CSE 2112 – Object Oriented Programming Lab
Project Report

"RESQURE" Disaster Management System

Submitted To

Dr. Chowdhury Farhan Ahmed
Professor
Md. Redwan Ahmed Rizvee
Lecturer

Submitted By

Group No. 02
Sumaiya Rahman Soma (Roll: 07)
Chowdhury Shafahid Rahman (Roll: 55)

Department of Computer Science & Engineering
University of Dhaka

Table of Contents

1. Introduction

1.1 Project Overview

“RESQURE” is a disaster management system designed to address the needs of citizens, administrators, and responders during emergency situations.

Disasters, both natural and man-made, cause significant damage to life and property. A well-coordinated disaster management system is crucial for responding to emergencies efficiently. The primary objective of this project is to create an application using JavaFX to facilitate disaster reporting, verification, volunteer coordination, and communication in order to improve response efficiency, ensure timely assistance and collaboration during challenging situations.

1.2 Objectives

The objectives of the Disaster Management System are:

- Allow users to report disasters, receive survival guidance, and access emergency contacts.
- To allow administrators to verify disaster reports and take necessary actions.
- To notify volunteers about verified disasters and track their performance and report task completion.
- To facilitate communication between users and the admin through a chat application.

1.3 Features

The Disaster Management System consists of three main roles:

- User: Reports disasters and can communicate with the admin.
- Admin: Verifies disaster reports, manages volunteer tasks, and interacts with users.
- Volunteer: Receives assigned disaster response tasks and can assess personal performance.

User Features:

- Disaster reporting with location and description.
- Chat service to communicate with the admin.
- Get survival tips and important contacts information.

Admin Features:

- Verify and manage reported disasters.
- Assign tasks to volunteers.
- Monitor disaster response progress.
- Communicate with users through chat.

Volunteer Features:

- View assigned disaster response tasks.
- Submit disaster reports from their dashboard.
- See their performance statistics.

1.4 Platform & Tools

To implement object oriented programming and design principles we used java and utilized IntelliJ as our IDE. For database management we used MySQL workbench. We handled compiling and building the source code with Maven, which also managed our project dependencies.
The user interface elements were developed using JavaFX, with designs created using scene builder. Maven's pom.xml was configured to include dependencies for JavaFX, logging and other necessary components. This setup allowed us to automatically download and incorporate the correct library versions, streamlining the development process and ensuring compatibility across different environments.

2. Design & Implementation

2.1 User Interface

The following is a brief overview of the UI of our project.

The Disaster Management System consists of three main roles:

- User
- Admin
- Volunteer

Landing Page:

At first there is a landing page where a user can choose to login or signup.

Figure: Landing Page

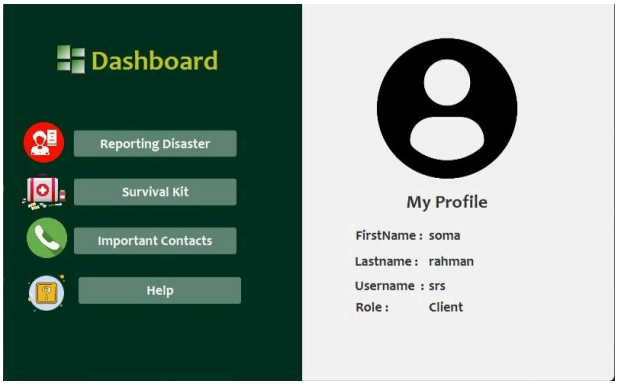
Login & SignUp Page:

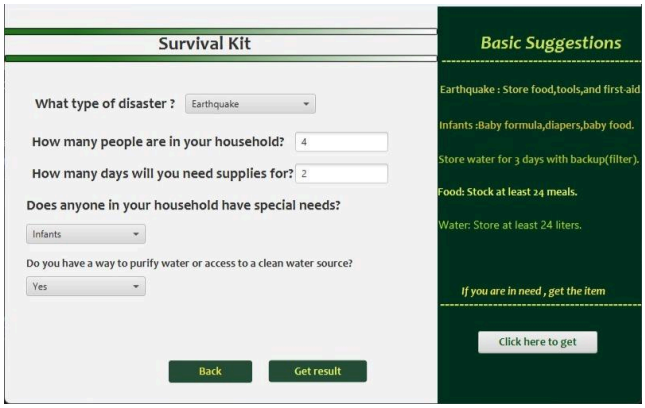
In the signup page, we can sign up for an account as a client or as a volunteer by providing necessary information. This information will be stored in the database. In the login page, we need to provide username and password and then it is authenticated with our database.

Figure: Signup and Login Page

Client Page

The client page has four features reporting disasters, survival kit, important contacts and chat help.





In the reporting disaster section there is a form which a client can fill up and report a specific incident. The report gets logged in a text file which stores all the reported disasters. Additionally this report is stored in the database.
Then if the client needs any information related to survival during a disaster he/she can go to the survival kit section and get information about basic instructions.
There is an important contact page which gives necessary information about individuals who can help people during a disaster.

Admin Page

The admin can login just like the client. The admin page has three sections: dashboard, assign tasks and helpline.

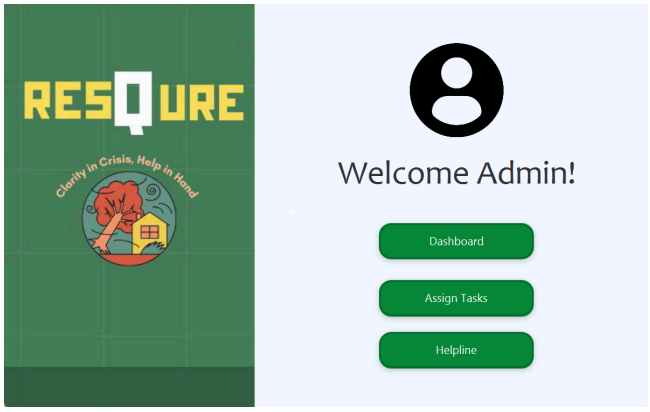


Figure: AdminPage

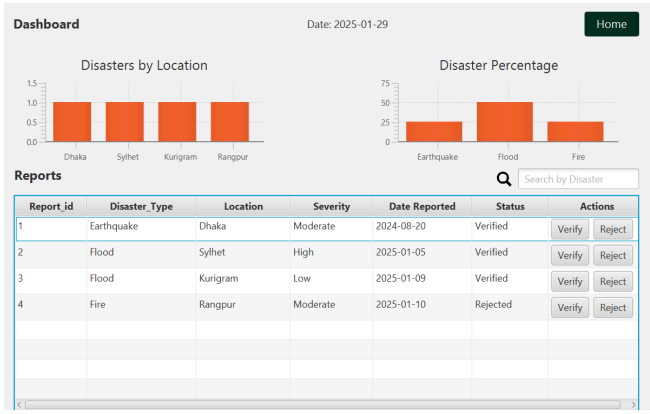


Figure: Admin Dashboard

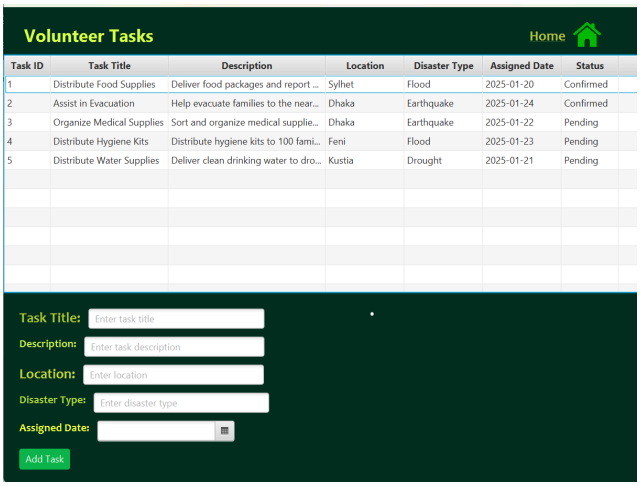
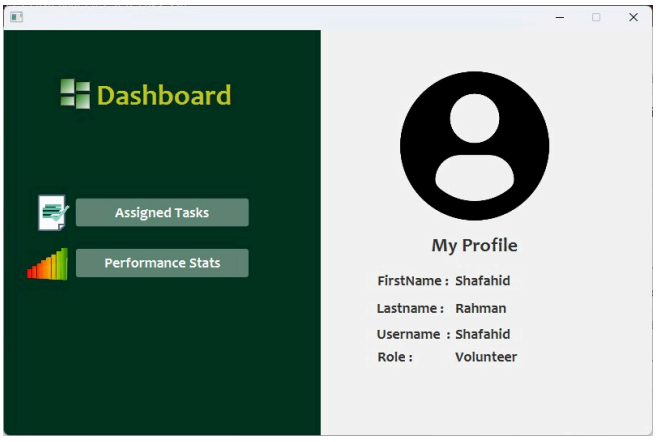


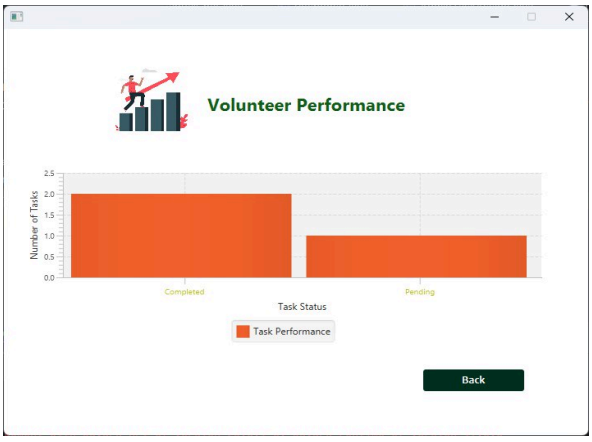
Figure: VolunteerTasks Page

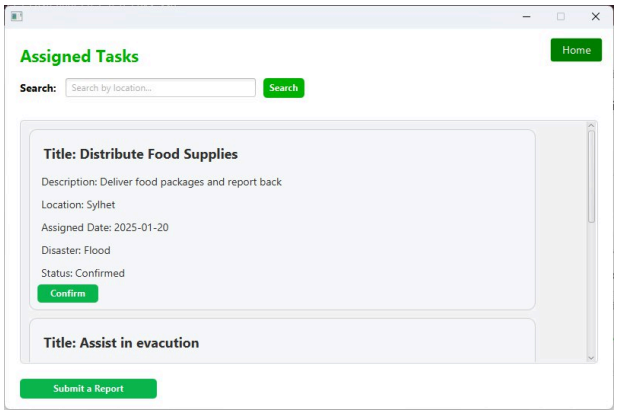
The admin can see all the reports from the client and then check for validity of the information and then verify and update its status in the dashboard page. The statistics of the disasters are shown through a bar chart to get a clear visualisation.
The admin can also assign tasks to the volunteers in response to those reported disasters in the Volunteer task page.

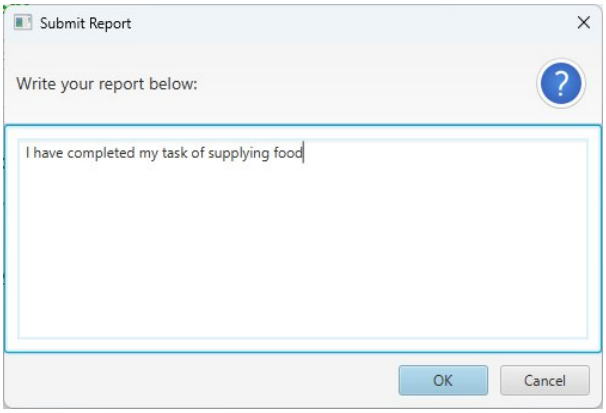
Volunteer Page

The admin can login just like the client. The volunteer page has two sections assigned tasks and performance.





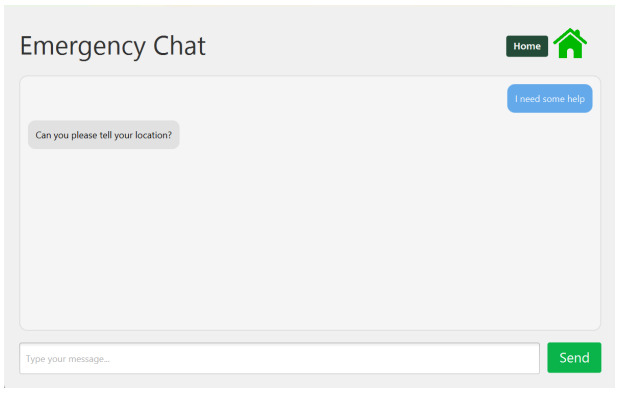


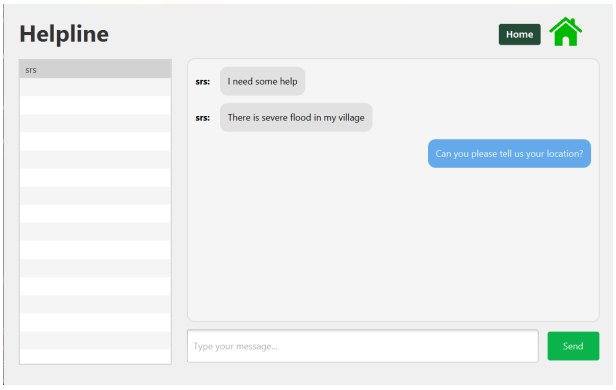


A volunteer can see the tasks assigned and can confirm a task if he/she wants to work on that task. Volunteers can also submit reports to explain their contribution.
We have also delivered the performance of each volunteer to showcase their efforts.

Chat Service

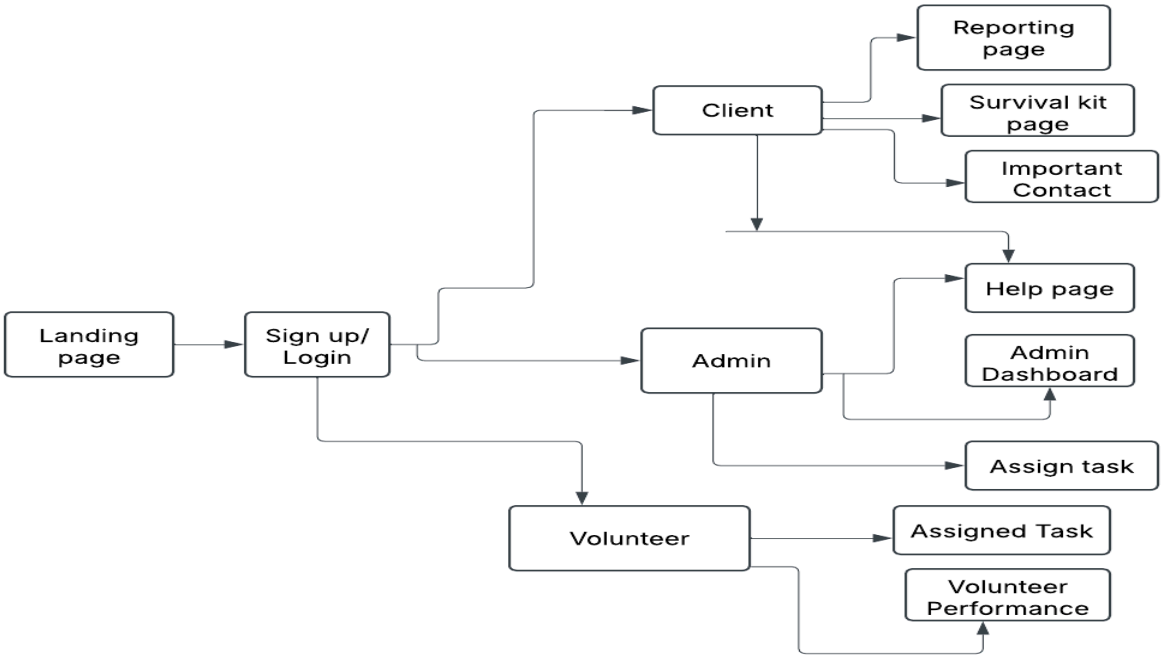
The clients can have real-time chat to communicate with the admin in case of an emergency situation. This ensures seamless communication and information transfer to help take necessary decisions for both administrators and citizens.





We have used socket programming to implement our chat service. Each client can initiate a chat with the admin, and the admin can manage multiple conversations simultaneously in separate interfaces. This ensures that disaster reports, volunteer coordination, and emergency updates can be communicated quickly and efficiently. The system follows a **client-server architecture**, where the client sends messages to the server, and the server routes them to the intended recipient.

2.2 WorkFlow



2.3 Implementation of OOP Principles

The main goal of this project was the implementation of object oriented programming concepts like polymorphism,abstraction,encapsulation and inheritance. We have implemented all those principles in our project as necessary. We created class and extended its functionalities to apply inheritance. For example, a user class can be the parent of Admin, Volunteer, and Client, sharing common attributes like username and password.The DatabaseConnection class abstracts the database connection logic, making it reusable across different components. We also implemented concepts like file handling while creating disaster report files. We have used socket programming to create a Client-Server Architecture for real-time chat service between admin and the client. We used threading to handle multiple clients in different threads.

3. Conclusion

3.1 Challenges

Our vision for the disaster management system was ambitious, aiming to create a platform that can handle the emergency situations in a disaster. However, we faced several challenges while developing this application such as showing data updates, efficient volunteer coordination, and communication between users and administrators.Implementing socket programming for live chat and disaster notifications required careful synchronization to ensure seamless interactions.

3.2 Future Plans

We can enhance this project by improving efficiency and user interface. In the future, we can build automated notification alerts to notify people about the critical situations in real time.We can also enhance the project by adding colorful maps to show where the disaster is, so everyone can see and understand it easily.By building upon the foundation laid thus far, we are confident about our project to make a real life impact for our society.

3.3 Repositories

Github Repository: srs4929/RESQURE-OOP-PROJECT-2-1-

Youtube Video Link: CSE -2112 JavaFx Project : Disaster Management System

3.3 References

1. The JavaFx Documentation (Getting Started with JavaFX)