03-134142-065 MATTIULLAH SHAHZAD

03-134142-073 NABEEL AHMAD KHAN

Auto Rescue Assistance

In partial fulfilment of the requirements for the degree of

Bachelor of Science in Computer Sciences

Supervisor: Tahir Iqbal

Department of Computer Sciences

Bahria University, Lahore Campus

June 2018

C e r t i f i c a t e

We accept the work contained in the report titled

"AUTO RESCUE ASSISTANCE"

written by

MATTIULLAH SHAHZAD

NABEEL AHMAD KHAN

as a confirmation to the required standard for the partial fulfilment of the degree of Bachelor of Science in Computer Sciences.

Approved by:

Supervisor: Mr. Tahir Iqbal

_____ (Signature)

June 4*th*, 2018

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-134142-065	MATTIULLAH SHAHZAD	
03-134142-073	NABEEL AHMAD KHAN	

Date : _____

Specially dedicated to

My beloved grandmother, mother and father

(Mattiullah Shahzad)

My beloved grandmother, mother and father

(Nabeel Ahmad Khan)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our project supervisor, Mr Tahir Iqbal for his invaluable advice, guidance and his enormous patience throughout the development of the project.

In addition, we would also like to express our gratitude to our loving parent and friends who had helped and given us encouragement.

Mattiullah Shahzad
Nabeel Ahmad Khan

AUTO RESCUE ASSISTANCE

ABSTRACT

Speed and collisions are the main leading causes of death in vehicle accident. The main objective of Auto Rescue Assistance is to reduce the response time of emergency service. Auto Rescue Assistance is an Android based application which will give the facility to the user to notify the nearest ambulance in the case of accident for rescuing the life of the user before it gets too late. Auto Rescue Assistance will generate a request depending on the condition, it might be manual request, or the accident automatically detected by different parameters i.e. speed threshold calculated using GPS or the pressure detected by the external sensor, FSR-09375, when the request occurs, the application will search for the nearest driver and the request will send to the selected driver and driver will be provided with the incident location and information of the user. Once this information is received by the driver of ambulance, he can approach the victim and the first aid can be provided on time. Auto Rescue Assistance will notify pre-defined emergency contact about the accident with current location of the victim, so they can approach the victim for the corrective actions. Reduction in emergency response time is attained with implementation of an automatic request mechanism and tracking of nearest active ambulance based on the victim's current location which will increase the survival rate of the victim.

TABLE OF CONTENTS

Table of Contents

Table of Figures:

CHAPTER 1

INTRODUCTION

1.1. Background:

Traffic accidents are considered a significant public issue all over the world. Injuries and deaths occur in larger numbers because of road traffic accidents that highlights the global concern for road safety. Collisions caused in traffic are the second major cause of causalities for people between the age group of 5-29 and third major cause of causalities for people between age group of 30-44. In most of the cases, the absence of a first aid facility or late intimation to rescue services becomes the reason of a person's death, delaying the help reached to the person suffered due to an accident.

There is an increased demand for automobiles these days which often causes problems in traffic control and hence leading to road accidents. This also increases the response time for rescue services to attending the victim and hence increasing the death toll. Auto Rescue Assistance provides an optimal solution for this problem through automatic accident detection and tracking nearest ambulance. This solution is most effective in reducing the number of causalities in daily road accidents due to problematic scenarios and worse traffic condition by reducing the time between when an accident occurs and when first emergency responders are dispatched to the scene of the accident

1.2. Problem Statement:

Traffic accidents are a major cause of causalities happening around the world daily. Due to increased population, there is a hike in demand of vehicles for day to day travelling which causes problems in controlling the flow of traffic and often lead to major road accidents. The most noticeable reason for a person's death during accidents is absence of the first aid facility

and in many situations the family members or the ambulance is not` informed in time. This results in delaying the help reached to the person suffered due to an accident. Thus, in case of incidents, response time of ambulance services is very important for the timely delivery of emergency medical services to accident victims and is effects greatly to reduce the number of causalities. Moreover, each minute is passed while an injured crash victim does not receive emergency medical care can make a large difference in their survival rate. Notify the family members on time is also important so they can approach the victim and corrective actions can be taken. The information about the accident on time could save many lives so the emergency services could get the accident information on time and reach the victim in time for the first aid services. Survival rate of accident victim is inversely proportional to the delay in medical services to the victim. As the delay goes down, the survival rate will be high. Reduction of delay in the medical services could save many lives.

1.3. Aims and Objectives:

Designing an automatic accident detection and rescue application which will

- Detect the accident according to given parameters, it will then notify the accident location with the victim's information to the available nearby ambulance drivers.
- Send a user's location as a quick message to emergency contacts to intimate the victims.
- Show the nearby emergency spots (i.e. Hospitals and Police Station) on integrated map.

1.4. Scope of Project:

The Auto Rescue Assistance is an Android based application which will give the facility to the user to notify the ambulance in the case of accident for rescuing the life of the user before it gets too late. Auto Rescue Assistance will generate a request it can be a manual or automatic once the request is generated, it will pass to the backend server, server will check the available nearby ambulance drivers according to the victim's location. Server will generate another request that will contain user location and information. Server will send this request via push notification to the selected driver. Once this request is received by the driver of ambulance, he can approach the victim.

Many lives could have been saved if the emergency services could get the accident information in time. To minimize the false requests, external sensors is integrated with the application. Vehicles have push button attach on them, so in case of emergency user can press the push button to request the ambulance.

Furthermore, Auto Rescue Assistance is also a GPS, Global Positioning System based application that provides a functionality of getting nearby emergency places i.e. Hospitals and Police Stations. User would be able to view these places on Google map that is integrated with the application.

CHAPTER 2

Software Requirement Specification

2.1 Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.1.1 Product Perspective

This system will consist of mobile application and hardware sensor. The mobile application will be used to detect the accident occurrence along with the hardware sensor to rescue the people during accident situations. The complete product will provide the detection of accident happening and corrective measures at that time to safe the victim's life by passing the victims information to the nearby ambulance driver because many lives could have been saved if the rescue teams get the accident information on time.

The mobile application need to communicate with the GPS, Global Positioning System to get the location of user can be track and if accident occurs then the application will pass the user location to the server and server will find the rescue driver with the nearest distance or minimum distance based on the location of user so the ambulance driver can approach the victim. The mobile application will automatically send the message with the current location of the victim to the SOS, Save Our Souls contacts, it's important to get them notify about the mishap. Using "Auto Rescue Assistance" you can also find the nearby hospitals and police stations based on your current location.

Since this is a data-centric product it will need somewhere to store the data. For that, a database will be used. Both the mobile application and web server will communicate with the database, however in slightly different ways. The mobile application will only use the database to store the user's information while the web server will use the database to get and store the location of nearby ambulance drivers to the victim location. All the database communication will go over the Internet.

2.1.2 **Product Functions**

With the mobile application, the users will be able to search for nearby hospitals and police stations and the result will be based on the user current location. User can see the hospitals and police stations on the map view and could get the idea about the nearest hospital and police station according to the current location on the map. Manual panic button provides the manual request to the rescue teams. User can add and delete SOS contacts and in case of accident the contacts will be notified about the mishap with the current location of user. Automatic accident detection will provide the ease to notify the SOS contacts and nearest ambulance driver automatically in case of accident occurrence. Automatic accident detection will be based on different conditions to minimize the factor of false requests. Speed breakdown threshold will be calculated through the mobile GPS, Global Positioning System and the collision can be detected via hardware sensor and the collision detected will be sent as the input to the mobile application and the speed threshold value can be calculated via GPS and the combined input if satisfies the condition then the situation will be considered as accident and corresponding request circle will start. The web server will provide the logic to send the request to the nearest ambulance based on the location received from the user.

2.1.3 **User Characteristics**

There are three types of users that interact with the system: users of the mobile application, drivers of rescue team and the admin of this application. Each of these three types of users has different use of the system so each of them has their own requirements.

The mobile application users can only use the application to request for the rescue services. The user can search for hospitals and police station near the current location. Users

can add and delete SOS contacts. They can see their current location on map and can request manually for the rescue services otherwise automatically accident could also be detected by the application when running in background.

The drivers of rescue team can accept the request of the victim and navigate to the victim location using maps which will be showing the accident location of the user. The driver will be able to raise the flag when moving towards the victim to provide the services, so the driver will show busy flag and next request to the same driver could not be applied till the completion of previous request.

The admin panel of this application will be managing the users of this application via web portal and will be able to add or delete users or they will be able to see the users (online) of application in web portal.

2.1.4 **Operating Environment**

Android device having API levels:

- Jellybean (API 17- 18)
- KitKat (API 19 - 20)
- Lollipop (API 21 – 22)
- Marshmallow (API 23)
- Nougat (API 24 – 25)
- Oreo (API 26 - 27)

2.1.5 **Constraints**

The mobile application is constrained by the system interface to the GPS navigation system within the mobile phone. Since there are multiple system and multiple GPS manufacturers, the interface will most likely not be the same for every one of them. Also, there may be a difference between what navigation features each of them provides.

The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function.

In-App permissions are required for the application to work properly like Contacts Permission, to get the contacts from phone that can be insert to SOS contacts list of Application and used when needed. GPS must be enabled to get the current location.

Bluetooth must be enabled to get the output from the pressure sensor via Arduino. Minimum SDK version for the app is 17 so the mobiles having Jelly Bean or latest version of Android will be able to use all the functions of this application.

2.1.6 **Assumptions and dependencies**

One assumption about the product is that it will always be used on mobile phones that have enough performance. If the phone does not have enough hardware resources available for the application, for example the users might have allocated them with other applications, there may be scenarios where the application does not work as intended or even at all.

The Application has given all the in-app permissions, so the app can work properly. Bluetooth, GPS and Internet connection must be enable in the device while using the application. The device running this application must have the SDK (Software Development Kit) version greater than or equal to 17 or the android version greater than or equal to Jelly Bean.

2.2. Functional Requirements:

1) **Download Mobile Application:** A user should be able to download the mobile application
   Processing The user will be able to download the application using an application store or similar service on the mobile phone. The application should be free to download.
   Output The application will be successfully downloaded in the device.

2) **Sign Up:** The user will have to provide the enough information to sign up for new account.

Processing When the user enters the required information in the fields and presses the sign-up button, his/her entered information would go on to the server via API call where it would be insert the whole information in the database residing on the server.

Output The user account will be successfully created, and the contact number and password will be used further for the log in purpose.

3) **Login:** The user would provide contact number and password in the Login dialog and press the login button to logon to the system.

   Processing When the user enters the contact number and password and presses the login button, his contact number and password would go on to the server via API call where it would be checked with the entries in the database residing on the server. If these will match, the user will be authenticated to logon to the system.

   Output The user will logon to the system after the contact number and password provided by the user is matched in the database. Immediately after the user has logged on to the system and the main activity page i.e. map view with the user's current location will be shown to the user and other features can also be access using this activity.

4) **Forget Password:** The user will be able to recover the password via email entered at the time of registration.

   Processing When the user will press the forget password button then the password from the database will be fetched and will send the password in the email id entered at the time of sign up process.

   Output The password will be successfully received to the user via email.

5) **Maintaining SOS Contacts:** The user will be able add and delete the contacts to send the message in an emergency. The user can also view the contacts added in SOS list.

   Processing When the user will press the add button it will add the contacts list to the list view to see the added contacts and user can also delete the contacts from the list.

   Output The messages will be received to all the contacts added in the SOS list with the current location of the user and message.

6) **Searching Near-by Hospital:** The user can search the nearby hospital based on the current location of the user.

Processing When the user presses the Near-By Hospital then the hospitals nearer to the current location of the user will be shown to the user on the map, so it can be approached easily using the maps.

Output The user will be able to see the near hospitals on the map view.

7) **Searching Near-by Police Station:** The user can search the nearby police station based on the current location of the user.

   Processing When the user presses the Near-By Police Station then the Police Station nearer to the current location of the user will be shown to the user on the map, so it can be approached easily using the maps.

   Output The user will be able to see the near Police Station on the map view.

8) **Panic Button:** The user can request manually for the rescue services instead of automatically detected via different approaches.

   Processing By pressing this button messages will send to the SOS contacts with the current location of the user and nearest ambulance will also get the details of the requested user to provide the medical service to the requested user.

   Output The user will be able to see the near Police Station on the map view.

9) **Automatic Accident Detection:** Accident occurrence can be automatically detected using different approaches.

   Processing The automatic accident detection can be done using different sensors like Accelerometer which is used to get the acceleration of a vehicle and hardware sensor will be used to detect the collision of vehicles and the output of sensor will be send via Audrino Bluetooth module.

   Output Accident occurrence detected via sensors.

10) **Profile Maintenance:** User can update its information.

    Processing User will be able to change or modify the information which was added or required at the time of registration. The update button will get the whole information from the fields and will call the API which will update the record in the databases.

    Output Successfully updated the information.

11) **User's Management:** Admin can manage the application users and accounts.

Processing The admin will be able to add or delete users of application. It can manage the users via web portal.

Output User's will be managed via web portal.

12) **Accident Tracking:** Admin can track the accidents in maps through web portal.

Processing A view of accidents can be seen in the map.

Output A view of accidents on the map.

2.3. Specific Requirements

This section contains all the functional requirements of the system. It gives a detailed description of the system and all its features.

2.3.1. External Interface Requirements

This section briefly illustrates the detailed description of inputs and outputs from the system. It also gives a description of the hardware, software and communication interfaces. It will also cover the prototypes of the user interface.

2.3.1.1 User interfaces

First-time user will see the landing page when he/she launches the application, Fig 1. Where the user can make a choice of signing up and logging to the system. If user is registered member, then he will proceed with Login, Fig 2. Otherwise signup button will take him to the registration page on Fig 3.

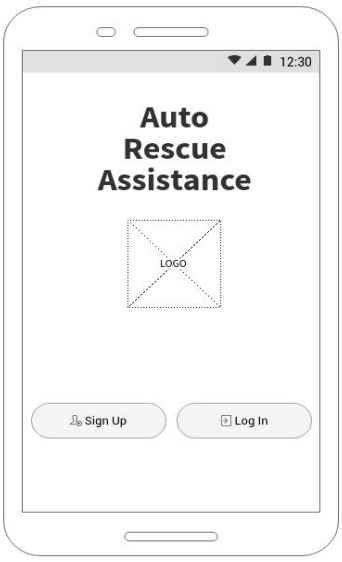


Figure 1: Landing Page

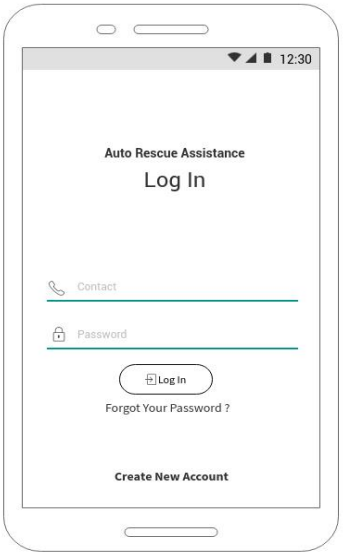


Figure 2: Login Page

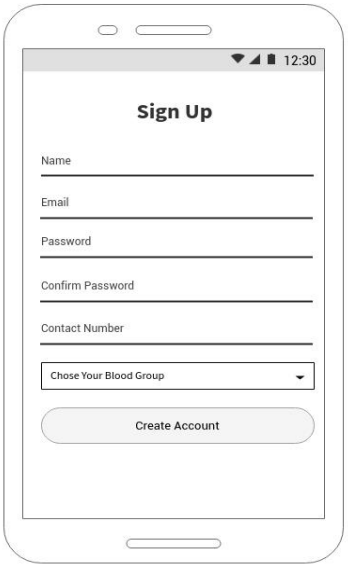


Figure 3: Signup Page

User will be redirected to the home page at Figure 4 after successfully login/signup. Home page is the main activity of the application. It has a map view which shows the current location of the user also the location of nearby ambulances. It has Assistance Rescue button; this button is also known as panic button. When the user is in an emergency he can press the button.

Figure 4: Home Page

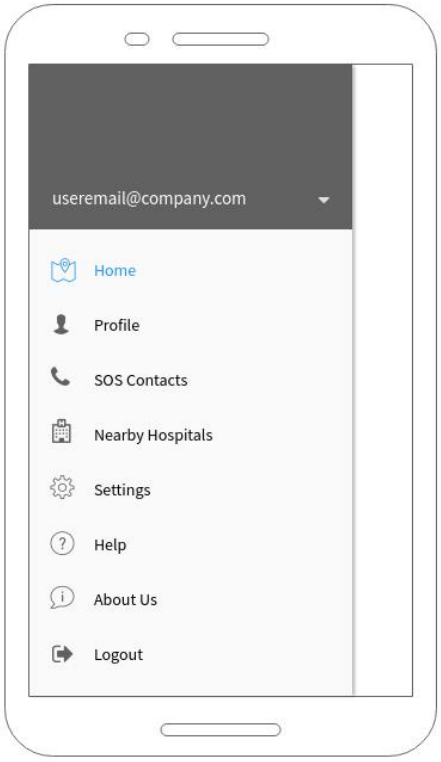


Figure 5: Drawer Navigation

Fig 5, shows the drawer navigation or the menu. It will open whenever the user taps on the three bars at the top-left of home page. Drawer navigation has all the menus that is link with the all the activities of the application. These menus are described below:

Menu	**Description**
Home	It will display the home page of the application
Profile	It will display the information of currently login user.
SOS Contacts	It has the list of emergency contacts.
Nearby Hospitals	It will show all the nearby hospitals in map.
Settings	It will allow user to set the settings of application.
Help	It will demonstrate the user how this application works.
About Us	It will show the details of the developers.
Logout	User will logout from the device.

Application also has a profile page where the currently logged in users can edit their personal information such as Name, e-mail address and phone number and user can also update their password see Fig 6.

In Fig 7 it shows the list of all the emergency contacts. Emergency contact can be added into the list by tapping add float button located at the bottom-right-corner of the screen. User can also remove the contacts by tapping trash icon.

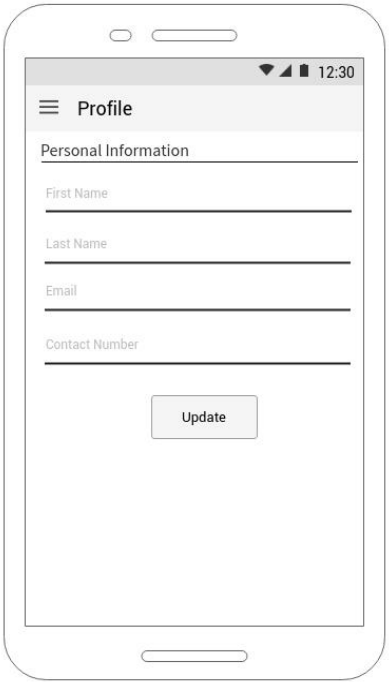


Figure 6: Profile Page

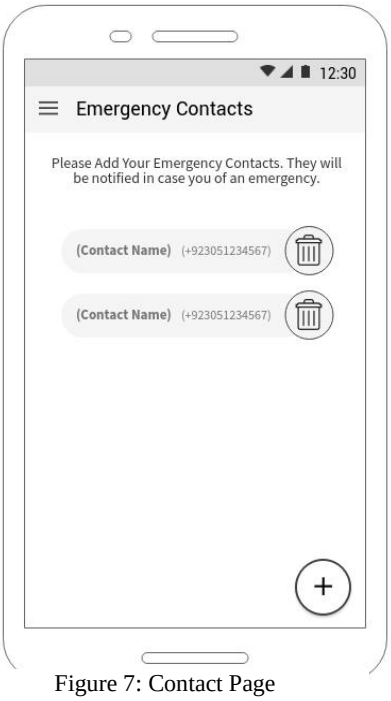


Figure 7: Contact Page

User can also view the about application page and help page at Fig 8 and Fig 9 respectively.

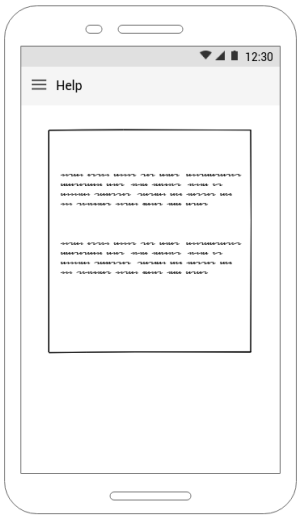


Figure 8: Help

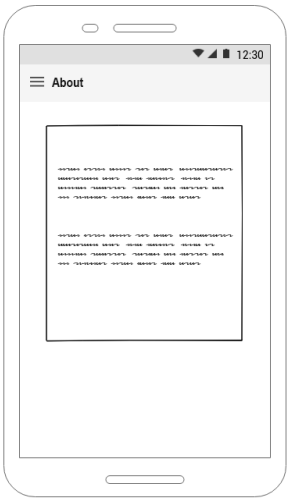


Figure 9: About

Web portal, Fig 10 is an admin panel of ARA. The administrators interact with the system through a web-portal. Admin has full access to users, settings, and applications. Admin can view the information of users, can update or delete them through web portal. Web portal has all the information of the user/rescuer. It also has report of all the past accidents and active accidents which can be seen via google map.

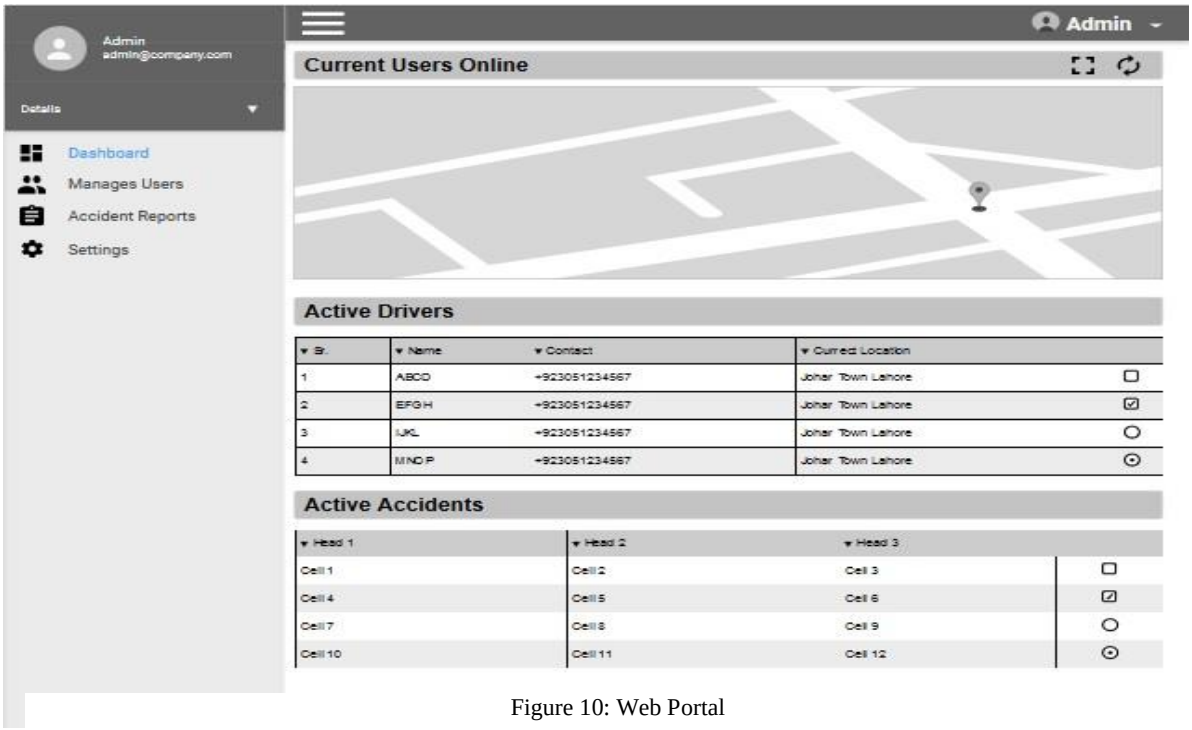


Figure 10: Web Portal

2.3.2. Hardware Interfaces

The minimum hardware requirements of ARA are different android built-in sensors i.e. GPS for getting approximate speed of vehicle and External hardware sensor i.e. Arduino UNO, Bluetooth module HC05, Button Sensors to detect the pressure from environment and then output will be sent to the Bluetooth module and it will send the output to the android device.

2.3.3. Software Interfaces

ARA can relate to SQLite to maintain emergency contacts list and PostgreSQL to implement backend logic as well as user and driver's data. ARA requires minimum Android API level 17.

2.3.4. Communication Interfaces

ARA requires an internet connection to connect with the Rest API. The application requires HTTP to communicate with server.
The application communicates with the Bluetooth module (HC05) to get the output from sensor.

2.4 Other Non-Functional Requirements

2.4.1 Performance Requirements:

Response Time: The application should response faster to every request received from application user.
Maintainability: The application should be easy to extend. The code should be written in a way that it favours implementation of new functionalities.
Testability: Test environments should be built for the application to allow testing of the application different functions.

2.4.2 Safety Requirements:

We need to ensure that the device should be kept in safe and proper place so chances of device crashing during any severe accident could be minimized.

We need to ensure that the external sensor must be fit in a proper place so chances of sensor crashing during any severe accident could be minimized and sensor can provide output properly. Wires must be properly connected in the sensor, so any hard jerk won't let the sensor work abnormally.

2.4.3 Security Requirement:

The system should secure the application user's information and all other information required by the application with some useful implementation.

2.5 System Requirement Chart

ID	Priority	Type	Source	Used in Use case	Description
1	High	Functional	End User's	U1	Download an app from Play store.
2	High	Functional	End User's	U2	Creates new account to use the application.
3	High	Functional	End User's	U3	Log in required to use the application features.
4	Medium	Functional	End User's	U4	Password recovered via email if forget.
5	Low	Functional	End User's	U5	View contacts added in SOS contact list.
5	High	Functional	End User's	U6	Adding contacts to SOS contacts list.
5	Low	Functional	End User's	U7	Deleting contacts to SOS contacts list.
6	Medium	Functional	End User's	U8	Nearby hospitals can be viewed on map.
7	Medium	Functional	End User's	U9	Nearby police station can be viewed on map.
8	High	Functional	End User's	U10	Manual Emergency request can be triggered via panic button.
9	High	Functional	End User's	U11	Accident occurrence can be detected

					automatically using algorithms.
10	Low	Functional	End User’s	U12	Profile information can be updated.
11	Low	Functional	End User’s	U13	User’s/Drivers accounts can be deleted due to abnormal activities.
11	Low	Functional	End User’s	U14	Admin can track the accidents in maps through web portal.
11	Low	Functional	End User’s	U15	Admin can view the accidents history through web portal.

CHAPTER 3

DESIGN AND METHODOLOGY

3.1 Design:

- Use case diagram
- Use case description
- Domain Model
- Design Class Diagram.
- Data Model

3.1.1 Use Case Diagram:

Following is use-case diagram whole system in which every bubble responds to a narrative part as above

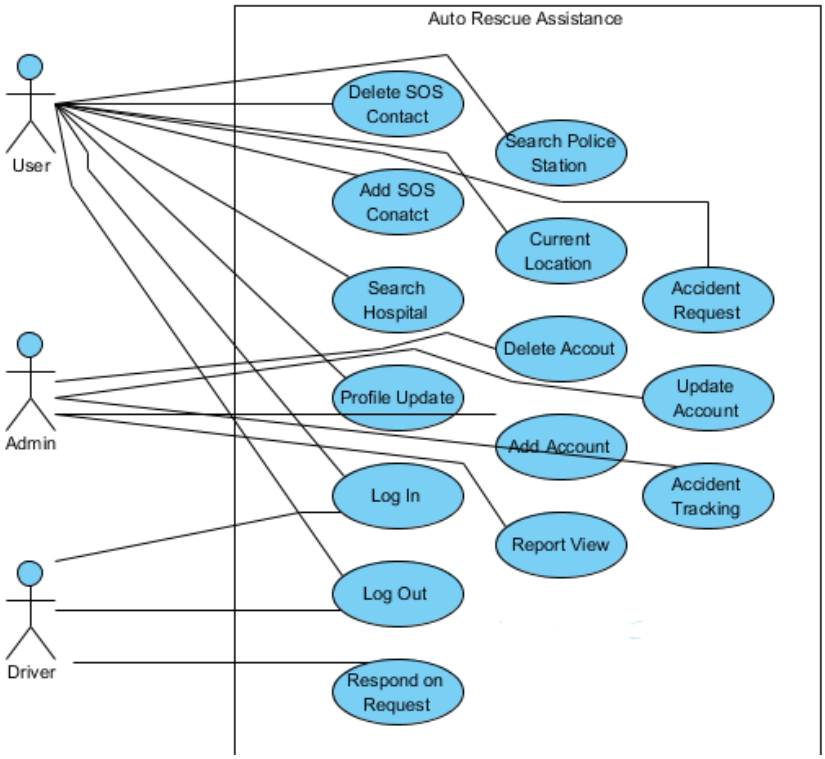


Figure 11: System Use Case

3.1.2 **Use case description**

Table 1: Use Case Login

Unique Identifier:	U2
Name	Login
Objective	A user of the System logs in to the System.
Actor(s)	Registered Users
Flow of Events	
Basic Flow	
This use case starts when a system user is not logged in to the system and goes to the login page. 1. The System prompts the user for a contact no. and password or register new account 2. The user enters his/her contact no. and password. 3. The system validates the entered contact no. and password by calling an API via Internet connection, making sure that the entered contact no. is a valid contact no. in the System, and that the required password is entered for the entered contact no. 4. When the user taps on the login button the system sends data to the API for authentication. 5. The user is signed in and returned to the home page as a Logged In User. 6. The use case ends.	
Alternate Flows	
Title	**Description**
User Fails Authentication	If the User entered an invalid contact no. and/or password, the following occurs: 1. The system describes the reasons why the User failed authentication. 2. The system presents the User with suggestions for changes necessary to allow the User to pass authentication. 3. The system prompts the User to re-enter the valid information. 4. The Basic Flow continues where the User enters new information, at step 2 of the Basic Flow.
Pre-Conditions	
Title	**Description**
Registered	User must have an account.

Post-Conditions	
Title	**Description**
Success	The User is authenticated, and the system displays a home page based on the user type.
Failure	User is unable to log in for one or more reasons.
Notes/Issues	
None	

Table 2: Use Case Registration

Unique Identifier:	U3
Name	Signup
Brief Description	A user of the System creates an account
Actor(s)	Android User
Flow of Events	
Basic Flow	
This use case starts when a system user has no account and goes to the Signup page. 1. The System prompts user for registration information, contact no., email, password, etc. 2. The user enters in their information. 3. System verifies information and creates account. 4. The use case ends.	
Alternate Flows	
Title	**Description**
Invalid Information Entered	1. User clicks submit after entering information system asked for. 2. System displays information with appropriate message to correct invalid information. 3. User re-enters information. 4. The Basic Flow continues where the User enters new information (see step 2 of the Basic Flow).
Pre-Conditions	
Title	**Description**
Account	User must not be a registered user.
Post-Conditions	
Title	**Description**

Success	The user entered successful information and is returned to the home page as a Logged in User
Failure	User is unable to sign up for one or more reasons and is returned to the login/signup selection page.
Extension Points	
None	

Table 3: Use Case Forgot Password

Unique Identifier:	U4
Name	Forgot Password
Brief Description	User will recover the password
Actor(s)	Registered User
Flow of Events	
Basic Flow	
This use case starts when the user taps on the forget password link. 1. The system will validate the user using contact number 2. The system will fetch the email corresponding to contact numbers 3. The system sends the email to email id with random password 4. Use Case ends.	
Alternate Flows	
Title	**Description**
Invalid Contact Number Entered	1. System displays information with appropriate message to correct invalid contact number. 2. User re-enters contact number. 3. The Basic Flow continues where the User enters contact number, step 1 of the Basic Flow.
Pre-Conditions	
Title	**Description**
Account	User must have an account.
Post-Conditions	
Title	**Description**
Success	Password will be reset and updated in database.

Table 4: Use Case View SOS Contacts

Unique Identifier:	U5
Name	View SOS Contacts
Brief Description	The user will able to view contacts list added in SOS
Actor(s)	Registered Android User
Flow of Events	
Basic Flow	
This use case starts when the user selects an option SOS Contacts from drawer navigation. 1. The System displays the list of contacts added in SOS. 2. Use Case ends.	
Alternate Flows	
Title	**Description**
Empty List	1. User selects the "Add" button. 2. System returns the all the available contacts. 3. User selects the desired contact from the list. 4. Selected contact will be added in list 5. The Basic Flow continues, step 1 of the Basic Flow.
Pre-Conditions	
Title	**Description**
Permissions	Read Contacts Permission should be enabled by the user.
Post-Conditions	
Title	**Description**
Success	Emergency Contacts displayed on view.
Notes/Issues	
None	

Table 5: Use Case Search Nearby Hospital

Unique Identifier:	U6
Name	Searching Near-by Hospital
Brief Description	User can search for nearby hospital based on current location
Actor(s)	Logged in Android User
Flow of Events	
Basic Flow	

This use case starts when the user selects the Near-by-Hospital option from drawer navigation. 1. The System gets the current location of the user. 2. The application search for nearest hospitals based on user current location. 3. The application displays the nearby hospitals on the map view. 4. Use Case ends.	
Pre-Conditions	
Title	**Description**
GPS	**GPS must be enabled by the user.**
Permission	GPS Permission must be given by the user to get current location.
Alternate Flows	
Title	**Description**
GPS Disabled	1. The system prompts the dialogue to enable the GPS. 2. User will enable the GPS. 3. The Basic Flow continues, step 1 of the Basic Flow.
Post-Conditions	
Title	**Description**
Success	Nearby Hospitals are successfully displayed on the Map View
Failure	Nearby Hospitals will not show on the Map View. System prompt the dialogue box to enable the GPS.
Extension Points	
None	

Table 6: Use Case Search Nearby Police Station

Unique Identifier:	U7
Name	Searching Near-by Police Station
Brief Description	User can search for nearby police station based on current location
Actor(s)	Logged in Android User
Flow of Events	
Basic Flow	

This use case starts when the user selects the Near-by-Police Station option from drawer navigation. 1. The System gets the current location of the user. 2. The application search for nearest police station based on user current location. 3. The application displays the nearby police station on the map view. 4. Use Case ends.	
Alternate Flows	
Title	**Description**
GPS Disabled	1. The system prompts the dialogue to enable the GPS. 2. User will enable the GPS. 3. The Basic Flow continues, step 1 of the Basic Flow.
Pre-Conditions	
Title	**Description**
GPS	**GPS must be enabled by the user.**
Permission	GPS Permission must be given by the user to get current location.
Post-Conditions	
Title	**Description**
Success	Nearby Police Station are successfully displayed on the Map View
Failure	Nearby Police Station will not show on the Map View. System prompt the dialogue box to enable the GPS.
Extension Points	
None	

Table 7: Use Case Panic Button

Unique Identifier:	U8
Name	Panic Button
Brief Description	Manual Request and SMS will be sent to the nearby drivers and SOS Contacts respectively
Actor(s)	Logged in User
Flow of Events	
Basic Flow	

This use case starts when a user taps on the Panic Button. 1. The system will get the current location of user 2. The system will search the nearby drivers 3. The System will send the user's current location to nearby drivers 4. The System will send the SMS with the current location of user to the Emergency Contacts 5. Use Case ends	
Pre-Conditions	
Title	**Description**
Emergency Contacts	SOS contacts list must be populated with emergency contacts
GPS	GPS must be enabled by the user
Permission	GPS permission must be given by the user to get current location
Post-Conditions	
Title	**Description**
Success	Request has successfully sent to nearby drivers Message has successfully delivered to emergency contacts
Issues/Notices	
None	

Table 8: Use Case Automatic Accident Detection

Unique Identifier:	U9
Name	Automatic Accident Detection
Brief Description	Accident will be automatically detected by the device built in sensors and the externals sensors
Actor(s)	Logged In User
Flow of Events	
Basic Flow	
This use case starts when a device detects the accident. 1. The System will get input from the external hardware. 2. The System will get the threshold orientation using gyroscope sensor. 3. The System will get the threshold speed value using accelerometer sensor. 4. If the system successfully pass the above three steps, the device will detect the situation as accident and then send the request to nearby drivers and SMS to SOS contacts. 5. Use Case End	

Pre-Conditions	
Title	**Description**
Bluetooth Enabled	Bluetooth must be enabled to get the input from external sensor.
GPS	GPS must be enabled by the user to get the current location.
Permission	GPS permission must be given by the user to get current location
Post-Conditions	
Title	**Description**
Success	Request has successfully sent to nearby drivers Message has successfully delivered to emergency contacts
Issues/Notices	
None	

Table 9: Use Case Profile Maintenance

Unique Identifier:	U10
Name	Profile Maintenance
Brief Description	User can update his/her profile.
Actor(s)	Logged in User
Flow of Events	
Basic Flow	
This use case starts when a device detects the accident. 1. User will navigate to the profile page using navigation drawer 2. User's information displayed on profile page 3. User can update any information by editing existing information in the fields 4. User will tap on update button 5. System will get the new information form filed and send the data to API 6. API will update the information in database 7. Use Case End	
Alternate Flows	
Title	**Description**
Empty Fields	1. The system prompts the error to fill all the required fields 2. The Basic Flow continues where user fill all the fields, step 3 of the Basic Flow.

Pre-Conditions	
Title	**Description**
Internet	Device must be connected to internet to call the update API
Post-Conditions	
Title	**Description**
Success	Information has been updated successfully
Issues/Notices	
None	

Table 10: Use Case Add SOS Contacts

Unique Identifier:	U11
Name	Add SOS Contacts
Brief Description	The user will able to add new contacts to SOS list
Actor(s)	Logged Android User
Flow of Events	
Basic Flow	
This use case starts when the user taps on add button in SOS Contact screen 1. The System prompts the list of available contacts stored in phone memory 2. User will select the desired contact 3. Selected contact added in the list 4. Use case ends	
Alternate Flows	
Title	**Description**
Permission	1. User selects the “Add” button. 2. System returns the all the available contacts. 3. User selects the desired contact from the list. 4. Selected contact will be added in list 5. Updated list will be shown.
Pre-Conditions	
Title	**Description**
Permissions	Read Contacts Permission should be enabled by the user.
Post-Conditions	
Title	**Description**
Success	Emergency Contacts displayed on view.

Notes/Issues
None

Table 11: Use Case Delete SOS Contacts

Unique Identifier:	U12
Name	Delete SOS Contacts
Brief Description	The user will able to delete added contacts from the SOS list
Actor(s)	Logged Android User
Flow of Events	
Basic Flow	
This use case starts when the user taps on delete button in SOS Contact screen 1. The user will navigate to the SOS contact from navigation drawer 2. User will tap on the delete button 3. The system will delete the contact from SOS list 4. Use case ends	
Pre-Conditions	
Title	**Description**
SOS List	Emergency Contact must be in the SOS list
Post-Conditions	
Title	**Description**
Success	Emergency Contacts successfully deleted from the list
Notes/Issues	
None	

Table 12: Use Case Delete Account

Unique Identifier:	U13
Name	Delete Account
Brief Description	The admin can delete the users account through web portal
Actor(s)	Admin
Flow of Events	
Basic Flow	

This use case starts when the admin clicks on delete option from the navigation drawer in web portal 1. The admin can delete the account. 2. Use case ends	
Pre-Conditions	
Title	**Description**
User account	User must be registered in system
Post-Conditions	
Title	**Description**
Success	Account has been successfully deleted
Notes/Issues	
None	

Table 13: Use Case Accident Tracking

Unique Identifier:	U14
Name	Accident Tracking
Brief Description	The admin can track the accidents through web portal
Actor(s)	Admin
Flow of Events	
Basic Flow	
This use case starts when the admin clicks on accident track option from the navigation drawer in web portal 1. The admin will able the view the accidents on map 2. Use case ends	
Pre-Conditions	
Title	**Description**
Accident Record	Accident records must be saved in the database
Post-Conditions	
Title	**Description**
Success	Accidents can easily be seen in map on respective location
Notes/Issues	
None	

Table 14: Use Case Accident Report View

Unique Identifier:	U15
Name	Accidents Report view
Brief Description	The admin can view the accidents history through web portal
Actor(s)	Admin
Flow of Events	
Basic Flow	
This use case starts when the admin clicks on view accidents report option from the navigation drawer in web portal 1. The admin will able the view the accidents history on the table 2. Use case ends	
Pre-Conditions	
Title	**Description**
Accident Record	Accident records must be saved in the database
Post-Conditions	
Title	**Description**
Success	Accidents history can easily be seen in the tables
Failure	Accidents history is empty
Notes/Issues	
None	

Table 15: Use Case Delete Account

Unique Identifier:	U16
Name	Driver Tracking
Brief Description	The admin can track the driver through web portal
Actor(s)	Admin
Flow of Events	
Basic Flow	
This use case starts when the admin clicks on dashboard option from the navigation drawer in web portal 1. The admin will able the view the drivers details on the table 2. Use case ends	
Pre-Conditions	
Title	**Description**

Active Drivers	Driver must have active status
Post-Conditions	
Title	**Description**
Success	Drivers details can be seen in the tables
Notes/Issues	
None	

Unique Identifier:	U17
Name	Respond on Request
Brief Description	The rescuer will respond on victim's request
Actor(s)	Driver
Flow of Events	
Basic Flow	
This use case starts when the victim requests for services. 1. Information alert will be generated. 2. Rescuer will track the victim location and contact information. 3. Use case ends	
Pre-Conditions	
Title	**Description**
GPS	GPS must be enabled
Internet	Internet must be available.
Post-Conditions	
Title	**Description**
Success	Victim's location and contact displayed on screen.
Notes/Issues	
None	

3.1.3 Sequence Diagram

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time. These can help to predict how a system will behave and to discover responsibilities a class may need to have in the process of modelling a new system.

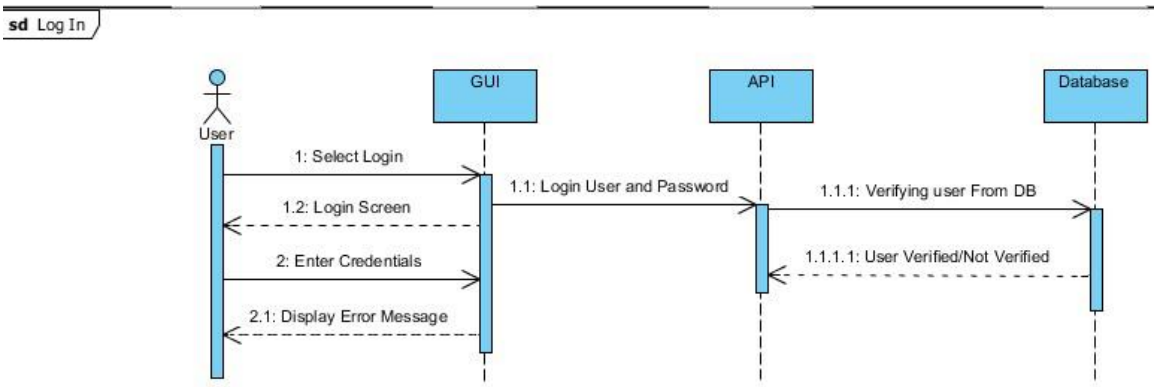


Figure 12: Login Sequence Diagram

Description:

In the above diagram, Login Sequence diagram is elaborated that the user will enter the credentials and the API will hit that will check the entered credentials will be database records and if exist then login successful otherwise error message will be generated.

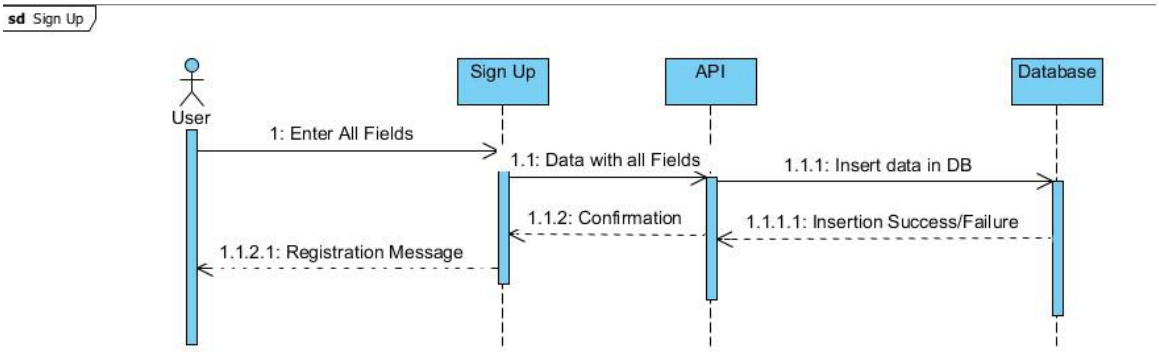


Figure 13: Sign Up Sequence Diagram

Description:

In the above diagram, the user will enter all the specific information in the required fields and then the API (Application Programming Interface) will be called which will insert the data in

the database. If it inserts successfully then the success message will show otherwise the error message will be shown to the user.

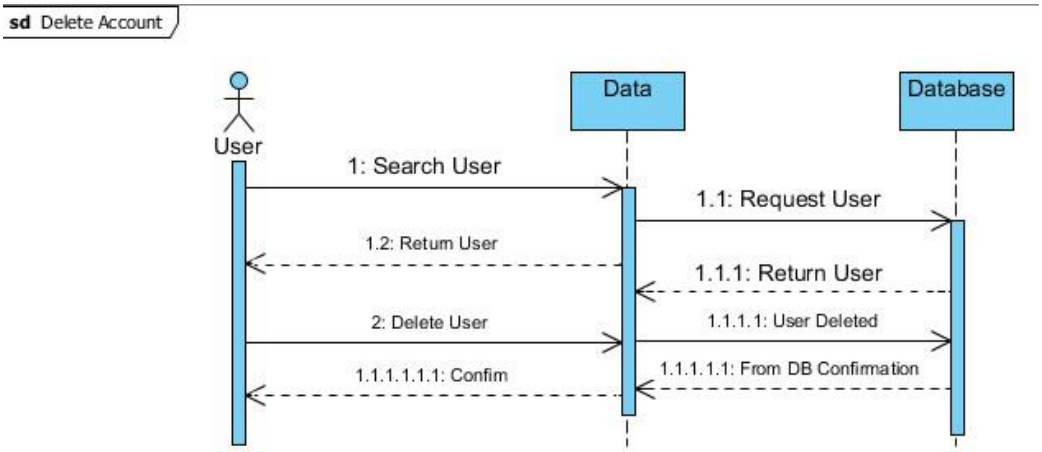


Figure 14: Delete Account Sequence Diagram

Description:

The above diagram shows the deletion of account by the admin. The admin will search for the specific user and if the users does not exist error message will be shown to the admin and if user exist then delete the user from admin panel and acknowledgement to the admin will be showed.

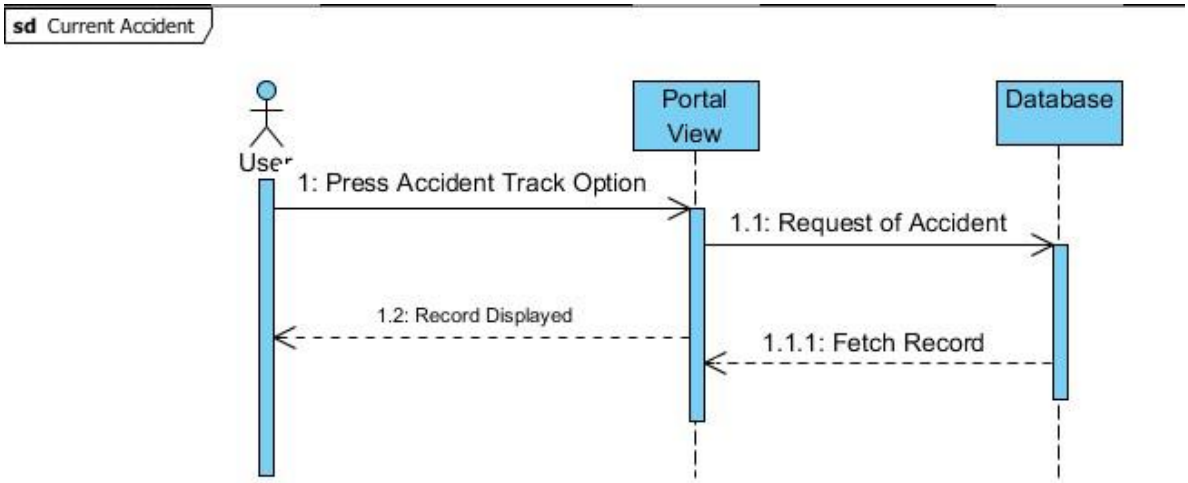


Figure 15: Current Account Sequence Diagram

Description:

In the above diagram, it is explained the admin can see the current accidents using the admin portal on web. The admin could be able to see the current accidents based on the requests put by the victims to rescue.

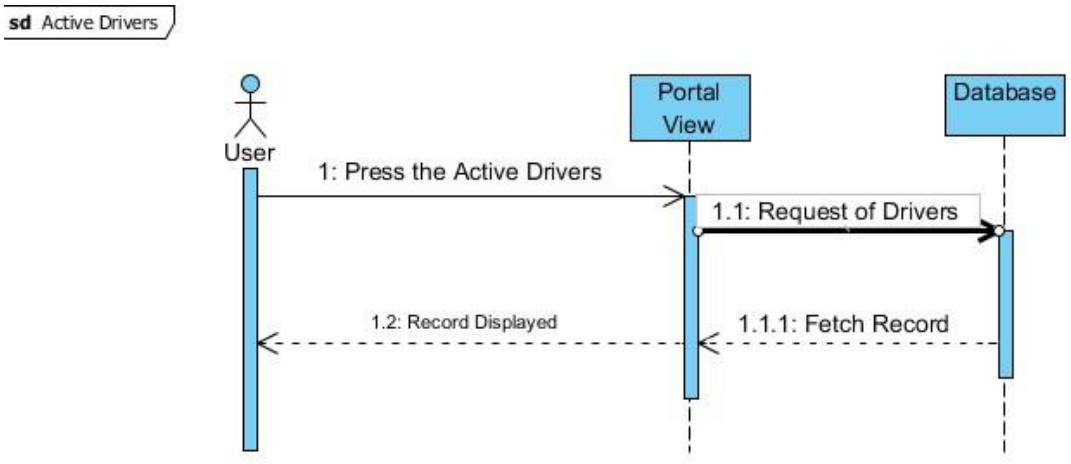


Figure 16: Active Driver Sequence Diagram

The above diagram shows the active drivers to the admin. The admin could be able to see the drivers which are on service and are readily active to serve the victims.

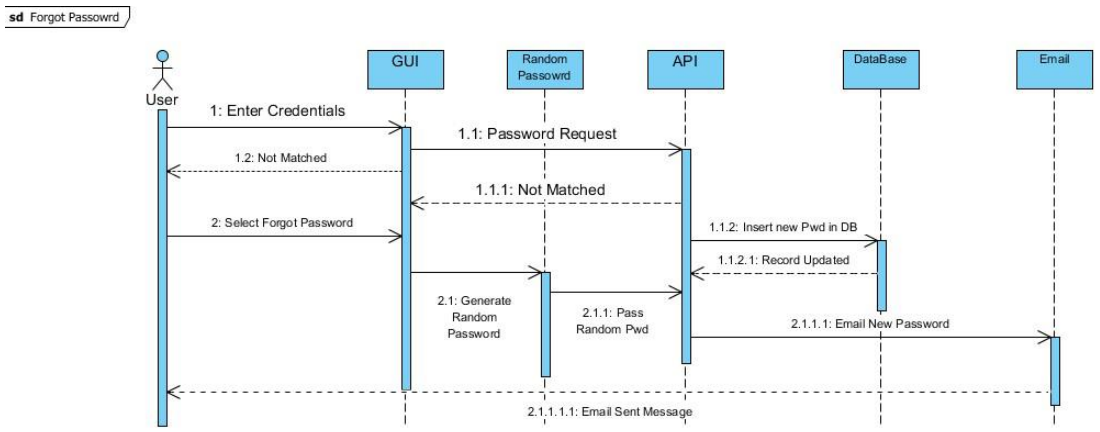


Figure 17: Forgot Password Sequence Diagram

Description:

The above diagram shows the sequence wise functionality to perform the forget password. The user enter the contact number and password but if passwords don't match over and over then the user could reset it password by clicking the link of forget password and when user clicks it

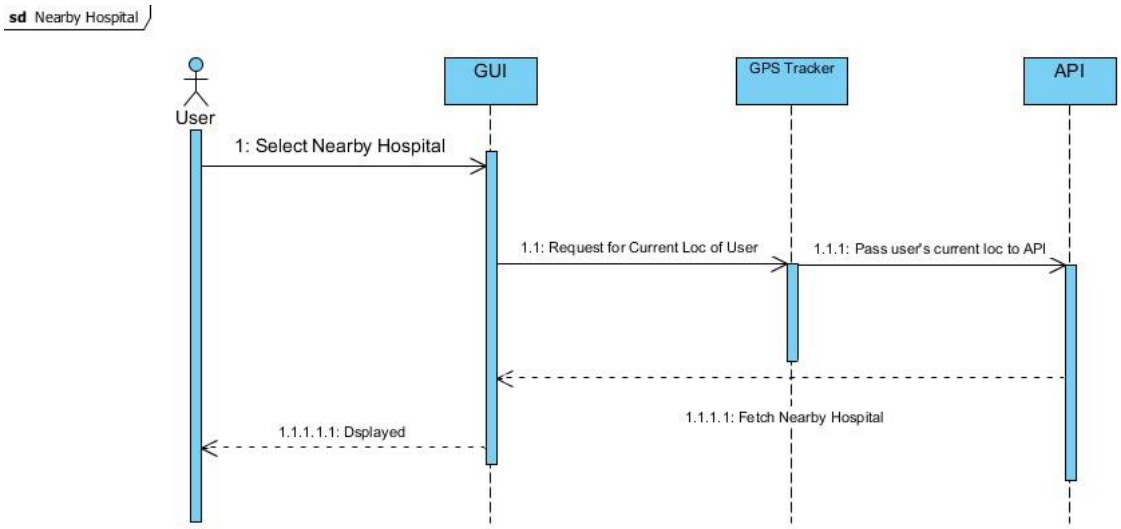


Figure 18: Nearby Hospital Sequence Diagram

the API hit and it fetch the record of user and mail the new password to the corresponding user's email.

Description:

In the above diagram, user will be able to see the nearby hospitals based on the current location. The user when press the nearby hospitals option then the users current location will be fetched and the API will hit that will return the nearby hospitals to the usr's map.

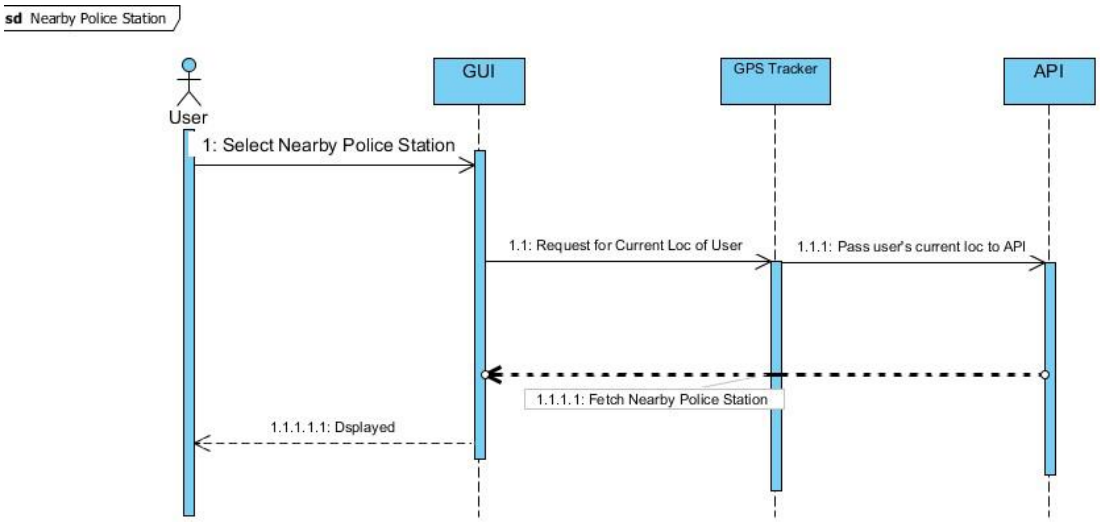


Figure 17: Nearby Police Sequence Diagram

Description:

In the above diagram, user will be able to see the nearby police station based on the current location. The user when press the nearby police station option then the users current location will be fetched and the API will hit that will return the nearby hospitals to the user's map

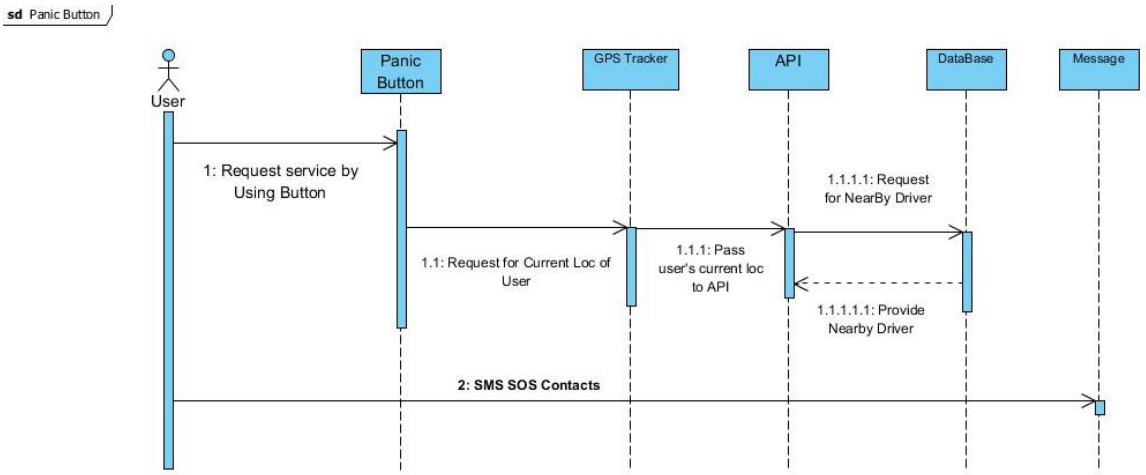


Figure 19: Panic Button Sequence Diagram

Description:

In this above diagram, the sequence of manual request described as the user presses the panic button then the users current location will be fetched and the API will hit and the nearby driver relative to users location will be fetched and the request will be send to that selected driver and the message to SOS contacts will also be send with the custom message and users current location.

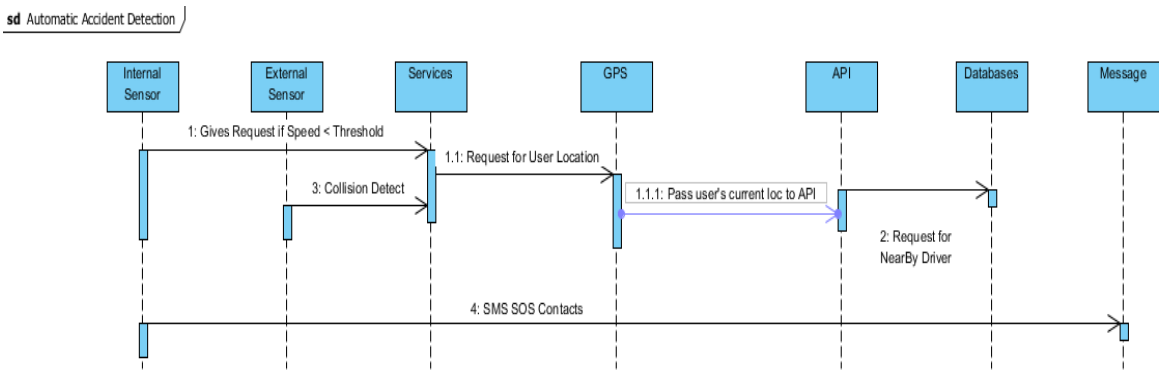


Figure 20: Automatic Accident Detection Sequence Diagram

Description:

In this above diagram, the sequence of automatic request described as the speed of vehicle and collision detection via sensor will be continuously monitored and if accident happens then the users current location will be fetched and the API will hit and the nearby driver relative to users location will be fetched and the request will be send to that selected driver and the message to SOS contacts will also be send with the custom message and users current location.

3.1.4 **Collaboration Diagram**

A Collaboration diagram is very similar to a Sequence diagram in the purpose it achieves; in other words, it shows the dynamic interaction of the objects in a system. A distinguishing feature of a Collaboration diagram is that it shows the objects and their association with other objects in the system apart from how they interact with each other. A Collaboration diagram is easily represented by modelling objects in a system and representing the associations between the objects as links. The interaction between the objects is denoted by arrows. To identify the sequence of invocation of these objects, a number is placed next to each of these arrows.

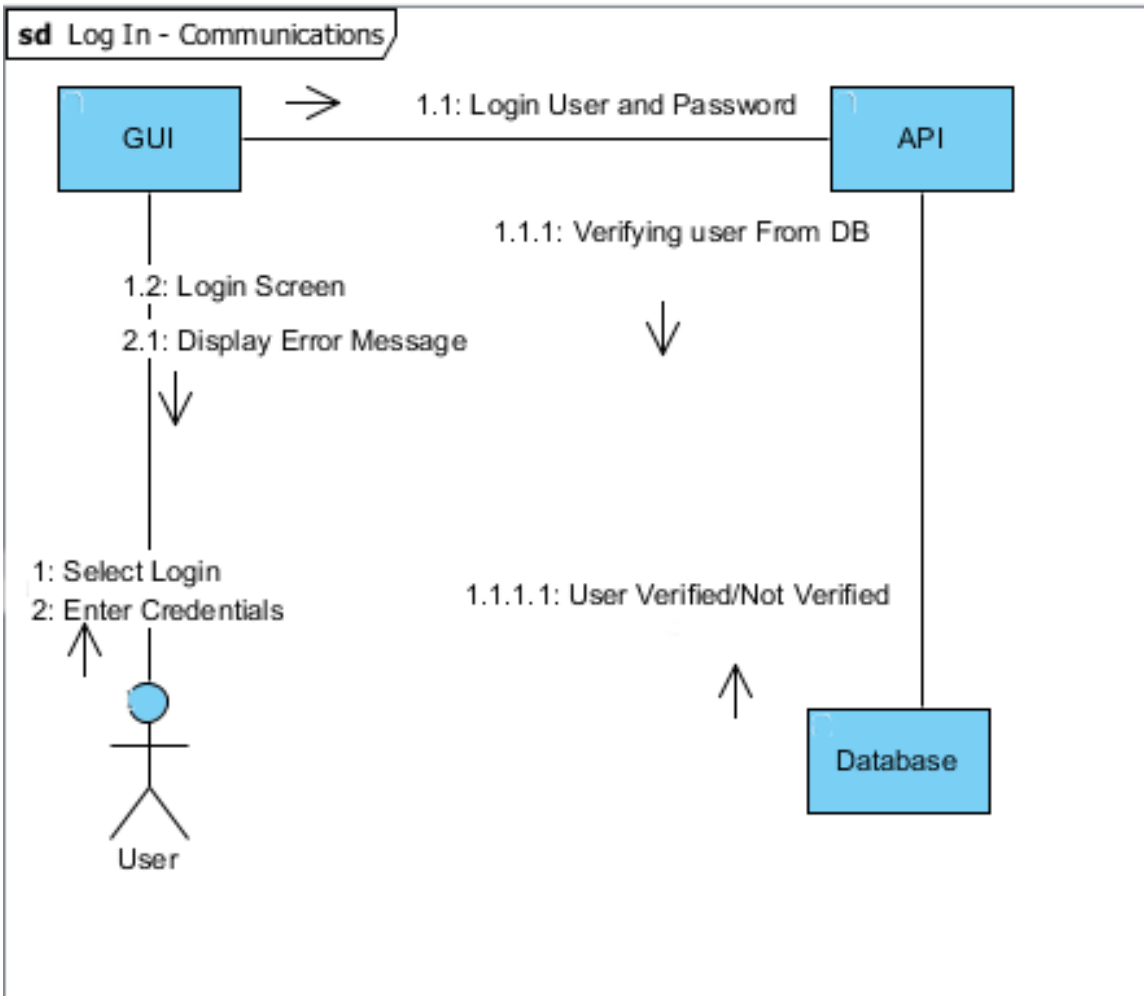


Figure 21: Log In Collaboration Diagram

Description:

The above diagrams show the communication between User's object and Database object

- User instance will be created
- Database instance will be created
- User instance populate from the interface and matched the contact no and password with the database instance.

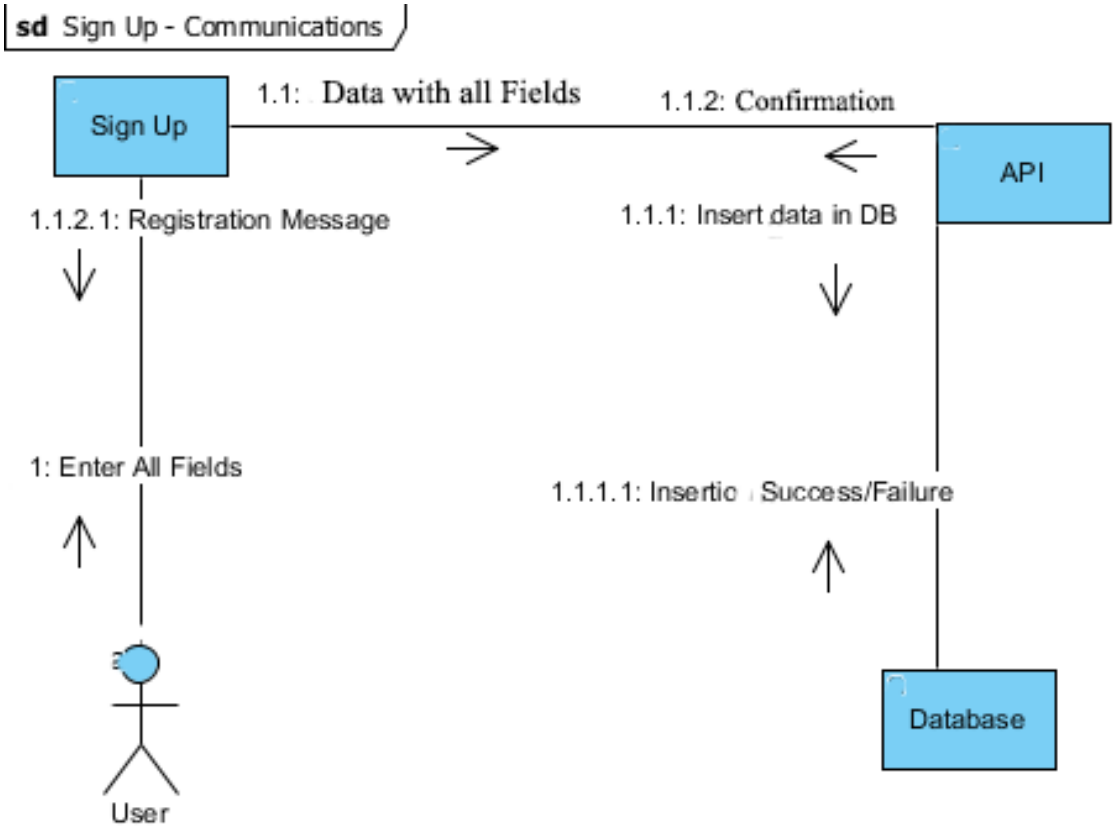


Figure 22: Sign Up Collaboration Diagram

Description:

The above diagrams show the communication between User's object and Database object

- User class instance will be created
- Database instance will be created
- User instance populate from the interface and insert in DB by communicating with the DB instance.

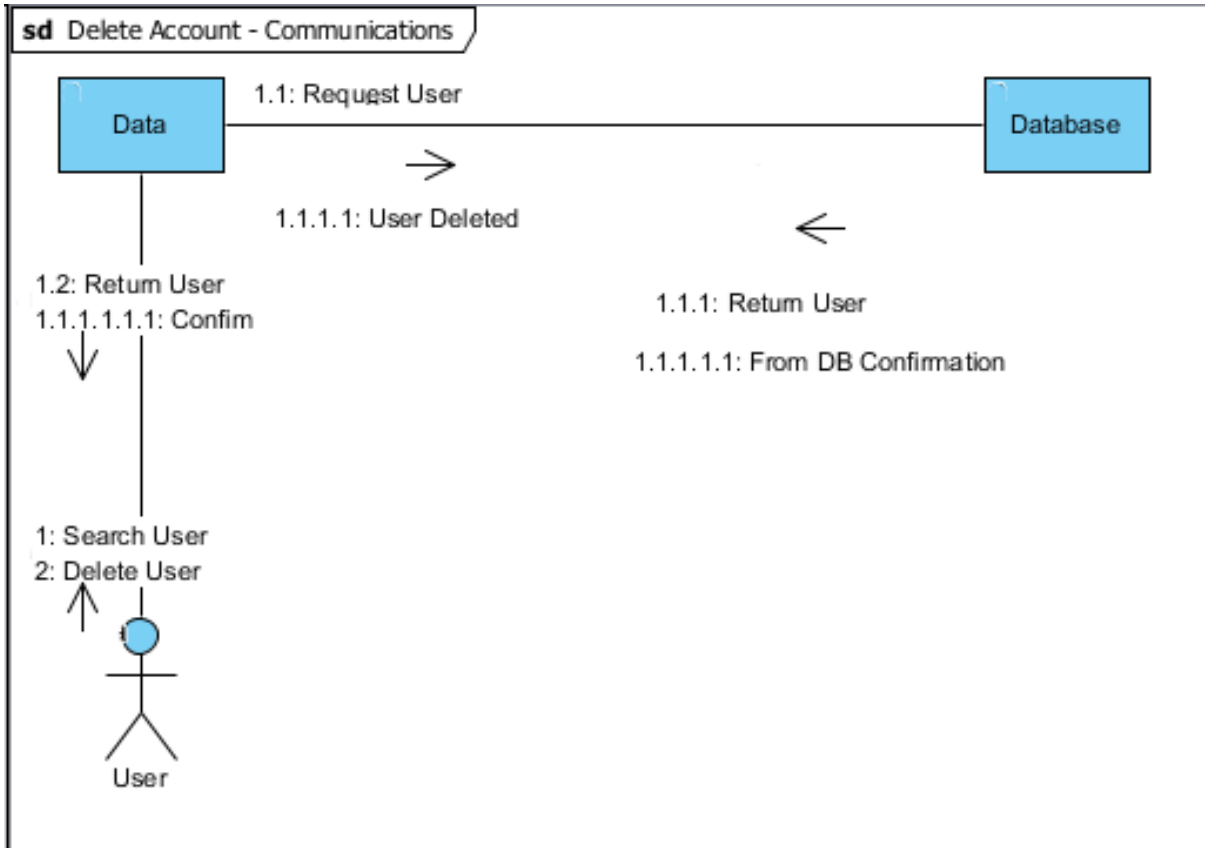


Figure 23: Delete Account Collaboration Diagram

Description:

The above diagrams shows the communication between Admin's object and Database object

- Admin's class instance will be created
- Database instance will be created
- Admin instance populate from the interface and delete specific user in DB by communicating with the DB instance.

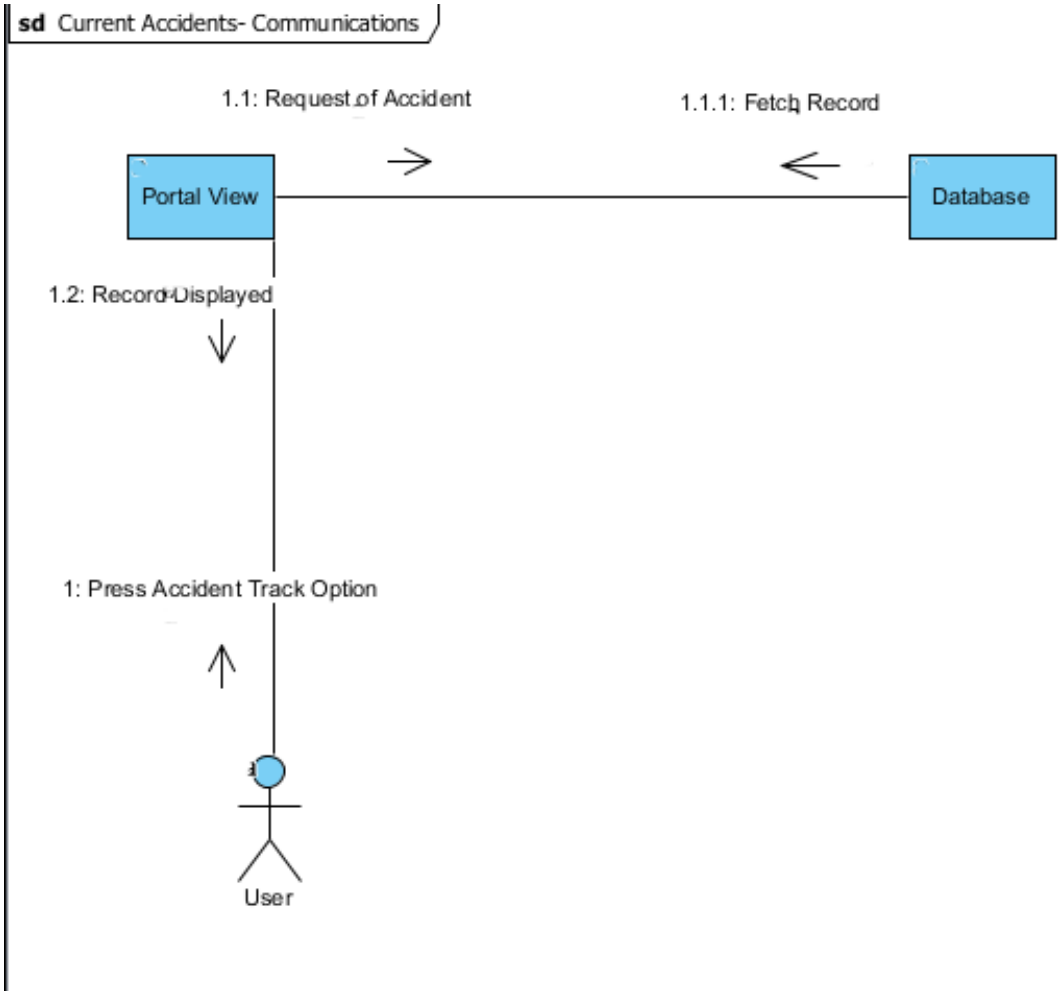


Figure 24: Current Accident Collaboration Diagram

Description:

The above diagrams show the communication between Admin's object and Database object

- Admin's class instance will be created
- Database instance will be created
- Admin interface will populate from the database instance to get the records of serving accidents from DB instance.

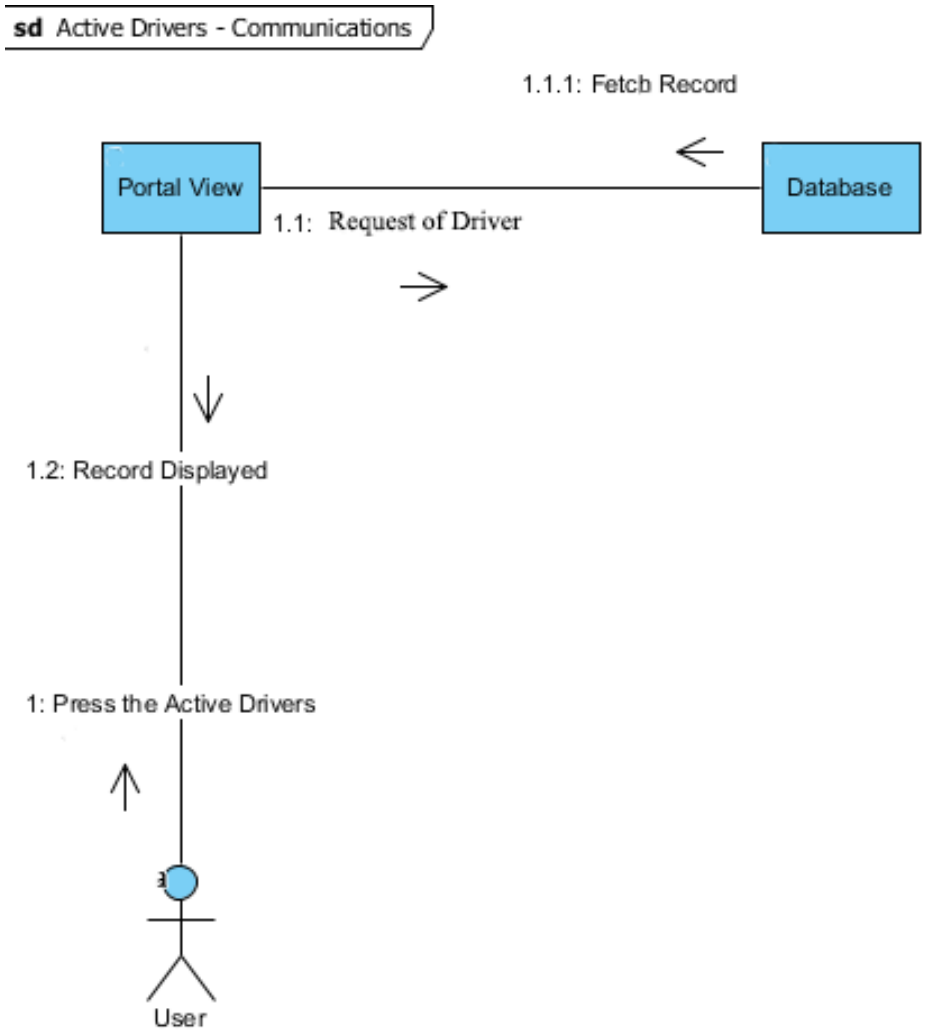


Figure 25: Active Drivers Collaboration Diagram

Description:

The above diagrams show the communication between Admin's object and Database object

- Admin's class instance will be created
- Database instance will be created
- Admin interface will populate from the database instance to get the records of active drivers from DB instance.

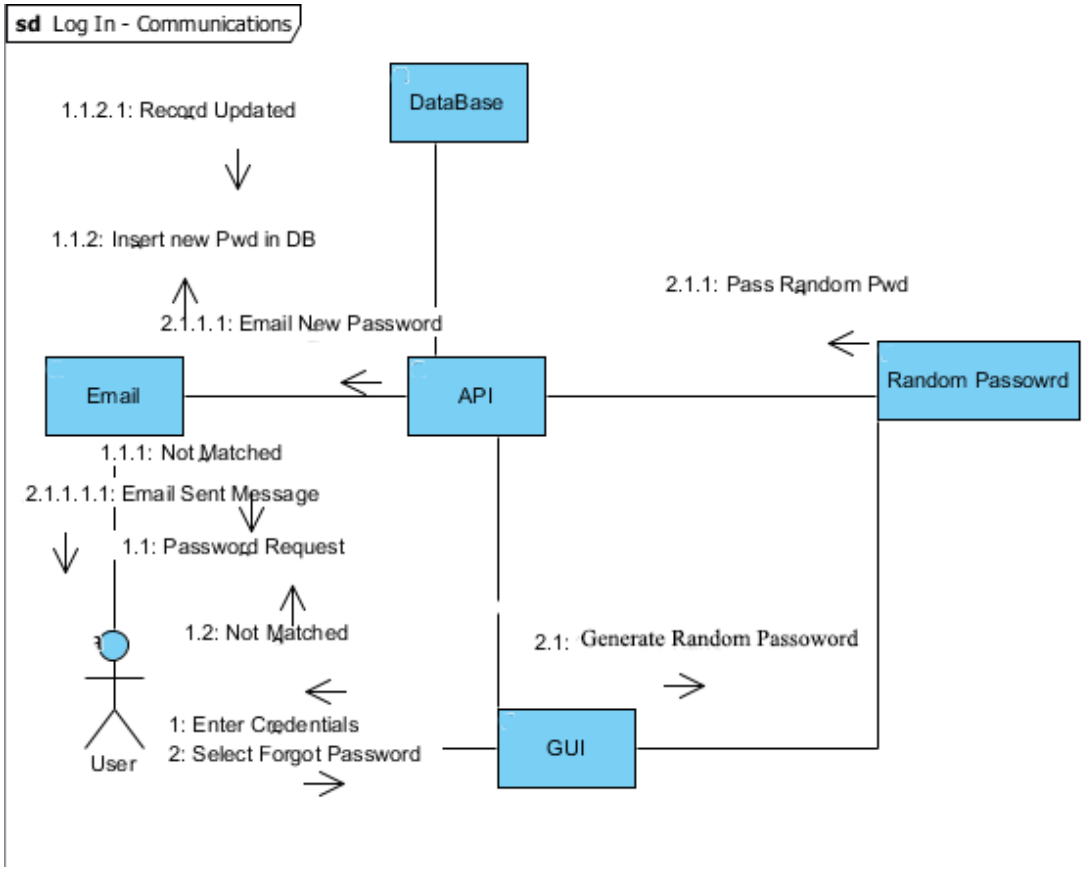


Figure 26: Update Password Collaboration Diagram

Description:

The above diagrams show the communication between User's object and Database object

- User's class instance will be created
- Database instance will be created
- Users interface will communicate with the database instance to update the new password to the DB and to send the email to users account.

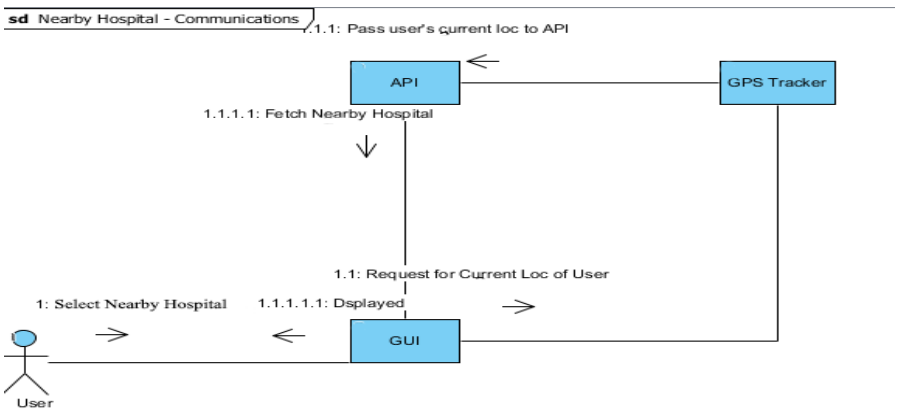


Figure 27: Nearby Hospital Collaboration Diagram

Description:

The above diagrams show the communication

- User's class instance will be created
- Users interface will communicate with the Google API instance to return the nearby hospitals based on user location.

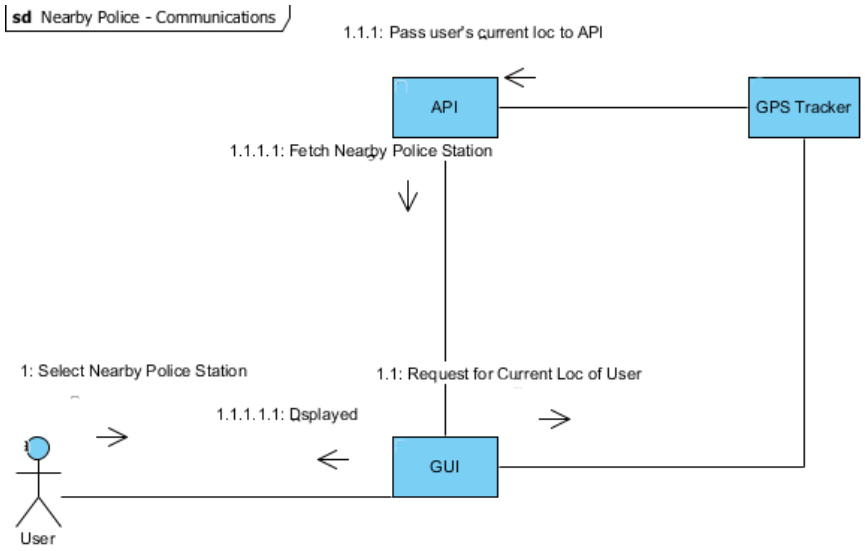


Figure 28: Police Hospital Collaboration Diagram

Description:

The above diagrams show the communication

- User's class instance will be created
- Users interface will communicate with the Google API instance to return the nearby police station based on user location.

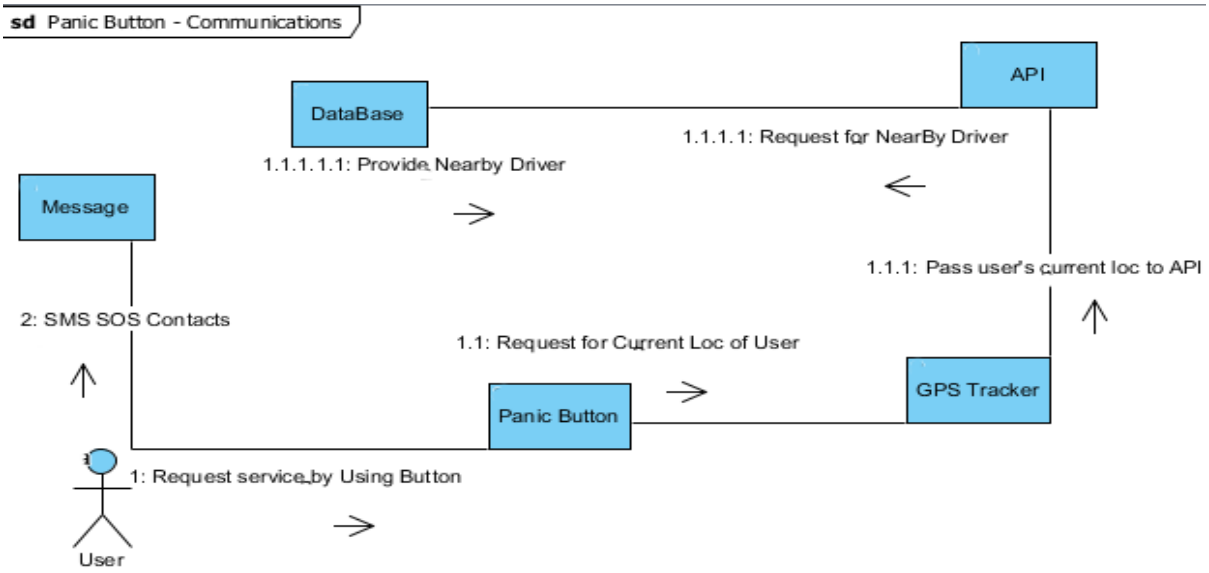


Figure 29: Panic Button Collaboration Diagram

Description:

The above diagrams show the communication Users object and Database object and Drivers object

- User's class instance will be created
- Database class instance will be created
- Users instance will communicate with the db instance upon request to get the nearby driver to user's location and to send the command to driver provide the rescue to victim.

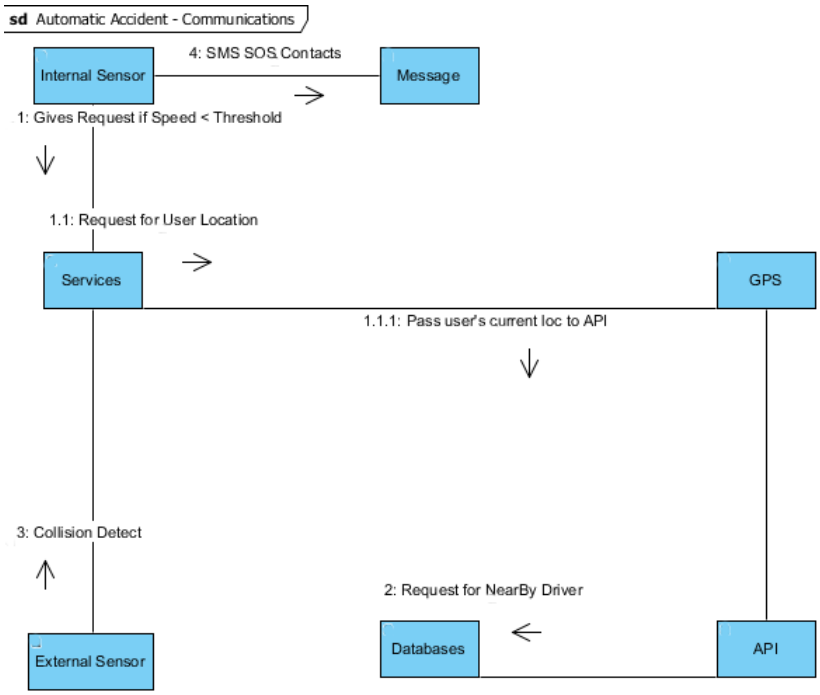


Figure 30: Automatic Accident Detection Collaboration Diagram

Description:

The above diagrams show the communication Users object and Database object and Drivers object

- User's class instance will be created
- Database class instance will be created
- Users instance will communicate with the database instance upon automatically detection to get the nearby driver to user's location and to send the command to driver provide the rescue to victim.

3.1.5 **Operation Contracts**

Name	Login
Responsibilities	Logged in user to application
Cross Reference	U2
Exception	Wrong contact number or password entered
Preconditions	User entered all required credentials
Post conditions	- User instance was created - Database instance was created - User instance populate from the interface and matched the contact no and password with the database instance.

Name	Sign Up
Responsibilities	To register user at Auto Rescue Assistance
Cross Reference	U3
Exception	Incomplete credentials.
Preconditions	User entered all required credentials
Post conditions	- User class instance was created - Database instance was created - User instance populate from the interface and insert in DB by communicating with the DB instance.

Name	Forget Password
Responsibilities	Send new random password to user email if its forget.
Cross Reference	U4
Exception	Incomplete or Wrong Contact number entered in the field
Preconditions	User must have an account
Post conditions	- User instance was created - Database instance was created - User instance pass contact number to DB instance and match with the contact no in DB. - Random password sent to user email.

Name	View SOS contacts
Responsibilities	User will be able to look at the added contacts in SOS list.
Cross Reference	U5
Exception	None
Preconditions	User must have permission to phone contact storage and Contacts must be added in DB.
Post conditions	- Data base instance created and instance populate the list view from SQL list DB.

Name	Search Nearby Hospital
Responsibilities	To find out the nearby hospital based on current location of user.
Cross Reference	U6
Exception	GPS and Internet connection not available.
Preconditions	GPS and Internet must be available.
Post conditions	- Show nearby Hospitals in Map view using an API.

Name	Search Nearby Police Station
Responsibilities	To find out the nearby police station based on current location of user.
Cross Reference	U7
Exception	GPS and Internet connection not available.
Preconditions	GPS and Internet must be available.
Post conditions	- Show nearby Police Station in Map view using an API.

Name	Panic Button
Responsibilities	To manually request the services.
Cross Reference	U8
Exception	Internet Connection not available or SMS permission not allowed.
Preconditions	Internet connection and GPS available and SMS permissions are given.
Post conditions	- User instance was created. - Database instance was created - Loc_Mapping instance was created - User instance communicate with DB instance to get the user information - User instance communicates with the Loc_Mapping instance to match the nearest driver for request. - Loc_Mapping instance insert the data in DB instance.

Name	Automatic Accident Detection
Responsibilities	To detect the accident automatically and request services automatically.
Cross Reference	U9
Exception	Bluetooth sensor not working properly or Internet Connection and Gps unavailable.
Preconditions	Sensors and Bluetooth module must be working properly and GPS and Internet available.
Post conditions	- User instance was created. - Database instance was created - Loc_Mapping instance was created - User instance communicate with DB instance to get the user information - User instance communicates with the Loc_Mapping instance to match the nearest driver for request. - Loc_Mapping instance insert the data in DB instance.

Name	Profile Maintenance
Responsibilities	To update user data in database.
Cross Reference	U10
Exception	Fields are not field or empty field's exception.
Preconditions	Fields properly filled and Internet available.
Post conditions	- User class instance was created - Database instance was created - User instance get the data from interface - User instance insert the data in DB instance

Name	Add SOS contacts
Responsibilities	To add contacts in app SOS list from phone contact memory.
Cross Reference	U11
Exception	Phone Storage permissions aren't allowed.
Preconditions	Permissions allowed.
Post conditions	- Sql lite instance was created. - Selected Contacts added in Sql lite instance.

Name	Delete SOS contacts
Responsibilities	To delete contact from SOS list.
Cross Reference	U12
Exception	None
Preconditions	Contacts must be present in SOS list
Post conditions	- Sql lite instance was created. - Selected Contacts delete in Sql lite instance.

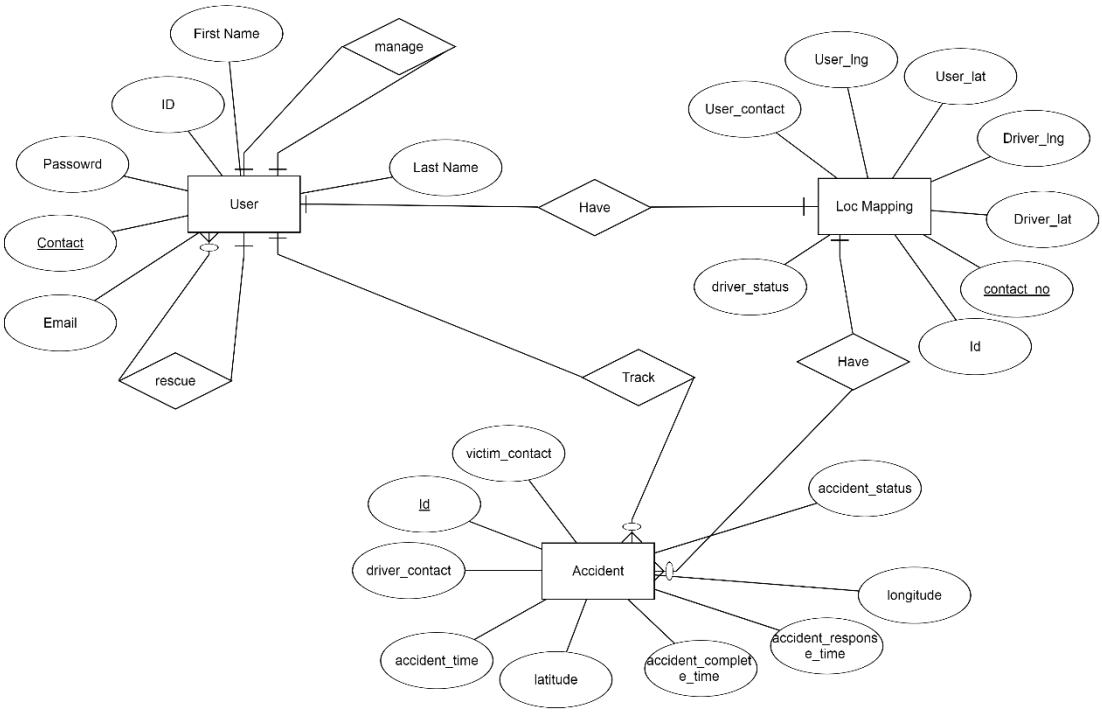
Name	Accident Tracking
Responsibilities	To online view of Accidents.
Cross Reference	U14
Exception	None
Preconditions	Admin must be logged in and data must be exist in DB.
Post conditions	- Accident class instance was created. - Database instance was created - DB instance put the accidents details to Accident instance that are in serving state

Name	Accident Report View
Responsibilities	Admin can view the accident history through web portal.
Cross Reference	U15
Exception	None
Preconditions	Admin must be logged in and accident history must be present in DB.
Post conditions	- Accident class instance was created. - Database instance was created - DB instance put the accidents details to Accident instance. - Accident instance generates report based on data exist in DB.

3.1.6 **Data Model**

The data model is a subset of the implementation model, which describes the logical and physical representation of persistent data in the system. Following is ERD and internal model of Auto Rescue Assistance database.

ERD



3.1.7 Internal Model

Users

Attribute	**Type**
ID	Serial
First Name	Character Varying (50)
Last Name	Character Varying (50)
Email	Character Varying (150)
Password	Character Varying (100)
Contact (PK)	Character Varying (15)
Blood Group	Character Varying (3)
Role	Integer
Is Active	Boolean
Insert Date	Timestamp with time zone
Update Date	Timestamp with time zone

Accident

Attribute	**Type**
ID (PK)	Serial
Victim Name	Character Varying (50)
Driver Name	Character Varying (50)
Latitude	Double Precision
Longitude	Double Precision
Accident Status	Character Varying (10)
Accident_time	Timestamp with time zone
Accident_Response_Time	Timestamp with time zone
Accident_Complete_Time	Timestamp with time zone

Loc Mapping

Attribute	**Type**
ID (PK)	Serial
Victim Name	Character Varying (50)
Driver Contact (FK)	Character Varying (20)
Driver_Lat	Double Precision
Driver_Long	Double Precision
Victim_Lat	Double Precision
Victim_Long	Double Precision
Driver_Status	Boolean

3.2 Methodology

The methodology we are using in this project is Agile development approach in which the complete process is segregated into several different development cycles. In this approach, it requires dividing an entire task into different sub tasks and thus, each of sub tasks become a separate module for a development team. In agile we use FDD, Feature Driven Development method.

3.2.1 Plan and Analysis:

The complete knowledge of the scope of the project and the current state of the domain targeted by the project will be tracked and analysed. We will build the feature list and plan by feature of the project. After the feature list is completed, the next step is to produce the development plan that divide features into sub-features and make groups of related features and dependencies of the features on another will be tracked and prioritized for smooth and efficient development. Research papers are the primary source for the analysis and research.

3.2.2 Design

In design phase we will follow the below given steps:

- Possible Activity Diagrams for each feature.
- Possible UML/Use cases for each feature.
- Design the user interfaces for each feature.

3.2.3 Implementation

Features List:

Following are the list of features that are extracted in earlier phases:

- Build the user interfaces.
- Integrate interfaces with database using Node JS.
- Implement the interfaces according to roles of user.
- Integration of Google Map APIs.
- Integration of Push Button Sensor with Application interface.
- Develop the database algorithms to maximize efficiency.
- Automatic accident detection.

Approach for Development of each Feature:

Following steps will be followed for every feature development:

- Develop the user interfaces.
- Integrate interfaces with database using Node JS.
- Apply the user rights according to the role.

3.2.4 Testing and Evaluation:

During the development process, unit testing will be done to ensure all modules are built correctly. System integration testing will be done after we have built all the components and combined them into the application.

CHAPTER 4

Implementation

4.1 Front-End Implementation:

4.1.1 ANDROID APPS:

The IDE we are using for the android application development is Android Studio. Android Studio is an integrated development environment (IDE) from Google that provides developers with tools needed to build applications for the Android OS platform. [1]

4.1.1.1 Libraries Used for UI:

- Fancy Buttons library used to design the attractive buttons for the applications.
- Card view library used to design the card view layout for the application.
- Sweet Alert library used to make the attractive alert box.
- Material design library used to make eye-catching fields for the forms like sign up and login.

4.1.1.2 Libraries used for HTTP:

Volley is an HTTP library that makes networking for Android apps easier and most importantly, faster. Volley offers the following benefits:

- Automatic scheduling of network requests.
- Multiple concurrent network connections.
- Transparent disk and memory response caching with standard HTTP cache coherence.
- Support for request prioritization.

- Cancellation request API. You can cancel a single request, or you can set blocks or scopes of requests to cancel.
- Ease of customization, for example, for retry and back off.
- Strong ordering that makes it easy to correctly populate your UI with data fetched asynchronously from the network.
- Debugging and tracing tools.

4.1.1.3 **JSON Requests in Volley:**

Volley has Requests for JSONObject and JSONArray. The structure of JSONRequest and most of the request classes included in Volley uses constructors like the following.
JsonObjectRequest request JsonObjectRequest(RequestMethod, URL, null, new ResponseListener(), new ErrorListener());
Parameters passed into the constructor:

- RequestMethod (GET, POST, PUT, DELETE, etc.)
- URL: String of the URL of the required object
- JSONObject: An optional object posted with the request, null if there is no object posted
- ResponseListener: Response Listener, whose callback method will contain the response
- ErrorListener: A Response.ErrorListener whose callback method will contain any problem with the request.

4.1.1.4 **Maps Library:**

Google Maps library used to integrate the map in the application that contains a set of utilities that can use to make their applications even better with enhanced maps like the custom markers and route on the maps.

4.1.1.5 **Notification Library:**

Pusher makes it easy to send push notifications to Android apps. It takes the hassle out of managing device tokens and interacting with Apple and Google's messaging services [2]

4.1.2 **WEB PORTAL:**

The web portal for the ARA is built in Angular as Angular has attested widespread domination in Open Source JavaScript Frameworks and it is highly appreciated amongst developers and enterprises for its high functioning solutions. Angular is a framework for building client applications in HTML and either JavaScript or a language like Type Script that compiles to JavaScript. Angular, an advanced client-side MVW framework, is highly adopted nowadays for web app development. Due to the fame of single page applications angular is highly demanded in market as the single page applications are very much faster due to loading of required component at a time instead of loading the whole page.[3]

- Components: Components help to build the applications into many modules. This helps in better maintaining the application over a period of time.
- Type Script: This is a superset of JavaScript and is maintained by Microsoft.
- Services: Services are a set of code that can be shared by different components of an application. So for example if you had a data component that picked data from a database, you could have it as a shared service that could be used across multiple applications.

4.2 **Server-Side Development:**

The API’s we built for the interaction of application with the database is NODE JS. Node.js is known for Event-driven I/O server-side JavaScript environment [5]

- Node.js is fast: Node.js uses JavaScript in the backend, and that’s enough to understand how fast the codes execute.

- The ever-growing NPM: The number of modules in the Node Package Manager (NPM) has increased at a considerable pace.
- It's perfect for handling lots of requests that are I/O driven (e.g. operations on database) and scales very nicely.
- Ideal for data-heavy websites and apps
- Capable of handling traffic spikes

4.2.1 **Libraries used in NODE JS:**

- Nodemailer: It is a module for Node.js applications to allow easy as cake email sending.
- Pusher: Push Notifications makes it easy to send push notifications to mob apps. It takes the hassle out of managing device tokens and interacting with Apple and Google's messaging services.

4.3 Backend Development

The database used for the backend development is PostgreSQL or simply called Postgres and it’s a type of object-relational database server with open source user license. Postgres database server has evolved through more than 15 years of development and maintenance and is considered one of the most reliable databases with extensive focus on data reliability and correctness. Postgres supports all the modern features of any database system and can be installed and run on variety of platforms including Linux, Windows, and all the versions of UNIX. [4] There are several reasons why we should use Postgres for database development purposes

4.3.1 Open Source

The first major benefit of using Postgres is that it is open source and can be customized.

4.3.2 Large Developer Community

Postgres has been in the market for more than 15 years and its developer community has immensely grown in this time. Large developer community means good support and help in solving database related problems.

4.3.3 Portable

Good thing about Postgres is that it is portable with almost all the major platforms and programming languages. This database is ideal for applications targeting multiple platforms.

4.3.4 Developers Tool and GUI

Postgres database server does not require extensive command line configurations. Several tools and GUI interfaces have been developed which can aid you in easy installation and management of the database server.

4.3.5 Reliability and Stability

Postgres is worldwide recognized as being the most reliable and stable database. The chances of database crashing are minimal and even if database crashes there are ways and features which allow you to restore and recover the data.

Chapter 5

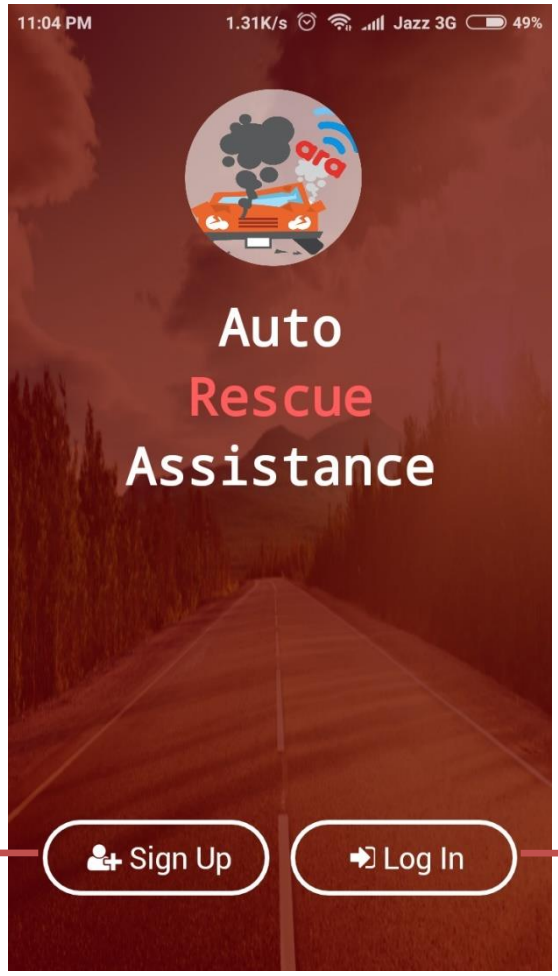
User Manual

5.1. User Android Application

5.1.1 Getting Started

5.1.1.1 Application Welcome Page:

When the application is launches for the first time, login/signup page will be displayed. If the user is not registered or the user wants to create an account, he must choose Sign Up button which will take him to the Sign-up screen. If the user is already registered, then he must choose login button that will take him to login screen.

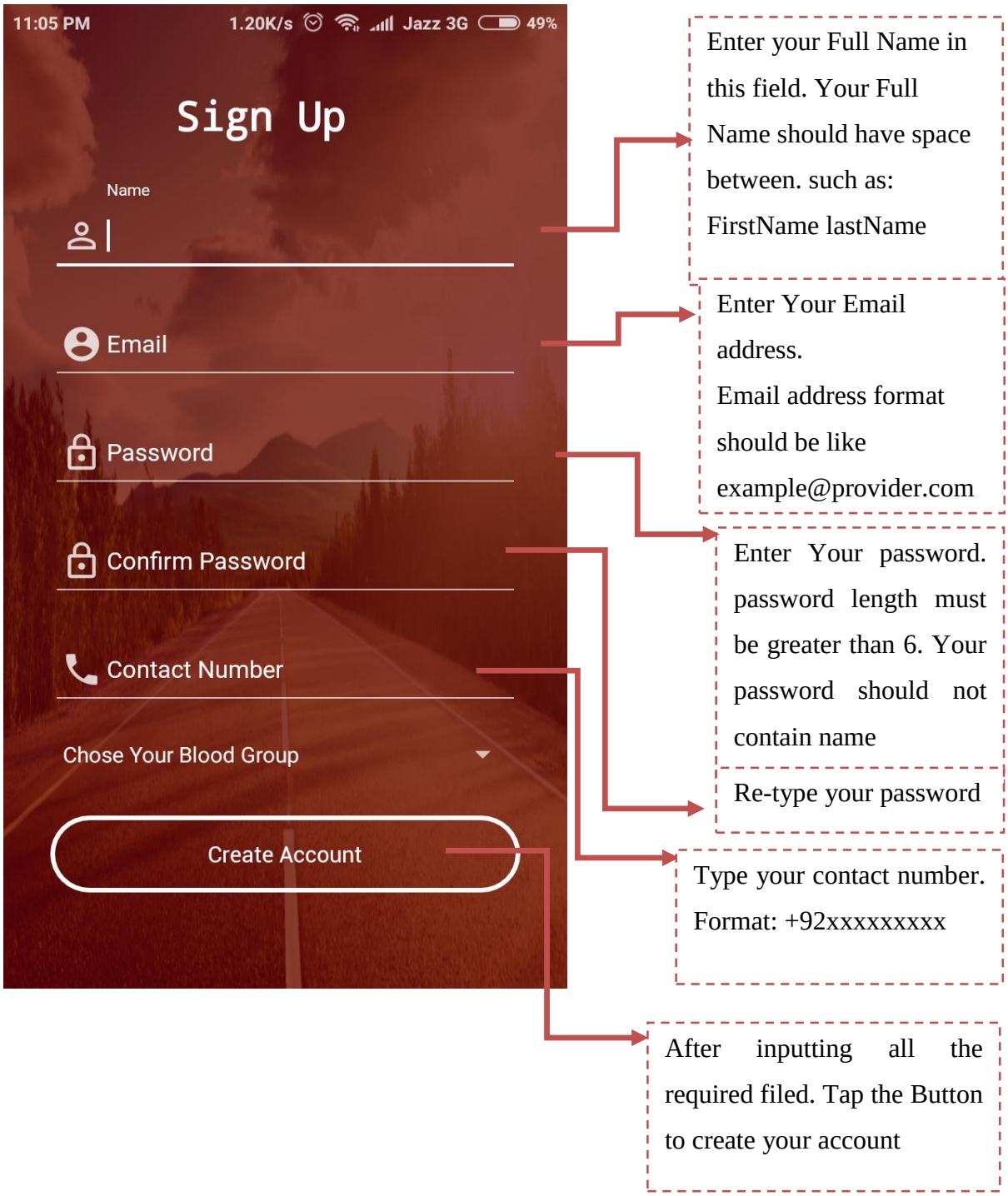


Pressing the button will take the user to signup screen.

Login button will launch the login page, where user can login to the app.

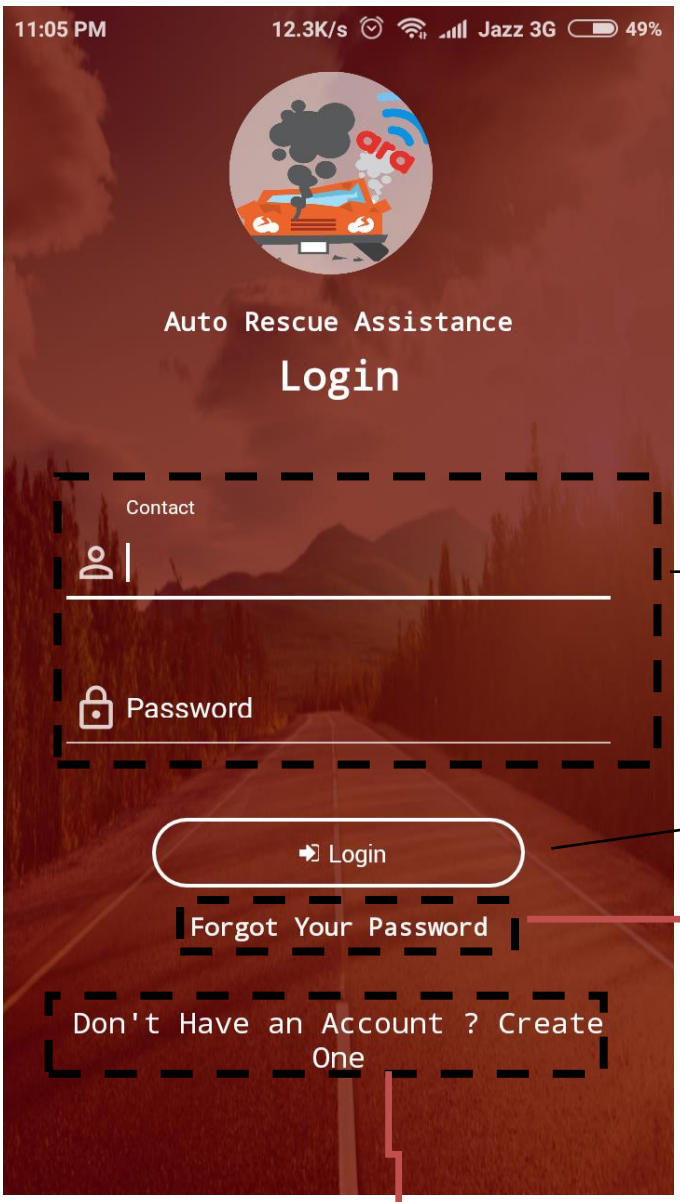
5.1.1.2 **Registration**

To use the Auto Rescue Assistance, you must be a registered user.



5.1.1.3 **Log In**

To use the app, you need to sign in using the email address and password of your ARA account.



Sign in to the app by typing in your contact and password in the allocated fields.

Press the "Sign In" button to complete the process.

If you've forgotten your password, you can reset it by selecting "Forgot Your Password". We'll send you an email with instructions.

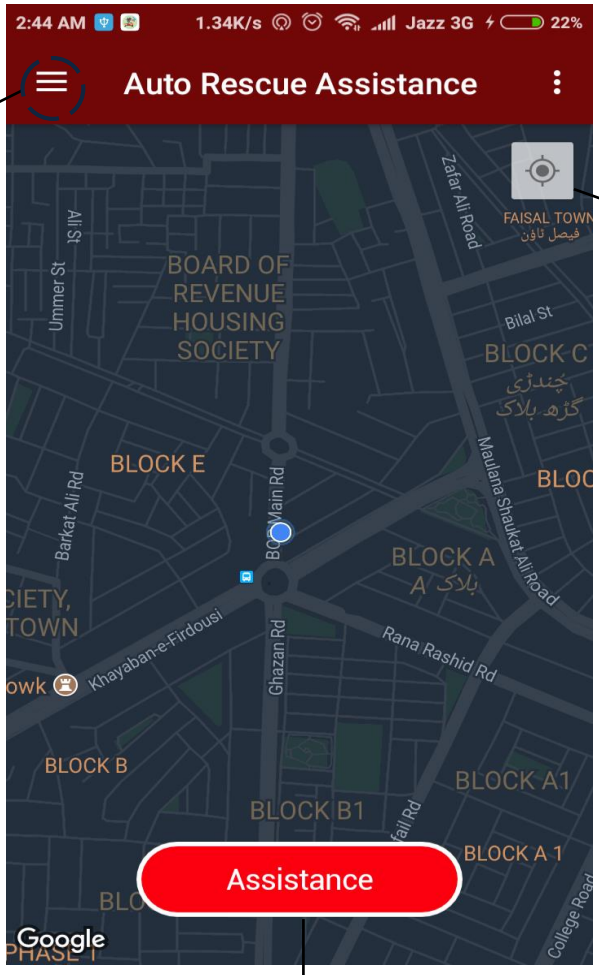
If you do not have an ARA account yet, tap the link to create your account

5.1.2 Application Overview

After successfully log in or sign up, application main/home screen will be launch. It will contain integrated google map with user's current location as shown in fig and other functionality as described below in detail.

5.1.2.1 Home Screen

Home screen is the main page of the application that contains the Assistance button, Current location button and the integrated google map with user's current location.



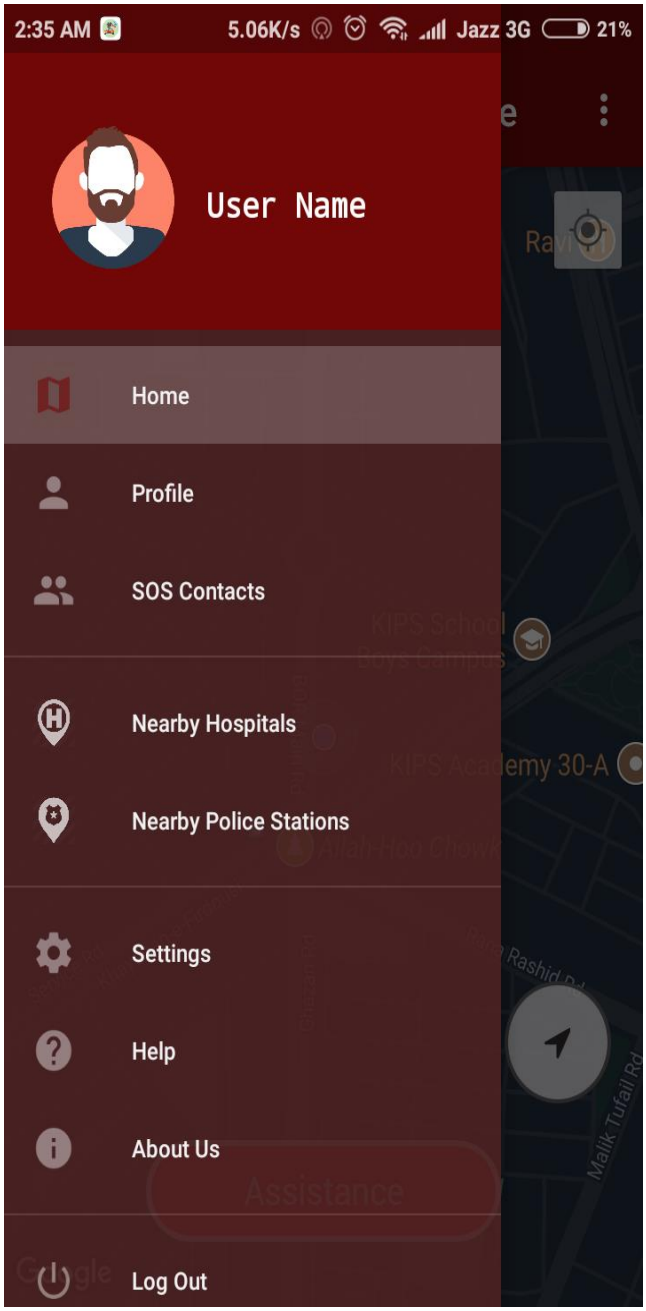
Opens the Application Drawer Menu as shown in article 5.1.2.2

It will move the map to user's current location.

Assistance button, it can be pressed whenever user is in an emergency to call the

5.1.2.2 **App Navigation Drawer**

The navigation drawer is a UI panel that shows app's main navigation menu. It is hidden when not in use but appears when the user swipes a finger from the left edge of the screen or, when at the top level of the app, the user touches the drawer icon in the app bar.

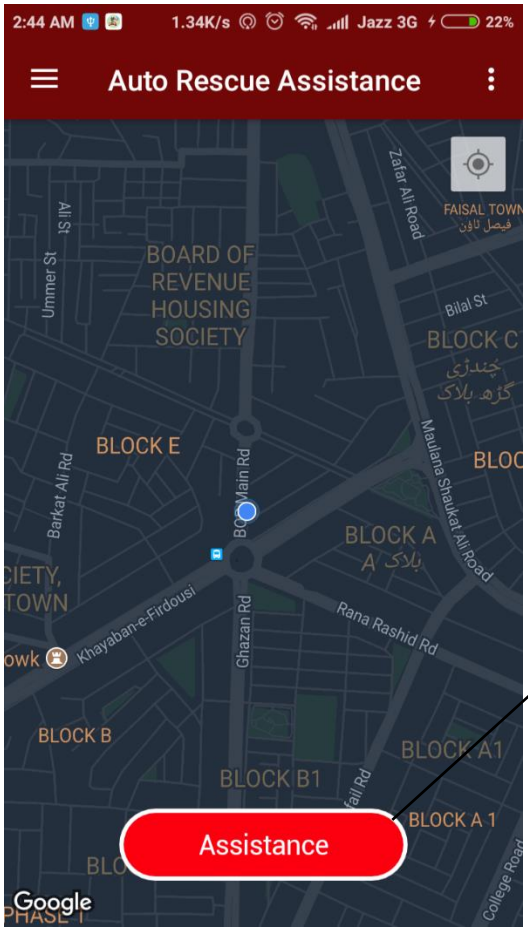


1. It will display the information of currently logged in user.
2. It has the list of emergency contacts.
3. It will show all the nearby emergency places, i.e. Hospitals, Police Stations.
4. It will allow user to set the settings of application.
5. It will demonstrate the user how this application works
6. It will show the details of the developers.
7. User will logout from the device.

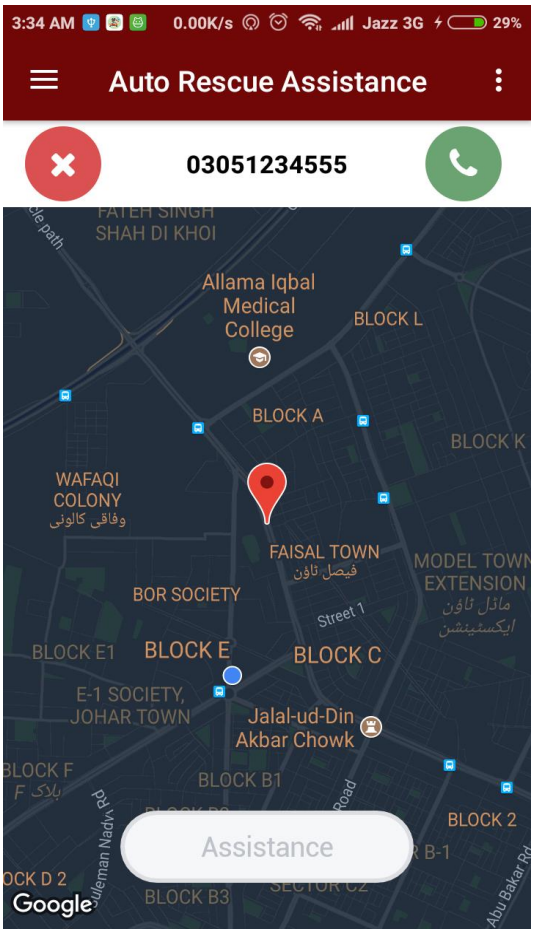
5.1.3 Application Functionality

5.1.3.1 Requesting an Emergency Service

When a user is in an emergency then he/she can request the emergency service by pressing the “Assistance” button in home page, which will send a request to nearby available ambulance driver. Map will populate with the driver’s information, when the request is accepted by the driver.



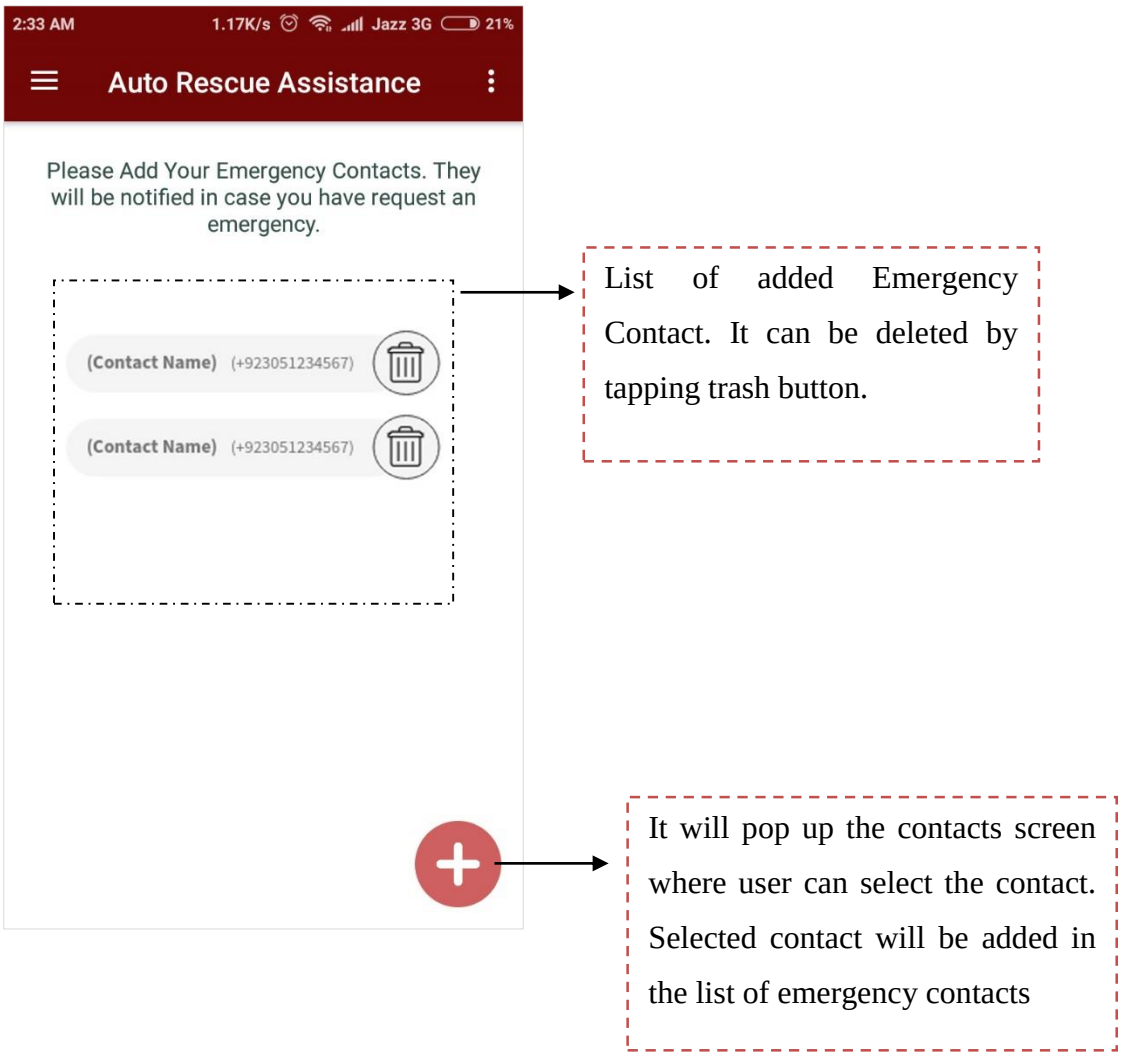
Pressing the “Assistance” button will send a request to nearby ambulance driver.



When a driver will accept request, map will populate with the driver’s contact number. User can also contact the driver by pressing the phone button.

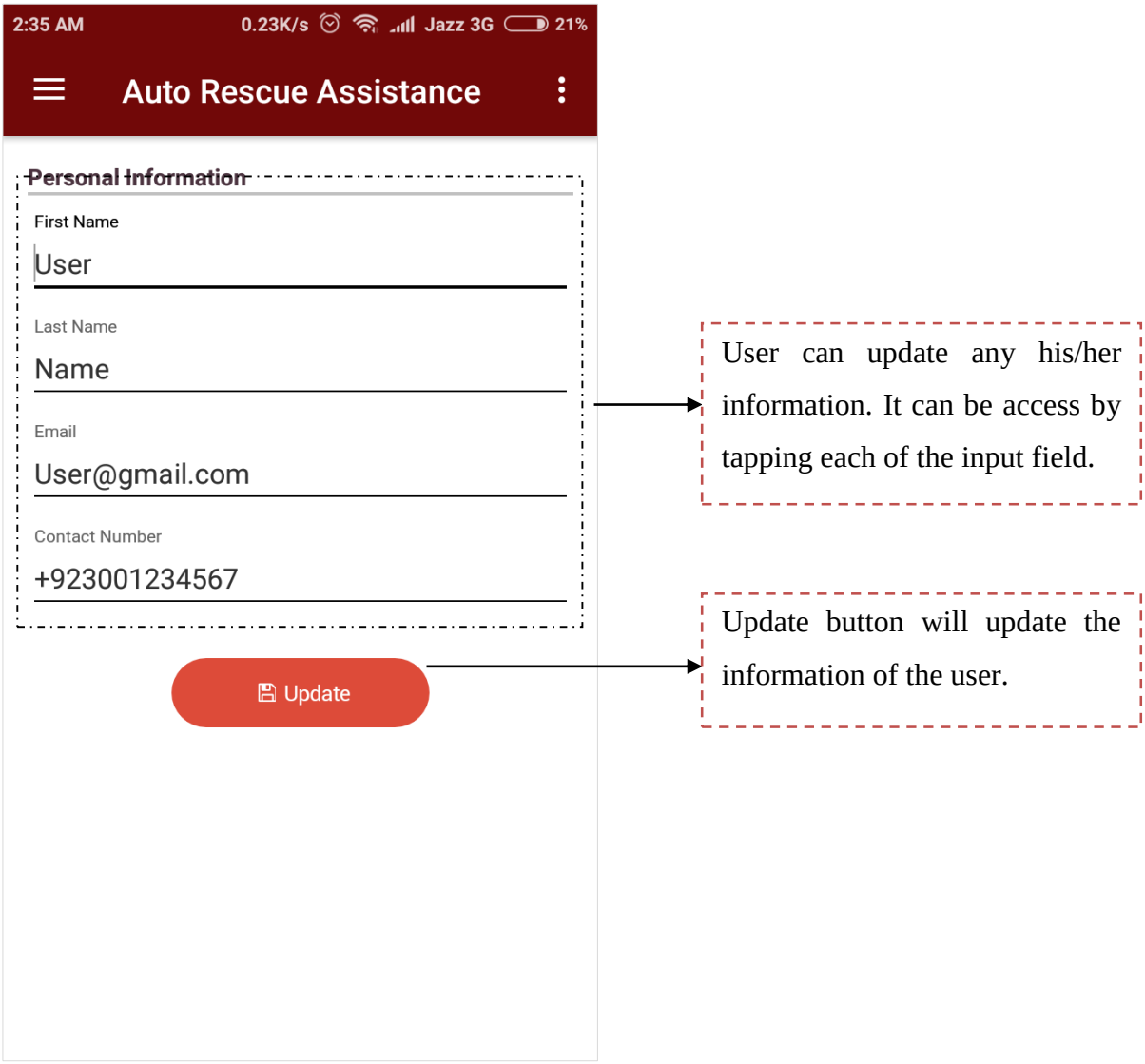
5.1.3.2 **Adding Emergency Contact**

User can add emergency contact to the list. Emergency contacts will receive a message with the user's location whenever use is in an emergency. User can add or view emergency contact in SOS Contact page. It can be called by tapping "SOS contact" menu in navigation drawer.

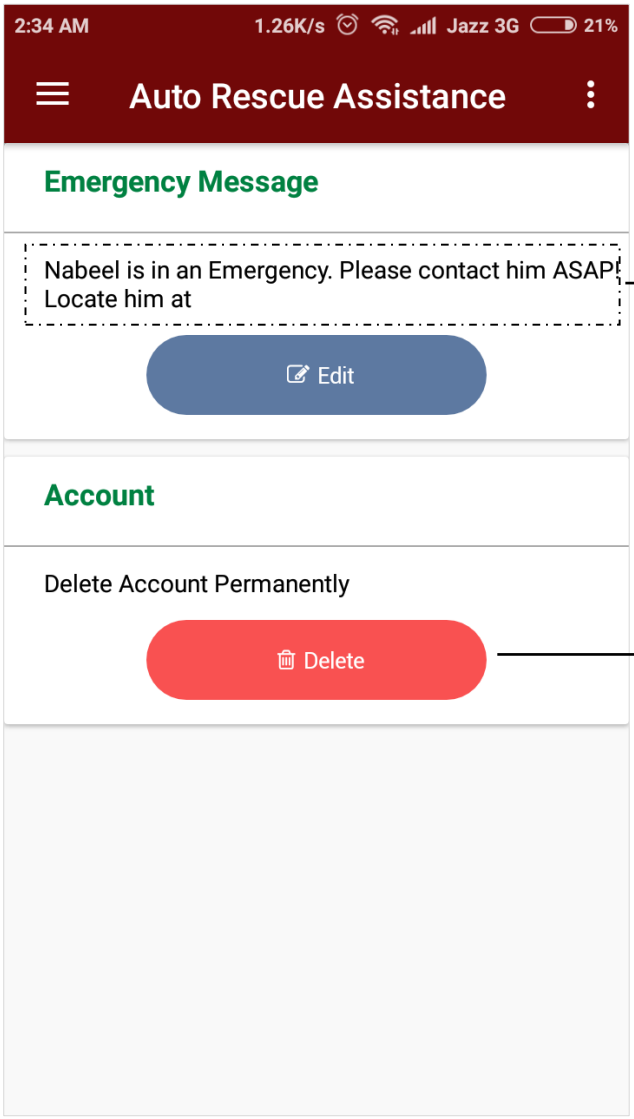


5.1.3.3 **Profile Management**

User can update his/her profile in profile page. Profile page can be open via navigation drawer.



5.1.3.4 **Settings**



It is the message that will send to the Emergency contact in case of emergency. It can be edit by tapping 'edit' button

User can delete his/her account by tapping delete button. Account will be deleted permanently from the system.

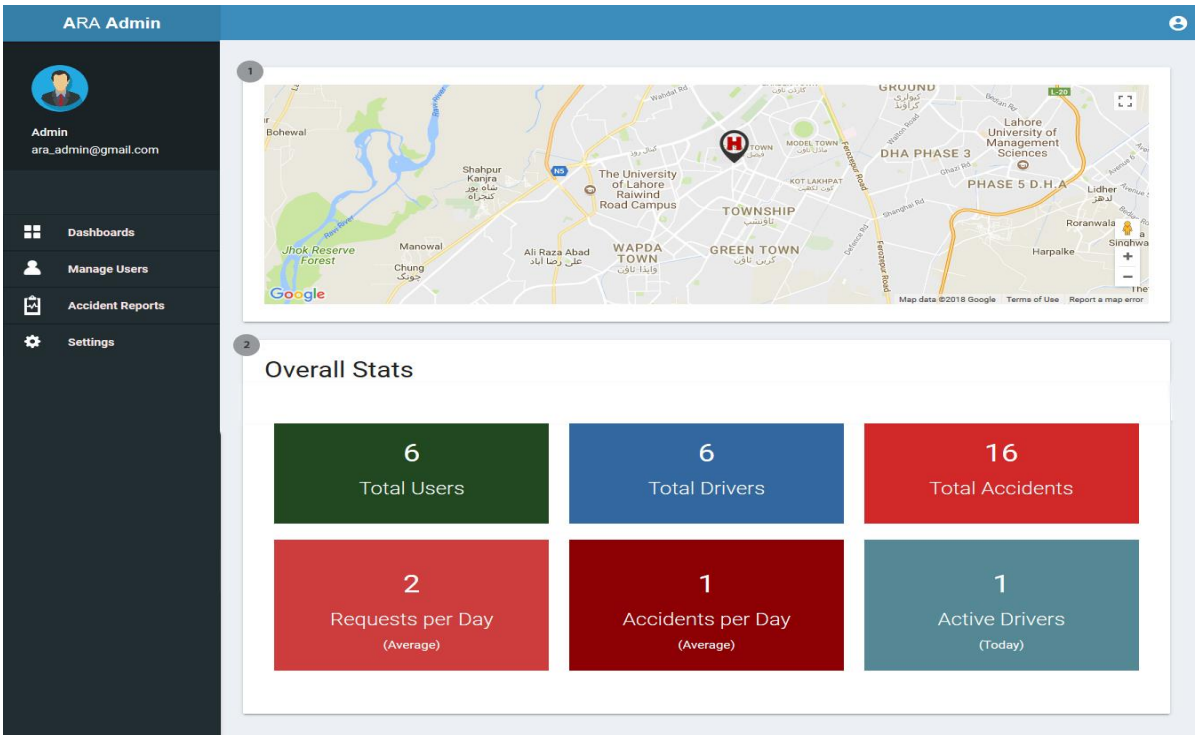
5.2 Web Application (Admin Panel)

5.2.1 Log in Page

Admin can access admin panel with his login credentials. Login credentials will be provided by the developers for the first-time use.

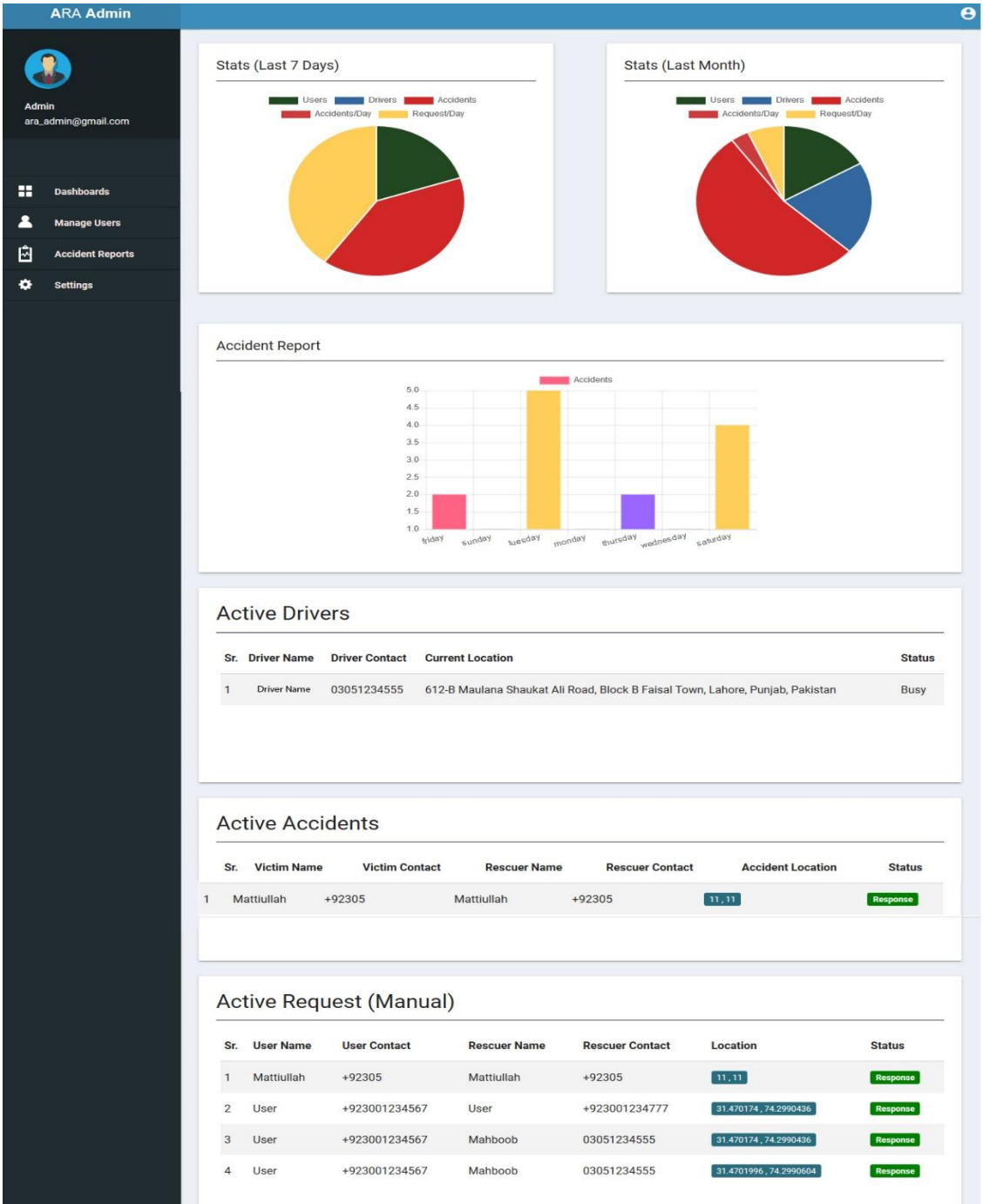
5.2.2 **Admin Panel Home Screen**

Admin panel contain the information of the current accident. Overall stats as described below in figures. Admin can view the active drivers and their location. Admin also has access to view the active accidents and their respective location.



1. It is integrated with the google map, that shows the location according to the given latitude and longitude. Map view displays the location of current activated accidents. Admin can track each accident on the map.
2. It displays all the overall stats. Overall stats include
    a. Total number of all the users.
    b. Total number of all the drivers.
    c. Total accidents that has occurred till date.
    d. Occurrence of manual request per day (average)
    e. Occurrence of automatic accidents per day (average)
    f. Total drivers that are active.

- Admin can also view the overall stats in the form of chart. Screen below shows the three charts that are Stats of the Last 7 days and Last month that are represent in pie chart.
- Bar char represents the accident on daily basis.
- Admin can also track the active drivers; active driver table contains the information of the driver. His location, name and contact of the driver.
- Active accidents can also be seen on admin portal.



5.3 Rescuer Android Application

5.3.1 Getting Started

5.3.1.1 Registration:

Driver can be register via admin panel. Admin has rights to add drivers manually via admin panel. Drivers will be provided with the email and password for login credentials.

5.3.1.2 Login

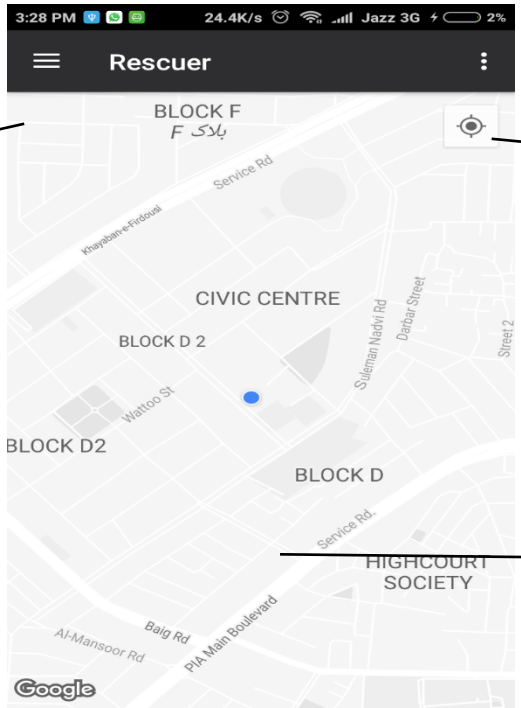
Driver can login to Rescuer app by using his login credentials.

5.3.2 Application Overview

After successfully log in, application main/home screen will be launch. It will contain integrated google map with driver's current location as shown in fig and other functionality as described below in detail.

5.3.2.1 Home Screen

Home screen is the main page of the application. Current location button and the integrated google map with user's current location.



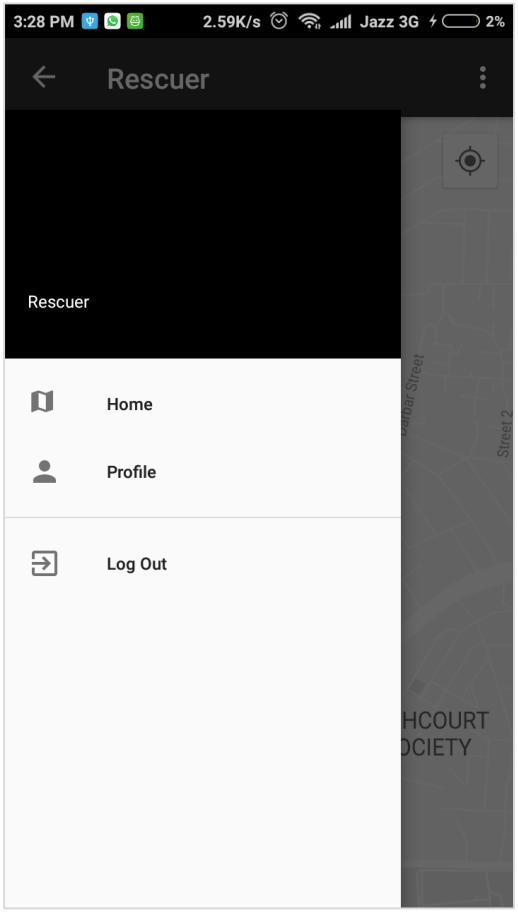
Opens the Application Drawer Menu as shown in article 5.3.2.2

It will move the map to driver's current location.

Current location of driver

5.3.2.2 **App Navigation Drawer**

The navigation drawer is a UI panel that shows app's main navigation menu. It is hidden when not in use but appears when the user swipes a finger from the left edge of the screen or, when at the top level of the app, the user touches the drawer icon in the app bar.

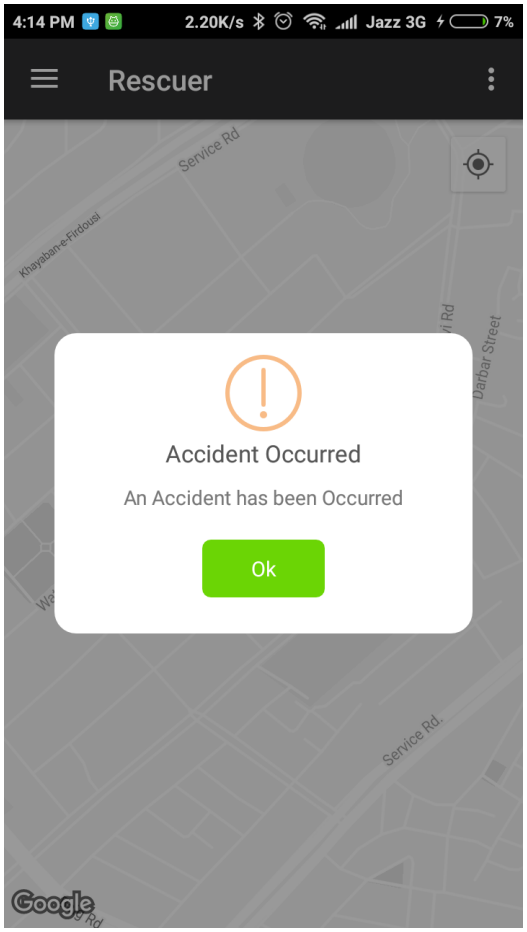


1. It will display the information of currently logged in driver.
2. User will logout from the device.

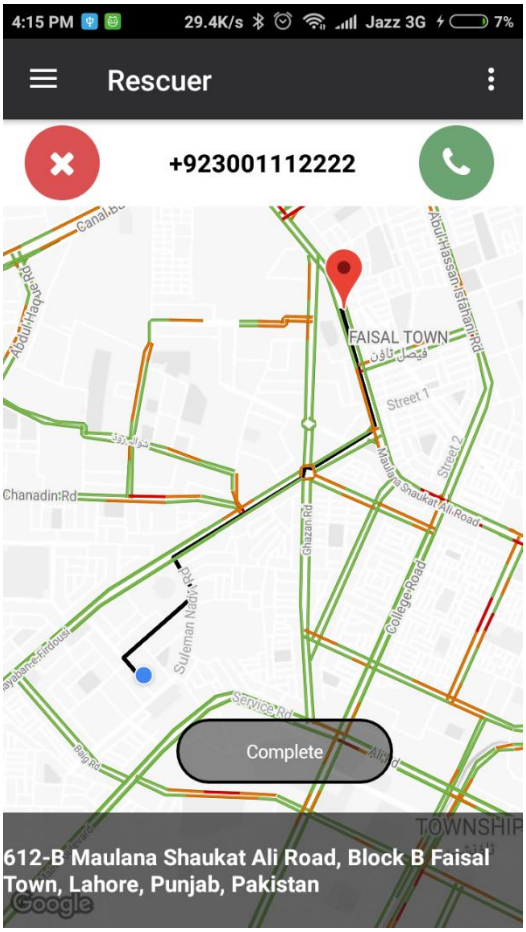
5.3.3 Application Functionality

5.3.3.1 Getting an Emergency Request

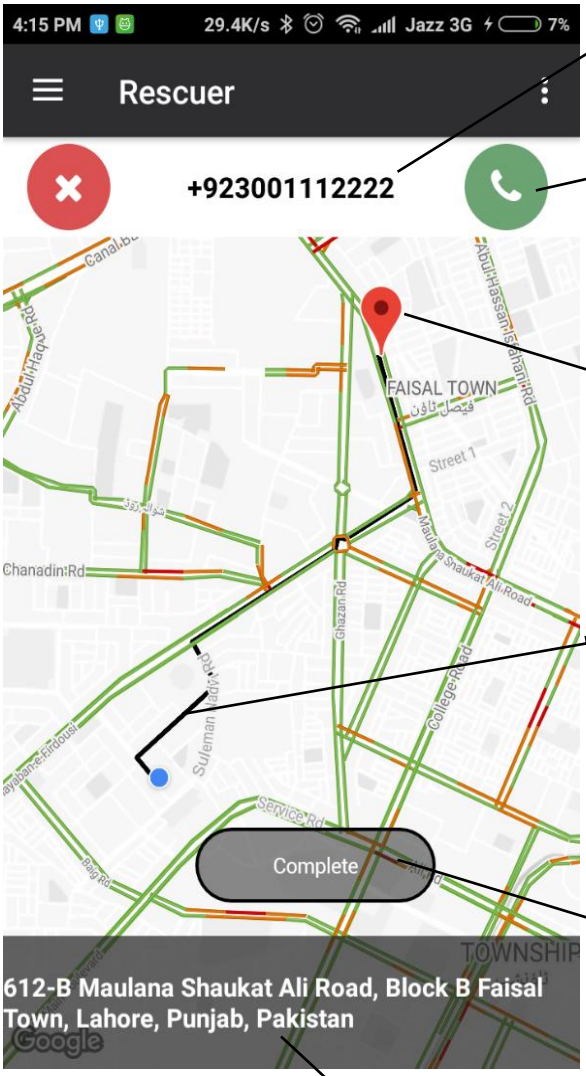
Driver will receive an emergency request from the victim. Emergency request will contain the information about the victim such as Contact Number, victim's location



When user request for an emergency, dialogue box will be pop up on driver's side.



Driver will receive a request that contain information of the user (location, contact number)



Victim Contact number.

Driver can call user by tapping call button.

Victim's current location

Path between the user and the victim

Driver can press the button on completion of the service

Address of the victim' location

CHAPTER 6

Test Cases

Login Test Case

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC_01	Check User's Login with valid contact and password	1) Open the application. 2) Grant the required permissions by the application. 3) Select the "Login" button. 4) Enter Contact Number. 5) Enter Password. 6) Click Login button.	Contact#=+92058165462, Password=123456	User should be login to the application.	As expected	Pass
TC_02	Check User's Login with invalid contact and password	1) Open the application. 2) Grant the required permissions by the application. 3) Select the "Login" button. 4) Enter Contact Number. 5) Enter Password. 6) Click Login button.	Contact#=03231234567, Password=124	User should not be login to the application.	As expected	Pass
TC_03	Check User's Login with	1) Open the application. 2) Grant the required permissions by the application.	Contact#='', Password=''	User should not be login to	As expected	Pass

	empty fields	3) Select the "Login" button. 4) Leave all the fields empty. 5) Click Login button.		the application.		

Nearby Hospitals

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
TC_04	Search Nearby Hospitals	1) Open the application. 2) Internet and GPS must be turned on. 3) Click the Nearby Hospitals option.	User should be able to see the nearby hospitals in the map with the markers on it.	As expected	Pass
TC_05	Search Nearby Hospitals	1) Open the application. 2) Internet and GPS must be turned off. 3) Click the Nearby Hospitals option.	User should not be able to see the nearby hospitals in the map.	As Expected	Pass

Add SOS Contact

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
TC_06	Adding contacts to SOS list	1) Login to the application after providing the valid user information. 2) Select the option Add SOS contact. 3) Tap the + button to add the contacts from the Phone memory. 4) Select the required contact.	The contact must be added to SOS list and must be visible to the user in the view.	As expected	Pass

Delete SOS Contacts

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Status
TC_07	Delete Contact from SOS list	1) Login to the application after providing the valid user information. 2) Select the option Add SOS contact. 3) Tap the bin button to delete the contacts from the SOS list	The contact must be deleted from the SOS list and must be visible to the user in the view.	As expected	Pass

Update Profile

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC_08	Update User profile	Login to the application after providing the valid user information. Grant all the permissions required by the application. Select the option Update User Profile. Fill the field which needs to be change. Tap on the Update button.		The User profile must be updated and changes should be visible in the user profile.	As expected	Pass

Assistance Button

Test Case ID	Test Scenario	Test Steps	Pre-Conditions	Expected Result	Actual Result	Status
TC_09	Press Assistance Button	1) Login to the application after providing the valid user information. 2) Press Assistance Button.	GPS and Internet must be enabled.	Request should be successfully triggered. Information of Rescuer displayed on the screen.	As expected	Pass

Automatic Accident Request

Test Case ID	Test Scenario	Test Steps	Pre-Conditions	Expected Result	Actual Result	Status
TC_10	Automatic Accident Request.	1) Login to the application after providing the valid user information.	GPS, Bluetooth and Internet must be enabled. Speed must be less than or equal the 1/3 of speed Exerted pressure greater than 300.	Request should be successfully triggered. Information of Rescuer displayed on the screen.	As expected	Pass

Request Response

Test Case ID	Test Scenario	Test Steps	Pre-Conditions	Expected Result	Actual Result	Status
TC_11	Driver response on victim's request.	1) Login to the application after providing the valid user information.	GPS and Internet must be enabled.	Request should be successfully triggered. Information of Victim displayed on the screen.	As expected	Pass

Accident Tracking

Test Case ID	Test Scenario	Test Steps	Pre-Conditions	Expected Result	Actual Result	Status
TC_12	Accident Tracking	1) Login to the web portal after providing the valid user information.		Map and tables will show the accident information on web portal.	As expected	Pass

Report View

Test Case ID	Test Scenario	Test Steps	Pre-Conditions	Expected Result	Actual Result	Status
TC_13	Report view	1) Login to the web portal after providing the valid user information.		Pie charts and bar chart will show the accidents summary on web portal.	As expected	Pass

SIGN UP

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Result	Actual Result	Status
TC_14	Check User's Registration with valid fields	1) Open the application. 2) Grant the required permissions by the application. 3) Select the "Sign Up" button. 4) Enter required fields 5) Click Sign Up button.	Name = User Name, Email= username@gmail.com, Password = 123456 Confirm Passowrd = 123456 Contact # = +923001234567 Blood Group = AB+	User should be registered to the application. Redirect to Main Page	As expected	Pass
TC_15	Check User's Sign Up with invalid fields	1) Open the application. 2) Grant the required permissions by the application. 3) Select the "Sign Up" button. 4) Fill the Fields 5) Click Create Account button.	Name = User, Email = username@gmail, Password = 123 Confirm Passowrd = 12345 Contact # = 03001234567 Blood Group = AB+	User should not be registered to the application. Error Message must be generated.	As expected	Pass
TC_16	Check User's Sign Up with empty fields	1) Open the application. 2) Grant the required permissions by the application. 3) Select the "Sign up" button. 4) Leave some fields empty. 5) Click Creat Account button.	Name = User Name, Email = ‘’, Password = ‘’ Confirm Passowrd = ‘’ Contact # = +923051234567 Blood Group = A+	User should not be Registered to the application. Error Message must be generated.	As expected	Pass

CHAPTER 7

Conclusion and Recommendation

7.1 CONCLUSION

Speed is one of the most significant causes of an accident. Nowadays, GPS has become an integral part of a smart phones. So, the GPS of smartphones can be used to monitor the speed and detect an accident based on some threshold value. The sensor will detect the pressure from the external environment to minimize the false request, so the accident occurrence can be detected using speed monitoring and sensor. Once the situation, considered as an accident then application will immediately trace nearest ambulance driver for the first aid services and send emergency alert message with the current location of the user to the SOS contacts for corrective measures. Beside the automatic detection system, the vehicle occupant will be able to manually send the accident situation by tapping the Assistance Button. A rescue measures in time with sufficient preparation at the correct place can save many lives. Thus, the application can serve the humanity by a great deal as human life is valuable. Hence, the application would play an important role in post-accident services and could lessen the effect due to accident remarkably.

7.2 RECOMMENDATION

7.2.1. Streamline flow of Ambulance:

At the accident time during some emergency situations ambulance may blocked in the signal it leads to major cause. To avoid this, based on all statistics, traffic signal should be controlled. For that purpose, there is some mechanism which could resolve this problem. IOT would play the role between ambulance and the traffic signals. Cloud computing provides the way for handling and managing the enormous amount of data that are

generated by these devices and it would be even used to send command to those devices to perform a task. This application will be based on the IOT and cloud to save the human life at critical situation. This application should be enhanced to establish the communication between the traffic signals and the ambulance so that the traffic signal can respond to the arrival of the ambulance and respond according to that. When the traffic signals are changes its states according to the position of the ambulance it can able to make a free way for the ambulance. Thus, the application will act as a life saver.

7.2.2. Integration with Multiple Sensors:

Multiple sensors will be integrated to detect the collision from all the possible sides of the car.

REFERENCES

Website

[1]““WhatIsAndroidIntroduction.” *Engineersgarage* ,https://www.engineersgarage.com/articles/what-is-android-introduction.

[2] pusher-http-node.” *github* , https://github.com/pusher/pusher-http-node.

[3] io angular.” *angular* , https://angular.io/docs

[4] docs postgresql.” *postgresql* , https://www.postgresql.org/docs/

[5] nodejs.” *nodejs*, https://nodejs.org/en/docs/