includes product functionalities, the users involved and the assumptions, dependencies and constraints.
3. Chapter 3 takes into the specific requirements like the interfaces used, functional requirements and the different unified modeling language diagrams and the non-functional requirements.
4. Chapter 4 contains the Alloy code and metamodel we have created.
5. Chapter 5 contains the effort spend by each of the Team Members.
6. Chapter 6 of the document has all the references.



# Chapter 2

## Overall description

### 2.1 Product perspective

#### 2.1.1 Class Diagram

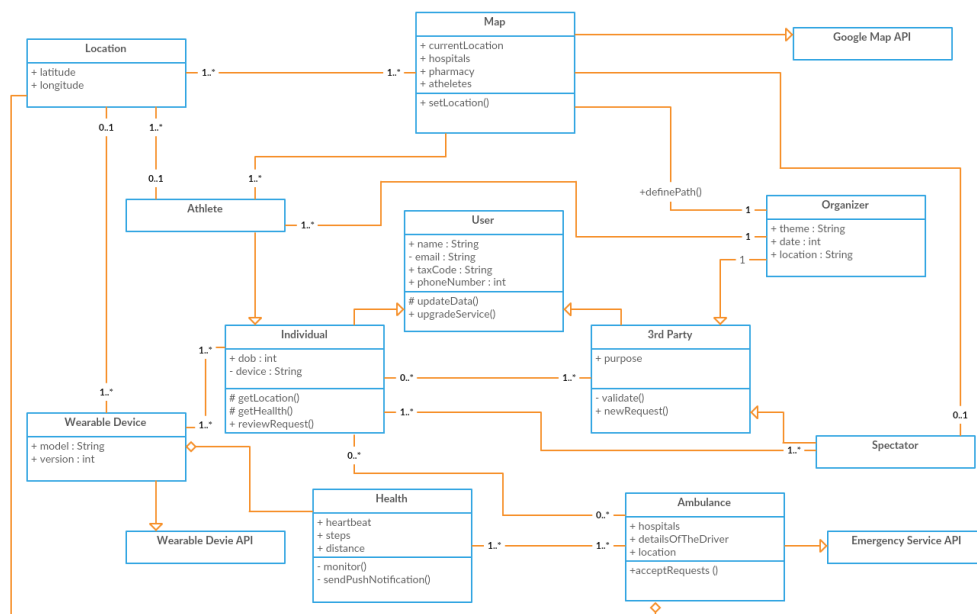


Figure 2.1: Class Diagram for TrackMe

### 2.1.2 State Charts

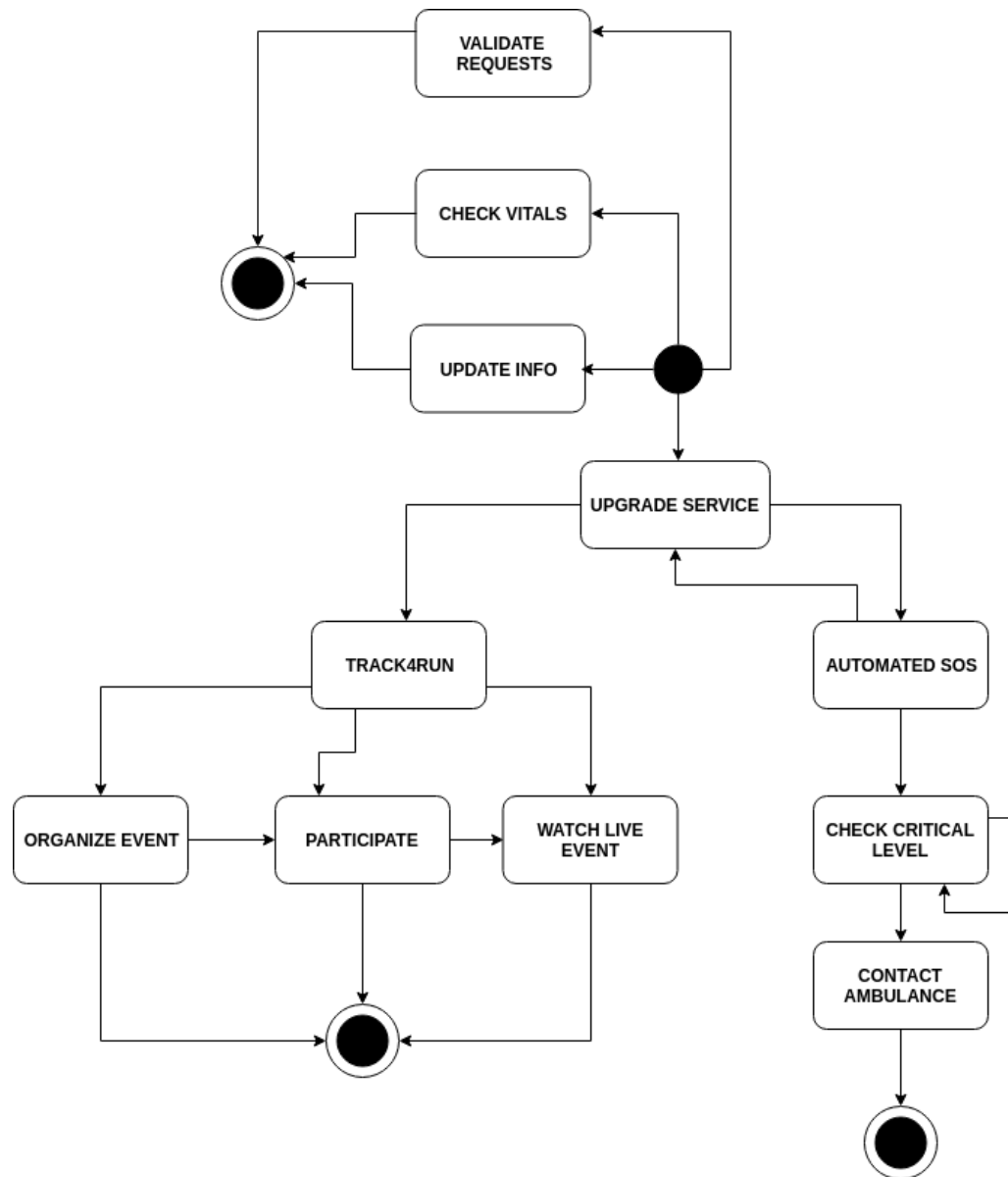


Figure 2.2: State Diagram for TrackMe

## **2.2 Product functions**

### **2.2.1 Vital Status**

TrackMe uses its service named Data4Help to share the health status of individuals to be monitored, to the third party on their requests. The health status of the individual will be extracted from a wearable device through the API of the wearable device about which the individual will mention while registering by selecting their device among the list of compatible wearable devices of TrackMe. The health parameters which are extracted includes heart beat rate, no. of steps, body temperature, detection of seizures, tremors etc. This is an important and crucial function of the software because it deals with individuals health and safety. Track4Me launches a new service called “AutomatedSOS” where TrackMe itself monitors the individual’s health status and do the needful.

### **2.2.2 Location Tracking**

Data4Help helps TrackMe to use monitor the location of the individual which can be later shared with the third party after validation. This service of location tracking can be helpful in case of emergencies which we come across in automated SOS. We basically use the Global Positioning System or the GPS to track the location of the individual. There are chances that we can lose the GPS connection in certain cases like may be inside our home due to poor connection to satellite. In such cases we can fetch the location from the nearest wi-fi to which our device get connected. In case of an emergency TrackMe system shall locate a minimum of 3 available ambulances that are closest to the emergency location by displaying a map and marking location of the emergency and ambulance nearby, which is would be taking less than a minute to fetch. We then share the data to the nearest ambulance within 5 seconds. We also use location tracking for athletes for a real time visualization of the positions of the athletes for a particular race which is explained further below.

### **2.2.3 Third Party Validation**

In order to maintain the privacy of an individual customer, the 3rd party is validated by the system. The 3rd party has to provide a valid proof and request the data of an individual customer. The customer, in turn, gets all the details of the 3rd party that has requested their data. Customer decides whether to accept or refuse the request of the 3rd party. TrackMe handles

the request made by the 3rd party, when the data requested is of a group of customers. The request is only approved when data requested is of more than 1000 customers.

#### **2.2.4 Organizing a Race**

TrackMe provides an interface for the subscribed 3rd party to organize a race. An organizer needs to set realistic goals and mention a purpose of the race. Then, an organizer has to select the location, date and time. The organizer needs to upload the Government Permission to organize a race. As an individual join the race organized by the organizer, the details of the individual is instantly available to the organizer. List View is provided of all the participants for the organizer.

#### **2.2.5 Expeditious Data Convey**

Track4Me monitors the health status themselves when the individual uses the service “AutomatedSOS” rather than sending the data to the third party to be monitored. When the health parameters go below the threshold, Track4Me tracks the location of the nearby ambulance to the patient. The nearby ambulance is tracked with the help of external emergency services API. The nearby ambulance drivers then receive a request through push notification and the one who accepts the request first goes to the location of the patient. A separate mobile application of Track4Me is to be exclusively developed for the ambulance drivers to send the location and other details of the patient in less than 5 seconds to ensure spontaneity in reaction to emergencies. All the hospitals in Milan are informed to ensure all the ambulance drivers installs TrackMe application.

#### **2.2.6 Real Time Visualization of Race**

This service is basically used by the spectators who wish to watch a race or marathon live, which comes as a part of the Track4Run. Once an athlete registers with the Track4Run service. His/Her location is tracked live with the help of the device using GPS. Track4Run fetches the track details way before and set it as the Track for a particular race or marathon and shows an enlarged view of the track while displaying the live positions of the participants.

## **2.3 User characteristics**

### **2.3.1 Admin**

Admin is the TrackMe application and the team handling the operations of the application. All the functions handled and done by TrackMe is denoted here to be done by the admin.

### **2.3.2 Individual**

A person using TrackMe for monitoring his/her vitals using the Data4Help service. He/She gets to register for the service by making an monthly payment and his/her data would be tracked and monitored by the TrackMe.

### **2.3.3 Third Party**

This can be a person or a company who wish to get the data for the users for some work of his/her concern. They have to register by making a certain payment amount as specified by TrackMe and a valid purpose for the data transfer. The transfer of data is based after proper analysis and validation by the TrackMe.

### **2.3.4 Organizers**

Organizers are those people who use the Track4Run service, who wish to organize an event like a race or a marathon for a cause. They have to register the event with the details of the event, location, date, time and so on. Track4Run is a paid service and hence organizers have to register with a specific cost.

### **2.3.5 Athletes**

These are those people who join for a specific event organized by mentioning details like name, contact and wearable device to be used and complete the registration by making the payment.

### **2.3.6 Spectators**

These are people who wish to avail the service of Track4Run just to watch the race. He/She can just join to watch a particular race by providing basic details like name, email and the one time payment amount. They can also

upgrade their service to watch the upcoming events or receive notifications on upcoming races.

### **2.3.7 Ambulance Drivers**

TrackMe has an application for the Ambulance drivers who gets push notifications from Automated SOS service.

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

### **2.4.1 Assumptions**

- [A1] Users should be registered in the TrackMe application to avail its services.
- [A2] There exist a mobile application of TrackMe for individuals and third party.
- [A3] Vitals taken from the wearable devices are reliable.
- [A4] Internet connection will be active in users phone.
- [A5] GPS provides accurate positions and in case the signal is lost, last active location would be considered.
- [A6] We assume that a certain number of hospitals are registered to with TrackMe which provide Ambulance services.
- [A7] We assume there are no network issues between the time we sent a message and the drivers receive it.
- [A8] Organizers can keep a maximum limit of the participants who can join the event.
- [A9] An event is organized for a reason.
- [A10] All athletes have their own wearable device.
- [A11] A user can be an individual or a third party. An individual can either belong to the people availing Data4Help or they can be spectator. An organizer belongs to the third party group. An athlete belongs to the individual group.
- [A12] All details entered by the users are genuine.

- [A13] We take permission from user to access the location and further details.
- [A14] There is an external service that will be in charge of the payment information and to ensure secure payment.
- [A15] Payment Details are genuine and transactions are processed positively.

### **2.4.2 Dependencies**

- [D1] Wearable device API
- [D2] Google Maps API
- [D3] Emergency services API
- [D4] Payment external gateway

### **2.4.3 Constraints**

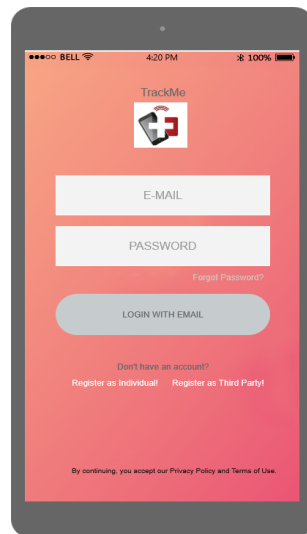
- [C1] Only a few wearable devices are compatible with application.
- [C2] The services provided by the application are limited to the citizens of Milan.
- [C3] This application is connected only with the major hospitals in Milan for Ambulance services.

# Chapter 3

## Specific requirements

### 3.1 External interface requirements

#### 3.1.1 User Interfaces



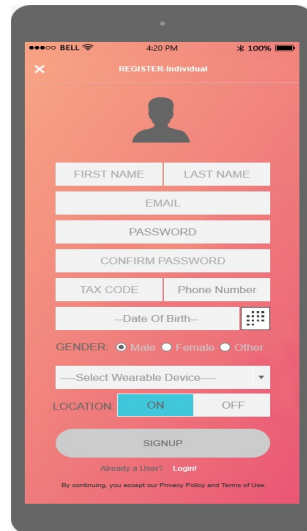
(a) Login for TrackMe

This is the login page where a user of type individual or third party can login by entering their credentials. If a user is new then they can register as individual or third party depending on their category.



## Data4Help

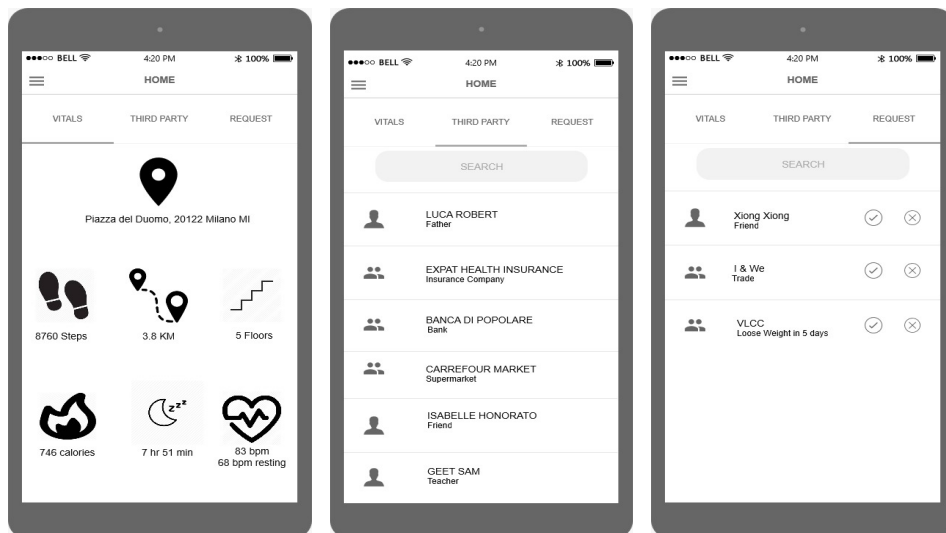
### 1. User Type: Individual



Registration form for individuals. The form includes fields for First Name, Last Name, Email, Password, Confirm Password, Tax Code, Phone Number, Date of Birth, Gender (Male, Female, Other), Select Wearable Device, and Location (ON/OFF). A SIGNUP button is at the bottom, along with a link for 'Already a User? Login!' and a note: 'By continuing, you accept our Privacy Policy and Terms of Use.'

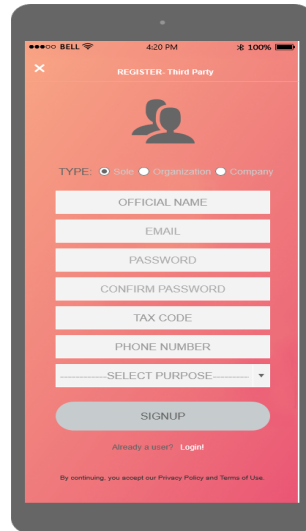
(a) Registration for individuals

The individual can check their vitals, and get to know about the third parties accessing their data and can receive the request of the third party which they can accept/reject when the third party asks the data about a specific individual.



(a) Individual Checks Vitals (b) Third party access- ing individual's data (c) Accept/Reject access request

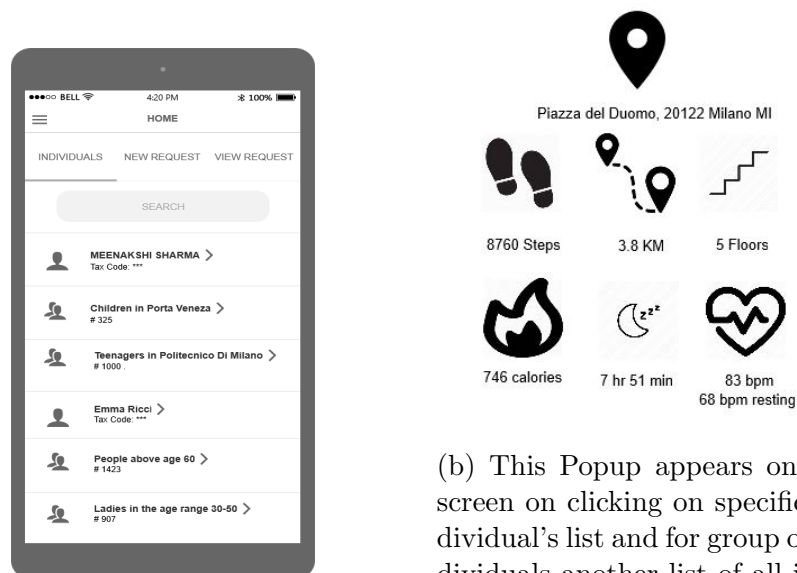
## 2. User Type: 3rd Party



The image shows a mobile app interface for registering a third party. The screen has a red-to-orange gradient background. At the top, there's a status bar with 'BELL', '4:20 PM', and '100%' battery. Below the status bar is a close button (X) and the title 'REGISTER - Third Party'. A person icon is centered. Below it, the 'TYPE' section has three radio buttons: 'Self' (selected), 'Organization', and 'Company'. The form includes input fields for 'OFFICIAL NAME', 'EMAIL', 'PASSWORD', 'CONFIRM PASSWORD', 'TAX CODE', and 'PHONE NUMBER'. There's a 'SELECT PURPOSE' dropdown menu. A 'SIGNUP' button is at the bottom. Below the button, it says 'Already a user? Login!'. At the very bottom, a small line of text reads 'By continuing, you accept our Privacy Policy and Terms of Use.'

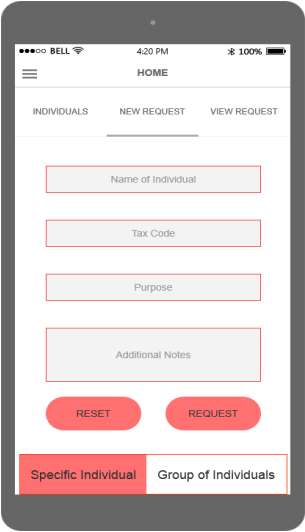
(a) Registration for Third Party

Third party can check individual's or group of individual's data, make new requests for specific individual's data and group of individual's data, and view the status of earlier requests made to know whether they are requested, accepted or pending.

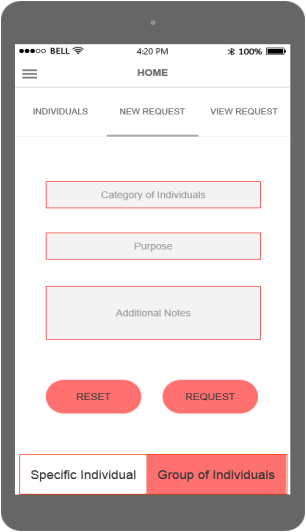


(a) Access individual's data

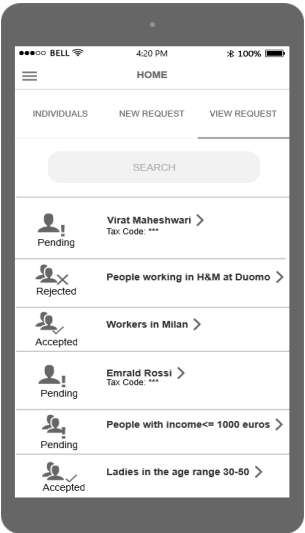
(b) This Popup appears on the screen on clicking on specific individual's list and for group of individuals another list of all individuals appears with their vitals.



(a) Make new request for individual



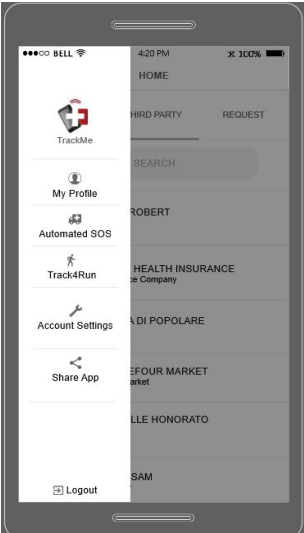
(b) Make new request for Group of Individuals



(c) View Requests

**Menu**

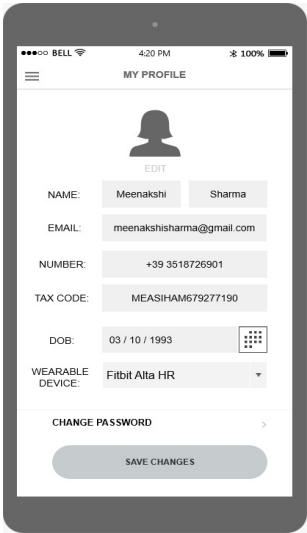
This includes all the extra features which is offered to the users. The menu remains same for both type of users: individual or third party.



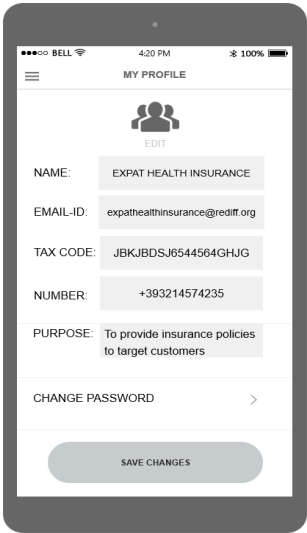
(a) Menu

**Menu-1. My Profile**

User can update their data in MyProfile section.



(a) Individual-MyProfile



(b) 3rd Party-MyProfile

## Menu-2. AutomatedSOS

This is only for individuals .

If they have not registered tot his service then they will be directed to the payment page where they can pay and register for the service.

If they have registered to this service, then they see a map with nearby hospitals and pharmacies.

In case of emergencies, they can track ambulance in the map.



(a) AutomatedSOS

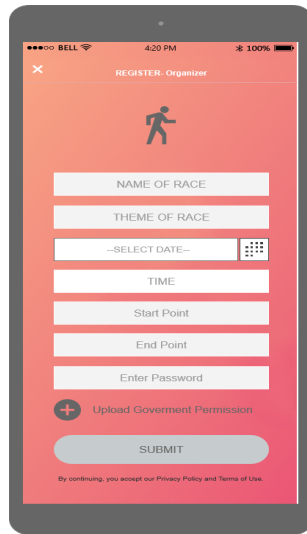
## Menu-3. Track4Run

This service is for both the users, individual and third party.

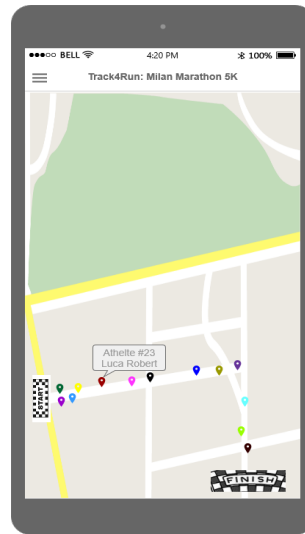
If the user has not upgraded to this service, then for individual who can be an athlete or a spectator and the third party who can be a spectator will be directed to a payment page to pay and use the service.

The third party organizer who has not registered will be directed to a form and then a payment page. The organizer defines the path, theme of race and other details of the race in the form.

If the user has registered, then they will see a live map with all the position of the athletes.

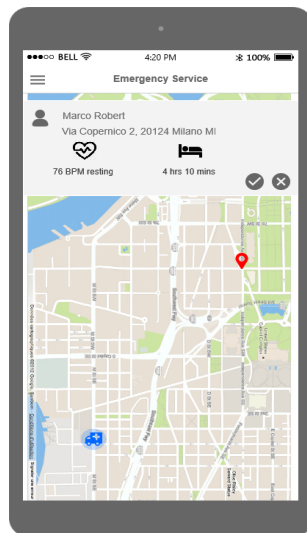


(a) Organizer form

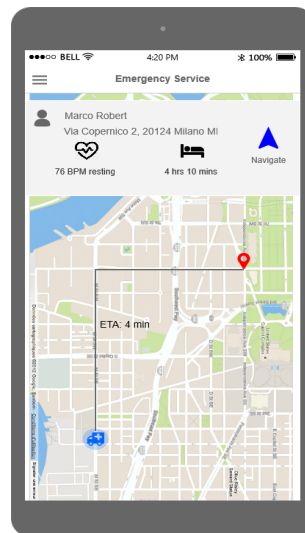


(b) Track4Run

**Mobile Application for Ambulance Drivers** This is an application made explicitly for Ambulance where they receive request in form of push notification and they accept it and the first driver to accept is navigates to the individual's location.



(a) Accept Request



(b) Navigate

### **3.1.2 Software Interfaces**

#### **1. Data4Help API**

TrackMe extracts location and health of the individual from the API of the wearable device and stores it in a database. An API of the database is created which is provided to the other services of TrackMe. This information is used for many purposes. “AutomatedSOS” and “Track4Run” uses the API of the “Data4Help” in order to provide their service. Both the services exploit the services of “Data4Help”. “AutomatedSOS” uses the API to monitor the health status of elderly people and help them by providing an ambulance. “Track4Run” uses the API to extract the location and health of individuals registered as athletes. The location will be displayed to all the users and the health status of athletes will be monitored by the third party organizers.

#### **2. Ambulance Application**

TrackMe develops an application for all the ambulance drivers associated with major Hospitals in the city of Milan. The application fetches the details and location of all the ambulance driver. When the health status of an subscribed individual is below a threshold value, the software sends a push notification to the nearby ambulance drivers phone with the details and location of the individual. Once an ambulance driver accept the request, the software removes the alert message from the application. The driver can navigate to the location of the individual in need of emergency using the map provided in the application.

#### **3. City Maps**

We will be using Google Maps API to facilitate the location monitoring for third party services and insertion of race location in Track4Run and race visualization.

### 3.1.3 Communication Interface

The TrackMe application provides the Data4Help API which is used by the third party after validating for accessing individual's details and Track4Run for accessing the location of athletes. It also provides the details to the ambulance application API to use Automated SOS. All the external API's like Emergency Service API, Wearable Device API and Google Map API are connected as they provide information.

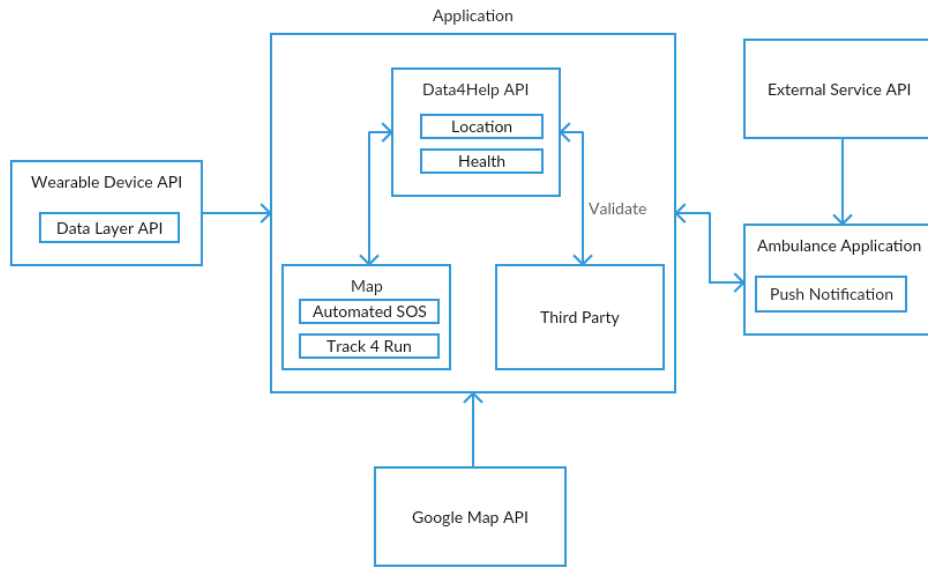


Figure 3.12: Communication Interface

## 3.2 Functional requirements

In order to fulfill the goals, with all the domain properties and assumption, listed in the above section of the document, the following requirements can be derived. Each goal has its own set of requirement and they are mentioned below.

- [G1] Providing a list of the wearable devices that an individual can choose from.
  - [R1] The location and the health status of an individual should be fetched by the wearable device API.



- [R2] Individual should be able to update their wearable device.
  - [A3] Vitals taken from the wearable devices are reliable.
  - [A5] GPS provides accurate positions and incase the signal is lost, last active location would be considered.
  - [C1] Only a few wearable devices are compatible with application.
- [G2] Expedite the request made by the 3rd party to an individual to access their details.
  - [R3] Individual must be able to view the name and the purpose/relationship of the 3rd party who requested the data.
  - [R4] The details of all the 3rd party accessing the individual's data must be visible.
- [G3] Provides an interface for an individual to analyze the request made by the 3rd party.
  - [R5] The system must provide a choice to an Individual to accept or reject the request of the 3rd party.
- [G4] Validating the request made by the 3rd party to provide the details of a group of individuals.
  - [R6] The system checks whether the request made by the 3rd party is of more than 1000 Individuals.
  - [R7] The details of the group of individual must be displayed in a list.
  - [R8] The details of an individual should be displayed in a pop up dialog.
- [G5] Quick access of the data to the 3rd party once the request is approved.
  - [R9] Allow a 3rd party to an instant access of the previously saved data after it is approved.

- [R10] 3rd party must view all the updated data of the individual.
- [G6] Immediate update of the data once an individual updates their own details.
  - [R11] An individual should be able to update all their details.
  - [R12] Allow a 3rd party to access the new data as soon as the individual updates it.
  - [A12] All details entered by the users are genuine.
- [G7] Provides an upgradation to the new services, Automated SOS or Track4Run.
  - [R13] They system shall redirect to a payment gateway once a new service is selected.
  - [D17] Payment Details are genuine and transactions are processed positively.
- [G8] Monitoring the health status of the subscribed individual to the Automated SOS service.
  - [R14] The system must compare the vitals of the individual with the Threshold value of health.
  - [R15] If the system detect that the health status of the individual is below the threshold value, send an alert to a nearby Ambulance with the location and details of the subscribed user.
  - [R16] To ensure the reaction time of less than 5 seconds, Push notification must be send to the ambulance driver application.
  - [C2]The services provided by the application are limited to the citizens of Milan.
  - [A6] We assume that a certain number of hospitals are registered to with TrackMe which provide Ambulance services.
  - [A7] We assume there are no network issues between the time we sent a message and the drivers receive it.

- [C3] This application is connected only with the major hospitals in Milan for Ambulance services
- [G9] Develops an interface to organize a race for the subscribed 3rd party to the Track4Run service.
  - [R17] An organizer get to add an event only after he/she/they register for Track4Run service in the application.
  - [R18] Registration involves, providing his/her email, phone number for verification and a unique user name.
  - [R19] An event can be added with the following details like event name, time and location of the event.
  - [A15] The payment details given are genuine and transactions are processed positively.
  - [A8] Organizers can keep a maximum limit of the participants who can join the event.
- [G10] Give access to the details of the race once an individual participate in it.
  - [R20] Participant has to register for the event.
  - [R21] While registering, his/her details like name, age and wearable device to be used are requested.
  - [A10] All athletes have their own wearable device.
- [G11] Live visualization of the race .
  - [R22] A spectator upon registration can view the live event.
  - [R23] He/She also gets to choose subscribe for upcoming events.
  - [A10] All athletes have their own wearable device.

## 3.3 UML Modeling

### 3.3.1 Use Case Diagrams

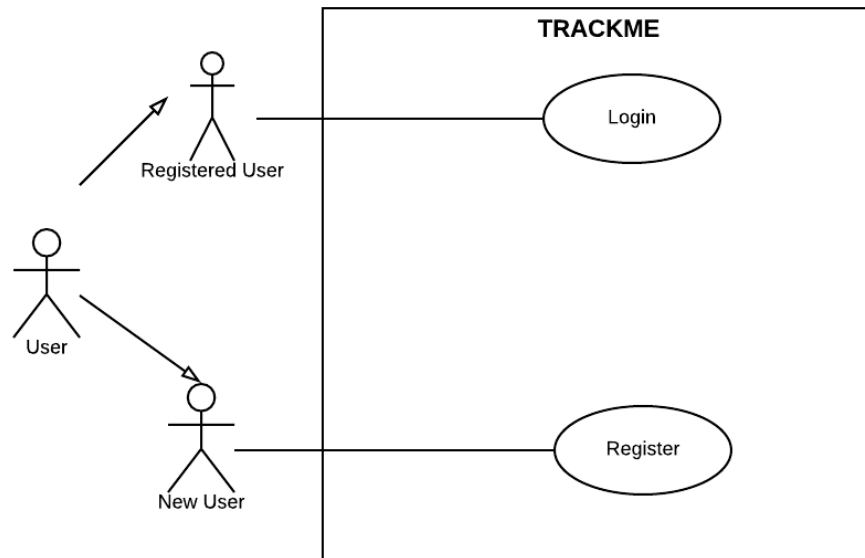


Figure 3.13: Use case for TrackMe application

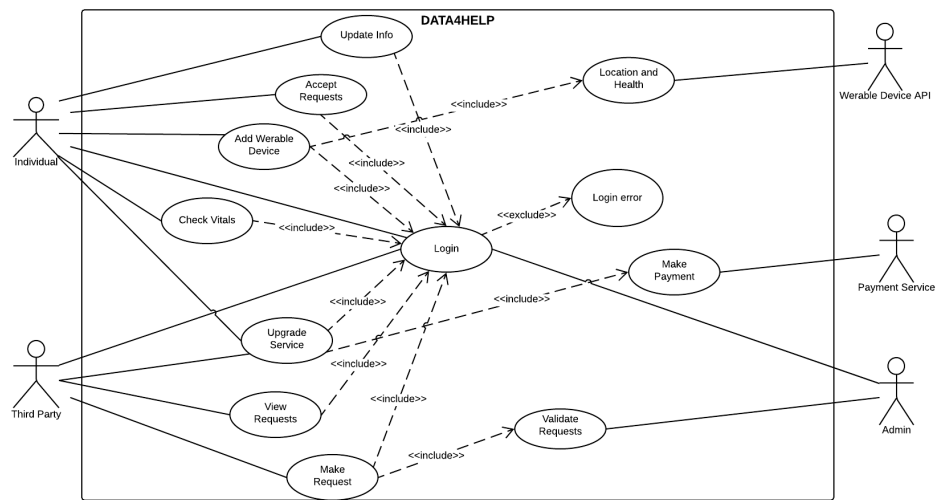


Figure 3.14: Use case for Data4help

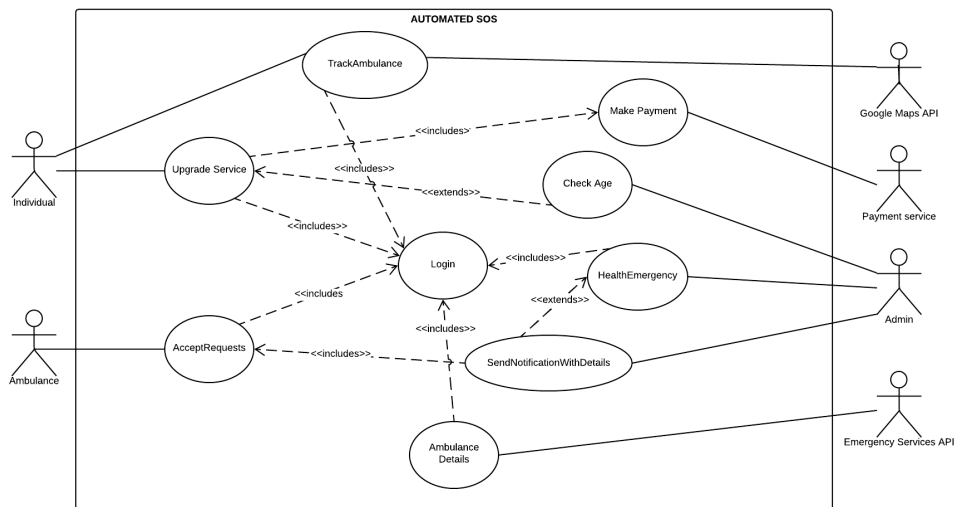


Figure 3.15: Use case for Automated SOS

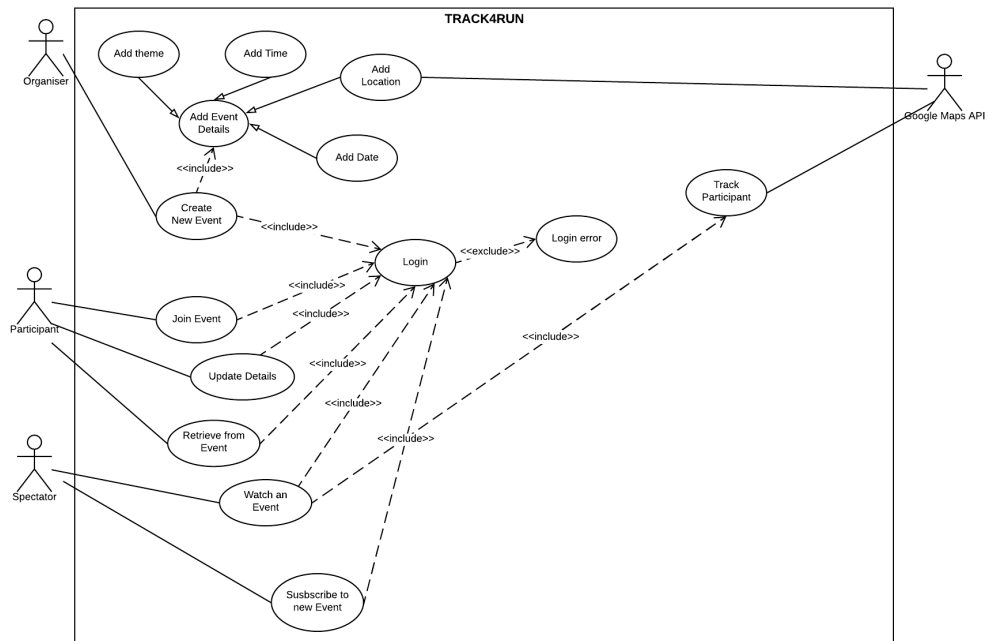


Figure 3.16: Use case for track4run

Name	Login
Actors	Registered Users(Individual or Third Party)
Entry condition	<ol style="list-style-type: none"> <li>1. The user has successfully logged into the system</li> <li>2. The user is not logged into the system yet</li> </ol>
Event Flow	<ol style="list-style-type: none"> <li>1. The user opens the welcome page of the application and forwards to sign in</li> <li>2. He/She provides the valid credentials for login, say user name and password.</li> <li>3. User gets to authenticate into the application incase his credentials are valid.</li> <li>4. His/Her dashboard shows the services to which he/she is registered with.</li> </ol>
Exit Conditions	The user successfully redirects to the dashboard.
Exception	The credentials provided by the user are invalid, in this case a login error message is shown and allows the user to input the username and password again.

Table 3.1: Use case for Login.

Name	Add wearable Device
Actors	Individual
Entry condition	<ol style="list-style-type: none"> <li>1. The individual has successfully logged into the system.</li> <li>2. The individual has a wearable device supported by trackme</li> </ol>
Event Flow	<ol style="list-style-type: none"> <li>1. Selects the add new wearable device button.</li> <li>2. Select the compatible device name from the given options.</li> <li>3. Connection successful.</li> </ol>
Exit Conditions	The individual successfully connects to the device and navigates back to dashboard.
Exception	All devices are not compatible with TrackMe application. The user selects there wearable device among the list of compatible device from Dropdown. An error message is shown in case of connection problems.

Table 3.2: Use case for adding a new wearable device.



Name	Accept or Reject Requests
Actors	Individual
Entry condition	No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. The individual can see those requests made by a third party for specific individual health vitals or location.</li> <li>2. He/She gets to accept or reject the requests.</li> <li>3. The status of the request on the individual's dashboard will be updated with the corresponding response</li> </ol>
Exit Conditions	The individual successfully redirects to the dashboard.

Table 3.3: Use case for Accept or reject requests.

Name	Update Info
Actors	User
Entry condition	No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. Upon clicking on the update Info button, user gets to change the information given earlier.</li> <li>2. Once he makes the corresponding changes, user can save it or cancel it.</li> <li>3. Upon saving changes will be updated.</li> </ol>
Exit Conditions	The user successfully redirects to the dashboard.
Exceptions	Upon providing invalid details, the user will be requested to enter the details again. Upon canceling the changes made would be lost.

Table 3.4: Use case for Update Info requests.

Name	Make Requests
Actors	Third Party
Entry condition	<ol style="list-style-type: none"> <li>1. The third party is successfully logged into the system as a third party.</li> <li>2. The third party needs health and location data of individuals or a group of individuals.</li> </ol>
Event Flow	<ol style="list-style-type: none"> <li>1. The third party clicks the Make Requests button.</li> <li>2. He/She/They are requested to select the type of request- individual or group of individuals.</li> <li>3. Enter the valid reason for the request</li> <li>4. Click the submit button.</li> <li>5. The request is then validated by Trackme or individual according to the category.</li> </ol>
Exit Conditions	The user successfully redirects to the dashboard once the requests is made.

Table 3.5: Use case for Making requests.

Name	Check Status
Actors	Third Party
Entry condition	1. No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. The Third Party clicks the Check status button.</li> <li>2. The list of requests made by him/her is available with the status(approved or still waiting to be approved) of the requests</li> </ol>
Exit Conditions	The Third Party successfully navigates back.
Exceptions	Incase the Third Party has not made any requests a message stating “ No requests found” will be shown.

Table 3.6: Use case for view requests.

Name	Upgrade Service
Actors	Third Party or Individual
Entry condition	The user needs to access a new service
Event Flow	<ol style="list-style-type: none"> <li>1. The user clicks the service they want to upgrade.</li> <li>2. Choose the duration for the service.</li> <li>3. Directs to the payment page</li> <li>4. Makes the payment and completes the transaction.</li> </ol>
Exit Conditions	The user successfully redirects to the dashboard.
Exceptions	Transaction errors can happen, in that case he/she is asked to provide the credentials again and try.

Table 3.7: Use case for Upgrade Service.

Name	Validate Requests
Actors	Admin
Assumptions	There are some requests made by the third party service.
Entry condition	No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. Checks for the latests requests and categorize them.</li> <li>2. Incase of requests for individuals the request is forwarded to the individual.</li> <li>3. Incase of request for group of data, validity of reasons provided are checked. If the group of people requested for is greater than 1000, th request is accepted.</li> <li>4. Status of requests are updated.</li> </ol>
Exit Conditions	None
Exceptions	If the requests are fake, then the requests are rejected.

Table 3.8: Use case for Validate Requests.

Name	Send Notification to Ambulance Drivers
Actors	Admin
Assumptions	The emergency health condition monitored is accurate as they are calculated on the basis of 3 readings.
Entry condition	No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. Check for emergency condition of any individual.</li> <li>2. Send push notification to all nearby ambulance drivers as soon as the emergency is encountered. The push notification guarantees a reaction time less than 5 seconds.</li> <li>3. The push notification contains all the details of the individual.</li> <li>4. The individual can also track the ambulance in their application's map.</li> </ol>
Exit Conditions	None
Exceptions	None.

Table 3.9: Use case for Send Push Notification.

Name	Accept individual request and navigate to their location
Actors	Ambulance Driver
Assumptions	The notification will be received by all nearby ambulance drivers and the first driver to accept the request will navigate to the individual's location.
Entry condition	Receive a push notification and accept it.
Event Flow	<ol style="list-style-type: none"> <li>1. The driver receives a push notification in the application for ambulance drivers.</li> <li>2. They accept the request.</li> <li>3. The first driver to accept the request navigates to individual's location with any emergency products if required.</li> </ol>
Exit Conditions	Reject request
Exceptions	None.

Table 3.10: Use case for Ambulance drivers accepting request.

Name	Create a new Event
Actors	Organizer
Assumptions	He/She needs to host a new event(race)
Entry condition	Must provide all event details and pay for organizing
Event Flow	<ol style="list-style-type: none"> <li>1. Click on the create event Button</li> <li>2. Enter the Details(Name, Location, Time, Reason)</li> <li>3. Submit the details</li> </ol>
Exit Conditions	Event is successfully created and navigated to the dashboard
Exceptions	Incase the user gives an invalid location, he/she will be asked to re try again. Incase the location and time is already booked for another event a warning would be shown.

Table 3.11: Use case for New Event/Race.

Name	Join Event
Actors	Participant
Assumptions	He/She needs to participate in an event
Entry condition	No entry conditions
Event Flow	<ol style="list-style-type: none"> <li>1. He/She registers for an event</li> <li>2. Adds the required details which includes the wearable device,name, place etc.</li> <li>3. Confirms the availability</li> <li>4. Submit the details</li> </ol>
Exit Conditions	The Athlete successfully joins the event.
Exceptions	An error message would be generated incase the maximum limit for the participants have reached.

Table 3.12: Use case for Joining an event/Race.

Name	Watch an Event
Actors	Spectator
Entry condition	Should be logged in as a spectator
Event Flow	<ol style="list-style-type: none"> <li>1. The user chooses the race which he wants to watch now.</li> <li>2. Upon clicking a view, he/she can watch the race live.</li> <li>3. He/She can navigate out whenever the user wants</li> </ol>
Exit Conditions	Navigate back to the dashboard
Exceptions	Incise the user is not subscribed for any event, he/she won't be able to see any. Incase the user has not made the payment, he/she won't be able to watch the event.

Table 3.13: Use case for watching an event/Race.

### 3.3.2 Sequence Diagram

1. Accept/Reject Third party requests:

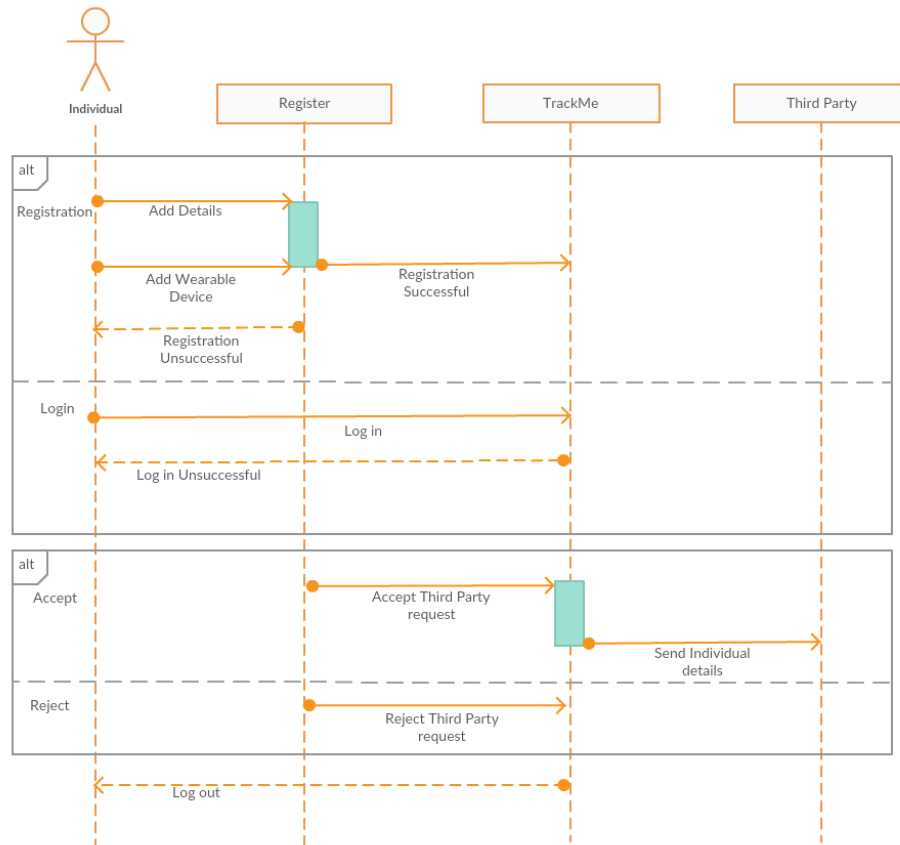


Figure 3.17: Accept/Reject Third party requests.



2. Third Party makes new request:

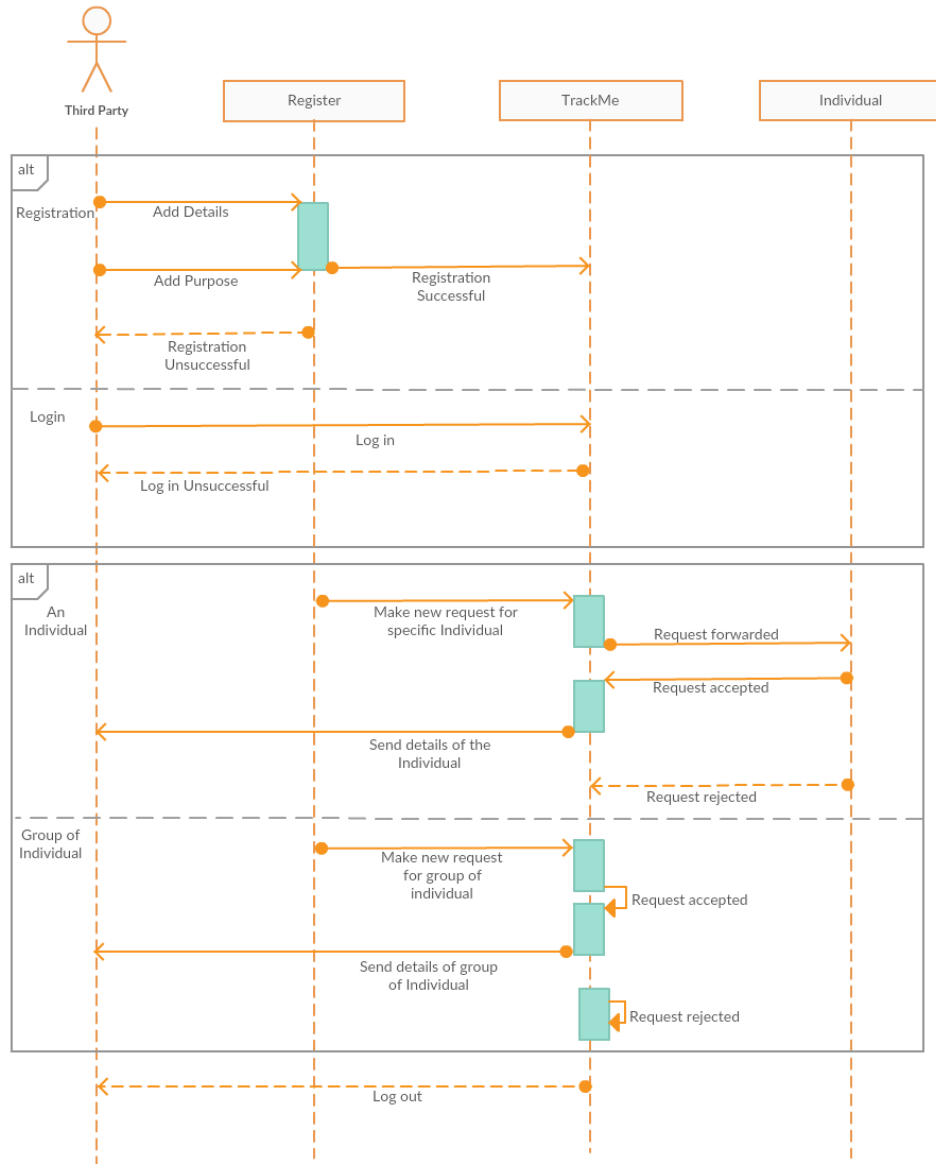


Figure 3.18: Third Party makes new request.

3. TrackMe validates request:

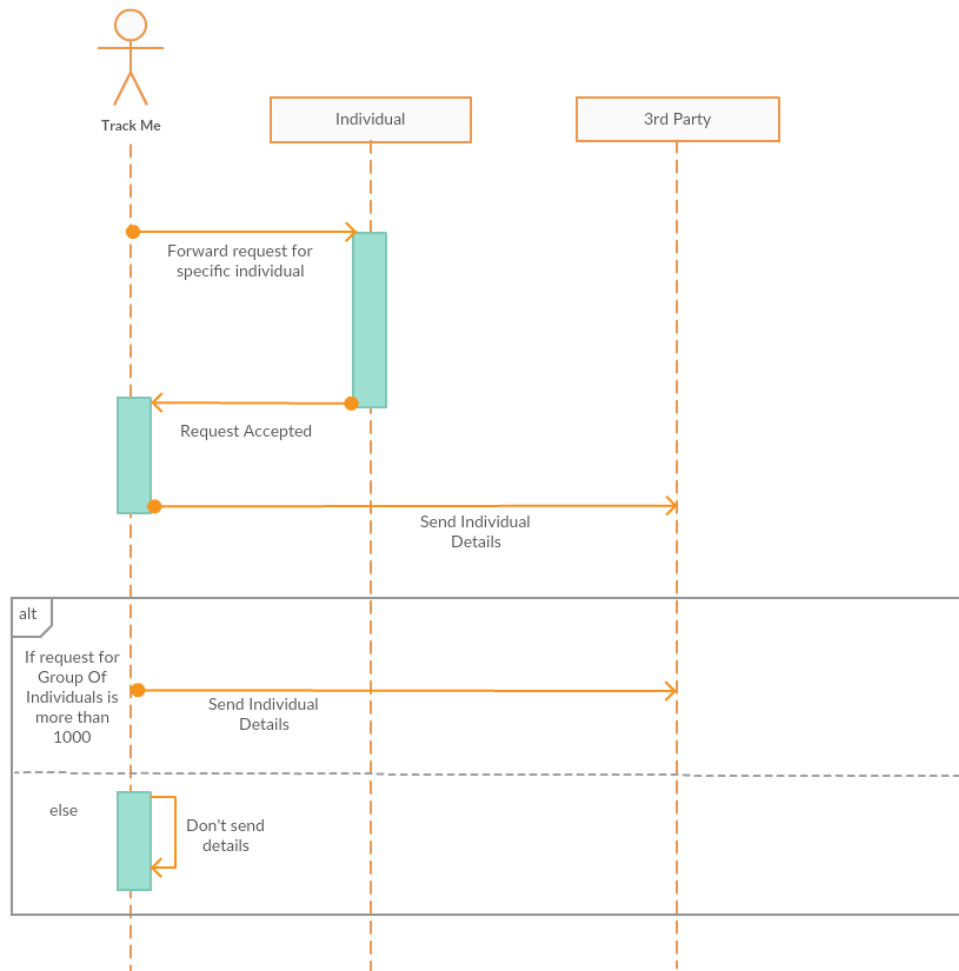


Figure 3.19: TrackMe validates request.

4. Check vitals and send push notification:

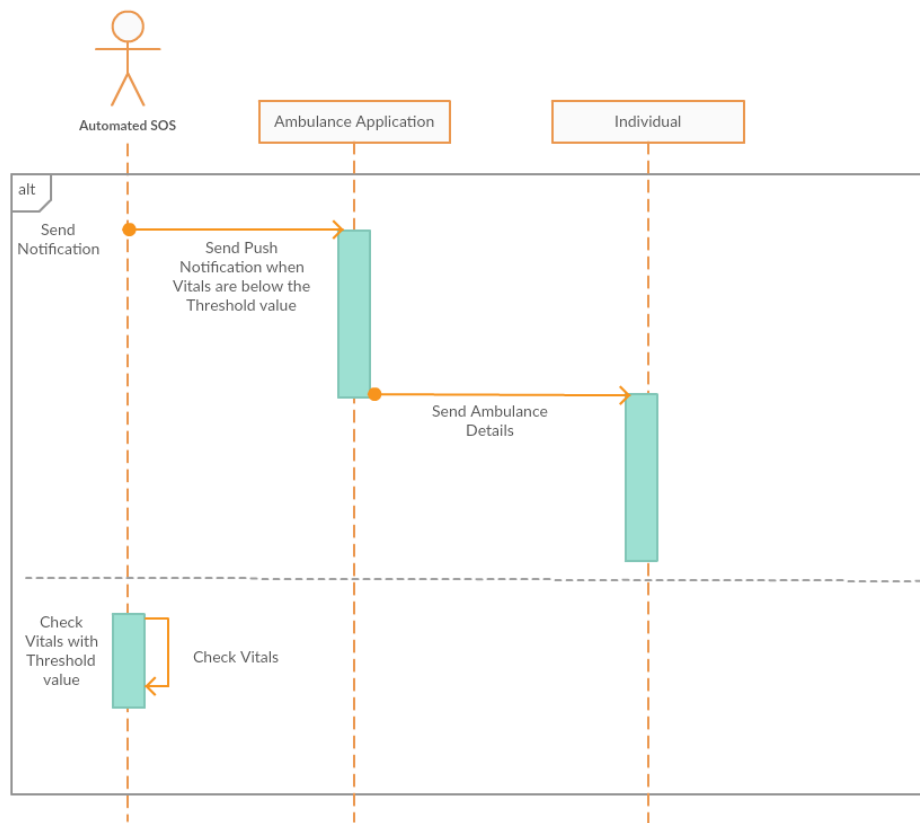


Figure 3.20: Check vitals and send push notification.

5. Ambulance Drivers accepts the request of push notifications and navigate to their location:

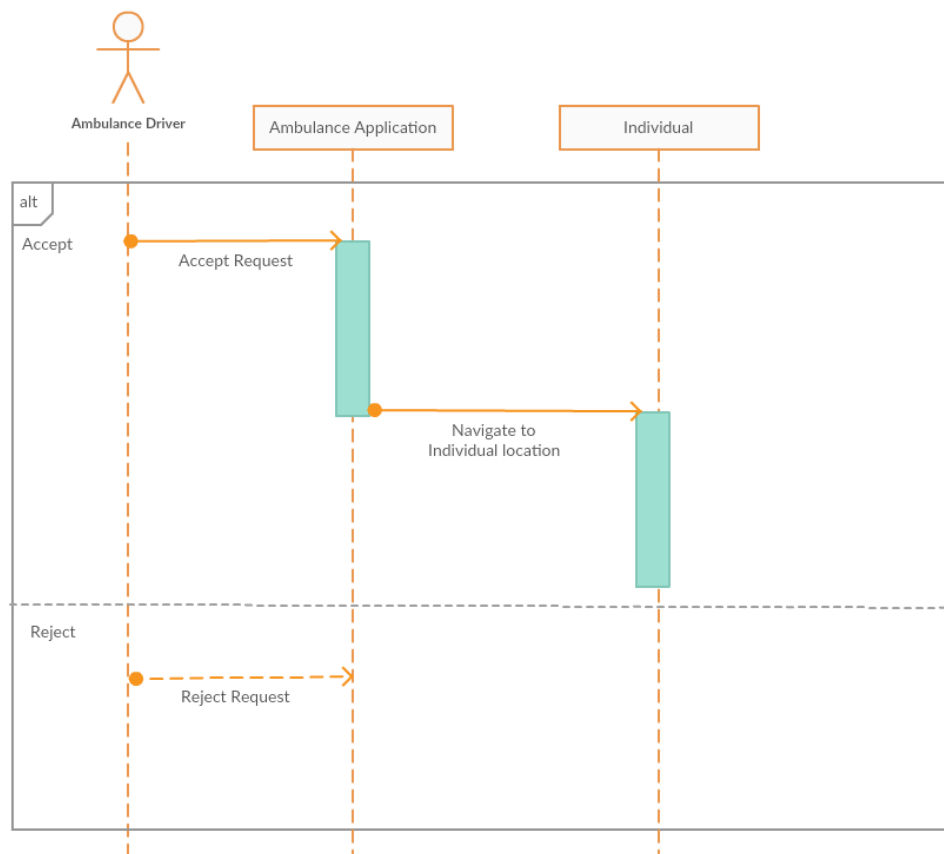


Figure 3.21: Accept request of push notifications and Navigate to their location.

6. Organizer adds race and defines path:

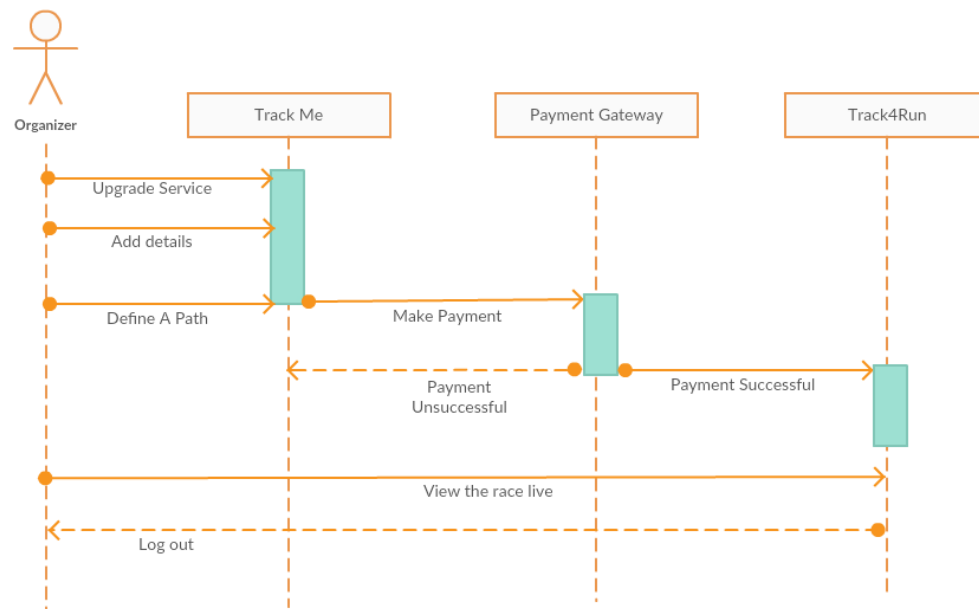


Figure 3.22: Organizer adds race and defines path.

7. Athletes participate and Spectators view the race:

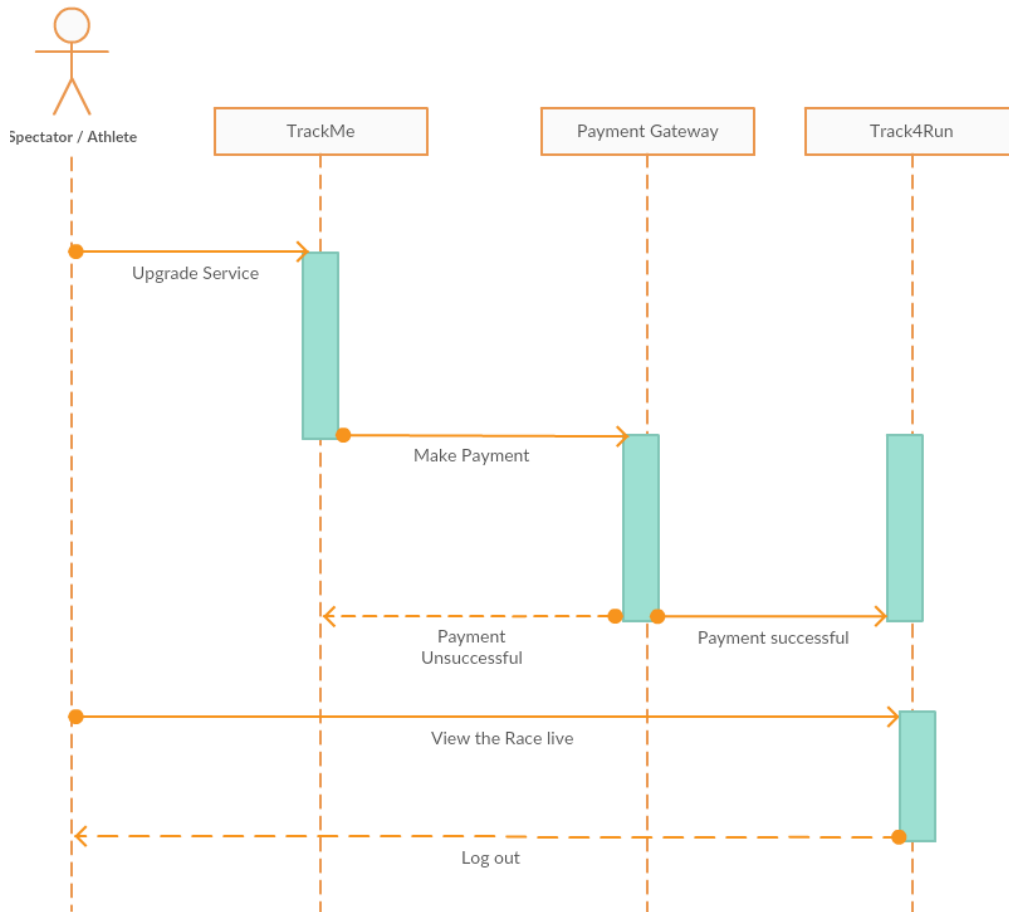


Figure 3.23: Athletes participate and Spectators view the race.

### 3.4 Performance Requirements

1. The system should at least support 1000 connected users at a time.
2. There is no limit in the number of registered users
3. The push notifications would take around a maximum of 5 seconds.

## **3.5 Design Constraints**

### **3.5.1 Standards Compliance**

Its users responsibility to ensure that the use of the system complies with the local laws and policies. The system must ask the user for permission to acquire, store and process the personal data and web cookies.

### **3.5.2 Hardware Limitations**

Only a certain set of wearable devices are supported by our application.

## **3.6 Software System Attributes**

### **3.6.1 Reliability**

The system is currently not designed to run in Web Interfaces. The reliability of the system depends on the server.

### **3.6.2 Availability**

The service shall be continuously available to the user.

### **3.6.3 Security**

We ensure the data from the individuals are safe and are only accessible by genuine third party users. The requests won't be accepted unless and until a thorough check of the reasons provided by the 3rd party users or an acceptance from the individual is received. We make sure that third party won't be able to access any sort of data upon continuous rejection of data requests by individuals. Users banking details are secured through payment via paypal gateway, hence providing a secure transaction. We also use push messages and thereby enabling the feature of sending encrypted messages rather than the normal notifications which can be accessed with the READ\_SMS permissions.

### **3.6.4 Maintainability**

The code should be well documented for the future reference for developers. The software development should follow an Object Oriented Model-View-Controller pattern. We will also be automating the testing phase.

### **3.6.5 Portability**

The software will run on any platform that supports JVM. Currently the application is only available for Mobile applications. The mobile application will be supported by the latest versions of Android and iOS.



# Chapter 4

## Alloy Model

### 4.0.1 Description

To execute the main objective, the alloy model depicts the following logics:

- There are two users individual and third party.
- Individual has one wearable device that gives Location and Health vitals of that Individual.
- Third Party can make Group Request and Individual Request.
- When the health status of an Individual is below the threshold value, the critical condition is True.
- The ambulance gets the details of the individual whose critical condition is true.
- The details of the individual is connected to the ambulance that is near to the individual location.
- An organizer is a Third Party who organizes the Event.
- An Athlete is an Individual who joins the Event.
- A spectator can be both Individual and Third party who watches the event.
- An event has date, time, and location.

## 4.0.2 Alloy code

```
sig Individual {  
  age : one Int,  
  device: wearableDevice,  
  ath: set Athlete,  
  sp: set Spectator,  
  critical: one Critical  
}  
{age > 0  
critical = True}  
  
sig Thirdparty{  
  groupReq: one GroupRequests,  
  indReq: one IndRequests,  
  org: set Organizer,  
  sp: set Spectator  
}  
  
sig wearableDevice{  
  location: one Location,  
  health: one Health  
}  
  
sig Location{  
  latitude: Int,  
  longitude: Int  
}  
  
sig Health{  
  heartbeat: Int,  
  steps: Int,  
  otherVitals: Int  
}
```

```

sig GroupRequests{
  noofRequest: one Int,
  noofInd: one Int
}
{noofRequest>0
noofInd>0}

```

```

sig IndRequests{
  noofRequest: one Int
}
{noofRequest>0}

```

```

one sig Athlete{
  event : one Event
}

```

```

one sig Organizer{
  event : one Event
}

```

```

one sig Spectator{
  event : one Event
}

```

```

sig Event{
  id : one Int,
  date: one Date,
  time: one Time,
  path: one Location
}
{id=1
#time=1}

```

```

sig Ambulance{
  ambno: one String,
  ambloc: one State,
  indloc: one Location,
  indhealth: one Health
}
{ambloc = near}

abstract sig State{}
one sig near extends State{}
one sig far extends State{}

sig Time{}
sig Date{}

abstract sig Critical{}
one sig True extends Critical{}
one sig False extends Critical{}

--Every Individual has one wearable Device
fact sameValuesperDevice{
  no disjoint w,w':wearableDevice | (w.location=w'.location and w.health=w'.health)
}

--All requests are linked to thirdparty
fact RequestsLinkedToThirdParty{
  all t:Thirdparty, g:GroupRequests, ir: IndRequests | (t.groupReq=g and t.indReq=ir)
}

--All Organizer are Third Party
fact OrganizerThirdparty{
  all t: Thirdparty, o: Organizer | t.org=o
}

```

```

--All Ahlete are Individual
fact AthleteIndividual{
all i: Individual, a: Athlete | i.ath=a
}

--All Spectator are either Individual or Third party
fact SpectatorIndividualORT hirdParty{
some i: Individual, t: Thirdparty, s: Spectator | (i.sp = s or t.sp =s)
}

--Organizers, Athlete and Spectator are related to event at a time.
fact sameEvent{
no disjoint e,e': Event, a: Athlete, o: Organizer, s:Spectator | (a.event=e and o.event=e and s.event=e) and
(a.event=e' and o.event=e' and s.event=e')
}

--Every Location and health is linked to wearable device
fact ValuesperDevice {
all w:wearableDevice, l:Location, h:Health | (w.location=l and w.health=h)
}

--Ambulance connected to Individual's health and location with critical condition is true
fact AmbulanceGettingDetails{
one i: Individual, a: Ambulance | ((i.critical=True and a.ambloc=near) implies ( a.indhealth = i.device.health and a .indloc = i.device.location))
}

--Every location and health maps to same device
assert DetailsPerWearableDevice {
all i: Individual, d: i.device, w:wearableDevice | d = w
}

```

```

--An oragnizer defines the path of the event which the athlete joins and is watched by spectator
assert OneEventPerAthleteAndOrganizer{
all e:Event, a: Athlete, o: Organizer, s:Spectator | (a.event=e and o.event=e and s.event=e)
}

pred show{
(one i: Individual | #i.device=1) and
(one t:Thirdparty | #t.groupReq=1) and
(one a:Ambulance | a.ambno="M18754")
}

run show

check OneEventPerAthleteAndOrganizer for 1

check DetailsPerWearableDevice for 1

```

### 4.0.3 Result

#### Executing "Check DetailsPerWearableDevice for 1"

Solver=sat4j Bitwidth=4 MaxSeq=1 SkolemDepth=1 Symmetry=20  
1210 vars. 198 primary vars. 3269 clauses. 31ms.  
No counterexample found. Assertion may be valid. 0ms.

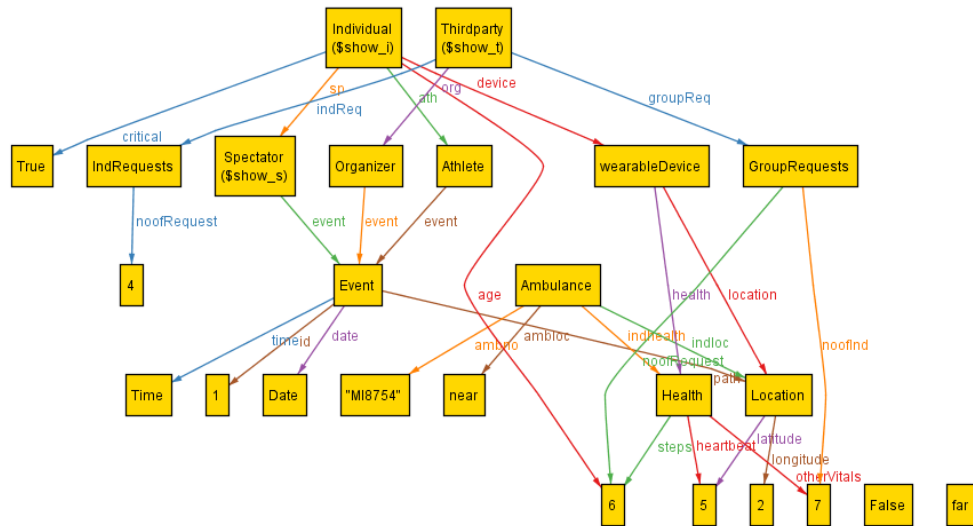
#### Executing "Check OneEventPerAthleteAndOrganizer for 1"

Solver=sat4j Bitwidth=4 MaxSeq=1 SkolemDepth=1 Symmetry=20  
1219 vars. 199 primary vars. 3285 clauses. 16ms.  
No counterexample found. Assertion may be valid. 16ms.

#### Executing "Run show"

Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20  
6043 vars. 646 primary vars. 14426 clauses. 62ms.  
Instance found. Predicate is consistent. 172ms.

#### 4.0.4 Alloy Model





# Chapter 5

## Effort

Hours of Work

### 5.0.1 Haritha Harikumar

Date	Task	Hours
15/10/2018	Understanding Project Specifications	1
16/10/2018	Brainstorming Problem(Group Work)	0.5
18/10/18	Identifying goals and set deadlines(Group Work)	2.5
20/10/2018	Goals, Assumptions and Purpose	2
21/10/2018	Product Functions	2
18/10/18	Identifying goals and set deadlines(Group Work)	2.5
25/20/2018	All Diagrams	2
29/10/2018	Use Case Diagram	2
31/10/2018	Functional Requirement	3
08/11/18	Final Draft of RASD(Group Work)	2
09/11/2018	State Diagram	3
07/11/2018	Alloy	2
11/11/2018	Alloy(Group Work)	3
	<b>Total</b>	<b>36</b>

### 5.0.2 Saloni Kyal

Date	Task	Hours
16/10/18	Brainstorming Problem(Group Work)	0.5
18/10/18	Identifying goals and set deadlines(Group Work)	2.5
19/10/18	Scope, Product Functions and Dependencies	2
20/10/18	Software Interfaces, Definitions and Acronyms	1
22/10/18	Discuss rest plan(Group Work)	3
30/10/18	All Diagrams Discussion(Group Work)	2
31/10/18	All Diagrams Discussion(Group Work)	2
02/11/18	Initiate Alloy(Group Work)	1
03/11/18	Alloy(Group Work)	3
05/11/18	User Interface: Mock-ups	4
06/11/18	User Interface: Mock-ups	2
06/11/18	Class Diagram	1
07/11/18	Use Case- Automated SOS and tables	2
08/11/18	Final Draft of RASD(Group Work)	2
09/11/18	Sequence Diagram- Draft	1
10/11/18	Alloy(Group Work)	1
11/11/18	Alloy(Group Work)	3
11/11/18	Alloy	3
	<b>Total</b>	<b>36</b>

### 5.0.3 Mohini Gupta

Date	Task	Hours
16/10/18	Brainstorming Problem(Group Work)	0.5
18/10/18	Identifying goals and set deadlines(Group Work)	2.5
19/10/18	Goals, Product Functions and Constraints	2
21/10/18	Software Interfaces	1
22/10/18	Discuss rest plan(Group Work)	3
27/10/18	Functional Requirements	1
30/10/18	All Diagrams Discussion(Group Work)	2
31/10/18	All Diagrams Discussion(Group Work)	2
02/11/18	Initiate Alloy(Group Work)	1
03/11/18	Alloy(Group Work)	3
05/11/18	User Interface: Mock-ups	3
06/11/18	User Interface: Mock-ups	2
06/11/18	Class Diagram	1
07/11/18	Communication Interfaces	1
08/11/18	Final Draft of RASD(Group Work)	2
09/11/18	Sequence Diagrams	3
10/11/18	Alloy(Group Work)	1
11/11/18	Alloy(Group Work)	3
11/11/18	Alloy	2
	<b>Total</b>	<b>36</b>

# Chapter 6

## References

- Specification Document: "Mandatory Project Assignment AY 2018-2019.pdf"
- Alloy model example: "<http://alloytools.org/tutorials/online/>"
- Requirement Engineering Slides I, II
- Alloy Slides