# INFORMATION COLLECTING WEBSITE USING FLASK

### A Mini Project Report Submitted by ARUMUGAM A K 21CSR009

**ARUNKUMAR K 21CSR010**

**DHANAVANDHAN M**

**21CSR029**

***in partial fulfilment of the requirements for the award of the degree***

***of***

# BACHELOR OF ENGINEERING IN

**COMPUTER SCIENCE AND ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



# KONGU ENGINEERING COLLEGE

**(Autonomous)**

**PERUNDURAI ERODE-638060**

# MAY2023

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING KONGU ENGINEERING COLLEGE

**(Autonomous) PERUNDURAI ERODE-638060**

**MAY2023**

## BONAFIDE CERTIFICATE

This is to certify that the Project Report entitled **INFORMATION COLLECTING WEBSITE USING FLASK** is the bonafide record of project work done by **ARUMUGAM AK (Register no:21CSR009), ARUNKUMAR K (Register no:21CSR010),**

**DHANAVAANDHAN M (Register no:21CSR029)**in partial fulfillment of the requirements for the award of the Degree of Bachelor of Engineering in **Computer Science and Engineering** of Anna University, Chennai during the year 2022-2023.

**COURSE INSTRUCTOR HEAD OF THE DEPARTMENT**

**Date:**

Submitted for the viva voce examination held on

## ABSTRACT

The information collecting system is a comprehensive solution designed to streamline and automate the process of gathering, organizing, and managing data. It provides a centralized platform for collecting and storing various types of information, ranging from customer data to research findings, survey responses, and more. This system eliminates the need for manual data collection methods, such as paper forms or spreadsheets, and offers advanced functionalities to enhance data accuracy, accessibility, and security.

The system enables users to create customizable data collection forms, tailored to their specific needs. These forms can include a wide range of data fields, such as text inputs, checkboxes, dropdown menus, and file uploads. Users can distribute these forms through various channels, including email invitations, web links, or embedded forms on websites, allowing them to collect data from a diverse set of sources.

The information collecting system offers features for data validation, ensuring that the collected information meets specified criteria or formats. It supports real-time data entry, allowing users to instantly capture and store data as it is submitted. Additionally, the system provides data storage and organization capabilities, allowing users to categorize and label data entries, making it easier to retrieve and analyze information later.

Data security is a critical aspect of the information collecting system. It implements robust security measures to protect sensitive data, including encryption, user authentication, and access control mechanisms. Users can define permissions and roles to ensure that only authorized individuals can access and manipulate the collected information.

The system also offers powerful reporting and analytics functionalities, allowing users to generate meaningful insights from the collected data. It provides visualization tools, charts, and graphs to present data in a clear and understandable manner, facilitating data- driven decision-making processes.

Overall, the information collecting system revolutionizes the way organizations gather, manage, and leverage data. It improves efficiency, accuracy, and accessibility, enabling users to make informed decisions based on reliable and well-organized information.

# CHAPTER 1

**FLASK**

## INTRODUCTION

Flask is a popular web framework written in Python, designed to be simple and lightweight. It follows the principle of "micro" framework, focusing on core functionality and flexibility. With its intuitive syntax and minimalistic design, Flask allows developers to quickly build web applications. Flask provides essential features such as routing, request handling, and template rendering. It supports the use of extensions to add extra functionality to the framework. Flask promotes modular and scalable development, making it suitable for projects of any size. It has a vibrant and active community that contributes to its ecosystem with various plugins and libraries.

## LIFECYCLE OF FLASK APPLICATION

The lifecycle of a Flask application consists of various stages and processes that occur from the initialization to the termination of the application. Here is an overview of the typical lifecycle of a Flask application:

Application Creation: An instance of the Flask class is created to represent the application.

Configuration: Configuration settings such as database connection details, secret keys, and other options are defined for the application.

Routes and Views: Routes are defined to map URLs to specific functions (views) that handle requests and return responses.

Request Handling: When a request is received, Flask matches the URL to the appropriate route and invokes the associated view function.

Pre-Request Processing: Flask performs any necessary pre-processing tasks before executing the view function, such as authentication and validation.

View Function Execution: The view function is executed, which processes the request, interacts with the necessary data sources, and generates a response.

Response Handling: The response generated by the view function is returned to the client.

Post-Request Processing: Flask performs any necessary post-processing tasks, such as logging or cleanup operations.

Application Termination: The Flask application is terminated, typically when the server is stopped or the script execution ends.

It's important to note that Flask applications can be run in different deployment environments, such as development, testing, and production, and each environment may have its own specific considerations and configurations.

## FLASK PACKAGES AND COMPONENTS

Flask is a lightweight web framework for Python that provides a flexible and modular approach to building web applications. While Flask itself is minimalistic, it offers various packages and components that extend its functionality and provide additional features. Some of the popular Flask packages and components are:

Flask-WTF: Provides integration with WTForms, a flexible form handling library, allowing easy creation and validation of forms in Flask applications.

Flask-SQLAlchemy: Offers integration with SQLAlchemy, a powerful Object-Relational Mapping (ORM) library, enabling developers to work with databases and perform database operations seamlessly.

Flask-Login: Provides user authentication and session management functionality, allowing developers to handle user login, logout, and access control easily.

Flask-Mail: Offers email sending capabilities, allowing developers to send emails from Flask applications using SMTP or other email providers.

Flask-RESTful: Simplifies the creation of RESTful APIs by providing a set of tools and decorators for defining API resources, handling HTTP methods, and serialization.

Flask-Cache: Provides caching capabilities to improve the performance of Flask applications by storing and retrieving data from cache instead of performing time- consuming operations.

Flask-Uploads: Offers file uploading and management functionalities, allowing developers to handle file uploads, storage, and retrieval in Flask applications.

Flask-Admin: Provides an administrative interface for Flask applications, allowing developers to easily create and manage an admin panel to perform CRUD operations on models.

Flask-Bcrypt: Offers password hashing and verification capabilities using the bcrypt hashing algorithm, ensuring secure storage of user passwords in the database.

Flask-RESTPlus: Extends Flask-RESTful with additional features like automatic API documentation generation using Swagger and support for namespaces and route grouping.

These are just a few examples of the many Flask packages and components available. Flask's modular design and extensive ecosystem make it easy to integrate and extend functionalities according to the specific requirements of your web application.

# CHAPTER 2 MODULES DESCRIPTION

## INTRODCUTION

The Information Collecting System is a web-based application designed to efficiently gather and manage information. It provides a user-friendly interface for collecting, storing, and manipulating data. The system consists of several modules, each serving a specific purpose.

### Login Page:

The Login page provides authentication functionality to ensure that only authorized users can access and manage the system. It typically requires a username and password combination for authentication purposes. Upon successful login, users gain access to the system's features and functionalities.

### Home Page:

The Home page serves as the entry point of the system. It prompts the user to enter a key or search term to retrieve specific data from the system. It provides a search functionality to fetch and display relevant information.

### Add Page:

The Add page allows users to input new data into the system. It presents a form or set of fields where users can enter the required information. Upon submission, the data is stored in the system's database for future reference.

### Delete Page:

The Delete page enables users to remove specific data entries from the system. It typically provides a search feature to locate the desired data and a confirmation step before deletion. Once confirmed, the data is permanently deleted from the system.

### Modify Page:

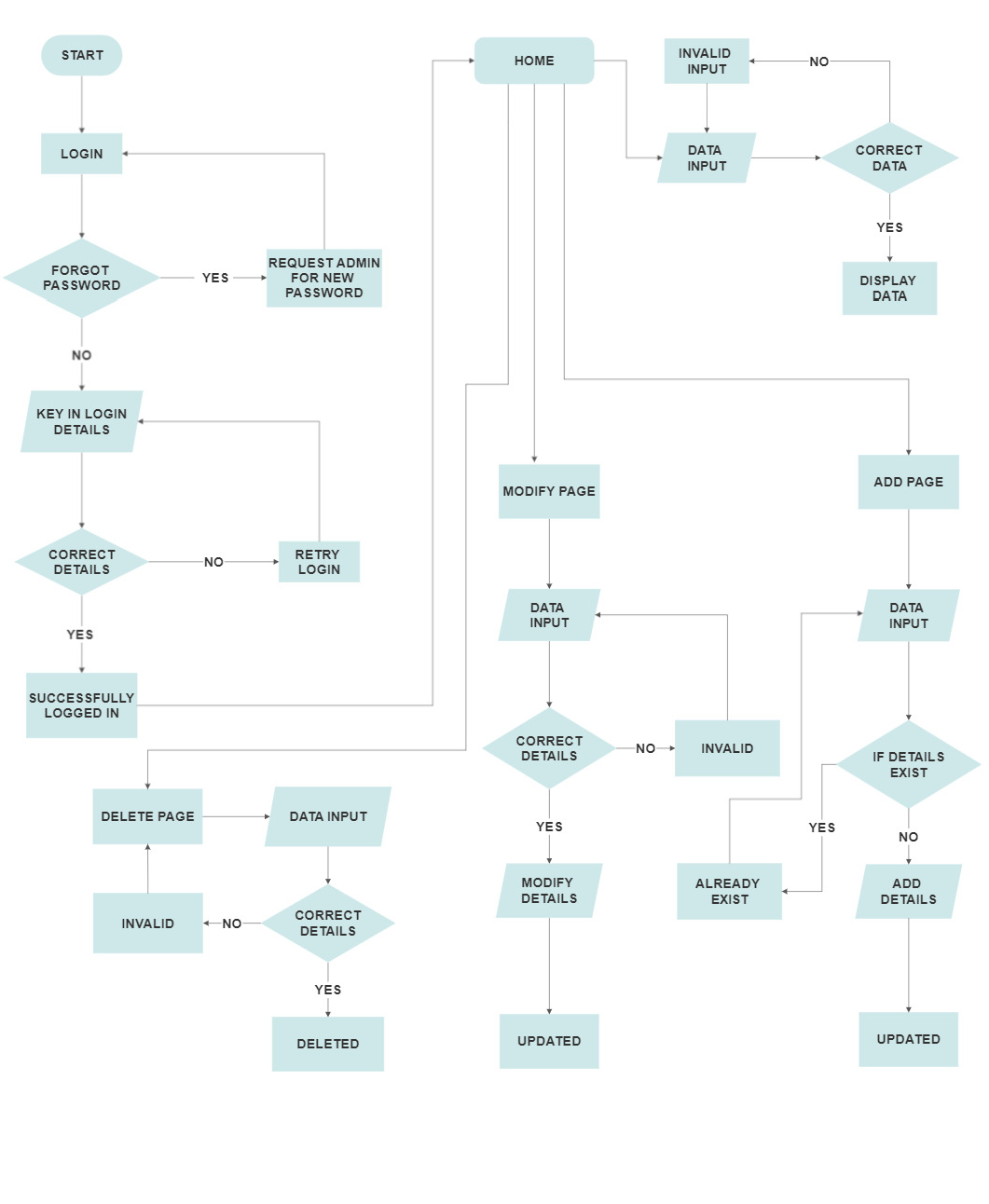
The Modify page allows users to update existing data entries. It presents the current values of the selected data and provides editable fields to modify the information. After making the necessary changes, users can save the updated data, which gets reflected in the system.

## DATABASE CREATION

MYSQL- MySQL("My S-Q-L", officially, but also called "My Sequel") is

(as of July 2013) the world's second most widely used open-source relational database management system (RDBMS). It is named after co-founder Michael Widenius’s daughter, My. The SQL phrase stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.MySQL is a popular choice of database for use in web applications and is a central component of the widely used LAMP open-source web application software stack (and other 'AMP' stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open source projects that require a full-featured database management system often use MySQL.

**WORK FLOW DIAGRAM**



**CHAPTER 3**

**CONCLUSION**

The information collecting website has proven to be a valuable tool for efficient data collection and management. Throughout the development process, several key aspects have emerged, highlighting the significance and benefits of such a system.

Firstly, the website provides a user-friendly interface, allowing users to easily enter and retrieve data. The intuitive design and clear navigation make it accessible to users of varying technical backgrounds, ensuring a seamless experience.

Secondly, the incorporation of the login page enhances data security and privacy. By implementing user authentication, the system ensures that only authorized individuals have access to sensitive information, preventing unauthorized usage or data breaches.

Furthermore, the add, delete, and modify pages offer extensive functionality for data manipulation. Users can effortlessly add new records, remove outdated entries, and modify existing data, enabling efficient updates and maintenance of the database.

Additionally, the key-based search feature implemented on the home page streamlines data retrieval. Users can enter specific keys or search terms to retrieve relevant information quickly, improving overall efficiency and productivity.

The database creation process has been carefully executed, ensuring proper organization, relationships, and constraints. This ensures data integrity and consistency, preventing duplication or inaccuracies within the system.

In conclusion, the information collecting website serves as a robust and reliable solution for efficient data collection and management. Its user-friendly interface, enhanced security, extensive functionality, and well-designed database structure contribute to its effectiveness. By implementing this system, organizations can streamline their data collection processes, improve data accessibility, and enhance overall productivity.

# APPENDIX

1 .CODE WITH SCREENSHOTS:

### Login Page:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document

</title>

</head>

<body style="background: linear-gradient(#e66465, #9198e5);">

<div style="display: flex; justify-content: center; align-items: center; height: 100vh;">

<form class="form" style="display: flex; flex-direction: column; gap: 10px; max- width: 300px; padding: 2em;backdrop-filter: blur(5px); box-shadow: 0 25px 45px rgba(0,0,0,0.1);border: 1px solid rgba(255,255,255,0.5);border-right: 1px solid

rgba(255,255,255,0.2);border-bottom: 1px solid rgba(255,255,255,0.2);background: rgba(255,255,255,0.1);border-radius: 10px; transition: .4s ease-in-out;" method="POST" action="/login">

<p id="heading" style="font-family:Arial, Helvetica, sans-serif; text-align: center; color: #fff; font-size: 1.5em;">Login</p>

<div class="field" style="display: flex; align-items: center; gap: 0.5em;; padding: 0.6em; border: none; outline: none; color: white;backdrop-filter: blur(5px);box-shadow: 0 25px 45px rgba(0,0,0,0.1);border: 1px solid rgba(255,255,255,0.5);border-right: 1px solid rgba(255,255,255,0.2);border-bottom: 1px solid rgba(255,255,255,0.2);background: rgba(255,255,255,0.1);border-radius:

10px;">

<svg class="input-icon" xmlns="<http://www.w3.org/2000/svg>" width="16" height="16" fill="currentColor" viewBox="0 0 16 16" style="height: 1.3em; width: 1.3em; fill: white;">

<path d="M13.106 7.222c0-2.967-2.249-5.032-5.482-5.032-3.35 0-5.646 2.318-5.646 5.702 0 3.493 2.235 5.708 5.762 5.708.862 0 1.689-.123

2.304-.335v-.862c-.43.199-1.354.328-2.29.328-2.926 0-4.813-1.88-4.813-

4.798 0-2.844 1.921-4.881 4.594-4.881 2.735 0 4.608 1.688 4.608 4.156 0

1.682-.554 2.769-1.416 2.769-.492 0-.772-.28-.772-

.76V5.206H8.923v.834h-.11c-.266-.595-.881-.964-1.6-.964-1.4 0-2.378

1.162-2.378 2.823 0 1.737.957 2.906 2.379 2.906.8 0 1.415-.39 1.709-

1.087h.11c.081.67.703 1.148 1.503 1.148 1.572 0 2.57-1.415 2.57-

3.643zm-7.177.704c0-1.197.54-1.907 1.456-1.907.93 0 1.524.738 1.524

1.907S8.308 9.84 7.371 9.84c-.895 0-1.442-.725-1.442-1.914z">

</path>

</svg>

<input autocomplete="off" placeholder="Username" class="input-field" type="text" name="username" style="background: none; border: none; outline: none; width: 100%; color: #d3d3d3;">

</div>

<div class="field" style="display: flex; align-items: center; gap: 0.5em; padding: 0.6em; border: none; outline: none; color: white; backdrop-filter: blur(5px);box-shadow: 0 25px 45px rgba(0,0,0,0.1);border: 1px solid rgba(255,255,255,0.5);border-right: 1px solid rgba(255,255,255,0.2);border-bottom: 1px solid rgba(255,255,255,0.2);background: rgba(255,255,255,0.1);border-radius: 10px; margin-top: 10px;">

<svg class="input-icon" xmlns="<http://www.w3.org/2000/svg>" width="16" height="16" fill="currentColor" viewBox="0 0 16 16" style="height: 1.3em; width: 1.3em; fill: white;">

<path d="M8 1a2 2 0 0 1 2 2v4H6V3a2 2 0 0 1 2-2zm3 6V3a3 3 0 0 0-6

0v4a2 2 0 0 0-2 2v5a2 2 0 0 0 2 2h6a2 2 0 0 0 2-2V9a2 2 0 0 0-2-

2z"></path>

</svg>

<input placeholder="Password" class="input-field" type="password" name="password" style="background: none; border: none; outline: none; width: 100%; color: #d3d3d3;">

</div>

<div class="btn" style="display: flex; justify-content: center; gap: 0.5em;">

<button class="button1" style="width: 100%;padding: 0.5em; border: none; outline: none; transition: .4s ease-in-out; backdrop-filter: blur(5px);box- shadow: 0 25px 45px rgba(0,0,0,0.1);border: 1px solid rgba(255,255,255,0.5);border-right: 1px solid rgba(255,255,255,0.2);border-bottom: 1px solid rgba(255,255,255,0.2);background: rgba(255,255,255,0.1);border-radius: 35px; color: white;margin-top: 10px;" type="submit">Login

</button>

</div>

<button class="button3" style="margin-bottom: 1em; padding: 0.5em; border: none; outline: none; transition: .4s ease-in-out; backdrop-filter: blur(5px);box-shadow: 0 25px 45px rgba(0,0,0,0.1);border: 1px solid rgba(255,255,255,0.5);border-right: 1px solid rgba(255,255,255,0.2);border-bottom: 1px solid rgba(255,255,255,0.2);background: rgba(255,255,255,0.1);border-radius: 35px; color: white;margin-top: 10px;margin-bottom: 40px;">Forgot

Password

</button>

</form>

</div>

</body>

</html>

### Home Page:

<!DOCTYPE html>

<html lang="en" dir="ltr">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Student Information</title>

<link rel="stylesheet" href="fpage.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font- awesome/5.15.1/css/all.min.css">

<script type = "text/javascript"> function moveone()

{

document.getElementById("move").innerHTML ="Robust SIS";

}

</script>

<style>

\*{

margin: 0;

padding: 0;

box-sizing: border-box;

}

.hero{

width: 100%; height: 100vh;

background-image: linear-gradient(rgb(19, 51, 3),)rgb(3, 51, 18); position: relative;

padding: 0 5%; display: flex;

align-items: center; justify-content: center;

}

.content{

text-align: center;

}

.content h1{

font-size: 130px; color: #e9e4e4; font-weight: 359; transition: 0.5s;

}

.content h1:hover{

-webkit-text-stroke: 2px #f5e6e7; color: transparent;

}

.back-video{ position: absolute; right: 0;

bottom: 0;

z-index: -1;

}

.content a:hover

{

background: transparent; color: #000;

}

header{

z-index: 999; position: fixed; top: 0;

left: 0;

width: 100%; display: flex;

justify-content: space-between; align-items: center;

padding: 20px 100px; transition: 0.6s;

}

header.sticky{ background: #ad1a1a; padding: 15px 100px;

}

header .brand{ color: #ede1e1; font-size: 30px; font-weight: 700;

text-transform: uppercase; text-decoration: none; letter-spacing: 2px;

}

header .menu{ position: relative; display: flex;

justify-content: center; align-items: center;

}

header .menu a{ color: #fff;

font-size: 16px; font-weight: 500;

text-decoration: none; margin: 0 30px;

padding: 0 10px; border-radius: 20px; transition: 0.3s;

transition-property: color, background;

}

header .menu a:hover{ color: #000;

background: #fff;

}

header .btn{ color: #ffffff; font-size: 25px; cursor: pointer; display: none;

}

@media (max-width: 1060px){ header .btn{

display: block;

}

header .menu{ position: fixed; background: #000;

flex-direction: column; min-width: 400px; height: 100vh;

top: 0;

right: -100%; padding: 80px 50px; transition: 0.5s;

transition-property: right;

}

header .menu.active{ right: 0;

}

header .menu .close-btn{ position: absolute;

top: 0;

left: 0; margin: 25px;

}

header .menu a{ display: block; font-size: 20px; margin: 20px; padding: 0 15px;

}

}

@media (max-width: 630px){

.section-main h1{ font-size: 50px; line-height: 60px;

}

}

.section-two{

min-height: 30vh; background: #a3a6a900; padding: 20px 40px;

}

.section-two h2{ color: #971616; font-size: 30px; font-weight: 700;

text-transform: uppercase; margin: 30px 0;

padding-bottom: 0px;

}

.section-two p{ color: #000; font-size: 18px; margin: 30px 0;

}

.section-tw{

min-height: 30vh; background: #a3a6a900; padding: 20px 40px;

}

.section-tw p{ color: #000; font-size: 18px; margin: 10px 0;

padding-top: 0px; padding-bottom: 0px; text-align: left;

}

.footer{

grid-template-columns: repeat(auto-fit, minmax(220px, 1fr)); grid-gap: 5px;

margin: 50px 0 5px; text-align: center;

}

.footer img{ width: 3%;

border-radius: 5px;

}

.footer a{

text-decoration: none; color: #999;

font-size: 22px; margin: 0 10px;

}

.footer hr{ background: #999; height: 1px; margin: 20px 0px;

}

.footer p{

padding-bottom: 15px;

}

.kitchen {

display: flex; position: relative;

flex-direction: column; align-items: center; justify-content: center;

}

.kitchen #intro { font-size: 45px; margin: 4px; color: #8d0003;

}

.kitchen #sum2 { margin-bottom: 0;

padding: 0;

}

.foodbox img { width: 100%;

}

.foodbox .fooddesc { font-weight: bold;

}

.rdesc {

padding-top: 6px;

}

.roomspic { display: flex;

justify-content: center; align-items: center;

}

.rooms img { width: 90%;

border-radius: 12px; padding: 0;

margin-bottom: 17px; margin-right: 2vh; margin-left: 2vh;

}

.rooms .btn {

margin-bottom: 30px;

}

.kitchen #intro { text-align: center; font-size: 40px; margin: 6px;

}

.kitchen #sum1, #sum2 {

text-align: center; font-size: 18px; color: #000; margin: 6px;

padding-top: 20px;

}

.kitchen #sum2 { font-size: 20px;

padding-bottom: 12px;

}

.foodbox { display: flex; margin: 12px;

flex-direction: row; align-items: center; justify-content: center;

}

.foodcolumn .fooddesc { color: #8d0003;

display: flex;

justify-content: center; padding-bottom: 8px;

}

.foodcolumn div:hover {

background-color: rgba(240, 240, 240, 0.945); border-radius: 34px;

}

.foodbox div { flex-wrap: wrap;

}

.foodbox img { width: 95%; padding: 30px; border-radius: 53px;

}

.foodbox .foodcolumn { box-sizing: border-box; margin-left: 24px; justify-content: center; align-items: center;

}

.rooms #rintro { text-align: center; font-size: 42px; margin: 18px; color: #8d0003;

}

.rooms #rsum1, #rsum2 {

text-align: center; font-size: 18px; margin: 18px; color: #8d0003;

}

.rooms #rsum2 {

font-size: 15px; padding-bottom: 12px;

}

.roomspic { display: flex;

}

.roomspic img { width: 40%; padding: 3px;

}

.rdesc {

flex-direction: column; padding-top: 25px; padding-left: 32px; padding-right: 22px; font-size: 18px;

} #r1, #r3 {

flex-direction: row;

} #r2, #r4 {

flex-direction: row-reverse;

}

button {

padding: 7px 75px; background-color: #ad1a1a; color: whitesmoke;

margin-top: 22px; cursor: pointer;

}

button:hover {

background-color: whitesmoke; color: #ad1a1a;

font-weight: bold;

}

.rooms{ height:auto; width:100%;

}

.rooms h1{ margin:50px 0 0 0; font-size: 13vh; font-weight: bold; text-align: center;

}

.rooms p{ margin:30px 0 0 0; font-size:5vh;

text-align: center;

color: #8d0003;

}

.roomimages{ width:100%; margin:50px auto; display:grid;

grid-template-rows: repeat(auto-fill,minmax(200px,1fr)); grid-template-columns: repeat(auto-fill,minmax(400px,1fr)); grid-auto-rows:1fr;

}

.innerimg{ margin:40px; text-align: center;

}

.innerimg:hover{

box-shadow: 0 0 3px black;

}

.img2:hover{ transform: scale(1.01);

}

@media all and (max-width:400px){

.rooms h1{ margin:50px 0 0 0; font-size: 10vh; font-weight: bold; text-align: center;

}

.rooms p{

font-size:3vh; text-align: center;

}

.innerimg{ margin:30px; text-align: center;

}

}

.content{

backdrop-filter: blur(5px);

box-shadow: 0 25px 45px rgba(0,0,0,0.1); border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); background: rgba(255,255,255,0.1);

border-radius: 10px; width: auto;

}

.action-btns { display: flex;

justify-content: center;

}

.action-btns button { margin: 0 10px; padding: 10px 20px; border: none;

border-radius: 5px; color: #fff;

cursor: pointer;

}

.btn-add {

background-color: #4CAF50;

}

.btn-delete {

background-color: #f44336;

}

.btn-modify {

background-color: #2196F3;

}

input[type="text"],input[type="submit"]

{

margin: 20px; width: auto; padding: 8px; border-radius: 10px;

box-sizing: border-box; margin-bottom: 10px;

background: rgba(255,255,255,0.2); border: none;

outline: none; color: white;

border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); font-size: 16px;

letter-spacing: 1px;

}

</style>

</head>

<body>

<div class="hero">

<video autoplay loop muted plays-inline class="back-video">

<source src="image/islands-2119.mp4" type="video/mp4">

</video>

<header>

<a id ="move" onMousemove="moveone()" class="brand">RS</a>

<div class="menu">

<div class="action-btns">

<form action="/add" method="get">

<button class="btn-add" type="submit">Add</button>

</form>

<form action="/delete" method="get">

<button class="btn-delete" type="submit">Delete</button>

</form>

<form action="/modify" method="get">

<button class="btn-modify" type="submit">Modify</button>

</form>

</div>

<div class="btn">

<i class="fas fa-bars menu-btn"></i>

</div>

</header>

<div class="content">

<form method="post" onsubmit="return validateForm()">

<label style="color: white;padding: 10px;" for="rollnumber">Roll Number:</label>

<input type="text" id="rollnumber" name="rollnumber" required>

<input type="submit" value="Submit">

</form>

</div>

</div>

<script>

function validateForm() {

var x = document.getElementById("rollnumber").value; if (x == "") {

alert("Roll Number must be filled out"); return false;

}

}

</script>

<div class="kitchen" id="kitchen">

<div id="intro">Managing Student Data With A Robust SIS</div>

<div id="sum2">

Managing student data is a critical aspect of running any educational institution. An SIS is a software solution that helps educational <br> institutions manage student data efficiently. With an SIS, schools can easily manage tasks such as enrollment, grading, attendance <br> tracking, scheduling, and communication with students and parents. Also, schools can quickly generate reports, track student <br> performance, and make data-driven decisions to improve academic programs. Overall, an SIS is a critical tool for educational <br> institutions, helping them streamline operations and improve outcomes for students.

</div>

<div class="rooms" id="rooms">

<div id="rintro">What Is Student Information System (SIS)</div>

<div class="roomspic" id="r4"><img src="image/4.jpg" alt="">

<div class="rdesc">

A Student Management System, also known as a Student Information System (SIS) enables coordinate scheduling and improved communications among the faculty and students. This system conducts student tracking operations for parents and administrative staff.

Student Management System (SMS) is a software designed to track and manage all the data created by the school such as a student's grades, attendance, interpersonal activity records, etc., as well as other school-related data. The comprehensive data that includes - student’s registration information, admission detail, fees payment status, hostel billing,

scholarships or any other financial aid provision, etc. can be shared with faculty, students and parents to make the entire process of students’ enrollment quick, systematic, and error-free. Furthermore, the entire students’ information can be saved at a centralized location & role-based login access can be given to all the stakeholders for ensuring 100% student data security. </div>

</div>

</div>

<div class="rooms">

<p>Features Overview</p>

<div class="roomimages">

<span class="innerimg">

<img src="image/1.png" class="img2">

<p class="h11">ADDING CONTENT</p>

<hi>Here we can add details about new student's information</hi>

</span>

<span class="innerimg">

<img src="image/2.png" class="img2">

<p class="h11">DELETING CONTEN</p>

<hi>it helps to remove the details of a student</hi>

</span>

<span class="innerimg">

<img src="image/3.png" class="img2">

<p class="h11">MODIFY CONTENT</p>

<hi>The details of the student can be modified here</hi>

</span>

</span>

</div>

</div>

<section class="section-two">

<h2>About Us:</h2>

<p>Welcome to our Student Management System! We are a dedicated team of professionals passionate about education and committed to providing efficient and effective tools for managing student information. With years of experience in the field, we understand the challenges faced by educational institutions and aim to simplify the process of student management.</p>

<h2>Our Mission:</h2>

<p>At Student Management System, our mission is to empower educational institutions with a comprehensive and user-friendly platform that streamlines student data management. We believe in the power of technology to enhance productivity and enable educators to focus on what they do best: educating and shaping young minds. </p>

<h2>Contact us:</h2>

</section>

<section class="section-tw">

<p>Robust SIS,</p>

<p>611 San Bruno Ave E,</p>

<p>San Bruno, California 94066, USA</p>

<p>Tel: (650) 589-5089 ;</p>

<p>Reservations: (650) 589-5089 ;</p>

<p>Fax: (650) 589-5089 ;</p>

<p>Property Email: [info@Robust.com](mailto:info@Robust.com) </p>

</section>

<script type="text/javascript"> window.addEventListener("scroll", function(){ var header = document.querySelector("header");

header.classList.toggle('sticky', window.scrollY > 0);

});

var menu = document.querySelector('.menu');

var menuBtn = document.querySelector('.menu-btn'); var closeBtn = document.querySelector('.close-btn'); menuBtn.addEventListener("click", () => { menu.classList.add('active');

});

closeBtn.addEventListener("click", () => { menu.classList.remove('active');

});

</script>

<div class="footer">

<a href="https://[www.instagram.com/mastersoft\_edu/](http://www.instagram.com/mastersoft_edu/)"><img src="image/instagram.png"></a>

<a href="https://twitter.com/atriumhotels?lang=en"><img src="image/twitter.png"></a>

<a href="https://[www.facebook.com/CampusManagementSolutions/](http://www.facebook.com/CampusManagementSolutions/)"><img src="image/facebook.png"></a>

<a href="https://[www.youtube.com/channel/UCvlpN5uS4\_g5TJ9u15xoCBQ](http://www.youtube.com/channel/UCvlpN5uS4_g5TJ9u15xoCBQ)"><img src="image/youtube.png"></a>

<a href="https://telegram.com/"><img src="image/telegram.png"></a>

<hr>

<p>Copyright @ 2023, A Robust SIS.</p>

</div>

</body>

</html>

### Add Page:

<!DOCTYPE html>

<html>

<head>

<title>Add Student

</title>

<style>

body

{

background: linear-gradient(#e66465, #9198e5); font-family: Arial, sans-serif;

margin: 20px;

}

form

{

display: flex;

flex-direction: column; gap: 10px;

padding: 2em;

backdrop-filter: blur(5px);

box-shadow: 0 25px 45px rgba(0,0,0,0.1); border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); background: rgba(255,255,255,0.1);

border-radius: 10px; transition: .4s ease-in-out; max-width: 400px; margin: 0 auto;

}

label

{

color: white; display: block; margin-top: 10px; margin-bottom:5px; font-weight: bold;

}

input[type="text"], input[type="date"], select,

textarea

{

width: 100%; padding: 8px; border-radius: 10px;

box-sizing: border-box; margin-bottom: 10px;

background: rgba(255,255,255,0.2); border: none;

outline: none; color: white;

border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); font-size: 16px;

letter-spacing: 1px;

}

input[type="submit"]

{

background-color: #e66465; color: white;

padding: 10px 16px;

border: none; border-radius: 4px; cursor: pointer; font-size: 16px;

}

input[type="submit"]:hover

{

background-color: #45a049;

}

</style>

</head>

<body>

<div style="text-align: center; margin-top: 29px;">

<h2 style="color: white;padding-top: 30px;"> Add a Student

</h2>

</div>

<form method="POST" action="/add">

<label for="rollnumber">Roll Number:

</label>

<input type="text" id="rollnumber" name="rollnumber">

<label for="firstname">First Name:

</label>

<input type="text" id="firstname" name="firstname">

<label for="lastname">Last Name:

</label>

<input type="text" id="lastname" name="lastname">

<label for="dateofbirth">Date of Birth:

</label>

<input type="date" id="dateofbirth" name="dateofbirth">

<label for="gender">Gender:

</label>

<select id="gender" name="gender">

<option value="Male">Male

</option>

<option value="Female">Female

</option>

<option value="Other">Other

</option>

</select>

<label for="email">Email:

</label>

<input type="text" id="email" name="email">

<label for="mobilenumber">Mobile Number:

</label>

<input type="text" id="mobilenumber" name="mobilenumber">

<label for="address">Address:

</label>

<textarea id="address" name="address">

</textarea>

<label for="city">City:

</label>

<input type="text" id="city" name="city">

<label for="state">State:

</label>

<input type="text" id="state" name="state">

<label for="pincode">Pincode:

</label>

<input type="text" id="pincode" name="pincode">

<label for="country">Country:

</label>

<input type="text" id="country" name="country">

<label for="department">Department:

</label>

<input type="text" id="department" name="department">

<label for="course">Course:

</label>

<input type="text" id="course" name="course">

<label for="yearofjoining">Year of Joining:

</label>

<input type="text" id="yearofjoining" name="yearofjoining">

<label for="yearofcompletion">Year of Completion:

</label>

<input type="text" id="yearofcompletion" name="yearofcompletion">

<input type="submit" value="Add Student">

</form>

</body>

</html>

### Delete Page:

<!DOCTYPE html>

<html>

<head>

<title>Delete Student

</title>

</head>

<body>

<style>

body

{

background: linear-gradient(#e66465, #9198e5); font-family: Arial, sans-serif;

margin: 0;

box-sizing: border-box; overflow: hidden; background-repeat: no-repeat; text-align: center;

height: 100vh;

}

form

{

display: flex;

flex-direction: column; gap: 10px;

padding: 2em;

backdrop-filter: blur(5px);

box-shadow: 0 25px 45px rgba(0,0,0,0.1); border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); background: rgba(255,255,255,0.1);

border-radius: 10px; transition: .4s ease-in-out; max-width: 400px; margin-top: 220px; margin-left: 35%;

}

label

{

color: white; display: block; margin-top: 10px; margin-bottom:5px; font-weight: bold;

}

input[type="text"]

{

width: 100%; padding: 8px; border-radius: 10px;

box-sizing: border-box; margin-bottom: 10px;

background: rgba(255,255,255,0.2); border: none;

outline: none; color: white;

border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); font-size: 16px;

letter-spacing: 1px;

}

input[type="submit"]

{

background-color: #7272e3; color: white;

padding: 10px 16px; border: none;

border-radius: 4px; cursor: pointer; font-size: 16px;

}

input[type="submit"]:hover

{

background-color: #45a049;

}

</style>

<form method="POST">

<label for="rollnumber">Roll Number:

</label>

<br>

<input type="text" id="rollnumber" name="rollnumber" placeholder="Enter the Roll No">

<br>

<input type="submit" value="Submit">

</form>

</body>

</html>

### Modify Page:

<!DOCTYPE html>

<html>

<head>

<title>Modify Student</title>

</head>

<body>

<style>

body

{

background: linear-gradient(#e66465, #9198e5); font-family: Arial, sans-serif;

margin: 20px;

}

form

{

display: flex;

flex-direction: column; padding: 20px;

backdrop-filter: blur(5px);

box-shadow: 0 25px 45px rgba(0,0,0,0.1); border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); background: rgba(255,255,255,0.1);

border-radius: 10px; transition: .4s ease-in-out; max-width: 570px;

margin: 0 auto;

}

label

{

color: white; display: block; margin-top: 10px; margin-bottom:5px; font-weight: bold;

}

input[type="text"], input[type="date"], input[type="email"], select,

textarea

{

width: 100%; padding: 8px; border-radius: 10px;

box-sizing: border-box; margin-bottom: 10px;

background: rgba(255,255,255,0.2); border: none;

outline: none; color: white;

border: 1px solid rgba(255,255,255,0.5); border-right: 1px solid rgba(255,255,255,0.2);

border-bottom: 1px solid rgba(255,255,255,0.2); font-size: 16px;

letter-spacing: 1px;

}

input[type="submit"]

{

background-color: #e66465; color: white;

padding: 10px 16px; border: none;

border-radius: 4px; cursor: pointer; font-size: 16px;

}

input[type="submit"]:hover

{

background-color: #45a049;

}

</style>

<h1 style="color: white;text-align: center;padding-top: 20px;">Modify Student Details

</h1>

<form method="POST" action="/modify">

<label for="rollnumber">Roll Number:

</label>

<br>

<input type="text" id="rollnumber" name="rollnumber">

<br>

<label for="firstname">First Name:

</label>

<br>

<input type="text" id="firstname" name="firstname">

<br>

<label for="lastname">Last Name:

</label>

<br>

<input type="text" id="lastname" name="lastname">

<br>

<label for="dateofbirth">Date of Birth:

</label>

<br>

<input type="date" id="dateofbirth" name="dateofbirth">

<br>

<label for="gender">Gender:

</label>

<br>

<select id="gender" name="gender">

<option style="color: gray;" value="Male">Male

</option>

<option style="color: gray;" value="Female">Female

</option>

<option style="color: gray;" value="Other">Other

</option>

</select><br>

<label for="email">Email:

</label>

<br>

<input type="email" id="email" name="email">

<br>

<label for="mobilenumber">Mobile Number:

</label>

<br>

<input type="text" id="mobilenumber" name="mobilenumber">

<br>

<label for="address">Address:

</label>

<br>

<textarea id="address" name="address">

</textarea>

<br>

<label for="city">City:

</label>

<br>

<input type="text" id="city" name="city">

<br>

<label for="state">State:

</label>

<br>

<input type="text" id="state" name="state">

<br>

<label for="pincode">Pincode:

</label>

<br>

<input type="text" id="pincode" name="pincode">

<br>

<label for="country">Country:

</label>

<br>

<input type="text" id="country" name="country">

<br>

<label for="department">Department:

</label>

<br>

<input type="text" id="department" name="department">

<br>

<label for="course">Course:

</label>

<br>

<input type="text" id="course" name="course">

<br>

<label for="yearofjoining">Year of Joining:

</label>

<br>

<input type="text" id="yearofjoining" name="yearofjoining">

<br>

<label for="yearofcompletion">Year of Completion:

</label>

<br>

<input type="text" id="yearofcompletion" name="yearofcompletion">

<br>

<input type="submit" value="Submit">

</form>

</body>

</html>

### Database :

from flask import Flask, render\_template, request, redirect, url\_for, flash

from flask\_login import LoginManager, UserMixin, login\_user, login\_required, logout\_user, current\_user

import mysql.connector

app = Flask( name )

app.secret\_key = 'some-secret-key' # Change this!

# Flask-Login setup login\_manager = LoginManager() login\_manager.init\_app(app)

login\_manager.login\_view = 'login' # Redirect to this view if user not logged in

class User(UserMixin):

def init (self, username, id): self.id = id

self.username = username

@login\_manager.user\_loader def load\_user(user\_id):

return User(user\_id, user\_id)

def authenticate(username, password):

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor() cursor.execute("USE manalur\_temple\_dev")

cursor.execute("SELECT \* FROM admin WHERE username = %s AND password =

%s", (username, password,)) data = cursor.fetchone() connection.close()

if data:

return True

else:

return False

@app.route('/login', methods=['GET', 'POST']) def login():

if request.method == 'POST':

username = request.form.get('username') password = request.form.get('password')

# Validate login credentials

if username and password and authenticate(username, password): user = User(username, username)

login\_user(user)

return redirect(url\_for('display\_table')) # Redirect to main page after successful

login

else:

flash('Wrong username or password!') return render\_template('login.html')

@app.route('/logout') @login\_required

def logout(): logout\_user()

return redirect(url\_for('login'))

# Replace these values with your own MySQL server credentials config = {

"host": "localhost",

"user": "root",

"password": "prathiish@123"

}

def get\_student\_data(roll\_number=None):

try:

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

if roll\_number:

cursor.execute("SELECT \* FROM StudentInformation WHERE RollNumber =

%s", (roll\_number,)) else:

cursor.execute("SELECT \* FROM StudentInformation") data = cursor.fetchall()

cursor.execute("DESCRIBE StudentInformation") columns = [column[0] for column in cursor.fetchall()]

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

return columns, data

@app.route('/', methods=['GET', 'POST']) @login\_required

def display\_table(): roll\_number = None table = ''

if request.method == 'POST':

roll\_number = request.form.get('rollnumber') if roll\_number:

columns, data = get\_student\_data(roll\_number) else:

pass

if data:

table = '''<table border="1"><tr>{}</tr>{}</table>'''.format( ''.join('<th>{}</th>'.format(column) for column in columns), ''.join('<tr>{}</tr>'.format(''.join('<td>{}</td>'.format(value) for value in row))

for row in data)

)

else:

table = '<p>No data found for the provided roll number.</p>' return render\_template('index.html', table=table)

# Your existing code...

@app.route('/add', methods=['GET', 'POST']) @login\_required

def add\_student():

if request.method == 'POST':

roll\_number = request.form.get('rollnumber') first\_name = request.form.get('firstname') last\_name = request.form.get('lastname') date\_of\_birth = request.form.get('dateofbirth') gender = request.form.get('gender')

email = request.form.get('email')

mobile\_number = request.form.get('mobilenumber') address = request.form.get('address')

city = request.form.get('city') state = request.form.get('state')

pincode = request.form.get('pincode') country = request.form.get('country') department = request.form.get('department')

course = request.form.get('course') year\_of\_joining = request.form.get('yearofjoining')

year\_of\_completion = request.form.get('yearofcompletion')

try:

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

cursor.execute("INSERT INTO StudentInformation (RollNumber, FirstName, LastName, DateOfBirth, Gender, Email, MobileNumber, Address, City, State, Pincode, Country, Department, Course, YearOfJoining, YearOfCompletion) VALUES (%s, %s,

%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)", (roll\_number, first\_name, last\_name, date\_of\_birth, gender, email, mobile\_number, address, city, state, pincode, country, department, course, year\_of\_joining, year\_of\_completion))

connection.commit()

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

return render\_template('add.html')

@app.route('/delete', methods=['GET', 'POST']) @login\_required

def delete\_student():

if request.method == 'POST':

roll\_number = request.form.get('rollnumber') try:

connection = mysql.connector.connect(\*\*config)

cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

cursor.execute("DELETE FROM StudentInformation WHERE RollNumber =

%s", (roll\_number,))

connection.commit()

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

return render\_template('delete.html') # You need to create this template

@app.route('/modify', methods=['GET', 'POST']) @login\_required

def modify\_student():

if request.method == 'POST':

roll\_number = request.form.get('rollnumber') first\_name = request.form.get('firstname') last\_name = request.form.get('lastname') date\_of\_birth = request.form.get('dateofbirth') gender = request.form.get('gender')

email = request.form.get('email')

mobile\_number = request.form.get('mobilenumber') address = request.form.get('address')

city = request.form.get('city') state = request.form.get('state')

pincode = request.form.get('pincode') country = request.form.get('country') department = request.form.get('department') course = request.form.get('course')

year\_of\_joining = request.form.get('yearofjoining') year\_of\_completion = request.form.get('yearofcompletion')

try:

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

cursor.execute("UPDATE StudentInformation SET FirstName = %s, LastName =

%s, DateOfBirth = %s, Gender = %s, Email = %s, MobileNumber = %s, Address = %s, City = %s, State = %s, Pincode = %s, Country = %s, Department = %s, Course = %s, YearOfJoining = %s, YearOfCompletion = %s WHERE RollNumber = %s", (first\_name, last\_name, date\_of\_birth, gender, email, mobile\_number, address, city, state, pincode, country, department, course, year\_of\_joining, year\_of\_completion, roll\_number))

connection.commit()

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

return render\_template('modify.html') # You need to create this template

if name == ' main ': app.run(debug=True)

import tkinter as tk

from tkinter import ttk, messagebox from ttkthemes import ThemedStyle import mysql.connector

config = {

"host": "localhost",

"user": "root", "password": "toor"

}

def get\_student\_data(roll\_number=None): try:

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

if roll\_number:

cursor.execute("SELECT \* FROM StudentInformation WHERE RollNumber =

%s", (roll\_number,)) else:

cursor.execute("SELECT \* FROM StudentInformation") data = cursor.fetchall()

cursor.execute("DESCRIBE StudentInformation") columns = [column[0] for column in cursor.fetchall()]

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

return columns, data

def login\_user():

username = username\_entry.get()

password = password\_entry.get()

try:

connection = mysql.connector.connect(\*\*config) cursor = connection.cursor()

cursor.execute("USE manalur\_temple\_dev")

cursor.execute("SELECT \* FROM admin WHERE username = %s AND password

= %s", (username, password)) data = cursor.fetchone()

except mysql.connector.Error as err: print("Error: {}".format(err))

finally:

cursor.close() connection.close()

if data is not None:

messagebox.showinfo("Login info", "Login successful") login\_frame.pack\_forget()

main\_frame.pack() else:

messagebox.showinfo("Login info", "Wrong username or password")

def back\_to\_login(): main\_frame.pack\_forget() login\_frame.pack()

def display\_data():

roll\_number = roll\_number\_entry.get() columns, data = get\_student\_data(roll\_number)

for i in range(len(data)):

for j in range(len(columns)):

e = tk.Entry(main\_frame, width=10, fg='blue') e.grid(row=i+3, column=j)

e.insert(tk.END, data[i][j]) root = tk.Tk()

style = ThemedStyle(root) style.set\_theme("arc") # set the theme

login\_frame = ttk.Frame(root) login\_frame.pack()

username\_label = ttk.Label(login\_frame, text="Username") username\_label.grid(row=0, column=0)

username\_entry = ttk.Entry(login\_frame) username\_entry.grid(row=0, column=1)

password\_label = ttk.Label(login\_frame, text="Password") password\_label.grid(row=1, column=0)

password\_entry = ttk.Entry(login\_frame, show="\*") password\_entry.grid(row=1, column=1)

login\_button = ttk.Button(login\_frame, text="Login", command=login\_user) login\_button.grid(row=2, column=0, columnspan=2)

main\_frame = ttk.Frame(root)

roll\_number\_label = ttk.Label(main\_frame, text="Roll Number") roll\_number\_label.grid(row=0, column=0)

roll\_number\_entry = ttk.Entry(main\_frame)

roll\_number\_entry.grid(row=0, column=1)

fetch\_button = ttk.Button(main\_frame, text='Fetch', command=display\_data) fetch\_button.grid(row=1, column=0, columnspan=2)

back\_button = ttk.Button(main\_frame, text='Back', command=back\_to\_login) back\_button.grid(row=1, column=2, columnspan=2)

root.mainloop()

import os print(os.urandom(16).hex())

# 13d953a2868ca0c80eb8e89ef489a04d

1. OUTPUT WITH SCREEN SHOTS:

### Login Page:

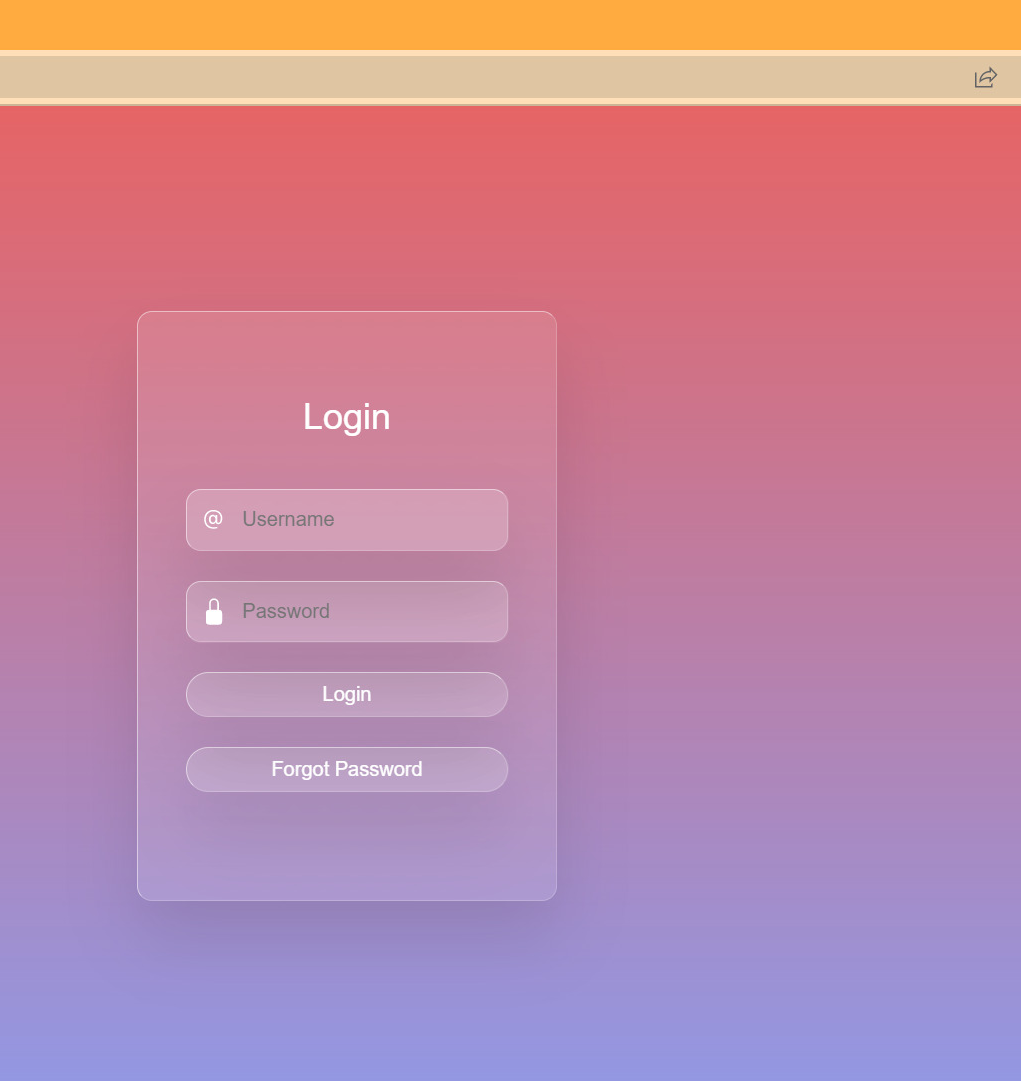


Figure: 1.1

### Home Page:

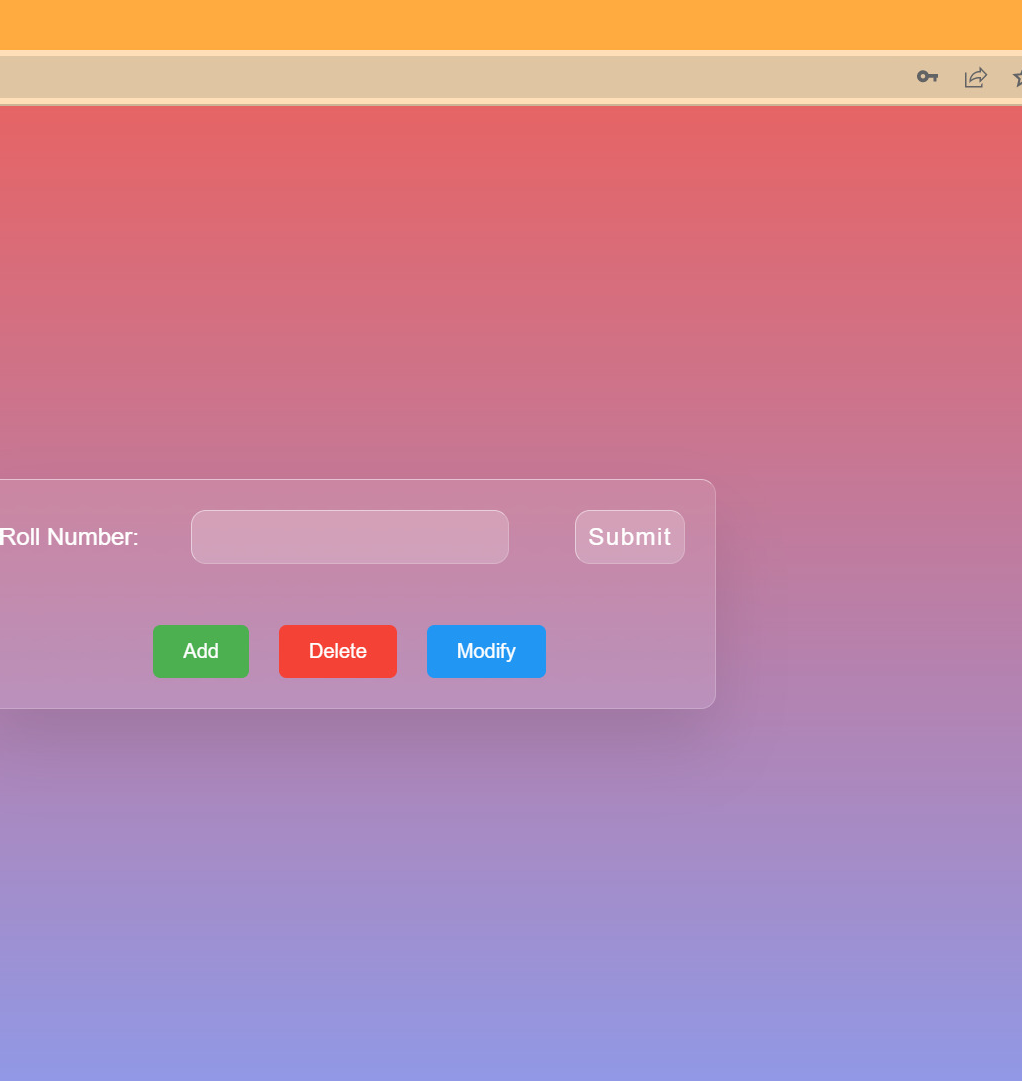
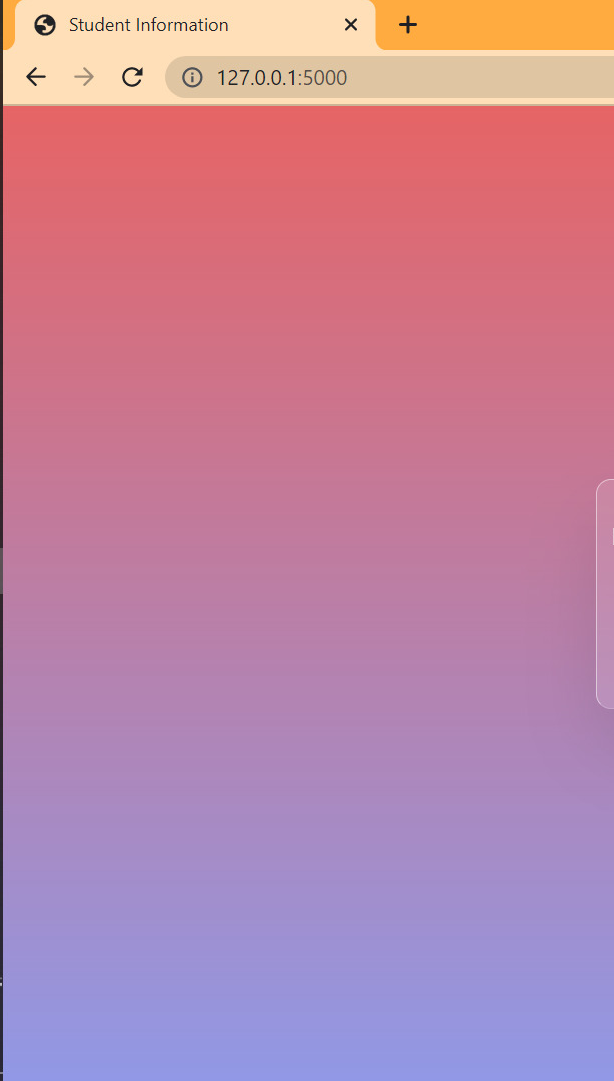


Figure: 1.2

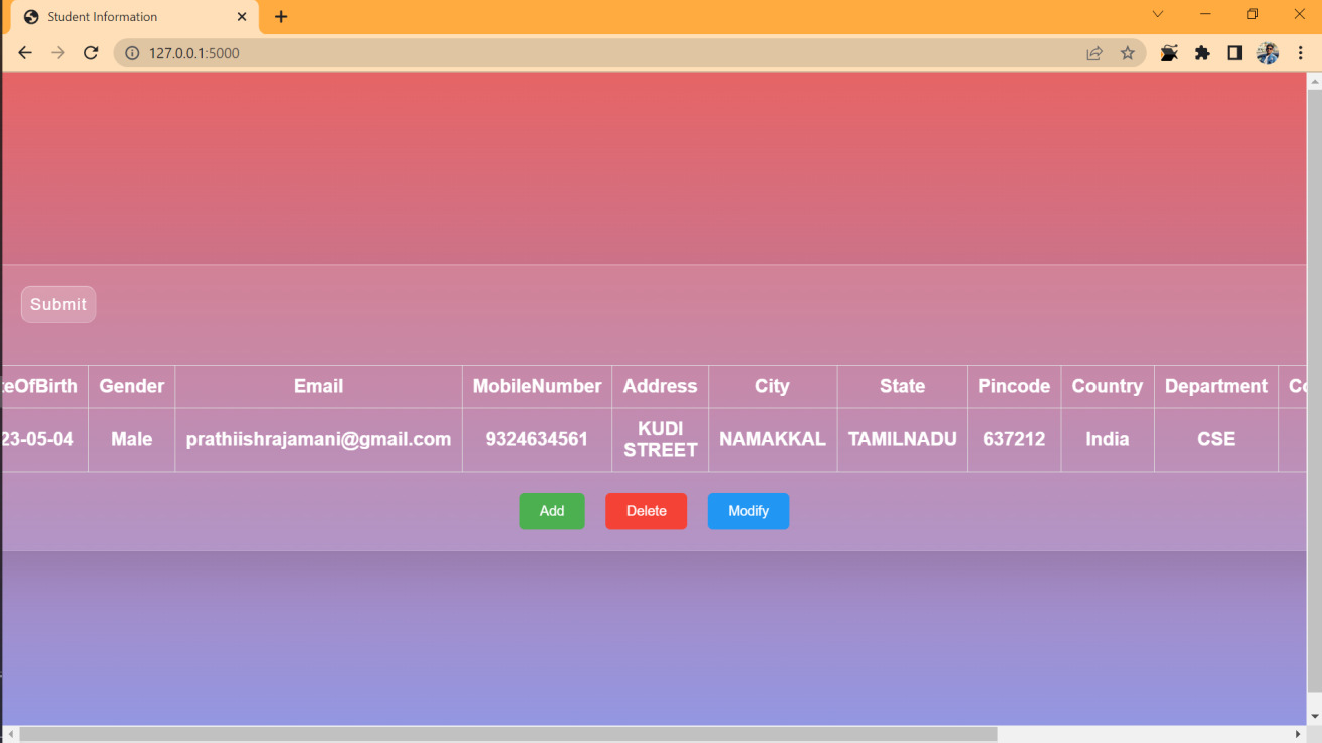


Figure: 1.3

### Add Page:

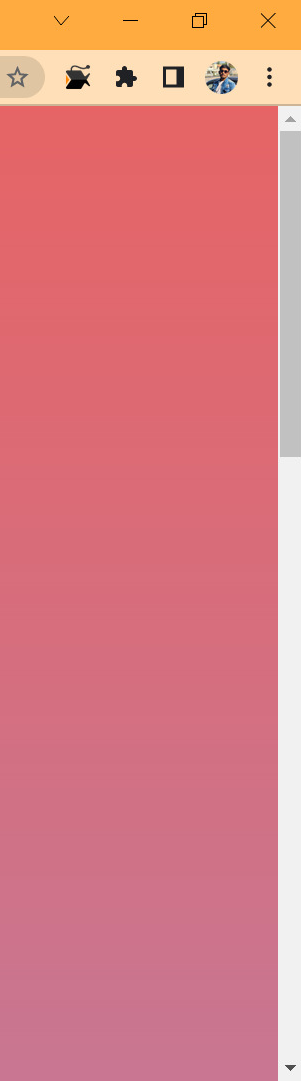
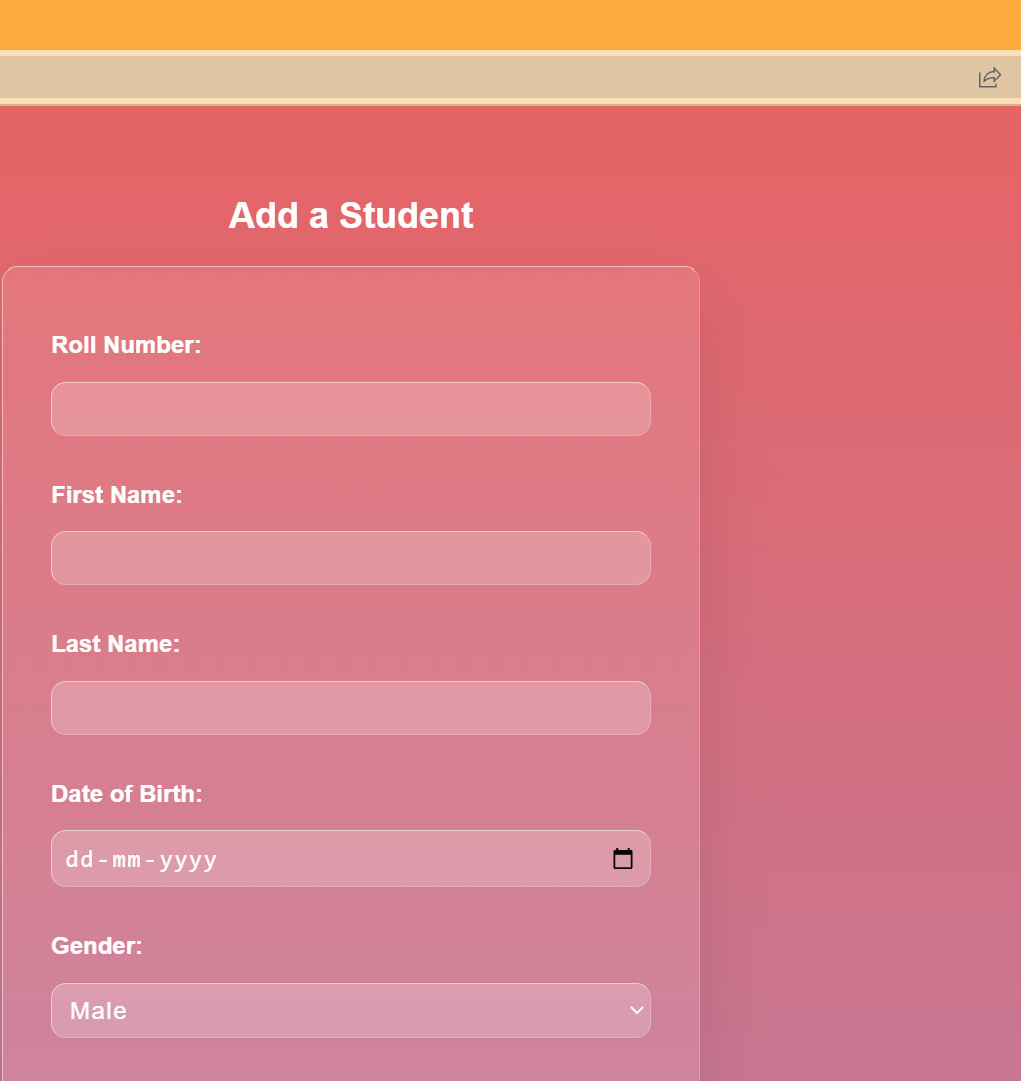
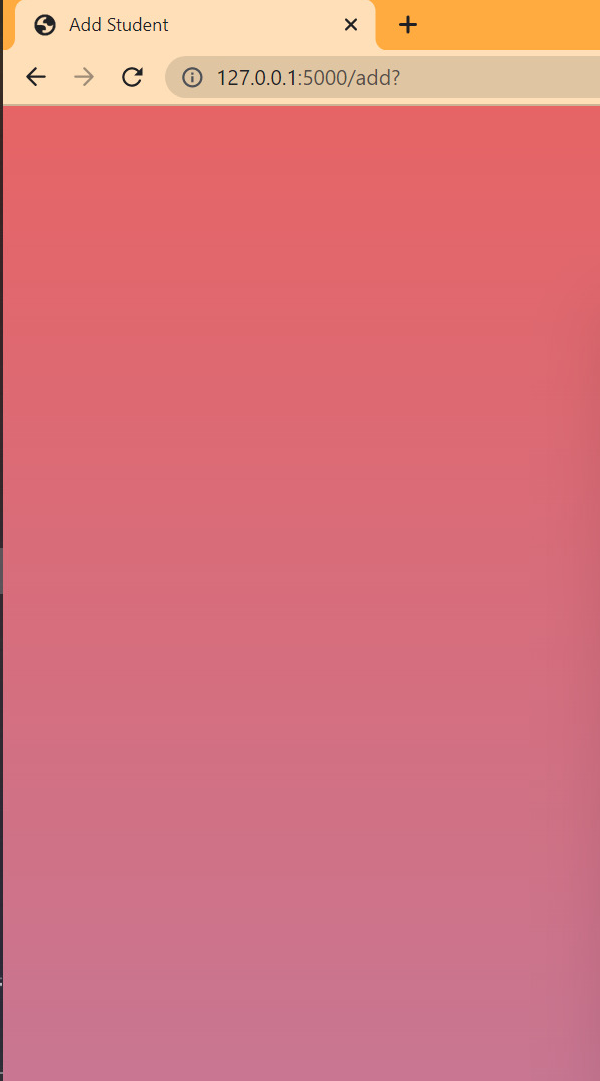


Figure: 1.4

### Delete Page:

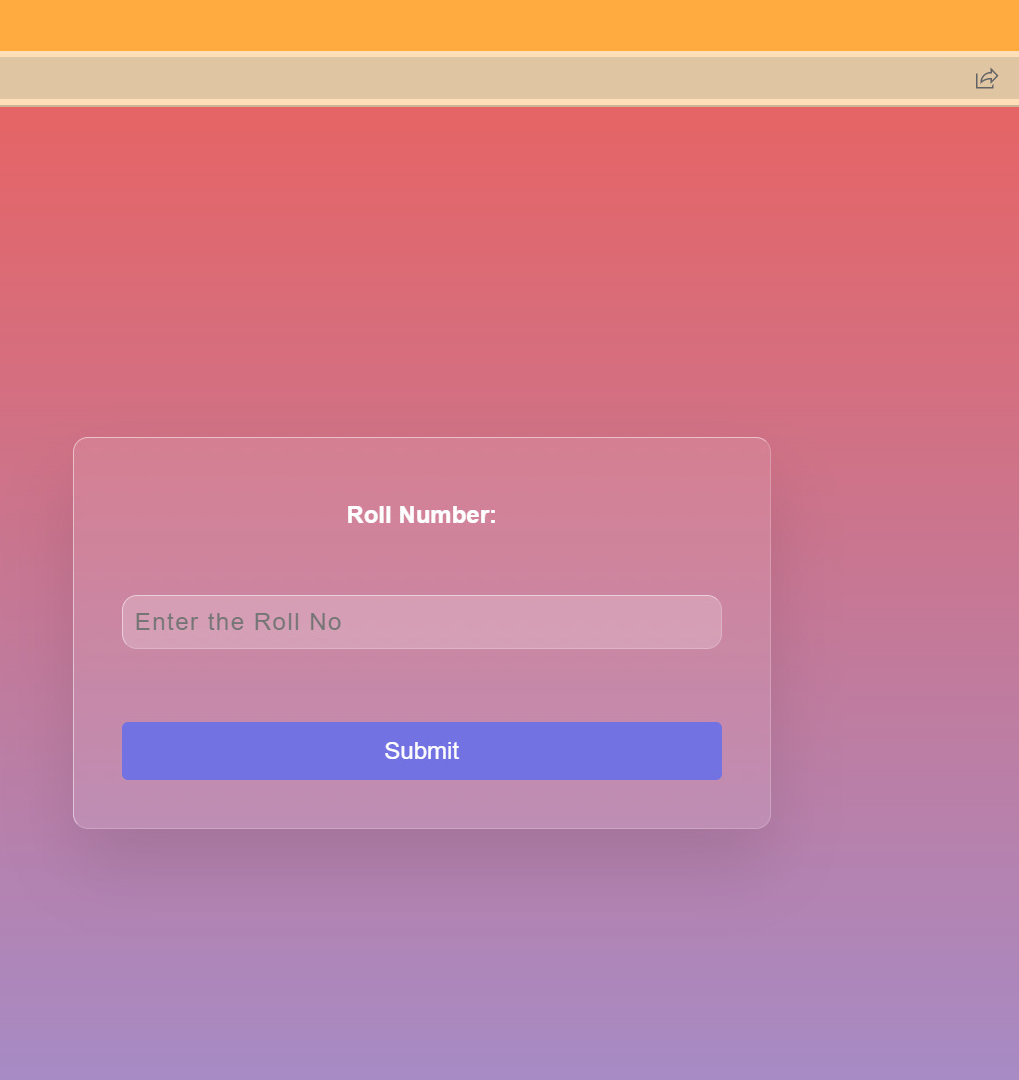


Figure: 1.5

### Modify Page:

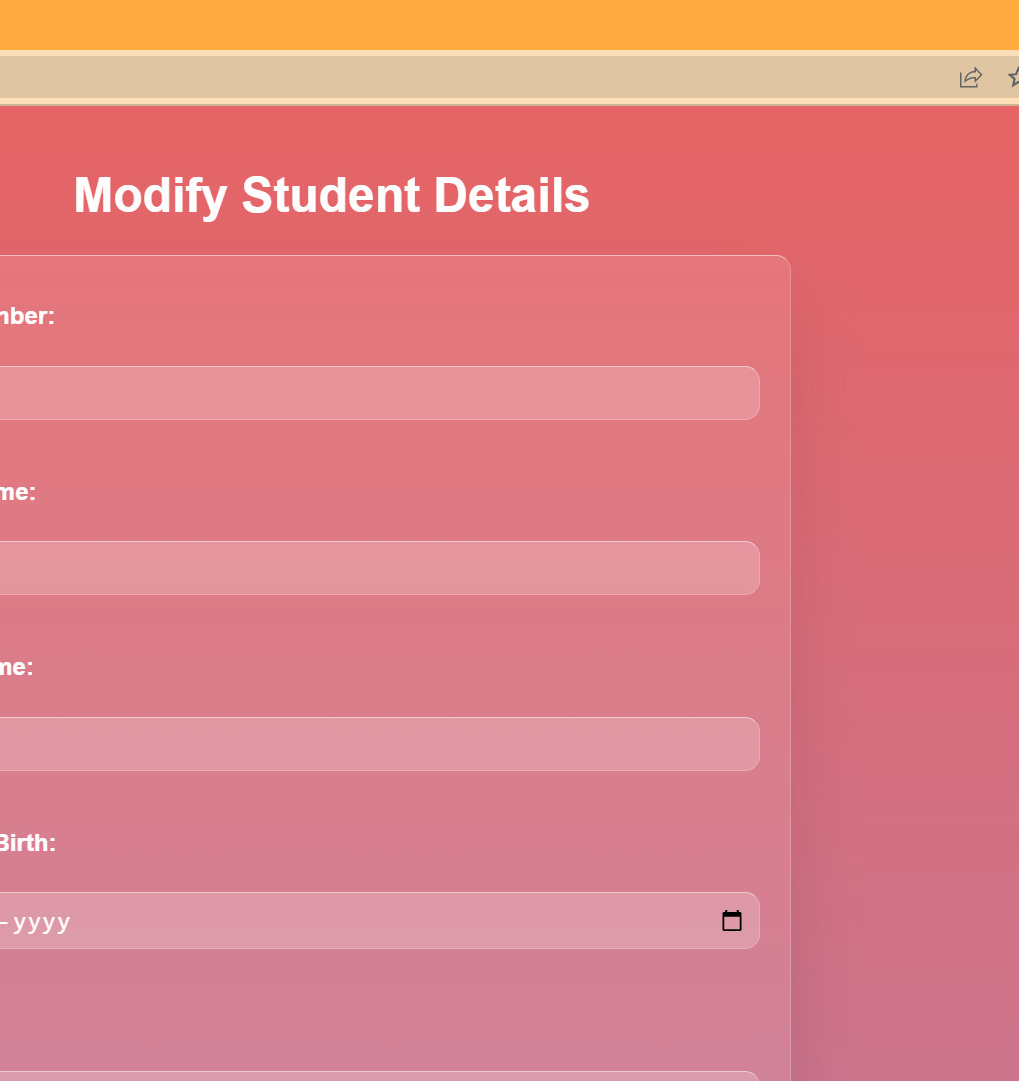
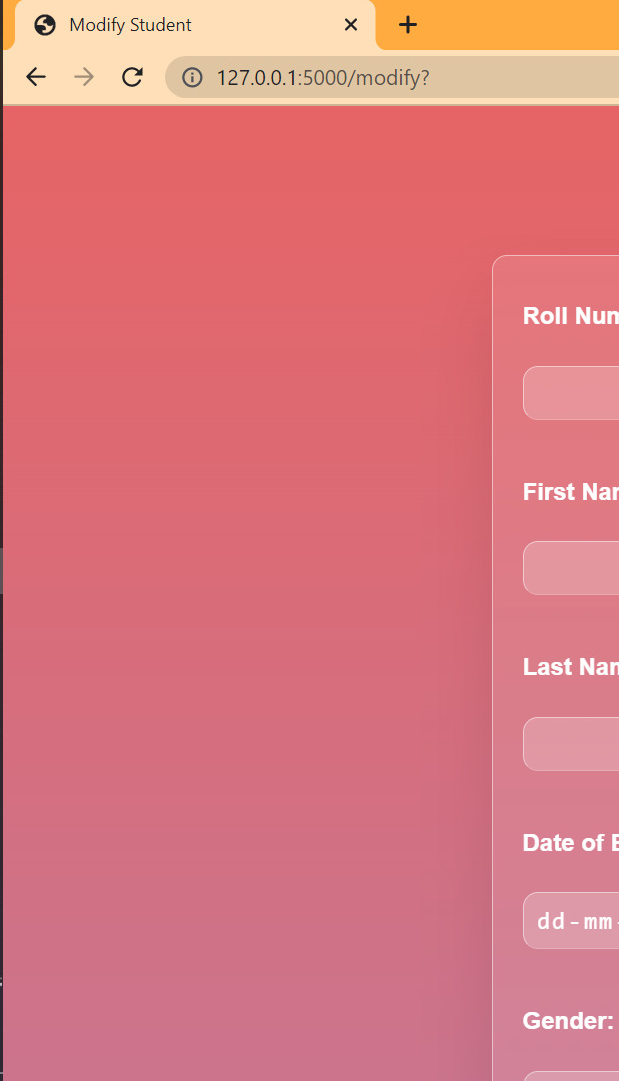


Figure: 1.6

### Database :

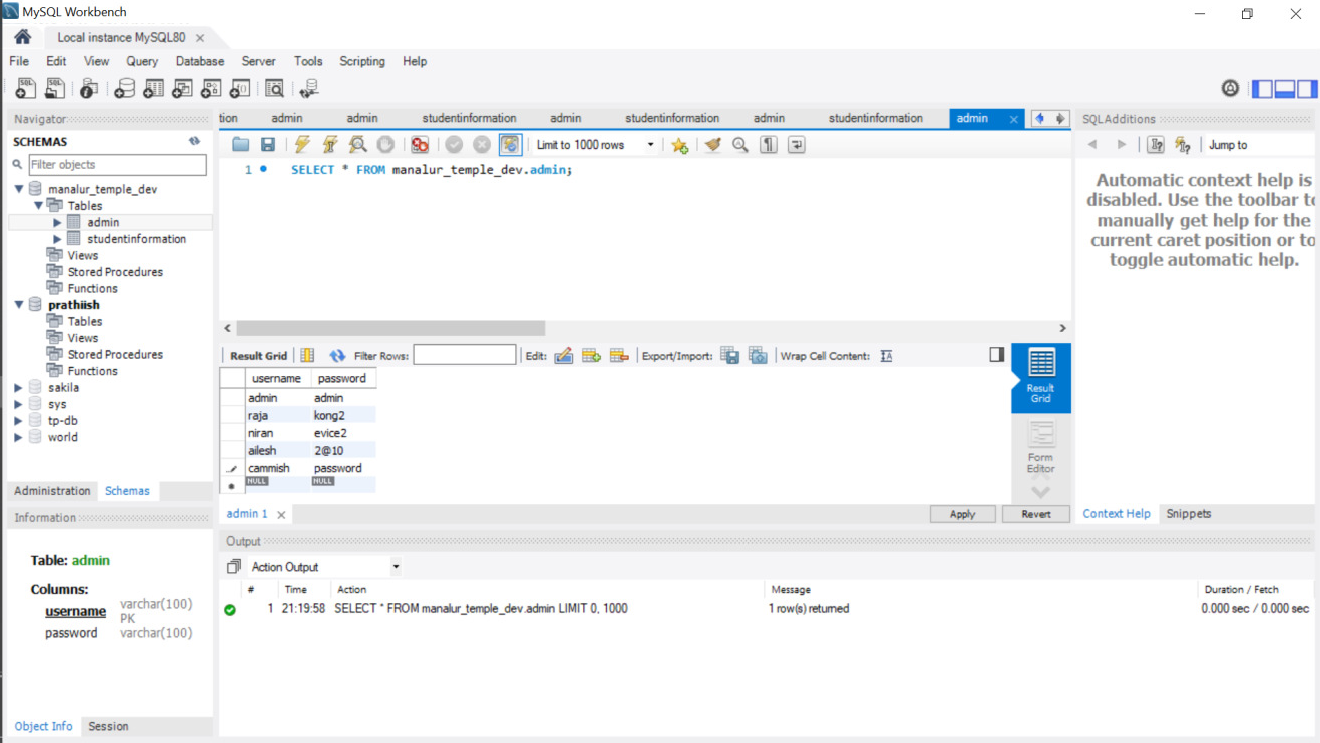


Figure: 1.7

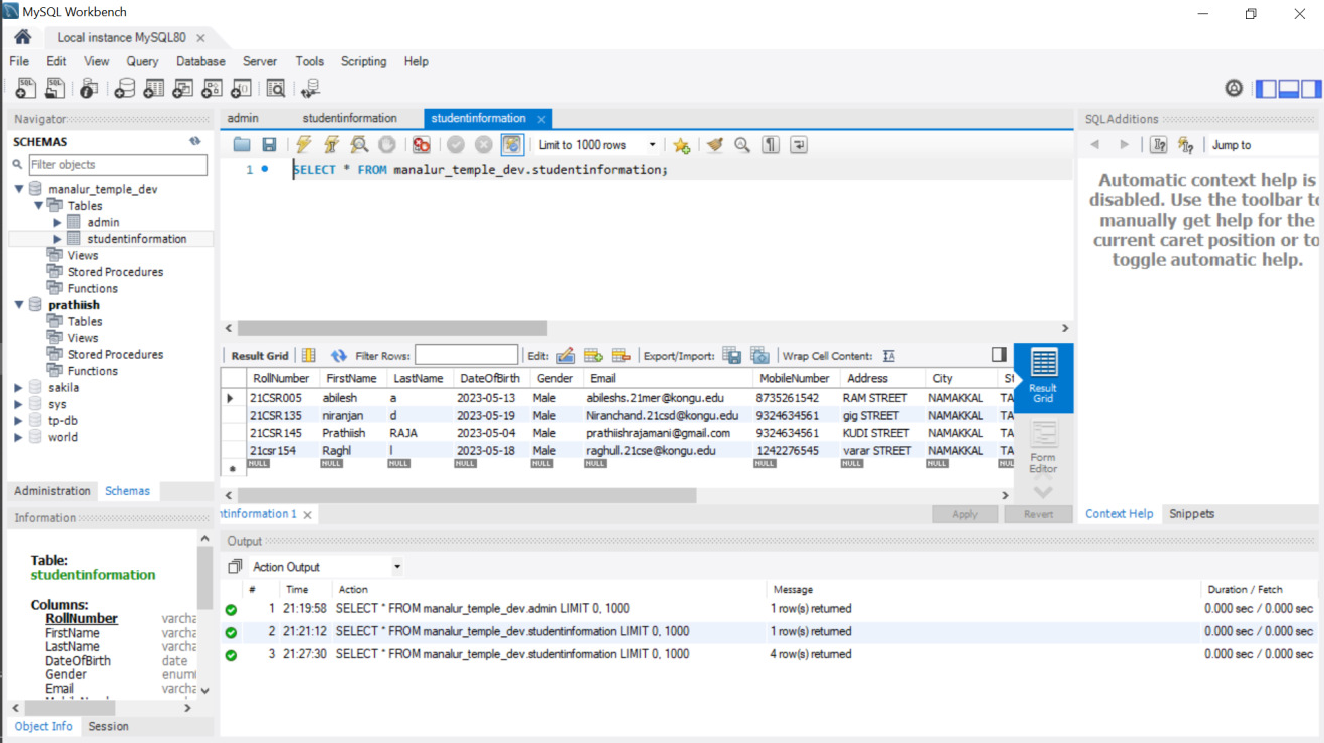


Figure: 1.8

# REFERENCE :

SQL:

"SQL Tutorial" by W3Schools

[Online]. Available: https://[www.w3schools.com/sql/](http://www.w3schools.com/sql/) Accessed: [Insert Date Accessed]

"SQL - Databases" by TutorialsPoint

[Online]. Available: https://[www.tutorialspoint.com/sql/sql-databases.htm](http://www.tutorialspoint.com/sql/sql-databases.htm) Accessed: [Insert Date Accessed]

Flask:

"Flask Documentation" by Flask

[Online]. Available: https://flask.palletsprojects.com/ Accessed: [Insert Date Accessed]

"Flask Mega-Tutorial" by Miguel Grinberg

[Online]. Available: https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i- hello-world

Accessed: [Insert Date Accessed] Visual Studio Code (VS Code):

"VS Code Documentation" by Microsoft

[Online]. Available: https://code.visualstudio.com/docs Accessed: [Insert Date Accessed]

"Visual Studio Code Tips and Tricks" by VSCodeTips [Online]. Available: https://vscode.tips/

Accessed: [Insert Date Accessed]