DIMA- A Disaster Management System

By

NC Abira Abid Bhutta

PC Mariam Arshad

NC Zunaira Tariq

Submitted to the Faculty of Computer Software Engineering

National University of Sciences and Technology, Islamabad in partial fulfillment

For the requirements of a B.E. Degree in Computer Software Engineering

MAY 2017

ABSTRACT

The project aims at developing a Disaster Management System for Military users as well as Government bodies (NDMA), Rescue 1122, police stations, hospitals, Relief agencies, Gated Communities (Bahria Town, DHA, Askaris) , Schools and Colleges. DIMA is a Disaster Management System. It is a GIS based command, control and intelligence (C2i) web application which is operating system independent. We are building a disaster management system which will help its users to navigate location of emergency buildings (such as hospitals, fire stations, police stations etc.) and find the shortest possible route for that location in case of disaster. A rudimentary platform has already been made for a C2I application this will be further enhanced to achieve required functionality for a disaster management system.

Our idea will provide the need for secure maps where data for locations can be saved offline by military and any other organization. It also provides portable intrusion detection system which notifies the user about security breach no matter where the user is, by the use of C2i application.

CERTIFICATE FOR CORRECTNESS AND APPROVAL

Certified that work contained in the thesis – A Disaster Management System carried out by Abira Abid Bhutta, Zunaira Tariq and Mariam Arshad in supervision of Lt.Col Adnan Rashdi for partial fulfilment of Degree of Bachelor of Software Engineering is correct and approved.

Approved by

Lt.Col Adnan Rashdi

CSE DEPARTMENT

MCS

DATED:

DECLARATION

No portion of the work presented in this dissertation has been submitted in

Support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose unflinching support and cooperation, a work of this magnitude would not have been possible.

.

ACKNOWLEDGEMENTS

We would like to thank Allah Almighty for His incessant blessings which have been bestowed upon us. Whatever we have achieved, we owe it to Him, in totality. We are also thankful to our families for their continuous moral support which makes us what we are.

We are extremely grateful to our project supervisor Lt.Col Adnan Rashdi from MCS who in addition to providing valuable technical help and guidance also provided us moral support and encouraged us throughout the development of the project.

We are highly thankful to all of our teachers and staff of MCS who supported and guided us throughout our course work. Their knowledge, guidance and training enabled us to carry out this whole work. Special thanks to Sir Sajjad Sarwar, an alumni of MCS NUST, his mentorship and guidance will help us go a long way.

Finally we are grateful to the faculty of Computer Software Department of the Military College of Signals, NUST.

In the end we would like to acknowledge the support provided by all our friends, colleagues and a long list of well-wishers whose prayers and faith in us propelled us towards our goal.

TABLE OF CONTENTS

TABLE OF FIGURES:

TABLE OF TABLES:

CHAPTER:1

INTRODUCTION

INTRODUCTION

1.1 Overview

DIMA is a Disaster Management System. It is a GIS based command, control & intelligence (C2i) web application which can help in easy navigation in disastrous events & find routes for emergency buildings. It can be easily adopted by the beneficiary since it is a basic web application which shows offline routes and emergency locations so it does not even require internet and any army personnel can use it anywhere. It also notifies the detection of intruder and has an edge because of its portability

DIMA is an offline application which uses map tiles available on the server. However it can also work in an online mode if need be. The application also uses a GPS sensor to fetch the coordinates of the current location of the user. User can navigate location of emergency buildings and find the shortest route for any location. Only authorized users will be able to access the application and integration of C2i application will help generate alerts on DIMA in case of security breach.

1.2 Problem Statement

We are building a disaster management system for Pakistan Army which will help the military officials to navigate location of emergency buildings (such as hospitals, fire stations, police stations etc.) and find the shortest possible route for that location in case of disaster. However, this application can also be used by commercial users, government bodies and most importantly by NDMA (National Disaster Management Authority) Pakistan.

1.3 Approach

Our idea will provide the need for secure maps where data for locations can be saved offline by military and any other organization. It also provides portable intrusion detection system which notifies the user about security breach no matter where the user is, by the use of C2i application.

1.4 Scope

The project aims at developing a System for people facing disastrous situations. It is to make navigation easy in case of a disaster and assist the users to find the shortest possible route. We are building a disaster management system for Pakistan Army which will help the military officials to navigate location of emergency buildings (such as hospitals, fire stations, police stations etc.) and find the shortest possible route for that location in case of disaster. However, this application can also be used by commercial users, government bodies and most importantly by **NDMA (National Disaster Management Authority)** Pakistan.

1.5 Aim & Objectives

The objectives of project include:

- Using software engineering techniques for gathering requirements during the development process, designing the software, implementing and testing requirements gathered.
- To learn java programming and SDLC
- To learn database design and development
- To learn data transfer through networking**.**

1.6 Contributions

The project has been qualified for FICS (Finding Innovative and Creative Solutions) where it has cleared the first two stages.

This is an industrial project designed and developed for R&D MCS.

1.7 Organization

The first part of thesis is the abstract which describes the main details of DIMA System, followed by the introduction section which specifies the problem statement, approach, scope and objectives. The literature review section state the various resources read online before the commencement of the project. They include learning about the disaster management system. The design and development part illustrate the diagrams which describe the detailed design of the DIMA System its components, interfaces and data necessary for the implementation phase. The analysis and evaluation part give details of the black box testing, unit testing and system integration testing; actual results against expected results. The future work gives states the enhancements that can be applied to the application.

1.8 Deliverables

Table 1-1: Deliverables

Deliverable Name	Deliverable Summary Description
Software Requirements Specification(SRS) Document	Complete Description of **what** the system will do, who will use it. Detailed description of functional and nonfunctional requirements and the system features.
Design Document	Complete description of **how** the system will be implemented i.e. the detailed design.
Code	Complete code.
Testing Document	The whole system is tested according to the specification described in the SRS document. Black box, unit and System integration testing is done.
Complete System	Complete working system.

CHAPTER: 2

LITERATURE REVIEW

2 LITERATURE REVIEW

Pakistan is one of the most disaster prone countries in the world. Generally divided into natural and man-made, all disasters are managed by a systematic process of disaster management that aims at minimizing the damage and restoration of people to their normal state. Pakistan is well familiar with disasters which have caused a heavy toll in terms of men and material.

However, due to its inadequate preparedness to manage disasters, it has failed to effectively cope with them. Though, after earthquake-2005, a systematic effort was geared up to develop a viable structure of disaster management evolving into establishment of NDMA, it has yet to achieve the required standards. The heavy floods of 2010 exposed its unpreparedness and frail management resulting in unprecedented proportion of losses and damages. Since, the magnitude of implications is too heavy to bear; the efficient disaster management comes, on the priority, second to none of other needs. Therefore, it is necessary to formulate an organized disaster management system to cope with disasters that may break out in future.

2.1 NDMA

However, despite establishment of National Disaster Management Authority (NDMA) the response was too slow to meet the magnitude of challenge. The purpose behind its establishment was to change national response to emergency situations from reactionary model to an active mitigation, preparedness, response and recovery model. NDMA is the executive arm of the National Disaster Management Commission (NDMC) headed by Prime Minister. Also, NDMA supervises Provincial Disaster Management Authorities (PDMA) and District Disaster Management Authorities (DDMA).

The National Disaster Management Authority has been assigned the task of coordinating the disaster risk management at the national level, implementing disaster risk management strategies, mapping the hazards, developing guidelines, ensuring the establishment of disaster management authorities and Emergency Operation Centers (EOCs) at provincial, district and municipal levels, providing technical assistance to concerned departments, organizing training to personnel, serving as a lead agency for NGOs and international cooperation, coordinating with the federal government through National Emergency Operation Centre (NEOC) and requiring any government department or agency to make available needed resources and personnel.

Despite establishment of this organization assigned with apparently multifarious tasks, disasters in Pakistan are hardly managed effectively. Its preparedness and response during recent floods were found inadequate. “For 10 days, the flooding was only in this province. But we didn't hear from the NDMA and nor did we see any NDMA official. No one even contacted us,” said Khyber-Pakhtunkhwa Information Minister Iftikhar Hussain.

2.1.1 DIMA

DIMA is a Disaster Management System with a command and control center that administers all the disastrous events on a single platform. It uses offline maps so the availability of internet is not necessary.

2.2 DIMA novel functionality:

2.2.1 Single Platform:

It brings all the rescue forces on one single platform i.e. hospitals, police stations and fire stations, providing necessary information about these stations.

2.2.2 Nearest Building:

It also provides information related to the nearest building available in case of disaster.

2.2.3 Addition of new Assets:

The system helps in the addition of new assets such as hospitals, police stations and fire stations and also allows tagging that location onto the map.

CHAPTER: 3

OVERALL DESCRIPTION

OVERALL DESCRIPTION

This part of the document contains information about the product, its features, perspective, user's characteristics and constraints.

3.1 Introduction

This document serves the purpose of providing information needed to adequately design and implement DIMA - a Disaster Management System, a GIS based command, control and intelligence (C2i) web application which is operating system independent.

3.1.1 Purpose

It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react. This document is intended for both the stakeholders and the developers of the system.

This document is meant to outline the features and requirements of DIMA, so as to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other.

3.1.2 Intended Audience and Reading Suggestions

The intended audiences for the DIMA include the project supervisor, the BESE 19 FYP group (developers), UG project evaluation team, and other persons at MCS CSE Department.

1. **Project Supervisor:**

It will help the supervisor to supervise the project and guide the team in a better way. This document will be used by her to check whether all the requirements have been understood and in the end whether the requirements have been properly implemented or not.

2. **BESE 19 FYP group (developers, testers, and documentation writers):**

For FYP group members, this document will provide the guideline for developing and testing the project.

3. **UG Project Evaluation Team:**

It will help the evaluation team to evaluate the progress of FYP project. The document will provide the evaluators with the scope, requirements and details of the project to be built. It will also be used as basis for the evaluation of the implementation and final project.

Reading suggestions:

The SRS begins with the title and table of contents. All level 1 and level 2 headings are given in the table of contents, but the lower sub headings are not included. Each main heading is succeeded by a number of sub headings, which are all in bold format. The product overview is given at the start, succeeded by the complete detailed features, including both functional and non-functional requirements. The entire interfaces are also described. The SRS ends with appendices, including a glossary.

3.1.3 Product Scope

The document only covers the requirement specifications for DIMA. This application will be used by military personnel in hard areas to navigate around, and to analyze locations not only by providing them with a list of the nearby locations of a particular category but also by giving sufficient information about any location, which will help them in analyzing the situation around them. For convenience and security purpose application features are mostly available in offline mode, however, additional functionality can be added using the online mode.

For	Research & Development Wing MCS-NUST
What	To provide location and navigation services to military personals or commercial users along with the following features: ● Information about routes(min route, max route) ● Satellite view of important locations (hospitals, fire stations, police stations etc.) ● Alerts in case of security breach

	● Geotagging
The	C2i version of Google Maps
Is	An extension of C2i systems
That	Provides: ● Assistance in disastrous situations ● Navigation ● Security

3.2 Overall Description

3.2.1 Product Perspective

DIMA is basically an extension of C2i which handles catastrophic situations by providing the locations of nearby facilities in the form of offline maps in hard areas which can be a better alternative than Google Maps since the availability of internet connection is not required for navigation.

3.2.2 Product Features

Following are the salient features:

1. Location tagging.
2. Viewing and navigation to the tagged location.
3. Displaying current location through GPS tracker.
4. Information about routes (min route, max route)
5. Provision of satellite view of all emergency locations.
6. Alerts in case of security breach.
7. Provides user's current location.
8. Provides user the capability of adding more information about a location.

The figure below shows the block diagram of DIMA application:

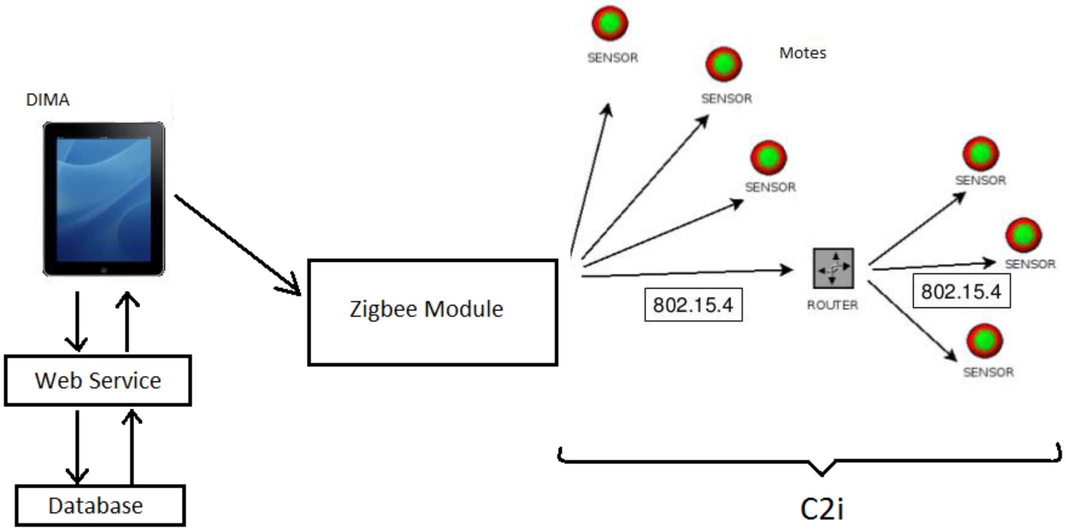


Figure 3.2.2-1: Block Diagram of DIMA

3.2.3 User Classes and Characteristics

The application can be used by customers interested in having a much secure version of Android Lollipop with minimal security breaches.

The different types of users are:

1. Pakistan Army (frequent user):

This Disaster Management System will be used by Pakistan Army in hard areas to navigate and find routes of emergency buildings.

2. Intelligence Agencies (frequent user):

Intelligence Agencies can also use DIMA in cases of emergencies and the accessed can be limited to authorized users only.

3. Government Bodies (occasional user):

Government bodies and commercial users can use DIMA in case of emergencies.

4. Tester (occasional user):

The testers will use the product at the time of the penetration testing to check the flaws of the product.

3.2.5 Design and Implementation Constraints

- For military purposes, DIMA will only be used on the authorized devices however this limitation is not necessary for commercial usage of DIMA.
- The shortest distance found by the application might not be highly accurate.
- The synchronization of local databases may be very difficult.
- There might be overlapping in the defined categories such that two buildings might fall in the same category.
- Since we need a remote web-server to ensure mobility, its maintenance would be hard.
- DIMA generates alerts in case of security breach, however, it does not guide the user how to deal with that alert.

3.2.6 User Documentation

Following are the guides for DIMA:

- Usage manuals with pictures and text for using the software

3.3 External Interfaces Requirements

3.3.1 User Interfaces

- User interface will contain:
- A login screen
- A screen with the map display and locations.
- A screen for marker addition(geotagging)
- A tab for C2i

3.3.2 Hardware Interfaces

- GPS Sensor will be used to detect the current location of the device.
- Communication between DIMA application and ZigBee module for alerts in case of security breach.

3.3.3 Software Interfaces

- Web service will act as an interface between the front end of the application and the server.
- The web service being used is REST web service.
- The database used for the Devices will be MySQL Workbench.

3.3.4 Communications Interfaces

- Application will be used to communicate with the GPS Sensor to detect the current location.

- No external hardware and software communication is being done.
- Alerts will appear whenever the system suspects a security breach.

3.4 System Features

This section describes in detail the system features of the DIMA. System features are classified into following use cases:

1. Addition of a new location:

The application shall help the user to add the location in the database.

2. Tagging that location on the map:

The application will assist in tagging the marker of the saved location in the database on the map.

3. Display of current location:

The application will also generate the current location of the user with the help of GPS sensor.

4. Category wise search:

The application will help in the category wise search which filters the data according to the user requirements.

5. Addition of a sub category in a category:

The application will also help in the addition of a sub category in a category to specialize the database.

6. Searching of routes:

The application will also assist in finding routes by providing additional information of the shortest possible route to a certain location.

7. Alert generation on security breach:

The application will also generate alerts on security breach with the support of the C2i application.

8. Adding/viewing information under a subcategory:

The application will also help to add and view information under a subcategory to provide details regarding a location.

This is the overall structure of DIMA:

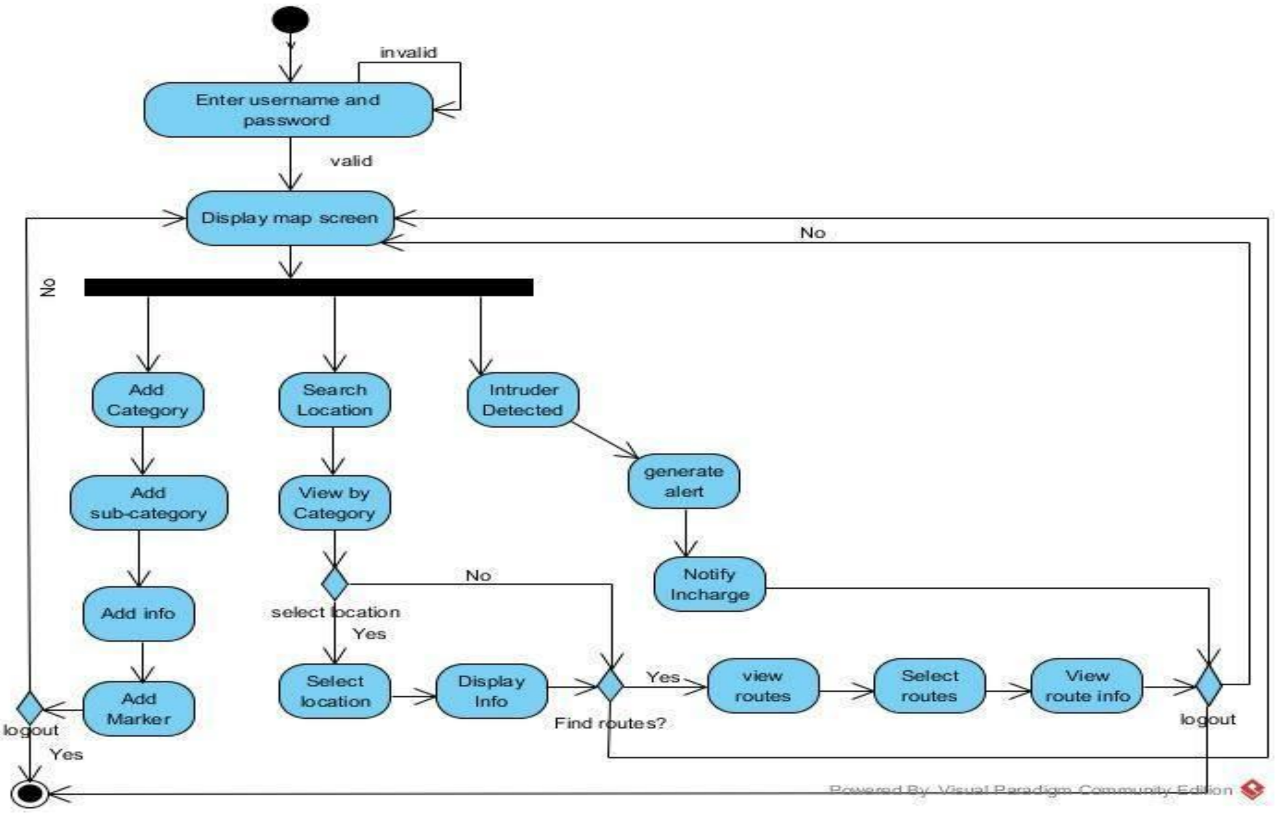


Figure 3-1: Structure of DIMA

3.5 Functional Requirements:

REQ-1: The system shall display marker on the given central point.

REQ-2: The system shall allow the user to add category wise locations on the map.

REQ-3: The system shall allow the user to tag the added location on the map.

REQ-4: The system shall allow the user to enter subcategory under an already added category.

REQ-5: The system shall give information about routes within the user's vicinity.

REQ-6: The system shall generate alerts in case of security breach.

REQ-7: The system shall allow the user to locate all the emergency buildings (hospitals, police stations, fire stations etc.) in the vicinity.

3.6 Nonfunctional Requirements

3.6.1 Safety Requirements

The use of the software product has no harms whatsoever; nor does it have any possibility of loss or damage that might be inflicted. If the app crashes during addition, deletion or editing there will be no change in the database.

3.6.2 Security Requirements

- The application should be password protected on different levels of security.
- The application shall only be accessed by authorized devices.

3.6.3 Performance Requirements

System should notify the user within 2 seconds of the security breach.

3.6.4 Software Quality Attributes

1. Usability

The graphical user interface of app is to be designed with usability as the first priority. The app will be presented and organized in a manner that is both visually appealing and easy for the user to navigate.

2. Interoperability:

Since DIMA is a web-based application therefore it is operating system independent and can be used on Windows, Linux etc.

3. Accuracy

To ensure reliability and correctness, there will be zero tolerance for errors in the algorithm that computes results.

4. Portability

DIMA is portable and can be accessed on all devices that have been authorized.

5. Availability

The application will be available 24/7.

6. Flexibility

The design and architecture of the application will be flexible enough for catering any new requirements that come at any later stage.

7. Data Integrity

If the app crashes during addition, deletion or editing there will be no changes.

8. Scalability

The application is expected to handle one user at a time. One instance of the application could be opened on a single device at a time.

3.7 Business Rules

- This product is being developed for R&D MCS.
- Developers have right to keep the modules in the product for later use.

3.8 Other Requirements

- Newer Versions of the product will support previous database.
- Data for DIMA system will be made available by R&D MCS to the development team.
- Qualified Army officials from R&D MCS will provide required assistance to development team.

CHAPTER:4

DESIGN AND DEVELOPMENT

DESIGN AND DEVELOPMENT

4.1 INTRODUCTION:

DIMA is a Disaster Management System. It is a GIS based command, control and intelligence (C2i) web application which is operating system independent. A rudimentary platform has already been made for a C2I application this will be further enhanced to achieve required functionality for a disaster management system.

4.1.1 Purpose of the document:

This chapter describes the architecture and system design of DIMA (Disaster Management System).It mostly contains different design diagrams and their explanation. The document is intended to inform stakeholders of the details of the design and the design process. This document will help the developer(s) in implementation and maintenance of the Application (app).

4.1.2 Scope of the Development Project

We are building a disaster management system for Pakistan Army which will help the military officials to navigate location of emergency buildings (such as hospitals, fire stations, police stations etc.) and find the shortest possible route for that location in case of disaster. However, this application can also be used by commercial users, government bodies and most importantly by **NDMA (National Disaster Management Authority)** Pakistan.

4.1.3 Definitions, acronyms, abbreviations

- **DIMA:** Disaster Management System
- **C2i:** Command, control and intelligence

- **NDMA:** National Disaster Management Association
- **UI:** User Interface
- **UML:** Unified Modelling Language
- **Fig:** Figure
- **DB:** Database

4.1.4 Overview of the document

This chapter shows the design and working of DIMA. It starts from higher level details for a non-technical reader to understand just by seeing the diagrams to the lower level details that aid the developer to code and understand other technical details of the application

.Section 2 the **System Architecture Description** gives a detailed overview of the application. Section 2.1 **Structure and Relationships** shows the higher level details system working by the means of System Block, Activity, State Transition, and Use Case diagrams. Lower level details are described using the Class, Chen's Entity Relationship, Sequence diagrams and Structure Chart. Section 2.2 describes how the application is designed to curb the tendency of **User Interface Issues** and problems during User Interaction.

In Section 3, **Detailed Description of Component** is given to show the working of modules with low level details. It shows the purpose, function, subordinates, dependencies, interfaces, resources, processing and data of the components and their relationships with each other.

Section 4 shows the **Reuse and Relationship to other Products** i.e.; information about work done in the same project before and any reuse of the same work. The section also provides a key to reuse this system for further upgrades.

Section 5 **Design Decisions and Tradeoffs** shows the architecture style and design pattern of the application, while in the Section 6 the **Pseudo Code** of the components is given in for human reading rather than machine reading. **Appendices** are included in section 7.

4.2 Work Breakdown Structure:

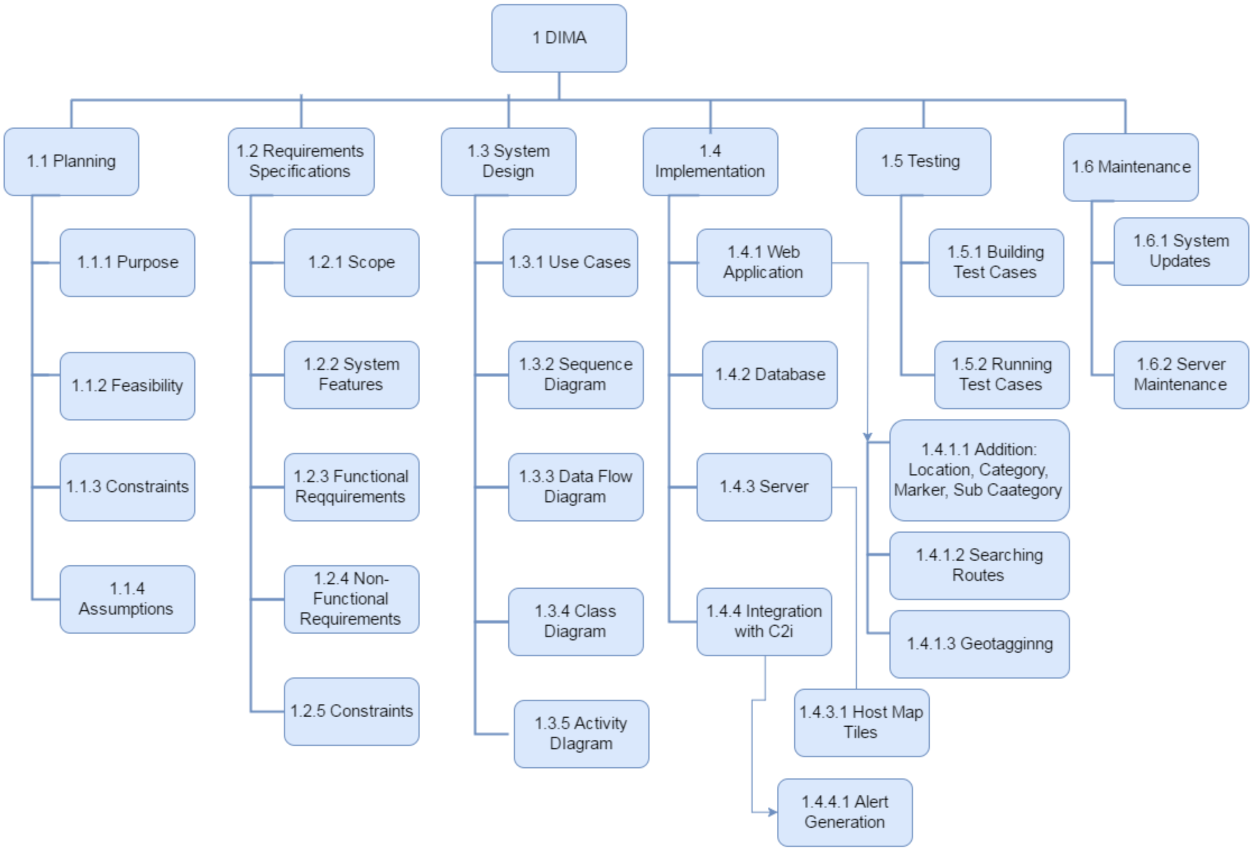


Fig 4.2-1: Work Breakdown Structure

4.3 System Architecture Description

Detailed description of system architecture and design pattern which this system is going to use is discussed later in the document in section 5 ‘Design Decisions and Tradeoffs’.

This Section overview of application, its higher and lower levels details and user interfaces.

4.4 Structure and Relationships:

This section covers the overall technical description of DIMA (refer Section 1.3). It shows the working of application in perspective of different point-of-views and also shows relationships between different components.

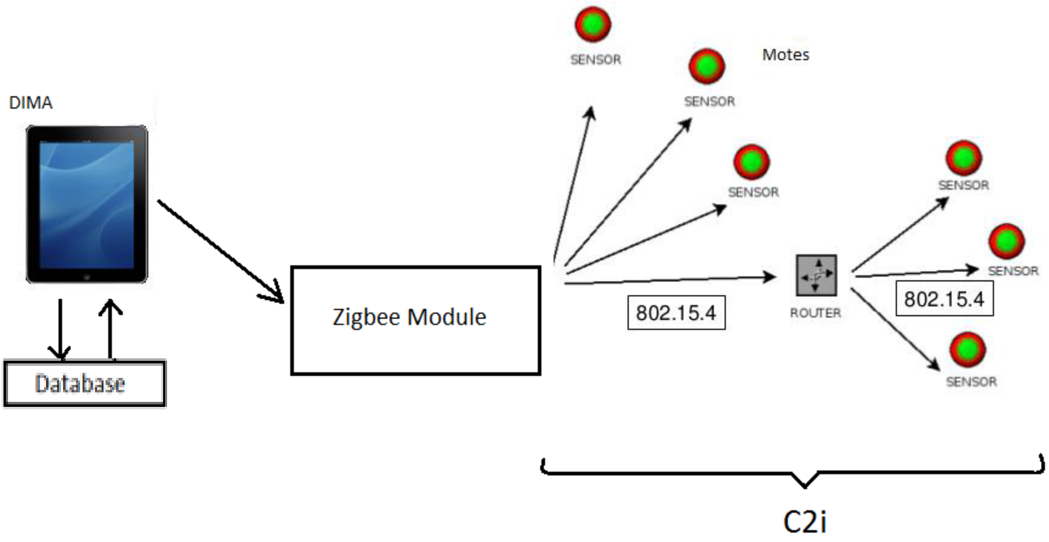


Figure 4.4-1: DIMA Application

4.4.1 System Block Diagram

This diagram shows the higher level description of the application. It shows generic working of the application and interaction with the user. In this application the user will interact with the DIMA Application. The application UI allows the user to add location, category, marker and sub-category. It provides geotagging, filtered data search, searching of routes and alert generation. The C2i application sends input to DIMA Application UI in case of security breach which in turn generates alert on DIMA Application. The server application hosts map tiles and returns record from database to Application UI.

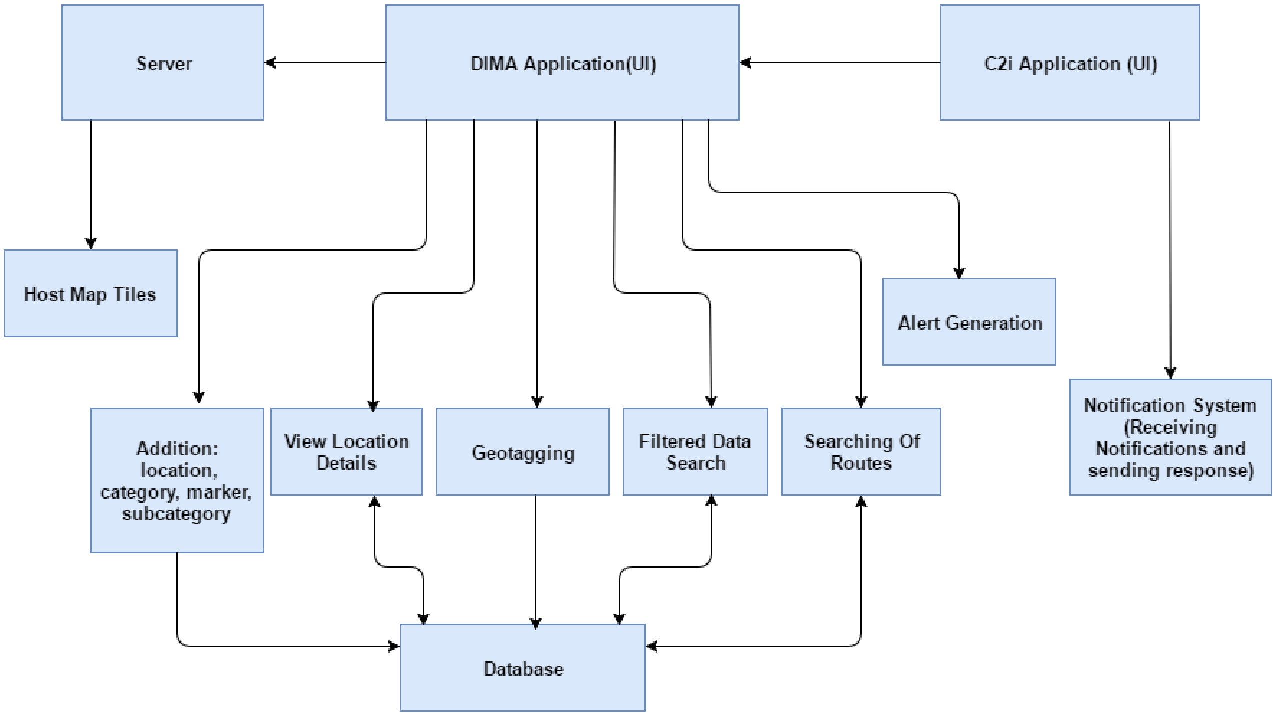


Fig 4.4.1-1: Block Diagram

4.4.2 User View (Use Case Diagram)

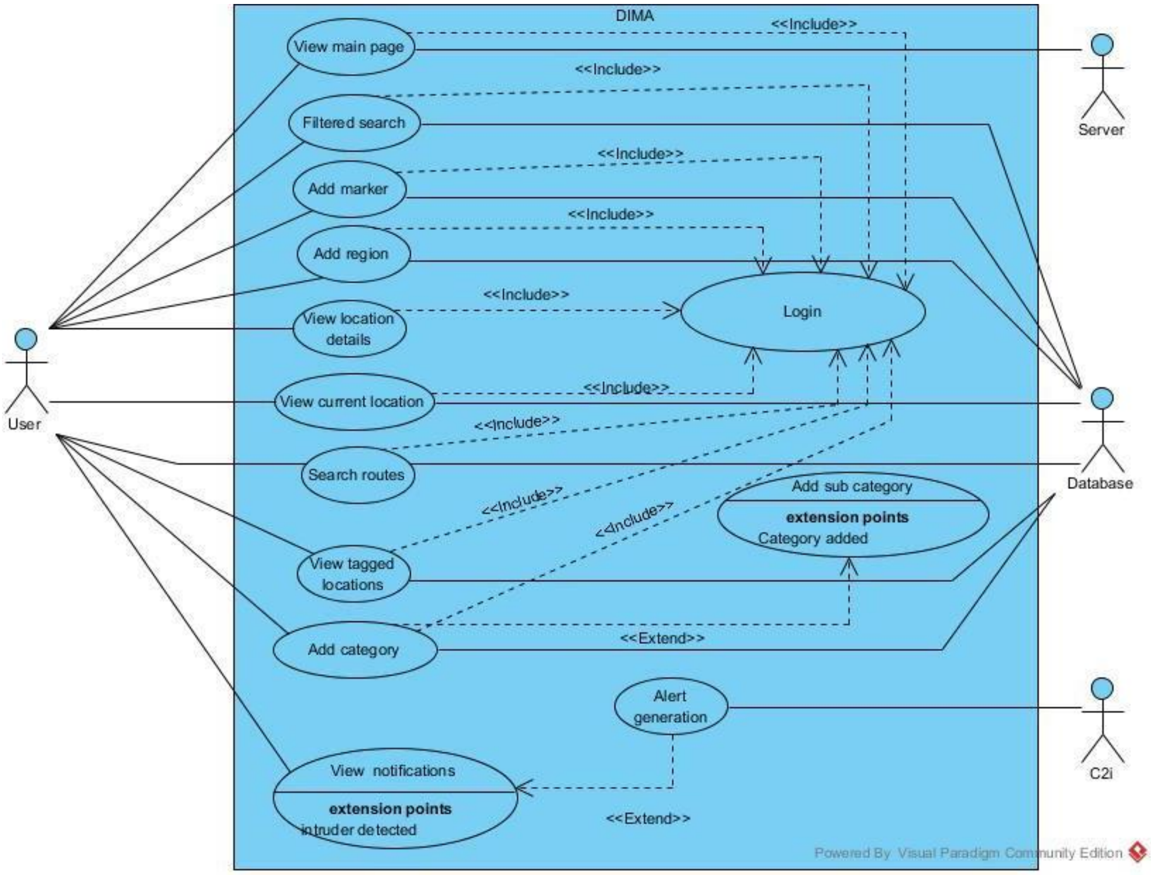


Fig: 4.4.2-1: Use Case Diagram

Actors:

- Military Users
- Commercial Users

Use Cases:

- Addition of Region
- Add marker/geotagging
- View current location
- Filtered search
- Addition of a category
- Searching of routes
- Alert generation on security breach

- Viewing location details

Use Case Description:

- **Use Case 1**

Name: Addition of Region
Use Case Requirement: The application will help the user to add the region in the database.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: Region added ● Exceptional: Error is generated
Normal Path: Location Added
Preconditions: The region being added does not exist.
Interactions: The region added is stored in the database.
Post conditions: The region added can be searched and viewed on the map.
Exceptional Path: Error is generated
Preconditions: The region being added already exists.
Interaction: An error message is sent by the system.
Post conditions: The beep sound is produced through the speakers of the computer.
Extends: N/A
Includes: Login

- **Use Case 2**

Name: Add marker/Geotagging
Use Case Requirement: The application will assist in tagging the marker of the saved location in the database on the map.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: Successful tagging of location. ● Exceptional: Error is generated

Normal Path: Successful tagging of location
Preconditions: The location being tagged already exists in the database.
Interactions: The location being tagged is clicked on the map by the user and information is added on that location.
Post conditions: The information is stored on the location which was tagged.
Extends: N/A
Includes: Login

- **Use Case 3**

Name: View current location
Use Case Requirement: The application will also generate the current location of the user with the help of GPS sensor.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: Display of the current location ● Exceptional: Error in the GPS tracker
Normal Path: Display of the current location
Preconditions: The GPS sensor is working perfectly.
Interactions: The GPS return the current location status to the system.
Post conditions: The map should display the current location status.
Exceptional Path: Error in the GPS sensor
Preconditions: The GPS sensor is not working correctly.
Interactions: The GPS does not return any information about the current location to the system.
Post conditions: Error message is generated on the screen.
Extends: N/A
Includes: Login

- **Use Case 4**

Name: Filtered search
Use Case Requirement: The application will help in the category wise search which filters the data according to the user requirements.
Actor(s):

● Military Users ● Commercial Users
Use Case Paths ● Normal: Displays information category wise ● Exceptional: Generates error message
Normal Path: Displays information category wise
Preconditions: The category already exists.
Post conditions: The search is filtered out category wise.
Exceptional Path: Generates error message
Preconditions: The category does not exist.
Post conditions: Error generated and beep sound is produced.
Extends: N/A
Includes: Login

- **Use Case 5**

Name: Addition of a category
Use Case Requirement: The application will also help in the addition of a category to specialize the database.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: ● The category is added ● The sub-category is added in a category.
Normal Path: The category is added
Preconditions: The region should be selected to add a category.
Post conditions: The category is added under a region.
Extends: Addition of subcategory
Includes: Login

- **Use Case 6**

Name: Searching of routes

Use Case Requirement: The application will also assist in finding routes by providing additional information of the shortest possible route to a certain location.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: The information about the searched route is displayed. ● Exceptional: Error message is generated
Normal Path: The information about the searched route is displayed
Preconditions: The location whose route information is needed, is selected.
Post conditions: The route information from the current location to the given location is displayed.
Exceptional Path: Error message is generated
Preconditions: The GPS is not working perfectly.
Post conditions: Error message is generated and beep sound is produced.
Extends: N/A
Includes: Login

- **Use Case 7**

Name: Alert generation on security breach
Use Case Requirement: The application will also generate alerts on security breach with the support of the C2i application.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: The C2i application will generate alerts on security breach. ● Exceptional: Error will be generated in case of C2i system failure.
Normal Path: The C2i application will generate alerts on security breach.
Preconditions: The C2i application will detect the intrusion.
Interactions: The DIMA system will interact with the C2i system in case of security breach.
Post conditions: Alerts will be generated on the DIMA system.
Exceptional Path: Error will be generated in case of C2i system failure.
Preconditions: The C2i application encounters a system error.

Interaction: The DIMA system will not interact with the C2i system in case of security breach.
Post conditions: Error message is generated and beep sound is produced.
Extends: View notifications
Includes: Login

- **Use Case 8**

Name: View location details
Use Case Requirement: The application will also help to view information under a subcategory to provide details regarding a location.
Actor(s): ● Military Users ● Commercial Users
Use Case Paths ● Normal: Information is viewed under a sub-category.
Normal Path: Information is viewed under a sub-category.
Preconditions: The sub-category containing information should be selected.
Post conditions: Information regarding that category is generated.
Extends: N/A
Includes: Login

4.4.3 Sequence Diagram

The sequence diagram of the DIMA is given below:

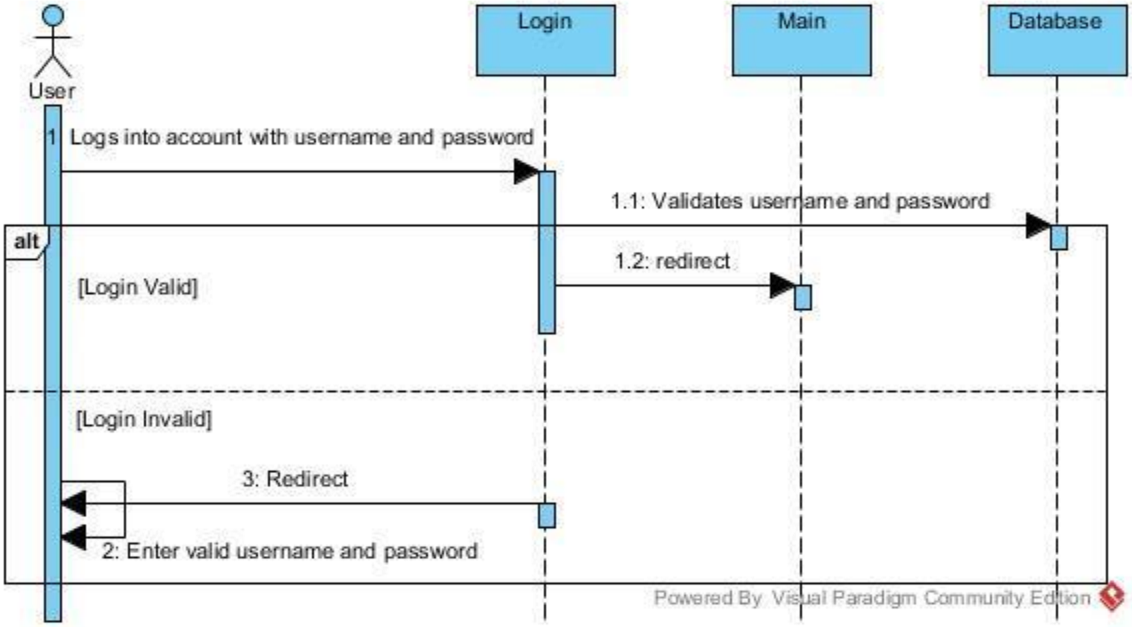

User
Login
Main
Database
1 Logs into account with username and password
1.1: Validates username and password
alt
[Login Valid]
1.2: redirect
[Login Invalid]
3: Redirect
2: Enter valid username and password
Powered By Visual Paradigm Community Edition

Fig 1: Login

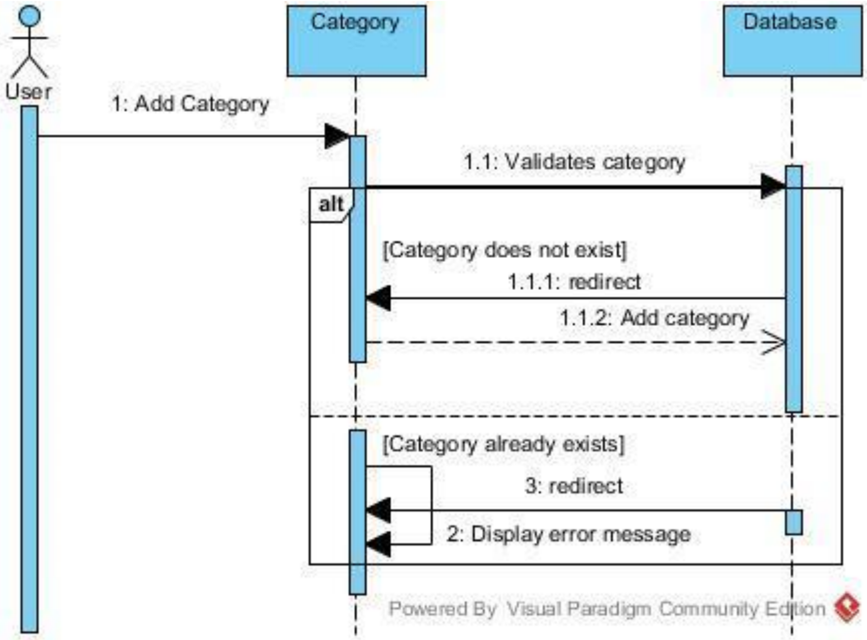


Fig 2: Add Category

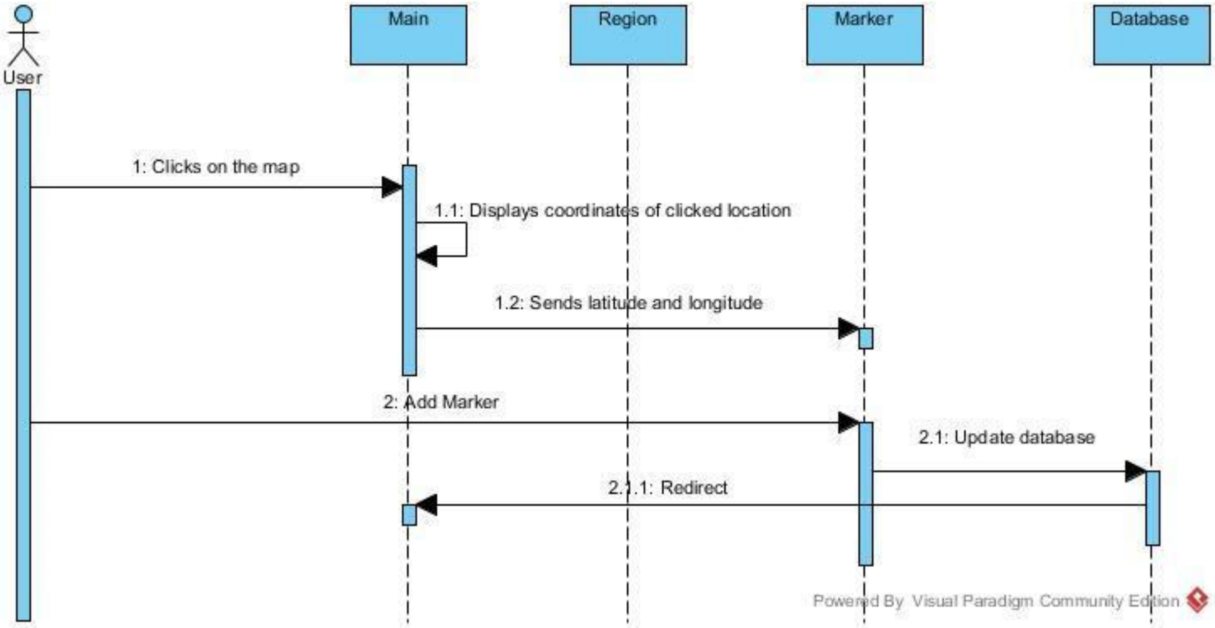

User
Main
Region
Marker
Database
1: Clicks on the map
1.1: Displays coordinates of clicked location
1.2: Sends latitude and longitude
2: Add Marker
2.1: Update database
2.1.1: Redirect
Powered By Visual Paradigm Community Edition

Fig 3: Add Marker

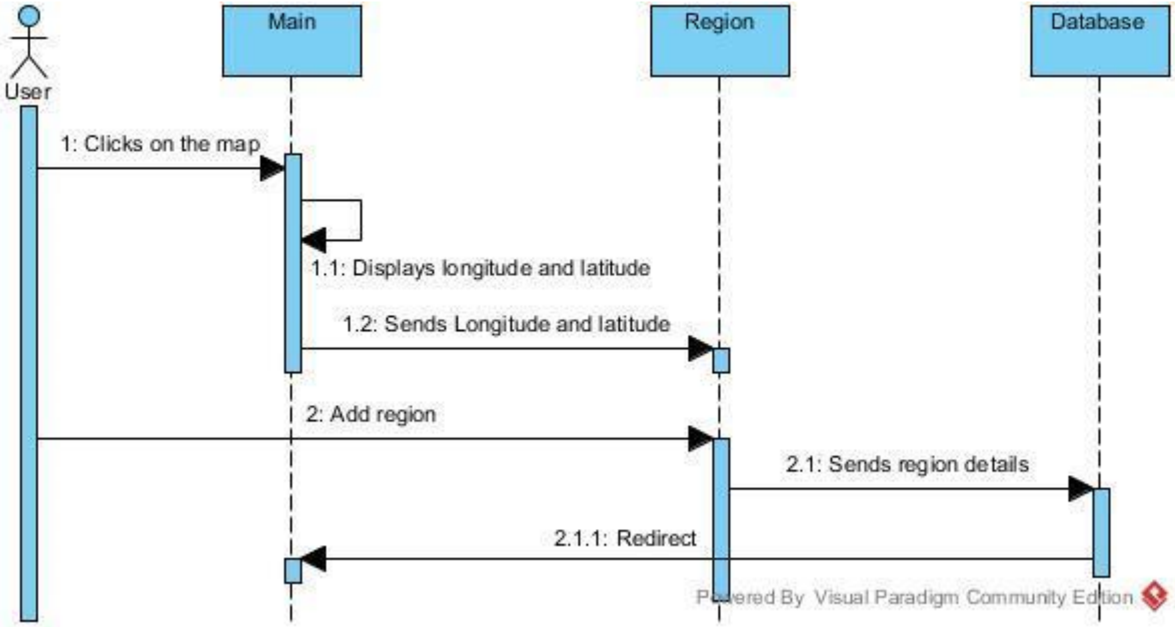

User
Main
Region
Database
1: Clicks on the map
1.1: Displays longitude and latitude
1.2: Sends Longitude and latitude
2: Add region
2.1: Sends region details
2.1.1: Redirect
Powered By Visual Paradigm Community Edition

Fig 4: Add Region

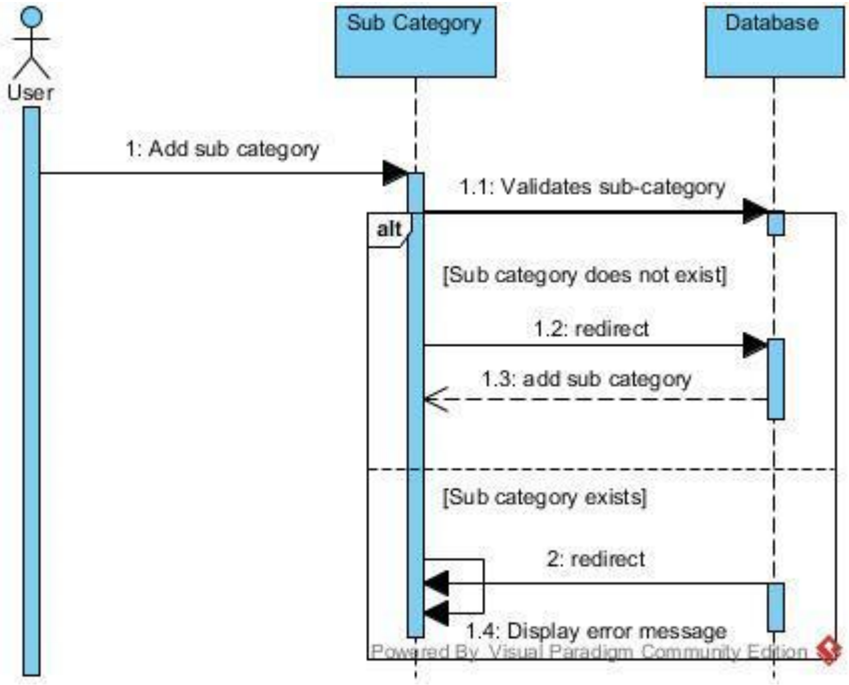

User
Sub Category
Database
1: Add sub category
1.1: Validates sub-category
alt
[Sub category does not exist]
1.2: redirect
1.3: add sub category
[Sub category exists]
2: redirect
1.4: Display error message
Powered By Visual Paradigm Community Edition

Fig 5: Add Sub-Category

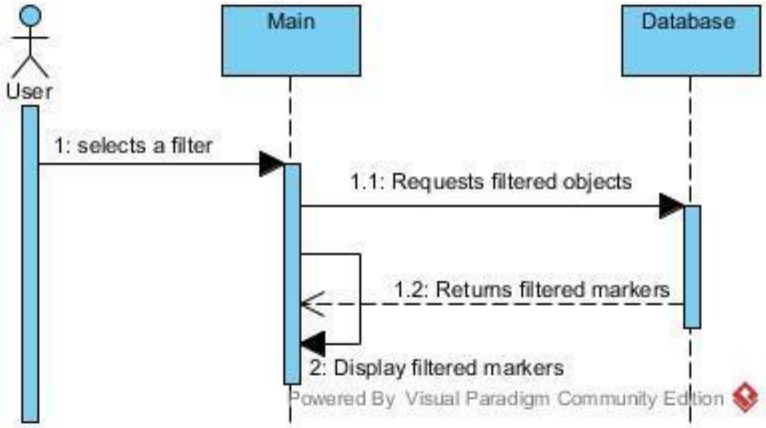

User
Main
Database
1: selects a filter
1.1: Requests filtered objects
1.2: Returns filtered markers
2: Display filtered markers
Powered By Visual Paradigm Community Edition

Fig 6: Filtered Search

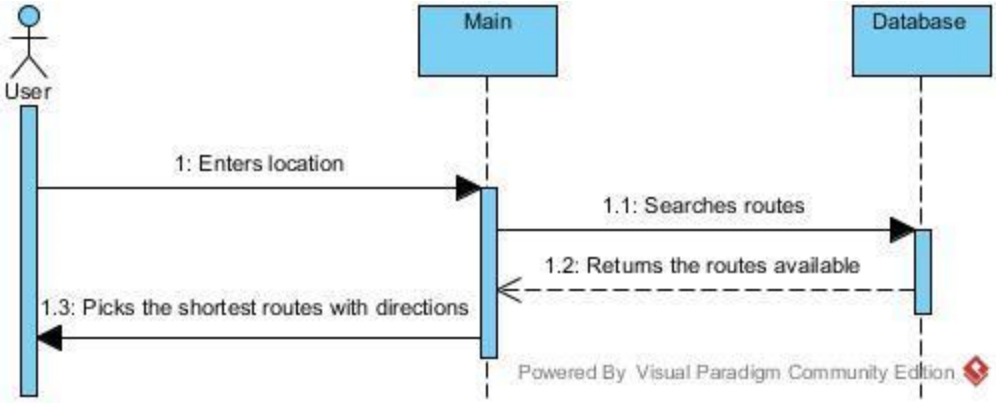

User
Main
Database
1: Enters location
1.1: Searches routes
1.2: Returns the routes available
1.3: Picks the shortest routes with directions
Powered By Visual Paradigm Community Edition

Fig 7: Searched Routes

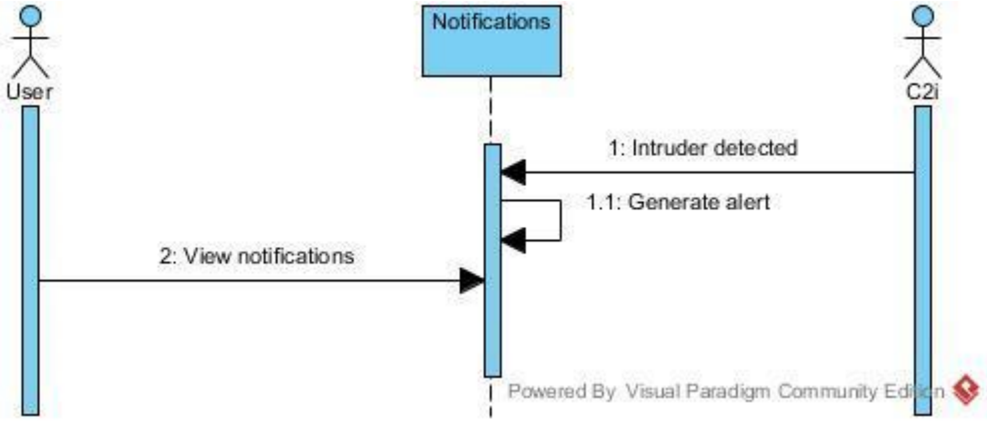


Fig 8: View notifications

Fig 4.4.3-1: Sequence Diagram

4.4.4 Logical View (State Transition Diagram)

The State Transition diagram is shown in the figure below:

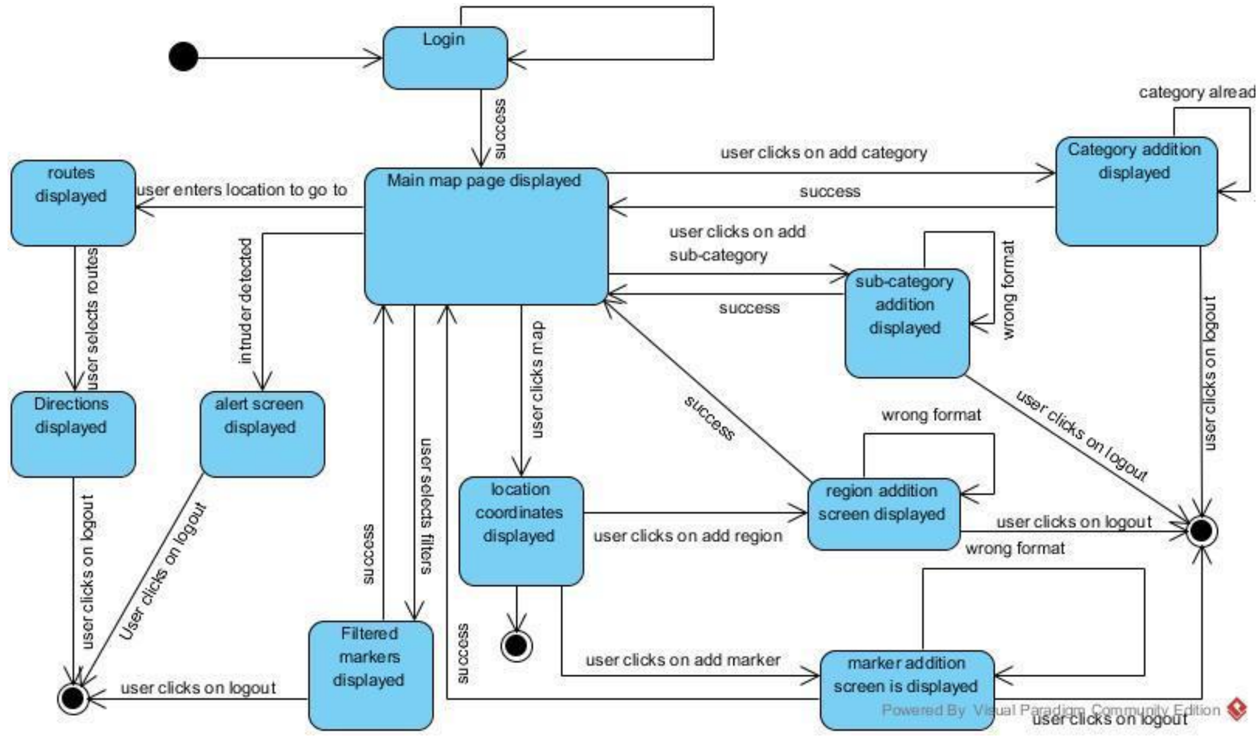


Fig 4.4.4-1: State Transition Diagram

4.4.5 Dynamic View (Activity Diagram)

In activity diagram, the dynamic view of the system is shown. All the activities are shown concurrently with their respective start and end states.

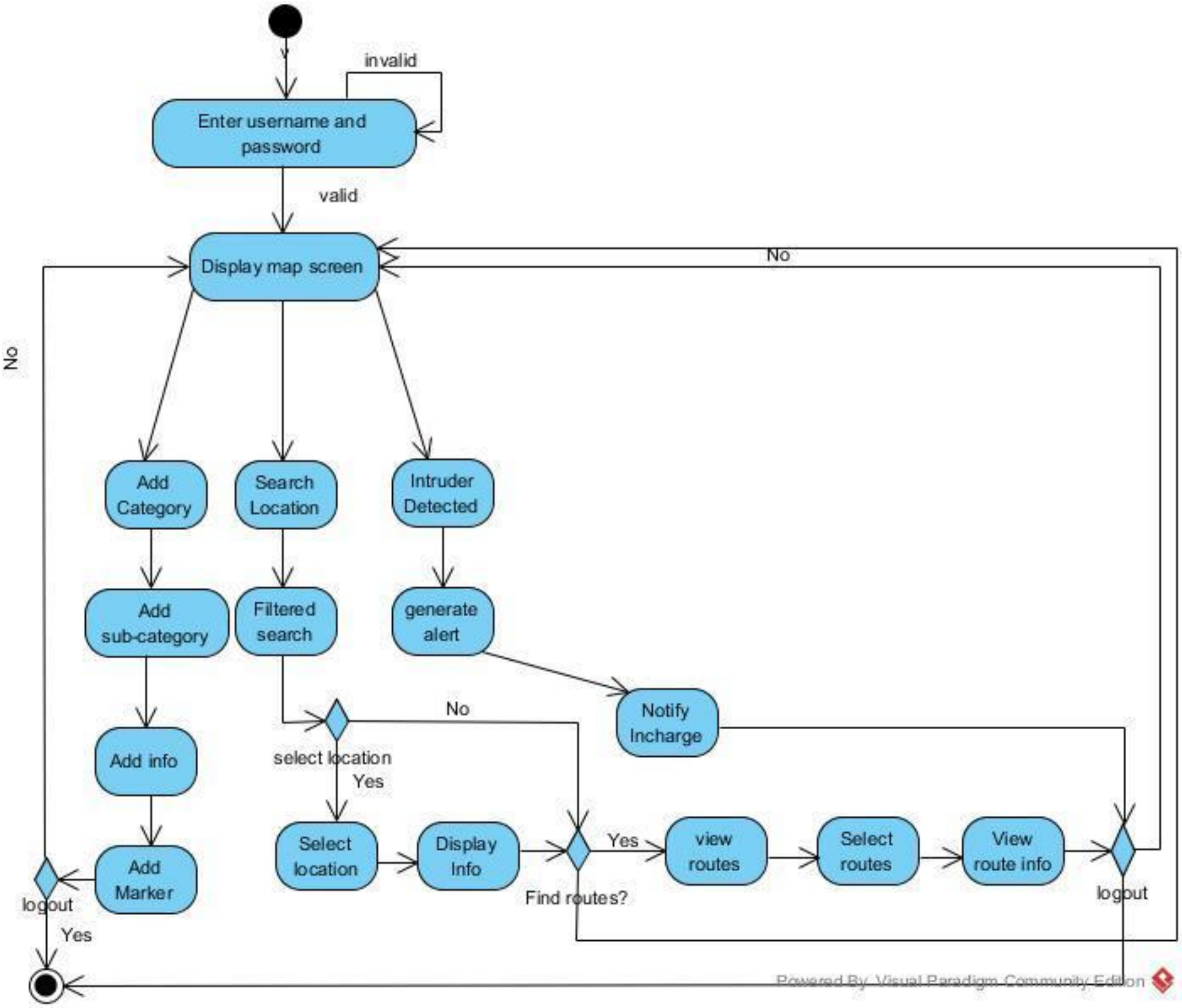
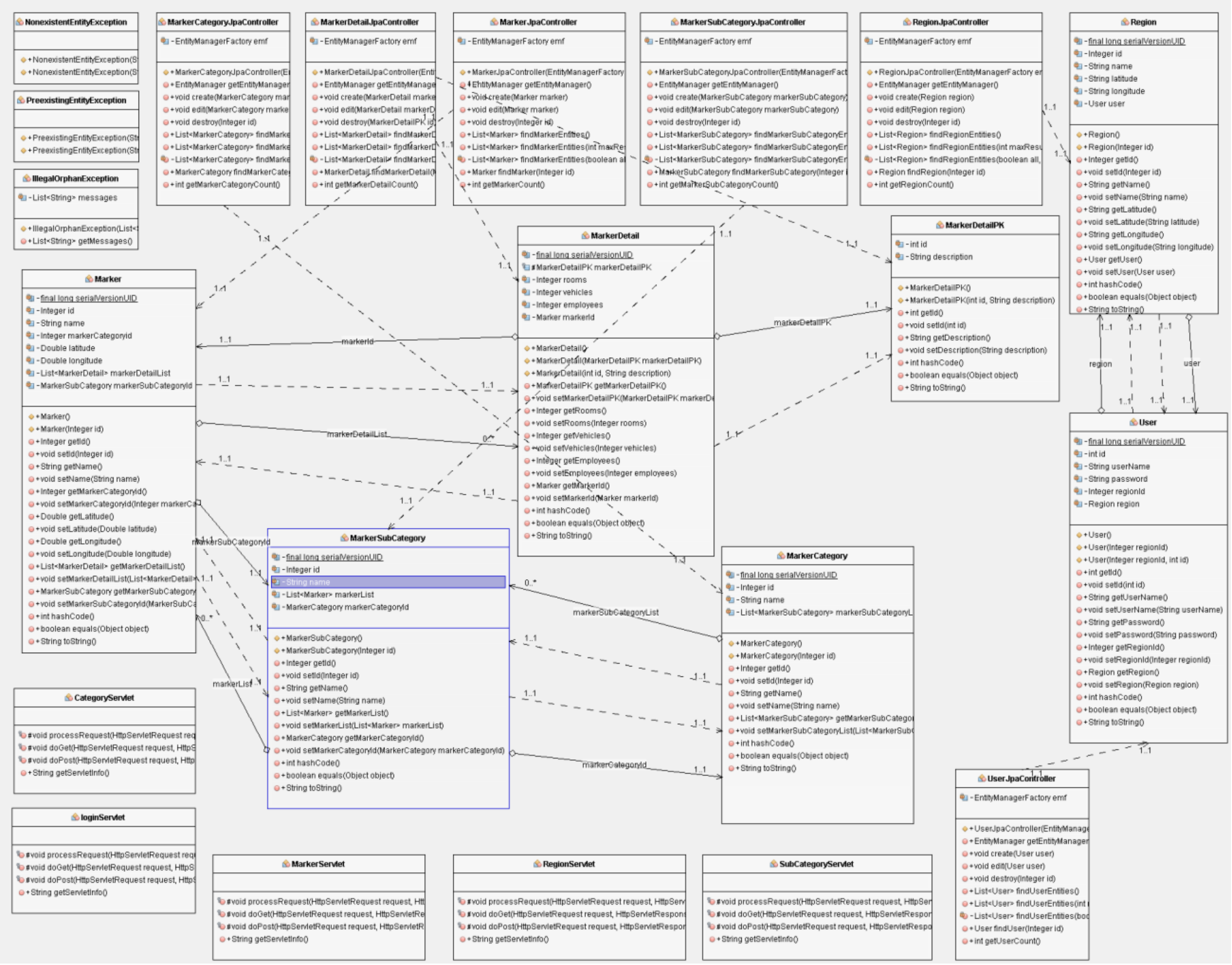


Fig 4.4.5-1: Activity Diagram

4.4.6 Implementation View (Class Diagram)


NonexistentEntityException
+NonexistentEntityException(S
+NonexistentEntityException(S
PreexistingEntityException
+PreexistingEntityException(St
+PreexistingEntityException(St
IllegalOrphanException
-List<String> messages
+IllegalOrphanException(List<
+List<String> getMessages()
MarkerCategoryJpaController
-EntityManagerFactory emf
+MarkerCategoryJpaController(E
+EntityManager getEntityManager
+void create(MarkerCategory mar
+void edit(MarkerCategory marke
+void destroy(Integer id)
+List<MarkerCategory> findMarke
+List<MarkerCategory> findMarke
-List<MarkerCategory> findMarke
+MarkerCategory findMarkerCate
+int getMarkerCategoryCount()
MarkerDetailJpaController
-EntityManagerFactory emf
+MarkerDetailJpaController(Ent
+EntityManager getEntityManage
+void create(MarkerDetail marke
+void edit(MarkerDetail markerD
+void destroy(MarkerDetailPK id
+List<MarkerDetail> findMarkerD
+List<MarkerDetail> findMarkerD
-List<MarkerDetail> findMarkerD
+MarkerDetail findMarkerDetail(
+int getMarkerDetailCount()
MarkerJpaController
-EntityManagerFactory emf
+MarkerJpaController(EntityManagerFactory
+EntityManager getEntityManager()
+void create(Marker marker)
+void edit(Marker marker)
+void destroy(Integer id)
+List<Marker> findMarkerEntities()
+List<Marker> findMarkerEntities(int maxRe
-List<Marker> findMarkerEntities(boolean al
+Marker findMarker(Integer id)
+int getMarkerCount()
MarkerSubCategoryJpaController
-EntityManagerFactory emf
+MarkerSubCategoryJpaController(EntityManagerFact
+EntityManager getEntityManager()
+void create(MarkerSubCategory markerSubCategory)
+void edit(MarkerSubCategory markerSubCategory)
+void destroy(Integer id)
+List<MarkerSubCategory> findMarkerSubCategoryEn
+List<MarkerSubCategory> findMarkerSubCategoryEn
-List<MarkerSubCategory> findMarkerSubCategoryEn
+MarkerSubCategory findMarkerSubCategory(Integer i
+int getMarkerSubCategoryCount()
RegionJpaController
-EntityManagerFactory emf
+RegionJpaController(EntityManagerFactory er
+EntityManager getEntityManager()
+void create(Region region)
+void edit(Region region)
+void destroy(Integer id)
+List<Region> findRegionEntities()
+List<Region> findRegionEntities(int maxResu
-List<Region> findRegionEntities(boolean all,
+Region findRegion(Integer id)
+int getRegionCount()
Region
-final long serialVersionUID
-Integer id
-String name
-String latitude
-String longitude
-User user
+Region()
+Region(Integer id)
+Integer getId()
+void setId(Integer id)
+String getName()
+void setName(String name)
+String getLatitude()
+void setLatitude(String latitude)
+String getLongitude()
+void setLongitude(String longitude)
+User getUser()
+void setUser(User user)
+int hashCode()
+boolean equals(Object object)
+String toString()
MarkerDetail
-final long serialVersionUID
#MarkerDetailPK markerDetailPK
-Integer rooms
-Integer vehicles
-Integer employees
-Marker markerId
+MarkerDetail()
+MarkerDetail(MarkerDetailPK markerDetailPK)
+MarkerDetail(int id, String description)
+MarkerDetailPK getMarkerDetailPK()
+void setMarkerDetailPK(MarkerDetailPK markerDe
+Integer getRooms()
+void setRooms(Integer rooms)
+Integer getVehicles()
+void setVehicles(Integer vehicles)
+Integer getEmployees()
+void setEmployees(Integer employees)
+Marker getMarkerId()
+void setMarkerId(Marker markerId)
+int hashCode()
+boolean equals(Object object)
+String toString()
MarkerDetailPK
-int id
-String description
+MarkerDetailPK()
+MarkerDetailPK(int id, String description)
+int getId()
+void setId(int id)
+String getDescription()
+void setDescription(String description)
+int hashCode()
+boolean equals(Object object)
+String toString()
Marker
-final long serialVersionUID
-Integer id
-String name
-Integer markerCategoryId
-Double latitude
-Double longitude
-List<MarkerDetail> markerDetailList
-MarkerSubCategory markerSubCategoryId
+Marker()
+Marker(Integer id)
+Integer getId()
+void setId(Integer id)
+String getName()
+void setName(String name)
+Integer getMarkerCategoryId()
+void setMarkerCategoryId(Integer markerCa
+Double getLatitude()
+void setLatitude(Double latitude)
+Double getLongitude()
+void setLongitude(Double longitude)
+List<MarkerDetail> getMarkerDetailList()
+void setMarkerDetailList(List<MarkerDetail
+MarkerSubCategory getMarkerSubCategory
+void setMarkerSubCategoryId(MarkerSubC
+int hashCode()
+boolean equals(Object object)
+String toString()
User
-final long serialVersionUID
-int id
-String userName
-String password
-Integer regionId
-Region region
+User()
+User(Integer regionId)
+User(Integer regionId, int id)
+int getId()
+void setId(int id)
+String getUserName()
+void setUserName(String userName)
+String getPassword()
+void setPassword(String password)
+Integer getRegionId()
+void setRegionId(Integer regionId)
+Region getRegion()
+void setRegion(Region region)
+int hashCode()
+boolean equals(Object object)
+String toString()
MarkerSubCategory
-final long serialVersionUID
-Integer id
-String name
-List<Marker> markerList
-MarkerCategory markerCategoryId
+MarkerSubCategory()
+MarkerSubCategory(Integer id)
+Integer getId()
+void setId(Integer id)
+String getName()
+void setName(String name)
+List<Marker> getMarkerList()
+void setMarkerList(List<Marker> markerList)
+MarkerCategory getMarkerCategoryId()
+void setMarkerCategoryId(MarkerCategory markerCategoryId)
+int hashCode()
+boolean equals(Object object)
+String toString()
MarkerCategory
-final long serialVersionUID
-Integer id
-String name
-List<MarkerSubCategory> markerSubCategoryL
+MarkerCategory()
+MarkerCategory(Integer id)
+Integer getId()
+void setId(Integer id)
+String getName()
+void setName(String name)
+List<MarkerSubCategory> getMarkerSubCatego
+void setMarkerSubCategoryList(List<MarkerSub
+int hashCode()
+boolean equals(Object object)
+String toString()
CategoryServlet
#void processRequest(HttpServletRequest req
#void doGet(HttpServletRequest request, HttpS
#void doPost(HttpServletRequest request, Http
+String getServletInfo()
loginServlet
#void processRequest(HttpServletRequest req
#void doGet(HttpServletRequest request, HttpS
#void doPost(HttpServletRequest request, Http
+String getServletInfo()
MarkerServlet
#void processRequest(HttpServletRequest request, Htt
#void doGet(HttpServletRequest request, HttpServletRe
#void doPost(HttpServletRequest request, HttpServletR
+String getServletInfo()
RegionServlet
#void processRequest(HttpServletRequest request, HttpServ
#void doGet(HttpServletRequest request, HttpServletRespon
#void doPost(HttpServletRequest request, HttpServletRespor
+String getServletInfo()
SubCategoryServlet
#void processRequest(HttpServletRequest request, HttpSer
#void doGet(HttpServletRequest request, HttpServletRespor
#void doPost(HttpServletRequest request, HttpServletRespo
+String getServletInfo()
UserJpaController
-EntityManagerFactory emf
+UserJpaController(EntityManage
+EntityManager getEntityManager
+void create(User user)
+void edit(User user)
+void destroy(Integer id)
+List<User> findUserEntities()
+List<User> findUserEntities(int
-List<User> findUserEntities(boo
+User findUser(Integer id)
+int getUserCount()
markerId
markerDetailList
markerDetailPK
markerSubCategoryId
markerSubCategoryList
markerList
markerCategoryId
region
user
1..1
0..*

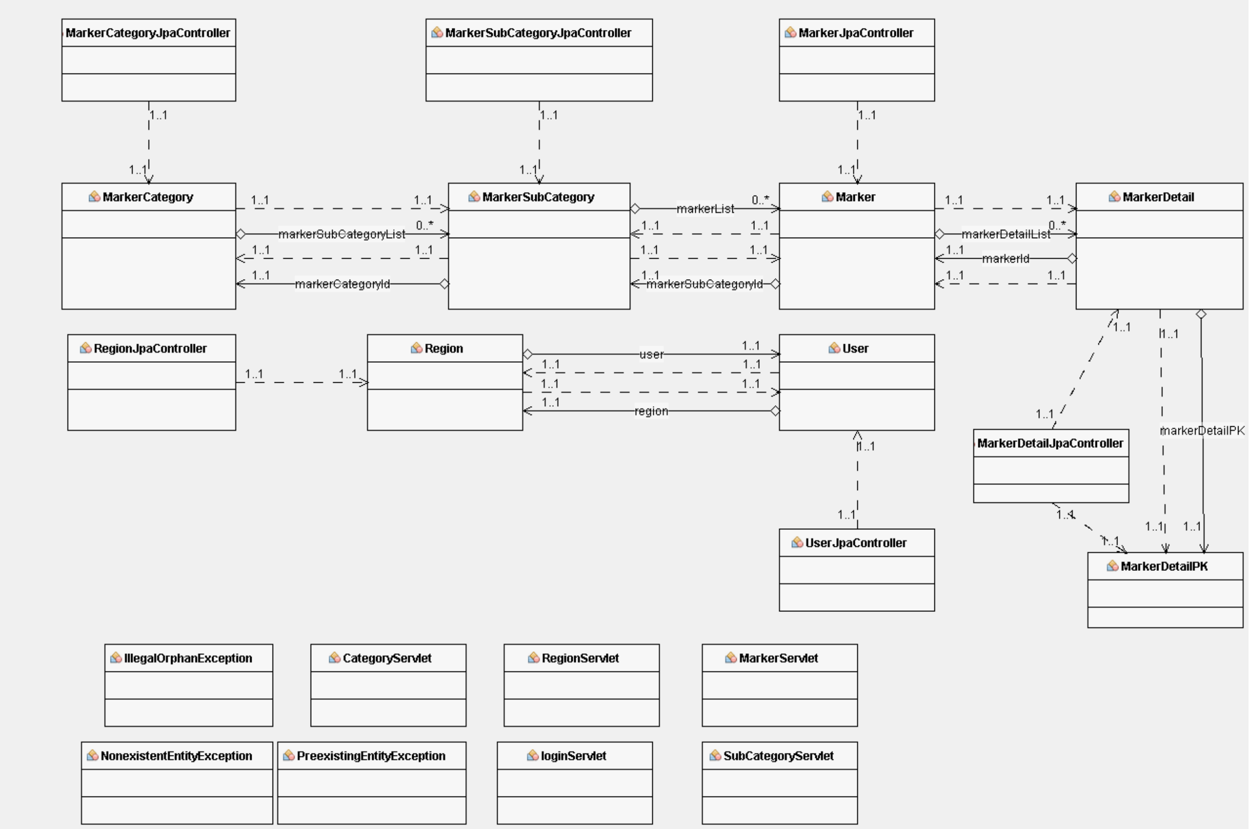


Fig 4.4.6-1: Class Diagram

Classes	Description
JSP Classes	JSP show the Graphical User Interface in the form of web pages.
DAO Classes	Data Access Object classes have getter and setter methods for the database.
JPAController Classes	JPA Controller classes are used for the connection of data access object classes with the view classes of the application.
Servlet Classes	Servlet classes are uses to process data between jsp pages.

4.4.7 Entity Relationship Diagram (ER Diagram)

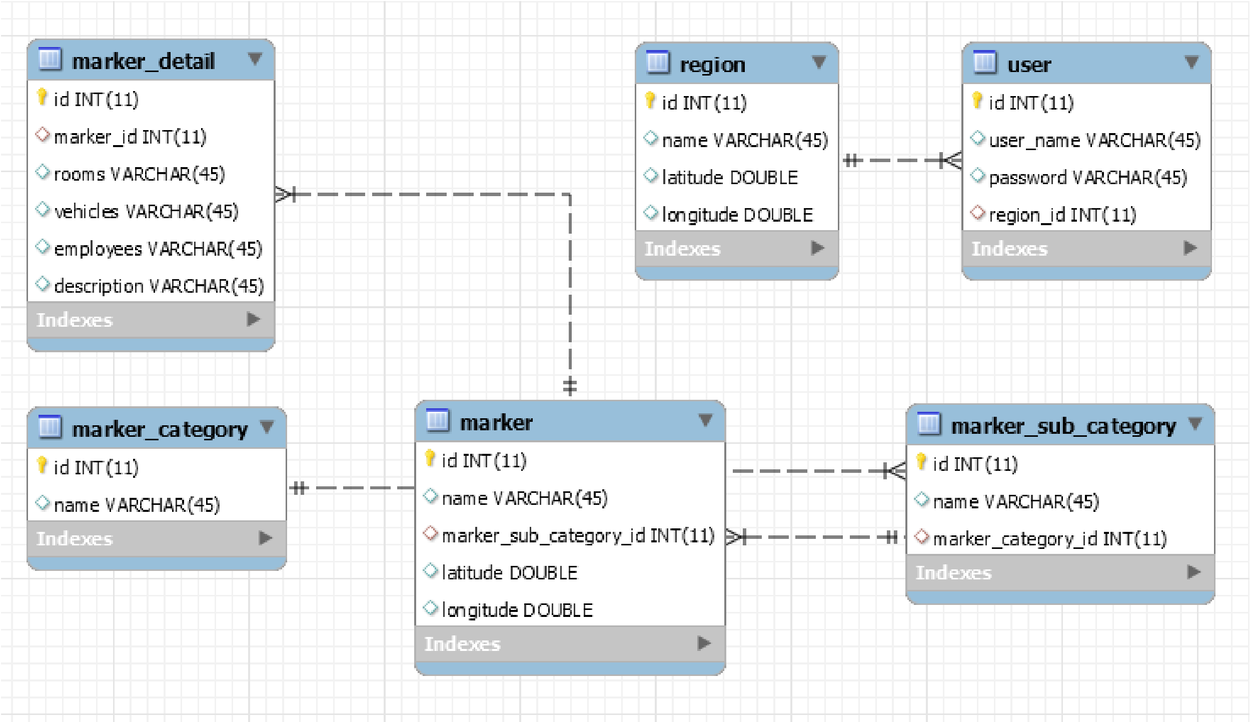


Figure 4.4.7-1: Entity Relationship Diagram

4.4.8 Structure Chart

This chart shows the breakdown of the application to its lowest manageable levels. It shows the modules and their corresponding functions which this application will implement. This chart basically shows the structure breakdown of the application starting from main modules to specific functions.

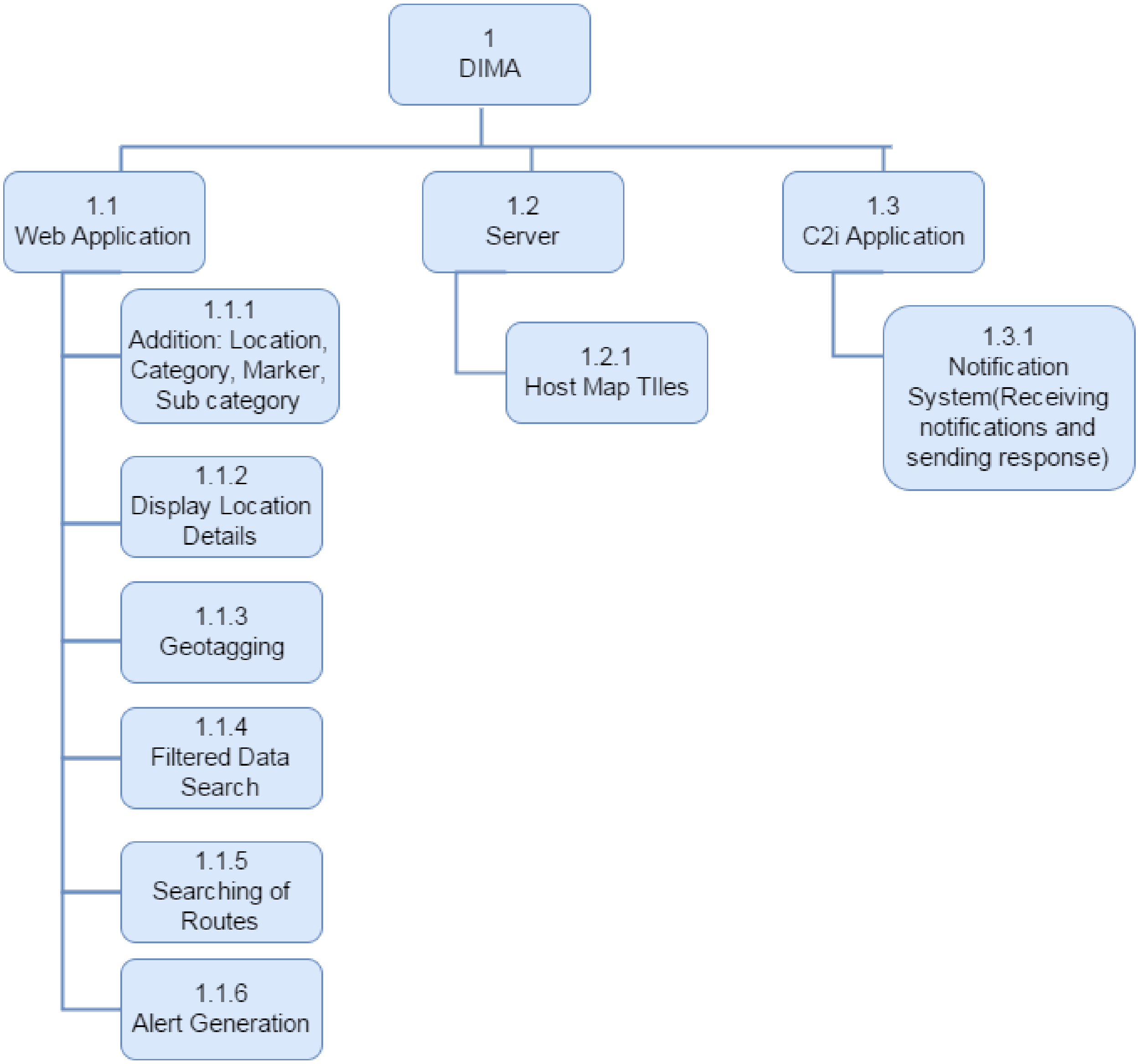
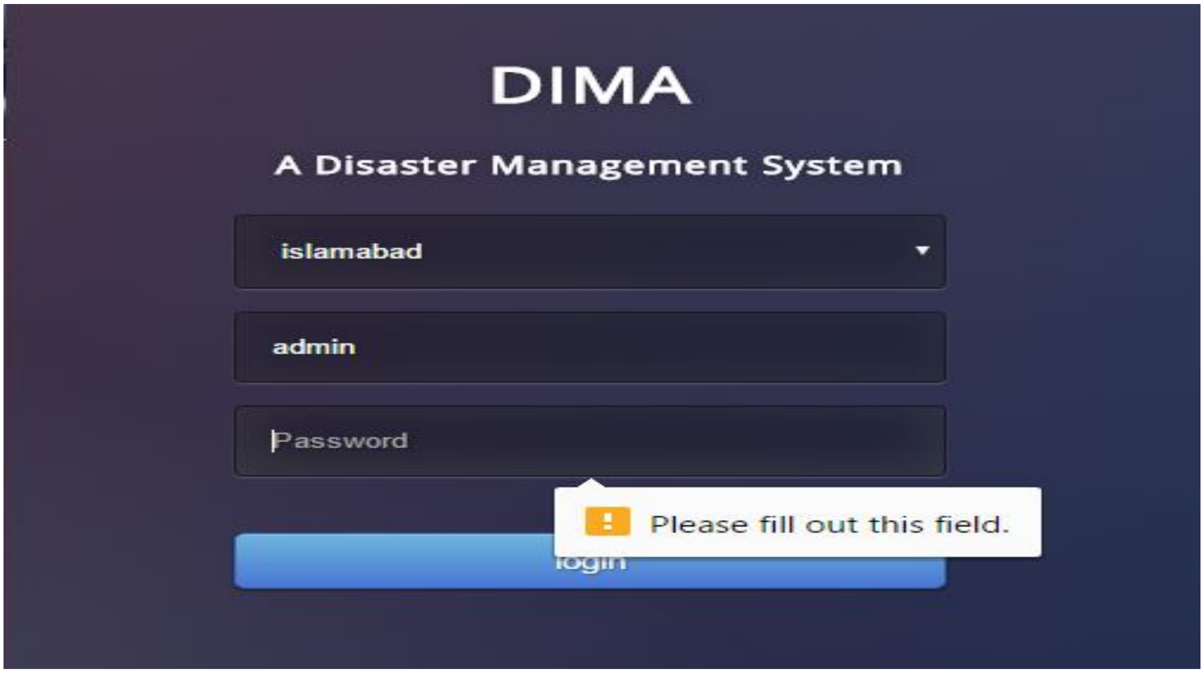


Fig 4.4.8-1: Structure Chart

4.5 User Interface

The user Interface of the DIMA is as follows:

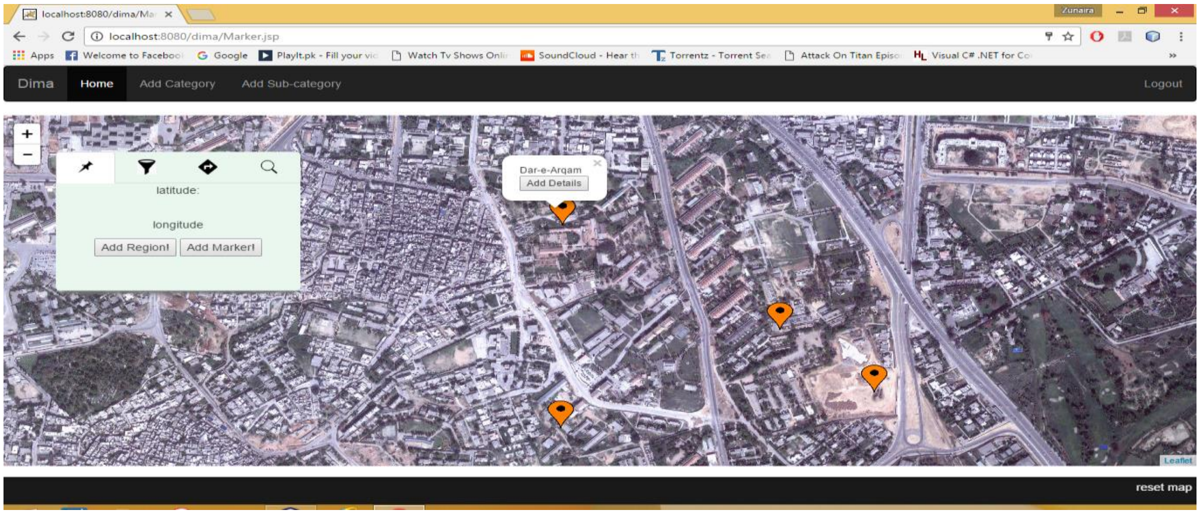

DIMA

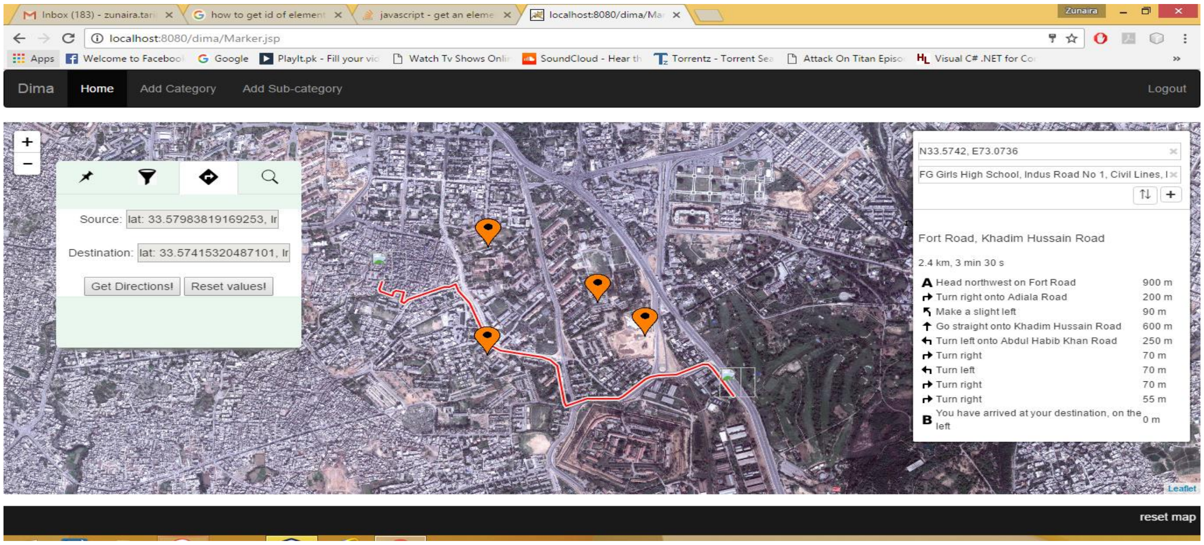
A Disaster Management System

islamabad

admin

Please fill out this field.


localhost:8080/dima/Marker.jsp



Source: lat: 33.57983819169253, lr

Destination: lat: 33.57415320487101, lr

N33.5742, E73.0736

FG Girls High School, Indus Road No 1, Civil Lines, I

Fort Road, Khadim Hussain Road

2.4 km, 3 min 30 s

A	Head northwest on Fort Road	900 m
↱	Turn right onto Adiala Road	200 m
↖	Make a slight left	90 m
↑	Go straight onto Khadim Hussain Road	600 m
↰	Turn left onto Abdul Habib Khan Road	250 m
↱	Turn right	70 m
↰	Turn left	70 m
↱	Turn right	70 m
↱	Turn right	55 m
B	You have arrived at your destination, on the left	0 m

Fig 4.5-1: User Interface

The Screen will have a login page which will ask for the valid username and password and enter into the main page containing the map tiles which will allow to add category, sub category, marker/region and show alerts with support of C2i application. It will also show markers of the screen of the tagged location.

4.6 Detailed Description of Components:

4.6.1 Application UI

Identification	*Name:* Application UI *Location:* Presentation layer of the system architecture
Type	UI component

Purpose	The User is able to see the converted text which is being read on the screen while the video is being made by the camera mounted on the ring. This component fulfills following functional requirements (as specified in SRS Document) related to user interface in the application: **REQ-1:** Application shall be able to show the map tiles of the selected region. **REQ-2:** The user shall be able to enter a category or a subcategory into the database. **REQ-3:** The user shall be able to enter a marker on the selected region. **REQ-4:** The application shall be able to provide the filtered search for any category or subcategory. **REQ-5:** The application shall be able to give the shortest route of the searched location with directions. **REQ-6:** The application shall also generate alerts on security breach with the support of the C2i application.
Function	This component displays map tiles of any region selected, allows the user to enter category and subcategory and displays markers of all tagged location. This component also interacts with the C2i application to generate alerts.
Subordinates	This component involves getting the maker details from the database and displaying the marker on the map. The component also interacts with the C2i module to get the notification information.
Dependencies	It interacts with *Server Application* and *C2i application,* whenever a user interacts with the application. It gets database records from server application and alert generation notification from support with C2i application.

Interfaces	The User interface is part of this section. It will display the map tiles, with markers and show alerts for security breach.
Resources	**Software:** NetBeans 8.1 and MySQL workbench using html, jsp, css, and java code using Hibernate and Spring Framework.
Processing	Take Input from the user. Display information regarding a location on screen. Also shows alert in case of security breach.
Data	Text entered in the form of latitude, longitudes, region, category and subcategory.

4.6.2 Server

Identification	*Name:* Server *Location:* Application logic layer of the system architecture
Type	Component
Purpose	Following functional requirements mentioned in SRS are fulfilled by this component: **REQ-1:** The application shall host the map tiles. **REQ-2:** The application shall generate results from the database.
Function	This component hosts map tiles and uses functions to return database records to UI application.
Subordinates	N/A
Dependencies	This component is dependent on database to return records to UI application.
Interfaces	The records will be sent to the UI Application.
Resources	**Software:** Apache Tomcat 7.0, web service using java servlets and jsp code
Processing	Server application gets records from database and return them to UI Application.
Data	Location, category and subcategory records from database.

4.6.3 C2i Application

Identification	*Name:* Compare Frames *Location:* Presentation layer of the system architecture
Type	Component
Purpose	Following functional requirements mentioned in SRS are fulfilled by this sub-component: **REQ-1:** The system shall provide input to Application UI to generate alerts in case of security breach.
Function	The function of this sub component is to provide frames of video in form of images to be used to convert into text for further processing.
Subordinates	It has two subordinates, one is to split video into frames and the other is to compare frames.
Dependencies	This component is independent.
Interfaces	The UI Application generates alerts in case of security breach.
Resources	Hardware: ZigBee module, Passive Infrared Sensors, Acoustic Sensors, X band Sensors etc.
Processing	This component sends alert information to Application UI.
Data	Sends alert information to Application UI.

4.7 Reuse and Relationships to Other Components

DIMA is a disaster management system which is being developed for Pakistan Army to manage disasters. We have integrated it with C2i system to handle intrusion detection as well which can helps to eliminate the occurrence of some of the disastrous occasions.

We can further enhance the features by adding new functionality and enhancing it to a bigger level. We can take the C2i system and enhance it to C4i as well which can transform our Disaster Management System to a next level involving the analytics and computing.

4.8 Design Decision and Tradeoffs

The Design of the device is very handy, it consists of a model which consists of data access object classes and database. A controller which consists of JPA Controller Classes and a View which consists of a Graphical User Interface consisting of jsp web pages and a C2i application view.

We have followed the Model View Controller architectural design for our application in this as shown below:

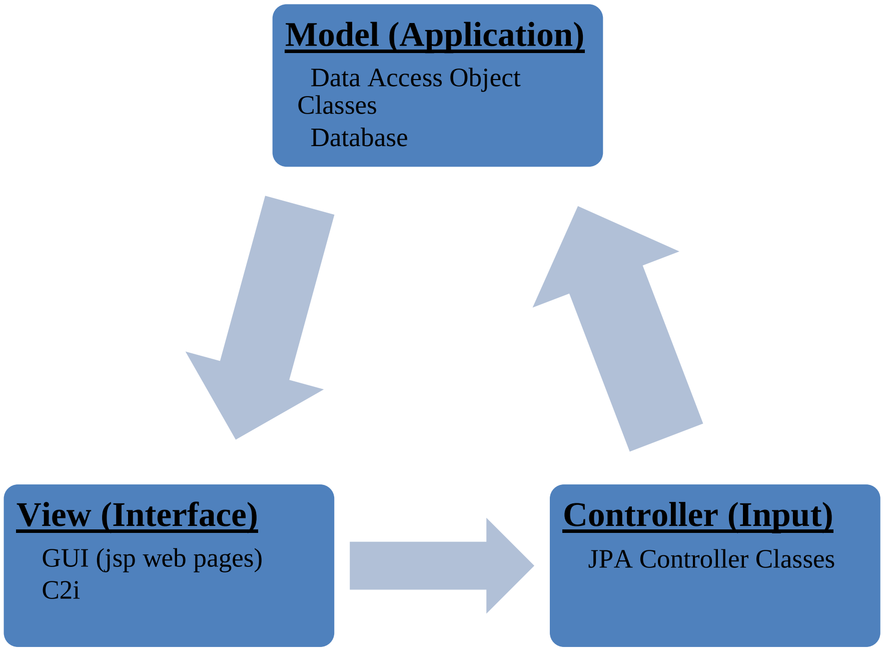


Fig 4.8-1: MVC Diagram

View is the user interface of the application, with which the user will be interacting.

Controller is the handler for whatever action is done on the view, which will communicate with view and pass on the information gathered from view to model (application / business logic). It will also communicate with model and get the information to be displayed on view from model and pass it on to the view. Controller is the mediator between model and view.

Model is the application or business logic; it includes all the algorithms that will be implemented.

CHAPTER: 5

SYSTEM IMPLEMENTATION

5 SYSTEM IMPLEMENTATION

DIMA is an offline application which works using map tiles available on the server. However it can also work in an online mode if need be. User can navigate location of emergency buildings and find the shortest route for any location. Only authorized users will be able to access the application and integration of C2i application will help generate alerts on DIMA in case of security breach.

5.1 Pseudo Code for Components

5.1.1 Application UI

```
Begin

login();

GetMapTiles();

ShowMapTiles();

GetMarkers();

ShowMarkers();

if( user clicks on category addition )

   SetCategory();
```

```
endif

if( user clicks on sub category addition)

  SetSubCategory();

endif

if(user clicks on map)

  if(user clicks add region)

  SetRegion()

  Endif

  if(user clicks add marker)
  addMarker()

  Endif

Endif

If (user selects filter)

  ShowfilteredMarkers()

Endif

If (user searches a route)

  ShowRoute();
```

```
Endif

End
```

5.1.2 Notification System:

C2i Application (UI)

```
Begin

If ( intruder detetected )

   generateAlert()

   SendNotification()

Endif

End
```

5.1.3 DimaApplication (UI)

```
Begin
If ( Notification Recieved )
   ShowAlert();
Endif
End
```

5.1.4 Server-Side Processing

```
Begin
If ( data updates recieved )
    updateRecords();
Endif
If ( data is needed )
    sendRecords();
End
```

CHAPTER: 6

ANALYSIS AND EVALUATION

6 Analysis and Evaluation

6.1 Introduction

This test plan document describes the appropriate strategies, process and methodologies used to plan, execute and manage testing of the Disaster Management System (DIMA) project. The test plan will ensure that DIMA meets the customer requirements at an accredited level.

Manual Testing will be followed which includes testing a software manually, i.e., without using any automated tool or any script. In this type, the tester takes over the role of an end-user and tests the software to identify any unexpected behavior or bug. Each Unit will be tested separately and then will be integrated with other units; therefore Unit Testing and Integration testing will be followed. For each unit Black box Testing is done and for combined units Acceptance Testing is done.

The test scope includes the Testing of all functional, application performance and use cases requirements listed in the *requirement document*

Software testing, depending on the testing method employed, can be implemented at any time in the development process. However, most of the test effort occurs after the requirements have been defined and the coding process has been completed.

This document includes the plan, scope, approach and procedure of test. The Test Plan document documents and tracks the necessary information required to effectively define the approach to be used in the testing of the product.

6.2 Approach

Acceptance test will be executed based on this acceptance test plan. And after all test cases are executed, a test report will be summarized to show the quality of Disaster Management System (DIMA). Following test approaches will be used in test execution:

- **Unit test**. Developers are responsible for unit test as white-box testing. The implementation of each module and individual component will be verified separately.
- **Integration test**. After the unit test is passed above the defined quality threshold, testers will execute the integration test cases. After all the modules are integrated, it’s crucial to test the product as a black-box. End-to-end scenarios will be tested to ensure the communication functionality.
- **Regression test**. After developers fix the bug in one feature, regression test will be executed by testers to ensure that the other functions are not affected.
- **Field test**. Firstly, untrained end users recreate one or more existing (but narrow) mass observation events in the DIMA Application. A number of observers will be invited to help with evaluation. After that, post event questionnaires will be used to collect quantitative usage data as well as qualitative data and further improvement will be taken into consideration.
- **Positive and negative testing design technique**. This approach will be combined with unit test and integration test. Test cases are designed in obvious scenarios, which ensure that all functional requirements are satisfied. What’s more, different test cases will also be covered to show how the system reacts with invalid operations.

6.3 Features to be tested

Following Features are tested:

- Login
- Addition of category
- Addition of region
- Addition of marker
- filtered search
- Addition of sub category
- Searching of routes

6.4 Item Pass/Fail Criteria

Details of the test cases are specified in section Test Deliverables. Following the principles outlined below, a test item would be judged as pass or fail.

Preconditions are met

Inputs are carried out as specified

The result works as what specified in output => Pass

The system doesn't work or not the same as output specification => Fail

6.5 Testing tasks

Develop Test Cases.

Execute tests on the basis of the test cases developed

Report defects during tests if any.

Complete the test report.

Manage the changes made after testing.

6.6 Test Deliverables

- Test cases

- Output from tools

6.7 Responsibilities

All developers of the project are responsible for the completion of all components testing and integration testing tasks.

6.8 Staffing and training needs

Basics knowledge of testing strategies and techniques is needed for the testing of the project.

All the developers will be testing each other's work and will be actively participating in the development and testing of the project simultaneously.

6.9 Risks and contingencies

6.9.2 Schedule Risk:

The project might get behind schedule so in order to complete the project in time we will need to increase the hours/day that the project is being worked on.

6.9.3 Budget Risk:

The budget will be compensated by using less costly alternatives to fit the budget requirements.

6.9.4 Operational Risks:

Operational risks will be eliminated by Scheduling daily meetings and regular deadlines to meet the goals of the project as well as provide proper communication within the group.

6.9.5 Technical risks:

Technical risks will be eliminated by keeping the once defined requirements constant.

6.9.6 Programmatic Risks:

In case of a programmatic risk the scope of the project will be limited in order to stay inside the constraints of the project.

6.10 Environmental Needs

6.10.2 Hardware

Laptops and PC's.

6.10.3 Software

6.11 Software

Windows 7, 8, 8.1, 10

Software: NetBeans 8.1, MySQL Workbench, Tomcat 7.0

6.12 Risks and contingencies

Efforts have been made to remove all and every chance of failure but there are certain unpredictable factors such as network issues, corrupt input data, or system failure that may lead to some issues. Error handling will be applied more deeply to cover all these issues but unforeseen circumstances may happen.

6.13 Test Cases:

Unit Testing and Integration Testing

Following are the Test Cases:

Table 6-1:

Test Case Name	Login
Test Case No	1
Description	The user can login only with valid username and password
Testing Technique Used	Black box testing
Preconditions	Application must be properly installed.
Input Values	Username and password
Steps	1. Open DIMA application. 2. Enter username and password. 3. Click login button.
Expected Output	Main map page should be displayed.
Actual Output	Main map page is displayed.

Table 6-2

Test Case Name	Addition of marker
Test Case No	2
Description	Testing the addition of marker details in the database.
Testing Technique Used	Black box testing
Preconditions	Main map page is displayed. User clicks on a location on the map and then clicks on ‘Add Marker’ button.
Input Values	User clicks on ‘Add Marker’.
Steps	1. Go to home page. 2. Click on a location on the map. 3. Enter marker details. 4. Click ‘Add to Database’ button.
Expected Output	User is redirected to home page.
Actual Output	User is redirected to home page.

Table 6-3

Test Case Name	Addition of region
Test Case No	3
Description	Testing the addition of region in the database.
Testing Technique Used	Black box testing
Preconditions	Main map page is displayed. User clicks on a location on the map and then clicks on ‘Add Region’ button.
Input Values	Region name

Steps	1. Go to home page. 2. Click on a location on the map. 3. Enter marker details. 4. Click ‘OK’ button.
Expected Output	User is redirected to home page.
Actual Output	User is redirected to home page.

Table 6-4

Test Case Name	Filtered search
Test Case No	4
Description	Testing the category wise search which filters the data according to the user requirement
Testing Technique Used	Black box testing
Preconditions	Main map page is displayed.
Input Values	Category and subcategory is selected.
Steps	1. Go to home page. 2. Select ‘filter’ tab. 3. Then select category and subcategory from dropdown menu. Click on ‘Apply Filters’ button.
Expected Output	Category wise information is displayed.
Actual Output	Category wise information is displayed.

Table 6-5

Test Case Name	Category addition
Test Case No	5
Description	Testing the addition of category in the database.
Testing Technique Used	Black box testing
Preconditions	The user selects 'Category tab'
Input Values	The user should select 'category tab' and enter category details.
Steps	1. Go to home page. 2. Select 'Category' tab. 3. Enter category details. 4. Click 'OK' button.
Expected Output	Category is added in the database and user is redirected to home page.
Actual Output	Category is added in the database and user is redirected to home page.

Table 6-6

Test Case Name	Subcategory addition
Test Case No	5
Description	Testing the addition of subcategory in the database.
Testing Technique Used	Black box testing
Preconditions	The user selects subcategory tab.
Input Values	The user should select 'Subcategory tab' and enter category and

	subcategory details.
Steps	1. Go to home page. 2. Select ‘Subcategory’ tab. 3. Enter category details. 4. Click ‘OK’ button.
Expected Output	Subcategory is added in the database and user is redirected to home page.
Actual Output	Subcategory is added in the database and user is redirected to home page.

Table 6-7

Test Case Name	Searching of routes
Test Case No	6
Description	Testing the route search to a certain location.
Testing Technique Used	Black box testing
Preconditions	The user clicks on ‘get directions’ button.
Input Values	The user selects source location and destination location. Then the user clicks on get directions button.
Steps	1. Go to home page. 2. Click on get direction icon. 3. Select source location and destination location. 4. Click on ‘Get Directions’ button.
Expected Output	Routes from source to destination is displayed.

Actual Output	Routes from source to destination is displayed.

CHAPTER: 7

FUTURE WORK

7 FUTURE WORK

A system of this magnitude always needs more and more work to evolve. There are a lot of possible changes and additions that can be done to the system to improve its performance and functionalities. The system has been made in a modular fashion which enables integrating new features very easy.

Some of the key features which current version doesn't have are:

- Report the tragedy as it happens with a single click
- Rapidly respond to the emergencies by allocating assets
- Track assets in real-time as they move in the field
- Minimize the response time for rescue operations
- Get geographical and temporal stats about crime hot-spots
- Use data mining techniques to intelligently and strategically place assets in the field

CHAPTER: 8

CONCLUSION

8 CONCLUSION

8.1 Overview

The main purpose of this project is the development of a system with Command Control and Intelligence web application that will help military officials locate emergency buildings in case of an emergency, and it will also help them navigate to these locations by calculating and showing the shortest possible routes to the concerned location.

8.2 Objectives Achieved

The Project is developed with the aim that it can be easily adopted by the beneficiaries since we will be targeting government bodies, schools, military forces and housing schemes such as DHA, Askaris, Bahria Town, etc. The reason that disaster relief agencies, government bodies and the housing scheme authorities and its residents will adopt this application is because **damage control** and **disaster relief** are among the main concerns these days and people are thoroughly looking forward to solutions to solve these problems. Application is designed in such a way keeping the **ease-of-use** of users in mind.

CHAPTER: 9

BIBLIOGRAPHY

9 BIBLIOGRAPHY

1. http://www.cssforum.com.pk/css-compulsory-subjects/essay/72358-essay-disaster-management-pakistan.html

2. http://www.ndma.gov.pk/

3. https://crimemapping.info/article/use-geographic-profiling-crime-analysis/

4. http://www.actionforenterprise.org/drr1.pdf

5. https://corporate.target.com/corporate-responsibility/safety-preparedness/disaster-preparedness-response

6. http://www.adpc.net/igo/contents/Training/training-schedule-event.asp?pid=759

7. http://stackoverflow.com/questions/16206322/how-to-get-js-variable-to-retain-value-after-page-refresh

8. https://www.npmjs.com/package/node-persist

9. http://www.webdeveloper.com/forum/showthread.php?223946-accessing-database-values-in-javascript

10. https://www.youtube.com/watch?v=N-GKvmWXBgQ

11. http://stackoverflow.com/questions/17662551/how-to-use-angular-directives-ng-click-and-ng-class-inside-leaflet-marker-popup?rq=1

12. http://pdma.gop.pk/

13. http://nidm.edu.pk/

APPENDIX A

USER MANUAL

Table of Contents

Table of Figures:

General Information

10 GENERAL INFORMATION

This section explains in general terms the system **DIMA** and the purpose for which it is intended.

10.1 System Overview:

DIMA is a disaster management system that will aid Pakistan army and military officials in navigating to emergency buildings (such as hospitals, fire stations, police stations etc.), in geotagging , viewing details of these and in finding the shortest possible route for a location in case of a disaster.

10.2 Organization of the manual:

The user’s manual consists of five sections: General Information, System Summary, Getting Started, Using the System.

1. **General Information** section explains in general terms the system and the purpose for which it is intended.
2. **System Summary** section provides a general overview of the system. The summary outlines the uses of the system's hardware and software requirements, system’s configuration, user access levels and system’s behavior in case of any contingencies.
3. **Getting Started** section explains how to setup the system and configure it for the first time. The section presents briefly system's settings.
4. **Using the System** section provides a detailed description of system functions.

System Summary

11 SYSTEM SUMMARY

System Summary section provides a general overview of the system. The summary outlines the uses of the system's hardware and software requirements, system’s configuration, user access levels and system's behavior in case of any contingencies.

11.1 System Configuration:

DIMA requires Java to be installed on the operating system. Minimum version required is JDK 1.6. System needs server connectivity as well. First time configuration of database is also needed for the system.

11.2 User Access Levels:

The System will be available for to all military officials, all authorized users will be able to perform all tasks (adding/viewing details, records, categories, sub categories, markers).

11.3 Contingencies:

In case of any errors or system crashes, the database will not be affected and all building records will remain safe. The system will restart from the login. If the user was in the middle of making any changes (tagging a location, adding a category subcategory etc.) then they will not be saved. User Settings will also remain same.

Getting Started

12 GETTING STARTED

Getting Started section explains how to configure the system and install it for the first time use. The section also presents briefly the system's menu.

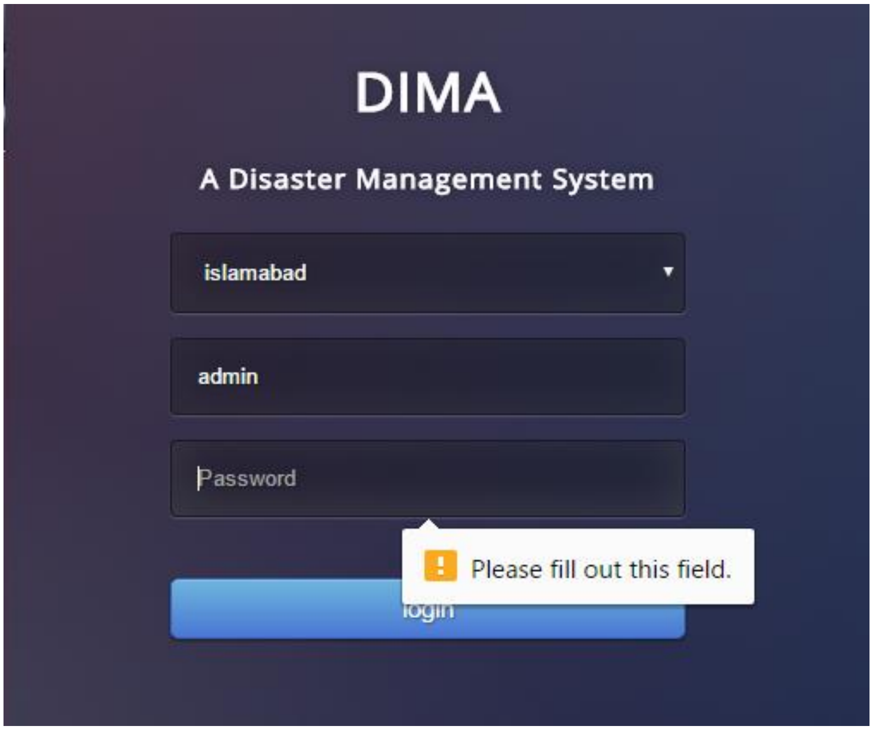
12.1 Installation:

The application can be installed by the exe file provided.

1. After installation, System has to be checked for network availability from Settings.
2. It should be connected to the web server
3. Database has to be started from settings menu,

12.2 Login:

The application starts with a login page, where user logs in based on their region, username and password.



DIMA

A Disaster Management System

islamabad ▾

admin

Password

⚠ Please fill out this field.

Figure 12-1 Login

12.3 Main Application Page:

After login the main page of the application opens up, you have multiple options

1. User can add category by clicking on add category on the menu.
2. User can add subcategory if clicked on the add subcategory button on the menu
3. If the user clicks on the magnifier icon to search for a building.
4. If the user clicks on the pin icon on the tabbed pane the user can tag a location.
5. On clicking the tabbed pane with the directions icon user can find a route.
6. On clicking the filter icon on the tabbed pane, user can filter markers.
7. User can reset page by clicking on the reset button on the bottom
8. User can logout using logout button on top right.

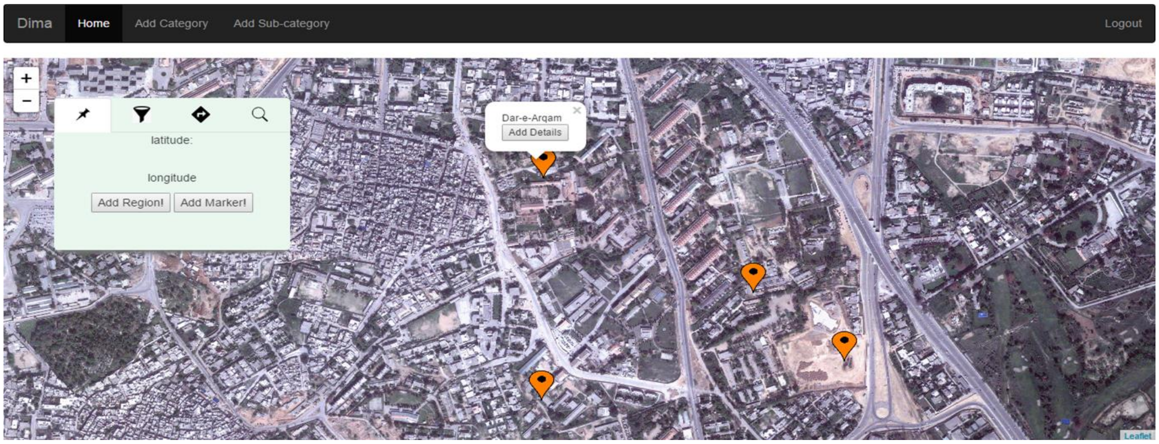


Figure 12-2 Main Screen

12.4 Exit Application:

The system will save all settings, Terminate database and network connections and exit the application.

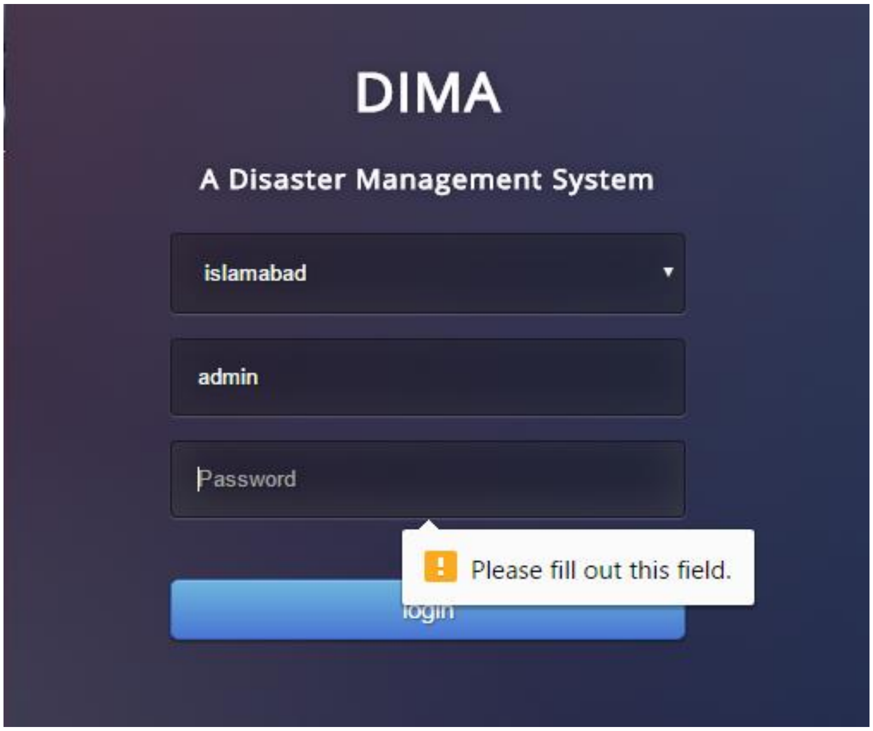
Using the System

13 USING THE SYSTEM:

This section provides a description of system functions and features.

13.1 Login:

The application starts with a login page, where user logs in based on their region, username and password.



DIMA

A Disaster Management System

islamabad

admin

Please fill out this field.

Figure 13-1 Login

13.2 Main Application Page:

13.3.1 After login the main page of the application opens up, you have multiple options:

13.3.2 If the user clicks on the add category part of the menu the application will take the user to the category addition page, where you can just enter a new category and add it to the database with the click of “Add to database” button.

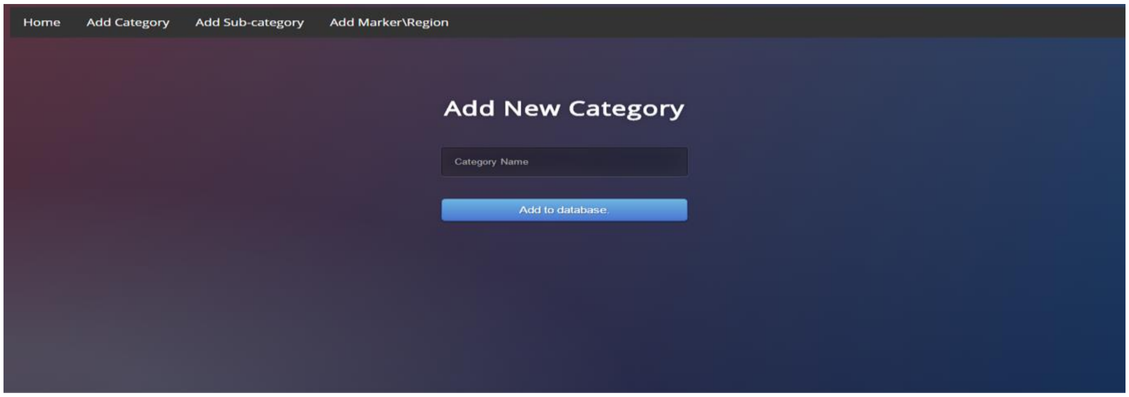
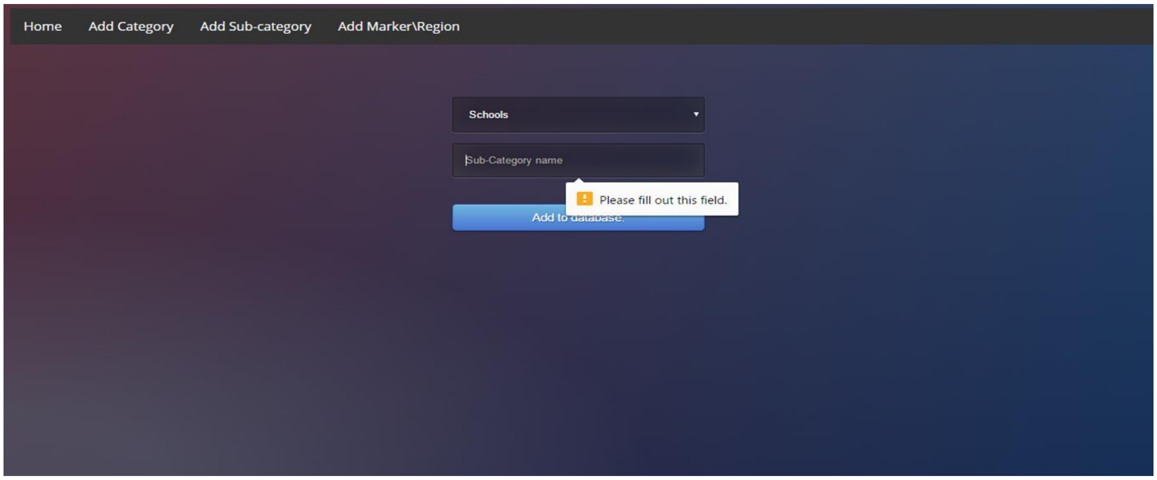


Figure 13-2 Addition of Category

13.3.3 If the user clicks on the add subcategory part of the menu the application will take the user to the subcategory addition page, where you can just enter a sub category from a category selected from the dropdown menu and add it to the database with the click of “Add to database” button.



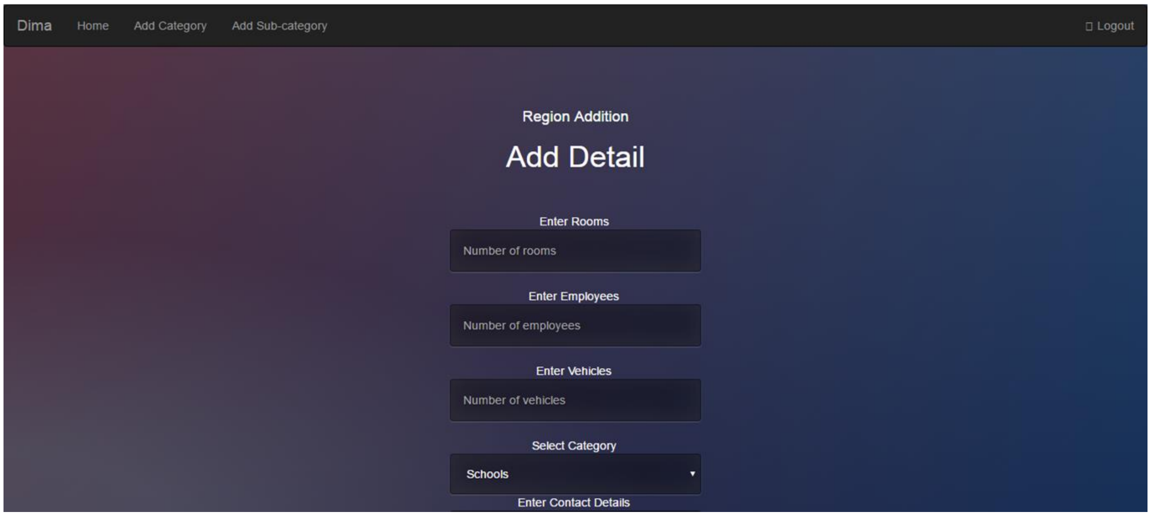
Schools

Please fill out this field.

Figure 13-3 Addition of Sub-Category

13.3.4 If the user clicks on the magnifier icon on the tabbed pane, the user can select a category and find the nearest building belonging to that category.

13.3.5 If the user clicks on the pin icon on the tabbed pane the user can tag a location, this can be done by clicking on the location you want to tag, two buttons and the coordinates of the location will appear on the tabbed pane, if the user clicks on the “add region” button a new region can be add with this location, if the user clicks on the “add marker” button then the user is directed to a page where they can add marker related information like it's category, subcategory, name and a marker will be added to the database.



Region Addition

Add Detail

Enter Rooms

Number of rooms

Enter Employees

Number of employees

Enter Vehicles

Number of vehicles

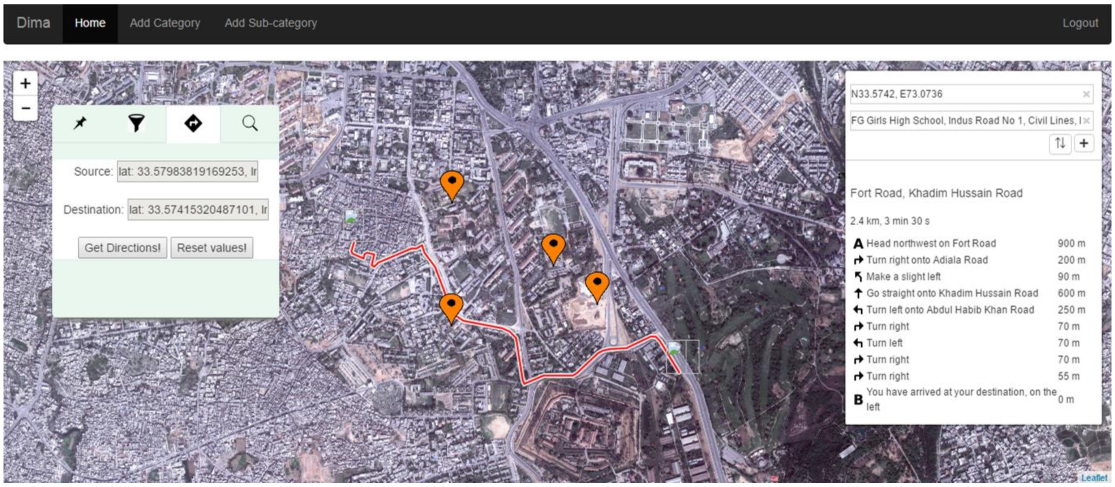
Select Category

Schools

Enter Contact Details

Figure 13-4 Addition of Details

13.3.6 On clicking the tabbed pane with the directions icon user can click on any two points on the map and the system shows a route and gives directions as well, clicking on the reset route button resets source destination and removes the directions controller.



Source: lat: 33.57983819169253, lr

Destination: lat: 33.57415320487101, lr

N33.5742, E73.0736

FG Girls High School, Indus Road No 1, Civil Lines, I

Fort Road, Khadim Hussain Road

2.4 km, 3 min 30 s

A	Head northwest on Fort Road	900 m
↱	Turn right onto Adiala Road	200 m
↖	Make a slight left	90 m
↑	Go straight onto Khadim Hussain Road	600 m
↰	Turn left onto Abdul Habib Khan Road	250 m
↱	Turn right	70 m
↰	Turn left	70 m
↱	Turn right	70 m
↱	Turn right	55 m
B	You have arrived at your destination, on the left	0 m

Figure 13-5 Routes

13.3.7 On clicking the filter icon on the tabbed pane, user can filter markers by selecting a category or both a category and subcategory and then clicking the filter button.

13.3.8 On the bottom of the page a reset button is available that resets the map to its initial state.

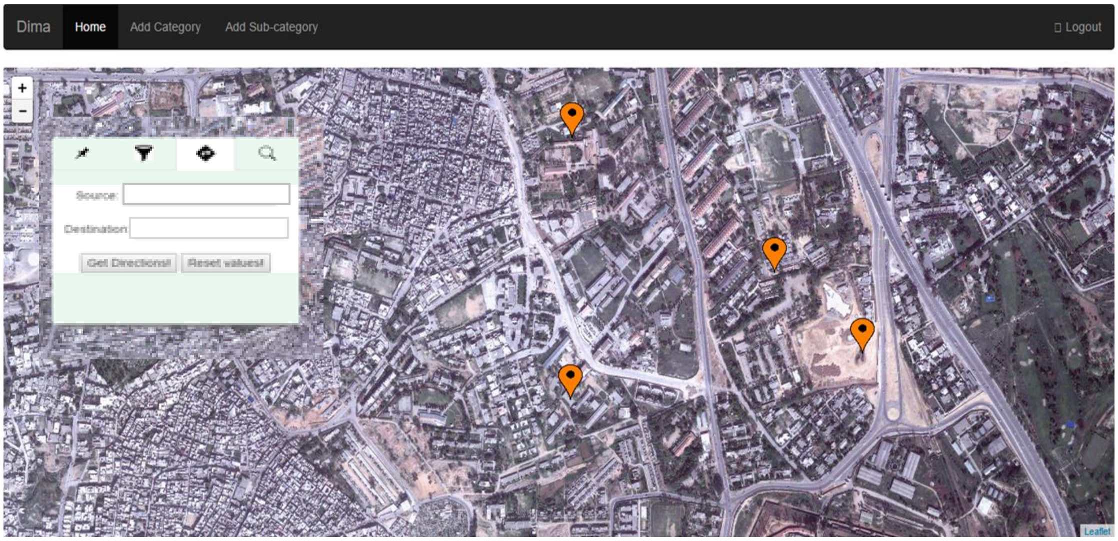


Figure 13-6 Reset to Main Screen

13.4 Logout:

On clicking the logout button on the top right of the screen you will bee directed to thee initial login screen.

13.5 Exit Application:

The system will save all settings, Terminate database and network connections and exit the application.