

CAMPUS CONNECT

Group Members

Harinandana K Biju:FIT22CS092

Fathima Nazar:FIT22CS077

Merin P Wilson:FIT22CS125

Josna John:FIT22CS108

Project Guide: Ms. Remya R

April 4, 2025

Introduction

Project Domain: Web Development

Campus Connect is a digital platform designed to streamline communication and collaboration among students and faculty, enhancing their campus experience.

Key Features: Group chats, announcements, event management, resource sharing, and a chatbot.

Problem Statement

- Educational institutions rely on outdated communication methods, leading to missed events, poor engagement, and inefficient resource sharing. The lack of a centralized system hinders real-time collaboration among students, faculty, and administrators.
- Campus Connect aims to solve this by providing an integrated platform for event management, announcements, resource sharing, and group communication, ensuring seamless interaction and improved engagement.

Project Objectives

- **Seamless Communication:** Group chats and AI Chatbot.
- **Event Management:** Efficient platform for event organization.
- **Resource Sharing:** Faculty can upload and share educational materials.
- **User Accessibility:** Simple, user-friendly interface.
- **Administrative Ease:** Digital automation of tasks.

Purpose and Need

Purpose: Create a unified platform that facilitates easy communication and collaboration among students, faculty, and administration. It provides essential campus updates, event management tools, and communication features, all while improving student engagement.

Need: The absence of an integrated communication system in many campuses leads to fragmented and inefficient methods of interaction. This project aims to bridge that gap by providing a centralized platform that streamlines communication, encourages student participation, and simplifies administrative tasks.

Scope of the Project

Objective: Develop a platform that integrates campus-related communication and collaboration features, enhancing the campus experience. Features include announcements, group chats, event management, and more.

Social Relevance

Campus Connect modernizes campus communication by replacing outdated methods with a unified platform.

- Encourages student participation and collaboration.
- Reduces dependency on outdated notice boards and emails.
- Provides a central hub for all academic and extracurricular updates.
- Enhances accessibility for all students, including remote learners.
- Supports sustainable practices by reducing paper usage.

Software Requirement Specification: Functional Requirements

User Authentication:

- REQ-1: User login and registration
- REQ-2: Profile management for students, faculty and admin.

Real-Time Communication:

- REQ-3: Group chat functionality for real-time messaging.
- REQ-4: Announcements and updates.

Event Management:

- REQ-5: Events created by admin and students can register for it.

Resource Sharing:

- REQ-6: Faculty access to upload and manage educational resources (PDFs, links).

AI Chatbot:

- REQ-7: Integration of an AI chatbot for answering FAQs related to campus activities.

Non-Functional Requirements

- **Security:** Secure data storage and transmission.
- **Usability:** User-friendly design.
- **Performance:** Efficient even under high load.
- **Reliability:** Ensure platform availability 24/7 without major downtimes.
- **Scalability:** Must support an increasing number of users and events.
- **Compatibility:** Should work across multiple devices and browsers.

System Architecture

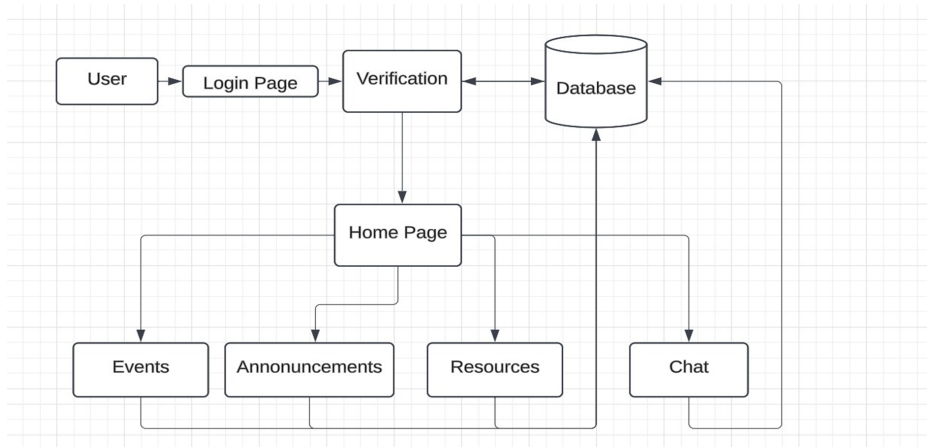


Figure: System Architecture Diagram

Campus Connect - GUI Overview

Faculty Sign In:

- Created using React.js and styled with CSS frameworks.
- Faculty can upload the resources and post the announcements.

Student Sign In:

- Created using React.js and styled with CSS frameworks.
- The students can view upcoming events and they can register for it. They can also access the resources uploaded by the faculty. Important announcements can be viewed.

Admin Sign In:

- Created using React.js and styled with CSS frameworks.
- Create and manage events. Monitors the groupchat.

API Design - Event Management Module

- **Endpoint:** /api/add-events
- **Method:** POST (For Admins)
- **Purpose:** Add new events to the database.
- **Input:** Event name, date, time, venue, description.
- **Output:** Confirmation message with event ID or Error message

API Design - Event Retrieval Module

- **Endpoint:** /api/get-events
- **Method:** GET (For Students/Faculty)
- **Purpose:** Fetch all upcoming events.
- **Output:** List of events in JSON format.

API Design - Login Module

- **Endpoint:** /api/login
- **Method:** POST
- **Purpose:** Allow students to login
- **Input:** Name, Password
- **Output:** On success, return token and username; on failure, return error message (Invalid username or password).

API Design - Resource Management Module

- **Endpoint:** /api/upload
- **Method:** POST (For Faculty)
- **Purpose:** Upload study materials or resources.
- **Input:** File upload, title, description.
- **Output:** Will display the success and error message.

API Design - Group Chat Module

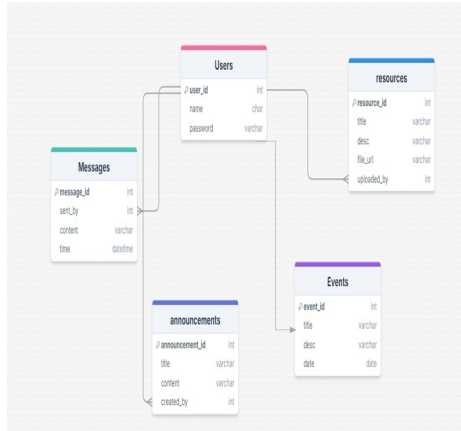
- **Endpoint:** /api/messages
- **Method:** GET
- **Purpose:** To fetch all messages from the database, ordered by timestamp (oldest first).
- **Output:** Displaying of real-time messages.

API Design - AI Chatbot Module

- **Endpoint:** /api/chatbot
- **Method:** POST
- **Purpose:** Provide instant responses to FAQs.
- **Input:** User query.
- **Output:** Relevant answer or suggestion.

Database Design

The database will store user details, academic schedules, event information, and shared resources. It will be designed to scale as the user base grows and will be fully integrated with the authentication system to manage access and permissions securely.



Tech Stack

- **Frontend:** React.js for the web application.
- **Backend:** Node.js with Express.js for server-side operations.
- **Database:** MongoDB for storing user data, academic schedules, and resources.

Data Flow Diagram - Level 0

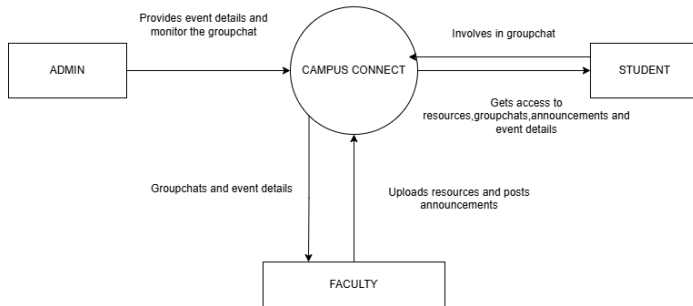


Figure: Data Flow Diagram - Level 0

Data Flow Diagram - Level 1

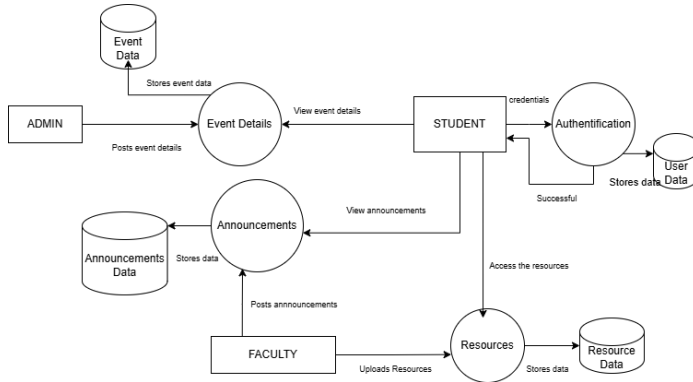


Figure: Data Flow Diagram - Level 1

Algorithm for User Login in Campus Connect

Start

1. Initialize State Navigation

- 1.1. Set up `useState` for `username`, `password`, and `message`.
- 1.2. Redirect to `/Home` if `localStorage` contains a login token.

2. Handle Input Submission

- 2.1. Capture input changes in `formData`.
- 2.2. On form submission:
 - 2.2.1. Validate inputs; show error if empty.
 - 2.2.2. Set `loading = true`, send POST request to `/login`.
 - 2.2.3. On success, store token in `localStorage`, redirect to `/Home`.

Stop

Algorithm for User Registration in Campus Connect

Start

1. Initialize State Navigation

- 1.1. Set up `useState` for form data and messages.
- 1.2. Use `useNavigate` for redirection.

2. Handle Input Validation

- 2.1. Capture input changes.
- 2.2. Validate inputs:
 - 2.2.1. Show error if fields are empty or passwords don't match.

Stop

Algorithm for Event Page in Campus Connect

Start

1. Fetch Event Data

- 1.1. Use `useEffect` to fetch events from MongoDB.
- 1.2. Store retrieved data in a `useState` variable.

2. Display Event List

- 2.1. Map through event data to display event cards.
- 2.2. Show event title, date, description, and a register button.

3. Handle Event Registration

- 3.1. On clicking "Register", navigate to `RegisterPage.js`.
- 3.2. Pass event details via state parameters.

Algorithm for Displaying Announcements

Initialize State

- Create `announcements` state to store announcement data.
- Create `title` and `message` state for new announcements.

Fetch Announcements

- Send a GET request to retrieve announcements from the server.
- If successful, update announcement state.

Add Announcement (Faculty Only)

- Validate input fields (`title`, `message`).
- Send a POST request to save the announcement.
- If successful, display a success message.

Display Announcements

- Show all announcements in descending order of `createdAt`.

Algorithm for Resource Uploading

Initialize State:

- Get `className` from `useParams()`.
- Set states for subjects, title, description, and file.

Populate Subjects:

- Set subjects based on `className`.

Handle File Selection:

- Update file state when a file is selected.

Handle Upload:

- Validate inputs (`title`, `description`, `file`).
- Prepare `FormData` with the data.
- Send POST request to upload via `axios`.
- Show success or error message.

Reset After Upload:

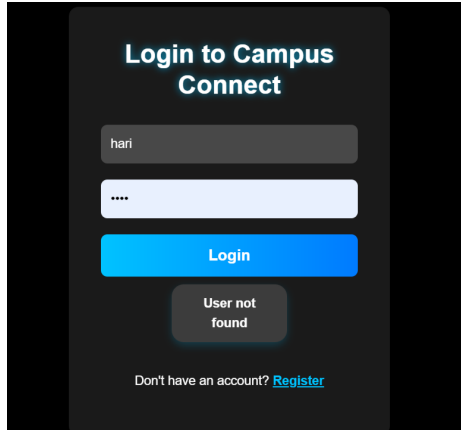
- Clear states and reset file input after successful upload.

Individual Contribution

- **Josna John** - Designed home page, Login Page,signup page and implemented group chat for communication through different logins of students.
- **Harinandana K Biju** - Created ChatBot Assistance with FAQs, Designed Resource page to upload pdf and other study materials, Fetch resource materials based on semester from db .
- **Fathima Nazar** - Designed Home page,Announcement Page to add important announcements ,Fetch details from db and displayed the announcements(backend).
- **Merin P Wilson** - Designed signUp page, Event Page which includes adding upcoming events of college (to backend). Fetch the event details from db, displaying the event details in the event page.

Unit Testing - User Authentication

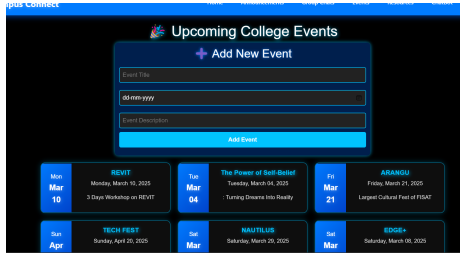
- **TC-01:** Verify that users can log in with correct credentials and access the dashboard.
- **TC-02:** Verify that users with invalid credentials receive an appropriate error message.



The screenshot displays the 'Login to Campus Connect' interface. It features a dark background with a central white card. The card has the title 'Login to Campus Connect' at the top. Below the title are two input fields: the first contains the text 'hari' and the second contains four dots, indicating a password field. A red 'Login' button is positioned below the input fields. A light gray error message box with the text 'User not found' is centered below the button. At the bottom of the card, there is a link that says 'Don't have an account? [Register](#)'.

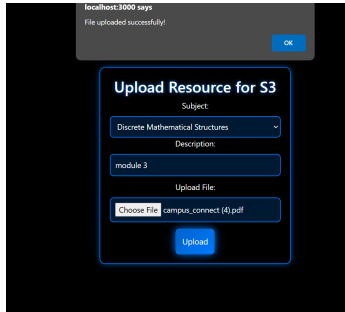
Unit Testing - Event Management

- **TC-03:** Verify that an admin or authorized user can create an event with valid details.
- **TC-04:** Verify that students can view upcoming events on the homepage.
- **TC-05:** Verify that after registrations, an email is sent to the students.



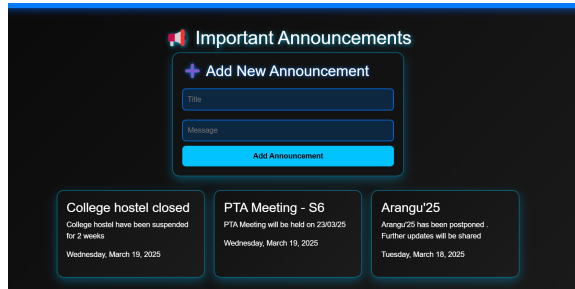
Unit Testing - Resource Sharing

- **TC-06:** Verify that faculty members can upload resources in different formats (PDF, DOCX, PPT, etc.).
- **TC-07:** Verify that students can access and download resources without issues.
- **TC-8:** Verify that only authorized users (faculty) can upload resources.
- **TC-9:** Verify that if a file is not uploaded, an error message is displayed.



Unit Testing - Announcements

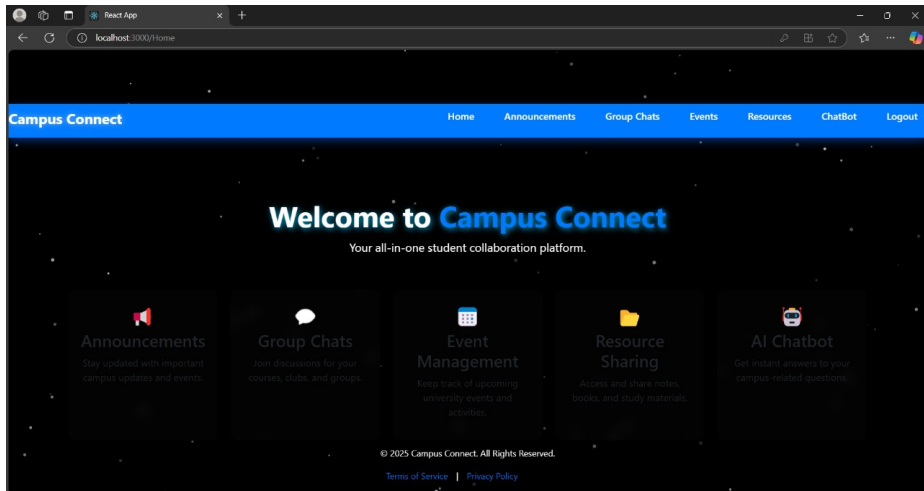
- **TC-10:** Verify that faculty can post announcements.
- **TC-11:** Verify that announcements can be viewed by students.
- **TC-12:** Verify that unauthorized users cannot post announcements.



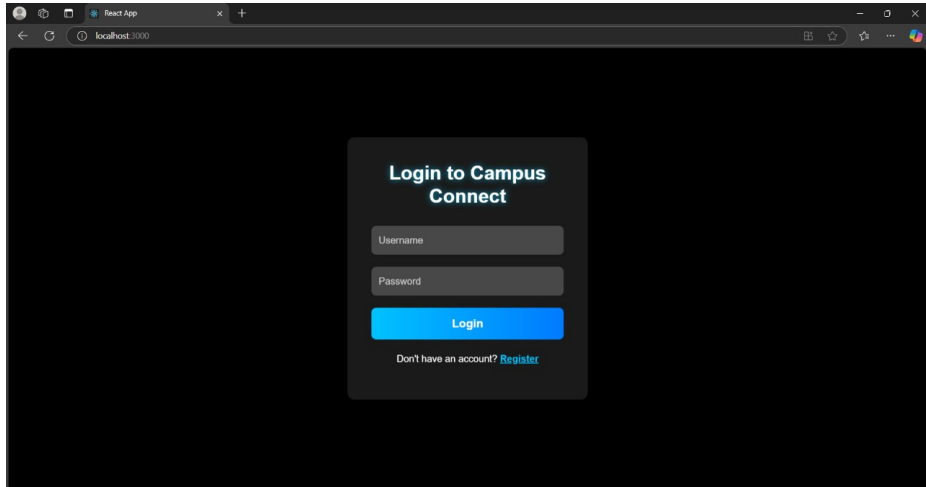
The screenshot displays a web interface titled "Important Announcements". At the top, there is a red icon followed by the title. Below the title is a form to "Add New Announcement". The form includes a blue plus icon, a "Title" input field, a "Message" input field, and a blue "Add Announcement" button. Below the form, there are three announcement cards, each with a title, a description, and a date.

Announcement Title	Description	Date
College hostel closed	College hostel have been suspended for 2 weeks	Wednesday, March 19, 2025
PTA Meeting - S6	PTA Meeting will be held on 23/03/25	Wednesday, March 19, 2025
Arangu'25	Arangu'25 has been postponed . Further updates will be shared	Tuesday, March 18, 2025

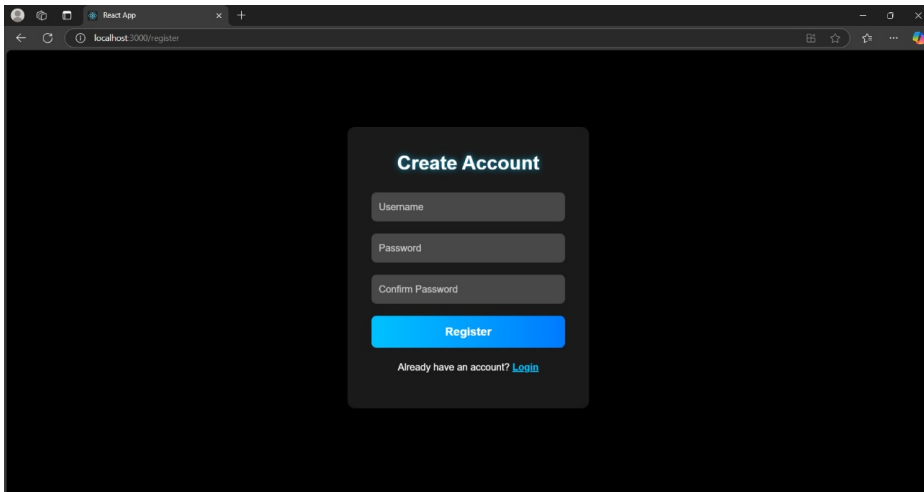
Home Page



Login Page



Signup Page



The screenshot shows a web browser window with a single tab titled 'React App'. The address bar displays 'localhost:3000/register'. The page content is a dark-themed registration form centered on the screen. The form has a title 'Create Account' in white. It contains three input fields: 'Username', 'Password', and 'Confirm Password', all with light gray borders. Below these fields is a prominent blue 'Register' button with white text. At the bottom of the form, there is a link that says 'Already have an account? [Login](#)'.

React App

localhost:3000/register

Create Account

Username

Password

Confirm Password

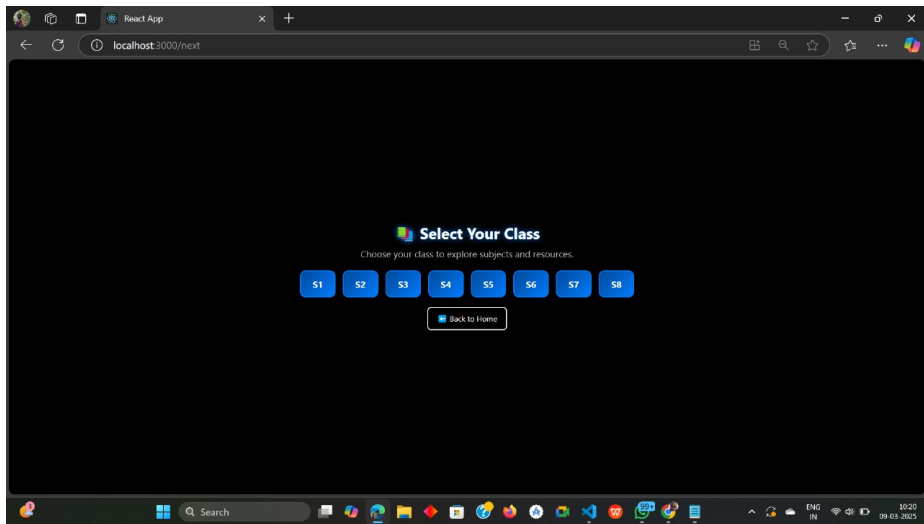
Register

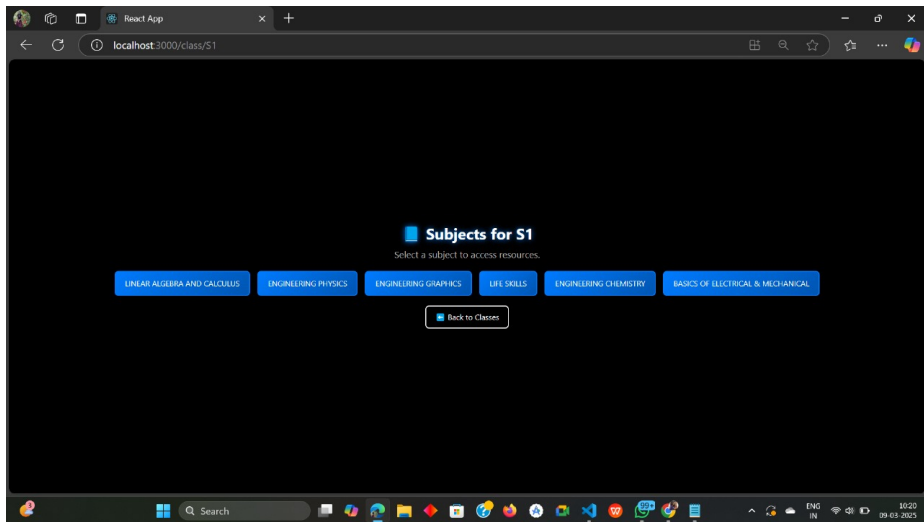
Already have an account? [Login](#)

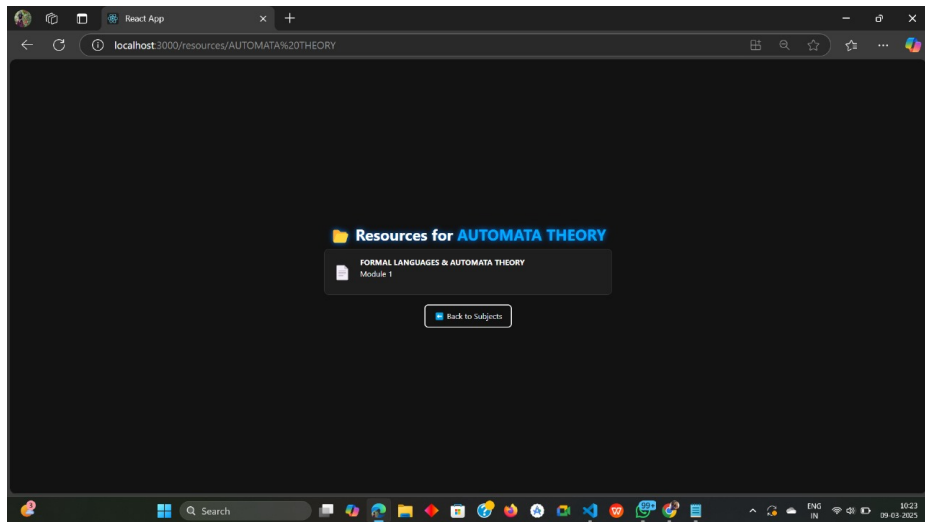
Group Chat



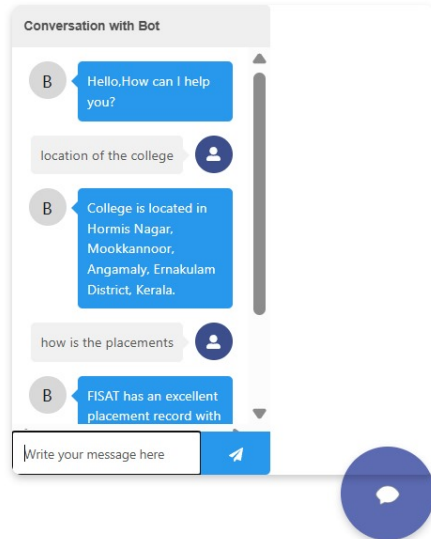
Resource Page



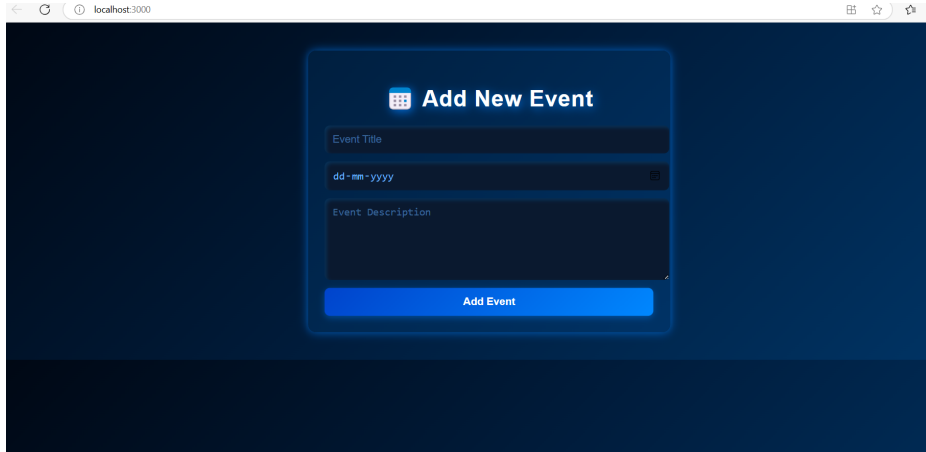




ChatBot



Event Page



The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. The main content area has a dark blue background. In the center, there is a rounded rectangle with a lighter blue border and a subtle glow. Inside this rectangle, at the top, is a calendar icon followed by the text 'Add New Event'. Below this, there are three input fields: 'Event Title', a date field with the placeholder 'dd-mm-yyyy' and a calendar icon, and a larger 'Event Description' field. At the bottom of the rounded rectangle is a bright blue button with the text 'Add Event'.

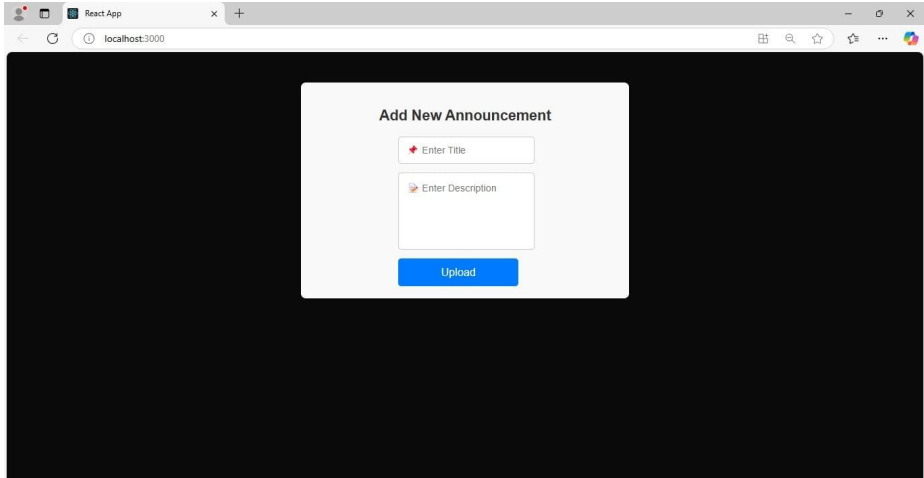
← ↻ ⓘ localhost:3000

🏠 ☆ 🌟

Upcoming College Events

<p>Tue Mar 04</p>	<p>The Power of Self-Belief</p> <p>Tuesday, March 04, 2025</p> <p>: Turning Dreams Into Reality</p>	<p>Mon Mar 10</p>	<p>REVIT</p> <p>Monday, March 10, 2025</p> <p>3 Days Workshop on REVIT</p>	<p>Fri Mar 21</p>	<p>ARANGU</p> <p>Friday, March 21, 2025</p> <p>Largest Cultural Fest of FISAT</p>
<p>Fri Apr 04</p>	<p>NAUTILUS</p> <p>Friday, April 04, 2025</p> <p>Coming soon..</p>	<p>Sun Apr 20</p>	<p>TECH FEST</p> <p>Sunday, April 20, 2025</p> <p>Largest technical fest at FISAT</p>	<p>Tue Jul 08</p>	<p>ACCTHPA – 2025</p> <p>Tuesday, July 08, 2025</p> <p>International Conference on Advanced Computing and Communication Technologies for High Performance Applications</p>

Announcement Page



React App

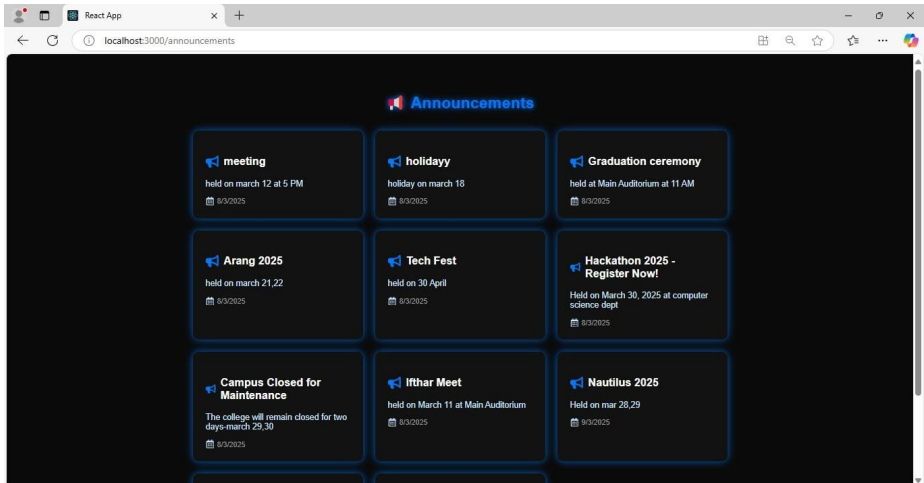
localhost:3000

Add New Announcement

✖ Enter Title

🖼 Enter Description

Upload



Project Schedule

Sl.No	Date	Activity
1	3/2/25 – 7/2/25	Login page (backend and frontend)
2	7/2/25 – 10/2/25	Signup page (backend and frontend)
3	12/2/25 – 13/2/25	Home page (Frontend)
4	15/2/25 -19/2/25	Event page (frontend and backend)
5	20/2/25 – 23/2/25	Resource page (frontend and backend)
6	25/2/25 - 28/2/25	Announcement page (frontend and backend)
7	1/3/25 – 4/3/25	Chatbot (frontend and backend)
8	5/3/25 – 9/3/25	Group chat (frontend and backend)

Conclusion

- **Streamlined Communication** – Centralized platform for students, faculty, and admins.
- **Real-Time Updates** – Instant notifications for announcements and events.
- **Event Resource Management** – Easy access to study materials and event coordination.
- **Chatbot Group Chat** – Quick information access.
- **Secure Scalable** – Built with modern technologies for reliability and ease of use.
- **Enhanced Campus Engagement** – Fosters a connected and efficient academic environment.

Bibliography

- MongoDB Documentation - <https://docs.mongodb.com/manual/>
- React Documentation - <https://reactjs.org/docs/getting-started.html>
- Node.js Documentation - <https://nodejs.org/en/docs/>
- Axios Documentation - <https://axios-http.com/docs/intro>
- Mongoose Documentation - <https://mongoosejs.com/docs/>