COMPLAINT REDRESSAL SYSTEM

A PROJECT REPORT

Submitted by

PRADEEPRAJ I

PRASHANTHA JN

PRAJWAL S

PRASHANTH SHINDE

VARSHITHA RN

R SANGEETHA

POONAM KUMARI

SURYA RAMYASREE

DARUR SWETHA

RAJENDRA PRASAD K

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Mphasis Ltd is a Company which builds the software system which is responsible for adding and processing of a product .

Mphasis Ltd plans to develop "COMPLAINT REDRESSAL SYSTEM" -Web Application,where users can register,login,raise their complaints to the management and resolve their problems in the system.

It provides the solutions for users to get in touch with management and complete work and raise complaints with their issues through this system .

1.1 SCOPE AND OVERVIEW

The 'COMPLAINT REDRESSAL SYSTEM' will be to provide the functionality as described below .this system will be developed on a windows operating system using Java,spring boot,hibernate,my sql.

1.2 SYSTEM OVERVIEW

The 'COMPLAINT REDRESSAL SYSTEM' should support basic functionalities for all users that CUSTOMER,MANAGER,ENGINEER and ADMIN.

1.2.1 AUTHENTICATION

Any end-user has been authenticated using a unique userid and password.

1.2.2 AUTHORIZATION

The operations supported and allowed based on user type For example Admin has right to access all the users from this system and he can see all the details of customer,manager and engineer.WhereAs Customer and Manager has a right to perform an crud operations like update,delete and Add the details from the system.

CHAPTER 2

FUNCTIONAL SPECIFICATION

2.1 FUNCTIONAL FLOW

The functional flow of this system has been performed across different applications as shown below .

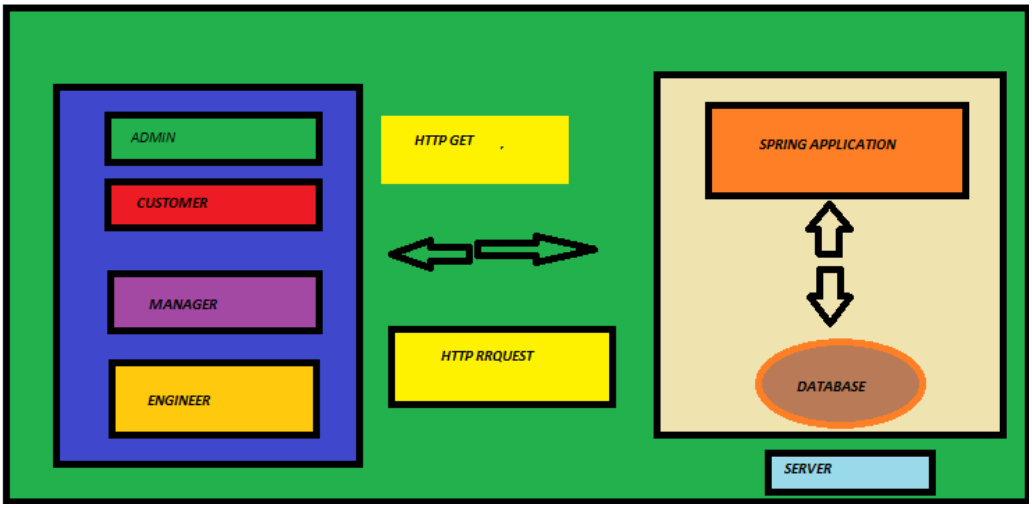

Figure 2.1: Admin Table

2.2 ENVIRONMENT

The system will be developed on any windows OS machine using My Sql,hibernate,spring boot

Intel Hardware Machine

Server - Apache Tomcat

Database - MySql

Eclipse IDE using Spring STS tool

CHAPTER 3

SUB SYSTEM DETAILS

The Complaint Redreesal System defined ,wherein all users need to login successfully before performing any of their respective operations.

Find Below Tables that provides functionality descriptions for each type of user/subsystem.againist each requirement ,indiactive data is listed in column'Data to include'.Further,suggested to add/modify more details wherever required with an approvel from customer/manager.

3.1 ADMIN

The Admin as a user is defined to perform below listed operations after successful login.

ID	OBJECTS	OPERATIONS	DATA TO INCLUDE	REMARKS
AD-21	customer	ADD,View, Delete,Modify	Customer id, Customer name, ticket id , ticketstatus, address, pin code	
AD-21	manager	View	Manager id, managername, assigned tickets, tickets status	
AD-21	Engineer	View	Eng Id,Eng name,Assigned tickets,ticket status	

Figure 3.1: Admin Table

3.2 Customer

The Customer as a defined to perform below listed operations after login succesfully.

CUST ID	OBJECTS	OPERATIONS	DATA TO INCLUDE
21	user	register	Custid,username, password,email, contactNumber, address,pincode
22	Ticket Generation	Login	Raise the ticket,view tickets,ticket status,manager id,manager name
23	LogOut	Click on logout button	Back to Home page

Figure 3.2: Customer Table

3.3 Manager

The Manager as a user is defined to perform below listed operations

MGR ID	OBJECTS	OPERATIONS	DATA TO INCLUDE
21	CUSTOMER	VIEW	cust id,cust name,address, pinocle
22	TICKETS	Update,delete and modify	Ticket id,ticket status
23	Logout	Click on logout button	Back to Home page

Figure 3.3: Manager Table

3.4 Engineer

The Engineer as a user is defined to perform below operations

ENGINEER ID	OBJECTS	OPERATIONS	DATA TO INCLUDE
21	CUSTOMER	VIEW	cust id,cust name,address, Pinocode,ticket id,ticket status
22	MANAGER	View	Manager id,manager name ,ticket id,ticket status
23	Logout	Click on logout button	Back to Home page

Figure 3.4: Engineer Table

3.5 Login And Logout

[Web Application - MySql, Hibernate, Spring]

-Go to Registration screen when you click on Register link.

-Go to Success screen when you login successfully after entering valid username password fetched from the database.

-Redirect back to Home page after click on logout

-Implement Session tracking for all logged in users before allowing access to application features. Anonymous users should be checked, unless explicitly mentioned.

CHAPTER 4

DATA ORGANIZATION

This section explains the data fields requirements of the admin, customer,manager and engineer and indicative data description along with suggested table (database) structure. The following section explains few of the tables (fields) with description. However, in similar approach need to be considered for all other tables.

4.1 Customer Details

FIELD NAME	DESCRIPTION
Customer id	Customer id is Auto generated after registration and it is used as LoginId
User name	Username of the customer
password	Customer password
email	Customer email ID
adress	Customer adress
pin code	Pin code of the current location

Figure 4.1: fields of the customer

4.2 Manager Details

FIELD NAME	DESCRIPTION
Manager id	Manager id which is here primary key
Manager name	Name of the manager
Customer id	Customer ID
Customer name	Name of the customer who raised the ticket
Customer adress	Customer adress
Ticket status	In progress or resolved or escalated

Figure 4.2: Fields Required in Manger

4.3 Engineer

FIELD NAME	DESCRIPTION
Manager id	Manager id
Manager Name	Name of the supposed manager who has been assigned the ticket to the engineer
Customer name	Name of the customer who raised the ticket
Customer adress	Customer adress
Ticket status	In progress or resolved or escalated

Figure 4.3: Fields required in Engineer

CHAPTER 5

REST API'S TO BE BUILT

Create following REST resources which are required in the application,

1.Creating User Entity: Create Spring Boot Application with Spring Data JPA

5.1 Techonology stack

-Spring boot

-spring rest

-spring data JPA

Here will have multiple layers into the application

1.Create an Entity:customer

2.Create a customer Repository interface and will make use of Spring Data JPA and Will have find By CustName method and Add the customer details.

3.Create a CustomerService class and will expose all these services.

4.Finally, create a CustomerRestController will have the following Uri's:

URI	METHODS	Description	Format
/Customers/customerName	GET	Give a single customer description searched based on username	JSON
/customers/customerid	GET	Give a single user description searched based on customer id	String
/customers	POST	Add the customer details	JSON
/customers	PUT	Update the customer details	JSON
/customers/customerid	DELETE	Delete customer by id	String

Figure 5.1: Uri and methods of the customer

5.2 Creating Manager entity

Build a RESTful resource for Manager, where CRUD operations to be carried out. Here will have multiple layers into the application:

1. Create an Entity: Manager

2. Create a ManagerRepository interface and will make use of Spring Data JPA

-Will have findByManagerName method.

-Add the Manager details method.

-Will have deleteManagerById method.

- Will have findAllManagers method.

3.Create a ManagerService class and will expose all these services.

4.Finally, create a ManagerRestController will have the following Uri's:

URI	METHODS	Description	Format
/Managers/ Manager Name	GET	Give a single Manager description searched based on username	JSON
/ Managers / Managerid	GET	Give a single user description searched based on Manager id	String
/ Managers	POST	Add the Manager details	JSON
/ Managers	PUT	Update the Manager details	JSON
/ Managers / Managerid	DELETE	Delete Manager by id	String

Figure 5.2: Uri and methods of the Managers

5.3 Creating Engineer entity

1.Create an Entity: Engineer

2.Create a EngineerRepository interface and will make use of Spring Data JPA

3.Create a EngineerService class and will expose all these services.

4.Finally, create a EngineerRestController will have the following Uri's:

URI	METHODS	Description	Format
/Engineers/ Engineer Name	GET	Give a single Engineer description searched based on username	JSON
/ Engineers / Engineerid	GET	Give a single user description searched based on Engineer id	String
/Engineers	POST	Add the Engineer details	JSON
/ Engineers	PUT	Update the Engineersdetails	JSON
/ Engineers / Engineerid	DELETE	Delete Engineer by id	String

Figure 5.3: Uri and methods of the Engineer

5.4 Creating Admin entity

1.Create an Entity: Admin

2.Create a AdminRepository interface and will make use of Spring Data JPA

3.Create a AdminService class and will expose all these services.

4.Finally create the All AdminRestControllers .

CHAPTER 6

OUTPUT

6.1 HomePage

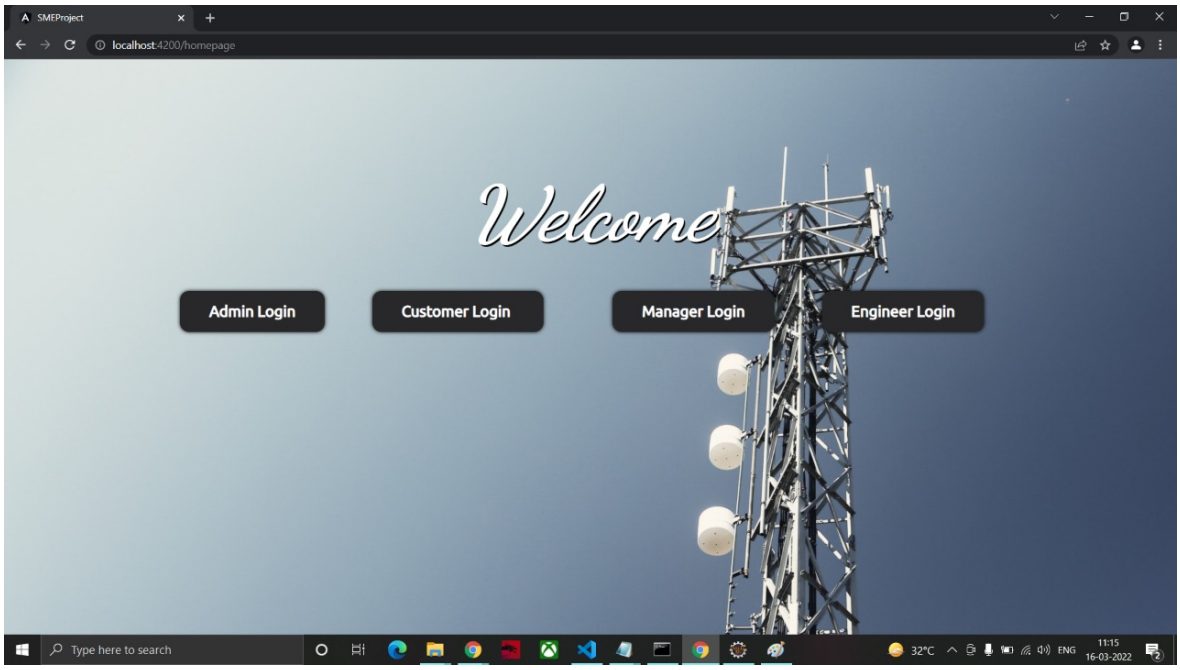

Figure 6.1: Home Page

6.2 Customer Login

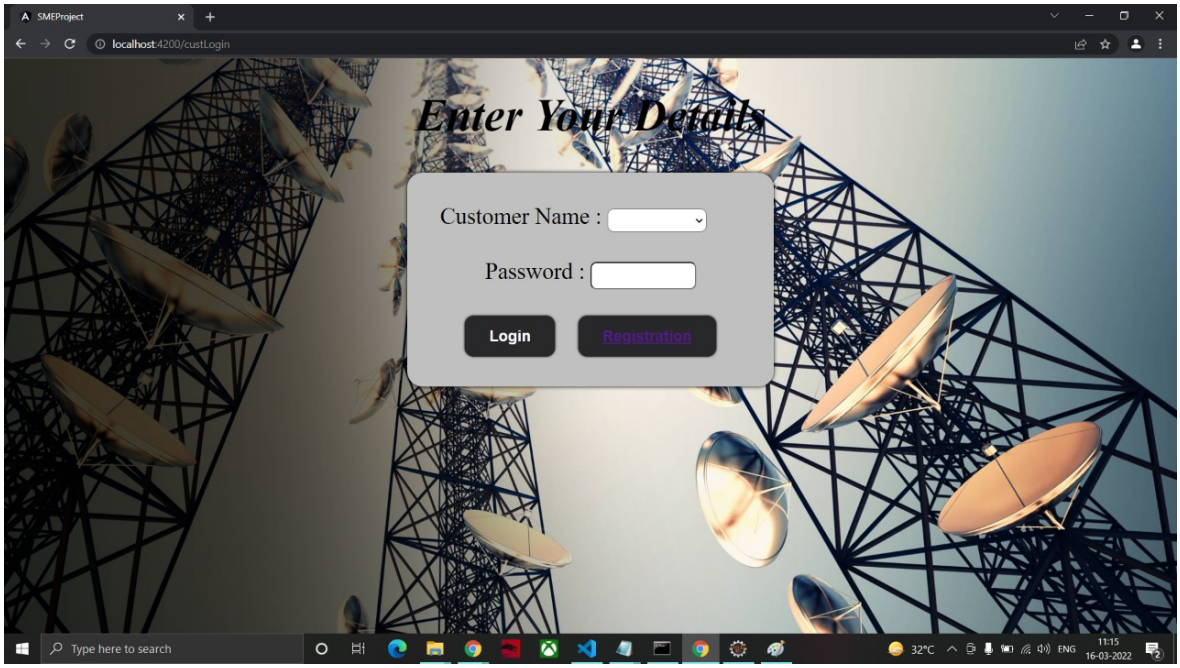

Figure 6.2: Customer login

6.3 Customer registration

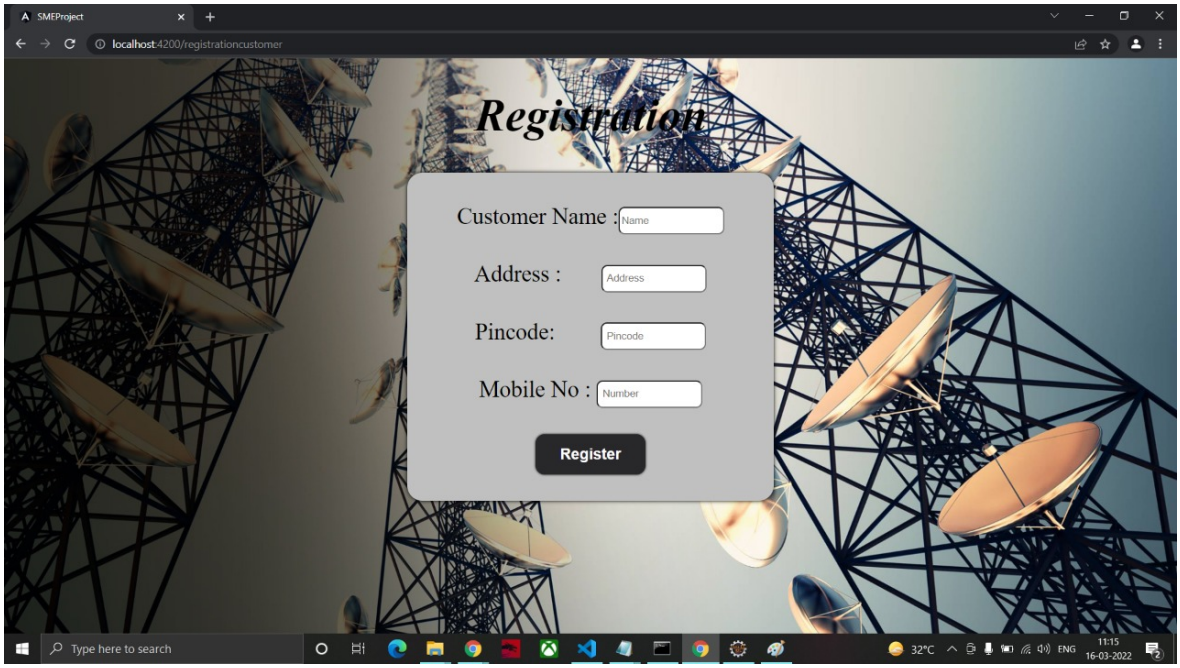

Figure 6.3: Customer registration page

6.4 Customer Dashboard

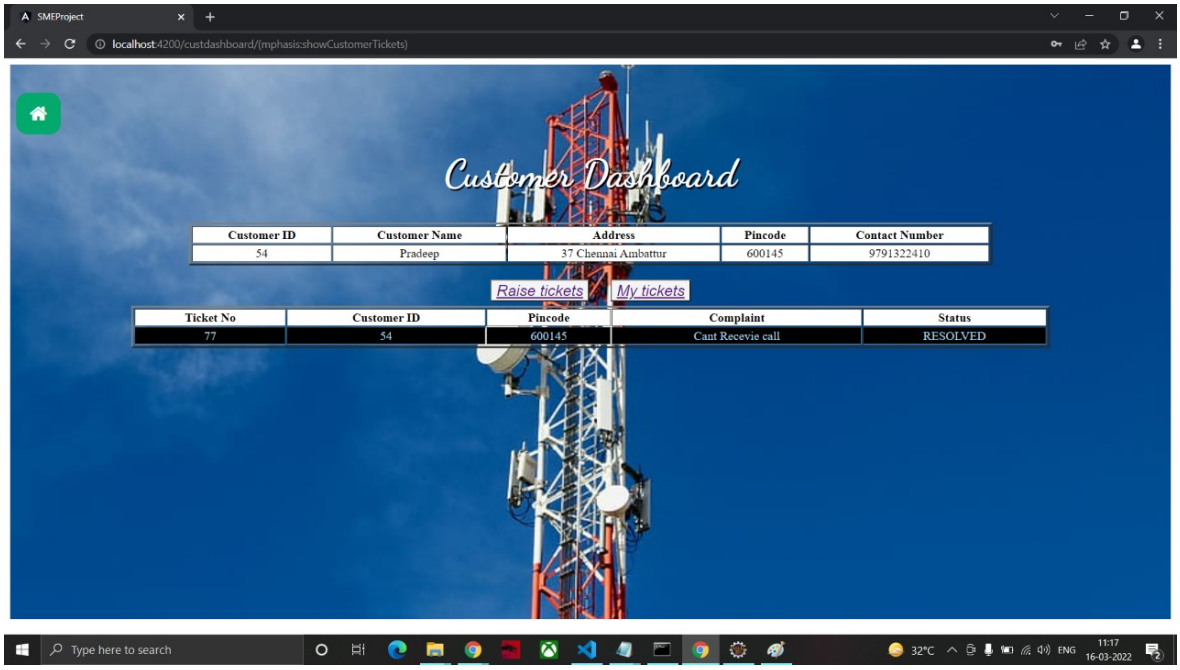

Figure 6.4: Customer dashboard

6.5 Manager Login

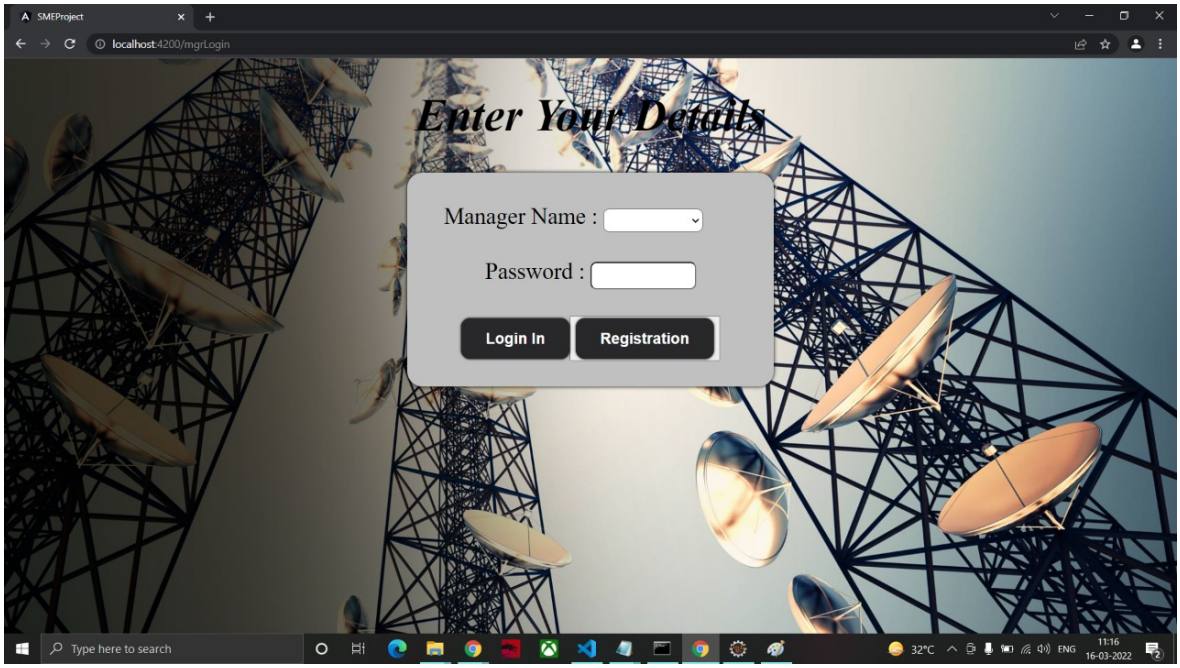

Figure 6.5: Manager Login

6.6 Manager Registration

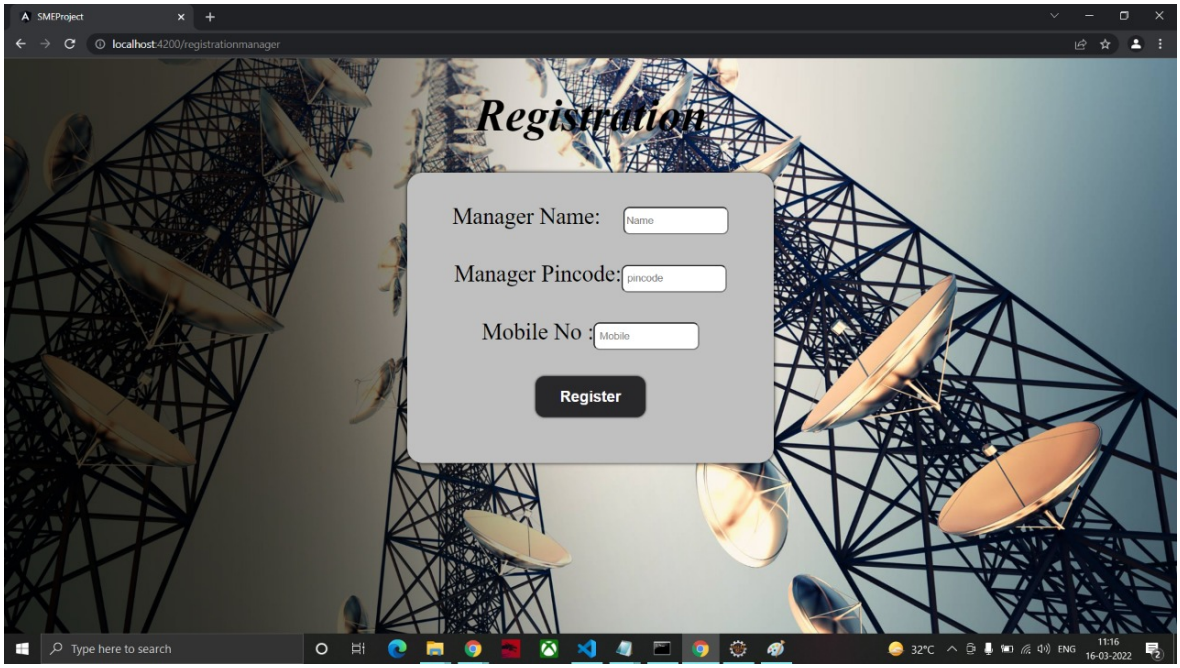

Figure 6.6: Manager Registration

6.7 Manager Dashboard

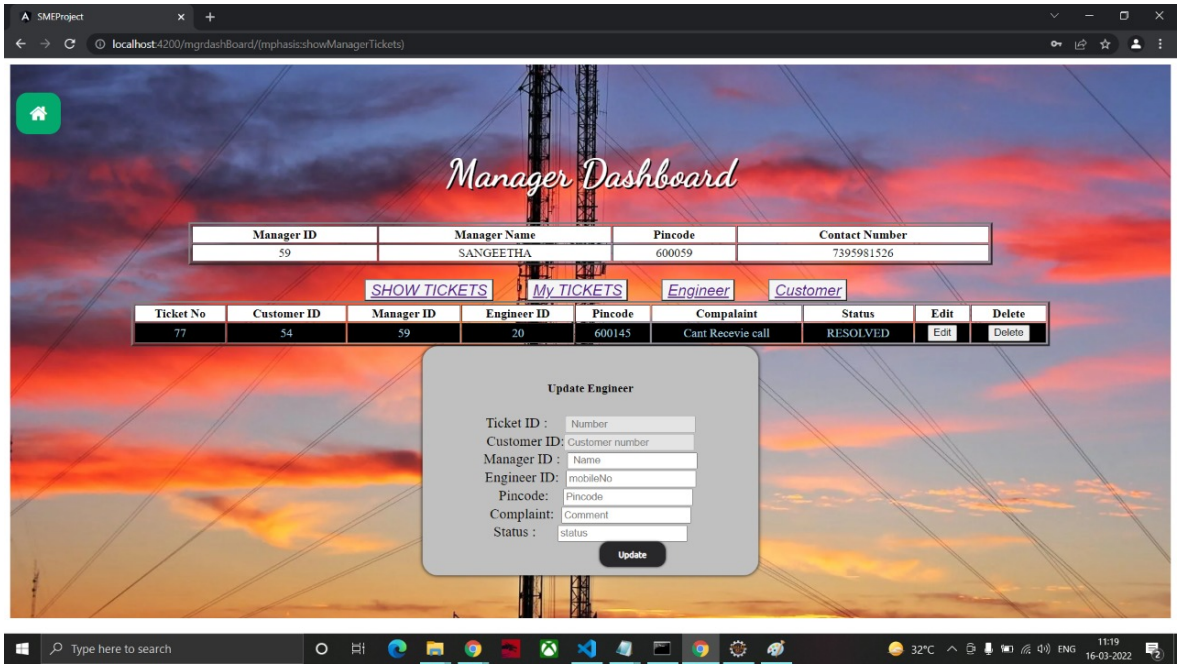

Figure 6.7: Manager Dashboard

6.8 Engineer Login

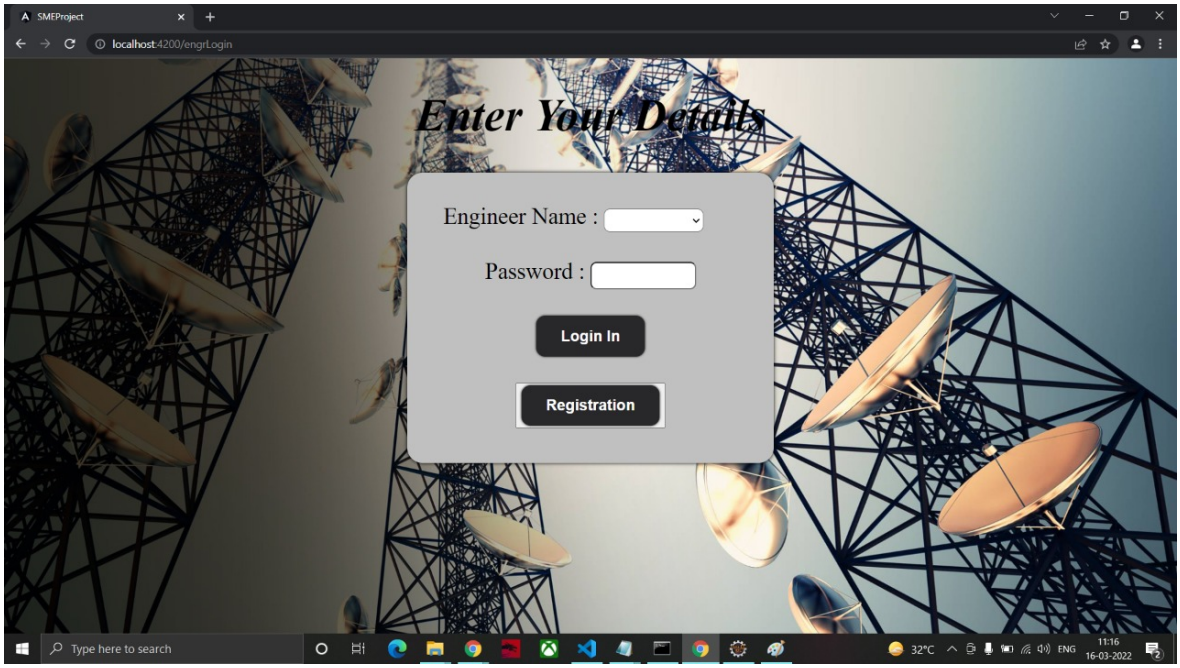

Figure 6.8: Engineer Login

6.9 Engineer Registration

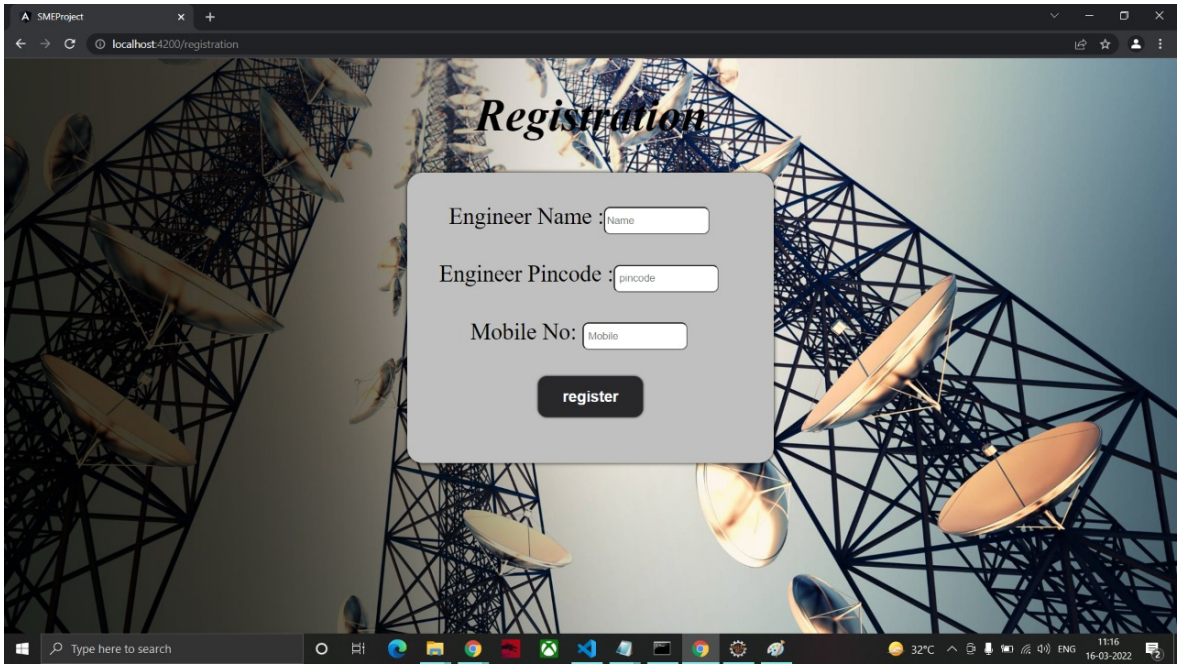

Figure 6.9: engineer registration

6.10 Admin Login

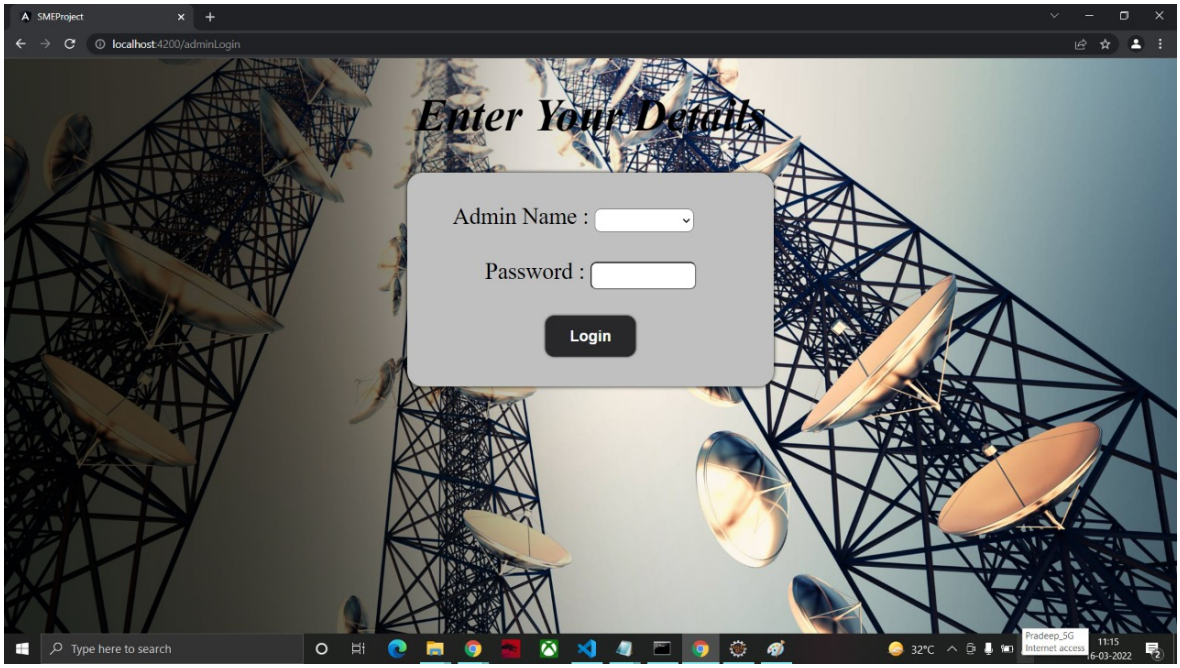

Figure 6.10: Admin Login

6.11 Admin dashboard

Figure 6.11: Admin DashBoard