**Cairo University**

**Faculty of Computers and Artificial Intelligence**

**Graduation Project**

**CS498**

**Project: Monqez Application**

|  |  |
| --- | --- |
| ID | Name |
| 20170093 | حسين أشرف عبد الحميد |
| 20170072 | إيهاب فوزي ابراهيم |
| 20170085 | حاتم ممدوح سعيد |
| 20170098 | خالد عزت احمد |

Supervised by: Dr. Mohamed El-Ramly

Acknowledgement

(in final doc. only -> optional)

Abstract

Monqez is an application that aims to help people in need. In this document we will start by an introduction explaining our project, why we chose it, the main problem, and our approach in solving it. The introduction includes the project time plan, our project development methodology, and tools used. The second chapter states some related works and mentions pros and cons and how they differ from our project. The third chapter is system analysis where we discuss the functional and non-functional requirements, use case diagram, access privileges, and sequence diagrams. The fourth chapter shows our application entity relationship diagram, class diagram, and the initial graphical user interface design. We then discuss our implementation methods, testing and their results. Finally, we reach a conclusion about our application.

Table of Contents

|  |  |
| --- | --- |
| **Chapter 1: Introduction** | # |
| 1.1. Background | # |
| 1.2. Motivation | # |
| 1.3. Problem Definition | # |
| 1.4. Suggested Solution | # |
| 1.5. Project Time Plan | # |
| 1.6. Project Development Methodology | # |
| 1.7. Tools | # |
| 1.8. Report Organization | # |
| **Chapter 2: Related Work** | # |
| **Chapter 3: System Analysis** | # |
| 3.1. Project Specification | # |
| 3.1.1. Functional Requirements | # |
| 3.1.2. Non-functional Requirements | # |
| 3.2. Use Case Diagram and Use Cases | # |
| 3.3. User Groups and Access Privileges | # |
| 3.4. Sequence Diagrams (can be moved to Design) | # |
| 3.5. System Test Cases (can be moved to Design) | # |
| **Chapter 4: System Design** | # |
| 4.1. System Component Diagram | # |
| 4.2. Application ERD | # |
| 4.3. Class Diagram | # |
| 4.4. Elementary Design (optional) | # |
| 4.5. GUI Design | # |
| **Chapter 5: Implementation and Testing** | # |
| 5.1. Methods | # |
| 5.2. Results | # |
| **Chapter 6: Conclusion** | # |
| **Chapter 7: Future Work (optional / if needed)** | # |
| **References** | # |

List of Figures

|  |  |
| --- | --- |
| **Figure 1:** Gantt chart from 4th of August to 29th of October | # |
| **Figure 2:** Gantt chart from 29th of October to 16th of November | # |
| **Figure 3:** Gantt chart from 14th of November to 27th of November |  |
| **Figure 4:** Gantt chart from 19th of November to 27th of November |  |
| **Figure 5:** Gantt chart from 10th of March to 27th of April |  |
| **Figure 6:** Use Case Diagram |  |
| **Figure 7:** “Sign up as Monqez” Sequence Diagram |  |
| **Figure 8:** “Sign in with Google” Sequence Diagram |  |
| **Figure 9:** “Add new Admin” Sequence Diagram |  |
| **Figure 10:** “Accept or Decline Monqez Application” Sequence Diagram |  |
| **Figure 11:** Application Entity Relationship Diagram |  |
| **Figure 12:** System Class Diagram |  |
| **Figure 13:** Register Normal User Screen |  |
| **Figure 14:** Login Screen |  |
| **Figure 15:** Normal User Home Screen |  |
| **Figure 16:** Request’s Additional Information Popup |  |
| **Figure 17:** Normal User’s Navigation Drawer |  |
| **Figure 18:** Chatbot Screen |  |
| **Figure 19:** Monqez Home Screen |  |
| **Figure 20:** Monqez Request Notification |  |
| **Figure 21:** Monqez Navigation Screen |  |
| **Figure 22:** Monqez Display Request’s Additional Information Popup |  |

List of Tables

|  |  |
| --- | --- |
| **Table 1:** Use Case 1 | # |
| **Table 2:** Use Case 2 | # |
| **Table 3:** Use Case 3 |  |
| **Table 4:** Use Case 4 |  |
| **Table 5:** Use Case 5 |  |
| **Table 6:** Use Case 6 |  |
| **Table 7:** Use Case 7 |  |
| **Table 8:** Use Case 8 |  |

**List of Acronyms & Abbreviations**

|  |  |
| --- | --- |
| **API** | Stands for Application Programming Interface which is a computing interface that defines interactions between multiple software intermediaries. |
| **App** | Stands for Application. |
| **EMS** | Stands for Emergency Medical Services. |
| **ETA** | Stands for Estimated Time of Arrival. |
| **IDE** | Stands for Integrated Development Environment which is a software application that provides comprehensive facilities to computer programmers for software development. |
| **Monqez** | The Arabic word for “savior” who, in our system, is a paramedic / helper that arrives to help the person calling for help. |
| **MVC** | MVC is an architectural pattern which stands for Model, View, and Controller. MVC separates an application into three components - Model, View, and Controller. |
| **GPS** | Global Positioning System is a global navigation satellite system that provides location, velocity, and time synchronization. |

Chapter 1: Introduction

* 1. Background:

People die before getting the first-aid they need when involved in an accident. For several years, people tried to solve this problem by calling for an ambulance or if the person in-need is lucky there might be a specialized person nearby. Our approach for solving this problem is making a video or voice call with a qualified specialist (Monqez) or requesting any nearby specialist to help give the first-aid needed.

To help connect the person in-need with the Monqez we need to use video conferencing technology which uses the camera in a smart phone. This technology needs about 2 mega bit per second. This needs a stable internet connection. Location tracking is used when the Monqez is reaching the person in-need, it helps him find the patient faster as well as it helps the patient request an on-site Monqez faster. The location tracking can be implemented only if the smartphone has a Global Positioning System (GPS).

* 1. Motivation:

On April 12, 2010, The Guardian stated that over 150,000 deaths could have been saved if the right first aid was applied to them [1]. Moreover, the University of Manchester reported that 59% of deaths that occurred before arriving to the hospital could have been saved [2]. With the rapid advances in mobile app development, a few applications were developed that aim at handling emergencies that require medical attention. However, these applications have their shortcomings and rely on hospitals to save the victims rather than offer on-site first aid.

* 1. Problem Definition:

Accidents occur every day all around the world and a lot of people fall victim to them. It could take some time for an ambulance to arrive even though every minute is crucial to the victim. That’s why first aid is vital for saving lives! Although 93% of people will call for an ambulance if they find someone with an injury, first aid intervention of any kind is infrequent [2]. This partly because many people lack the basic first aid knowledge and partly because sometimes the situation requires a trained professional to handle the injuries. So there lies the problem; how can we help people who just witnessed an accident save the victim’s life?

It’s unquestionable that we need a fast and easy method of communication with paramedics and what better way to have this communication than to have a mobile app that notifies paramedics and connects us with them through voice and video calls? Some applications that provide EMS exist and have gained popularity. However, not all these applications are cross platform or free. Furthermore, they don’t notify the nearest first-aid-certified person to the accident’s location or offer video calls with paramedics or just provide contact with a first aid chatbot. Even the applications that do exist offering these features target certain emergencies only (CPR emergencies only for instance).

Suggested Solution:

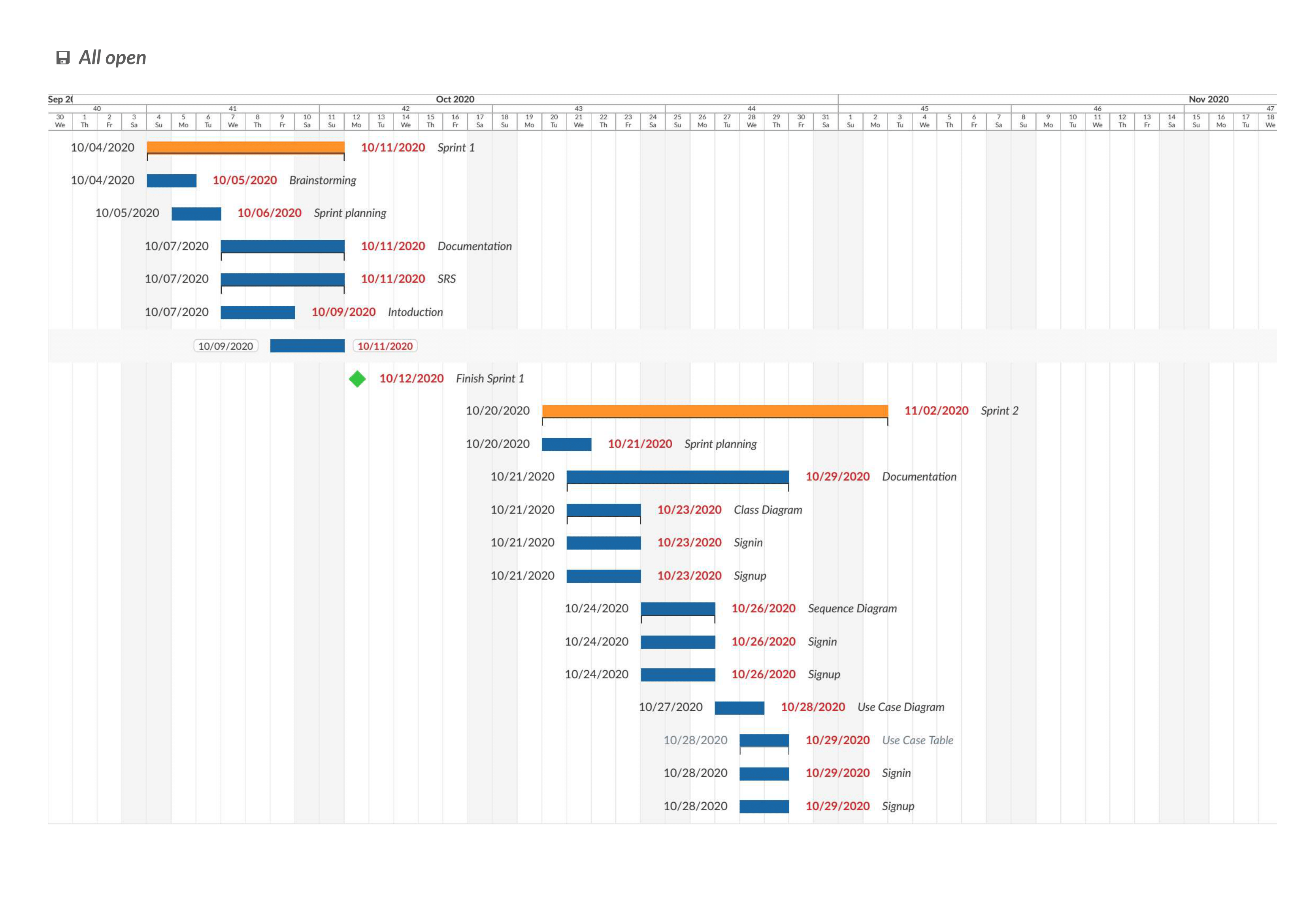
We propose developing a cross platform mobile application called “Monqez” which helps people who need first aid by informing the closest available paramedics to the patient’s location in order to help the patient and save his/her life.

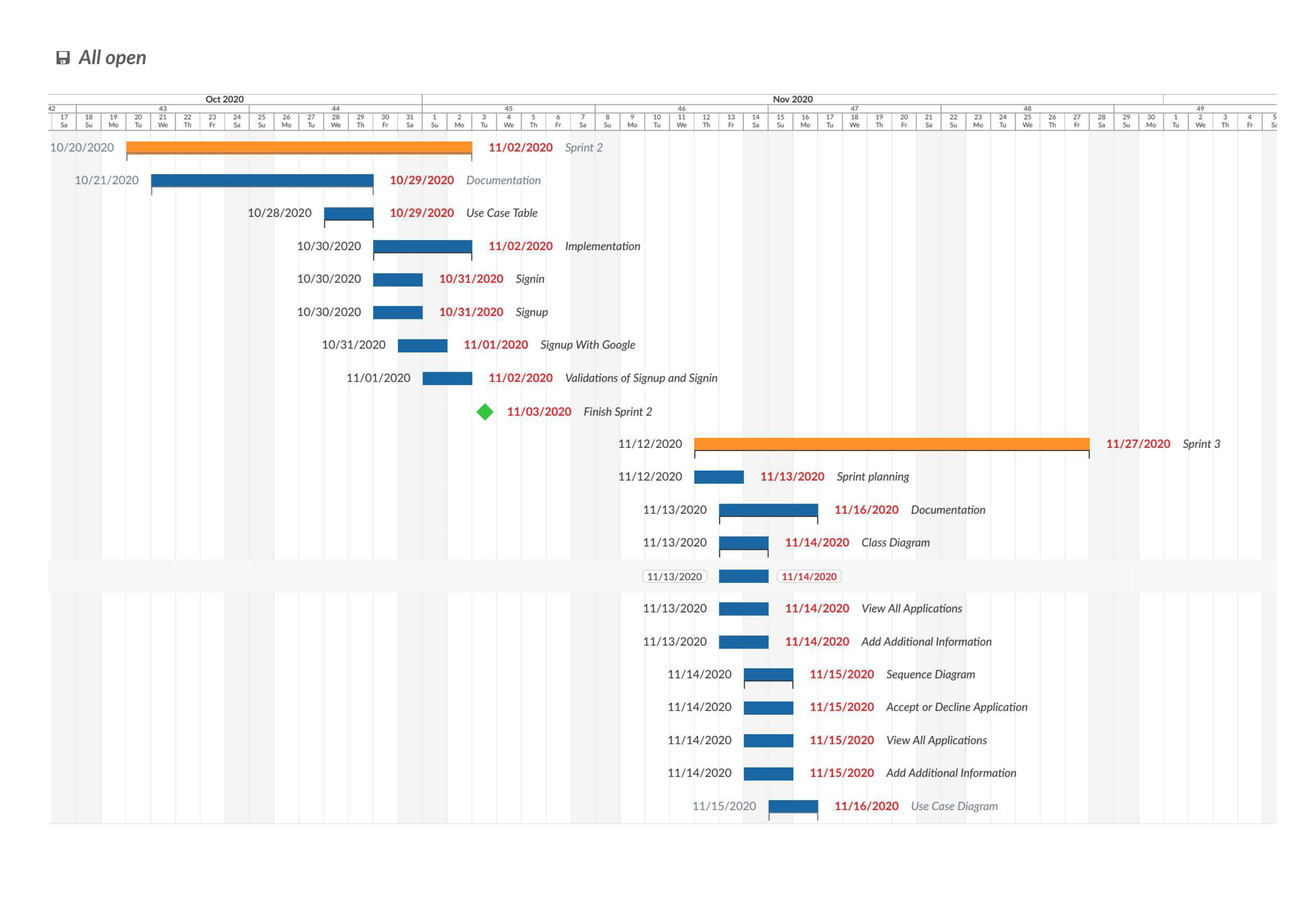
The main goal of Monqez is to provide quick help to the person in need until an ambulance arrives. In order to achieve this goal, the app must be able to connect the user requesting help to the nearest available paramedic “Monqez” as well as provide some basic first aid tips either via pictures or voice and video calls.

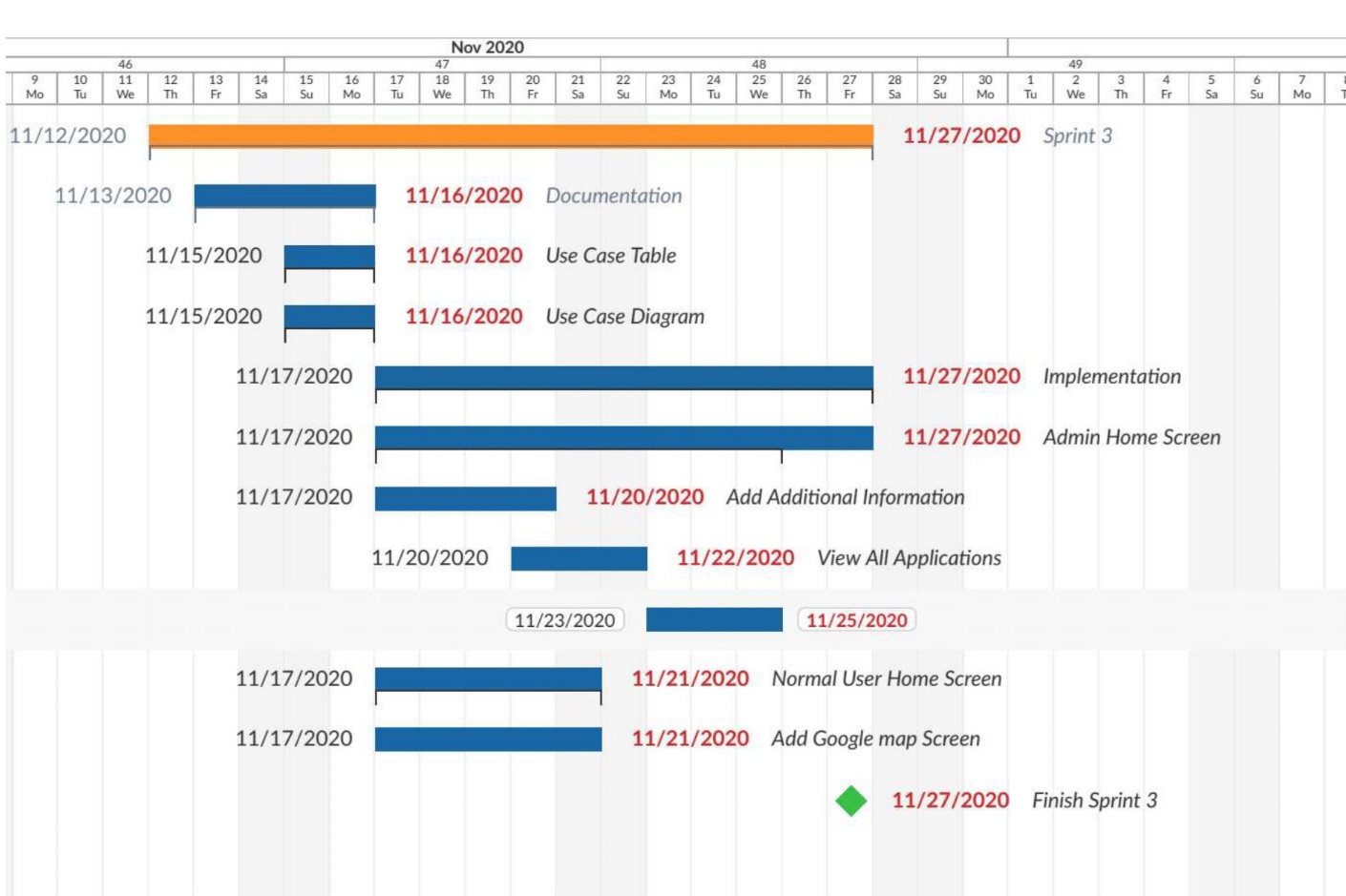
Of course, all registered paramedics must be certified and need to have passed a first aid course from accredited locations.

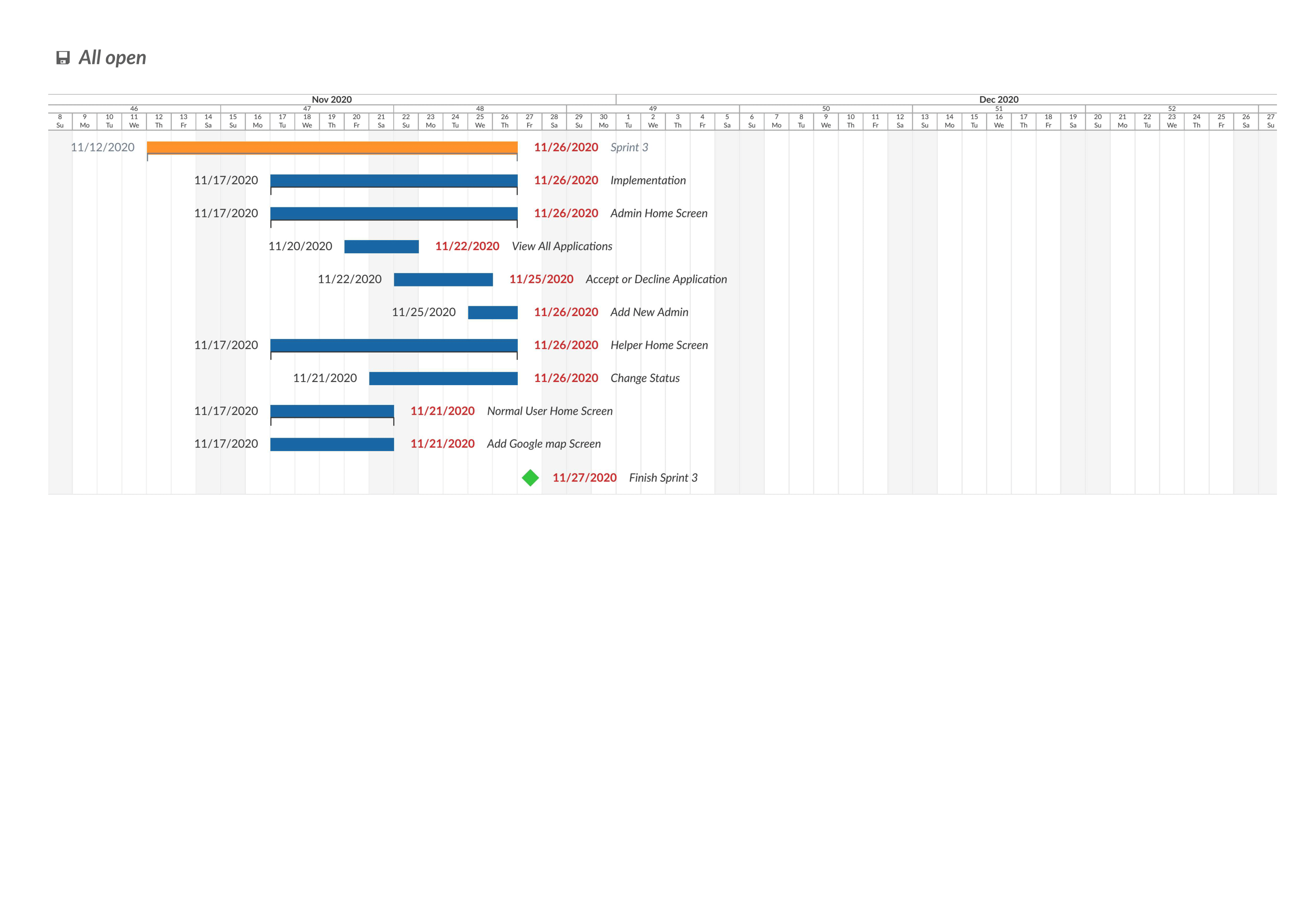
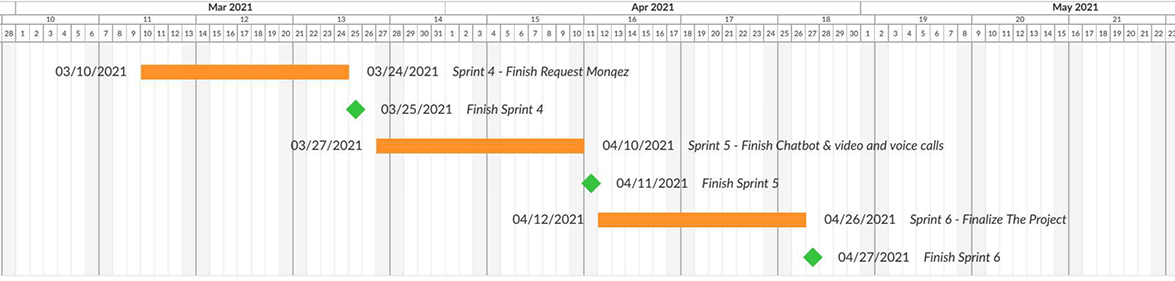
* 1. Project Time Plan:

***The following figures show the time plan we will follow to develop Monqez application:***

 ***Figure 1:*** Gantt chart from 4th of October to 29th of October

 ***Figure 2:*** Gantt chart from 29th of October to 16th of November

*** Figure 3:*** Gantt chart from 16th of November to 27th of November

 ***Figure 4:*** Gantt chart from 19th of November to 27th of November

***Figure 5:*** Gantt chart from 10th of March to 27th of April

* 1. Project Development Methodology:

Agile Development:

Agile is a term used to describe approaches to software development emphasizing incremental delivery, team collaboration, continual planning, and continual learning, instead of trying to deliver it all at once near the end.

Agile focuses on keeping the process lean and creating minimum viable products (MVPs) that go through a number of iterations before anything is final. Feedback is gathered and implemented continually and in all, it is a much more dynamic process where everyone is working together towards one goal.

Feature Driven Development (FDD):

Feature Driven Development (FDD) is an agile framework that, as its name suggests, organizes software development around making progress on features. Features in the FDD context, though, are not necessarily product features in the commonly understood sense. They are, rather, more akin to user stories in Scrum. In other words, “complete the login process” might be considered a feature in the Feature Driven Development (FDD) methodology.

Why we used Feature Driven Development (FDD)?

1. Gives the team a very good understanding of the project’s scope and context.
2. Breaks feature sets into smaller chunks and regular iterative releases, which makes it easier to track and fix coding errors, reduces risk, and allows us to make a quick turnaround to meet our project’s needs.
3. Requires fewer meetings. One of the frequent complaints about agile is that there are too many meetings. Scrum uses the daily meetings to communicate. FDD uses documentation to communicate
   1. Tools:
4. Node.JS
   * + Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux. Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.
5. Dart Programming language
   * + Dart is a client-optimized programming language for apps on multiple platforms. It is developed by Google and is used to build mobile, desktop, server, and web applications. Dart is an object-oriented, class-based, garbage-collected language with C-style syntax. Dart can compile to either native code or JavaScript. It supports interfaces, abstract classes, reified generics, and type inference.
6. Flutter
   * + Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, it allows you to create a native mobile application with only one codebase. This means that you can use one programming language and one codebase to create two different apps (for iOS and Android).

Flutter consists of two important parts:

* + - 1. An SDK (Software Development Kit): A collection of tools that are going to help you develop your applications. This includes tools to compile your code into native machine code (code for iOS and Android).
      2. A Framework (UI Library based on widgets): A collection of reusable UI elements (buttons, text inputs, sliders, and so on) that you can personalize for your own needs.

Flutter uses Dart as a programming language.

1. Firebase
   * + Firebase is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google’s infrastructure. Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents
2. Google Maps Platform
   * + The Google Maps Platform is a set of APIs and SDKs that allows developers to embed Google Maps into mobile apps and web pages, or to retrieve data from Google Maps
   1. Report Organization:

The next chapters will cover the design and implementation aspects of Monqez. First, chapter 2 will provide an overview of some similar applications. Then, chapters 3 and 4 will dive into the design of Monqez application. Chapter 3 details all the functional and quality requirements of Monqez while chapter 4 contains specific information about the expected input, output, classes and functions. In addition, the interactions between classes to meet the desired requirements will be outlined in several figures in chapter 4. Chapter 5 will talk about how the application was implemented and tested. It will also present some of the results obtained from the testing process. Chapter 6 will include a discussion of what has been achieved so far as well as any concluding remarks. Finally, chapter 7 will discuss the future plans we have for Monqez to make it the perfect choice for users.

The intended reader groups for this document are the project manager, developers, supervising professor and any person interested in developing an application like Monqez.

Chapter 2: Related Work

As we mentioned before, a few applications exist that aim at handling emergencies which require medical attention. Each one of these apps offers a variety of features. We surveyed these apps and determined their advantages and disadvantages from a user’s perspective. In the next few paragraphs, we will discuss some of these apps in more detail.

* **New Zealand Red Cross First Aid [3]:** This web application contains courses and step-by-step instructions for many emergencies including CPR, burns, heart attacks, open wounds and more. However, it doesn't send medic to help the victim right away; it just relies on normal users to understand and follow the first aid instructions shown.
* **E-Medic [4]:** This application allows receiving any new communication to report an accident as well as accepting or canceling the report, drawing the best paths to the location of the incident and taking notes. In addition, it allows sending any amendments to the report to the operating room so that the operating room is aware of the emergency teams and following them on an ongoing basis. The application also allows emergency teams to communicate with hospitals and choose the most appropriate hospital for the case. The only drawback we found is that it does not send a paramedic or help the user perform the first aid right away; it just sends an ambulance to take the victim to the hospital.
* **eMedic (for iPhones) [5]:** It is an EMS app for iPhones. Used by more than 20,000 paramedics, EMTs, and nurses, eMedic provides quick access to basic anatomical illustrations, medical calculators, emergency quick reference cards, ACLS algorithms, popular medicines, and medical acronyms. Unfortunately, eMedic is not a free app and it only targets users with a strong medical background.
* **PulsePoint Respond [6]:** It is a 911-connected app that can immediately inform users of emergencies occurring in their community and can request their help when CPR is needed nearby. This app helps create an informed and engaged community that drives a “Culture of Action,” a key strategy in strengthening the Chain of Survival for cardiac arrest victims. In addition to nearby “CPR-needed” notifications, the user can choose to be notified of significant events that may impact him/her and his/her family.
* **Asifny - إسعفني [7]:** It is an application developed by Saudi Red Crescent Authority which includes the following features:

1- Opening an emergency communication with the Saudi Red Crescent Authority and increasing the accuracy of the location.

2- Send urgent distress in the extreme emergencies of both the Red Crescent and people close to you by SMS service.

**3**- Record the details of your medical history, the diseases you suffer and the medicines you use to get to know your condition as soon as possible.

**4**- Inform you of the medical facilities close to you such as hospitals, dispensaries and pharmacies with your directions and chart the route on the map of the facility you want to go to.

This application should send an ambulance - not anyone with a first-aid certificate - to any patient in order to save his life. This makes the process slower to reach the patient.

* **Inaa’sh [8]:** It is combined first aid and CPR Accreditation Program made especially for government and commercial organizations as well as community at large. It provides an up-to-date first aid and CPR knowledge and skills information through certified courses and also provide the require equipment for organizations which helps perform CPR and 1st aid procedures during emergencies. It doesn’t provide first-aider to the person in need’s location nor does it provide online help that the person can contact when in need.

Chapter 3: System Analysis

* 1. Project Specification:
     1. Functional Requirements:

***The following are the functional requirements that the system should be able to do:***

1. Allow the user to create a normal or Monqez account in the database.
2. Log the user into his/her account and allow the user to edit his/her information and to log out.
3. Show the first aid instructions for any type of injury.
4. Allow the normal user to order a Monqez for himself or for someone else in need of first aid.
5. Notify all the nearby Monqez users when a first aid request is received.
6. Connect the normal user to a Monqez so that the Monqez shall arrive to the emergency site.
7. Track and show the current location of the Monqez who accepted the request.
8. Provide video and voice calls between a normal user and any available Monqez.
9. Allow normal users to add friends and family members in order to quicken the process of requesting a Monqez in case of an emergency.
10. Notify the normal user’s emergency list when an emergency happens to the user.
11. Allow both the normal user and the Monqez user to make complaints about any incident that happened during the process of fulfilling a request.
12. Allow the normal user to rate the Monqez that has fulfilled his request.
13. Show the Monqez rating to the normal user.
14. Allow an admin user to add another admin.
15. Allow an admin user to either approve or deny a Monqez application.
16. Show the complaints about a specific request to an admin.
17. Show the Monqez applications and their certificates to an admin.
18. Allow an admin to ban any account he/she finds inappropriate.
    * 1. Non-functional Requirements:

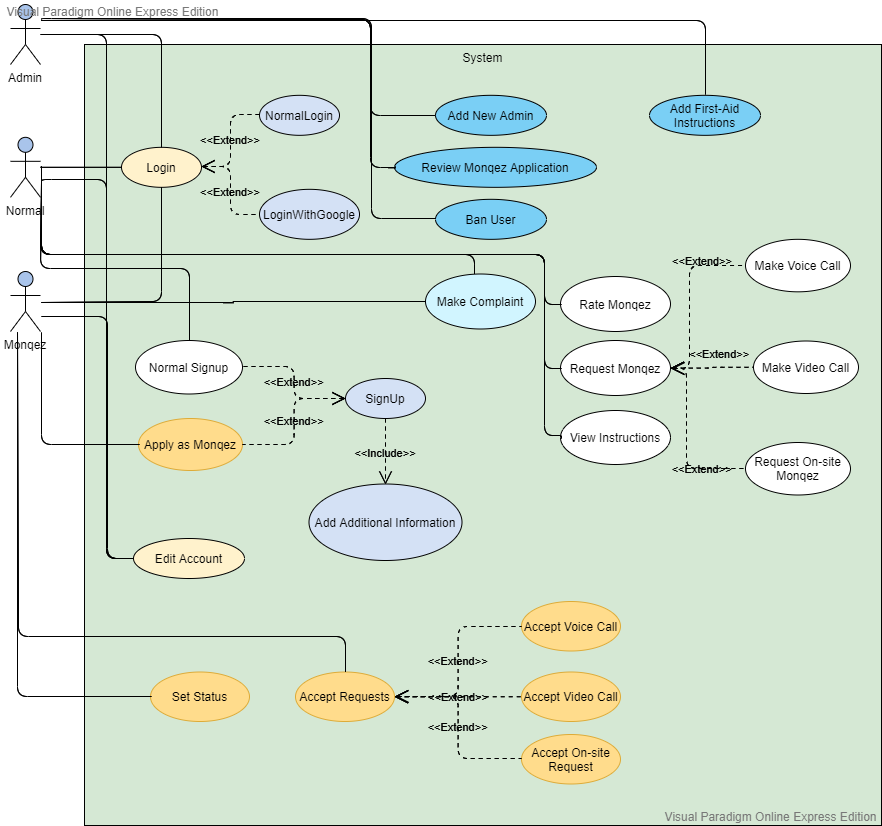
***The following are the quality requirements that the system should be able to achieve:***

1. **Performance:** The normal user request should be matched as quickly as possible (at most 2 minutes) since the duration is an important factor to save the patients’ lives.
2. **Safety:** During the registration process, the user is asked to attach a photo of his/her national ID to be used in the event of any illegal or unethical activity.
3. **Reliability:** The system should be able to work with full functionality in worst cases (peak demand).
4. **Availability:** The application should always be available to serve the users (at least 95% of the time). The maximum tolerated down time is 72 minutes per day and the meantime between failure and recovery must not exceed 30 minutes.
5. **Security:** All user information must be encrypted to protect it in the event of any breach. The system will protect the data and services from unauthorized access.
6. **Portability**: This software should be usable on different environments such as iOS, web and Android.
7. **Usability:** The application should be very easy to use with a user-friendly interface. It should allow the user to do all the basic features by 6 taps as a maximum.
8. **Robustness:** The system should be robust and handle invalid user input (invalid emails, passwords, etc.) without crashing.
9. **Fault tolerance:** The system should continue to operate properly even if a service fails.
10. **Documentation:** A user guide should be available in the application and software documentation should be available online.

Regarding the application platform, the client can use this software on different environments such as (iOS 10.0+ and Android 7.0+). Also, 4G connection needs to be established from the user’s mobile for stable video and voice call.

The server-side components of the software system must operate within a Windows operating system environment running Node.JS.

* 1. Use Case Diagram and Use Cases:



***Figure 6:*** Use Case Diagram

|  |  |
| --- | --- |
| Use Case ID: | 1 |
| Use Case Name: | Sign up |
| Brief Description: | User creates a new account in the system. |
| Actors: | Normal user, Helper (Monqez) |
| Preconditions: | No account exists for the user in the database. |
| Postconditions: | A new account is added to the database with the user information. |
| Main Scenario: | 1. System displays the signup page. 2. User enters the credentials (email and password). 3. System stores the credentials. 4. System redirects the user to the “Add Additional Information” page. |
| Exception Flow: | 1. System displays an error message. 2. System displays the signup page. |

***Table 1:*** Use Case 1

|  |  |
| --- | --- |
| Use Case ID: | 2 |
| Use Case Name: | Add additional information |
| Brief Description: | User fills in data about himself into the system. |
| Actors: | Normal user |
| Preconditions: | User is signed up. |
| Postconditions: | Additional information is added to the existing user. |
| Main Scenario: | 1. System displays the “Add Additional Information” page. 2. User enters the required data (full name, phone number, national ID, date of birth, address and gender). 3. System stores the information in the database. 4. System redirects the normal user to his/her home page. |
| Exception Flow: | 1. System displays an error message. |

***Table 2:*** Use Case 2

|  |  |
| --- | --- |
| Use Case ID: | 3 |
| Use Case Name: | Apply as Monqez |
| Brief Description: | Monqez fills the signup application into the system. |
| Actors: | Helper (Monqez) |
| Preconditions: | User is signed up. |
| Postconditions: | Application is submitted and pending approval. |
| Main Scenario: | 1. System displays the “Add Additional Information” page. 2. User enters the required data (full name, phone number, national ID, date of birth, address, gender and first-aid certificate). 3. System stores the information in the database. 4. System adds the application to the Monqez Application Queue. 5. System redirects to the login page. |
| Exception Flow: | 1. System displays an error message. 2. System displays “Add Additional Information” page. |

***Table 3:*** Use Case 3

|  |  |
| --- | --- |
| Use Case ID: | 4 |
| Use Case Name: | Log in |
| Brief Description: | Actors authenticate and get into the system. |
| Actors: | Normal user, Helper (Monqez), Administrator |
| Preconditions: | User must exist in the database and has not yet authenticated. |
| Postconditions: | User is authenticated and allowed to use the system functions. |
| Main Scenario: | 1. System displays the login page. 2. User enters the credentials (email and password). 3. System validates the credentials. 4. System checks that the user has added his additional information. 5. System displays the appropriate user home page. |
| Alternative Scenario: | 1. System displays the login page. 2. User enters the credentials (email and password). 3. System validates the credentials. 4. System checks and finds that the user has not added his additional information. 5. System displays “Add Additional Information” page. |
| Exception Flow: | 1. System displays an error message. 2. System displays the login page. |

***Table 4:*** Use Case 4

|  |  |
| --- | --- |
| Use Case ID: | 5 |
| Use Case Name: | Log in with Google |
| Brief Description: | Actors authenticate and get into the system. |
| Actors: | Normal user, Helper (Monqez), Administrator |
| Preconditions: | - |
| Postconditions: | User is authenticated and allowed to use the system functions. |
| Main Scenario: | 1. System displays the login page. 2. User selects his Google account. 3. System validates the credentials. 4. System checks that the user has added his additional information. 5. System displays the appropriate user home page. |
| Alternative Scenario: | 1. System displays the login page. 2. User selects his Google account. 3. System validates the credentials. 4. System checks and finds that the user has not added his additional information. 5. System displays “Add Additional Information” page. |
| Exception Flow: | 1. System displays an error message. |

***Table 5:*** Use Case 5

|  |  |
| --- | --- |
| Use Case ID: | 6 |
| Use Case Name: | Sign up with Google |
| Brief Description: | User creates a new account in the system. |
| Actors: | Normal user, Helper (Monqez) |
| Preconditions: | No account exists for the user in the database. |
| Postconditions: | User is authenticated and a new account is added to the database with the user information. |
| Main Scenario: | 1. System displays the login page. 2. User selects his Google account. 3. System validates the credentials. 4. System checks and finds that the user has not added his additional information. 5. System displays the “Add Additional Information” page. |
| Alternative Scenario: | 1. System displays the login page. 2. User selects his Google account. 3. System validates the credentials. 4. System checks that the user has added his additional information. 5. System displays the appropriate user home page. |
| Exception Flow: | 1. System displays an error message. |

***Table 6:*** Use Case 6

|  |  |
| --- | --- |
| Use Case ID: | 7 |
| Use Case Name: | Add new Admin |
| Brief Description: | An administrator adds a new admin to the system |
| Actors: | Administrator |
| Preconditions: | Administrator is logged in. |
| Postconditions: | A new admin is added to the database. |
| Main Scenario: | 1. System displays the “Add new admin” page. 2. Admin enters the email and password of new admin. 3. System stores the credentials in the authentication database. 4. System marks the created user as an admin. 5. System redirects the original admin to the “Admin Home Screen” page. |
| Exception Flow: | 1. System displays an error message. |

***Table 7:*** Use Case 7

|  |  |
| --- | --- |
| Use Case ID: | 8 |
| Use Case Name: | Review Monqez application |
| Brief Description: | An administrator reviews a Monqez application and either accepts or declines the Monqez |
| Actors: | Administrator |
| Preconditions: | Administrator is logged in.  A user has applied as a Monqez. |
| Postconditions: | Monqez application is reviewed. |
| Main Scenario: | 1. System displays all unreviewed Monqez applications. 2. Admin selects a Monqez Application. 3. System displays all the details of the application (Name, age, phone number, gender, and the First-Aid certificate. 4. Admin chooses to accept the application. 5. System emails the Monqez with his acceptance and allows him to login to the system as well as mark the application as reviewed. |
| Alternative Scenario: | 1. System displays all unreviewed Monqez applications. 2. Admin selects a Monqez Application. 3. System displays all the details of the application (Name, age, phone number, gender, and the First-Aid certificate. 4. Admin chooses to decline the application. 5. System emails the Monqez with his rejection and marks the application as reviewed. |
| Exception Flow: | 1. System displays an error message. 2. System displays all unreviewed Monqez applications. |

***Table 8:*** Use Case 8

|  |  |
| --- | --- |
| Use Case ID: | 9 |
| Use Case Name: | Edit Account |
| Brief Description: | User edits information at his profile. |
| Actors: | Normal user, Helper (Monqez), Administrator |
| Preconditions: | User logged in. |
| Postconditions: | User’s information updated. |
| Main Scenario: | 1. System displays the edit account page. 2. User enters the data (full name, phone number, date of birth, address, and gender) he wants to update. 3. System updates the data in the database. |
| Exception Flow: | 1. System displays an error message. |

***Table 9:*** Use Case 9

* 1. User Groups and Access Privileges:

***The following are the user groups that exist in the system:***

1. **Normal user:** is a user who was at the scene of the accident reporting and asking a paramedic to aid for him/her or another person. The normal user can perform the following functions:

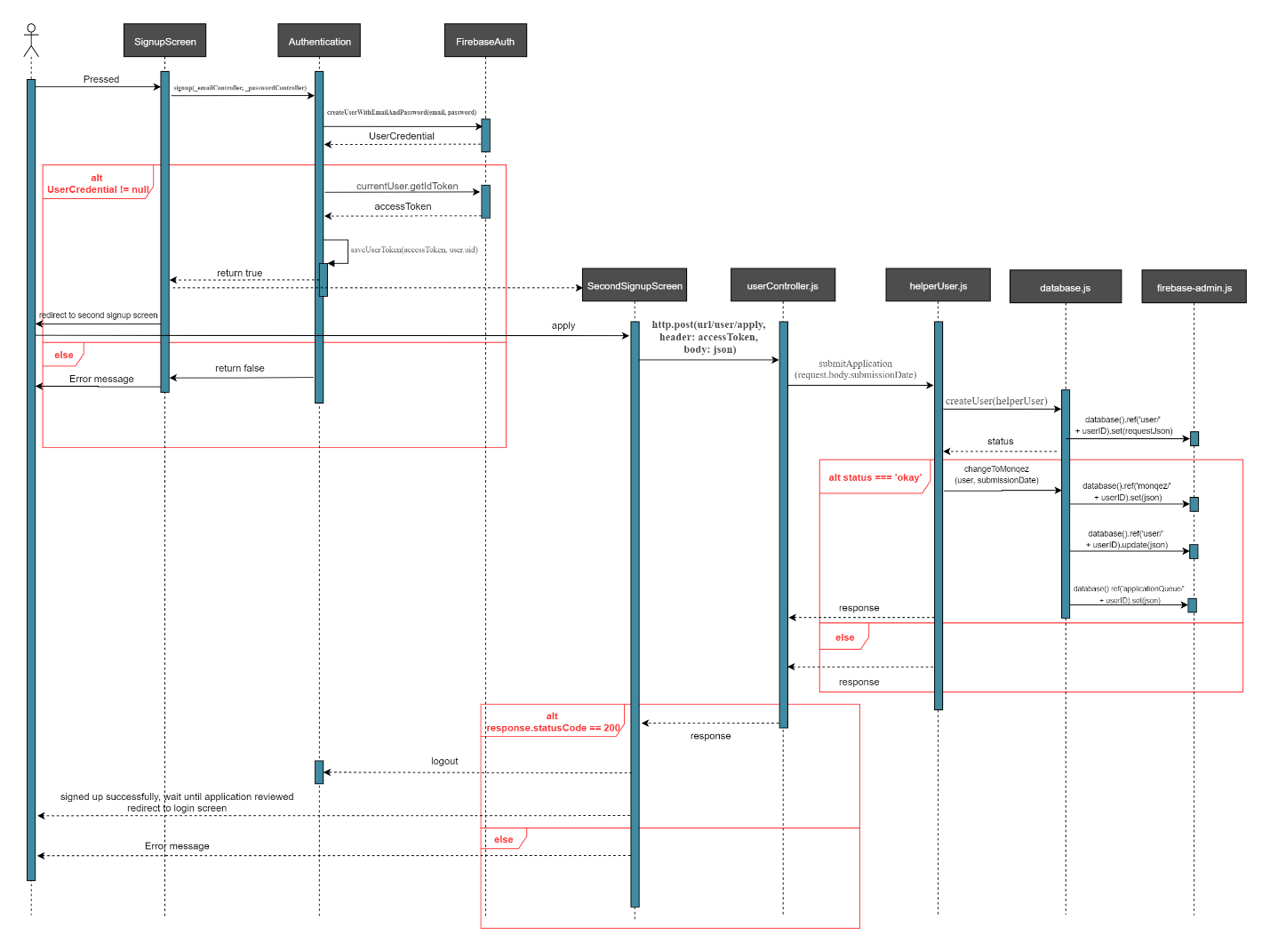
* ***Sign up:*** The normal user can register his/her details to be able to call paramedics.
* ***Log in:*** The normal user can log in using his/her phone number or email to be able to use the application functionalities.
* ***Ask for Monqez:*** The normal user has the capability to ask for Monqez in case he/she or others were involved in an accident or need quick medical care. The normal user can ask for either onsite or online (video/voice) Monqez.
* ***View first aid instructions:*** The normal user can view first aid instructions about many different types of injuries.
* ***Rate the Monqez.***

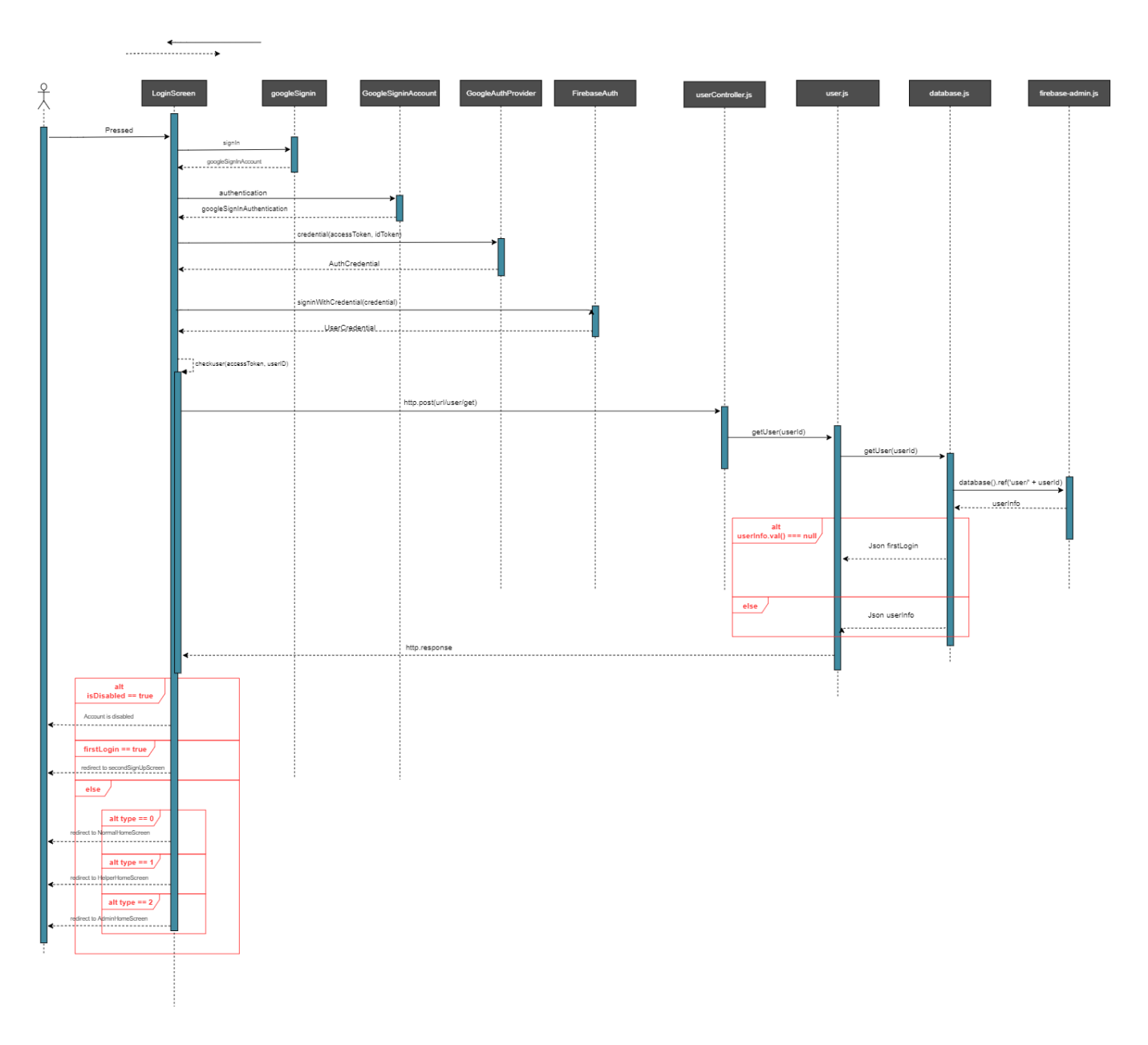
1. **Monqez user:** is a user who registers in our system as a rescuer (Monqez) by attaching an accredited certificate of the first aid training course he/she took from any accredited organization and then awaits approval of his application to join the rescuer list until the certificate is reviewed. The Monqez can perform the following functions:

* ***Sign up:*** The Monqez user can add his/her information attached with the first aid certificate to the rescuers queue to be reviewed.
* ***Log in:*** The Monqez user can log in using his/her phone number or email to be able to use the application functionalities.
* ***Specify working time:*** The Monqez user has the option to select his/her status and when he/she will be available for work.
* ***Accept requests:*** The Monqez user can accept either onsite or online (video/voice) requests.
* ***Request Description:*** The Monqez user can view the description of the request provided by the normal user who made it.
* ***Make complaint:*** The Monqez user can complain about any inappropriate events that happened while fulfilling a request.

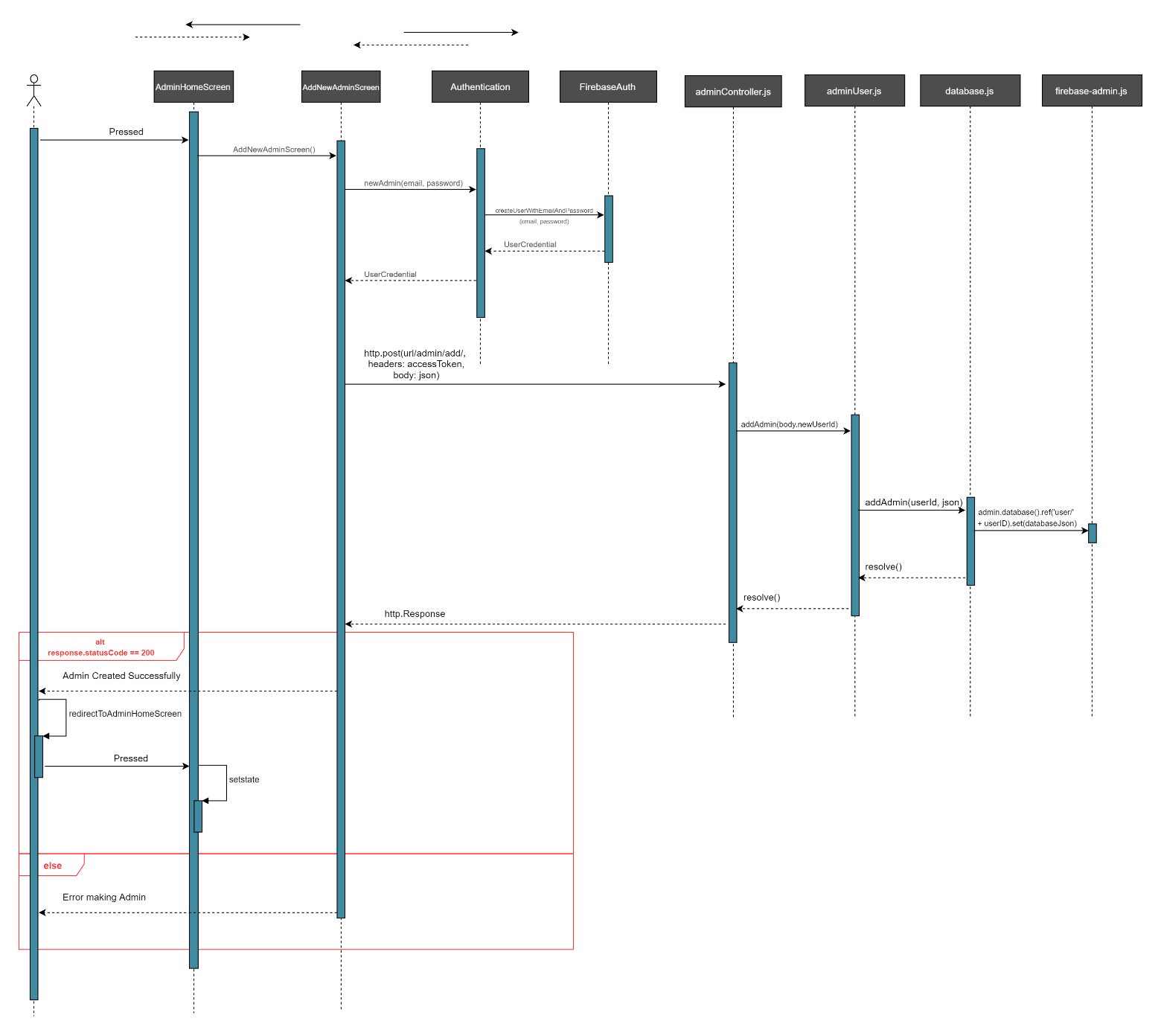
1. **Admin user:** The Admin user accepts or rejects requests to join the rescuers list based on the validity of the certificates that the rescuers attach when creating an account. The admin can perform the following functionalities:

* ***Log in:*** The admin can log in using his/her phone number or email to be able to use the application functionalities.
* ***Add another admin.***
* ***View Monqez applications requests:*** The admin can retrieve all the non-reviewed Monqez applications requests and browse them.
* ***Accept/decline Monqez application requests:*** The admin user has the capability of accepting or declining the Monqez application requests after reviewing the first aid certificate.
* ***View ratings and complaints:*** The admin can review complaints and ratings.
* ***Block users:*** The admin can block any user if there are many valid complaints against that user.
  1. Sequence Diagrams:

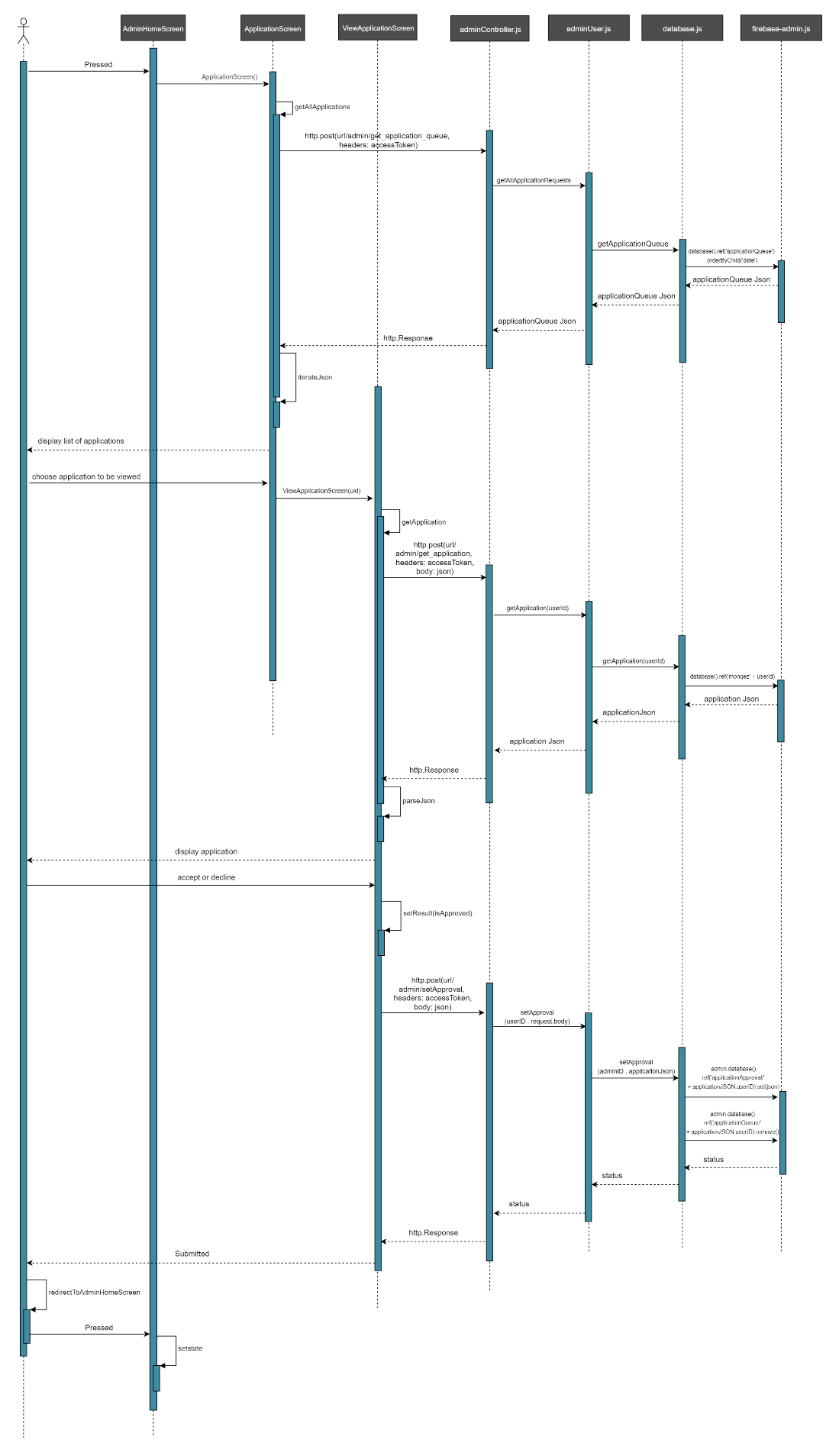
***Figure 7:*** “Sign up as Monqez” Sequence Diagram



***Figure 8:*** “Sign in with Google” Sequence Diagram



***Figure 9:*** “Add new Admin” Sequence Diagram

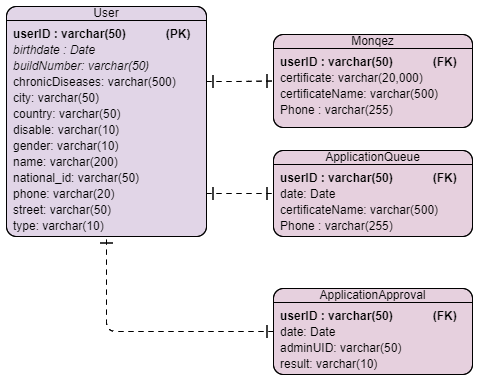


***Figure 10:*** “Accept or Decline Monqez Application” Sequence Diagram

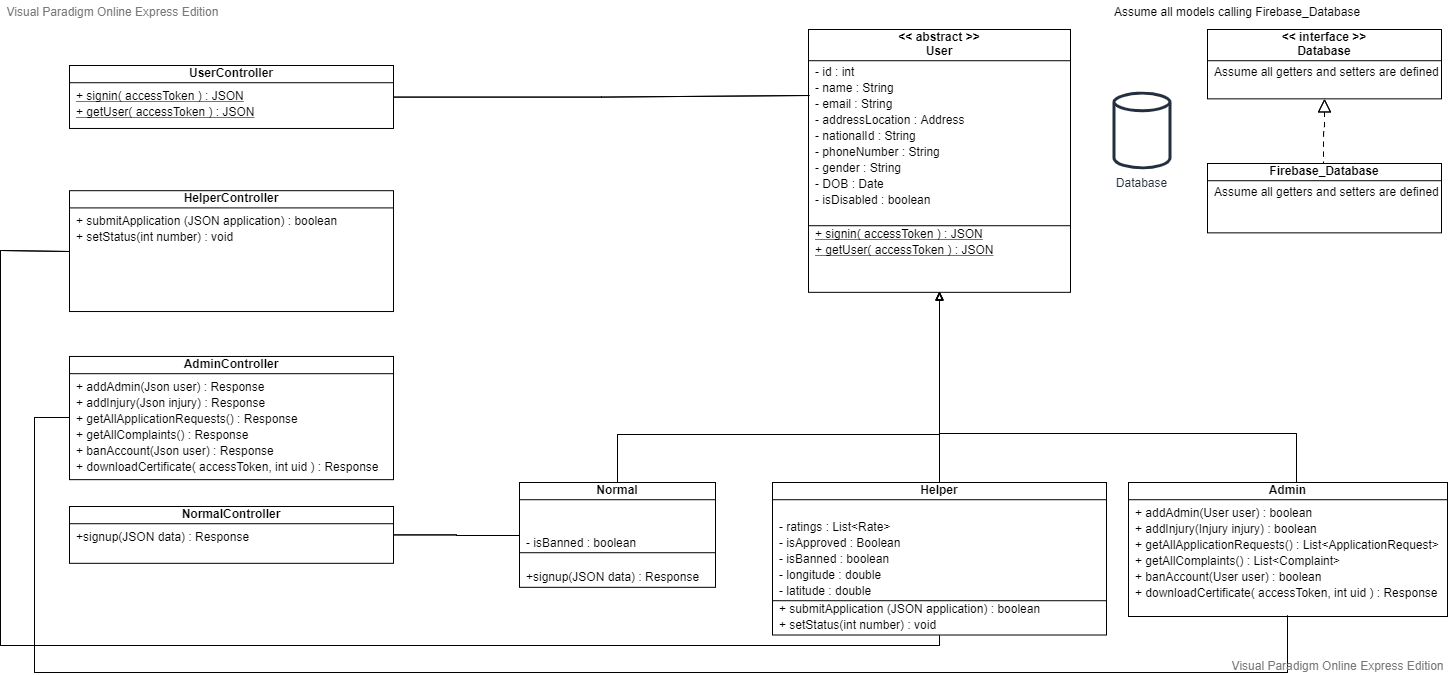
* 1. System Test Cases:

Chapter 4: System Design

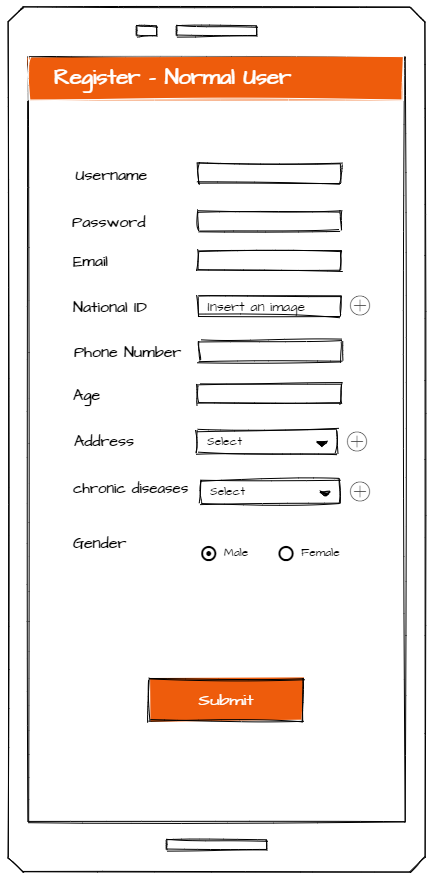
* 1. System Component Diagram:
  2. Application ERD:

 ***Figure 11:*** Application Entity Relationship Diagram

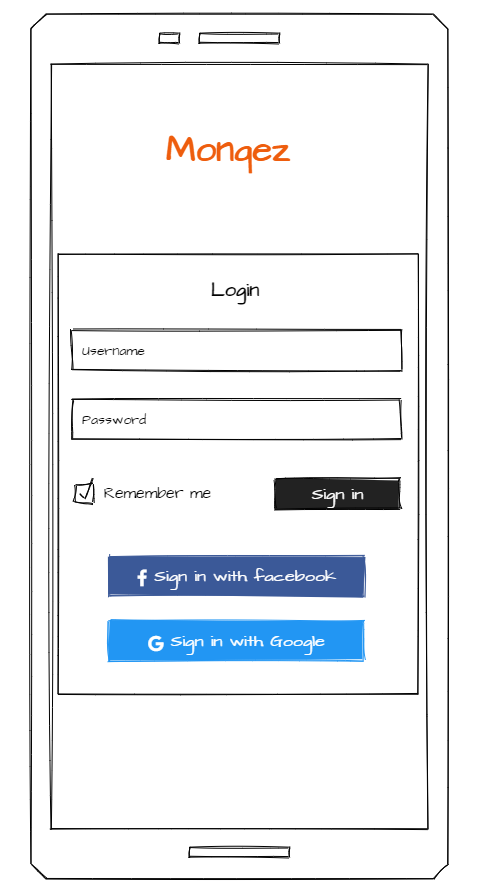
* 1. Class Diagram:

 ***Figure 12:*** System Class Diagram

* 1. Elementary Design:
  2. GUI Design:



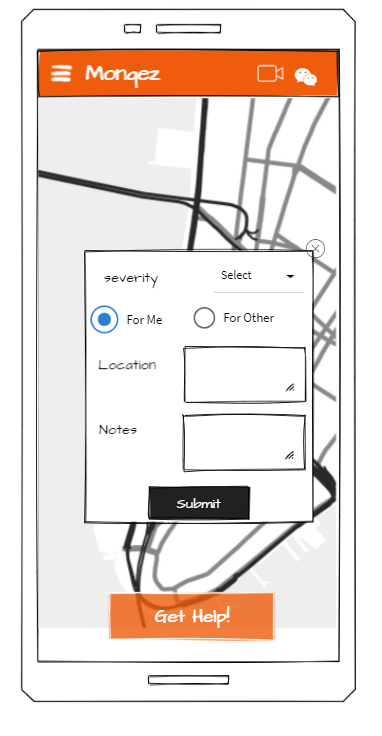
***Figure 13:*** Register Normal User Screen



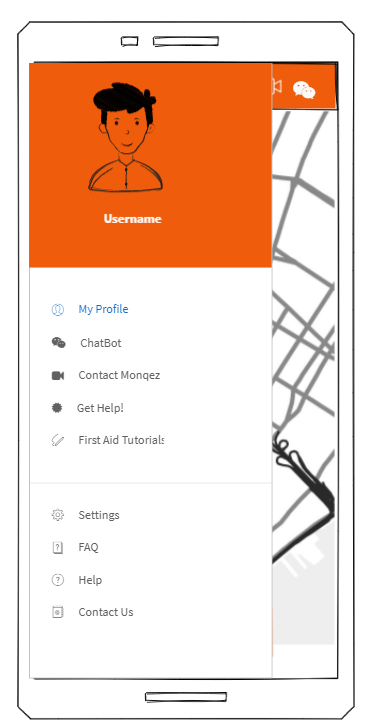
***Figure 14:*** Login Screen



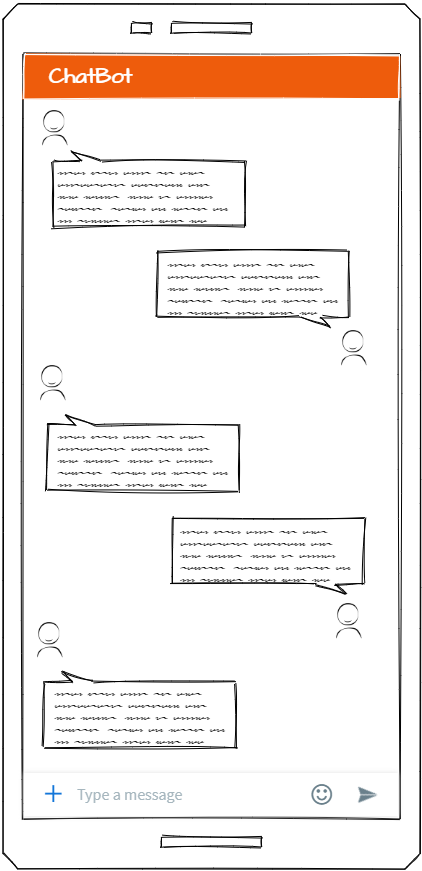
***Figure 15:*** Normal User Home Screen



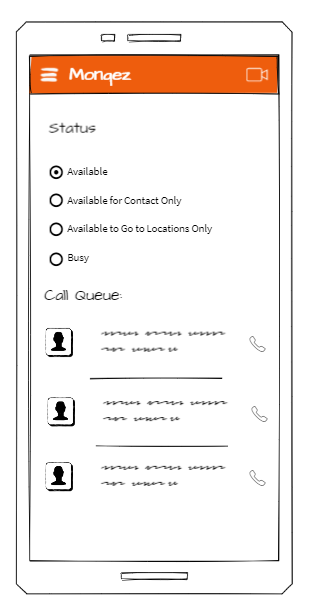
***Figure 16:*** Request’s Additional Information Popup



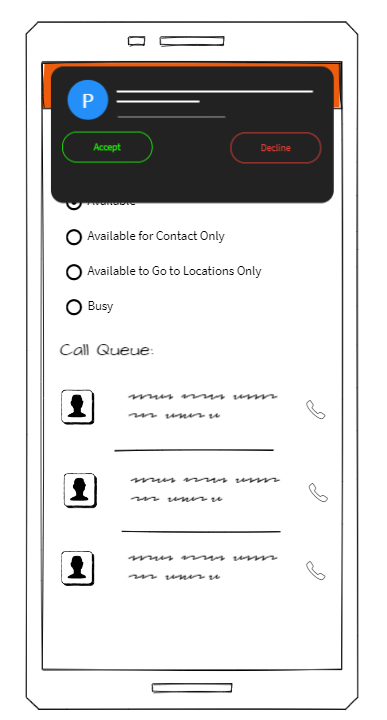
***Figure 17:*** Normal User’s Navigation Drawer



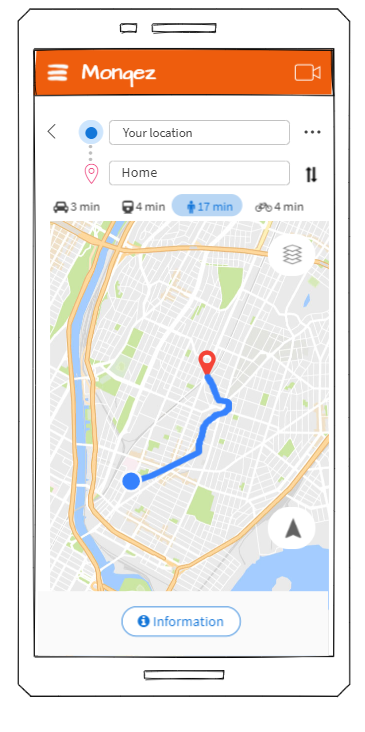
***Figure 18:*** Chatbot Screen



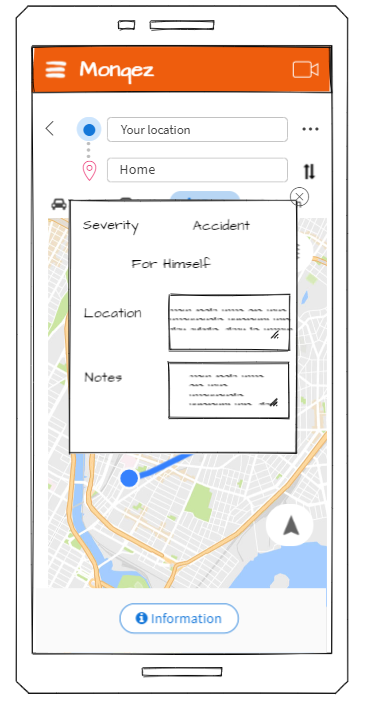
***Figure 19:*** Monqez Home Screen



***Figure 20:*** Monqez Request Notification



***Figure 21:*** Monqez Navigation Screen



***Figure 22:*** Monqez Display Request’s Additional Information Popup

Chapter 5: Implementation and Testing

* 1. Methods:

We used two architectures in our project, first one is **Client-Server** architecture which is used to connect the Front-End Application (Flutter) with the Back-End Server (node JS), and the second is **MVC** architecture which used to Design the Server, the server designed to contain a lot of models which is used in other sections of the server, the entry point of our server is the controller layer which accepts any request , authorizes it, and routes the request to the correct function to be processed. Another layer in the server is the database where we used firebase’s real-time database, the server is responsible for communicating with this layer through the server models.

In our system we used two level of authentication, this done in both sign up and sign in. In sign up the first level of authentication is creating the user’s account in the firebase Auth where we take user’s email and password. The second level is where we take all the user’s additional information and insert it in the real-time database. We also used two level of authentication in sign in, the first one is authenticating the user credential used firebase Auth. The second one is retrieving the type of the user from the real-time database.

* 1. Results:

Chapter 6: Conclusion

Chapter 7: Future Work

References

[1] <https://www.theguardian.com/society/2010/apr/12/first-aid-skills-deaths>

[Last visited: January 2021]

[2] <https://www.manchester.ac.uk/discover/news/over-half-of-deaths-from-injury-could-be-prevented-if-public-knew-first-aid/>

[Last visited: January 2021]

[3] <https://www.redcross.org.nz/first-aid/all-available-courses/>

[Last visited: January 2021]

[4] <https://www.hiamag.com/%D9%85%D9%86%D9%88%D8%B9%D8%A7%D8%AA/%D8%A7%D8%AE%D8%A8%D8%A7%D8%B1/848121-%D8%A7%D9%84%D9%87%D9%84%D8%A7%D9%84-%D8%A7%D9%84%D8%A3%D8%AD%D9%85%D8%B1-%D8%A7%D9%84%D8%B3%D8%B9%D9%88%D8%AF%D9%8A-%D9%8A%D8%AF%D8%B4%D9%86-%D8%A7%D9%84%D9%85%D8%B1%D8%AD%D9%84%D8%A9-%D8%A7%D9%84%D8%AB%D8%A7%D9%86%D9%8A%D8%A9-%D9%85%D9%86-%D8%AA%D8%B7%D8%A8%D9%8A%D9%82-%D8%A7%D9%84%D9%85%D8%B3%D8%B9%D9%81-%D8%A7%D9%84%D8%A5%D9%84%D9%83%D8%AA%D8%B1%D9%88%D9%86%D9%8A>

[Last visited: January 2021]

[5] eMedic (Apple App Store): <https://apps.apple.com/app/emedic/id349527488>

[Last visited: January 2021]

[6] PulsePoint Respond (Google Play):

<https://play.google.com/store/apps/details?id=mobi.firedepartment&hl=en_US>

[Last visited: January 2021]

[7] Asifny (Google Play):

<https://play.google.com/store/apps/details?id=com.srca.apps.sos&hl=en&gl=US>

[Last visited: January 2021]

[8] Inaa’sh (Apple App Store):

<https://apps.apple.com/eg/app/inaash/id993321301>

[Last visited: January 2021]