FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Project Report

in the course

Mobile Computing

to attain the degree of one

Master of Engineering (M. Eng.)

Title

Disaster Management Application

Submitted by:

DIAB ELMEHDI (1324316)
Magdalena Pękacka (1345023)

In the Winter-Semester 20/21 at Department: Electrical Engineering Frankfurt University of Applied Sciences

****

Examiner: Prof. Dr. Ulrich Trick

Delivery Date: 08.03.2021

Abstract:

The research project investigates the possibility to deploy an emergency communication infrastructure in case of a natural disaster destroyed or heavily damaged the existing communication infrastructure. The emergency communication infrastructure is constructed from battery-supplied multi-radio wireless outdoor-routers which are deployed by first responders in the affected region. The outdoor-routers are connecting to each other and are building a wireless mesh network (WMN)-based disaster network, which provides the basis for an IP-based communication. Various network functionalities and services for the end-users are provided within the WMN-based disaster network by utilizing the concept of NFV. This Project consists of the design and development of a disaster management application which will be provided to the end-user groups of the WMN-based disaster network. The application can consider any aspect that could be relevant regarding the management within a disaster and could either focus on the management of the ordinary citizens, the official government groups, a combination of both or other relevant aspects.

Keywords— ***WMN, NFV, disaster management application***

FRANKFURT UNIVERSITY OF APPLIED SCIENCES

Table of Contents

Table of Figures

List of Abbreviations

NFV	Network Function Virtualization
WMN	Wireless Mesh Network
IP	Internet Protocol
SOS	Save Our Souls
APP	Application
JWT	JSON Web Token

Introduction

In the last couple of years, almost every country has seen a dramatic increase in the number of disasters and crises. Increased population, climate change and technical progress has resulted in breakdowns of energy grids, floods, wildfires, and technical accidents such as destruction of dams and oil leakages. Unfortunately, very view countries commit sufficient resources to disaster management, regardless of past experience. Because of the lack of real infrastructure for responding to disasters and crisis, it is usually very difficult for people responsible for managing events to respond quickly and effectively, and it is also very difficult for the people affected by such an event to respond to it accordingly. In this report, we will present a web-based system for managing disasters and crisis to help the victims or the emergency cases to get in contact with rescue helpers quickly also to avoid huge damages in order to help the maximum people in a short amount of time. This application contains three different pages each of these had specific functionality. The main objective of this report is to describe functionality of the application then an explanation of the components building the disaster management application and their relations to each other in third part Description of the individual implementation on the backend, the fourth part will be an installation guide of all steps required to put the application into operation and on last part we will show a simulation and the usage of the application.

Objectives and functionality of the Application

This web application is designed to help the victim or the emergency cases to rescue them in a short amount of time also to ease the communication between the person who is asking for help and the helper. The architecture of this application is supposed to be easy handling for every person who is using it. It contains three different pages namely: Home Page, Supplies/ Survival kits and About. The first one is considered as the front page where the user (victim) or admin (rescue Helper/ response team) can access the app. At the center, an SOS button is dedicated to the emergency cases where the user needs the help and does not have time to register himself based on his situation so in this case, once he presses the button immediately a popup asks him to share his location to the rescue helper to locate himself. On the other hand, the rescue team receives his address Ip and location that pins into the map then immediately he calls everyone to go the specified location, at the same time the helper set and activate the timer and the countdown start working on the other side as sign that the helpers are coming on this amount of time. If it is not an emergency case the user still can reach for the help via the victim button where he should fill the form in order to sign in then he is redirected to another page to select one of the disaster that he is witnessing, then the website asks him for the username once he provides it, a modal pops up and their the victim can text the response team while they will reply based on the information they have received from the form also a notification must be seen on the victim's modal. The Second Page represents a list of supplies as a (medical, clothing, household) that the person should have in the case of emergency or need. Last Page is about general contact and the main purpose of this whole application.

Programming tools:

Hypertext Markup Language (HTML) is the set of markup symbols or codes inserted into a file intended for display on the Internet. The markup tells web browsers how to display a web page's words and images.

CSS is the acronym of “Cascading Style Sheets”. CSS is a computer language for laying out and structuring web pages (HTML or XML). This language contains coding elements and is composed of these “cascading style sheets” which are equally called CSS files (. CSS)

JavaScript is a programming language commonly used in web development. It was originally developed by Netscape as a means to add dynamic and interactive elements to websites.
While JavaScript is influenced by Java, the syntax is more similar to C and is based on ECMAScript, a scripting language developed by Sun Microsystems

Spring Boot is an open-source Java-based framework used to create a micro-Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. This chapter will give you an introduction to Spring Boot and familiarizes you with its basic concepts.

Database:

PostgreSQL is a powerful, open-source object-relational database system that uses and extends the SQL language combined with many features that safely store and scale the most complicated data workloads. The origins of PostgreSQL date back to 1986 as part of the POSTGRES project at the University of California at Berkeley and has more than 30 years of active development on the core platform.

Framework:

React.js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications. It is used for handling the view layer for web and mobile apps. React also allows us to create reusable UI components.The main purpose of React is to be fast, scalable, and simple.

IDE'S:

IntelliJ IDEA is an Integrated Development Environment (IDE) for JVM languages designed to maximize developer productivity. It does the routine and repetitive tasks for you by providing clever code completion, static code analysis, and refactoring's, and lets you focus on the bright side of software development, making it not only productive but also an enjoyable experience.

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages (such as C++, C#, Java, Python, PHP, Go) and runtimes (such as .NET and Unity)

Spring Security is a framework that focuses on providing both authentication and authorization to Java applications.

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Components and their relation to each other

1. Home Page

This page contains three components:

- Victim: it allows the person who is in less danger to contact the response team by following some instruction.
- Rescue Helpers: the people who link the victims or SOS cases to the response team.
- SOS: Easy access to get the help in a short amount of time

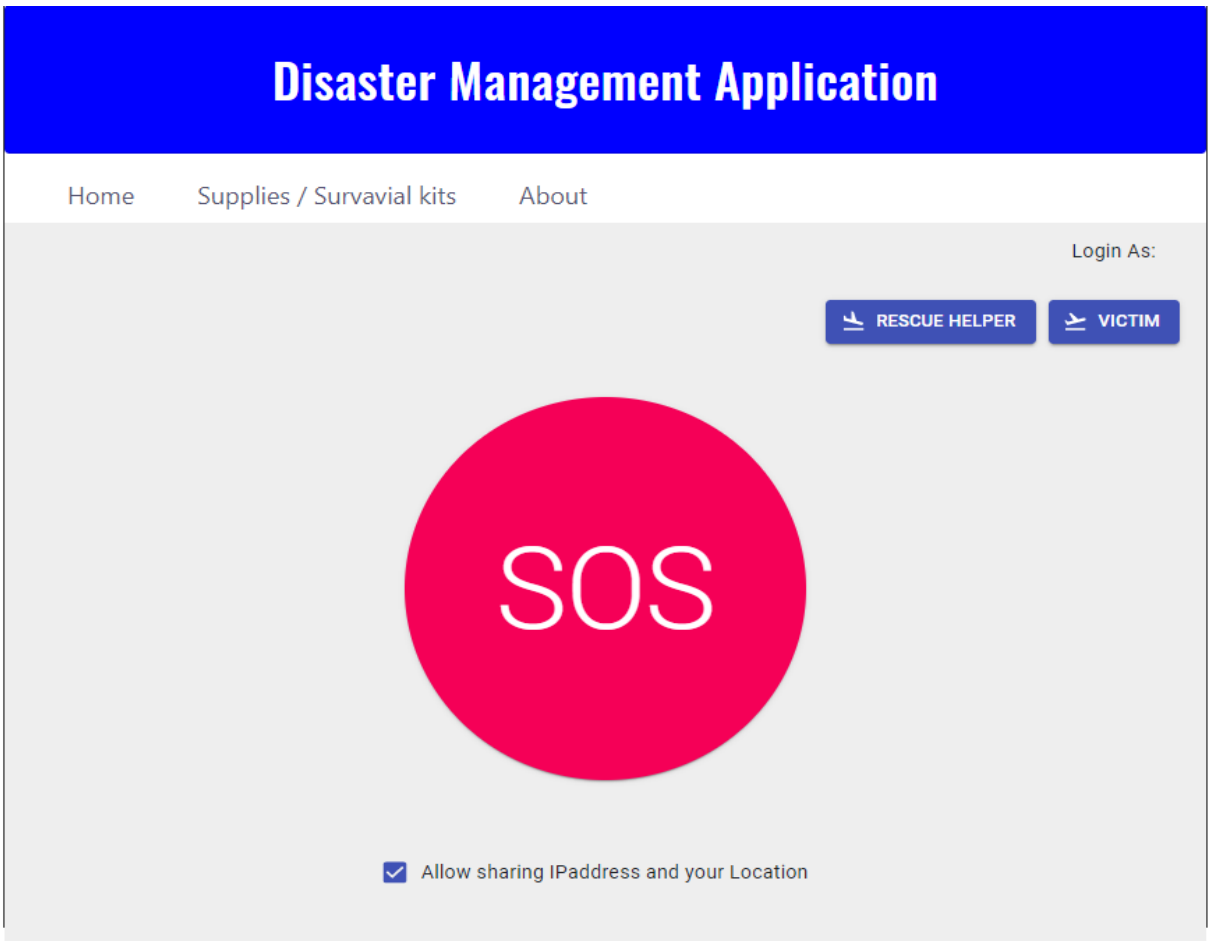


Figure 1 : The application home page

a. SOS

On this page the user should accept to share his own IP address and location then he should wait for the timer to start.

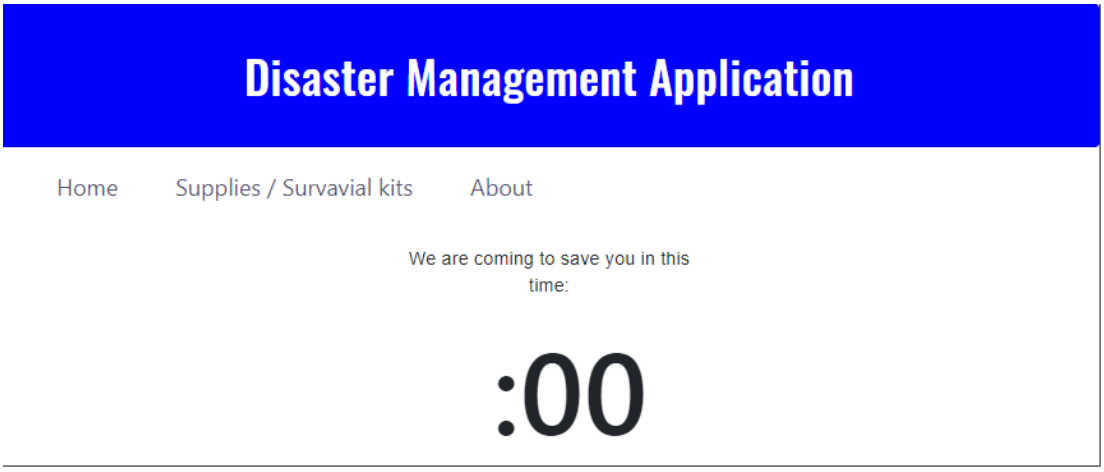


Figure 2: SOS Interface

b. Victim

Once the victim clicks on the button first he should provide some information by filling out the form as it is mentioned below:

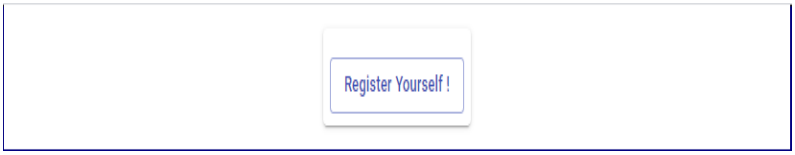


Figure 3 : Registration button

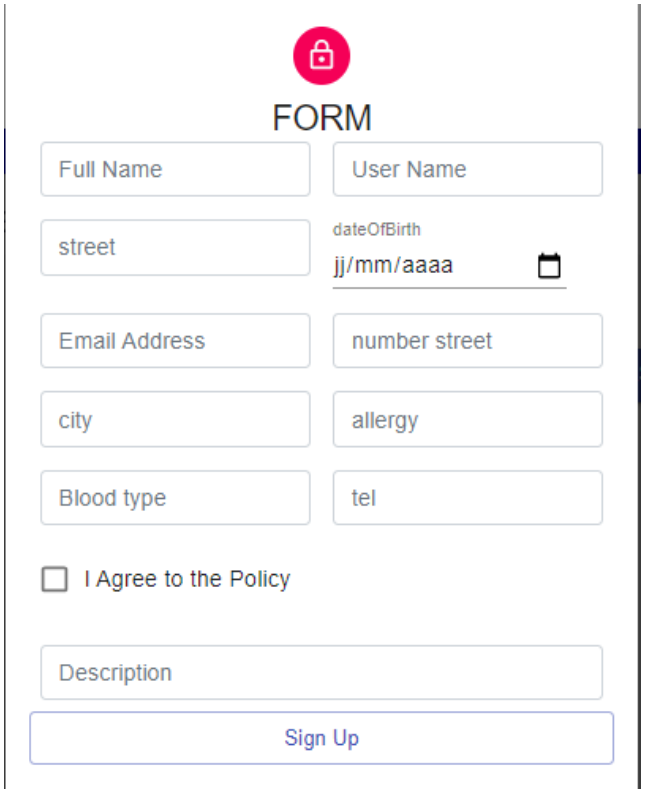


Figure 4 : Sign in as a Victim

Once he fills the form, he redirects to the another page where he can choose the type of emergency by clicking on the circle button.

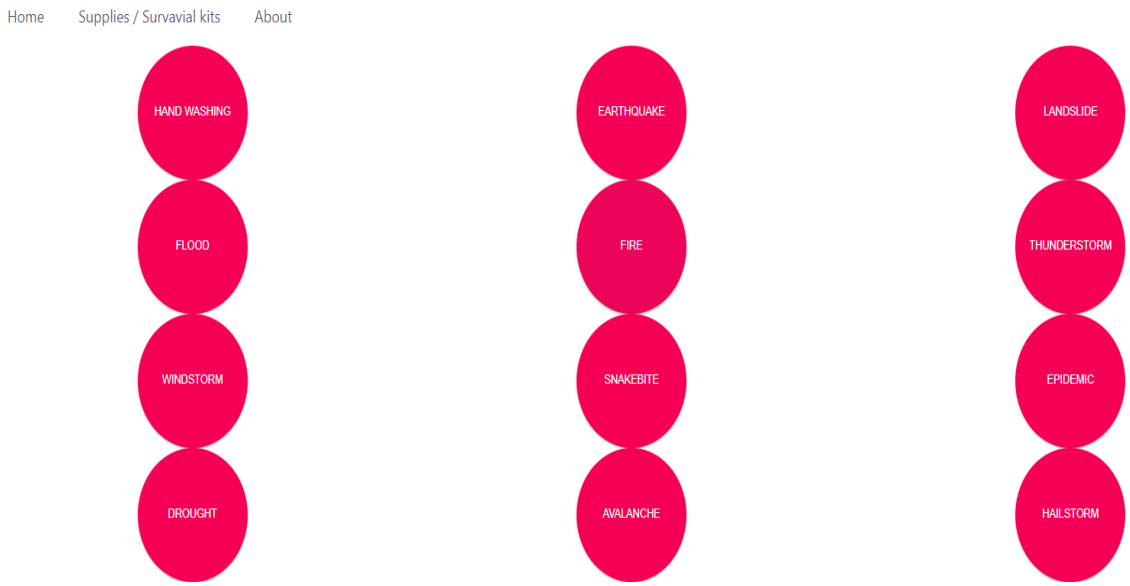


Figure 5 : Choose disaster interface.

Once he chose the concerned disaster, he should provide the same username that he used in the form.

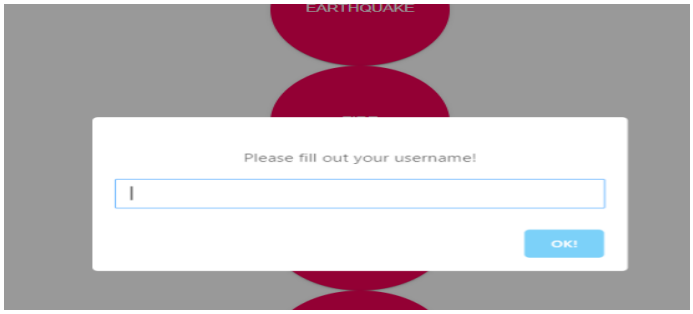


Figure 6 : confirmation of the Victim's username

After that, a modal shows up then if he wants to text the response team, the victim must use again the same username and his message then he submits it.

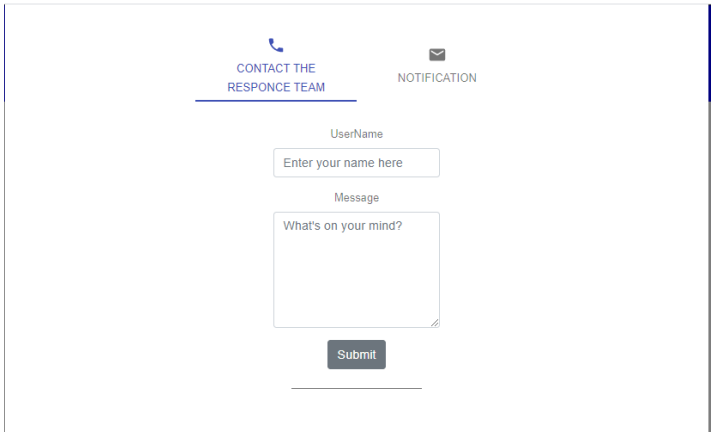


Figure 7: send message to rescue helper

Once the rescue helper got the message, they send back a reply where it can be seen in notification, but the victim must click on the button in order to refresh the zone text by clicking on the button "Get the message from Rescue Helper".

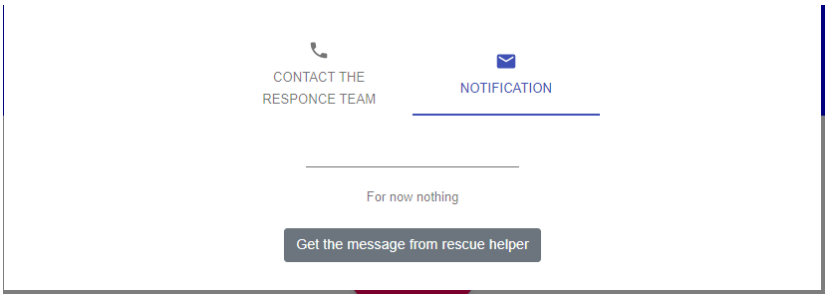


Figure 8: notification message

c. Rescue Helper

On this Component the rescue helper login via his matriculation number and his own password if he does not have an account he must sign in by completing the form.

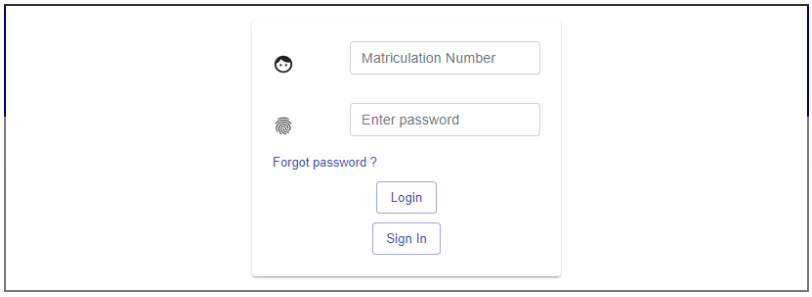


Figure 9 : Login as a Rescue Helper

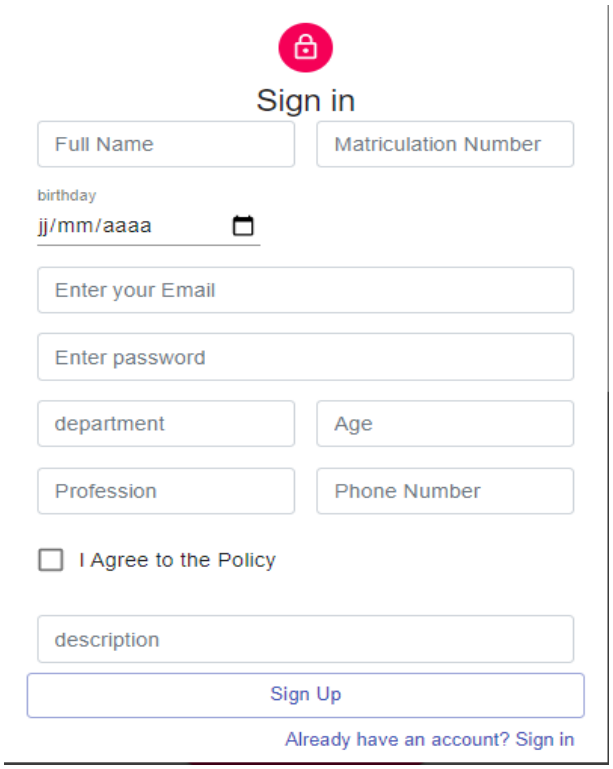


Figure 10: sign in as a rescue helper.

After the login, the rescue helper interface contains two tables one for the victim details the other one for SOS cases. Below the table, a map shows the position of each one based on the shared location and the one provided by the victim, the red pin refers to the SOS case and blue one for the victim. The last component is a text-box to send a feedback message to the victim.

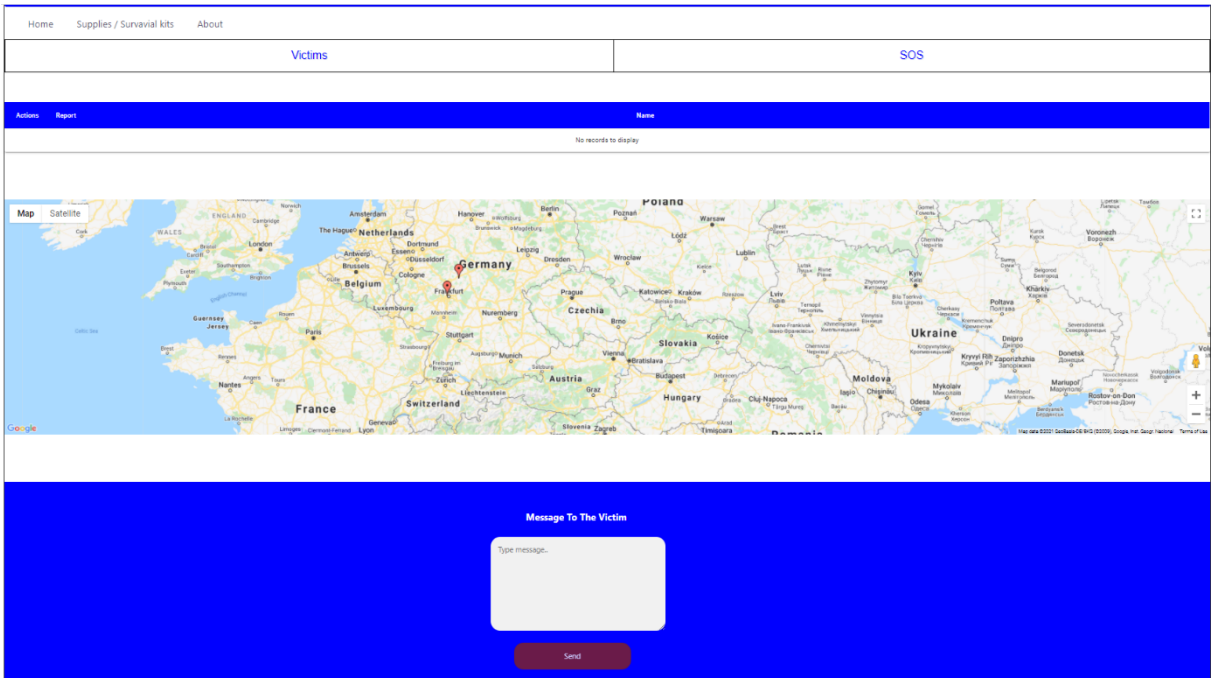


Figure 11: Rescue Helper Portal

If the admin forgets his own password, he can get it set a new one via this page.

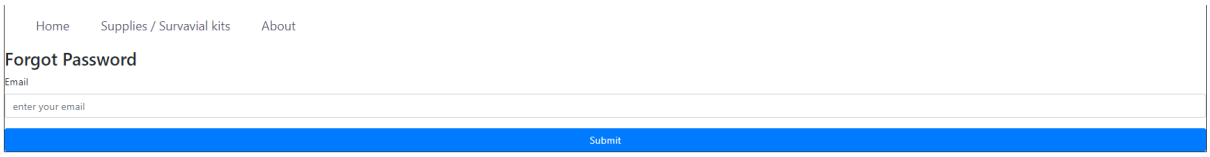


Figure 12: forgot password interface

2. Supplies/ survival kits

The second page contains a set of information to be considered mandatory equipment for any outdoor enthusiast. You never know when something will go wrong, placing your very life will be in danger. But, if you have a well-conceived survival kit with you, your odds of survival will improve greatly.

Figure 13: Supplies/Survival Kits page

3. About

The last page consists of the main purpose of this application in addition to a general contact to reach for more information.

Figure 14: general information page

Implementation of the Application:

The component diagram describes the modeled system as reusable components and highlights their dependency relationships. The following figure shows the diagram of project components:

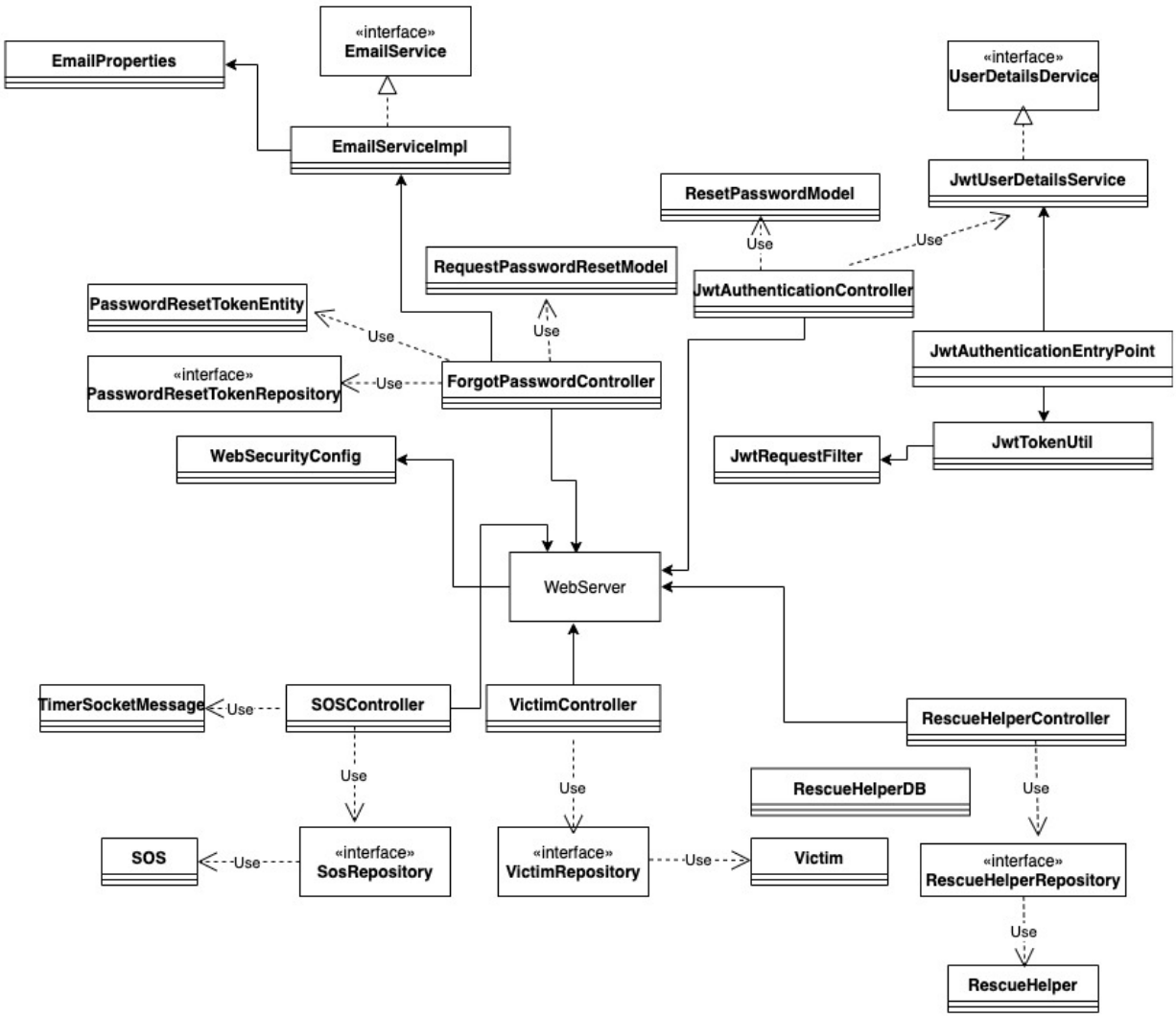


Figure 15:Web Platform Components Diagram

Class diagram:

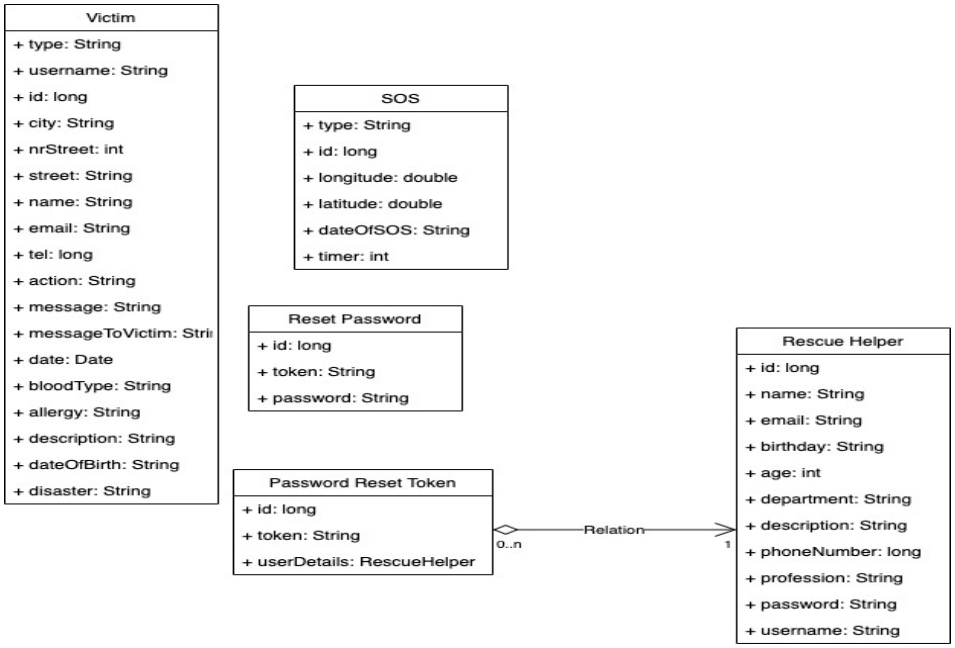


Figure 16: Database Architecture

Login Implementation:

Basic authentication is restricted to username and password authentication.

1. Users will start by authenticating with a username and password.
2. Once authenticated, the client will receive a JWT representing an access token.
3. The client will include the access token in the authorization header of every request to a secure endpoint.
4. The server will validate the access token and determine if it has the right permissions, using the information within the token.

Installation Guide:

Docker:

Dockerfiles were used to dockerize our web application. We created a Docker image on Linux-based system - Debian that has *nginx* installed that is used to create the container we need for automatic builds. We installed Java, Postgres and Node, run the .jar file and served the frontend with *nginx*.
We also created the **docker-compose** file to develop the project locally.

The containers which we implemented are:

```
build: ./containers/backend
container_name: disaster-backend
depends_on:
  - postgres
ports:
  -  8080:8080
volumes:
  - ./backend:/application
command: ['/startApp.sh']
```

Figure 17 : Java container for Spring Boot Application

```
postgres:
  image: postgres
  container_name: disaster-postgres
  ports:
    - 5433:5432
  environment:
    - POSTGRES_DB=victims_db
    - POSTGRES_HOST_AUTH_METHOD=trust
```

Figure 18 :Postgres database container

```
frontend:
  build:
    context: ./
    dockerfile: ./containers/frontend/Dockerfile
  container_name: disaster-frontend
  ports:
    -  3000:3000
  volumes:
    - ./frontend:/application
    - /application/node_modules
```

Figure 19:Node container for the React Application.

```
pgadmin:
  image: dpage/pgadmin4
  container_name: disaster-pgadmin
  environment:
    - PGADMIN_DEFAULT_EMAIL=pgadmin4@pgadmin.org
    - PGADMIN_DEFAULT_PASSWORD=admin
  volumes:
      - pgadmin:/root/.pgadmin
  ports:
    - "5050:80"
```

Figure 20: pgadmin4 for the visualization of the database

To run our web application, it is needed to execute the following command:

docker run -d -p 8080:8080 -p 3000:80 --name dm madzikowska1/disaster-management.

It provides mapping from the Docker container port 8080 and 80 to host machine port 8080 and 3000.

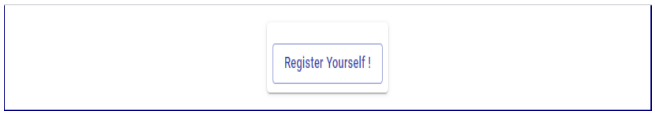
madzikowska1/disaster-management is a name of the docker image on the Docker Hub account.

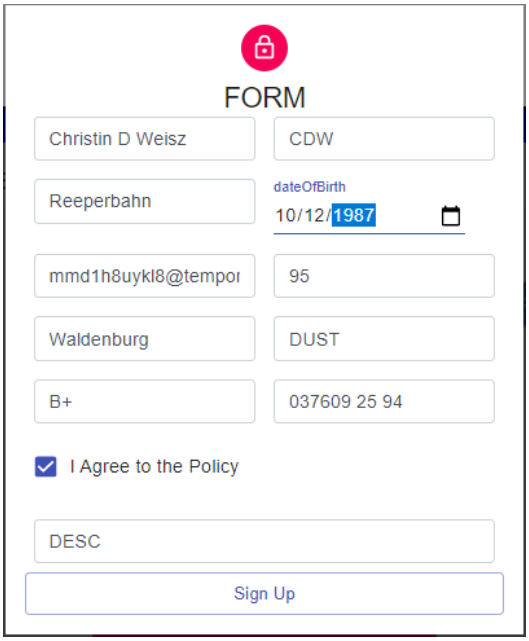
To test our container, navigate to:
-Backend : **localhost:8080**
-Frontend **localhost:3000**

Simulation regarding the Usage of the Application:

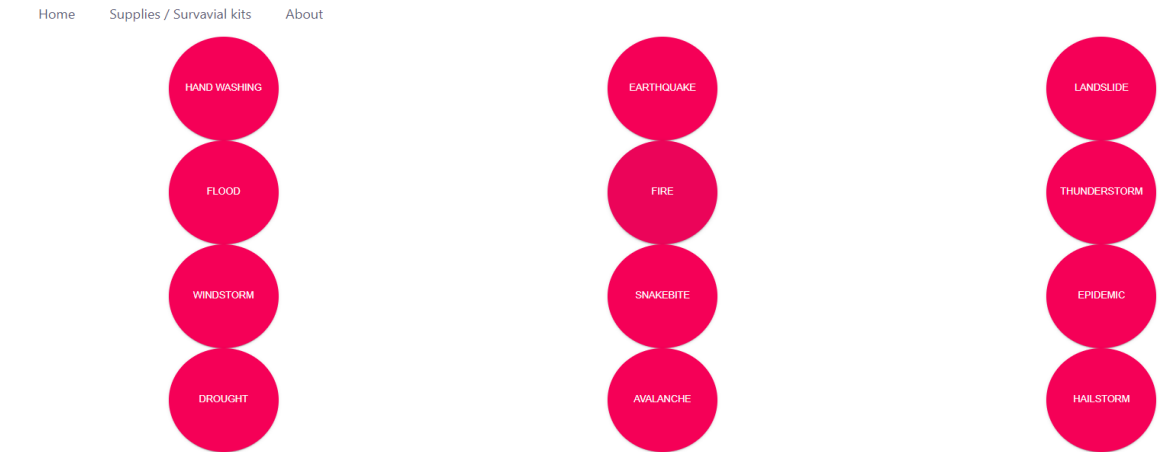
On this part we will show a real simulation for the 2 persons who want to reach for help using our Web Application.

Person1 witnessed an earthquake but he is in less danger so in this case he is considered as a victim. The first step is to enter the home page then click on the victim button once he registers himself as it shown below.

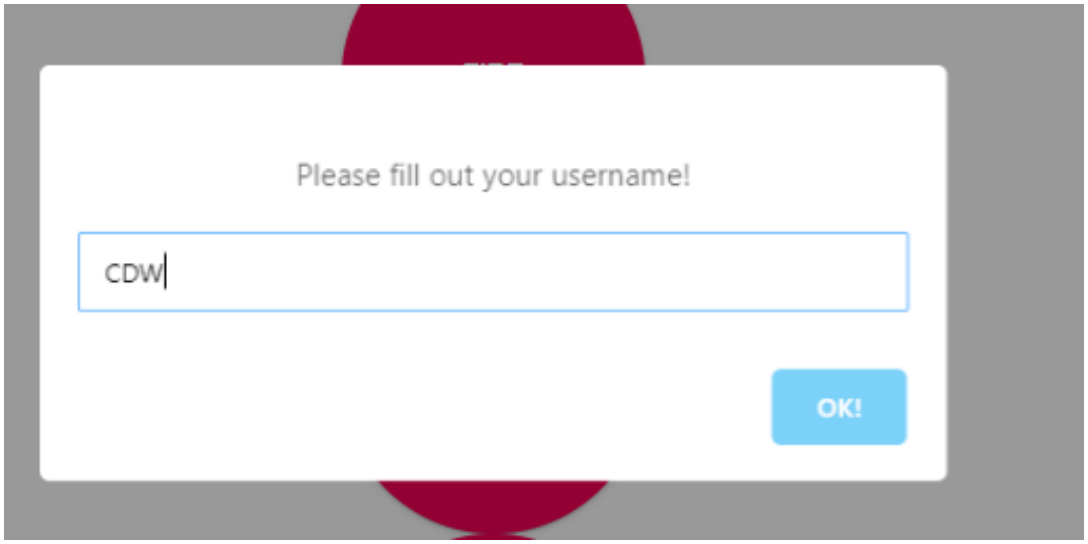


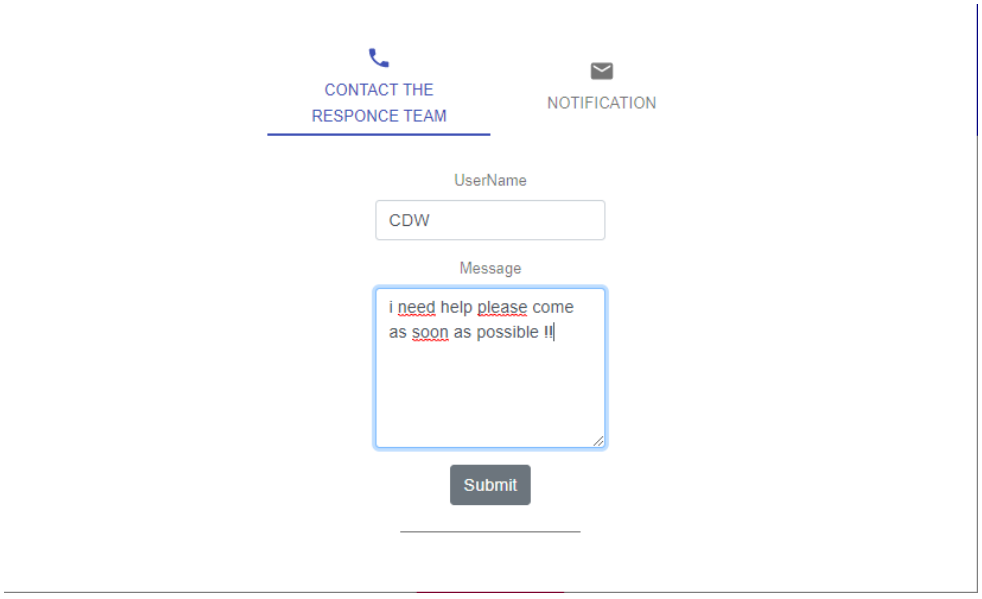


Then it comes step 2, Christin chooses the earthquake button then confirms his username in order to contact the rescue helper.

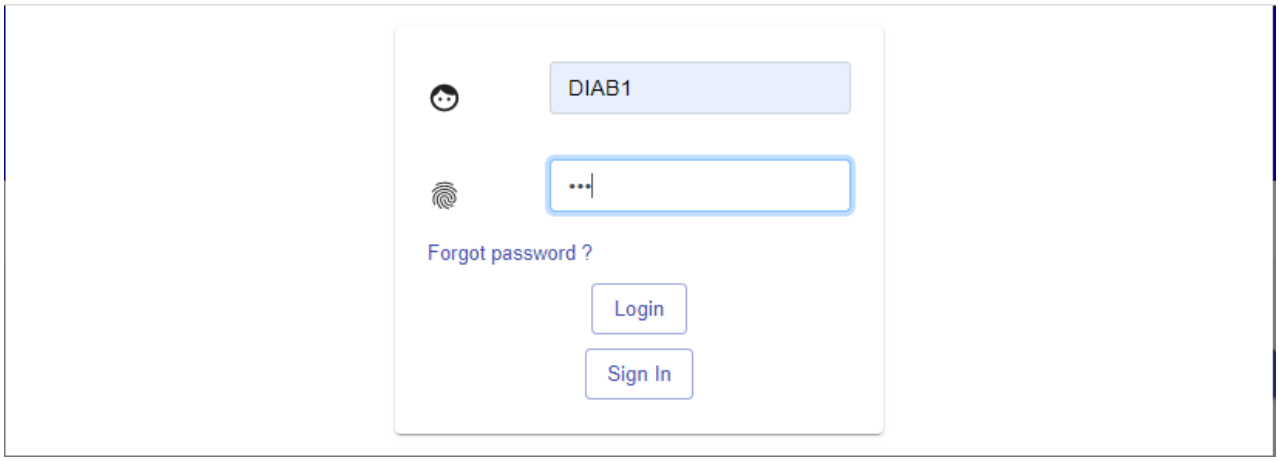


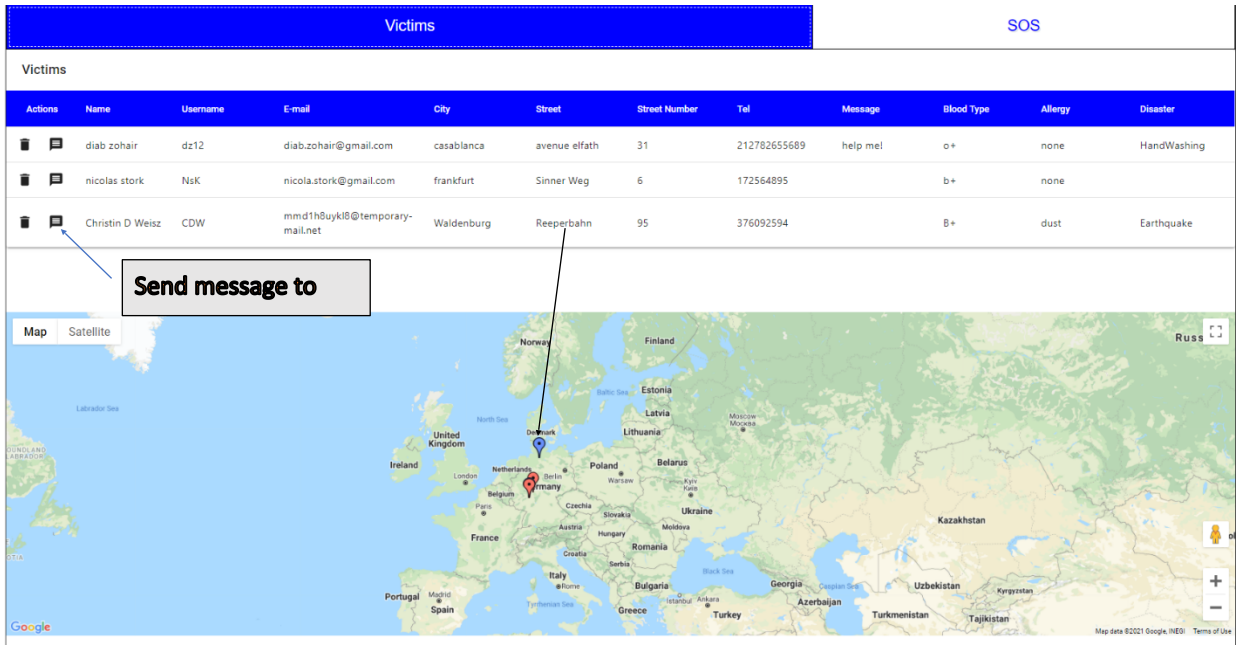
CDW is being used as a username, in this case the message sent directly to rescue helper.



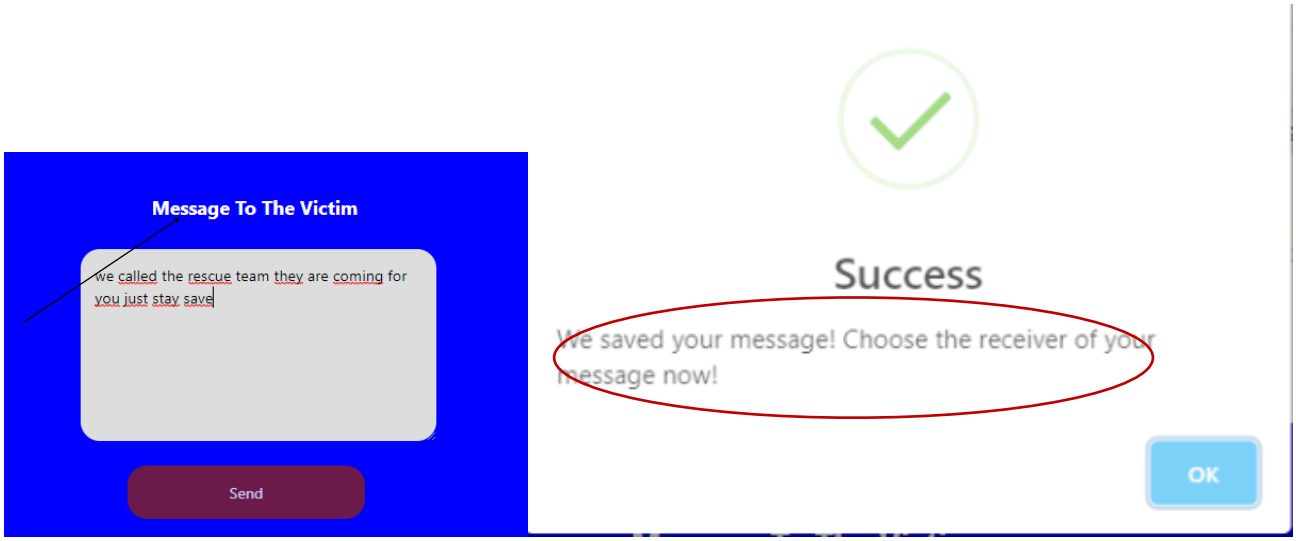


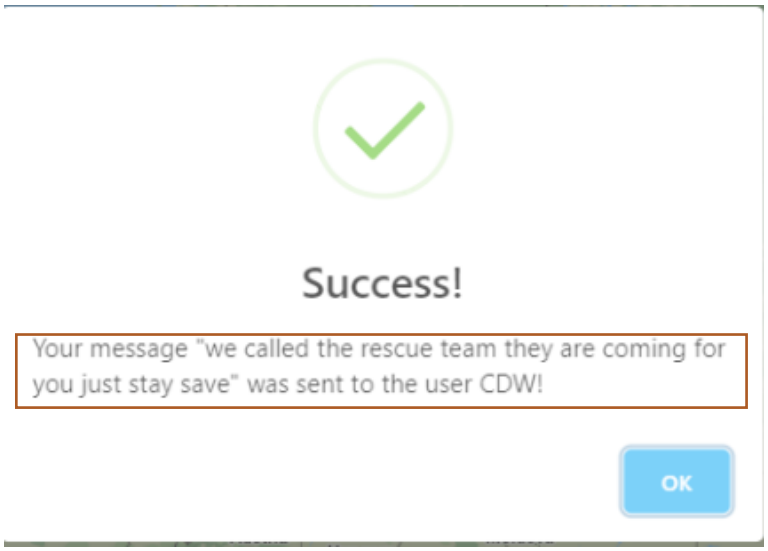
Then comes step 3 where the rescue helper already logged in and checked for the victims table there when saw a new row with a message “help me!” also a blue pin on the map indicates his position.



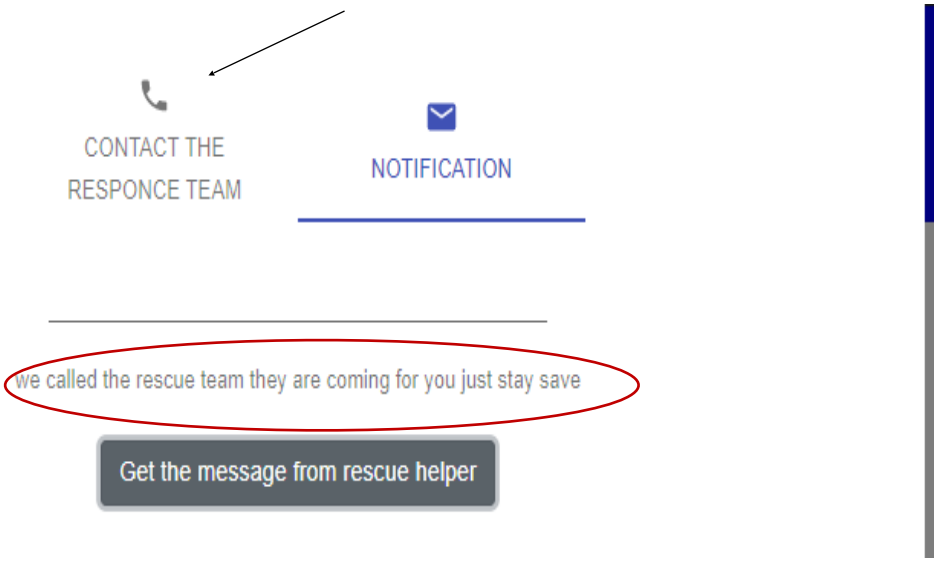


Immediately the rescue helper calls the responsible rescue team to reach the place at the same time he sends a feedback message to confirm the victim that the help is coming. This procedure should be held by clicking on submit then a popup message asks him to choose which receiver must get the message then he scrolls up to the table where there is a button to send a message to Christian. At the end process a popup confirmation must be seen.

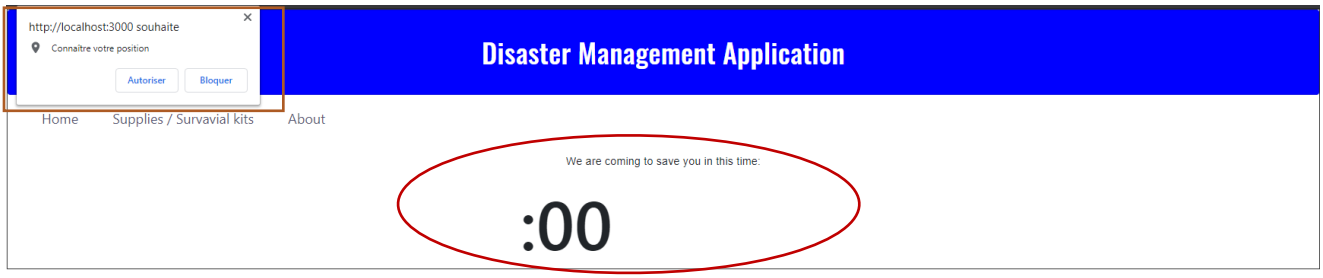




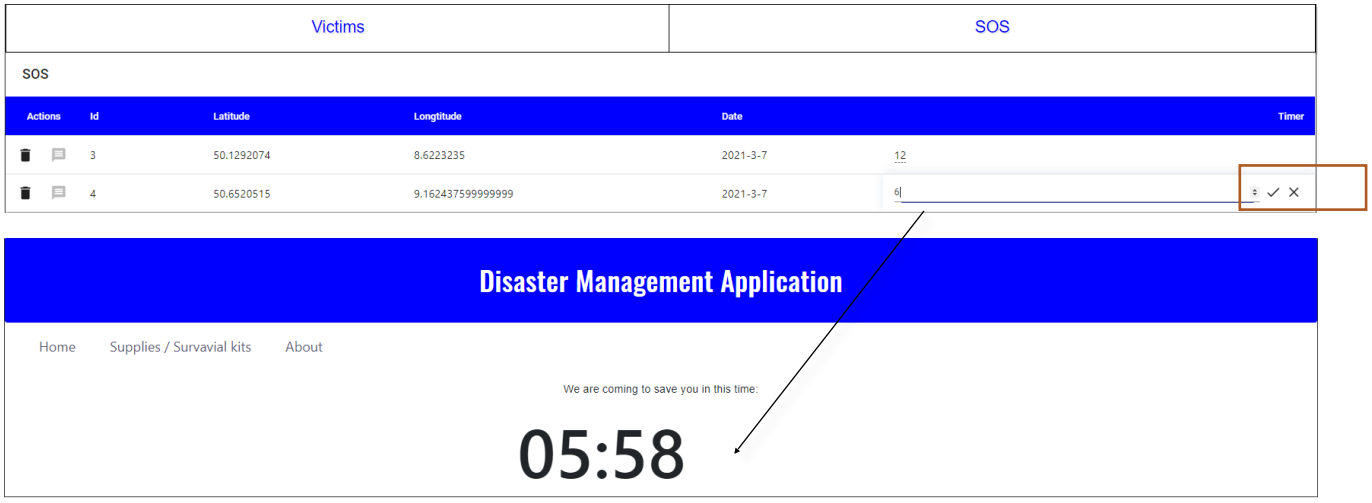
On the other part the victim get notification.



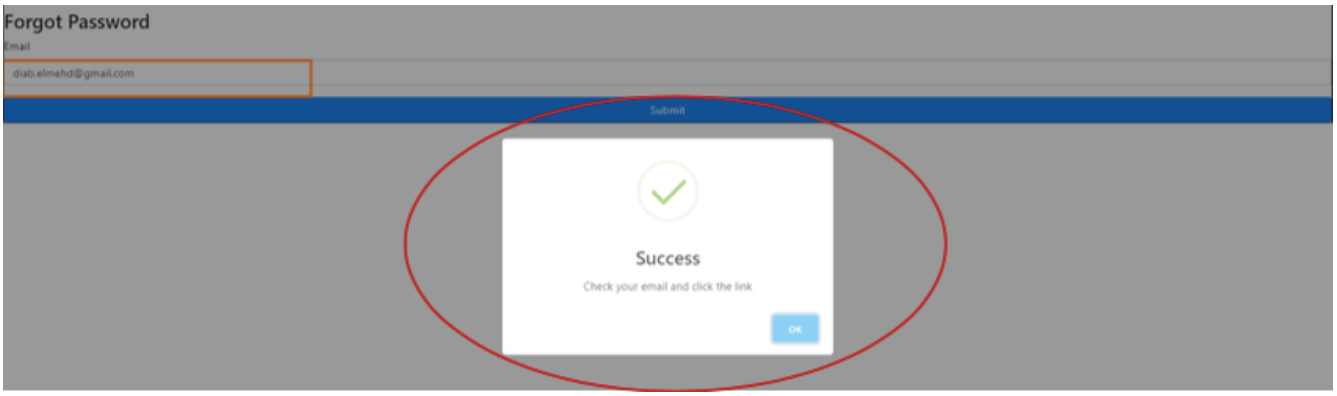
Person 2 is in danger and he wants to access the rescue helper in short time so he clicks on the SOS button then automatically popup told him to share his IP + location then sees a reset timer.



The rescue helper receive the data in SOS table then instantly he calls everyone to go help the person then he estimates the time they suppose to get there and then he activates the timer to inform the person 2 how much time he should hold till the rescue team arrives.



The second part the rescue helper tries to login, but he forgot his own password, in this case he will click the link “forgot password” then it redirects him to this page, while he should submit his email address that he used to sign in.



The next step an automatic link generates and sent to the same email to update his password as it is mentioned bellow:

Conclusion and Perspective

Disaster Management Application shown to be an extremely important and useful tool to support and help victims and emergencies case to reduce or avoid the potential losses from hazards, assure prompt and appropriate assistance to the victims of a disaster, and achieve a rapid and effective recovery. For the future scope, we want to develop the email verification till it can verify only the correct emails.

Bibliography:

https://docs.docker.com/

https://spring.io/guides/tutorials/react-and-spring-data-rest/

https://reactjs.org/docs/getting-started.html

https://material-ui.com/getting-started/

https://www.w3schools.com/

https://www.researchgate.net/publication/278030685_Web_based_Disaster_Management_System_including_Early_Warning_System_and_Online_Public_Complaint_System