Software Engineering and Web Technologies Laboratory

Integrated Project on

Departmental Events

Bachelor of Engineering

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Team No: C14

Div: C

Gayatri Betageri	01FE21BCS029	303
Gauri Thambkar	01FE21BCS098	312
Krishna Hasaraddi	01FE21BCS100	313
Aishwarya Patil	01FE21BCS112	315

Faculty In charges

Mrs. Indira B, Mr. Prashanth Naryankar

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

HUBLI–580 031 (India).

Academic year 2023-24

Table of Content

Chapter 1
INTRODUCTION

1.1 Preamble

Our project is centered around the development of a dedicated module aimed at enhancing the efficiency of managing departmental events. Our primary focus is to create a user-friendly interface that enables event organizers to seamlessly input and edit event details. This involves the design of intuitive data entry forms, ensuring precision and significantly reducing the time and effort required for maintaining comprehensive event profiles.

A critical aspect of our project is our unwavering commitment to timely data retrieval and storage. The system ensures that edited data is promptly saved, encouraging responsive user interaction. Additionally, we introduce a dynamic frontend to our module, enhancing the user experience for students. This feature enables students to explore past events, providing insights into the rich history of our department's activities. Furthermore, a user-friendly registration system empowers students to register for upcoming events, contributing to increased participation and engagement.

Our incorporation of front-end features not only enhances the functionality of our module but also adds a significant boost to the department's profile. The website becomes a vibrant platform where students actively participate in and contribute to events, elevating the department's visibility. In summary, our module represents a substantial leap forward in the realm of departmental event management, contributing to a more efficient and engaging environment. This project successfully implements functionalities for displaying, adding, and editing events, thereby improving the overall management of departmental events and enhancing the overall appeal of our institutional website.

1.2 Problem Definition

Problem Definition: To develop a centralized platform that facilitates the effective management of departmental events.

Project Description: Our project revolves around the development of a comprehensive website dedicated to optimizing the management of departmental events. This initiative aims to provide a user-friendly interface where event organizers can efficiently plan, organize, and update event details. The website will offer intuitive features for seamless registration and participation tracking, allowing students and faculty to engage effortlessly with various events within the department.

Key functionalities include a dynamic front-end that enables users to explore past events, gaining insights into the department's rich event history. Additionally, a user-friendly registration system empowers participants to register easily for upcoming events, promoting increased engagement and interaction. The website's design prioritizes accessibility and responsiveness, ensuring a positive user experience across different devices.

Administrators will benefit from tools that facilitate the swift retrieval of specific event information, contributing to a collaborative event data management environment. The project's overarching goal is to enhance the efficiency and engagement associated with departmental events, ultimately contributing to a more vibrant and connected academic community. In summary, our website development project focuses on creating an innovative platform that simplifies the planning, registration, and participation processes, providing a valuable resource for both organizers and attendees in managing departmental events.

1.3 Objectives

- A user-friendly platform for event organizers to manage event details, summaries, and photo uploads.
- Design a user-friendly registration system that empowers participants to easily register for upcoming events, promoting increased participation and interaction within the department.
- Create a dynamic front-end that enables users to explore past events, providing insights into the department's event history and fostering a sense of continuity and engagement
- Implement an alert system to notify users about upcoming events and registration deadlines.
- Facilitate the uploading and archiving of event photos for future reference.
- Improving participant engagement and satisfaction through user-friendly interfaces and communication tools.
- Design the website with scalability in mind, allowing it to adapt and accommodate the growing needs of the department and an increasing number of events.
- Implement robust security measures to safeguard sensitive event information, ensuring the confidentiality and integrity of data stored on the platform.

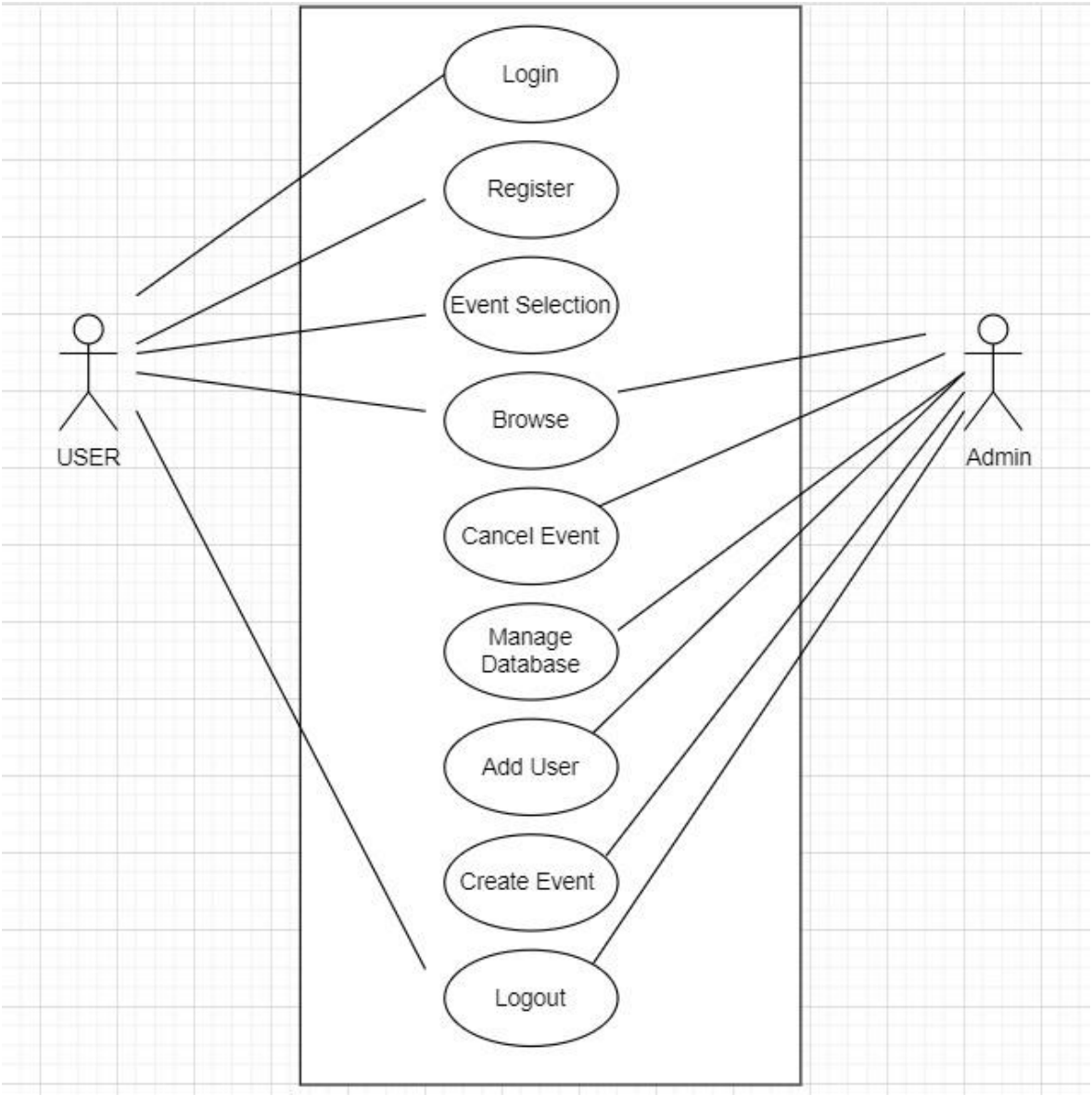
Chapter 2

SOFTWARE REQUIREMENT SPECIFICATIONS

2.1.1 Functional Requirements

- The organizer should be able to create and edit event summaries and upload multiple photos associated with each event.
- The system should display a list of upcoming events, including relevant details such as date, time, and location.
- Participants should be able to register for events, and organizers should have access to a list of registered students.
- A centralized dashboard should provide an overview of past events, event summaries, uploaded photos, upcoming events, and registration status.
- The system should support user registration and login. It should also differentiate between different roles with corresponding permissions.
- The system should send alerts when participants register for a competition.

2.1.2 Use Case Diagram



2.2 Non-Functional Requirements

- Participants must be able to view the home page of the application within 5 seconds after successfully logging in.
- The system should retrieve data within 10 seconds, ensuring efficient and timely access to information.
- The system should save edited data within 2 seconds, facilitating swift updates and ensuring seamless user interaction.
- The system should be able to add the details within 5 seconds to the database.
- The system should load event summaries and photos efficiently, even with a significant number of entries.
- The system should strictly enforce role-based access control, ensuring that each user can only access functionalities and data relevant to their assigned role.
- The system should be able to handle a growing number of events, photos, and registered participants without significant degradation in performance.
- The system should reliably store and retrieve event summaries, photos, and registration data without loss or corruption.
- The user interface should be intuitive, with clear navigation and user-friendly forms for both organizers and participants.
- The system should be compatible with commonly used web browsers.
- Alerts and notifications should be delivered promptly, within 1 minute of a participant registering for a competition.
- The administration login should be hidden from the participants to avoid misuse of their functionalities.

2.3 Hardware and Software Requirements

1. The web application will be built on the MERN (MongoDB, Express, ReactJS, NodeJS) stack.
2. The front end will be built using ReactJS, HTML, CSS, and JavaScript.
3. The backend server will be built using NodeJS and ExpressJS libraries.
4. The backend server will be connected to the MongoDB database.
5. The backend server will have multiple routers – each with a specific URL and method (GET, POST, DELETE, PUT).
6. The front end will send requests to these routers(URL) with a certain request method using Axios.
7. The Mongoose library will be used to create course models and student models will be stored in the MongoDB database.

2.4 Test plan and Test cases

Test Case No.	Test Case Description	Input Values	Predicted Output	Actual Output	Test Case Result
1.	User login with correct credentials	Email: 01fe21bcs098@kletech.ac.in, Password: Gauri3105	Successful login	Successful login	Pass
2.	User login with incorrect password	Email: 01fe21bcs098@kletech.ac.in, Password: person12	Invalid credentials	Invalid credentials	Pass
3.	User login with incorrect email	Email: Email: pson1@gmail.com, Password: Gauri3105	Invalid credentials	Invalid credentials	Pass
4.	Admin login with correct credentials	Email: admin@kletech.ac.in, Password: Admin@123	Successful login	Successful login	Pass
5.	Admin login with incorrect password and incorrect email	Email: persn@gmail.com, Password: person12	Invalid credentials	Invalid credentials	Pass
7.	Admin uploads the event details 1	Academic details 1	Updated successfully	Updated successfully	Pass
8.	The user uploads the registration details 2	Academic details 2	Updated successfully	Updated successfully	Pass
9.	User registering for an event	Name: Gauri Thambkar Email: 01fe21bcs098@kletech.ac.in, USN: 01fe21bcs098 Sem:5	Registered Successfully	Registered Successfully	Pass
10.	User registering for an event	Name: Gauri Thambkar Email: 01fe21bcs098@kletech.ac.in, USN: 01fe21bcs098 Sem:5	Already Registered	Already Registered	Pass

Chapter 3
SYSTEM DESIGN AND IMPLEMENTATION

3.1 MVC Architecture

Client-Server architecture

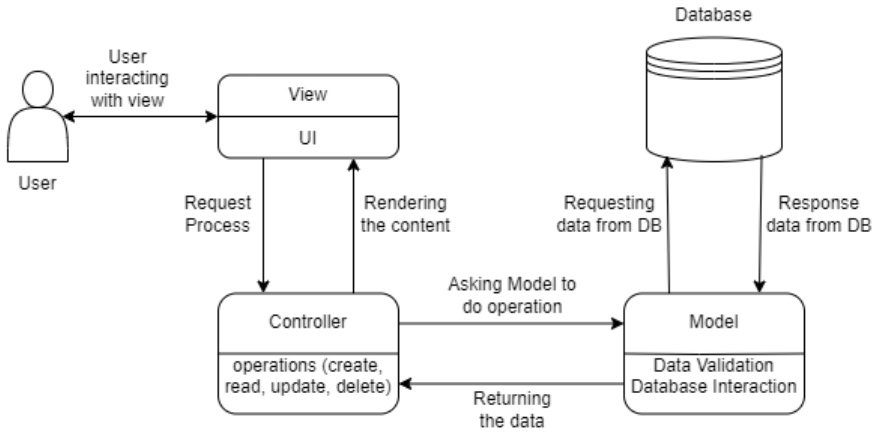


Fig. 3. Client-Server Architecture

The client-server architecture is highly suitable for the Result Analysis web application, providing a robust and scalable framework to meet the specific requirements of managing student grades, generating result summaries, and visualizing performance trends.

Advantages:

1. Separation of concerns: Clear distinction between frontend and backend functionalities.
2. Scalability: Components can be scaled independently based on demand.
3. Security: Critical operations, such as authentication and database access, are centralized and controlled on the server side.
4. Maintainability: Easier to update and maintain different components independently.

3.2 MERN framework

The MERN stack is a popular JavaScript-based framework for building web applications, consisting of four main components: MongoDB, Express.js, React, and Node.js. Each component plays a specific role in the development process, collectively providing a full-stack solution. Let's explore each component in the context of the Result Analysis web application:

1. **MongoDB:**
   MongoDB is well-suited for storing student information, grades, and other academic data. Its flexibility accommodates changes in the data structure, providing scalability for future enhancements.

2. **Express.js:**
   Express.js serves as the backend framework, handling HTTP requests from the client, routing those requests to the appropriate endpoints, and managing communication with the MongoDB database.

3. **React:**
   React is used on the client side to create a responsive and interactive user interface. It facilitates the rendering of result summaries, graphs, and other visual components, providing a seamless user experience.

4. **Node.js:**
   Node.js serves as the server runtime, enabling the execution of server-side logic. It works in conjunction with Express.js to handle incoming HTTP requests, process data, and communicate with the MongoDB database.

3.3 Detailed Database Description (Mongo DB)

Schema's

1. Admin Login Schema:
   - email: Email address of the admin (String, unique, required)
   - password: Admin password (String, required)
2. Events Schema:
   - name: Name of the event (String)
   - date: Date of event (Date)
   - time: Time of the event (String)
3. Student Registration Schema:
   - name: Name of the student (String)
   - email: Email address of the student (String, unique, required)
   - password: Student password (String, required)
   - usn: USN of the student (String)
   - sem: Semester of the student (Number)
4. Event 1 Registration Schema:
   - name: Name of the student (String)
   - email: Reference to Student Registration
   - usn: Reference to Student Registration
   - sem: Reference to Student Registration
5. Event 2 Registration Schema:
   - name: Name of the student (String)
   - email: Reference to Student Registration
   - usn: Reference to Student Registration
   - sem: Reference to Student Registration
6. Event 3 Registration Schema:
   - name: Name of the student (String)
   - email: Reference to Student Registration
   - usn: Reference to Student Registration
   - sem: Reference to Student Registration

7.Event 4 Registration Schema:

- name: Name of the student (String)
- email: Reference to Student Registration
- usn: Reference to Student Registration
- sem: Reference to Student Registration

Operations

Create (POST):

- Route: /insert/${routeName}
- Description: This operation allows the insertion of new data into the specified entity.
- Implementation:
    - The route expects a JSON body representing the new data to be inserted.
    - A new instance of the Mongoose model for the specified entity is created with the provided data.
    - The save() method is then called to persist the new data in the database.
    - If successful, the server responds with a success message.
- Read (GET - All):
- Route: /fetch/${routeName}
- Description: This operation retrieves all records from the specified entity.
- Implementation:
    - The route uses the find() method on the Mongoose model for the specified entity to retrieve all records.
    - The password field is excluded from the response for security reasons.
    - The server responds with a JSON array containing all records.

Read (GET - Filtered):

- Route: /fetch/${routeName}/:keys/:values
- Description: This operation retrieves records from the specified entity based on specified keys and values.
- Implementation:

- The route expects keys and values as URL parameters, which are used to create a query object.
- The find() method is used with the query object to retrieve filtered records.
- The password field is excluded from the response for security reasons.
- The server responds with a JSON array containing the filtered records.

Update (PUT):

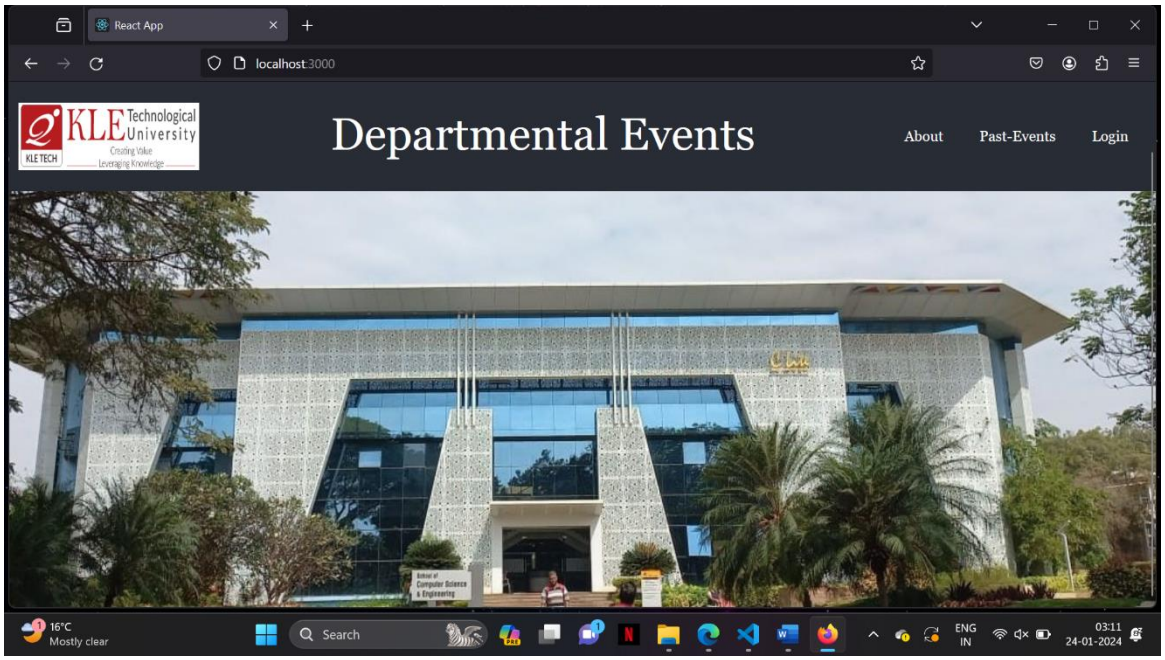
- Route: /update/${routeName}/:keys/:values
- Description: This operation allows updating records in the specified entity based on specified keys and values.
- Implementation:
  - The route expects keys and values as URL parameters to identify the records to be updated.
  - The findOneAndUpdate() method is used with the query object to find and update the specified records.
  - The updated data is sent in the request body.
  - If the update is successful, the server responds with a success message.

Delete (DELETE):

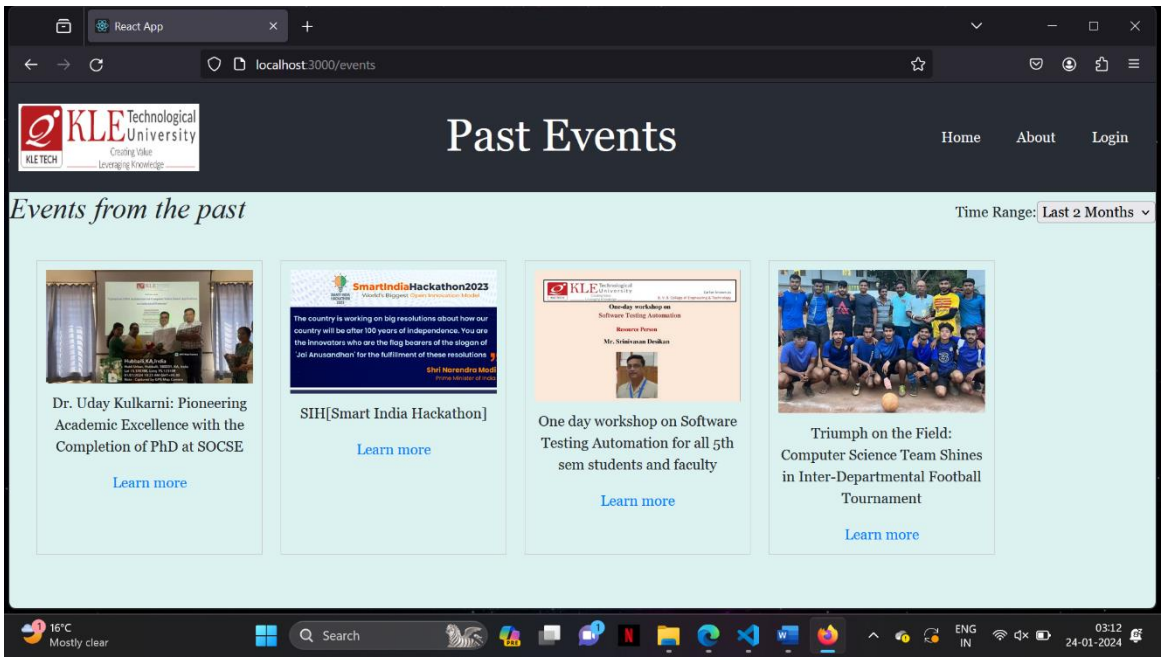
- Route: /delete/${routeName}/:keys/:values
- Description: This operation allows deleting records from the specified entity based on specified keys and values.
- Implementation:
  - The route expects keys and values as URL parameters to identify the records to be deleted.
  - The deleteOne() method is used with the query object to delete the specified records.
  - If records are deleted successfully, the server responds with a success message.

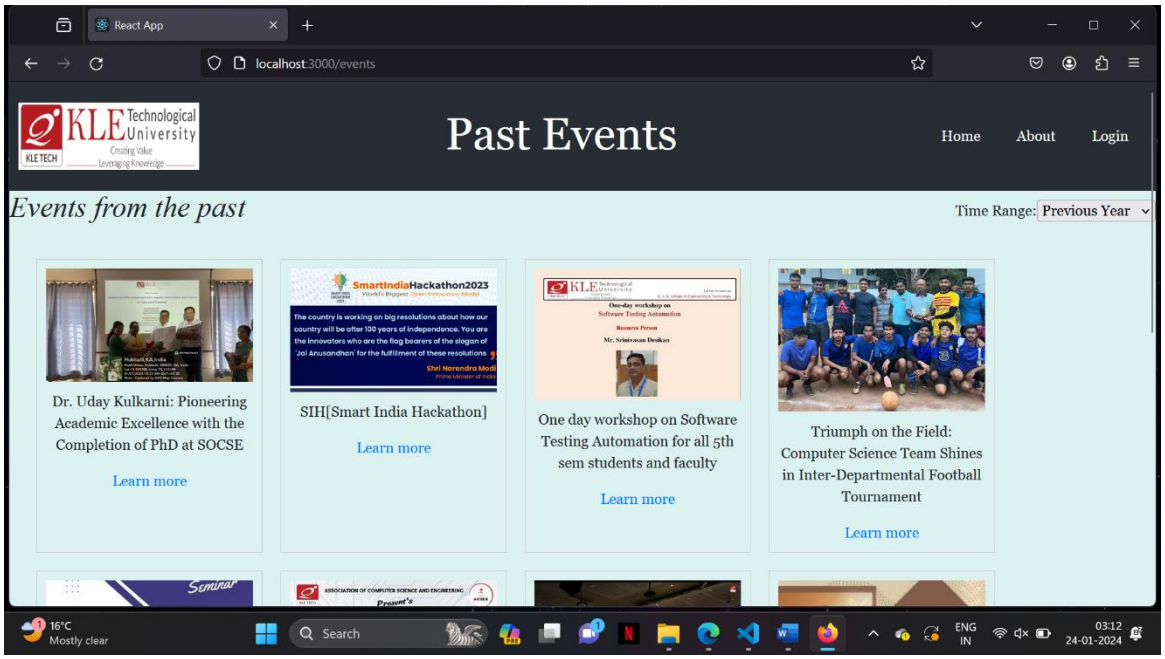
3.4 Modules description

- **Dashboard**

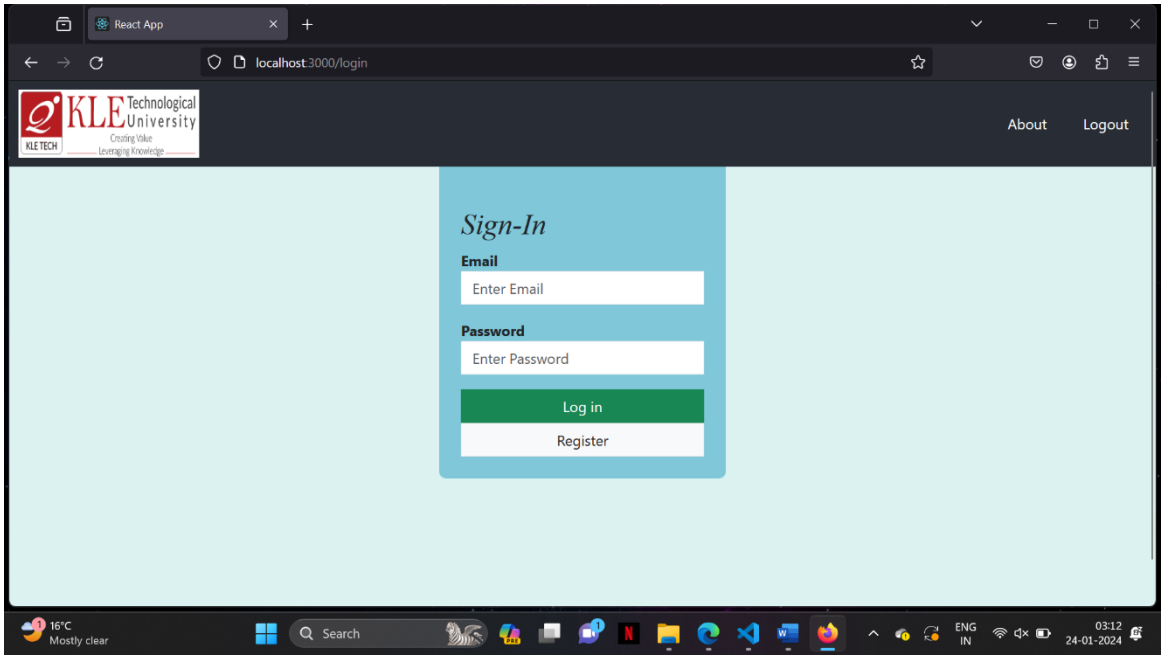


- **Past Events**

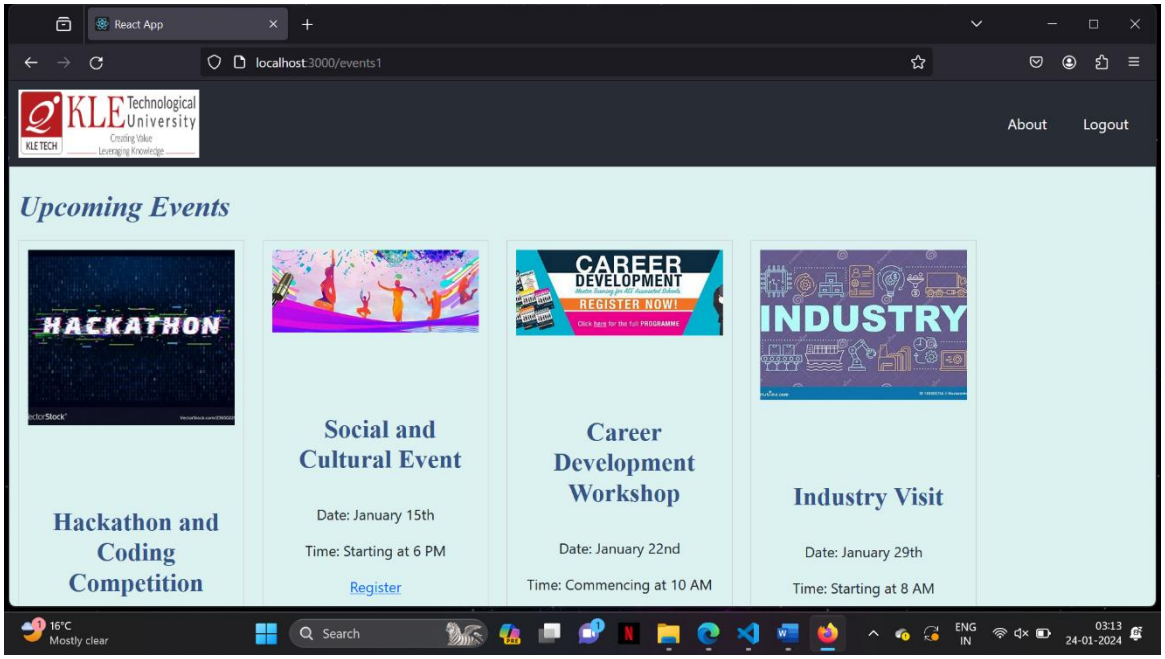



React App
localhost:3000/events
KLE Technological University
Past Events
Home
About
Login
Events from the past
Time Range: Previous Year
Dr. Uday Kulkarni: Pioneering Academic Excellence with the Completion of PhD at SOCSE
Learn more
SmartIndiaHackathon2023
SIH[Smart India Hackathon]
Learn more
One day workshop on Software Testing Automation for all 5th sem students and faculty
Learn more
Triumph on the Field: Computer Science Team Shines in Inter-Departmental Football Tournament
Learn more
16°C
Mostly clear
Search
ENG
IN
03:12
24-01-2024

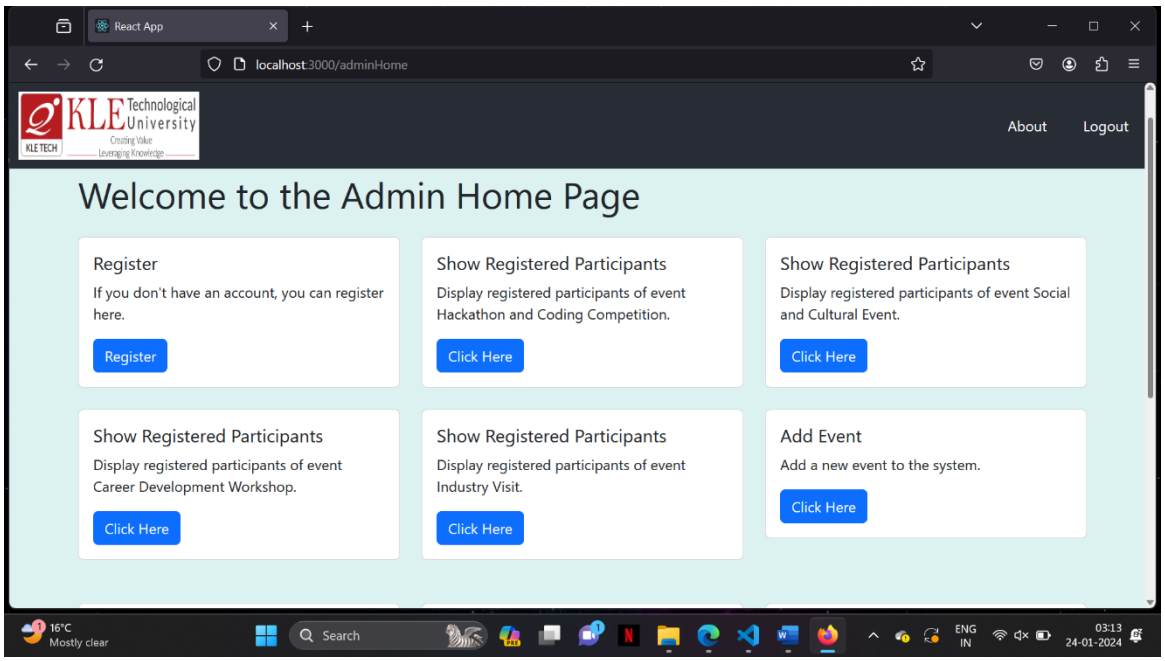
- **Sign-In**

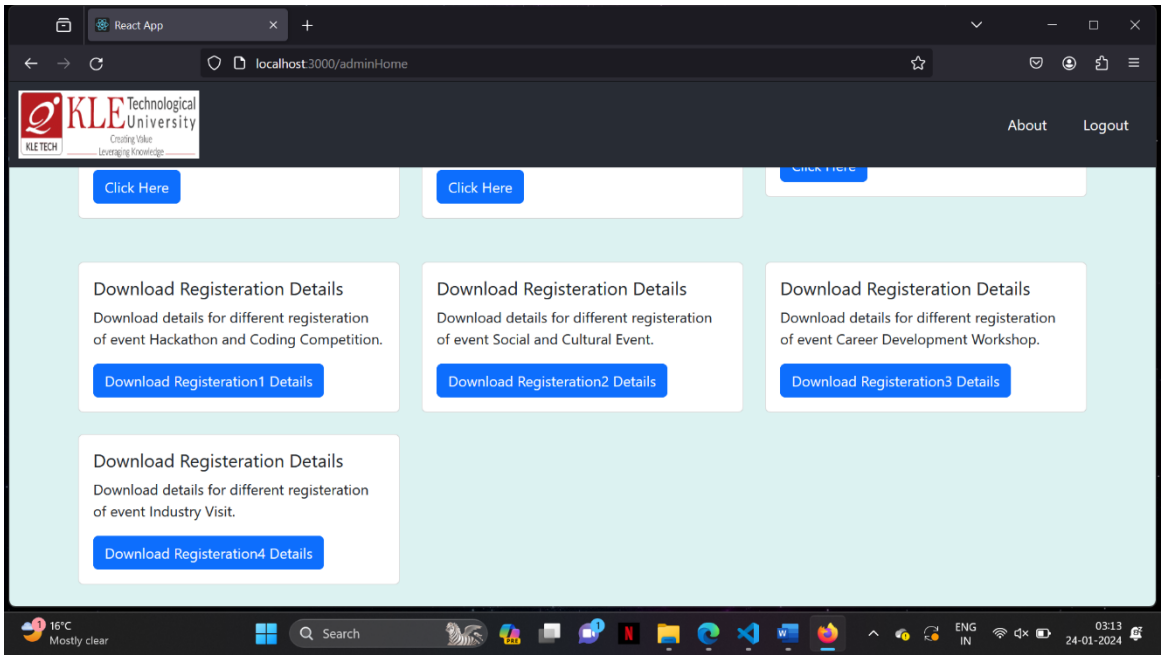

React App
localhost:3000/login
KLE Technological University
About
Logout
Sign-In
Email
Enter Email
Password
Enter Password
Log in
Register
16°C
Mostly clear
Search
ENG
IN
03:12
24-01-2024

- **Upcoming Events**

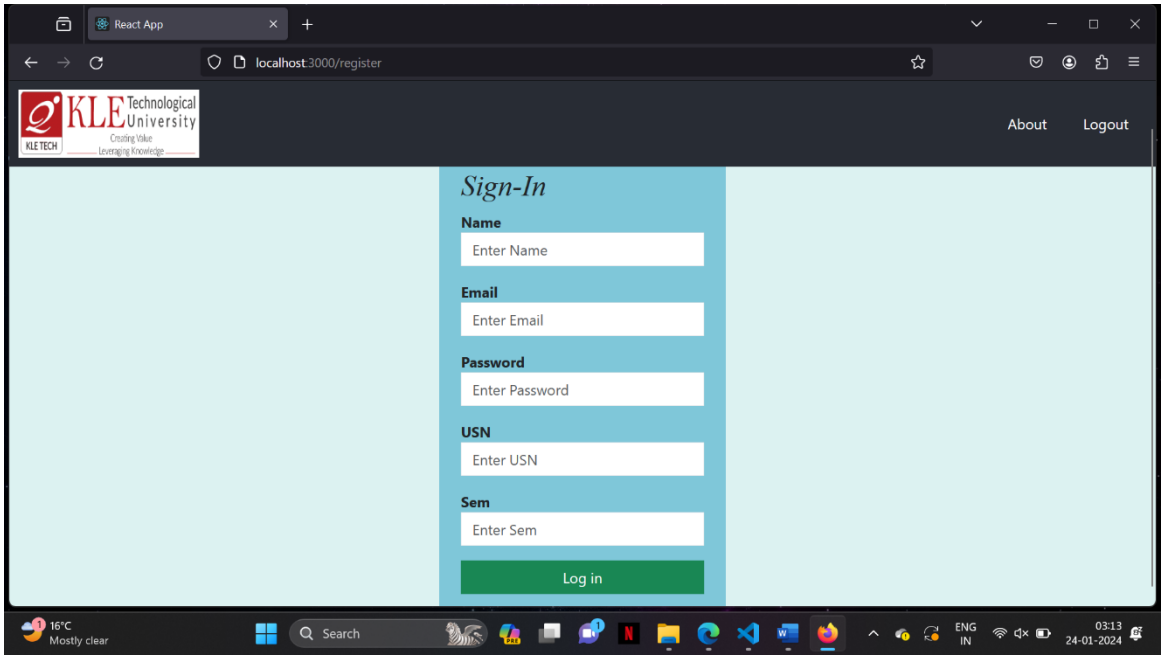


- **Admin Home Page**

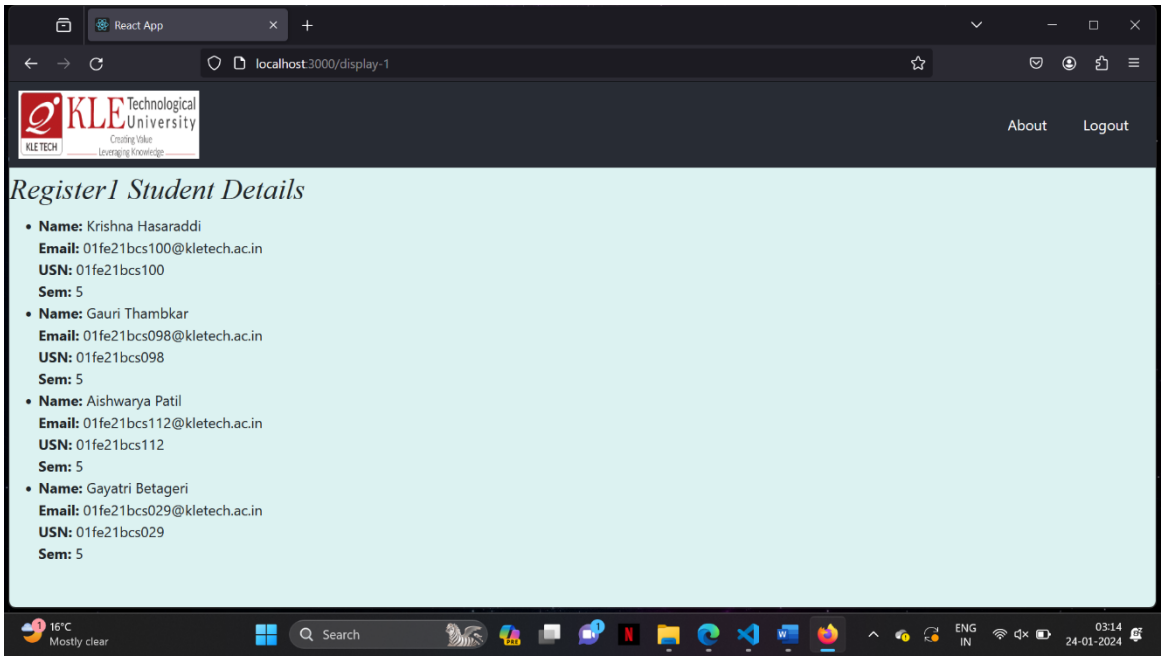



React App
localhost:3000/adminHome
KLE Technological University
About
Logout
Click Here
Click Here
Download Registeration Details
Download details for different registeration of event Hackathon and Coding Competition.
Download Registeration1 Details
Download Registeration Details
Download details for different registeration of event Social and Cultural Event.
Download Registeration2 Details
Download Registeration Details
Download details for different registeration of event Career Development Workshop.
Download Registeration3 Details
Download Registeration Details
Download details for different registeration of event Industry Visit.
Download Registeration4 Details
16°C
Mostly clear
Search
ENG
IN
03:13
24-01-2024

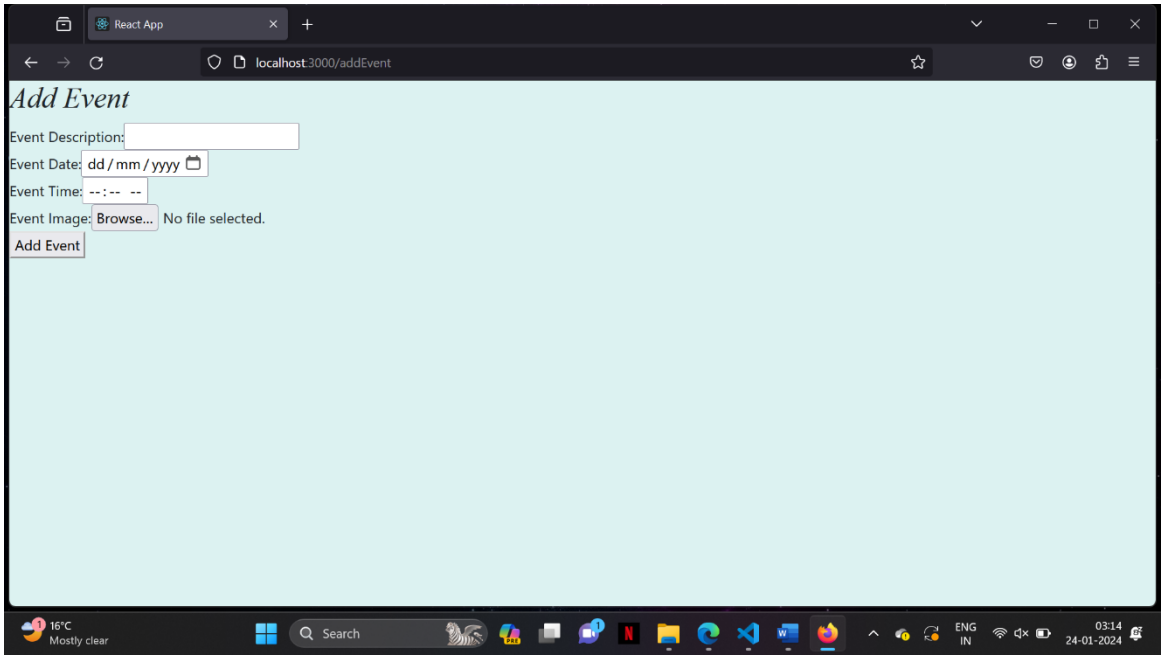
- **Register**


React App
localhost:3000/register
KLE Technological University
About
Logout
Sign-In
Name
Enter Name
Email
Enter Email
Password
Enter Password
USN
Enter USN
Sem
Enter Sem
Log in
16°C
Mostly clear
Search
ENG
IN
03:13
24-01-2024

- **Registered Student Details**



- **Add Event**



Chapter 4
RESULTS AND DISCUSSIONS

4.1 Results/Snapshots with description

A] DISPLAY successful:

The "FetchAllFaculty" component successfully retrieves and displays registered faculty details, dynamically rendering "FacultyCard" components with essential information such as name, email, and role. This streamlined and responsive presentation enhances user accessibility and transparency within the KLETech faculty management system.

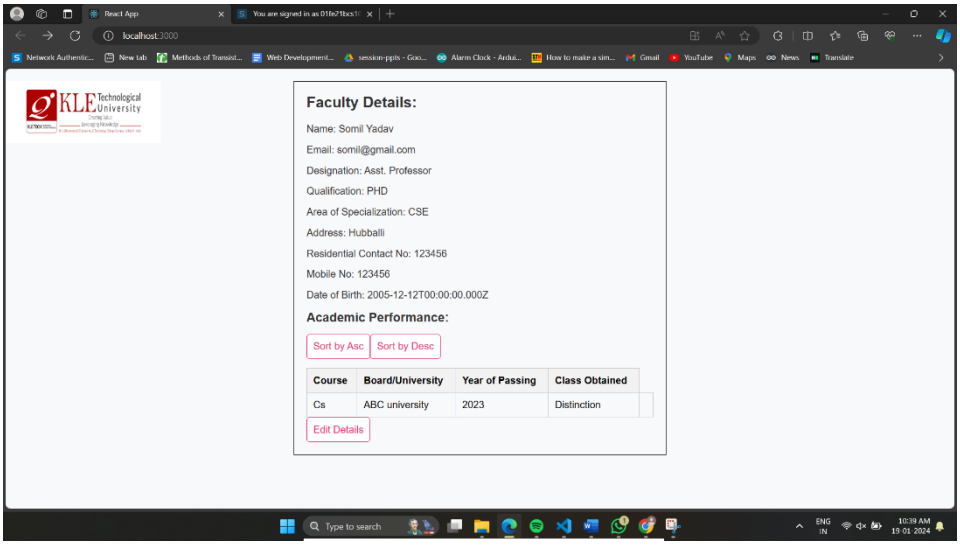


Fig. 9. Display details

B] EDIT successful:

The faculty details editing functionality in the KLETech system seamlessly allows for the modification of personal and academic information. Through user-friendly interfaces and efficient data handling, faculty members can successfully update their profiles, ensuring accurate and up-to-date information within the system.

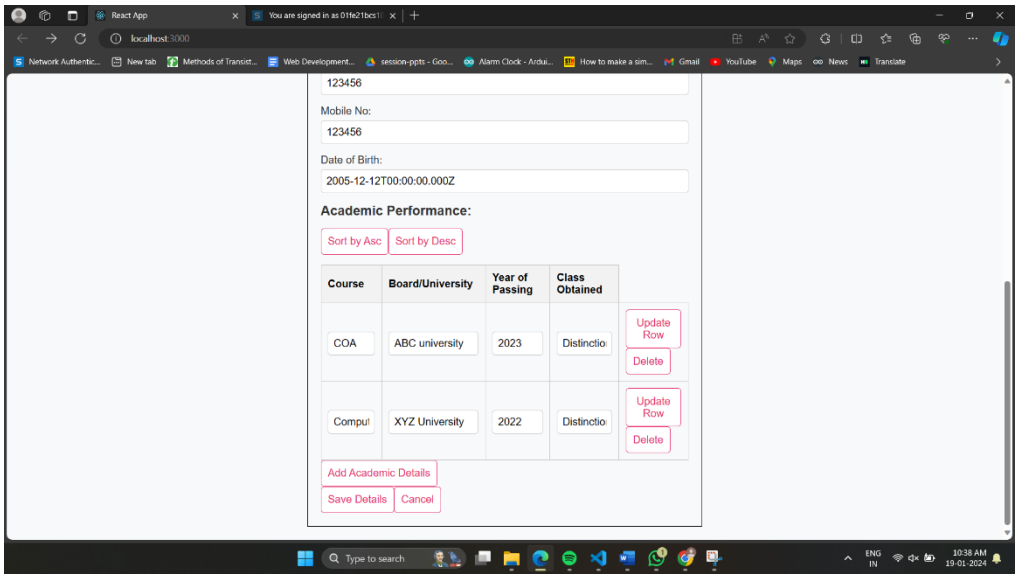


Fig. 10. Edit details

C] ADD successful:

The KLETech system excels in facilitating the seamless addition of details for new faculty members. With an intuitive interface and efficient data entry mechanisms, the system ensures a smooth onboarding process, allowing for the successful incorporation of accurate and comprehensive information for each new user.

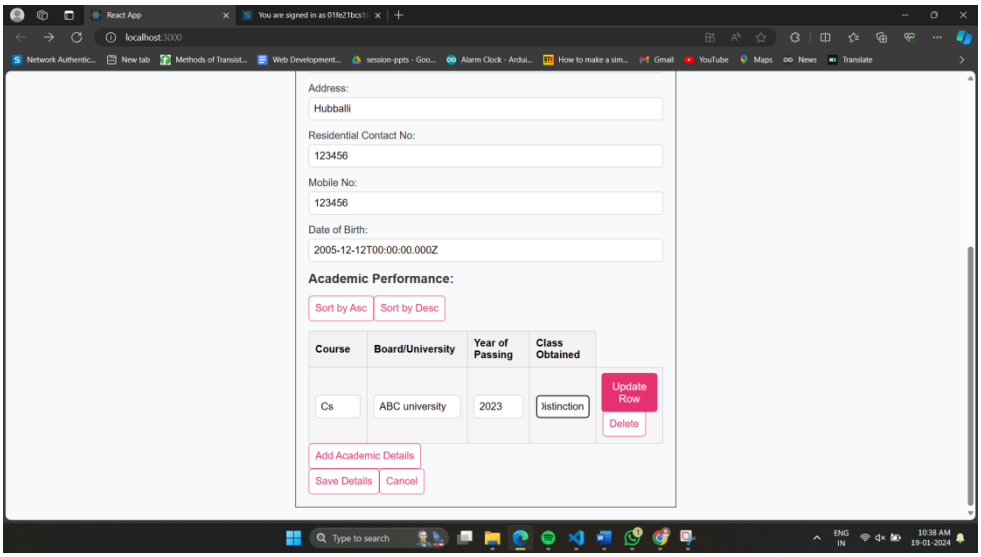


Fig. 11. Add details (a)

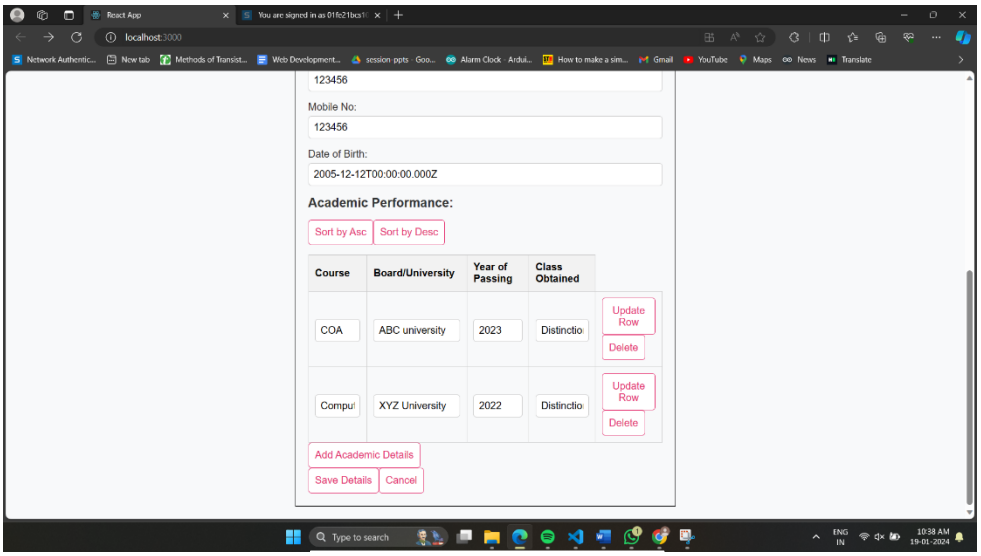
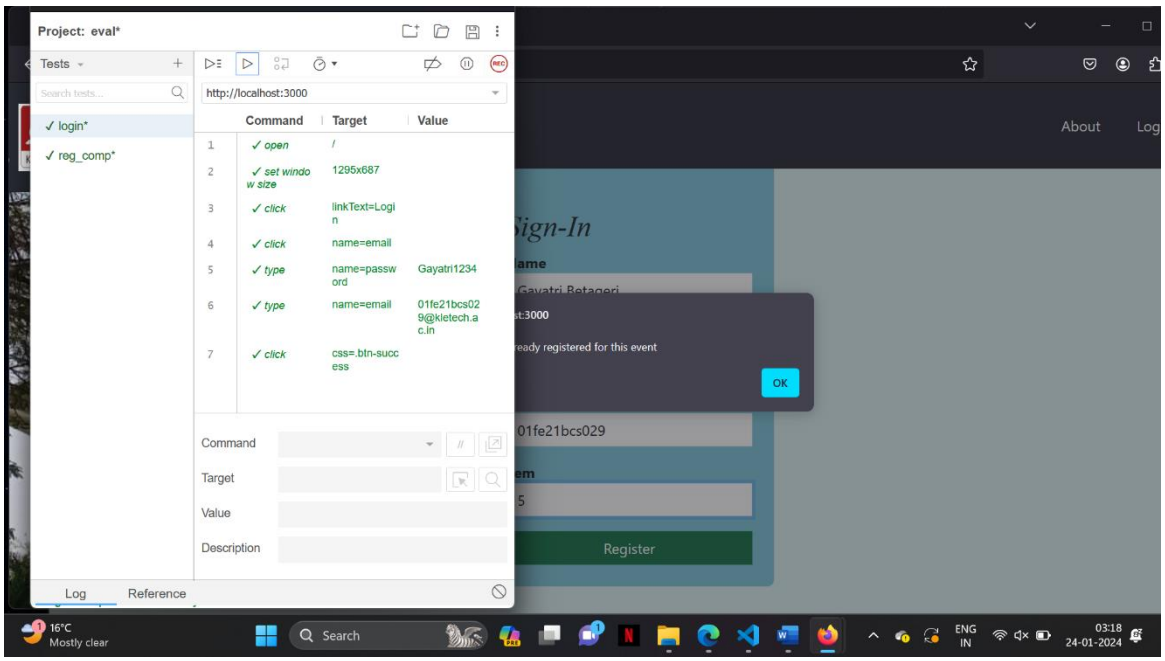


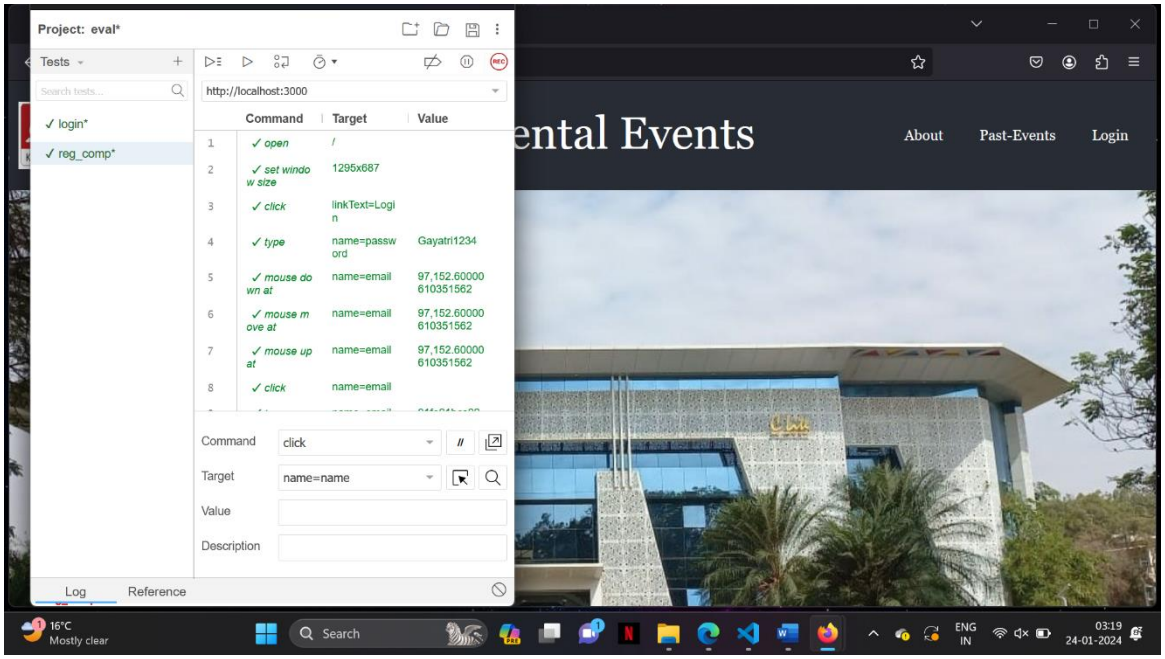
Fig. 11. Add details (b)

4.2 Testing Report

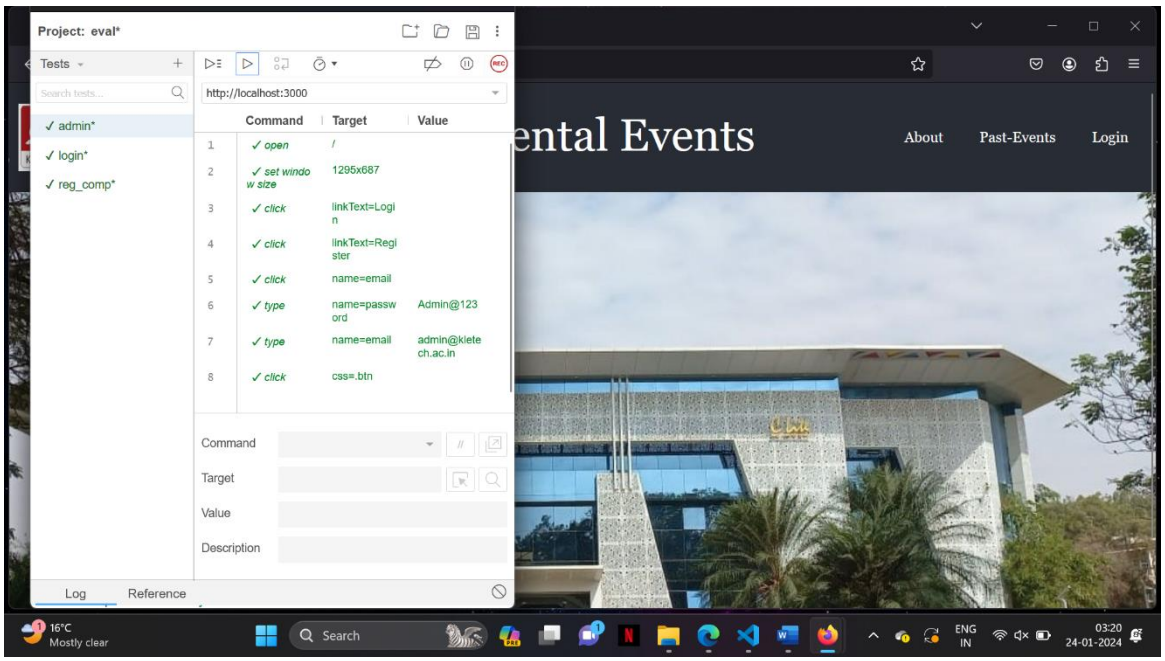
- **Login Testing**

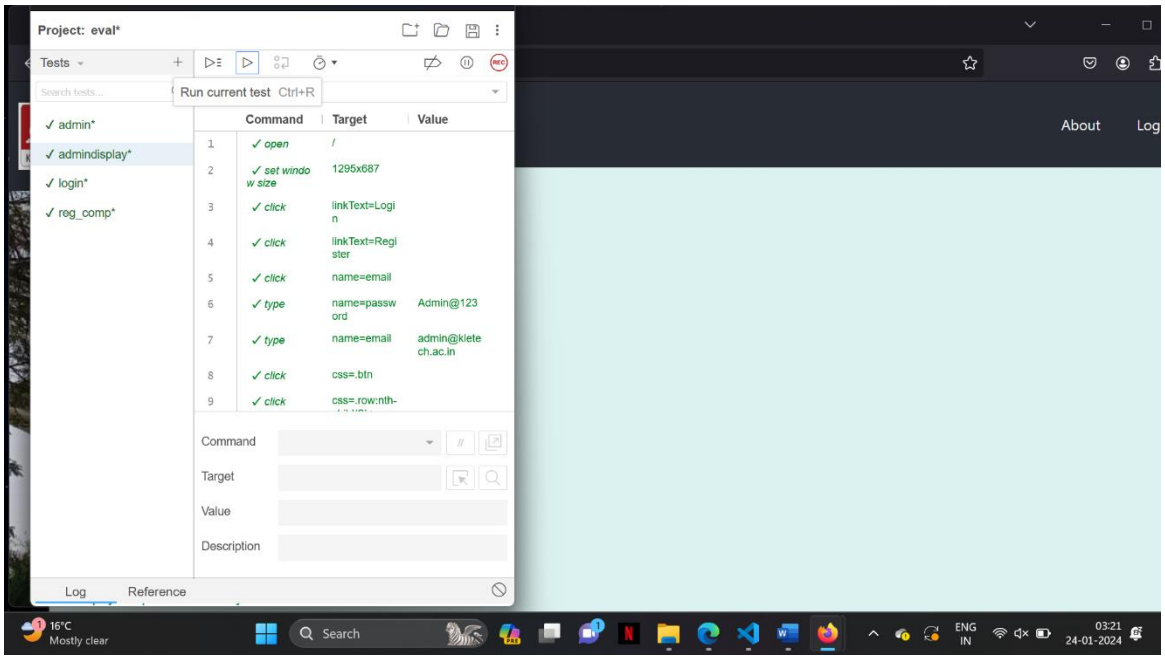


- **Registration Testing**



- **Admin Testing**





4.3 Testing Tool

Selenium IDE:

In the context of our faculty management system, Selenium proves to be an indispensable tool for ensuring the optimal performance and reliability of web-based functionalities. Acting as an essential component in our testing toolkit, Selenium facilitates the seamless execution of test scripts, allowing for the efficient validation of critical system functionalities. Its user-friendly interface and compatibility as a browser extension make it accessible for both testers and developers involved in our project. Selenium's role extends to validating the accurate addition and editing of faculty details, ensuring the smooth operation of the system's data management features. By incorporating Selenium into our testing framework, we streamline the verification process, contributing to the robustness and effectiveness of our faculty management system.

4.4 Continuous integration and continuous delivery (CI/CD)

Continuous Integration and Continuous Delivery (CI/CD) are integral components of modern software development, and Jenkins plays a pivotal role in implementing these practices within our project. Jenkins is an open-source automation server that facilitates seamless integration of code changes from multiple contributors in real-time. It automates the build, test, and deployment processes, ensuring that every code modification is thoroughly tested and integrated into the project. Through its extensibility and vast plugin ecosystem, Jenkins enables the orchestration of complex workflows, including code compilation, unit testing, and deployment to various environments. This not only accelerates the development cycle but also enhances the overall reliability and stability of our web technology project. Jenkins' user-friendly interface, combined with its ability to integrate with various version control systems, makes it an invaluable tool for achieving efficient CI/CD pipelines, contributing to a robust and streamlined development workflow.

5 Conclusion And Future Work

In conclusion, the development and implementation of the Automated Classroom and Laboratory Allocation System for In-Semester Assessments represent a significant stride toward enhancing the efficiency and transparency of educational resource management. By leveraging advanced algorithms and a user-friendly interface, the system streamlines the often intricate process of allocating classrooms and laboratories, addressing the complexities associated with manual scheduling. The project aligns with the broader objective of modernizing academic management practices, introducing automation to save time, minimize errors, and create an organized framework. The successful integration of the MERN (MongoDB, Express.js, React, Node.js) stack and the adherence to MVC architecture ensure a robust, scalable, and maintainable solution. This innovative system not only caters to the needs of administrators in optimizing resource utilization but also empowers faculty and students with a smoother and more transparent testing experience. The real-time notification system adds an extra layer of responsiveness, fostering improved communication and coordination. As we navigate the ever-evolving landscape of education in the digital age, the Automated Classroom and Laboratory Allocation System stands as a testament to our commitment to adaptability, efficiency, and the continuous pursuit of excellence in academic administration. This project serves as a foundation for creating a dynamic and responsive academic ecosystem that meets the evolving needs of educational institutions, ultimately contributing to a more streamlined and effective educational experience for all stakeholders.