Project Management System

A PROJECT REPORT

Submitted by

Raj Kumar Singh (23BCS11393)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

Chandigarh University

Nov, 2025

BONAFIDE CERTIFICATE

Certified that this project report "**Project Management Syste"** is the bonafide work of **"Raj Kumar Singh"** who carried out the project work under my/our supervision.

SIGNATURE **SIGNATURE**

Dr. Sandeep Singh Kang

BATCH HEAD **SUPERVISOR**

BE-CSE **BE-CSE**

TABLE OF CONTENTS

INTRODUCTION

1.1. Introduction to Project

The **Project Management System** is a full-stack web application designed to simplify and automate the process of managing projects, assigning tasks, and tracking work progress across multiple roles — *Admin, Manager,* and *Employee*.
It provides an integrated dashboard-based interface for all users where:

- The **Admin** can create and assign projects to Managers.
- The **Manager** can assign tasks to Employees and monitor their performance.
- The **Employee** can view assigned tasks and update their completion status.

The system is built using **ReactJS** for the frontend, **Spring Boot** for the backend, and **MySQL** as the database. REST APIs handle communication between the frontend and backend using **Axios**.

1.2. Identification of Problem

In many organizations, project tracking and task management are often handled manually through emails, Excel sheets, or physical meetings.
This approach leads to:

- Lack of centralized visibility of project progress.
- Difficulty in assigning or monitoring tasks.
- Poor communication between managers and employees.
- Errors due to manual data entry.

Hence, a centralized **web-based project management system** is required to improve efficiency and transparency in the workflow.

LITERATURE REVIEW/BACKGROUND STUDY

2.1 Existing Solutions

Several existing solutions like **Trello**, **Asana**, and **Jira** offer project management capabilities.
However, these tools are often complex or subscription-based, making them unsuitable for small or mid-sized organizations.
Moreover, they don't allow full customization for specific workflows and are dependent on third-party cloud integrations.

2.2 Problem Definition

The goal is to design and develop a **custom, lightweight, and role-based project management system** that:

- Allows Admins, Managers, and Employees to log in securely.
- Enables Admins to create, assign, and monitor projects.
- Provides Managers the ability to assign tasks and check Employee progress.
- Lets Employees view assigned tasks and update their status.

2.3 Goals/Objectives

1. To create a full-stack web application using **ReactJS** and **Spring Boot**.
2. To implement **RESTful APIs** for communication between client and server.
3. To ensure **secure user authentication** and role-based authorization.
4. To maintain project and employee data in a **MySQL database**.
5. To provide real-time project updates for Admin, Manager, and Employee roles.
6. To design a **responsive and user-friendly interface** using **Bootstrap**.

DESIGN FLOW/PROCESS

3.1 System Architecture

The system follows a **three-tier architecture**:

1. **Frontend Layer (ReactJS):** Handles user interface, navigation, and API requests using Axios.
2. **Backend Layer (Spring Boot):** Manages business logic, authentication, and API endpoints.
3. **Database Layer (MySQL):** Stores all project, user, and task-related data.

Flow:

```
User → ReactJS (Frontend) → REST API (Spring Boot) → MySQL Database
```

3.2 Features

- **User Authentication:** Secure login and registration for Admin, Manager, and Employee.
- **Admin Module:** Add/view projects, assign projects to Managers.
- **Manager Module:** Assign projects to Employees, view Employee performance.
- **Employee Module:** View assigned tasks, update task status.
- **Responsive Dashboard UI:** Built with Bootstrap and React Router for smooth navigation.
- **Database Integration:** MySQL database to store persistent project data.
- **Error Handling:** Backend validation using Spring Boot annotations and frontend alerts.

IMPLEMENTATION

Implementation of Solution

1. **Frontend (ReactJS):**
    - Implemented using **React Router DOM** for navigation.
    - Axios used for API communication.
    - Separate dashboard components for each user role.
2. **Backend (Spring Boot):**
    - Created REST API endpoints for authentication and project management.
    - Used **Spring Data JPA** to interact with the MySQL database.
    - Implemented **CORS configuration** to allow frontend-backend communication.
3. **Database (MySQL):**
    - Designed tables: `users`, `projects`, and `tasks`.
    - Each user has a role (Admin/Manager/Employee).
    - Relationships:
        - One Admin → Many Managers
        - One Manager → Many Employees
        - One Employee → Many Projects

RESULTS ANALYSIS AND VALIDATION

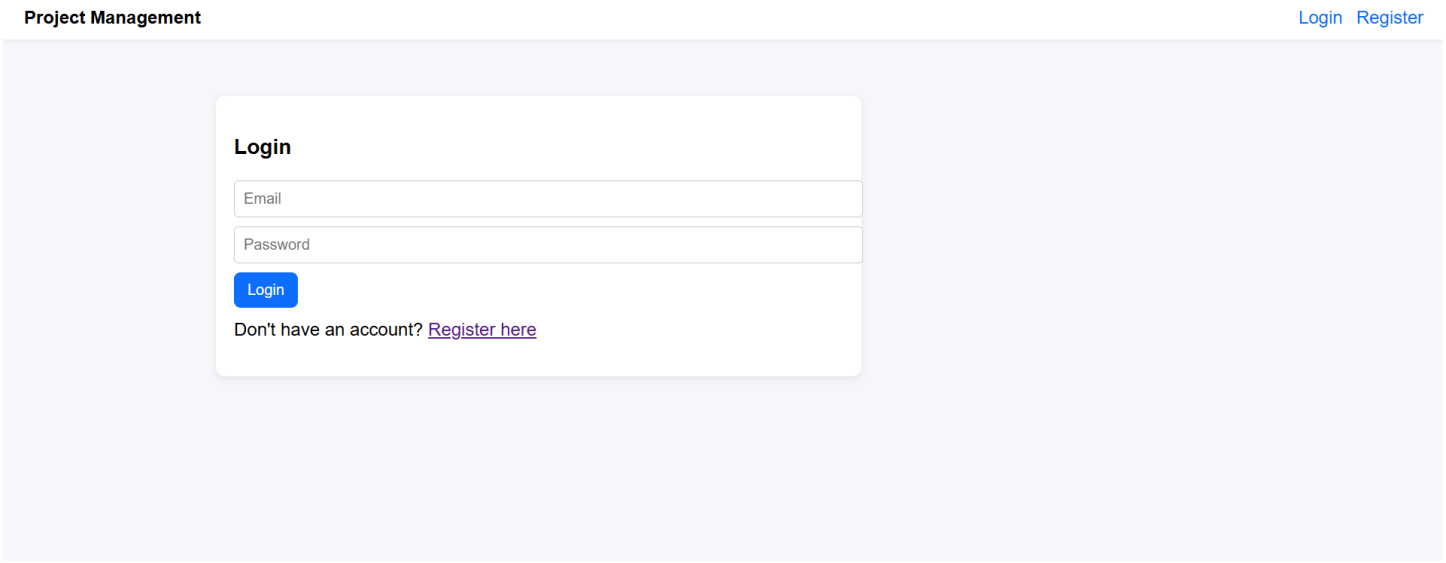
Results or Outputs

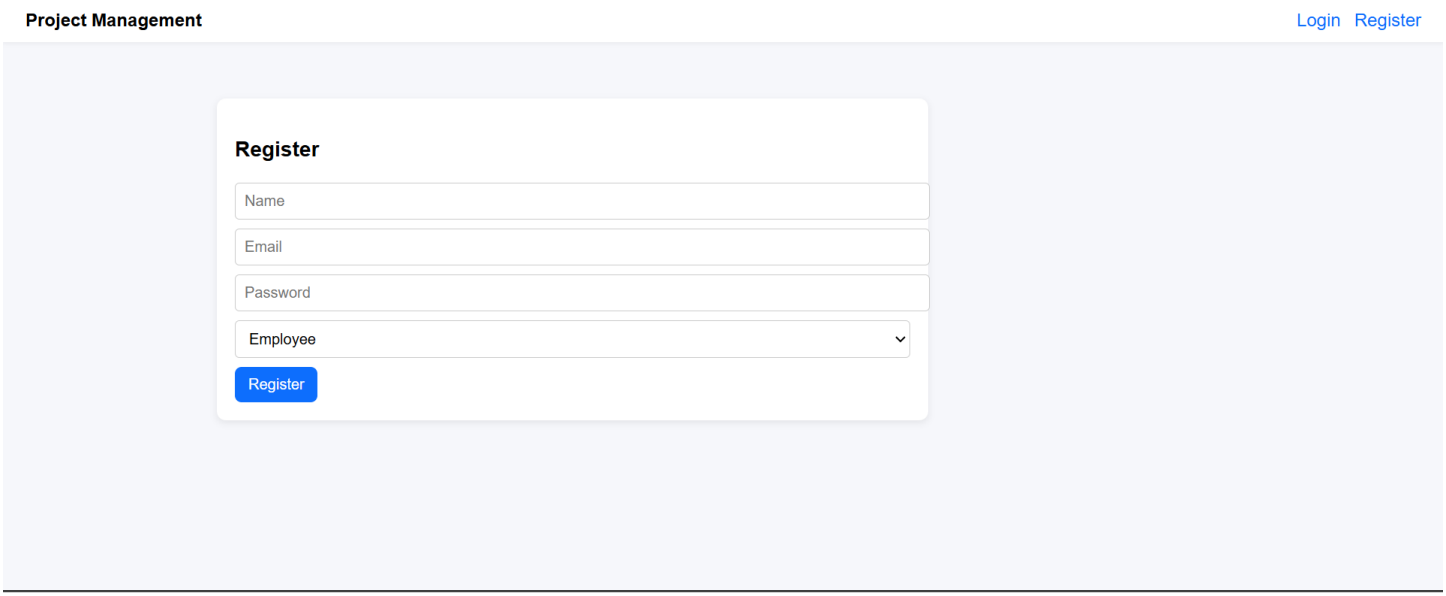
- Successful authentication of all user roles.
- Admin dashboard displays all projects and allows new additions.
- Manager dashboard allows assigning projects to Employees and viewing their status.
- Employee dashboard shows assigned projects and enables task status updates.
- Data stored and retrieved seamlessly from MySQL database.
- Fully functional communication between frontend and backend via REST APIs.

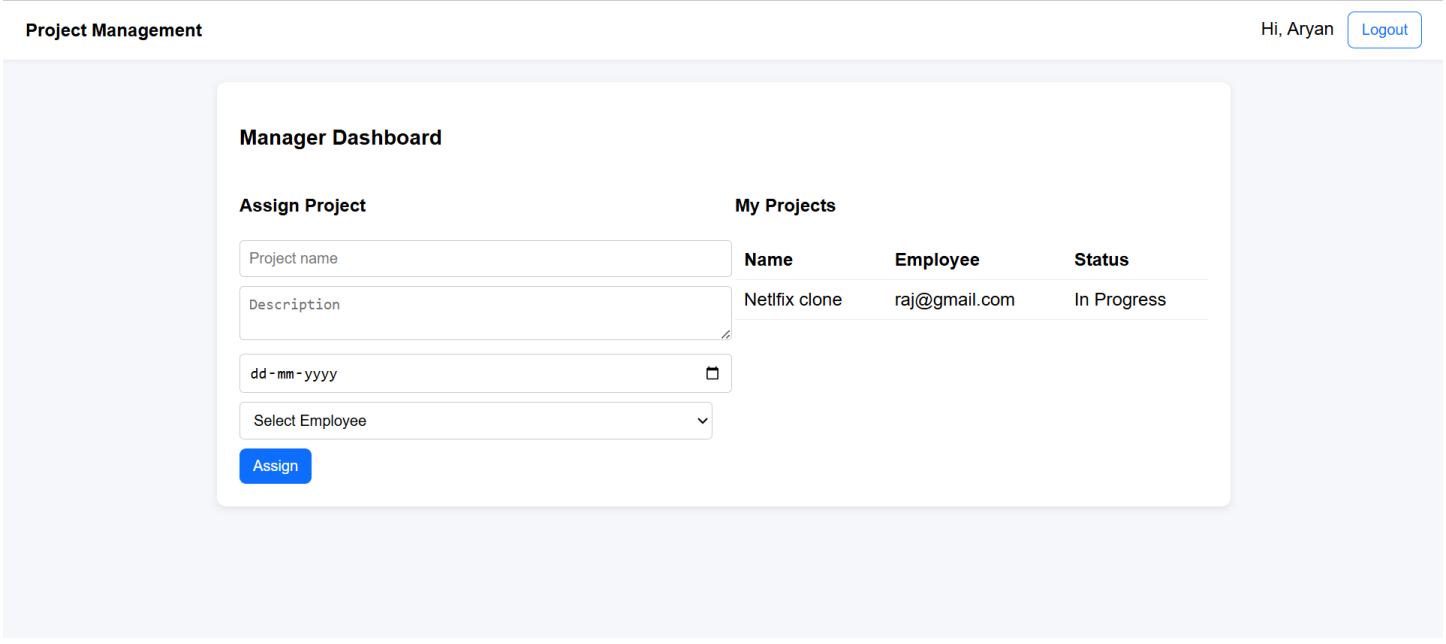
Screenshots :

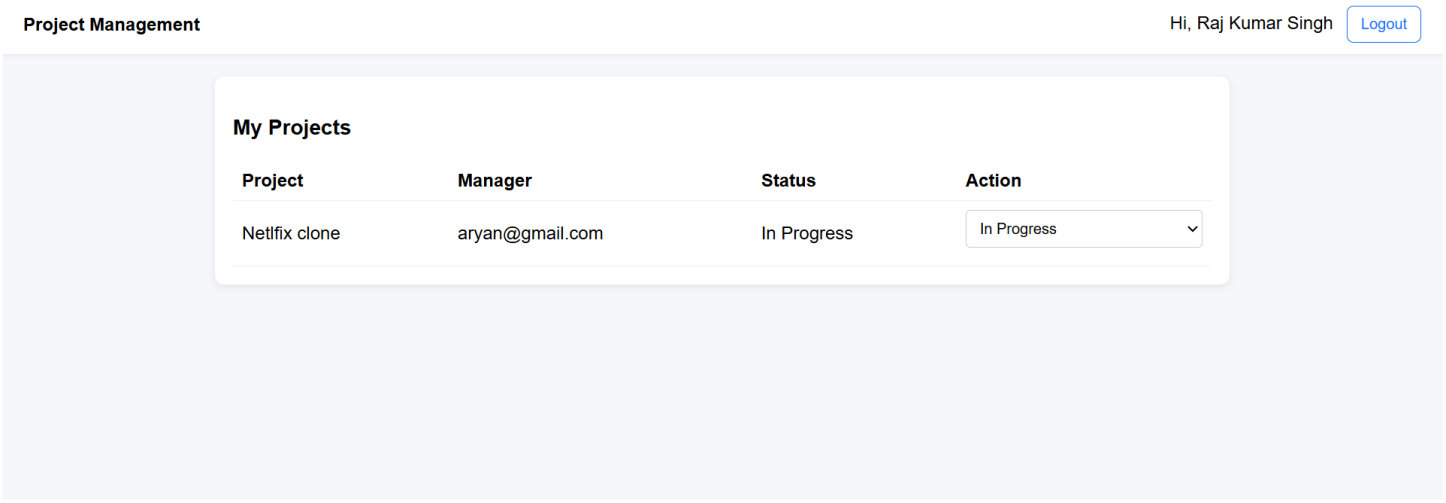
1. Login & Register Pages
2. Admin Dashboard (Add/View Projects)
3. Manager Dashboard (Assign/View Status)
4. Employee Dashboard (Update Task)

RESULTS/ OUTPUT:








Project Management
Hi, Raj Kumar Singh
Logout
My Projects
Project
Manager
Status
Action
Netlfix clone
aryan@gmail.com
In Progress
In Progress

CONCLUSION AND FUTURE WORK

6.1 Conclusion

The **Project Management System** successfully automates project and task tracking.
It provides a role-based, user-friendly interface for efficient collaboration between Admin, Manager, and Employees.
Using **ReactJS, Spring Boot, and MySQL**, it ensures scalability, modularity, and real-time data synchronization.

6.2 Future Work

1. Add **Email notifications** for new task assignments.
2. Implement **File Uploads** (documents or reports) per project.
3. Integrate **Analytics Dashboard** for progress tracking.
4. Add **JWT-based Authentication** for enhanced security.
5. Deploy the system to **cloud platforms** like AWS or Render for real-world use.