SRS Document for Railway Crossing Status Application

ADMIN EMAIL(HARD CODED) : yadavdhiraj.admin@gmail.com
ADMIN PASSWORD(HARD CODED) : d h i r a j 1 9 9 7

USER EMAIL(FROM DATABASE) : y1dhiraj@gmail.com

USER PASSWORD(FROM DATABASE) : dhiraj1997

1. Introduction

1.1 Purpose :

The purpose of this document is to outline the requirements for the development of a railway crossing status communication application. The application aims to provide the public with advance information about the status of railway crossings. It also includes Features for government personnel to manage and update the status of railway crossings. This document will serve as a reference for the development team and stakeholders involved in the project.

1.2 Scope:

The application will be developed using Java EE, with a frontend implemented using Java Server Pages (JSP) and a backend implemented using Servlets. The application will interact with a MySQL database using either JDBC or Hibernate for CRUD operations. The main features of the application include user authentication, fetching and displaying railway crossing details, search functionality, marking crossings as favorites, and government-level operations such as login, adding and deleting crossings, updating crossing status, and Providing easy navigation.

1.3 Definitions, Acronyms, Abbreviations and Project Configurations:

- SRS: Software Requirements Specification
- Editor: Eclipse
- Browser: Chrome
- Java EE: Java Enterprise Edition
- JSP: Java Server Pages
- Dynamic Web Module Version : 4.0
- Tomcat Version : 9.0

1.4 References:

1. Java Server Pages (JSP) Documentation: Official documentation for Java Server Pages, which includes information on how to develop dynamic web pages using JSP Technology.
    - Oracle JSP Documentation: https://docs.oracle.com/javaee/7/tutorial/jsf-intro.htm
2. Servlets Documentation: Official documentation for Java Servlets, which provides

information on how to create dynamic web applications using servlet technology.

- Oracle Servlets Documentation: https://docs.oracle.com/javaee/7/tutorial/servlets.htm

3. JDBC Documentation: Official documentation for Java Database Connectivity (JDBC), which provides information on how to interact with databases using JDBC for performing CRUD operations.
    - Oracle JDBC Documentation: https://docs.oracle.com/javase/tutorial/jdbc/index.html
4. MySQL Documentation: Official documentation for MySQL, a widely used open-source relational database management system.
    - MySQL Documentation: https://dev.mysql.com/doc/
5. HTML Documentation: Official documentation for HTML, the markup language used for creating web pages.
    - https://developer.chrome.org/en-US/docs/Web/HTML
6. CSS Documentation: Official documentation for CSS, the styling language used for designing web pages.
    - https://developer.chrome.org/en-US/docs/Web/CSS
7. Agile Software Development: A methodology that emphasizes iterative development, collaboration, and flexibility. Agile methods such as Scrum or Kanban can be used to manage the project.
    - Agile Manifesto: https://agilemanifesto.org/
    - Scrum Guide: https://scrumguides.org/

2. Overall Description:

2.1 Product Perspective :

The application will be a standalone system designed to communicate railway crossing status information to the public and enable government personnel to manage and update the crossing status. It will interact with a database to store and retrieve data.

2.2 User Classes and Characteristics:

Public Users: They will use the application to view railway crossing details, check the status, search for specific crossings, and mark crossings as favorites.

- Government Personnel: They will have access to the admin dashboard to manage railway crossings, including adding, deleting, and updating their status.

2.3 Operating Environment:

The application will be developed using Java EE and will be deployed on a suitable web server. The system will support modern web browsers.

2.4 Design and Implementation Constraints:

- The application will be implemented using Java EE technologies.
- The frontend will be developed using Java Server Pages (JSP) and HTML.
- The backend will be implemented using Servlets.
- The application will use JDBC for database interaction.
- The database management system used will be MySQL.

2.5 User Documentation:

The application will be accompanied by user documentation that provides instructions on how to use the features and functionalities of the application.

3. System Features:

3.1 Public Features :

1. Create an Account:
    - Allow users to register and create an account with basic details like name, email, and password.
    - Validate user input and ensure the uniqueness of email addresses.
2. User Login:
    - Provide an option for registered users to log in to the application.
3. Fetch Railway Crossings:
    - Retrieve and display the details of railway crossings from the database.
4. Display Railway Crossings with Status:
    - Display the list of railway crossings along with their current status (open or closed).
5. Search Crossings:
    - Implement a search functionality to allow users to search for specific railway crossings by name.
6. Mark Crossing as Favorite:
    - Allow users to mark a railway crossing as a favorite.
7. View Favorite Crossings:
    - Display a list of favorite railway crossings separately.

3.2 Government Features :

1. Admin Login:
    - Provide a secure login mechanism for government personnel to access the admin dashboard.
    - Authenticate users using pre-created email and password stored in the database.

2. Access Government Dashboard:
    - Allow authorized government personnel to access the admin dashboard.
3. Add Railway Crossing:
    - Provide a form for government personnel to add new railway crossings to the application.
    - Capture details such as name, address, landmark, train schedules, person in charge, and initial status.
4. Delete Railway Crossing:
    - Display a list of railway crossings and allow the deletion of selected crossings.
5. Update Crossing Status:
    - Enable government personnel to update the status of a railway crossing (open or closed).
6. Navigation Options:
    - Implement a navigation bar for easy access to all features and operations.

Algorithm:

1. Start the application.
2. Display the welcome screen with application name and developer details.
3. Display options for user interaction: User(Public) or Government.
4. If the user selects "User":
    - 4.1. Authenticate the user by providing options to createan account or login.
    - 4.2. Fetch details of railway crossings from the database.
    - 4.3. Display the list of railway crossings along with their status.
    - 4.4. Provide a search option to find a railway crossing by name.
    - 4.5. Allow the user to mark railway crossings as favorites.
    - 4.6. Display the list of favorite railway crossings separately.
5. If the user selects "Government":

    5.1. Authenticate the user as an administrator using pre-created email and password.

    5.2. Provide access to the government dashboard.

    5.3. Implement CRUD operations for managing railway crossings:
    - Add a new railway crossing to the application.
    - Delete a railway crossing from the application.
    - Update the status of a railway crossing.
    - Implement searchfunctionality to find specific railway crossings.
6. Handle exceptions and validate user input.
7. End the application.
8. **GITHUB Linke : https://github.com/dhiraj0803/JAVA-fsd.git**