HOSTEL MANAGEMENT SYSTEM

A project report submitted in partial fulfilment of the requirement for degree

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

SUBMITTED BY

R170288 V Sindhu

R170351 N Vennela

R170103 P Jnana Prasanna

UNDER THE GUIDANCE OF

Ms C. Suneetha

Asst.Prof In Department Of Computer Science & Engineering

AP IIIT,RGUKT-RK Valley,
Vempalli,Kadapa(Dist),Andhra Pradesh-516330,India

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008)
RGUKT-RK Valley
Vempalli,Kadapa,Andrapradesh-516330.

CERTIFICATE OF PROJECT COMPLETION

This is to certify that the report entitled "**HOSTEL MANAGEMENT SYSTEM** " submitted by V Sindhu bearing ID NO: R170288, N Vennela bearing ID NO: R170351 and P Jnana Prasanna bearing ID NO: R170103 in partial fulfillment of the requirements for the award of Bachelor Of Technology in Computer Science Engineering is a bonafide work carried out by them under my supervision and guidance.

The report has not been submitted previously in part or in full to this or any other university or institution for the award of any degree or diploma.

Project Guide

C.Suneetha,

Assistant Professor,

Dept of CSE,

RK Valley, RGUKT.

Head of Department,

N Satyanandaram,

Assistant Professor,

Dept of CSE,

RK Valley, RGUKT.

DECLARATION

We hereby declare that this report entitled “Hostel Management System” submitted by us under the guidance and supervision of Miss C Suneetha is a bonafide work.We also declare that it has not been submitted previously in part or in full to this university or other university or institution for the award of any degree or diploma.

We also declare that this project is a result of our own effort and has not been copied or imitated from any source.Citations from any websites are mentioned in the references.

We will be solenly responsible if any kind of plagiarism is found.

Date:

Place:RK Valley

R170288,V Sindhu

R170351,N Vennela

R170103,P Jnana Prasanna

\

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to C.Suneetha, Our project guide for valuable suggestions and keen interest throughout the period of project work.

We are grateful to N.Satyanandaram,HOD of CSE for providing excellent computing facilities and a congenial atmosphere for progressing with our project.

We would also like to extend our deepest gratitude & reverence to the Director of RGUKT , RK Valley **Prof.K.Sandyarani** and HOD of Computer Science and Engineering **Mr. N. Satyanandaram** for their constant support and encouragement.

We express our thanks to all those who contributed for the successful completion our project work.

With gratitude,

V Sindhu R170288

N vennela R170351

P Jnana Prasanna R170103

TABLE OF CONTENTS

PROBLEM STATEMENT

The Hostel Management System is developed for automating the complaints of Hostel. The software will be the great relief to the users The software will help user in case of complaints and see the status of the complaints and create as many as request is possible and can delete or edit the request .The aim of Hostel Management System is to carry out the complaints of hostel in an efficient way. It will take the operations of Hostel complaints to an upper level by providing faster access to data and allowing to create, delete ,edit and assign the work to the person . In existing system there is lot of manual work to overcome this we use .NET technology to build a website.

ABSTRACT

"Hostel Management System" is a web based portal for students complainting about hostel issues for hostel.Where a student can raise a ticket about an issue in hostel and admin can assign a work for raised ticket to the employees.This portal is fully user controlled where a user can raise a complaint and user can see the status of the issue.The admin can be any registered user.He can control all the activities in the portal.He is responsible for assigning the complaints to employees that is raised by student.This portal is designed for the hostel students to complaint about the issues related to hostel.This portal is user friendly to students.

We can conclude that our proposed model produced the highest accuracy while using .NET technology to build web based portal to complaint about issues in hostel.

INTRODUCTION

This document has the requirements of hostel complaints.The Hostel Management system is used to complaint about hostel related issues to the warden or admin**.**

1.1:Purpose

The purpose of this document is to gather the requirements that are needed for implementing the Hostel Management System.It also focuses on various key features,the product,product vision and scope,product overview.The main purpose of Hostel Management System is to provide a platform to the university students to complaint about hostel issues to the admin/warden.

1.2:Intended Audience

The intended audience will be the users who can access the platform to complaint about issues related to hostel to the admin and admin can assign work to the respected employees.The user can see the status of the complaint he raised and create as many request as possible.

USERS:

1.Students
2.Admin
3.Employee

Product vision:

Vision Statement:

The product vision is to develop a Hostel Management System,which is user friendly and easily accessible.The Hostel Management System helps to complaint about issues in hostel to admin where admin can assign work too the employee.

Technologies:

1.ASP.NET CORE MVC
2.HTML
3.SQL

Hardware Used:

1.Microsoft visual studio 2022
2.Sql Server Management-SQL 2019-SSEI-Expr

3.SSMS-SETUP-EMU

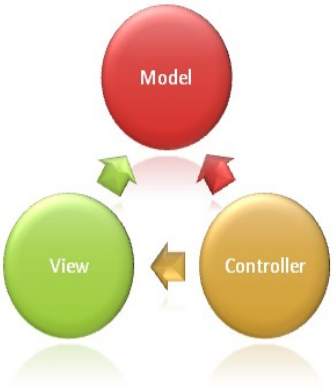
ASP.NET CORE MVC:

ASP.NET CORE MVC is a rich framework for bulding web apps and APIs using the Model-View-Cotroller design pattern.

MVC Pattern

The Model-View-Controller (MVC) architectural pattern separates an application into three main groups of components: Models, Views, and Controllers. This pattern helps to achieve separation of concerns. Using this pattern, user requests are routed to a Controller which is responsible for working with the Model to perform user actions and/or retrieve results of queries. The Controller chooses the View to display to the user, and provides it with any Model data it requires.

The following diagram shows the three main components and which ones reference the others:



Model Responsibilities

The Model in an MVC application represents the state of the application and any business logic or operations that should be performed by it. Business logic should be encapsulated in the model, along with any implementation logic for persisting the state of the application. Strongly-typed views typically use ViewModel types designed to contain the data to display on that view. The controller creates and populates these ViewModel instances from the model

View Responsibilities

Views are responsible for presenting content through the user interface. They use the Razor view engine to embed .NET code in HTML markup. There should be minimal logic within

views, and any logic in them should relate to presenting content. If you find the need to perform a great deal of logic in view files in order to display data from a complex model, consider using a View Component, ViewModel, or view template to simplify the view.

Controller Responsibilities

Controllers are the components that handle user interaction, work with the model, and ultimately select a view to render. In an MVC application, the view only displays information; the controller handles and responds to user input and interaction. In the MVC pattern, the controller is the initial entry point, and is responsible for selecting which model types to work with and which view to render (hence its name - it controls how the app responds to a given request).

SQL

- SQL stands for Structured Query Language
- SQL lets you access and manipulate databases
- SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.

The SQL UPDATE Statement:

The `UPDATE` statement is used to modify the existing records in a table.

UPDATE *table_name*
SET *column1* = *value1*, *column2* = *value2*, ...
WHERE *condition*;

The SQL SELECT Statement:

The `SELECT` statement is used to select data from a database.

The data returned is stored in a result table, called the result-set.

SELECT *column1*, *column2, ...*
FROM *table_name*;

SQL CREATE DATABASE Statement:

The CREATE DATABASE statement is used to create a new SQL database.

CREATE DATABASE *databasename*

EXISTING SYSTEM

The existing system is manual based and need lot of efforts and consume enough time.In the existing system we are complainting about hostel issues in the warden office.It leads a lot of time to solve the issues as well as to assign the work to the respective employee.The user cannot view the raised complaint status.

DRAWBRACKS OF EXISTING SYSTEM:

- All the work is done manually .
- Hostel complaints and other information are all kept in a booklet.
- All complaints about hostel like repair in taps ,fans,bulbs, etc have to complaint in warden office.
- Student complaints, and staff information all these information are kept in registers.
- The complaint related to fans,bulbs,lights will be solve lately due to manual work.
- The employee have to come and check the work assigned to him manually and the process will be lately.
- The student will not know the status of the complaint in the manual process.

Here we use the manual system, but this could not get the high accuracy.

PROPOSED SYSTEM

The project is aimed at developing a system for keeping records and showing information about hostel issues. The system will help the hostel office to be able to manage the problems of the hostel.It will shows the complaints raised by student to admin so that admin can assign work to the respected employees and they can see the work assigned to them.

ADVANTAGES OF USING THIS:

- A single user can raise multiple complaints and the admin can see the all the complaints raised by different users so that he can assign work to that respected employee.
- The student can see the status of the complaint in the student module.
- The admin can see the issues raised by the students .
- The admin can allot the work to the respected employee like plumber,electrician etc.
- The employee can check the work assigned to them individually.
- If the employee solve the problem he can edit the progress of issues raised by the students.
- The complaint will be solved in intime.

Hence, proper classification is important to manage this system , which will be possible using our proposed method.

MODULES

1.ADMIN:

- The admin can assign the work to the employee.
- The admin can delete the complaint raised by user.
- The admin can create a new request of compliant.
- The admin can see the list of complaints raised by the users.
- The admin can be any registered user.

2.EMPLOYEE:

- The employee can solve the issue raised by the student.
- The employee can see the list of works assigned by the admin.
- The employee can edit the status of the compliant.

3.STUDENT:

- The user will register using the registration form.
- After register user will login to the student module.
- After login user can compliant about the issue using the create new request link.
- User can see the list of raised by them**.**
- User can edit or delete the particular request.
- User can see the request and assigned employee.

SOURCE CODE

MODELS:

USER

```
namespace TMS.Models{
public class User{
public string? UserId { get; set; }
public string? FirstName { get; set; }
public string? LastName { get; set; }
public string? Email { get; set; }
public string? Occupation { get; set; }
public string? Gender { get; set; }
public string? Address { get; set; }
public string? PassWord { get; set; }
public string? Department { get; set; }
}
}
```

REQUEST DETAILS

```
using TMS.Models;
namespace TMS.Models
{public class RequestDetails
{
public string? UserId { get; set; }
public string? RequestType { get; set; }
public string? RequestDescription { get; set; }
public string? RequestId { get; set; }
public string? RequestStatus { get; set; }
public string? AssignedTo { get; set; }
public string? Comments { get; set; }
}
}
```

VIEWS:

ABOUT:

INDEX

```
@{
ViewData["Title"] = "Index";}

<h3>About us</h3>
```

ASSIGN:

```
@model TMS.Models.RequestDetails
@{
```

```
ViewData["Title"] = "Edit";
}
<h4>RequestDetails</h4>
<hr />
<div class="row">
<div class="col-md-4">s
<form asp-action="Assign">
<div asp-validation-summary="ModelOnly" class="text-danger"></div>
<div class="form-group">
<label asp-for="UserId" class="control-label"></label>
<input asp-for="UserId" class="form-control" readonly/>
<span asp-validation-for="UserId" class="text-danger"></span>
</div>
<div class="form-group">
<label asp-for="RequestType" class="control-label"></label>
<input asp-for="RequestType" class="form-control" readonly />
<span asp-validation-for="RequestType" class="text-danger"></span>
</div>
<div class="form-group">
<label asp-for="RequestId" class="control-label"></label>
<input asp-for="RequestId" class="form-control" readonly />
<span asp-validation-for="RequestId" class="text-danger"></span>
</div>
<div class="form-group">
<label asp-for="RequestStatus" class="control-label"></label>
<select asp-for="RequestStatus" class="select" name="RequestStatus">
<option value="">Select</option>
<option value="Hold">Hold</option>
<option value="Closed">Closed</option>
<option value="Inprogress">Inprogress</option>
<option value="Assigned">Assigned</option>
</select>
<span asp-validation-for="RequestStatus" class="text-danger"></span>
</div>
<div class="form-group">
<label asp-for="AssignedTo" class="control-label"></label>
<input asp-for="AssignedTo" class="form-control"/>
<span asp-validation-for="AssignedTo" class="text-da
nger"></span
</div>
<div class="form-group">
<label asp-for="Comments" class="control-label"></label>
<input asp-for="Comments" class="form-control"/>
<span asp-validation-for="Comments" class="text-da
```

```
nger"></span>
</div>
<div class="form-group">
<input type="submit" value="Save" class="btn btn-primary" />
</div>
</form>
</div>
</div>

<div>
<a asp-action="Index">Back to List</a>
</div>
```

HOME:

INDEX:

```
@{
ViewData["Title"] = "Home Page";
}
<div class="text-center">
<h1 class="display-4">Hostel Management System</h1>
<p class="home_description">This website will help you to solve a problem by raising a ticet to hostel management team
and you can track the status of the ticekt using ticket id which is geneated by the system at the time of raising a ticket.
users will get the ticekt id at the time of raising a ticket. users can see the details for which the ticekt is assigned and
status of the ticket in this website. users can raise a ticket regarding any kind of hostel issues like water, electricity, lights,
taps, windows and doors.
</p>
</div>
<section class="h-100 gradient-form" style="background-color: #eee;">
<div class="container py-5 h-100">
<div class="row d-flex justify-content-center align-items-center h-100">
<div class="col-xl-10">
<div class="card rounded-3 text-black">
<div class="row g-0">
<div class="col-lg-6">
<div class="card-body p-md-5 mx-md-4">
@using (Html.BeginForm(FormMethod.Post))
{
<div class="user_login_form">
```

```
<h3>Login to your Account</h3>
<div class="form-outline mb-4">
<input type="email" id="form2Example11" name="txtUserName" class="form-
control”placeholder="Please Enter Email Id" />
<label class="form-label" for="form2Example11">Username</label>
</div>
<div class="form-outline mb-4">
<input type="password" id="form2Example22" name="txtPassWord" class="form-control" />
<label class="form-label" for="form2Example22">Password</label>
</div>
<div class="text-center pt-1 mb-5 pb-1">
<input type="submit" value="Log in" asp-action="Login" asp-controller="Home" />
<a class="text-muted" href="/User/ForGotPassWord">Forgot password?</a>
</div>
</div>
}
<form id="form_login">
<div class="user register here">
<div class="d-flex align-items-center justify-content-center pb-4">
<p class="user_register_here">
<b>Don't have an account?</b> @using (Html.BeginForm(FormMethod.Get))
{
<input type="submit" value="Register Here" asp-action="Index" asp-controller="User" />
}
</p>
</div>
</div>
</form>

</div>
</div>
</div>
</div>
</div>
</div>
</div>
</section>
```

ADMIN CONTROLLER:

```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using System.Data.SqlClient;
```

```
using TMS.Models;
namespace TMS.Controllers
{
public class AdminController : Controller
{
// GET: AdminController
public ActionResult Index()
{
string con = "Server=localhost\\SQLEXPRESS;Database=TMS;Trusted_Connection=True;";
RequestDetails requestDetails = new RequestDetails();
List<RequestDetails> lstRequestDetails = new List<RequestDetails>();
using (SqlConnection myConnection = new SqlConnection(con))
{
string oString = "Select * from RequestDetails";
SqlCommand oCmd = new SqlCommand(oString, myConnection);
myConnection.Open();
using (SqlDataReader oReader = oCmd.ExecuteReader())
{
while (oReader.Read())
{
lstRequestDetails.Add(new RequestDetails()
{
UserId = oReader["UserId"].ToString(),
RequestId = oReader["RequestId"].ToString(),
RequestDescription = oReader["RequestDescription"].ToString(),
RequestType = oReader["RequestType"].ToString(),
RequestStatus = oReader["RequestStatus"].ToString(),
AssignedTo = oReader["AssignedTo"].ToString(),
Comments = oReader["Comments"].ToString()
});
}
myConnection.Close();
}
}
return View(lstRequestDetails);
}
public ActionResult SearchRequestStatus(IFormCollection collection)
{
string con = "Server=localhost\\SQLEXPRESS;Database=TMS;Trusted_Connection=True;";
RequestDetails requestDetails = new RequestDetails();
List<RequestDetails> lstRequestDetails = new List<RequestDetails>();
using (SqlConnection myConnection = new SqlConnection(con))
{
string oString = "Select * from RequestDetails where RequestType=@RequestType";
```

```
SqlCommand oCmd = new SqlCommand(oString, myConnection);
oCmd.Parameters.AddWithValue("@RequestType", collection["RequestType"].ToString());
myConnection.Open();
using (SqlDataReader oReader = oCmd.ExecuteReader())
{
while (oReader.Read())
{
lstRequestDetails.Add(new RequestDetails()
{
UserId = oReader["UserId"].ToString(),
RequestId = oReader["RequestId"].ToString(),
RequestDescription = oReader["RequestDescription"].ToString(),
RequestType = oReader["RequestType"].ToString(),
RequestStatus = oReader["RequestStatus"].ToString(),
AssignedTo = oReader["AssignedTo"].ToString(),
Comments = oReader["Comments"].ToString()
});
}
myConnection.Close();
}
}
return View(lstRequestDetails);
} // GET: AdminController/Details/5
public ActionResult Details(int id)
{
return View();
}
// GET: AdminController/Create
public ActionResult Create()
{
return View();
}
// POST: AdminController/Create
[HttpPost]
[ValidateAntiForgeryToken]
```

STUDENT CONTROLLER:

```
using Microsoft.AspNetCore.Mvc;
using System.Data.SqlClient;
using System.Drawing;
using TMS.Models;

namespace TMS.Controllers
{
publicclassStudentController : Controller
{
```

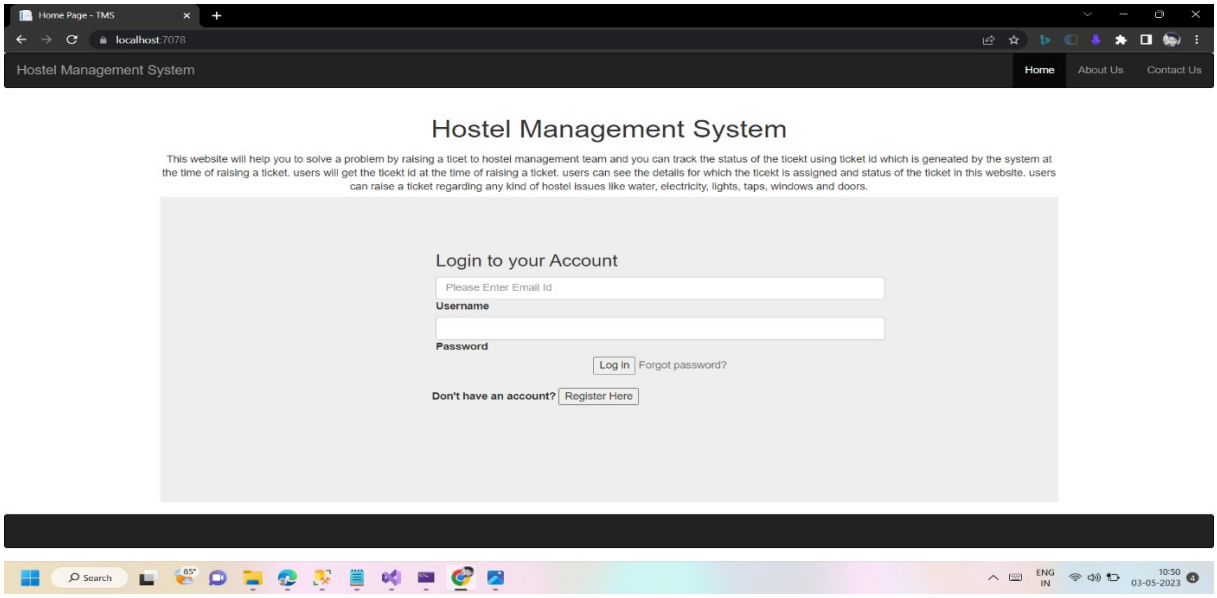
```
// GET: StudentController
public ActionResult Index()
{

string con = "Server=localhost\\
SQLEXPRESS01;Database=TMS;Trusted_Connection=True;";
RequestDetails requestDetails = new RequestDetails();
List<RequestDetails> lstRequestDetails = new List<RequestDetails>();

using (SqlConnection myConnection = new SqlConnection(con))
{
string oString = "Select * from RequestDetails where UserId=@UserId";
SqlCommand oCmd = new SqlCommand(oString, myConnection);
oCmd.Parameters.AddWithValue("@UserId", HttpContext.Session.GetString("userId"));
myConnection.Open();
```

EMPLOYEE CONTROLLER:

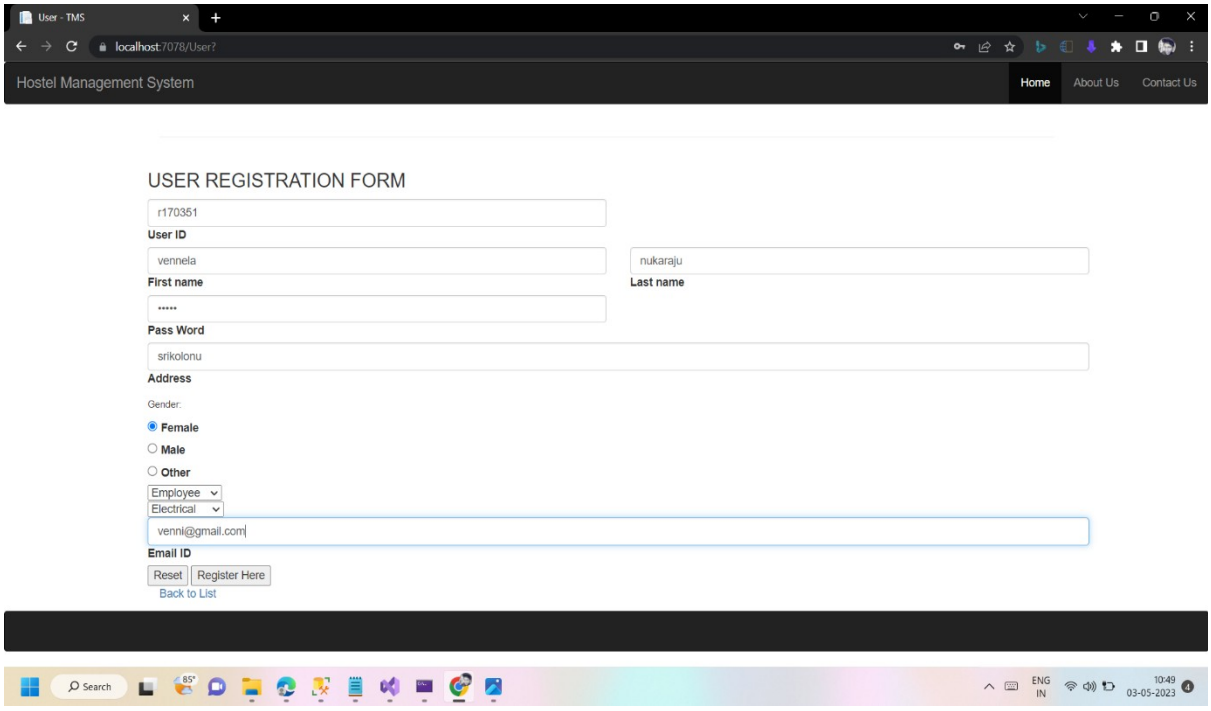
```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.VisualBasic;
using System.Collections.Generic;
using System.Data.SqlClient;
using TMS.Models;

namespace TMS.Controllers
{
publicclassEmployeeController : Controller
{
// GET: EmployeeController
public ActionResult Index()
{
string con = "Server=localhost\\
SQLEXPRESS01;Database=TMS;Trusted_Connection=True;";
RequestDetails requestDetails = new RequestDetails();
List<RequestDetails> lstRequestDetails = new List<RequestDetails>();

using (SqlConnection myConnection = new SqlConnection(con))
```

RESULTS

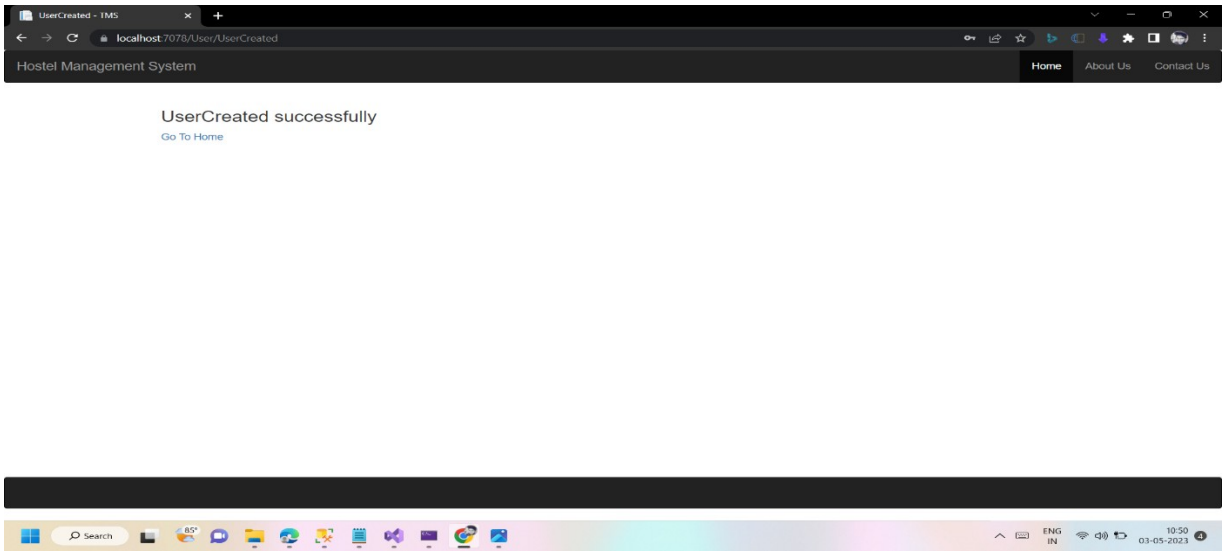
HOME PAGE:



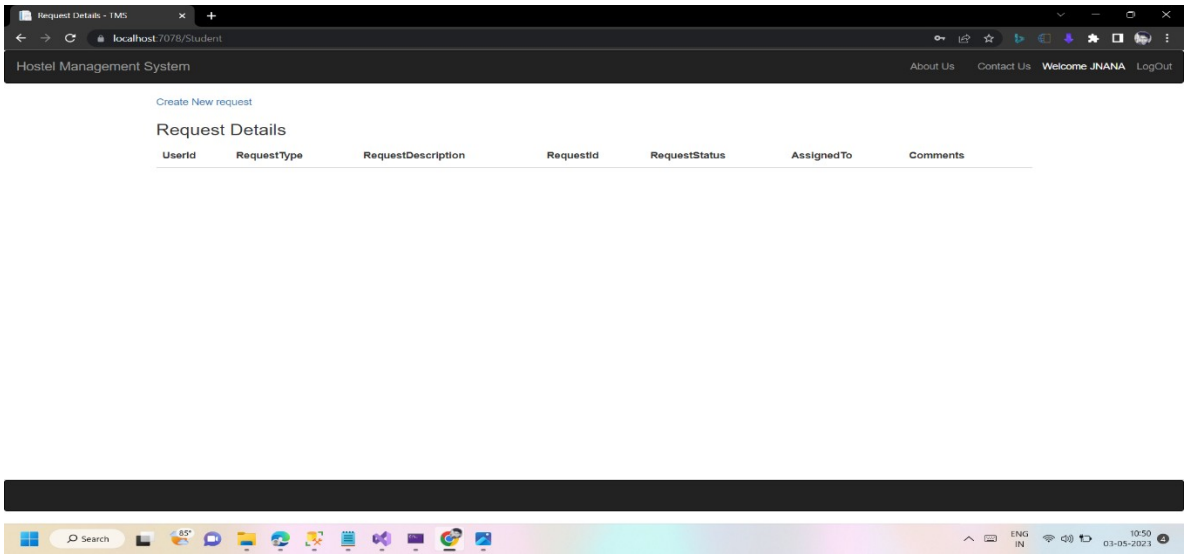
USER REGISTRATION:



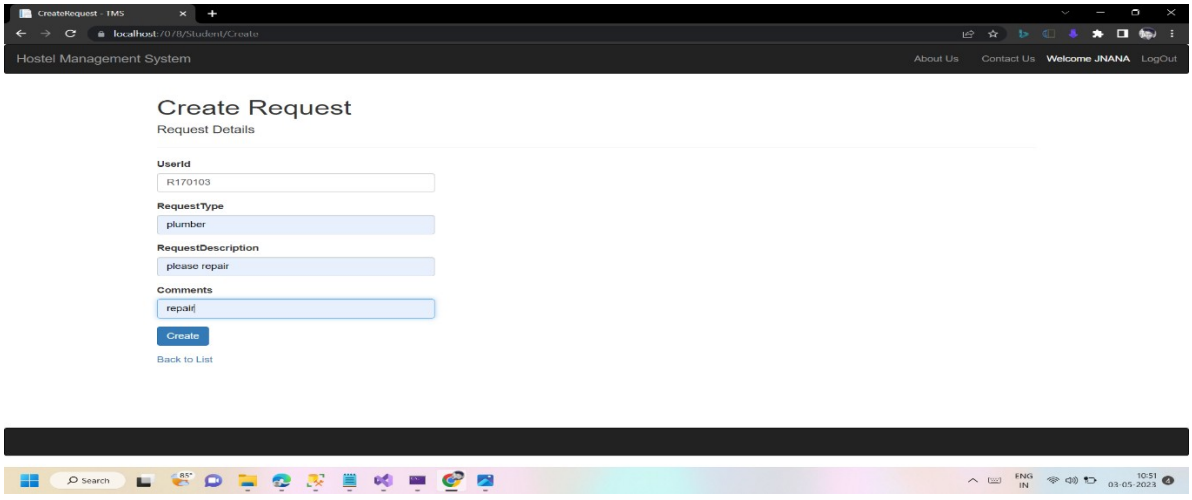
USER CREATE SUCCESSFULLY:



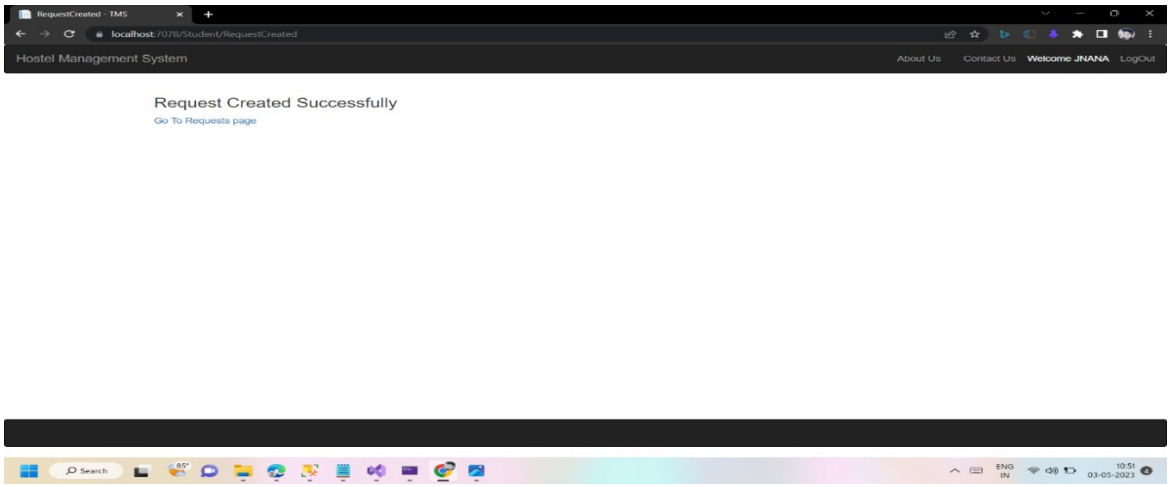
USER LOGIN PAGE:



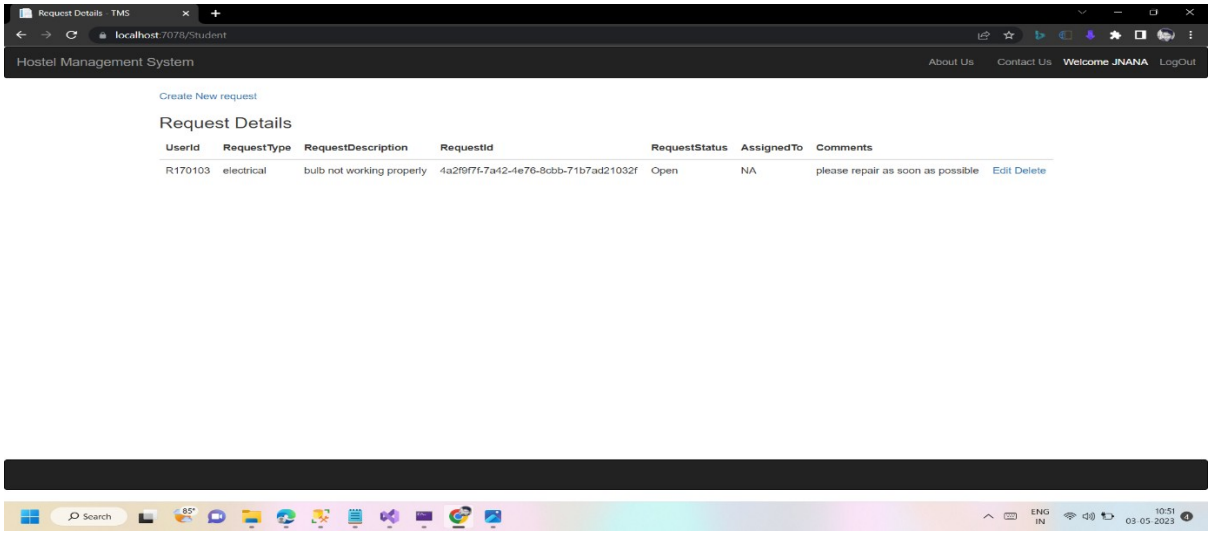
CREATE REQUEST



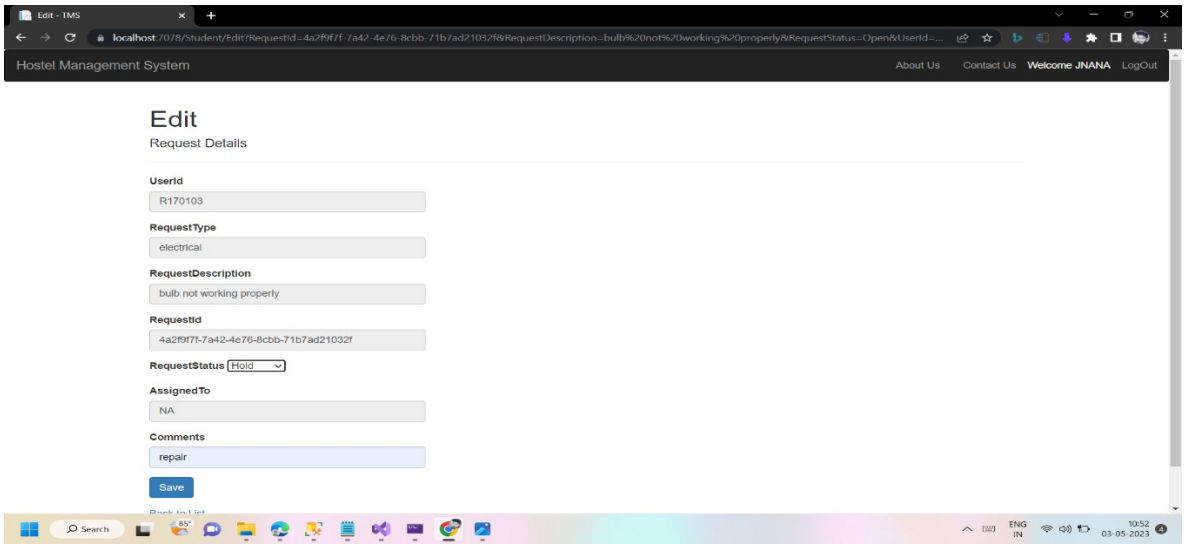
REQUEST CREATE SUCCESSFULLY:



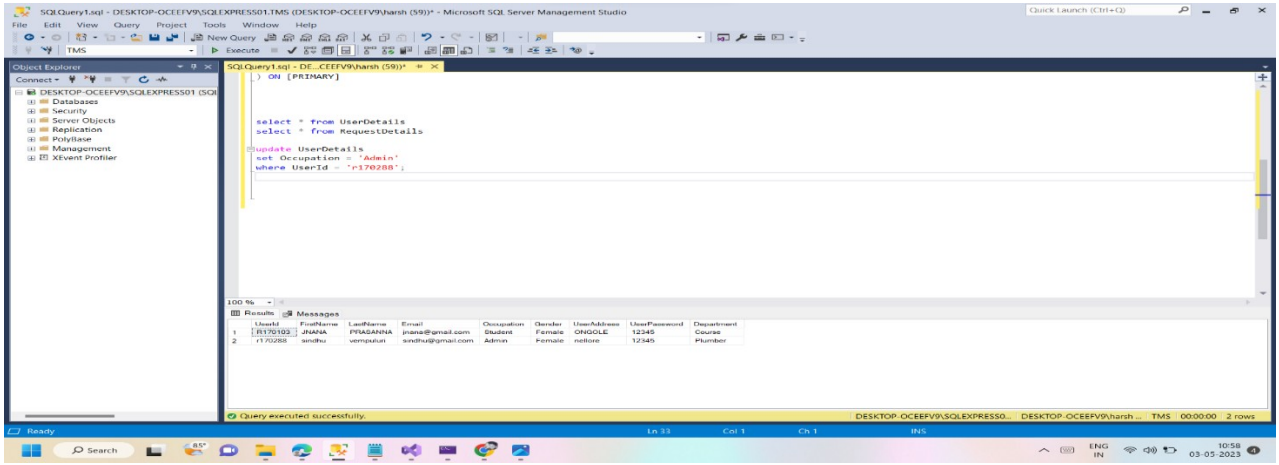
AFTER CREATE REQUESTS:



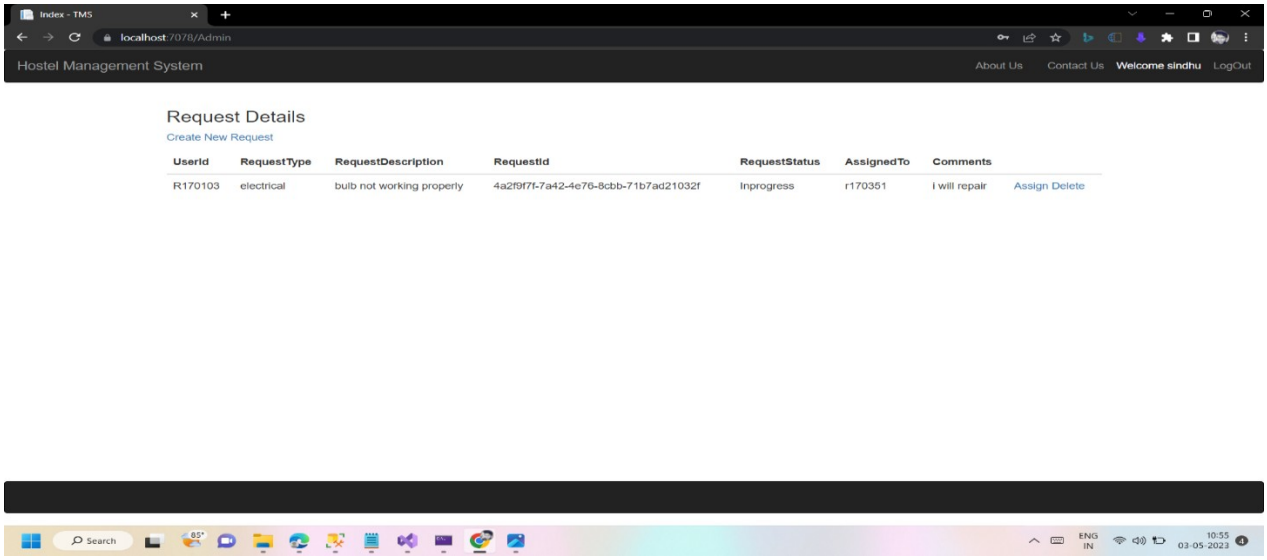
USER EDIT REQUEST:



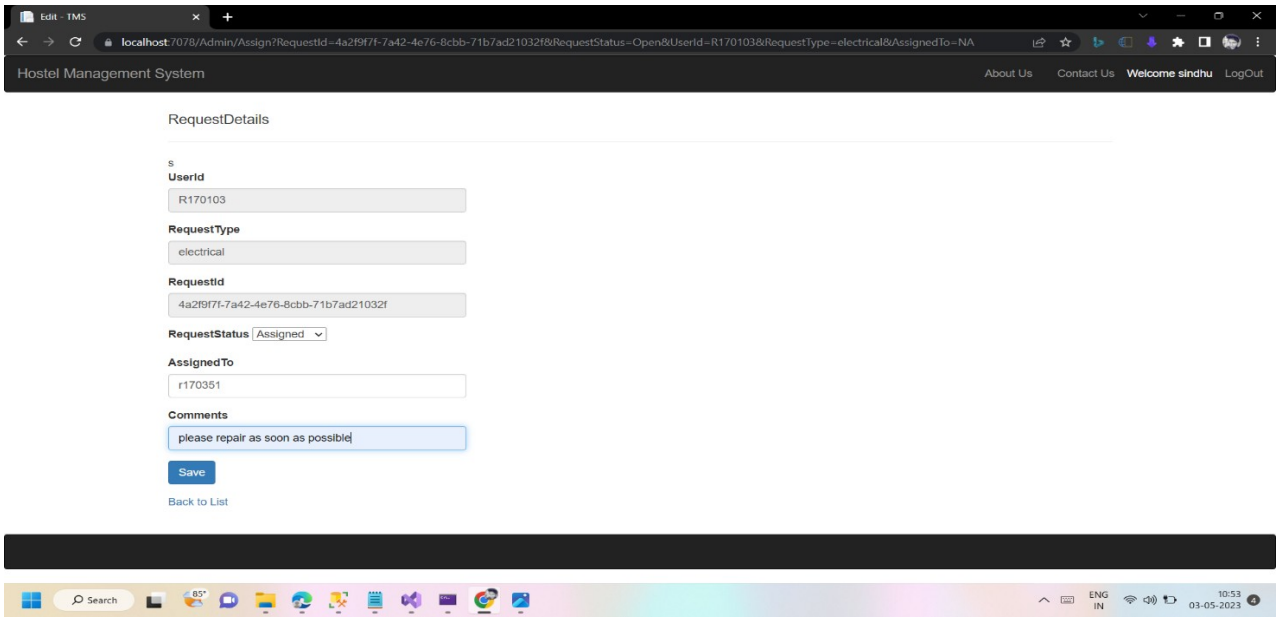
MAKING AN EMPLOYEE AS A ADMIN:



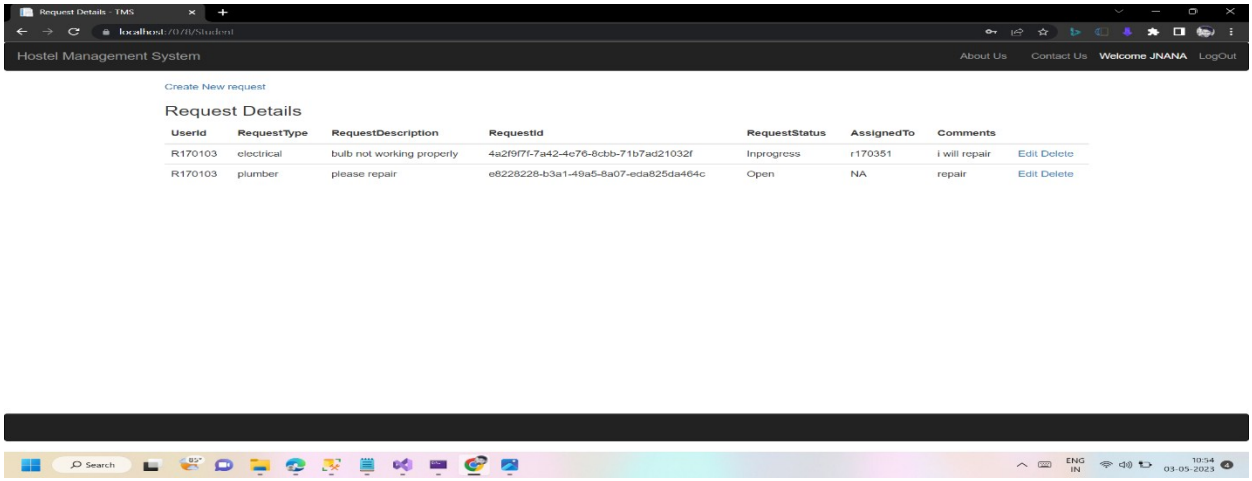
AFTER ADMIN LOGINS:



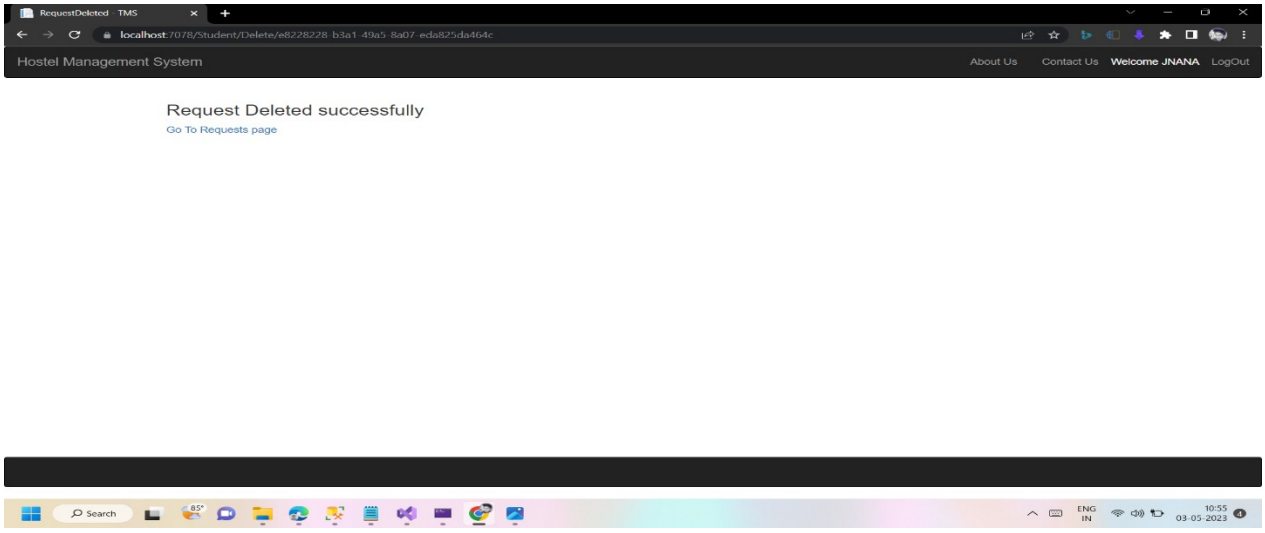
ASSIGN WORK TO EMPLOYEE:



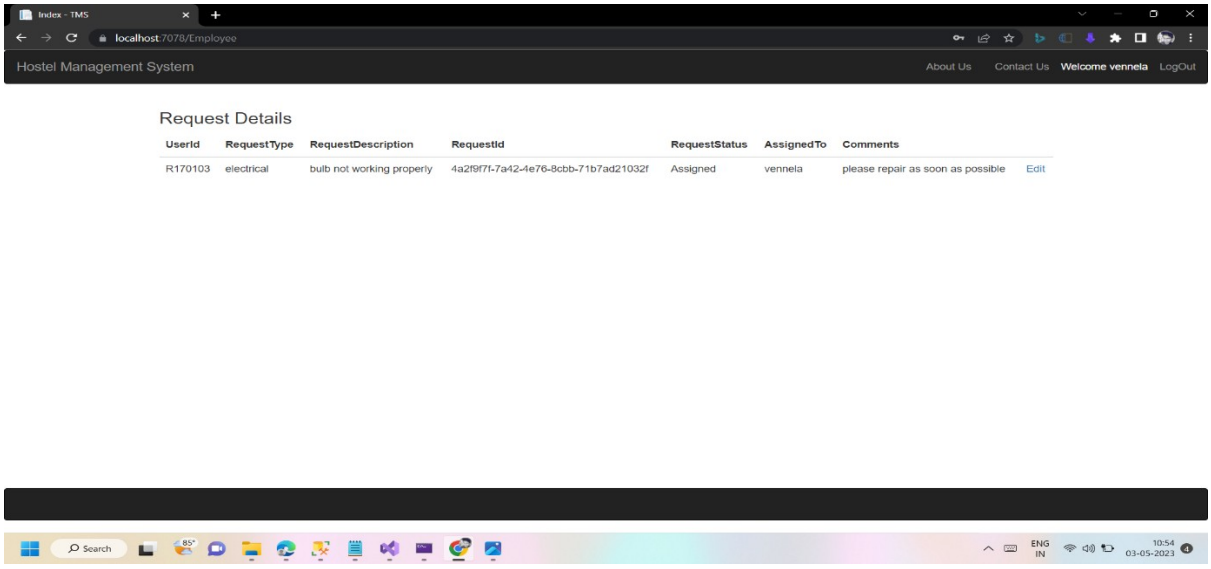
ADMIN OR USER OR EMPLOYEE CAN DELETE THE ISSUE:



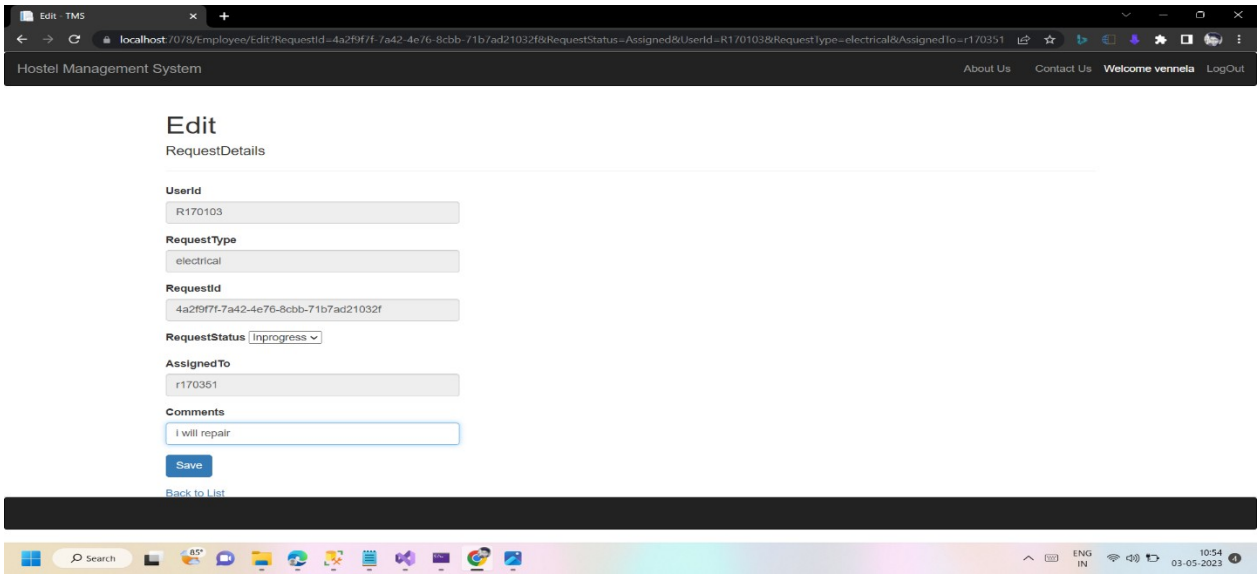
AFTER DELETING ISSUE:



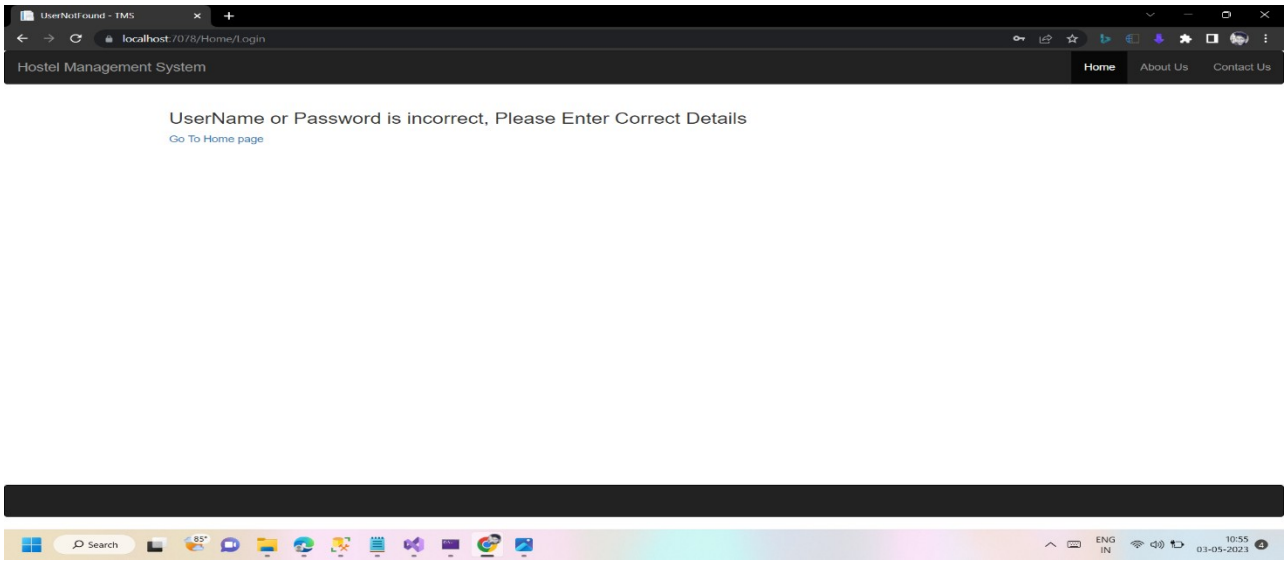
EMPLOYEE LOGINS AND SEE THE ISSUE:



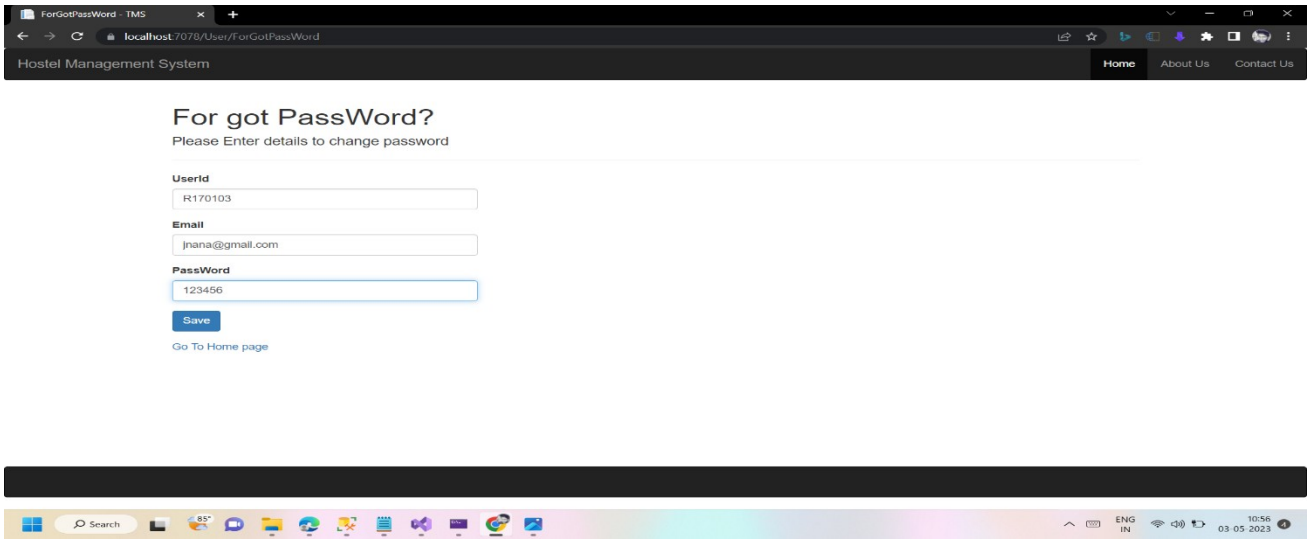
EMPLOYEE EDIT THE ISSUE:



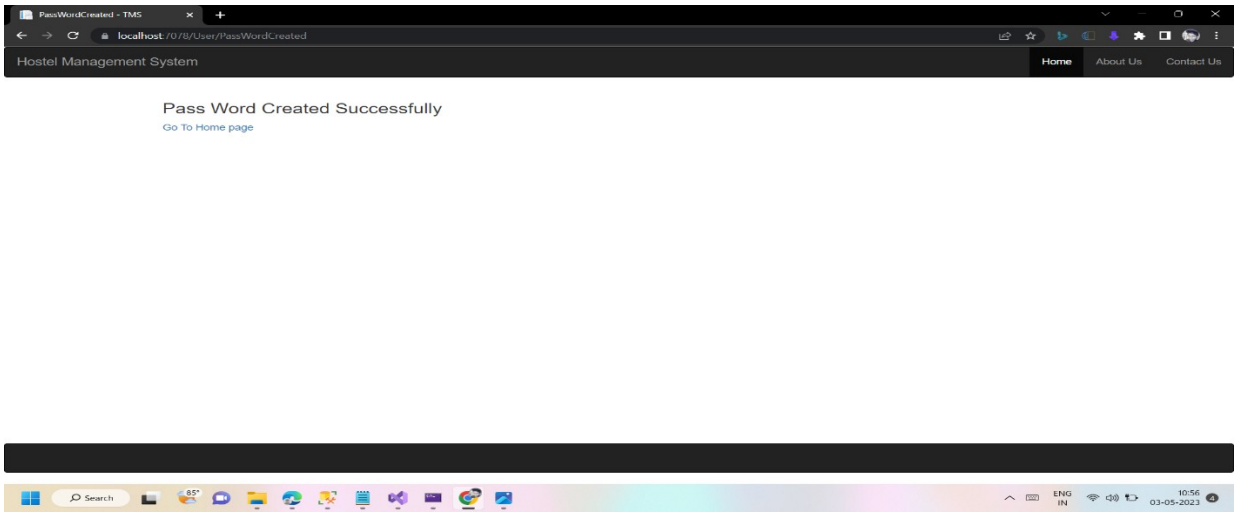
FORGOT PASSWORD IN HOME PAGE:



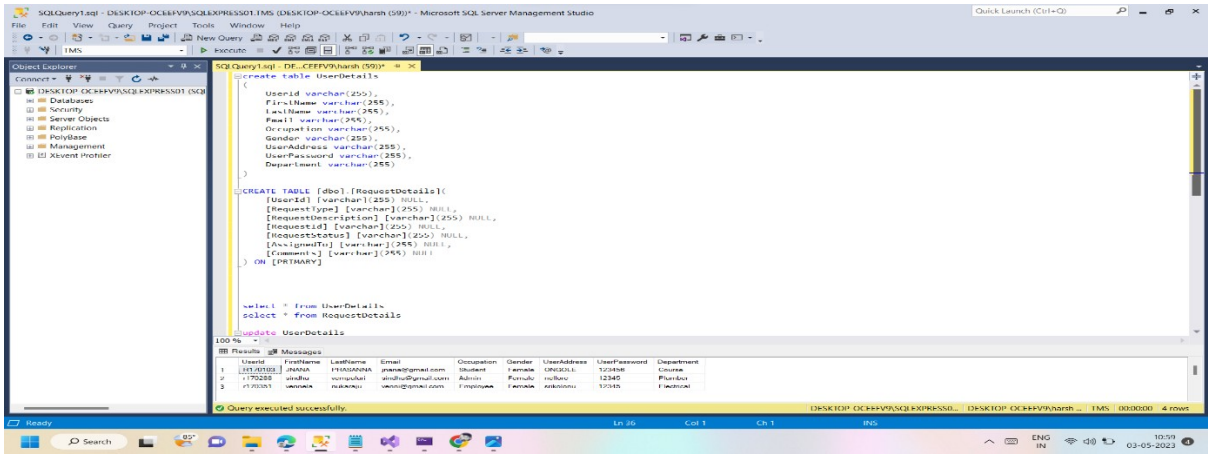
RECOVERING PASSWORD:



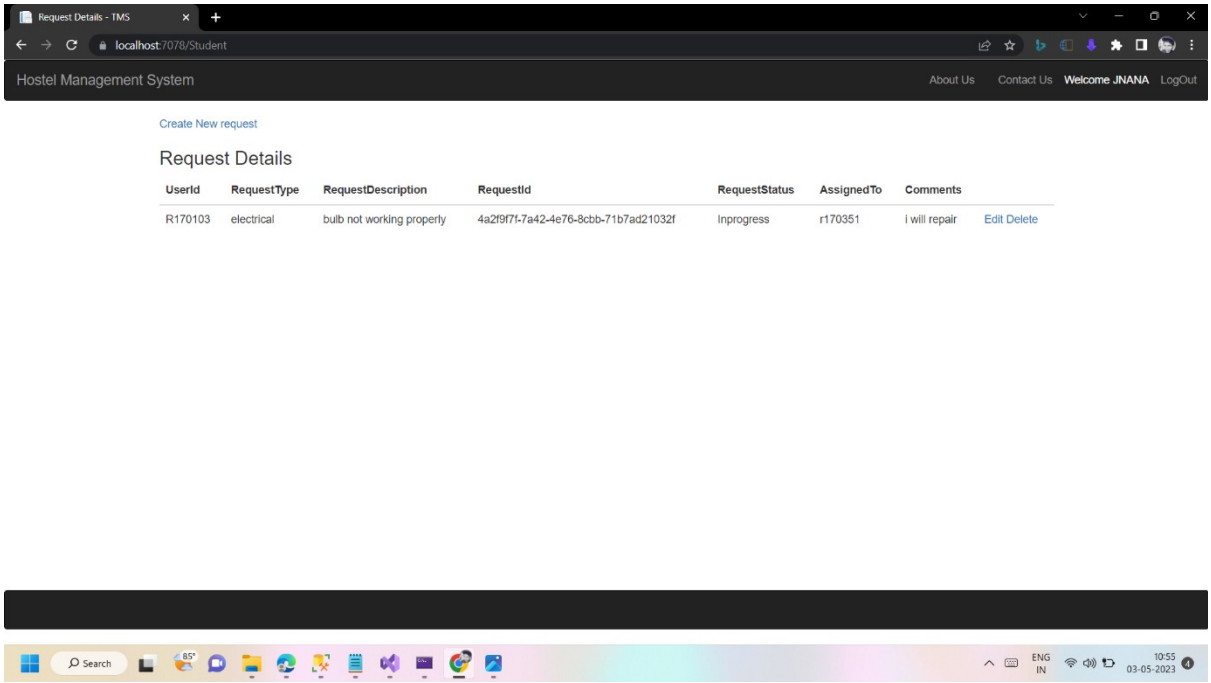
PASSWORD CHANGED SUCCESSFULLY:



CREATE DATABSE AND TABLES



STATUS IS UPDATED I N ADMIN AND USER BY EMPLOYEE:



FUTURE SCOPE

The future implementation of this project is adding css (cascading style sheet) and deploying in the cloud to manage it .Based on this everyone can use this website by deploying in cloud. After adding css (cascading style sheet) the page will look attractive to use.

CONCLUSION

The Hostel Management System helps your organization improve workforce productivity and boost overall well-being by assigning work to the particular employee and user can easily raise a issue to solve. The Hostel Management System is developed in order to computerize the activities which take more time, if done manually.

This study takes similar route ,but with an improved and novel method and by using this portal the user can save their productive time .This website provides more accuracy to solve the problems within the time. The Hostel Management System is used to the student raise the problems by using the website and admin can assign or delete the work raised by student by using employee id and employee can see the work assigned by admin and edit the status of employee.

References:

https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-7.0#model-binding
https://youtu.be/4IgC2Q5-yDE -kudvenkat channel
https://www.w3schools.com/sql
https://geekforgeeks.com
https://javatpoint.com