KEC EVENT MANAGEMENT SYSTEM

TEAM MEMBERS:

|  |  |
| --- | --- |
| ROLL NUMBER: | NAME: |
| VARSHA S | 22ALR109 |
| SIDHARTH S | 22ALR092 |
| YASWANTH RAJ E | 22ALR114 |

**INTRODUCTION:**

An Event Management System (EMS) is a software solution designed to automate and streamline the management and operations of event planning and execution. It integrates various modules and functionalities to digitize administrative tasks, enhance coordination, and improve overall efficiency. The system encompasses features such as event creation, approval workflows, participant registration, and reporting. By implementing an EMS, organizations can optimize their workflows, reduce paperwork, enhance coordination among stakeholders, and ultimately deliver better event experiences.

**Key Features:**

1. **User Logins**:

* Central, Student, and Event Organizer have distinct login roles.

1. **Event Creation**:

* Event Organizers can enter event details including name, date, time, location, and description.

1. **Event Approval**:

* Central users review and approve or reject event submissions. Approved events are displayed to students.

1. **Event Display and Registration**:

* Students can view a list of approved events and register for them.

1. **Participant Management**:

* Event Organizers can access details of registered participants.

1. **Document Generation**:

* Participant details can be generated as a PDF.

1. **Calendar Scheduling**:

* Events are managed through an integrated calendar system to avoid conflicts and ensure proper timing.

1. **Secure Login**:

* Implements a two-step verification process for secure access.

1. **Real-Time Notifications**:

* Sends notifications to users regarding event approvals, registrations, and updates.

1. **Reporting and Analytics**:

* Provides detailed reports and analytics on event participation, approval rates, and other metrics.

1. **Mobile-Friendly Interface**:

* Ensures the web application is accessible and fully functional on mobile devices.

**REQUIREMENTS:**

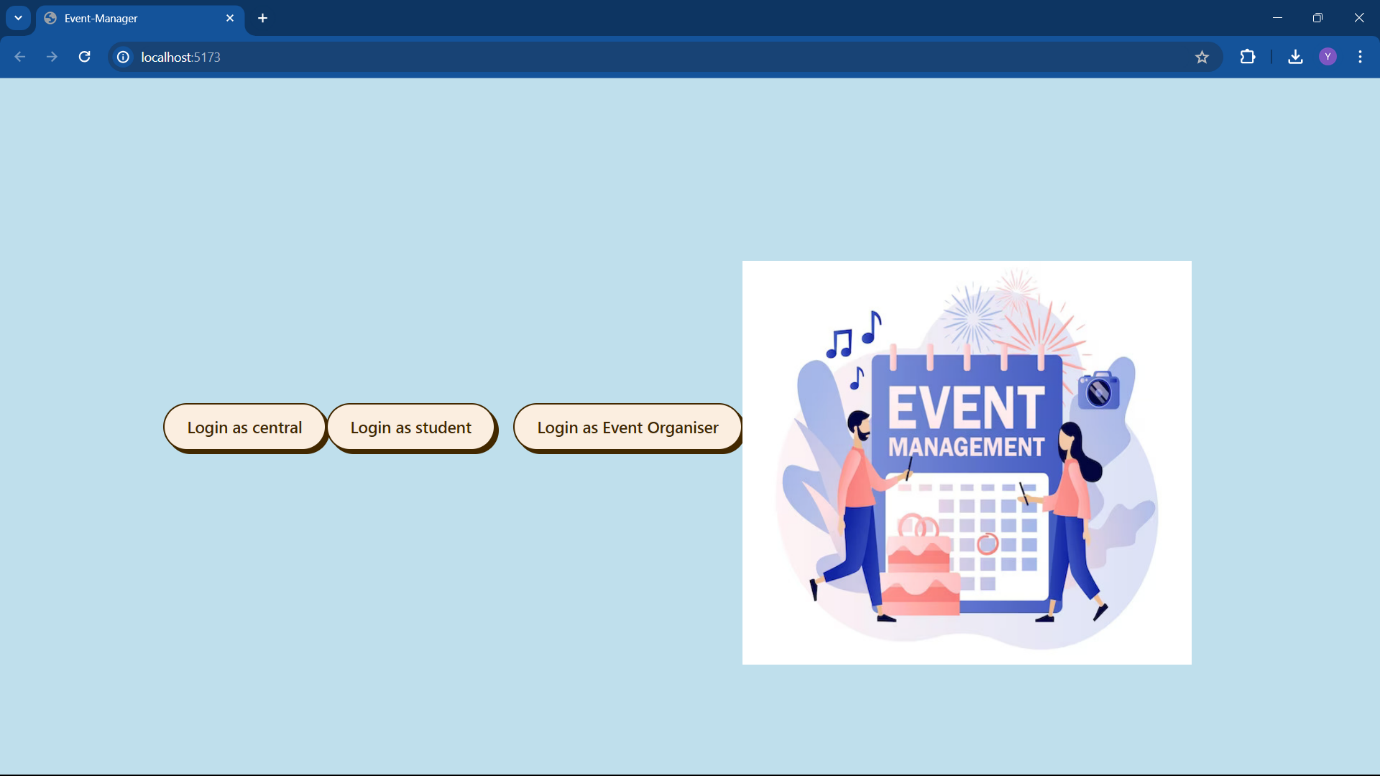
**Front-End Programming:**

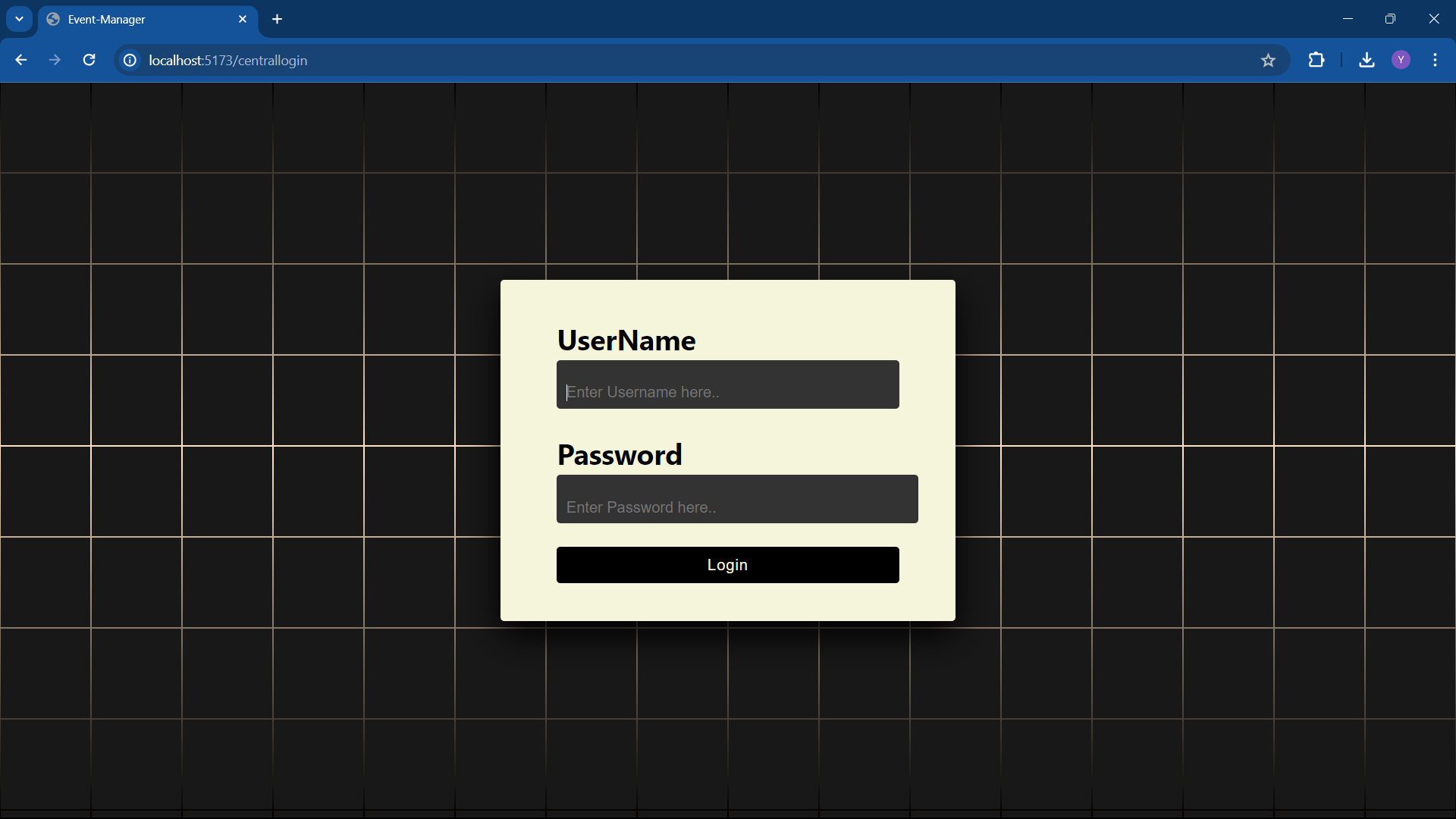
1. **CSS (Cascading Style Sheets)**:
   * CSS is used to define the visual appearance and layout of web pages. It allows developers to apply styles, such as colors, fonts, spacing, and positioning, to HTML elements, creating a visually appealing and consistent user interface.
2. **JavaScript**:
   * JavaScript is a powerful programming language that adds interactivity and dynamic behavior to web pages. It enables developers to manipulate and modify HTML and CSS, handle user interactions, perform calculations, validate forms, and make asynchronous requests to servers.
3. **React**:
   * React is a popular JavaScript library for building user interfaces. It provides a component-based approach to web development, allowing developers to create reusable UI components that update efficiently based on changes in data. React uses a virtual DOM (Document Object Model) to optimize performance and facilitate the building of complex web applications.

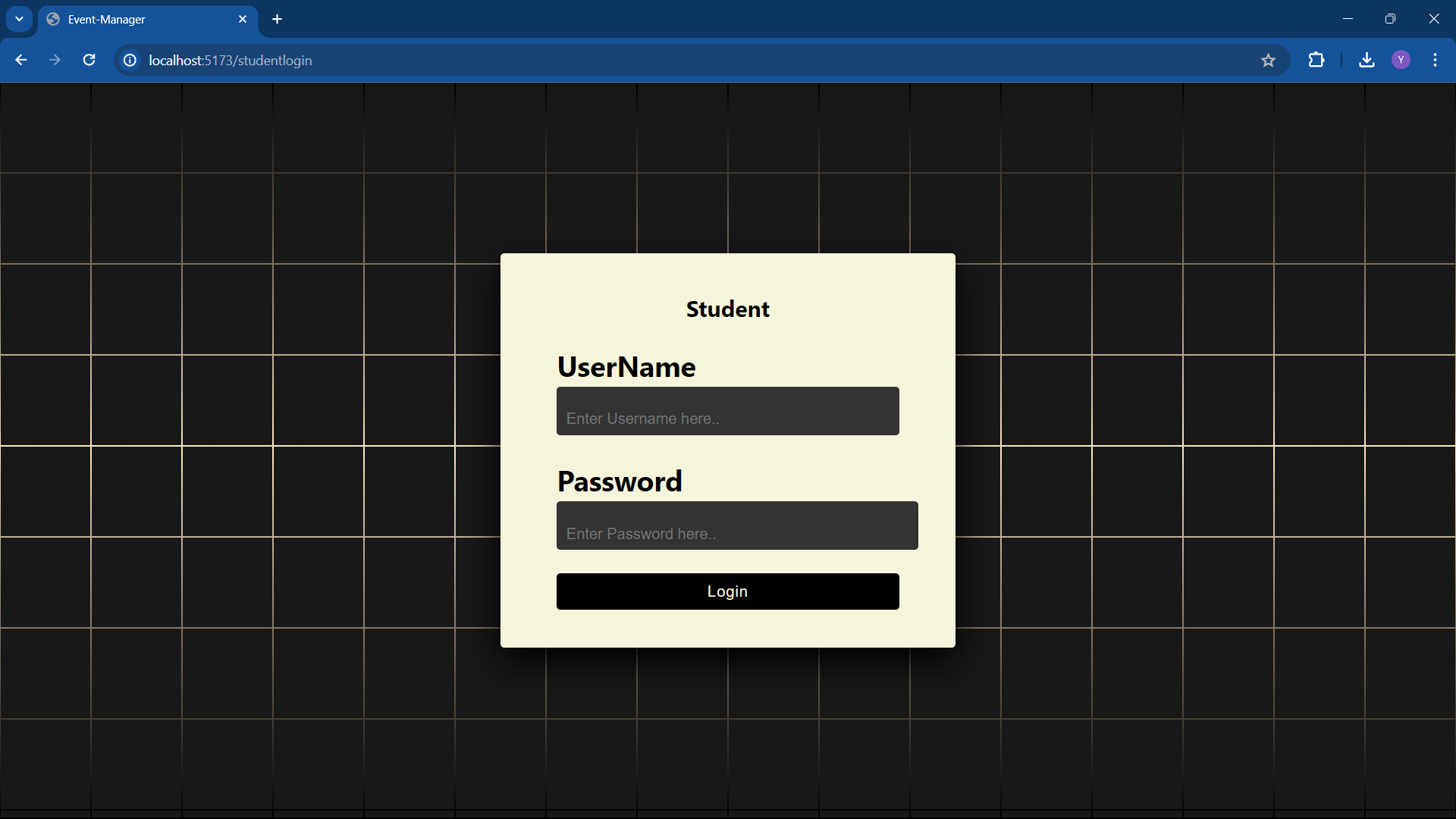
**Back-End Programming**

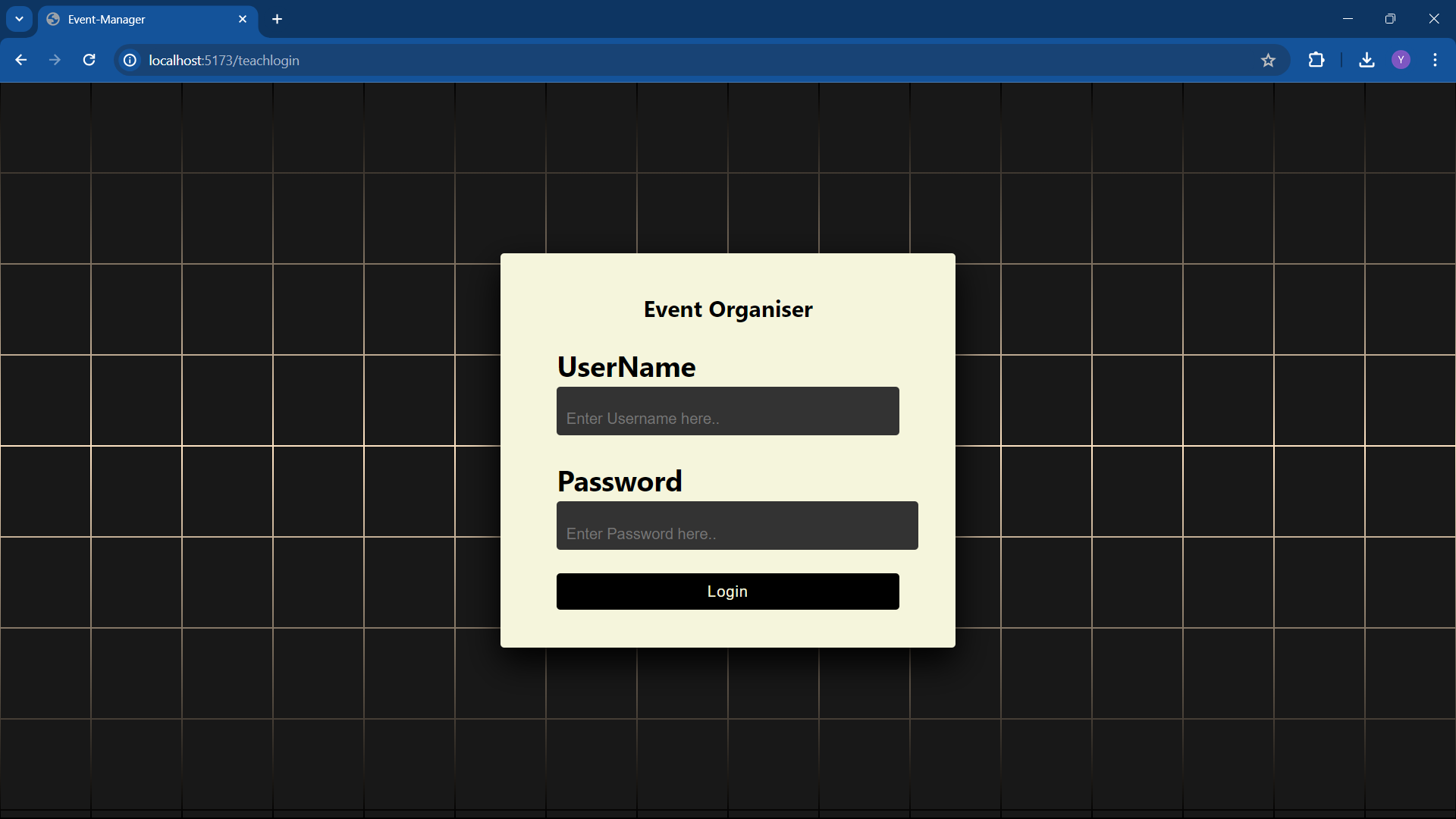
1. **Database Management**:
   * Backend programming involves working with databases to store and retrieve data. Developers interact with databases using query languages like SQL or NoSQL databases. For this system, MongoDB is used to design and implement database schemas, optimize queries, and ensure data integrity and security.
2. **APIs (Application Programming Interfaces)**:
   * Backend developers build APIs that allow communication between the frontend and backend components of a web application. APIs define the protocols and rules for how different software components can interact and exchange data. Commonly used API standards include RESTful APIs and GraphQL.

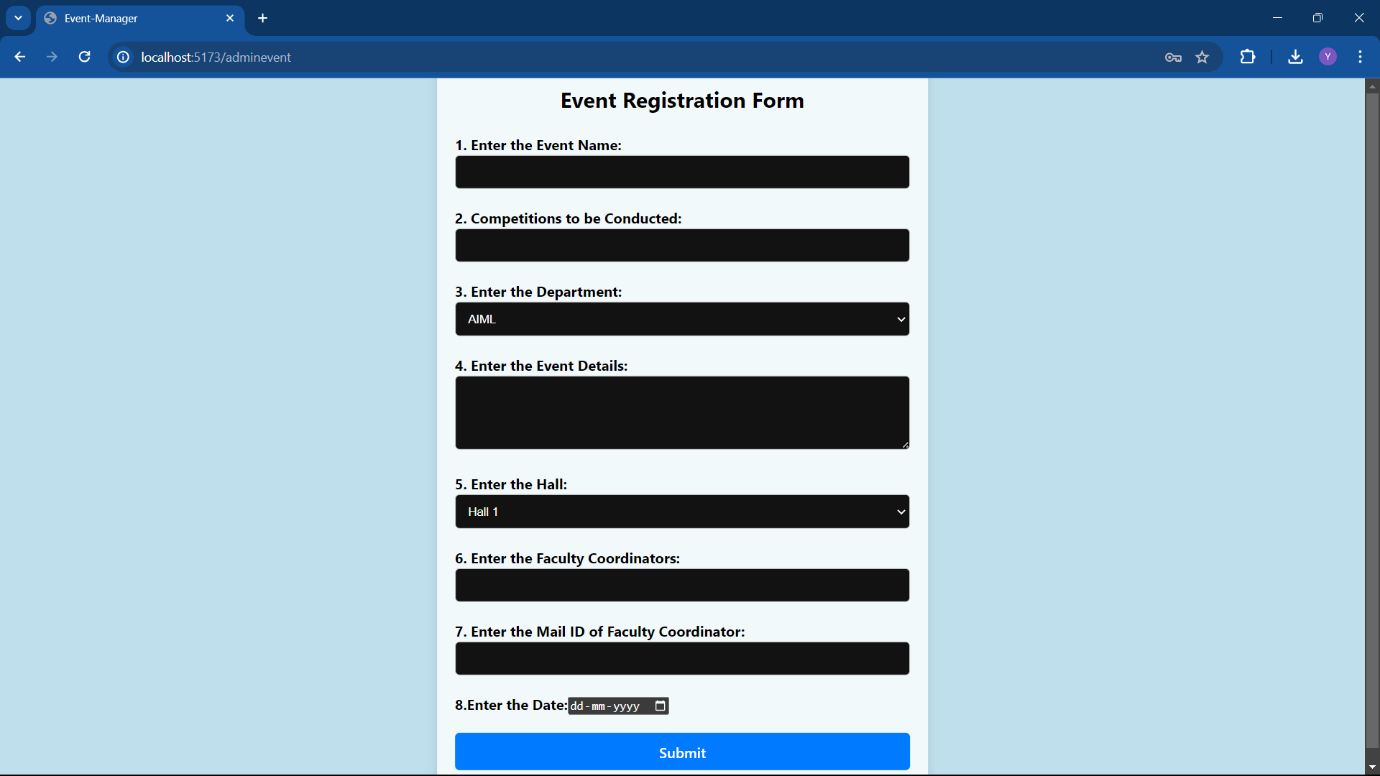
**UI DESIGN:**

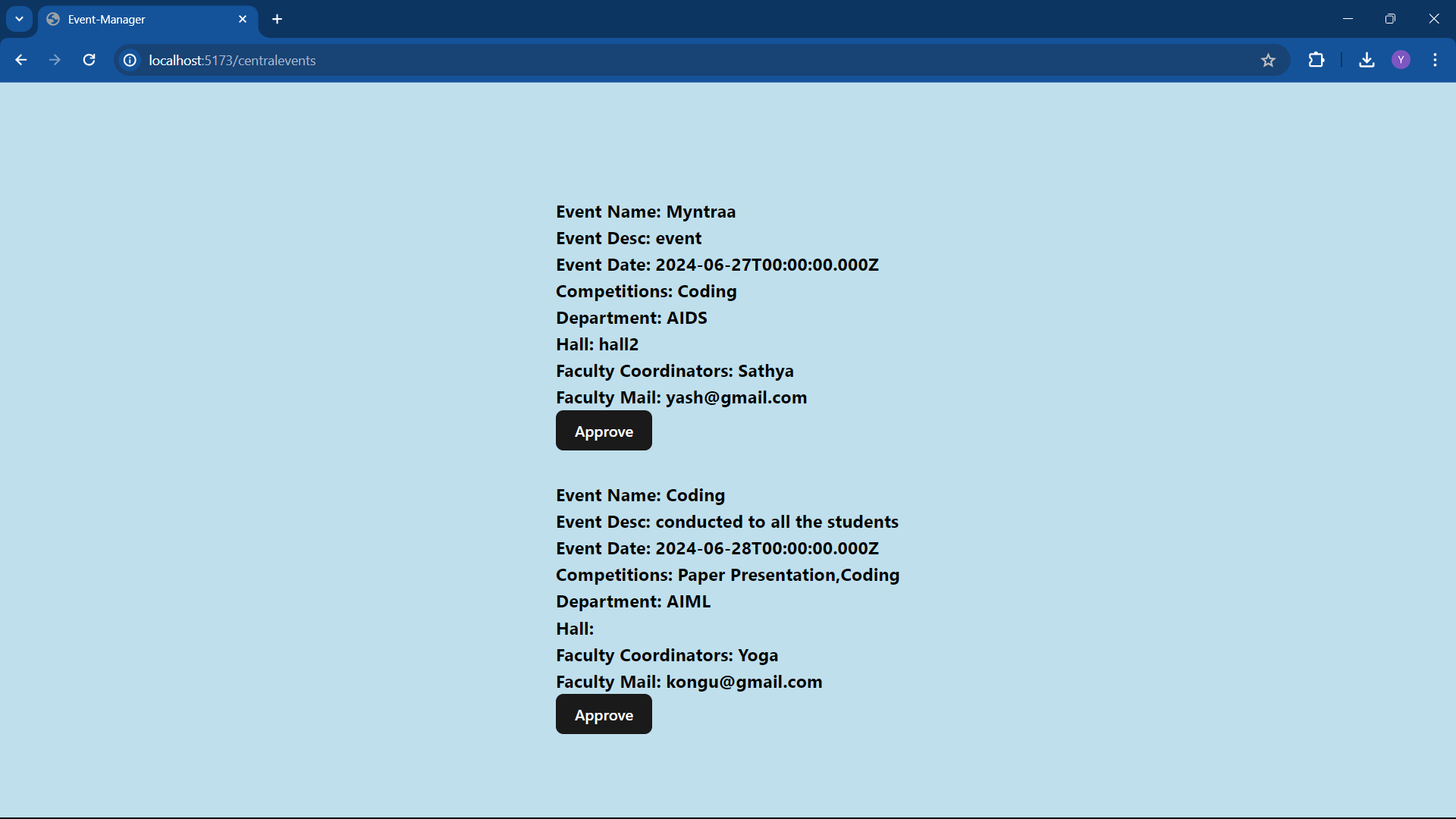
****

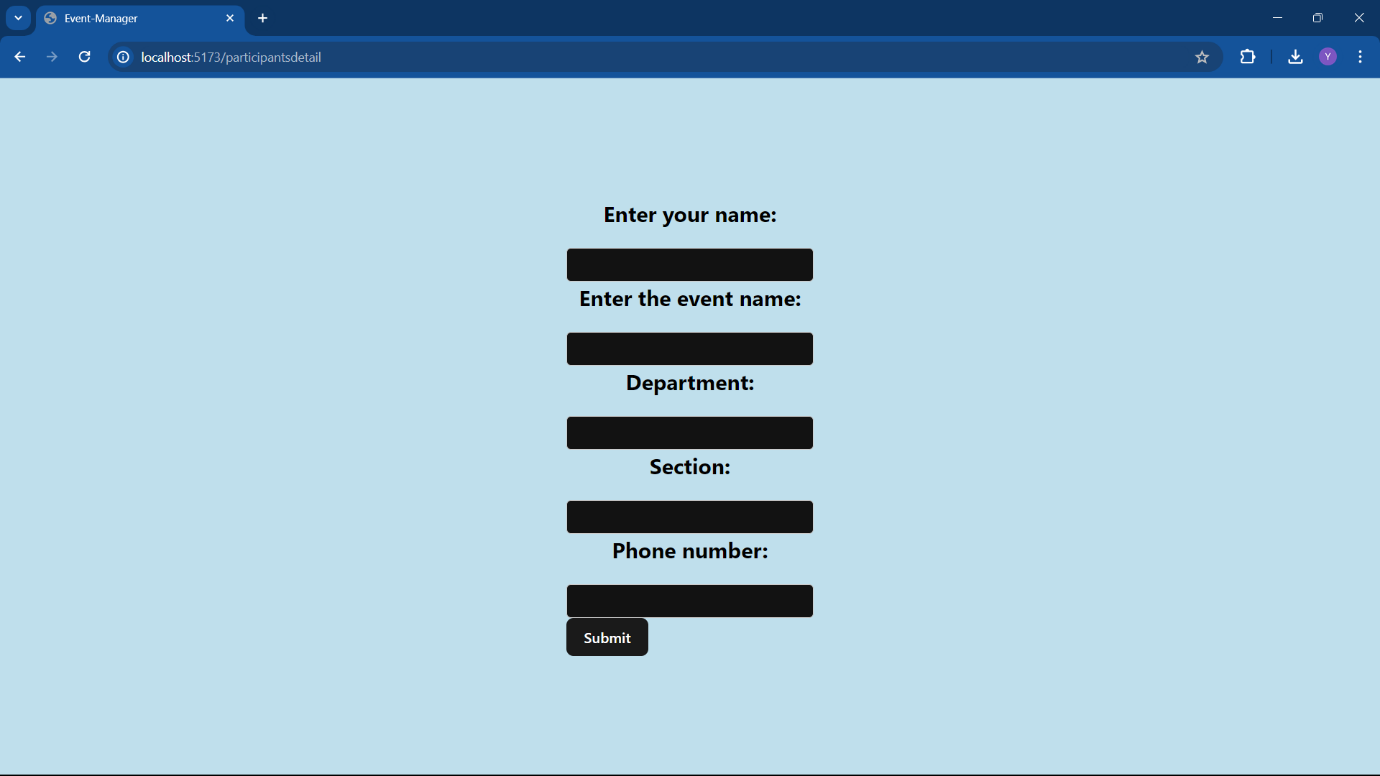
****

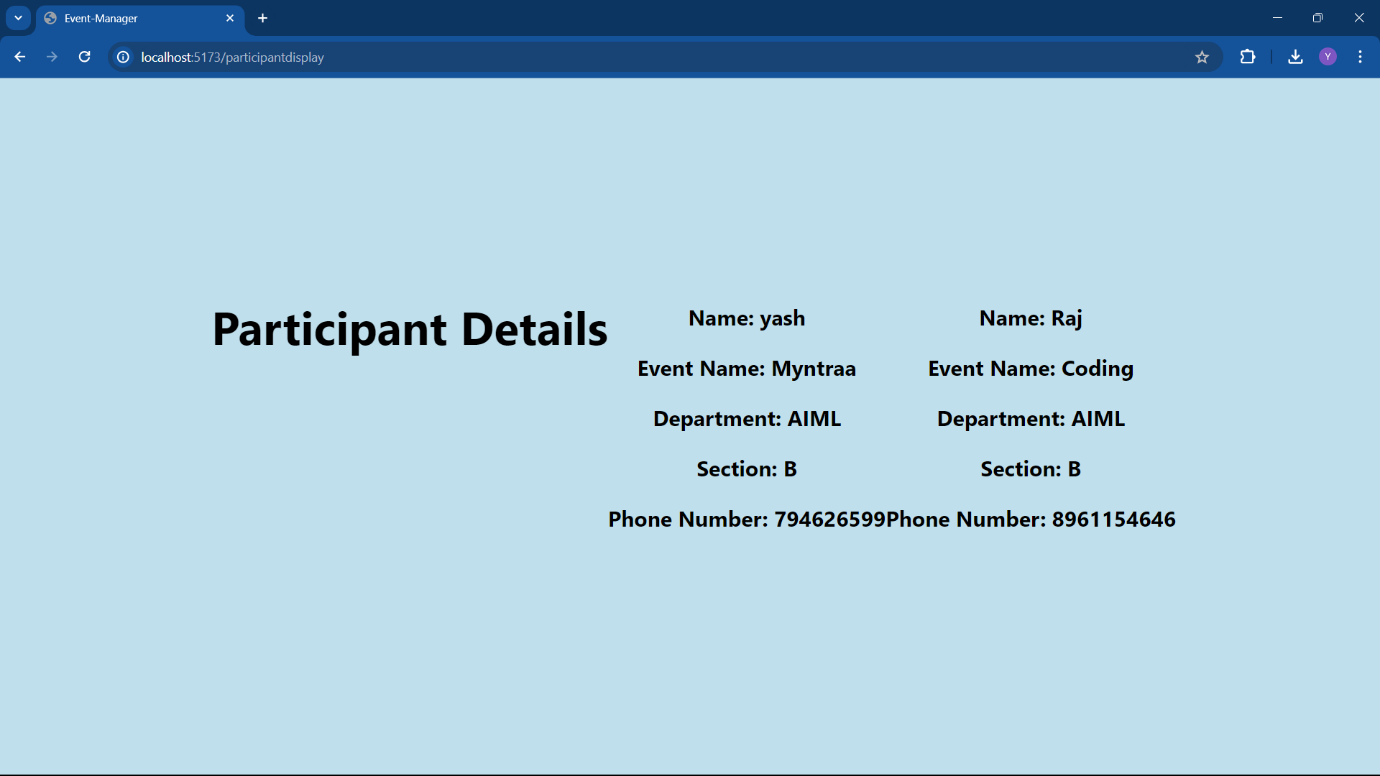
****

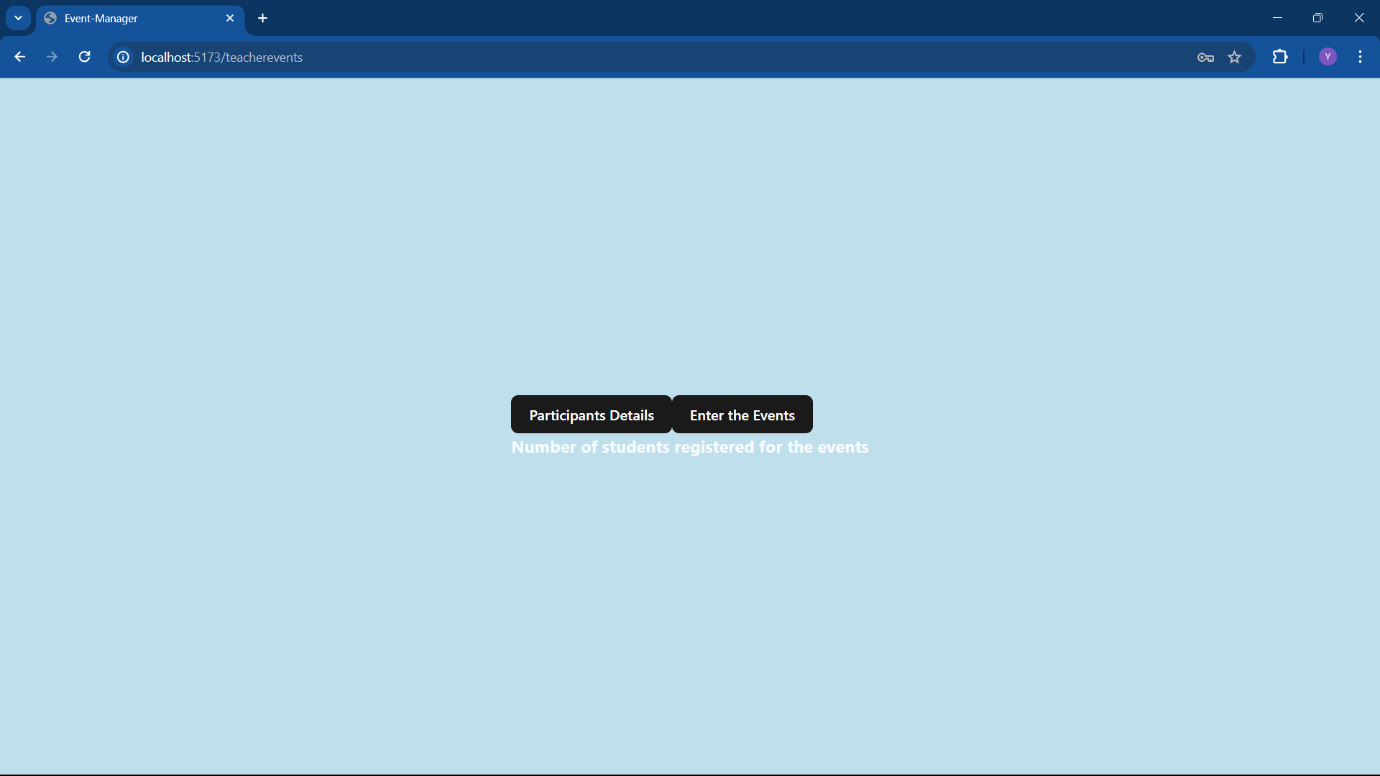
****

****

****

****

****

****

**SAMPLE CODING:**

**import { useState } from 'react'**

**import reactLogo from './assets/react.svg'**

**import viteLogo from '/vite.svg'**

**import './App.css'**

**import Home from '../components/Home.jsx'**

**import { BrowserRouter as Router,Routes,Route } from 'react-router-dom'**

**import StudentLogin from '../components/StudentLogin.jsx'**

**import TeachLogin from '../components/TeachLogin.jsx'**

**import EventDetails from '../pages/EventDetail.jsx'**

**import ParticipantsDetail from '../pages/ParticipantsDetail.jsx'**

**import Register from '../pages/Register.jsx'**

**import TeachEvents from '../pages/TeacherEvents.jsx'**

**import ParticipantDisplay from '../pages/ParticipantDisplay.jsx'**

**import AdminEvent from '../pages/AdminEvent.jsx'**

**import CentralLogin from '../components/Central.jsx'**

**import CentralEvent from '../pages/CentralEvent.jsx'**

**function App() {**

**return (**

**<div>**

**<Router>**

**<Routes>**

**<Route element={<Home/>} path='/'/>**

**<Route element={<StudentLogin/>} path='/studentlogin' />**

**<Route element={<TeachLogin/>} path='/teachlogin' />**

**<Route element={<EventDetails/>}path='/eventdetails'/>**

**<Route element={<ParticipantsDetail/>} path='/participantsdetail'/>**

**<Route element={<Register/>} path='/register'/>**

**<Route element={<TeachEvents/>} path='/teacherevents'/>**

**<Route element={<ParticipantDisplay/>} path='participantdisplay' />**

**<Route element={<AdminEvent/>} path='adminevent' />**

**<Route element={<CentralLogin/>} path='centrallogin'/>**

**<Route element={<CentralEvent/>} path='centralevents'/>**

**</Routes>**

**</Router>**

**</div>**

**)**

**}**

**export default App**

**import { useRef, useState, useEffect } from "react";**

**import { useNavigate } from "react-router-dom";**

**import TeachEvents from "../pages/TeacherEvents"**

**import './Login.css'**

**import axios from "axios";**

**const CentralLogin =()=>{**

**const userRef = useRef();**

**const errRef = useRef();**

**const [username, setUsername] = useState("");**

**const [password, setPassword] = useState("");**

**const [success, setSuccess] = useState(false);**

**const [error, setError] = useState("");**

**const navigate = useNavigate();**

**useEffect(() => {**

**userRef.current.focus();**

**}, []);**

**useEffect(() => {**

**setError("");**

**}, [username, password]);**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**try {**

**const response = await axios.post("http://localhost:8000/login", {**

**username,**

**password,**

**});**

**console.log(response);**

**if (response) {**

**console.log(response);**

**navigate("/centralevents");**

**}**

**} catch (err) {**

**console.log(err);**

**}**

**};**

**return(**

**<div>**

**<body>**

**<div class="signin">**

**<form onSubmit={handleSubmit}>**

**<div class="content">**

**<div class="form">**

**<div class="inputBox">**

**<label htmlFor="username" className="name">UserName</label>**

**<input**

**type="Text"**

**className="InputField"**

**placeHolder="Enter Username here.."**

**ref={userRef}**

**onChange={(e) => setUsername(e.target.value)}**

**/>**

**</div>**

**<div class="inputBox">**

**<label htmlFor="password" className="name">Password</label>**

**<input**

**type="password"**

**className="InputField"**

**placeHolder="Enter Password here.."**

**onChange={(e) => setPassword(e.target.value)}**

**/>**

**</div>**

**<div class="inputBox">**

**<input type="submit" value="Login" />**

**</div>**

**</div>**

**</div>**

**</form>**

**</div>**

**</section>**

**</body>**

**</div>**

**)**

**}**

**export default CentralLogin**

**import { useRef, useState, useEffect } from "react";**

**import { useNavigate } from "react-router-dom";import TeachEvents from "../pages/TeacherEvents"**

**import './Login.css'**

**import axios from "axios";**

**const Login =()=>{**

**const userRef = useRef();**

**const errRef = useRef();**

**const [username, setUsername] = useState("");**

**const [password, setPassword] = useState("");**

**const [success, setSuccess] = useState(false);**

**const [error, setError] = useState("");**

**const navigate = useNavigate();**

**useEffect(() => {**

**userRef.current.focus();**

**}, []);**

**useEffect(() => {**

**setError("");**

**}, [username, password]);**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**try {**

**const response = await axios.post("http://localhost:8000/login", {**

**username,**

**password,**

**});**

**console.log(response);**

**if (response) {**

**console.log(response);**

**navigate("/eventdetails");**

**}**

**} catch (err) {**

**console.log(err);**

**}**

**};**

**return(**

**<div>**

**<body>**

**<div class="signin">**

**<form onSubmit={handleSubmit}>**

**<div class="content">**

**<div class="form">**

**<div class="inputBox">**

**<h2 style={{color:'black'}}>Student</h2>**

**<label htmlFor="username" className="name">UserName</label>**

**<input**

**type="Text"**

**className="InputField"**

**placeHolder="Enter Username here.."**

**ref={userRef}**

**onChange={(e) => setUsername(e.target.value)}**

**/>**

**</div>**

**<div class="inputBox">**

**<label htmlFor="password" className="name">Password</label>**

**<input**

**type="password"**

**className="InputField"**

**placeHolder="Enter Password here.."**

**onChange={(e) => setPassword(e.target.value)}**

**/>**

**</div>**

**<div class="inputBox">**

**<input type="submit" value="Login" />**

**</div>**

**</div>**

**</div>**

**</form>**

**</div>**

**</section>**

**</body>**

**</div>**

**)**

**}**

**export default Login**

**import { useState,useEffect } from "react";**

**import axios from "axios";**

**import './AdminEvent.css'**

**const AdminEvent = ()=>{**

**const [eventName, setEventName] = useState('');**

**const [competitions, setCompetitions] = useState('');**

**const [department, setDepartment] = useState('');**

**const [eventDesc, setEventDesc] = useState('');**

**const [eventDate, setEventDate] = useState('');**

**const [hall, setHall] = useState('');**

**const [facultyCoordinators, setFacultyCoordinators] = useState('');**

**const [facultyMail, setFacultyMail] = useState('');**

**const handleSubmit = async(e)=>{**

**e.preventDefault();**

**const ApproveStatus = 'FALSE'**

**try{**

**const response = await axios.post('http://localhost:8000/eventdetails',{**

**eventName,**

**eventDesc,**

**eventDate,**

**competitions,**

**department,**

**hall,**

**facultyCoordinators,**

**ApproveStatus,**

**facultyMail,**

**})**

**console.log(response);**

**}**

**catch (error) {**

**console.error('Error saving event details:', error);**

**}**

**}**

**return(**

**<body class="adminbody">**

**<div class="container">**

**<h2 style={{color:'black'}}>Event Registration Form</h2>**

**<form action="#" method="post">**

**<div class="form-group">**

**<label for="eventName">1. Enter the Event Name:</label>**

**<input type="text" id="eventName" name="eventName" required onChange={(e)=>setEventName(e.target.value)}/>**

**</div>**

**<div class="form-group">**

**<label for="competitions">2. Competitions to be Conducted:</label>**

**<input type="text" id="competitions" name="competitions" required onChange={(e)=>setCompetitions(e.target.value)}/>**

**</div>**

**<div class="form-group">**

**<label for="department">3. Enter the Department:</label>**

**<select id="department" name="department" required onChange={(e)=>setDepartment(e.target.value)}>**

**<option value="AIML">AIML</option>**

**<option value="AIDS">AIDS</option>**

**<option value="CSE">CSE</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="eventDetails">4. Enter the Event Details:</label>**

**<textarea id="eventDetails" name="eventDetails" rows="4" required onChange={(e)=>setEventDesc(e.target.value)}/>**

**</div>**

**<div class="form-group">**

**<label for="hall">5. Enter the Hall:</label>**

**<select id="hall" name="hall" required onChange={(e)=>setHall(e.target.value)}>**

**<option value="hall1">Hall 1</option>**

**<option value="hall2">Hall 2</option>**

**<option value="hall3">Hall 3</option>**

**</select>**

**</div>**

**<div class="form-group">**

**<label for="coordinators">6. Enter the Faculty Coordinators:</label>**

**<input type="text" id="coordinators" name="coordinators" required onChange={(e)=>setFacultyCoordinators(e.target.value)}/>**

**</div>**

**<div class="form-group">**

**<label for="coordinatorEmail">7. Enter the Mail ID of Faculty Coordinator:</label>**

**<input type="email" id="coordinatorEmail" name="coordinatorEmail" required onChange={(e)=>setFacultyMail(e.target.value)}/>**

**</div>**

**<div className="form-group">**

**<label >8.Enter the Date:</label>**

**<input type="date" onChange={(e)=>setEventDate(e.target.value)}/>**

**</div>**

**<button className="adminbutton" type="submit" onClick={handleSubmit}>Submit</button>**

**</form>**

**</div>**

**</body>**

**)**

**}**

**export default AdminEvent**

**import React, { useState, useEffect } from "react";**

**import axios from "axios";**

**import './Central.css'**

**const CentralEvent = () => {**

**const [events, setEvents] = useState([]);**

**useEffect(() => {**

**fetchData();**

**}, []);**

**const fetchData = async () => {**

**try {**

**const response = await axios.get("http://localhost:8000/eventdetails");**

**const notapprovedData =response.data.filter(event=>event.ApproveStatus!=='true')**

**setEvents(notapprovedData);**

**console.log(notapprovedData);**

**} catch (error) {**

**console.error("Error in fetching event details", error);**

**}22**

**};**

**const handleApprove = async (eventId) => {**

**try {**

**await axios.put(`http://localhost:8000/approve-event/${eventId}`);**

**setEvents(prevEvents => prevEvents.filter(event => event.\_id !== eventId));**

**} catch (error) {**

**console.error('Error approving event:', error);**

**}**

**};**

**return (**

**<div>**

**{events.map(event => (**

**<div key={event.\_id} class="prp" style={{marginTop:'2rem'}}>**

**<h3>Event Name: {event.eventName}</h3>**

**<h3>Event Desc: {event.eventDesc}</h3>**

**<h3>Event Date: {event.eventDate}</h3>**

**<h3>Competitions: {event.competitions}</h3>**

**<h3>Department: {event.department}</h3>**

**<h3>Hall: {event.hall}</h3>**

**<h3>Faculty Coordinators: {event.facultyCoordinators}</h3>**

**<h3>Faculty Mail: {event.facultyMail}</h3>**

**<button class="rtm" onClick={() => handleApprove(event.\_id)}>Approve</button><br />**

**</div>**

**))}**

**</div>**

**);**

**};**

**export default CentralEvent;**

**// ParticipantDetailsPage.js**

**import React, { useState, useEffect } from 'react';**

**import axios from 'axios';**

**import './edit.css'**

**const ParticipantDisplay = () => {**

**const [participants, setParticipants] = useState([]);**

**useEffect(() => {**

**fetchData();**

**}, []);**

**const fetchData = async () => {**

**try {**

**const response = await axios.get('http://localhost:8000/participants');**

**setParticipants(response.data);**

**} catch (error) {**

**console.error('Error fetching participant details:', error);**

**}**

**};**

**return (**

**<div class="yash">**

**<h1>Participant Details</h1>**

**{participants.map(participant => (**

**<div key={participant.\_id}>**

**<h2 >Name: {participant.name}</h2>**

**<h2>Event Name: {participant.eventName}</h2>**

**<h2>Department: {participant.department}</h2>**

**<h2>Section: {participant.section}</h2>**

**<h2>Phone Number: {participant.phoneNumber}</h2>**

**</div>**

**))}**

**</div>**

**);**

**};**

**export default ParticipantDisplay;**

**CONCLUSION:**

In conclusion, an Event Management System (EMS) is a comprehensive software solution that plays a crucial role in transforming the management and execution of events. By digitizing and automating administrative and operational processes, an EMS improves the efficiency, accuracy, and quality of event management and also ensures the security of data. With features such as user logins for central, student, and event organizers, event creation and approval, participant management, calendar integration, secure login, and real-time notifications, an EMS streamlines workflows, reduces manual tasks, enhances event coordination, and promotes better engagement among participants.

Implementing an EMS brings numerous benefits, including improved operational efficiency, reduced errors, optimized resource utilization, enhanced participant experience, and increased satisfaction for all stakeholders involved. It empowers event organizers to focus more on delivering successful events while the system takes care of administrative tasks and data management. Overall, an EMS revolutionizes the way events are managed by leveraging technology to streamline processes, improve decision-making, and drive better outcomes. It serves as a central platform for managing event information, participant data, and operational details, ultimately leading to more efficient and effective event execution.