

Orbit Campus

Orbit Campus: A smart campus management platform designed to streamline student services, enhance communication, and simplify academic operations.



Empowering Students and Administration with Smart Technology: (Orbit campus)

A Seamlessly Integrated Web System Crafted with Frontend and Backend Technologies

Submitted By

D. Chiranjeevi

23072-CM-016

Report Submitted to

DR. B. R. AMBEDKAR GOVERNMENT MODEL RESIDENTIAL POLYTECHNIC

Under the Guidance of

Kum. N.SANDYA RANI (L/CME, M. TECH)

DEPARTMENT OF COMPUTER ENGINEERING



Dr. B.R. Ambedkar Government Model Residential Polytechnic

RAJAMAHENDRAVARAM-533124 2025-2026

©2025, Orbital Campus All rights reserved

DEPARTMENT OF COMPUTER ENGINEERING

CERTIFICATE

**This is to certify that this project work entitled
“ORBIT CAMPUS “is the Bonafide record work done by Mr./Ms.**

**_____ Bearing
the Pin Number. _____ of the final
year, along with batch mates submitted on the partial fulfilment of the
Requirement for the award of DIPLOMA IN COMPUTER ENGINEERING
during the academic year 2023-2026.**

Signature of project guide

Kum. N.SANDYA RANI L/CME, M. TECH

Head of the Department

P.V. LAKSHMINARAYANA, M. Tech

Signature of the External Examiner

ACKNOWLEDGEMENT

I have made every effort to complete this project with the support and valuable suggestions from my classmates, ensuring it functions efficiently and effectively.

I would like to express my sincere gratitude to my project guide, Kum. N. Sandya Rani, for her guidance, cooperation, and valuable suggestions, which motivated me to successfully complete this project.

I also extend my thanks to our Senior Lecturer, P.V.LAKSHMINARAYANA, and Lecturers Smt. G.SriDevi, Smt.CH. Kalyani Bindu, Smt. N. Pavani for their valuable suggestions throughout the course of the project.

I am thankful to our Head of the Computer Engineering Department and project coordinator, Sri P.V.LAKSHMINARAYANA, for his cooperation and unwavering dedication, which greatly inspired me to complete this project.

I also grateful to our respected Principal, Sri A. Murali, for his kind support in the successful completion of this project.

Lastly, I extend my gratitude to all the teaching and non-teaching staff of the Computer Engineering department, who have assisted me, either directly or indirectly, in the completion of this project.

PROJECT ASSOCIATES

Name	Pin Number
D. Chiranjeevi	23072-CM-016
K. Manikanta Karthik	23072-CM-029
P. Nanda Kishore	23072-CM-038
B. Siddhardha	23072-CM-004
P. Jaswanth Laxman	23072-CM-047
B. Bala Manohar	23072-CM-058

TABLE OF CONTENT

S.no	CONTENTS	PG.no
1	ABSTRACT	
2	SOFTWARE REQUIREMENTS	
3	INTRODUCTION	
4	SYSTEM ANALYSIS AND FEASIBILITY STUDY	
5	SRS & DESIGN	
6	SOFTWARE INSTALLATION	
7	SOURCE CODE AND TESTING	
8	USER MANUAL	
9	OUTPUT SCREENS AND BIOGRAPHY	

ABSTRACT

Campus Sphere is a web-based campus management system engineered to centralize academic, administrative, and communication workflows within a unified digital platform. The system employs a modular architecture integrating role-based access control (RBAC), secure authentication mechanisms, and a scalable backend to streamline operations for students, faculty, and administrators. Core functionalities include automated attendance processing, digital notice dissemination, event and timetable management, academic content delivery, and two-way communication modules.

The platform is built using modern web technologies to support efficient data handling, real-time updates, and responsive user interactions. A structured relational database ensures reliable storage, retrieval, and synchronization of campus data, while optimized backend APIs facilitate smooth transactions across modules. By digitizing manual procedures and enabling seamless information flow, Campus Sphere enhances operational efficiency, reduces redundancy, and improves overall campus connectivity. This project demonstrates a technically robust approach to campus digitization through secure architecture, modular design, and user-centric functionality.

Existing System:

- Most educational institutions still rely on traditional and semi-digital methods to manage campus activities, communication, and student services. Many processes—such as announcements, attendance, timetable management, event updates, and academic resource sharing—are handled manually or distributed across multiple unconnected platforms. This leads to inefficiencies, delays, and a lack of centralized control.
- Current campus management systems face several limitations:
- Fragmented Communication – Notices and updates are shared through physical notice boards, SMS groups, or separate apps, causing inconsistent information flow.
- Manual Administrative Processes – Attendance, event registration, and academic documentation are often handled manually, making them time-consuming and error-prone.
- Lack of Centralized Access – Students and faculty must use multiple channels to access timetables, study materials, announcements, and services.
- Limited Transparency – Tracking activities such as academic progress, campus events, or service requests becomes difficult without a unified system.
- Accessibility Issues – There is no single platform for accessing real-time campus information anytime, anywhere.
- These limitations highlight the need for an integrated, automated, and efficient campus management solution.

Proposed System:

Smart Campus Sphere introduces a unified, web-based campus management platform designed to streamline academic, administrative, and communication processes within a single system. The platform provides role-based access for students, faculty, and

administrators, enabling seamless interaction and real-time information sharing across the campus ecosystem.

Key features and advantages include:

- Centralized Communication Hub – Instantly shares notices, announcements, and updates to all users through a unified dashboard.
- Automated Administrative Tasks – Digital attendance, event management, and academic record handling reduce manual work and improve accuracy.
- Unified Resource Access – Provides students and staff with organized timetables, study materials, and campus services from one platform.
- Real-Time Updates – Ensures immediate access to academic schedules, upcoming events, and urgent notifications.
- Secure and Scalable Architecture – Implements authentication, role-based access control (RBAC), and structured database management, ensuring data privacy and future scalability.
- Smart Campus Sphere significantly enhances campus efficiency, transparency, and connectivity by combining automation, centralized access, and real-time information delivery into a single intelligent platform.

SOFTWARE REQUIREMENTS

SOFTWARE REQUIREMENTS :

1. Frontend Requirements

- Your frontend is built using React + Vite, so the following software is required:

- Development Tools
- Node.js (v16+)
- npm (for installing dependencies)
- Vite (already used inside your project)

Code Editor:

- VS Code (recommended)

Frontend Libraries:

- React JS
- React Router
- Axios (for API communication)
- Tailwind / CSS (if included)
- Node Modules (installed using npm install)

Browser Requirements

- Google Chrome
 - Mozilla Firefox
 - Microsoft Edge
- (Any browser supporting ES6+ JS and React)

2. Backend Requirements

- Your backend uses Core PHP + REST APIs, so:
- Server Software
- Apache Server (included in XAMPP)
- PHP 7.4 or above
- MySQL
- phpMyAdmin (for database management)
- PHP Extensions Required
 - mysqli
 - pdo
 - json
 - fileinfo
 - curl (optional for external APIs)
- Backend Structure Found
 - /api folder with endpoints
 - config/cors.php — for CORS handling

- config/database.php — DB connection
- models/ — PHP model classes
- database.sql — to create DB tables
- File upload scripts (profile photo, documents)

3. Operating System Compatibility

- Windows 10 / 11
- Linux (Ubuntu recommended)
- macOS

HARDWARE REQUIREMENTS

1. Minimum Hardware Requirements

- Processor: Dual-core 2.0 GHz
- RAM: 4 GB
- Storage: 2–4 GB free space (Node modules + PHP + Database)
- Display: 720p or higher
- Input: Keyboard & Mouse

2. Recommended Hardware Requirements

- Running React + PHP + MySQL + Node.js together requires more power:
- Processor: Intel i5 / Ryzen 5 or higher
- RAM: 8 GB or above
- Storage: 10–20 GB free (node_modules are large)
- Display: Full HD (1080p)

3. Server-Side Requirements

- If your project will run on an internal server or cloud:
- Server Hardware
- Processor: Quad-core CPU
- RAM: 8–16 GB

- Storage:
- 100+ GB SSD for documents, logs, DB backups
- Network:
- High-speed LAN or WiFi
- Stable internet (for cloud apps)
- Power Backup
- UPS or generator (for physical campus server)

4. Client Device Requirements

- Students/Faculty can access your system using:
- Smartphone (Android/iOS)
- Laptop/Desktop
- Tablet
- Minimum:
- RAM: 2 GB
- Browser: Any modern browser
- OS: Android/iOS/Windows/Linux

WEB BROWSER:

- CHROME.

INTRODUCTION

Smart Campus Sphere is a web-based campus management system designed to digitalize and streamline essential academic and administrative activities within an educational institution. The platform provides a unified interface for students, faculty, and administrators to manage daily campus operations efficiently. Built with a **React + Vite frontend** and a **PHP–MySQL backend**, the system ensures fast performance, secure data handling, and smooth communication between all campus stakeholders.

The objective of Smart Campus Sphere is to eliminate manual paperwork, reduce administrative delays, and create a centralized digital ecosystem where users can access services such as academic records, document requests, fee information, notifications, and profile management.

The backend follows a structured REST API architecture, enabling seamless data exchange with the modern, interactive frontend.

The system uses a **role-based access model**, ensuring that students, faculty, and administrators only access features relevant to their roles. With modules like user authentication, document uploads, request handling, dashboards, and communication features, Smart Campus Sphere enhances transparency, reduces human errors, and improves overall campus workflow efficiency.

Objectives

- **Centralize Campus Services** – Provide a single digital platform for students, faculty, and admin workflows.
- **Improve Information Accessibility** – Enable users to quickly access academic and administrative data from anywhere.
- **Enhance Operational Efficiency** – Reduce paperwork and manual processing through automated modules.
- **Streamline Communication** – Deliver real-time updates, notices, and notifications to users.
- **Ensure Security & Reliability** – Implement secure authentication and database-driven data management.
- **Core Functionalities Supported**
- **User Authentication** – Secure login for students, faculty, and administrators.
- **Dashboard System** – Role-based dashboards for personalized access to campus services.
- **Document Management** – Uploading, requesting, and tracking academic documents.
- **Profile Handling** – Updating user profiles and uploading profile photos.
- **Fee & Academic Records** – Viewing fee details, academic information, and history.
- **Notifications & Announcements** – Real-time updates from the administration.

Smart Campus Sphere is designed to be **scalable, secure, and user-friendly**, leveraging modern web technologies to provide a responsive interface and efficient backend processing. This document outlines the system's technical design, functional modules, and requirements necessary for the successful development and deployment of the platform.

1. HTML (HyperText Markup Language)

Purpose in the Project:

- Forms the **structural foundation** of all web pages.
- Defines layout elements such as login forms, dashboards, tables, cards, and content sections.
- Works together with React components to display user interfaces dynamically.

2. CSS (Cascading Style Sheets)

Purpose in the Project:

- Responsible for the **styling and visual design** of the interface.
- Provides layout formatting, color schemes, animations, spacing, and responsive design.
- Ensures a clean and user-friendly UI for students, faculty, and administrators.

3. JavaScript

Purpose in the Project:

- Powers the **interactive and dynamic behavior** of the frontend.
- Used extensively inside **React components** to manage state, events, routing, and API calls.
- Controls dashboard elements, form validations, data rendering, and navigation.
- Communicates with backend APIs using **Axios** or Fetch requests.

4. React JS

(Framework built using JavaScript)

Purpose in the Project:

- Used to build the **entire frontend application** with reusable components.
- Handles:
 - Routing between pages
 - Dashboard rendering
 - Data fetching from backend
 - State management
 - Real-time UI updates
- Offers a highly responsive, fast, and modern user interface for the Smart Campus Sphere.

5. PHP (Hypertext Preprocessor)

Purpose in the Project:

- Serves as the **backend language** responsible for:
 - Handling user authentication

- Verifying login credentials
- Managing requests for documents, fees, notices, and user data
- Running business logic for admin and student modules
- Implements **REST API endpoints** to communicate with the React frontend.
- Ensures secure data processing using sessions, validation, and database queries.

6. MySQL (Structured Query Language)

Purpose in the Project:

- Acts as the **database system** for storing and managing:
 - User details
 - Login credentials
 - Student records
 - Document uploads
 - Fees, notices, and academic information
- Provides efficient data retrieval for dashboards and API responses.
- Ensures secure and structured storage through relational database design.

7. JSON (JavaScript Object Notation)

Purpose in the Project:

- Used for **data exchange** between frontend and backend.
 - API responses from PHP backend to React frontend are formatted in JSON.
 - Makes communication lightweight, fast, and structured.
-

8. Vite Build Tool

Purpose in the Project:

- Used for compiling and bundling the React frontend.
- Provides ultra-fast development server and optimized production build.

- Ensures smooth and fast UI loading for end users.

SYSTEM ANALYSIS AND FEASIBILITY STUDY

1. System Analysis

Smart Campus Sphere is designed as a full-stack web-based platform that simplifies and digitizes campus-related operations such as student management, document handling, profile updates, notifications, and administrative tasks. The uploaded ZIP files show a React + Vite frontend communicating with a PHP + MySQL backend, indicating a modern and scalable system architecture.

1.1 Existing Problems Identified

- Before the system was designed, the following limitations were observed in traditional campus workflows:
- Manual handling of student records leads to delays and errors.
- No centralized platform for accessing academic, fee, and personal information.
- Communication gaps between students, faculty, and administration.
- Document requests and verifications handled offline, causing delays.
- No automated system to track user activities, updates, or notifications.
- Lack of transparency and real-time updates for academic/administrative processes.

1.2 System Requirements Identified

- From the project structure, the following requirements were derived:
- A secure authentication system (present in the PHP backend).
- A role-based dashboard for different users (students/admin).
- A user-friendly UI built with React components.
- Database connectivity for storing profile, documents, and fee details.
- API-level communication using JSON responses.
- Support for uploading files (profile photos, documents).
- A fast frontend build system (Vite).

1.3 System Workflow Overview

- User logs into the system (React → PHP API → MySQL).
- Backend validates credentials and returns role-based data.
- React renders dynamic dashboard using fetched JSON responses.

- Users can update profiles, request documents, or check records.
- Admins can add notices, process requests, and manage students.
- All updates are stored in MySQL and retrieved on demand.

2. Feasibility Study

A feasibility study was conducted to analyze the practicality of developing and deploying Smart Campus Sphere. Based on the source code, the system is technically and operationally feasible for real-world use.

2.1 Technical Feasibility

- Smart Campus Sphere uses widely supported and reliable technologies:
- Frontend Technology Feasibility
- React JS ensures a fast, component-driven UI.
- Vite provides high-speed bundling and development experience.
- Fully compatible across devices (mobile, desktop).
- Backend Technology Feasibility
- Core PHP is lightweight, easy to deploy, and compatible with shared hosting.
- REST APIs make frontend-backend communication structured and efficient.
- MySQL ensures stable and secure data storage.
- The technologies used are easy to maintain, scalable, and have a strong community support base, making the system technically feasible.

2.2 Operational Feasibility

- The system is easy to use and reduces workload for both students and administrators.
- Operational Benefits
- Clean React UI ensures high user usability.
- Centralized database simplifies management and retrieval.
- Automated workflows reduce manual errors and delays.
- Role-based dashboards improve clarity and operations.
- Admin panel supports quick updates and management.
- Users (students/admins) require very minimal training, proving the system's operational feasibility.

2.3 Economic Feasibility

- Smart Campus Sphere is cost-effective because:
- Development Cost
- Uses open-source technologies (React, PHP, MySQL).

- No need for expensive servers — works on XAMPP or shared hosting.
- Maintenance Cost
- Easy to maintain due to modular frontend and API-based backend.
- Requires only basic hosting and domain expenses.
- Deployment Cost
- Can run on low-cost servers or educational institution infrastructure.
- Overall, the system is economically feasible with minimal setup and maintenance costs.

2.4 Schedule Feasibility

- Based on the folder structure and development progress:
- Frontend and backend modules are already well-structured.
- API endpoints and React components show modular development.
- Can be completed within a typical academic project timeline.
- Thus, the system meets realistic schedule feasibility.

SOFTWARE REQUIREMENTS & SPECIFICATION (SRS)

1. Introduction

The Smart Campus Sphere system is a web-based campus management platform designed to streamline student services, administrative workflows, and data management. The system uses a React + Vite frontend and PHP + MySQL backend, communicating through REST APIs to provide a fast, secure, and user-friendly experience.

This SRS document describes functional and non-functional requirements essential for the system's development, deployment, and maintenance.

2. Overall Description

- Smart Campus Sphere provides:
- A centralized dashboard for students and administrators
- Profile management and document upload features
- Real-time notifications
- Academic and personal data handling
- Admin controls for user management and updates
- It operates through a client–server architecture where the browser (React UI) communicates with PHP APIs connected to a MySQL database.

3. Software Requirements

Below are the tools, technologies, and libraries required to build and run the system.

3.1 Frontend Software Requirements

Technologies Used

Component	Details
React JS	For building UI components and dynamic pages
Vite	For bundling, fast development server, hot reloading
JavaScript (ES6+)	Main programming language for frontend logic
HTML5 & CSS3	Structure and styling of UI
Axios / Fetch API	For making API calls to backend
Node.js + NPM	For installing and managing frontend dependencies

- Purpose in Project
- Rendering responsive dashboards
- Handling user interactions (login, forms, updates)
- Displaying data returned from PHP API
- Providing smooth UI/UX with reusable components
- Managing state and session data on client side

3.2 Backend Software Requirements

Component	Details
PHP 7+	Server-side scripting, API handling, authentication
MySQL / MariaDB	Database for storing user details, records, documents
Apache Server	Required for running PHP APIs (XAMPP/WAMP/LAMP)
JSON	Data format for frontend–backend communication
phpMyAdmin	DB management tool

- Purpose in Project
- Authenticate users and manage roles (student/admin)
- Store and retrieve profile, academic, and admin data

- Handle document uploads (photos, PDFs, etc.)
- Return JSON responses to the React frontend
- Perform CRUD operations on database records

3.3 Additional Software Requirements

- Browser Support
- Chrome
- Edge
- Firefox
- Safari
- Operating System Compatibility
- Windows 10/11
- Linux (Ubuntu/CentOS)
- macOS

4. Functional Requirements

4.1 User Authentication

- User login with email and password
- Role-based access (student/admin)

4.2 Student Dashboard

- View personal information
- Update profile details
- Upload required documents
- Check academic/administrative notifications

4.3 Admin Dashboard

- Manage students (add/update/delete)
- Approve/verify student documents
- Post campus announcements
- Monitor system usage

4.4 Database Management

- Store student profiles
- Maintain academic and personal records

- Store uploaded documents securely

5. Non-Functional Requirements

5.1 Performance Requirements

- Fast load times ensured by Vite + React
- API responses optimized for low latency
- Database queries must execute under 2 seconds

5.2 Security Requirements

- Passwords stored using hashing
- Prevention of SQL injection
- Secure API endpoints with validation
- Document access restricted by user roles

5.3 Reliability

- System must handle multiple concurrent users
- Database backup and restore options must be available

5.4 Usability

- Simple and intuitive UI for students and admin
- Responsive design for mobile and desktop

5.5 Scalability

- Can integrate future modules: attendance, fees, hostel, exams
- React and PHP architecture supports horizontal scaling

6. Interface Requirements

- User Interface
- React-based dashboards for both student and admin
- Forms, tables, modals designed using HTML/CSS/JS
- Backend API Interface
- REST API endpoints returning JSON
- Supports GET, POST, PUT, DELETE operations

7. Database Requirements

- MySQL database with structured tables:
- students
- admins
- documents
- notifications
- activity_logs
- Proper indexing for fast queries
- Foreign keys for relational integrity

SYSTEM DESIGN

System Design

System design for Campus Sphere involves structuring the architecture, components, and workflow to ensure an efficient, scalable, and reliable college management platform.

Architecture Overview

1. Campus Sphere follows a classic three-tier architecture:
2. **Presentation Layer (Frontend):** The user interface built with React and TypeScript. This is what users (students, teachers, admins, etc.) interact with to access information, submit data, and view dashboards.
3. **Application Layer (Backend):** A PHP-based server that handles all business logic. It processes API requests from the frontend, manages user authentication, interacts with the database, and executes core functionalities.
4. **Data Layer (Database):** A MySQL database that stores all persistent data for the application. This includes user records, academic information, financial details, and service requests, all defined by the campus_sphere.sql schema.

System Components

- User Interface (UI): Built using React, TypeScript, and styled with Tailwind CSS. It consists of interactive components for each user role.
- PHP Backend: Handles user authentication, serves API endpoints for all CRUD (Create, Read, Update, Delete) operations, and manages application logic.

- MySQL Database: Stores all relational data, including users, courses, grades, fee_records, and service_requests tables.
- Authentication & State Management: A context-based authentication system (AuthContext.tsx) manages user sessions and roles on the frontend.

System Workflow

1. **User Authentication:** Secure login for all roles (Student, Teacher, Admin, Parent, HOD). The backend verifies credentials against the users table in the database.
2. **Data Submission:** Users submit data via forms on the frontend (e.g., a student requesting a document via NewDocumentRequest.tsx).
3. **API Request:** The frontend sends the data to the appropriate PHP API endpoint (e.g., create_document_request.php).
4. **Backend Processing:** The PHP script sanitizes the input, applies business logic, and executes a SQL query to interact with the MySQL database.
5. **Data Storage & Retrieval:** The query result is stored in or retrieved from the relevant database table (e.g., a new entry in service_requests). The result is sent back to the frontend.
6. **User Access:** The frontend updates dynamically, allowing users to view the results of their actions (e.g., the student sees their new request in the "My Documents" list).

System Overview

Campus Sphere is a comprehensive, web-based college management platform that streamlines administrative and academic processes. The system provides role-based dashboards and modules for students, teachers, parents, administrators, and Heads of Department (HODs), facilitating seamless interaction and data management across the institution.

Key Components

- **Role-Based Dashboards:** A central interface for each user role (AdminDashboard.tsx, StudentDashboard.tsx, etc.) to access relevant modules and at-a-glance statistics.
- **Core Modules:**
 - **User Management** – (Admin) Create, update, and manage all user accounts.
 - **Document Request System** – (Student/Admin) Allows students to request official documents and admins to process them.
 - **Fee Management** – (Student/Admin/Parent) Manage, view, and process fee payments.

- **Academic & Grade Management** – (Teacher/Student) Enables teachers to enter grades and students to view their academic records.
- **Parent-Student Linking** – (Admin) A system for linking parent accounts to student accounts to view their progress.
- **Profile & Settings:** A common area for all users to manage their personal information and security settings.

Technology Stack

- **Backend:** PHP – Manages all API requests, server-side logic, and database interactions.
- **Frontend:** React, TypeScript, Vite, Tailwind CSS – Builds a modern, fast, and responsive user interface.
- **Database:** MySQL – Stores all application data in a relational format.
- **IDE:** Visual Studio Code – A flexible development environment.

User Management

- Secure registration and login system with credential verification against the database.
- Role-based access control (RBAC) implemented throughout the frontend and backend to ensure users can only access features appropriate for their role.

Accessibility & Scalability

- As a web-based system, it is accessible from any device with a browser, centralizing data and reducing manual administrative work.
- The component-based frontend and modular backend architecture allow for easy expansion with new features and scaling to handle more users.

Unified Modeling Language (UML)

The Unified Modeling Language (UML) is a standardized visual language used for designing and modeling software systems. It helps developers, designers, and stakeholders visualize a system's structure, behavior, and interactions using diagrams. UML consists of structural and behavioral diagrams that define system components and their interactions.

A Use Case Diagram visually represents how different users (actors) interact with a system.

It includes:

- **Actors:** External entities (users or systems) that perform actions.
- **Use Cases:** Functionalities represented as ovals.
- **Relationships:** Lines connecting actors to use cases, showing interactions.

Class Diagram

A Class Diagram shows the system's structure by defining its classes, attributes, and relationships.

- **Classes:** Represent system components with attributes and methods.
- **Relationships:**
 - **Association:** One class interacts with another.
 - **Dependency:** One class depends on another.

Data Flow Diagram (DFD)-Level 0

A DFD Level 0 represents the entire system as a single process interacting with external entities.

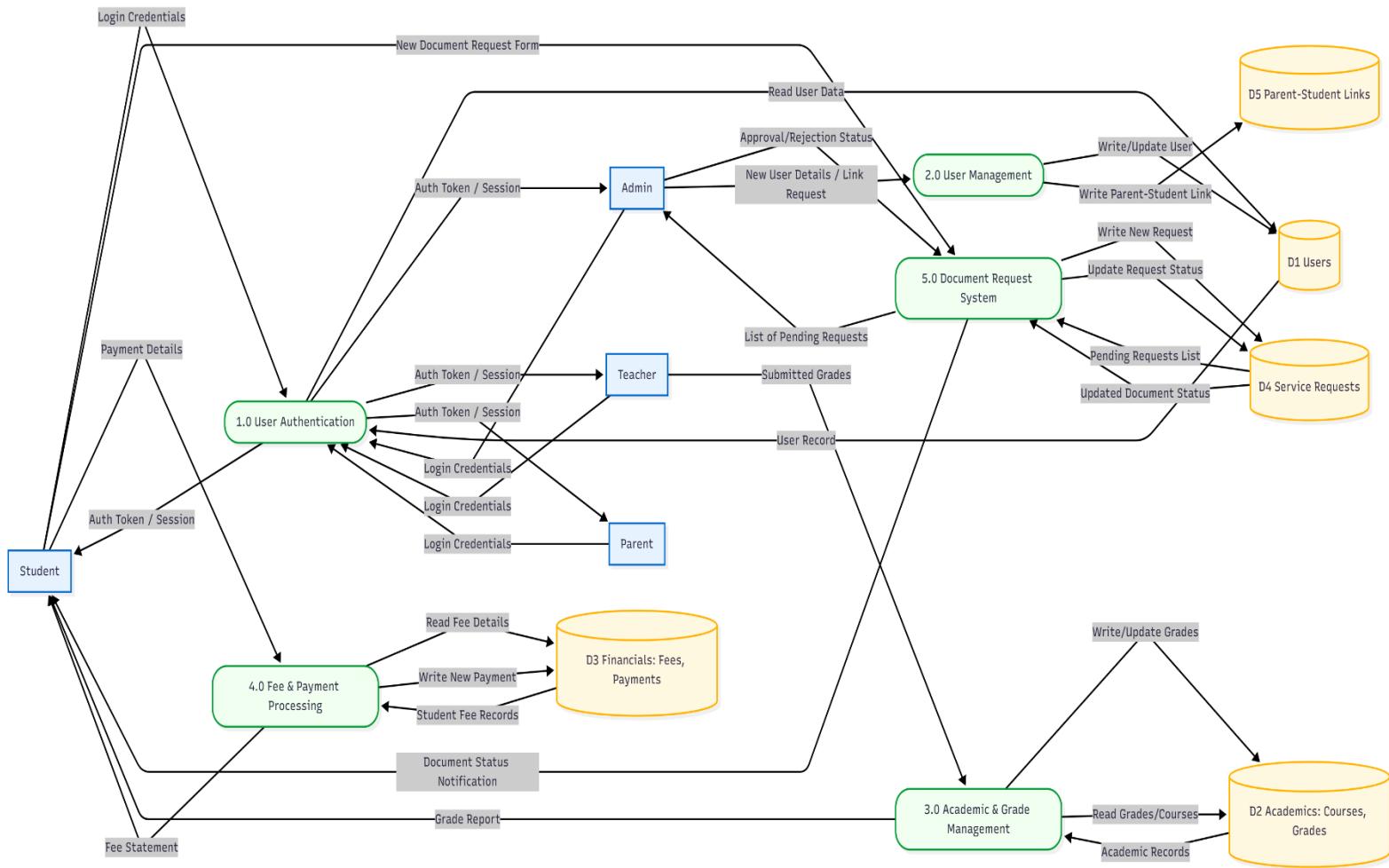
- **External Entities:** Users or external systems that exchange data.
- **Processes:** The central system handling input and output.
- **Data Flow:** Arrows representing data movement between entities and the system.

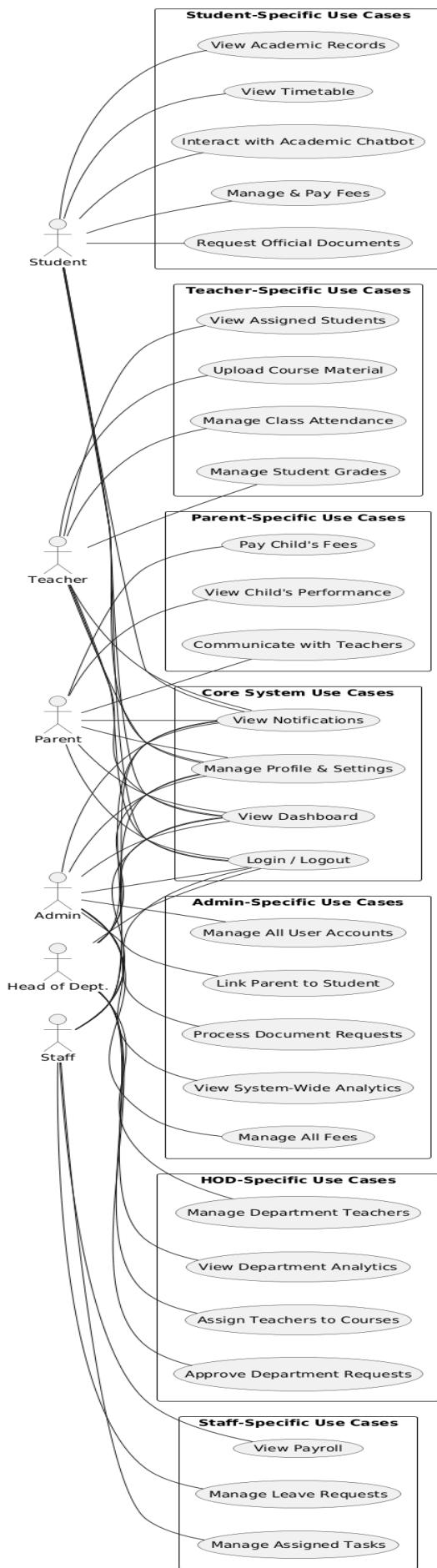
Data Flow Diagram (DFD)-Level 1

A DFD Level 1 breaks down the system into multiple sub-processes.

- **Sub-processes:** Individual tasks like data input, processing, and storage.
- **Data Stores:** Represent where information is saved (e.g., database tables).
- **Data Flow:** Shows how data moves between sub-processes, entities, and data stores

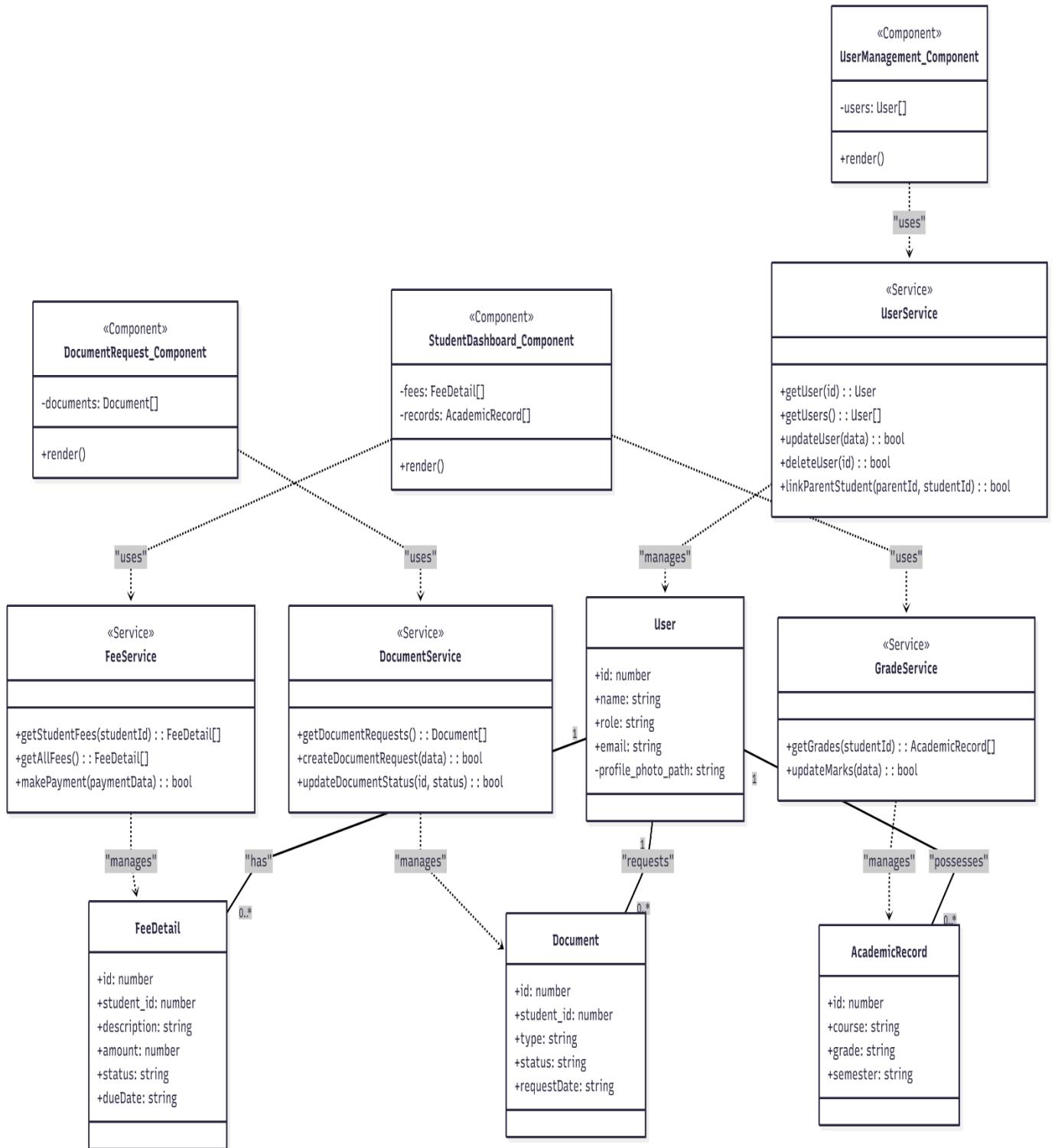
DATA FLOW DIAGRAM



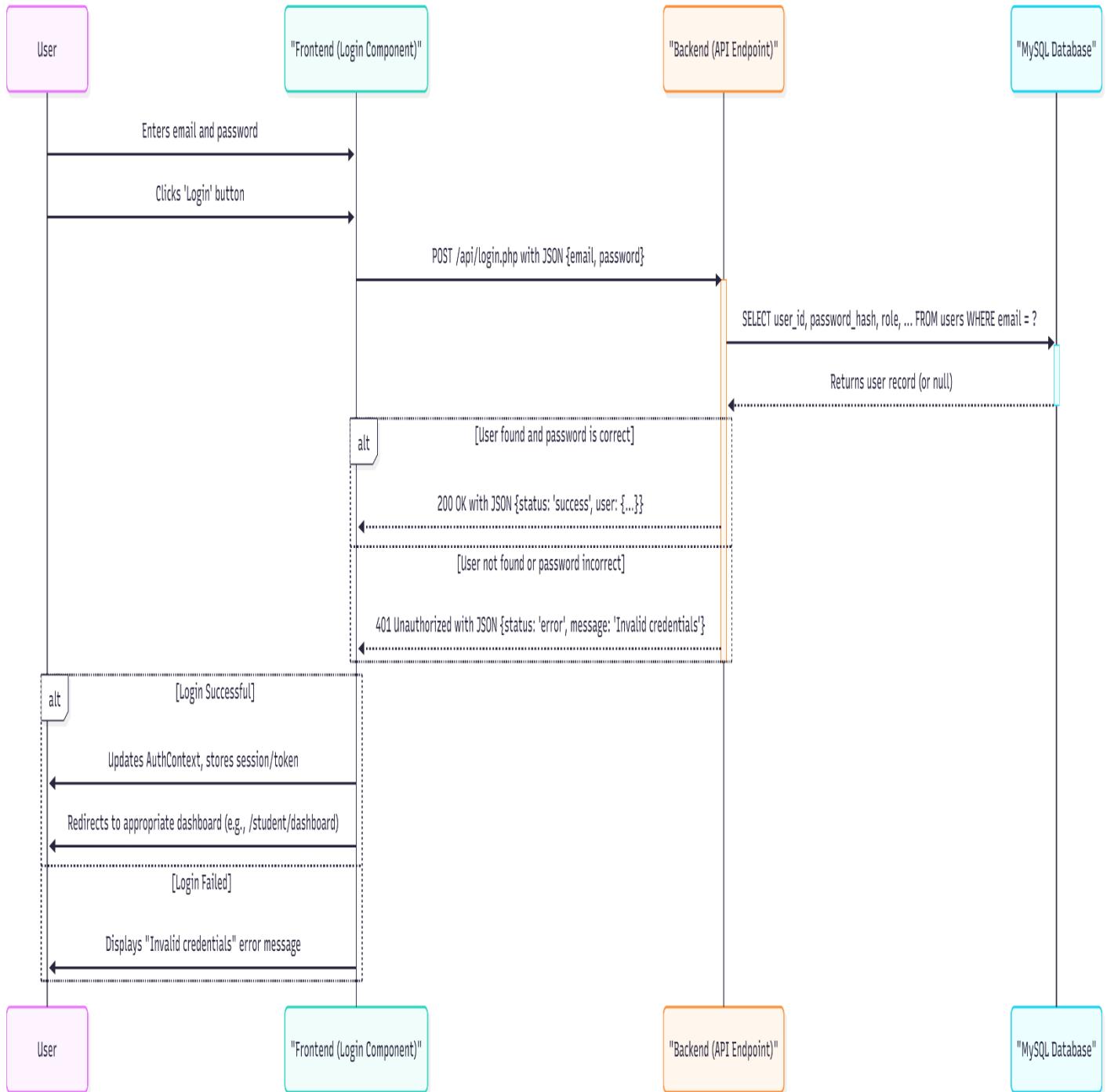


USE CASE DIAGRAM

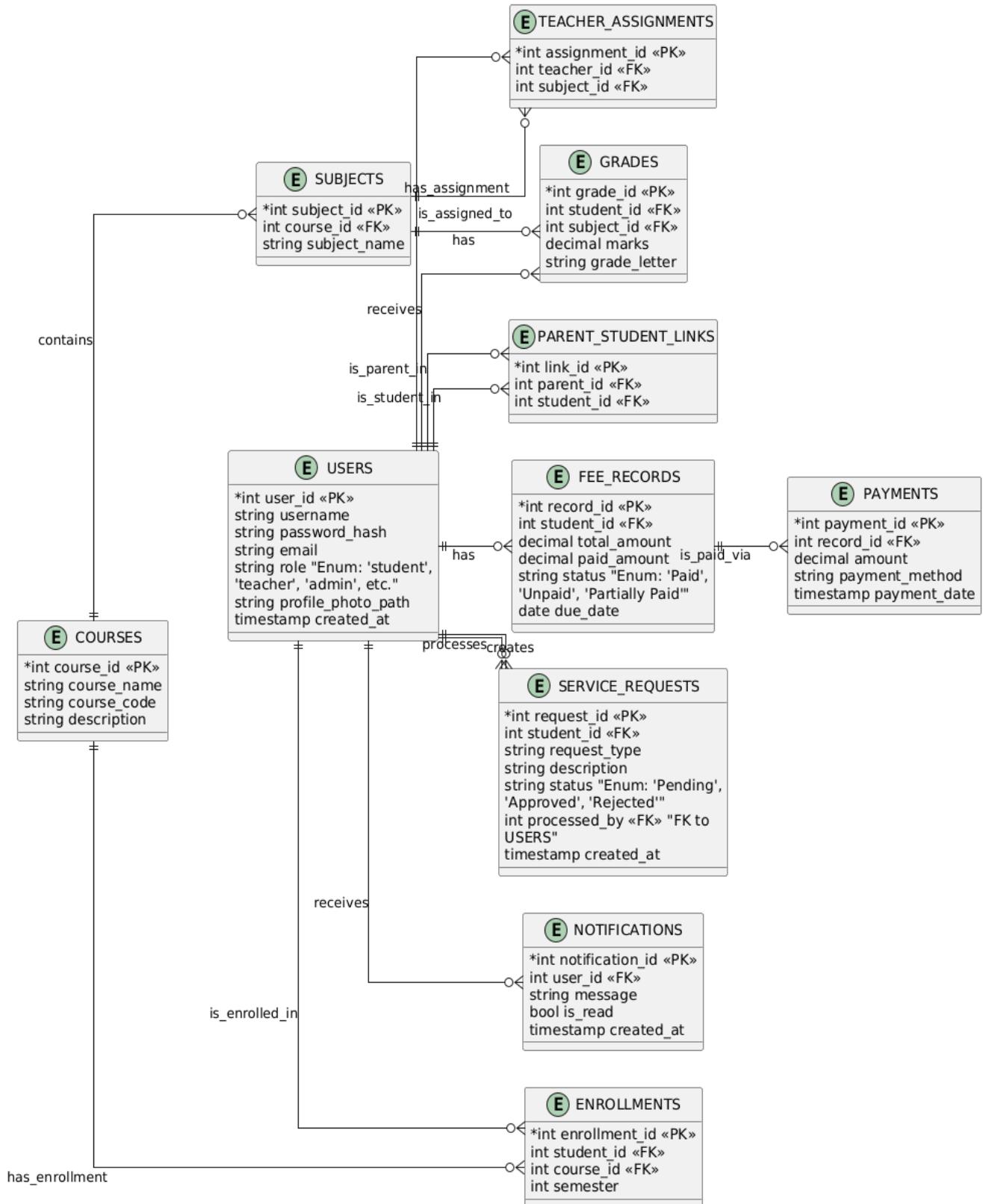
CLASS DIAGRAM



SEQUENCE DIAGRAM



ER DIAGRAM



SOFTWARE INSTALLATION

Software Installation:

The following software tools and dependencies are required for the development and deployment of the Smart Campus Sphere web-based campus management system. The platform uses a React + Vite frontend and a PHP–MySQL backend powered by XAMPP.

1. Frontend Technologies:

- Node.js (18+)
- Required to run the React development server.
- Provides npm (Node Package Manager) to install frontend dependencies.
- React (18.x)
- Framework used for building dynamic user interfaces.
- Handles routing, real-time UI updates, and component rendering.
- Vite
- A fast frontend build tool providing a lightning-fast dev server.
- Used for bundling the React application for production.
- NPM Packages Installed
- react, react-dom – Core React library.
- react-router-dom – Routing and navigation.
- axios or fetch – API integration with PHP backend.
- tailwindcss – Fast CSS utility framework.
- lucide-react – Icons for the UI.
- @types/react, @types/node – TypeScript definitions (if TS enabled).
- These packages ensure smooth rendering, styling, and data communication.

2. Backend Technologies:

- PHP (7+ recommended)
- Main backend scripting language.
- Handles authentication, validation, file uploads, and role-based operations.
- Generates JSON responses for the React frontend.
- XAMPP
- Provides:
 - Apache Server to host PHP backend APIs.
 - MySQL Server to store campus records.
 - phpMyAdmin for database management.

- MySQL Database
- Stores:
- User accounts
- Student information
- Faculty & parent details
- Academic records
- Fee information
- Document requests
- Notifications & announcements
- Required PHP Extensions
- These come pre-configured in XAMPP:
- mysqli
- openssl
- json
- fileinfo
- session

3. Additional Tools:

- For testing backend APIs.
- VS Code
- Used for writing React and PHP code with syntax support.
- Git
- Manages project version control and deployments.

Software Installation & Environment Setup

The following steps explain how to set up the complete development environment for Smart Campus Sphere.

Step 1: Install Node.js:

- Check if Node is installed:
- node -v
- If not installed, download from:
<https://nodejs.org>

Step 2: Install Frontend Dependencies:

- Navigate to the project folder:
- cd smart-campus-sphere
- Install all required packages:

- npm install
- This installs React, Vite, Tailwind CSS, and all other dependencies.

Step 3: Configure Vite Development Server:

- Start the frontend development server:
- npm run dev
- Vite will run the app on:
- http://localhost:5173/

Step 4: Install XAMPP:

- Download XAMPP from:
<https://www.apachefriends.org/>
- Install and start:
- Apache (for PHP backend)
- MySQL (for database)

Step 5: Configure PHP Backend:

- Place your backend folder inside:
- C:\xampp\htdocs\campus-backend\
- Start Apache and MySQL from XAMPP.
- Test backend:
- http://localhost/campus-backend/api/test.php

Step 6: Create MySQL Database:

- Open phpMyAdmin:
- http://localhost/phpmyadmin/
- Create a new database:
- campus_sphere
- Import your SQL schema:
- Go to Import
- Select the SQL file from your backend
- Click Go
- This creates all required tables:
- users
- academic_records
- fee_details
- notifications
- documents
- parent_records

- staff_records
- tasks

Step 7: Configure Backend Environment Variables:

Create a config.php file:

```
<?php

$host = "localhost";

$user = "root";

$pass = "";

$db  = "campus_sphere";

$conn = new mysqli($host, $user, $pass, $db);

if($conn->connect_error){

die("Connection failed: " . $conn->connect_error);

}

?>
```

Step 8: Connect React Frontend with PHP Backend:

- Set API base URL in your React configuration (usually inside services/api.js):
- export const BASE_URL = "http://localhost/campus-backend/";

Step 9: Test Full System:

- Open frontend:
http://localhost:5173
- Login as student/teacher/admin.
- Ensure:
- Dashboard loads correctly
- API calls reach the backend
- Data is fetched from MySQL
- Profile uploads work
- Document requests are stored

Step 10: Build Frontend for Deployment:

- For production build:

- npm run build
- Vite generates optimized files inside:
 - dist/
- These can be hosted on:
 - Apache Server
 - Nginx
 - Any static hosting platform

Source Code And Testing

Coding:

In the **OrbitCampus** project, coding forms the core foundation that enables seamless interaction between the frontend user interface, backend logic, and database management systems. The system is built using a modern tech stack consisting of React + Vite, PHP, and MySQL, ensuring high performance, data security, and a smooth user experience.

Below are the primary areas where coding is applied:

1. Frontend Development (React JS, HTML, CSS, JavaScript):

a. React (Main Frontend Framework):

React is used to build the entire user interface using component-based architecture.

Key Implementations:

- Page routing using react-router-dom.
- Dynamic dashboards for students, faculty, and administrators.
- State management using React Hooks (useState, useEffect, useContext).
- Fetching backend data through Axios or Fetch API.
- UI updates without reloading the page (SPA architecture).

b. HTML (Structure Layer):

Used within React components (JSX) to define:

- Login forms
- Dashboard layouts

- Tables, cards, modals, and content sections
- Document upload interfaces

c. CSS (Styling & UI Design):

CSS is used to create:

- Responsive layouts
- Color themes
- Component spacing and alignment
- Modern UI elements (buttons, cards, grids)
- Smooth animations and transitions

d. JavaScript (Logic & Interaction):

JavaScript powers:

- Form validations
- API requests to the PHP backend
- Displaying real-time data
- User interactions (clicks, navigation, menu toggles)
- Rendering dynamic elements inside React components

2. Backend Development (PHP):

PHP functions as the server-side engine of OrbitCampus.

Key Backend Coding Responsibilities:

a. REST API Development:

PHP scripts are written to create API endpoints for:

- Login authentication
- Fetching student/faculty/admin dashboards
- Document requests and uploads
- Profile updates

- Notification retrieval

b. User Authentication:

- Secure login using hashed passwords
- Session-based authentication
- Role-based access (student, admin, faculty)

c. Business Logic Implementation:

Backend code handles:

- Validating form data
- Approving or rejecting document requests
- Updating fee records
- Managing announcements and notifications

d. Error Handling & Security:

PHP includes:

- SQL injection prevention (prepared statements)
- Input sanitization
- Session handling
- File upload restrictions

3. Database Layer (MySQL):

MySQL is used to store all essential campus data.

Key Coding Activities:

- Writing SQL queries for CRUD operations
- Designing relational tables for users, documents, notices, fees, and profiles
- Managing foreign key relationships
- Optimizing query performance
- Integrating MySQL with PHP using mysqli or PDO

Examples of data stored:

- Login credentials
- Student & faculty details
- Uploaded documents
- Fee structures
- Announcements

4. Data Communication Layer (JSON):

JSON format is used for exchanging data between:

React Frontend ↔ PHP Backend

Examples:

- Sending login credentials from React to PHP
- Returning dashboard data in structured JSON arrays
- Sending request statuses and form responses

JSON ensures lightweight, secure, and fast data transfer.

5. Development & Build Tooling (Vite):

Role of Vite in Coding

- Provides ultra-fast development environment
- Hot Module Reloading (HMR) for instant UI preview
- Optimizes React code for production builds
- Bundles CSS, JS, and assets efficiently

LOGIN PAGE

The screenshot displays two instances of the Firebase Studio interface, each showing the 'Login.tsx' file from a React application named 'CampusSphere'.

Top Editor (Initial State):

```
frontend > src > components > Login > Login.tsx
const Login: React.FC = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState('');
  const [loading, setLoading] = useState(false);
  const { login } = useAuth();

  const handle.onSubmit = async (e: React.FormEvent) => {
    e.preventDefault();
    setError('');
    setLoading(true);

    try {
      const success = await login(email, password);
      if (!success) {
        setError('Invalid email or password.');
      }
    } catch (err) {
      console.error('Login component error:', err);
      setError('Login failed. Please check the console for more details.');
    } finally {
      setLoading(false);
    }
  };

  return (
    <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 flex items-center justify-center p-4">
      <div className="max-w-md w-full space-y-8">
        <div className="text-center">
          <div className="flex justify-center">
            <div className="bg-blue-600 p-3 rounded-full">
              <GraduationCap className="h-8 w-8 text-white" />
            </div>
          </div>
          <h2 className="mt-6 text-3xl font-bold text-gray-900">Campus Sphere</h2>
          <p className="mt-2 text-sm text-gray-400">Your centralized college management platform</p>
        </div>
      </div>
    </div>
  );
}

export default Login;
```

Bottom Editor (Refactored State):

```
frontend > src > components > Login > Login.tsx
const Login: React.FC = () => {
  const { login } = useAuth();

  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [error, setError] = useState('');
  const [loading, setLoading] = useState(false);

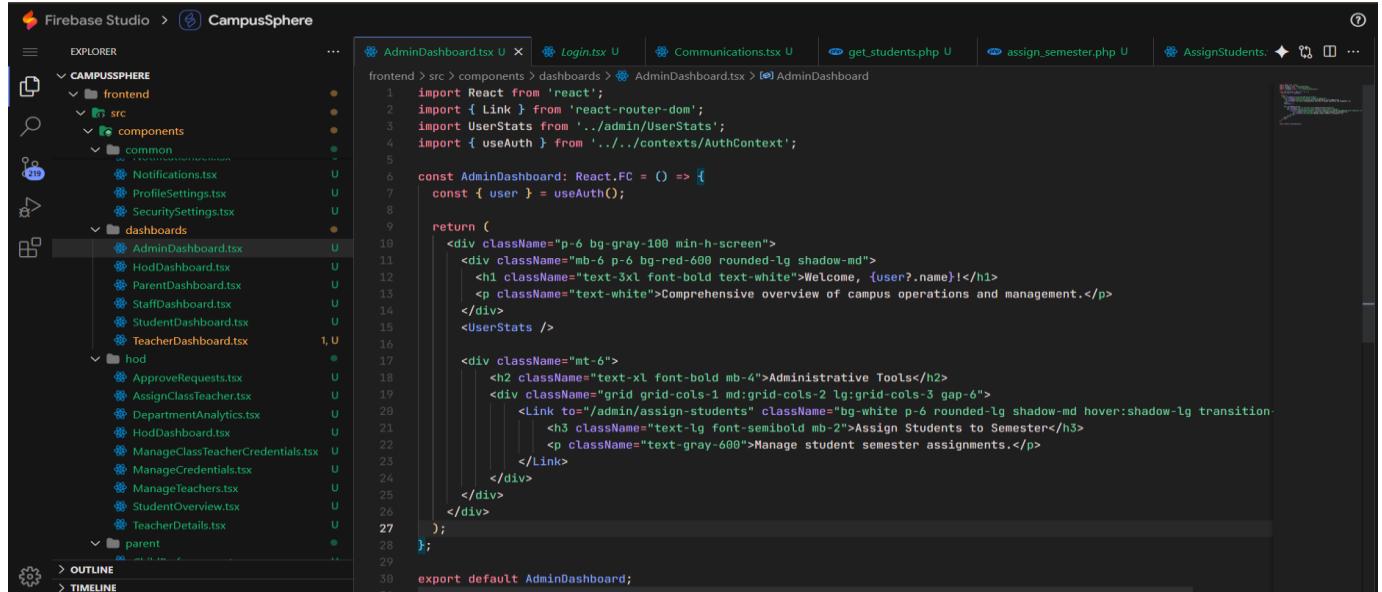
  const handle.onSubmit = async (e: React.FormEvent) => {
    e.preventDefault();
    setError('');
    setLoading(true);

    try {
      const success = await login(email, password);
      if (!success) {
        setError('Invalid email or password.');
      }
    } catch (err) {
      console.error('Login component error:', err);
      setError('Login failed. Please check the console for more details.');
    } finally {
      setLoading(false);
    }
  };

  return (
    <div className="min-h-screen bg-gradient-to-br from-blue-50 to-indigo-100 flex items-center justify-center p-4">
      <div className="max-w-md w-full space-y-8">
        <div className="text-center">
          <div className="flex justify-center">
            <div className="bg-blue-600 p-3 rounded-full">
              <GraduationCap className="h-8 w-8 text-white" />
            </div>
          </div>
          <h2 className="mt-6 text-3xl font-bold text-gray-900">Campus Sphere</h2>
          <p className="mt-2 text-sm text-gray-400">Your centralized college management platform</p>
        </div>
      </div>
    </div>
  );
}

export default Login; // Now using the centralized login logic from AuthContext
```

ADMIN DASHBOARD



Firebase Studio > CampusSphere

EXPLORER

- CAMPUSSPHERE
 - frontend
 - src
 - components
 - common
 - Notifications.tsx
 - ProfileSettings.tsx
 - SecuritySettings.tsx
 - dashboards
 - AdminDashboard.tsx
 - HodDashboard.tsx
 - ParentDashboard.tsx
 - StaffDashboard.tsx
 - StudentDashboard.tsx
 - TeacherDashboard.tsx
 - ApproveRequests.tsx
 - AssignClassTeacher.tsx
 - DepartmentAnalytics.tsx
 - HodDashboard.tsx
 - ManageClassTeacherCredentials.tsx
 - ManageCredentials.tsx
 - ManageTeachers.tsx
 - StudentOverview.tsx
 - TeacherDetails.tsx
 - parent

frontend > src > components > dashboards > AdminDashboard.tsx

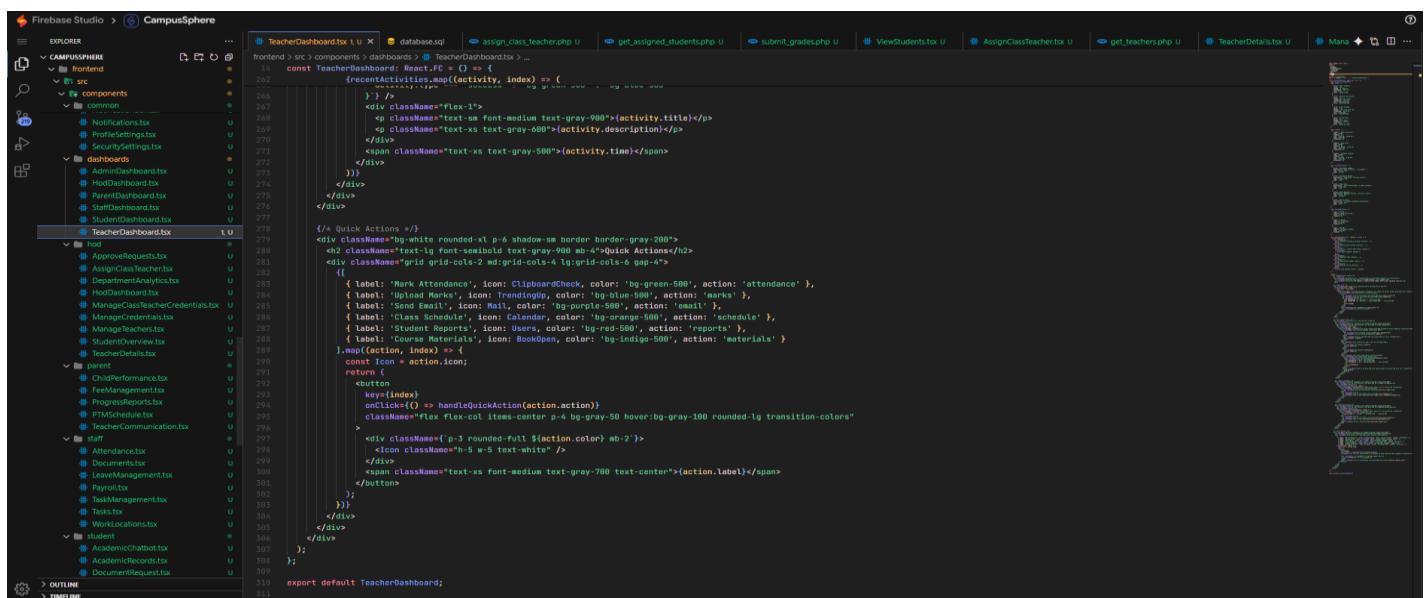
```
import React from 'react';
import { Link } from 'react-router-dom';
import UserStats from '../admin/UserStats';
import { useAuth } from '../../contexts/AuthContext';

const AdminDashboard: React.FC = () => {
  const { user } = useAuth();

  return (
    <div className="p-6 bg-gray-100 min-h-screen">
      <div className="mb-6 p-6 bg-red-600 rounded-lg shadow-md">
        <h1 className="text-3xl font-bold text-white">Welcome, {user?.name}!</h1>
        <p className="text-white">Comprehensive overview of campus operations and management.</p>
      </div>
      <UserStats />

      <div className="mt-6">
        <h2 className="text-xl font-bold mb-4">Administrative Tools</h2>
        <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-3 gap-6">
          <Link to="/admin/assign-students" className="bg-white p-6 rounded-lg shadow-md hover:shadow-lg transition">
            <h3 className="text-lg font-semibold mb-2">Assign Students to Semester</h3>
            <p className="text-gray-600">Manage student semester assignments.</p>
          </Link>
        </div>
      </div>
    </div>
  );
};

export default AdminDashboard;
```



Firebase Studio > CampusSphere

EXPLORER

- CAMPUSSPHERE
 - frontend
 - src
 - components
 - common
 - Notifications.tsx
 - ProfileSettings.tsx
 - SecuritySettings.tsx
 - dashboards
 - AdminDashboard.tsx
 - HodDashboard.tsx
 - ParentDashboard.tsx
 - StaffDashboard.tsx
 - StudentDashboard.tsx
 - TeacherDashboard.tsx
 - ApproveRequests.tsx
 - AssignClassTeacher.tsx
 - DepartmentAnalytics.tsx
 - HodDashboard.tsx
 - ManageClassTeacherCredentials.tsx
 - ManageCredentials.tsx
 - ManageTeachers.tsx
 - StudentOverview.tsx
 - TeacherDetails.tsx
 - parent
 - ChildPerformance.tsx
 - FeeManagement.tsx
 - ProgressReports.tsx
 - PTMSchedule.tsx
 - TeacherCommunication.tsx
 - staff
 - Attendance.tsx
 - Documents.tsx
 - LeaveManagement.tsx
 - Payroll.tsx
 - TaskManagement.tsx
 - TaskStatus.tsx
 - WorkLocations.tsx
 - student
 - AcademicChatbot.tsx
 - AcademicRecords.tsx
 - DocumentRequest.tsx

frontend > src > components > dashboards > TeacherDashboard.tsx

```
const TeacherDashboard: React.FC = () => {
  const recentActivities = [
    {activity: 'New Student Enrolment', index: 0, time: '1 hour ago'},
    {activity: 'Attendance Marking', index: 1, time: '2 hours ago'},
    {activity: 'Assignment Submission', index: 2, time: '3 hours ago'},
    {activity: 'Grade Update', index: 3, time: '4 hours ago'}
  ];

  const quickActions = [
    {label: 'Mark Attendance', icon: 'ClipboardCheck', color: 'bg-green-500', action: 'attendance'},
    {label: 'Upload Marks', icon: 'TrendingUp', color: 'bg-blue-500', action: 'marks'},
    {label: 'Send Email', icon: 'Mail', color: 'bg-purple-500', action: 'email'},
    {label: 'Class Schedule', icon: 'Calendar', color: 'bg-orange-500', action: 'schedule'},
    {label: 'View Reports', icon: 'BarChart', color: 'bg-red-500', action: 'reports'},
    {label: 'Course Materials', icon: 'BookOpen', color: 'bg-indigo-500', action: 'materials'}
  ].map((action, index) => {
    const Icon = action.icon;
    return (
      <div key={index}>
        <button onClick={() => handleQuickAction(action.action)}>
          <div className="flex flex-col items-center p-3 rounded-full $({action.color}) mb-2">
            <Icon className="h-5 w-5 text-white" />
            <span className="text-xs font-medium text-gray-700 text-center">{action.label}</span>
          </div>
        </button>
      </div>
    );
  });
};

export default TeacherDashboard;
```

TEACHER DASHBOARD

The screenshot shows the Firebase Studio interface with the project 'CampusSphere' selected. The left sidebar displays the project structure under 'EXPLORER'. The main area is a code editor for the file 'TeacherDashboard.tsx'. The code is a React component that handles various actions like attendance, marks, email, schedule, reports, and materials. It includes logic for opening different interfaces and handling user input for email subjects. The right side of the interface shows a detailed 'PROBLEMS' panel with numerous errors and warnings related to the code, such as 'Expected identifier, got 'stat.title'' and 'Expected identifier, got 'stat.value''. The bottom navigation bar includes 'OUTLINE', 'TIMELINE', and other tabs.

```
import React, { FC } from 'react';
import { useState } from 'react';
import { toast } from 'react-toastify';
import { useAuth } from '../../context/authContext';
import { useFirestore } from '../../context/firestoreContext';
import { useUser } from '../../context/userContext';
import { useDocument } from '../../context/documentContext';
import { useClass } from '../../context/classContext';
import { useAttendance } from '../../context/attendanceContext';
import { useMarks } from '../../context/marksContext';
import { useEmail } from '../../context/emailContext';
import { useSchedule } from '../../context/scheduleContext';
import { useReports } from '../../context/reportsContext';
import { useMaterials } from '../../context/materialsContext';
import { useStudentOverview } from '../../context/studentOverviewContext';
import { useTeacherDetails } from '../../context/teacherDetailsContext';

const TeacherDashboard: FC = () => {
  const [user] = useAuth();
  const [firestore] = useFirestore();
  const [userRef] = useUser();
  const [documentRef] = useDocument();
  const [classRef] = useClass();
  const [attendanceRef] = useAttendance();
  const [marksRef] = useMarks();
  const [emailRef] = useEmail();
  const [scheduleRef] = useSchedule();
  const [reportsRef] = useReports();
  const [materialsRef] = useMaterials();
  const [studentOverviewRef] = useStudentOverview();
  const [teacherDetailsRef] = useTeacherDetails();

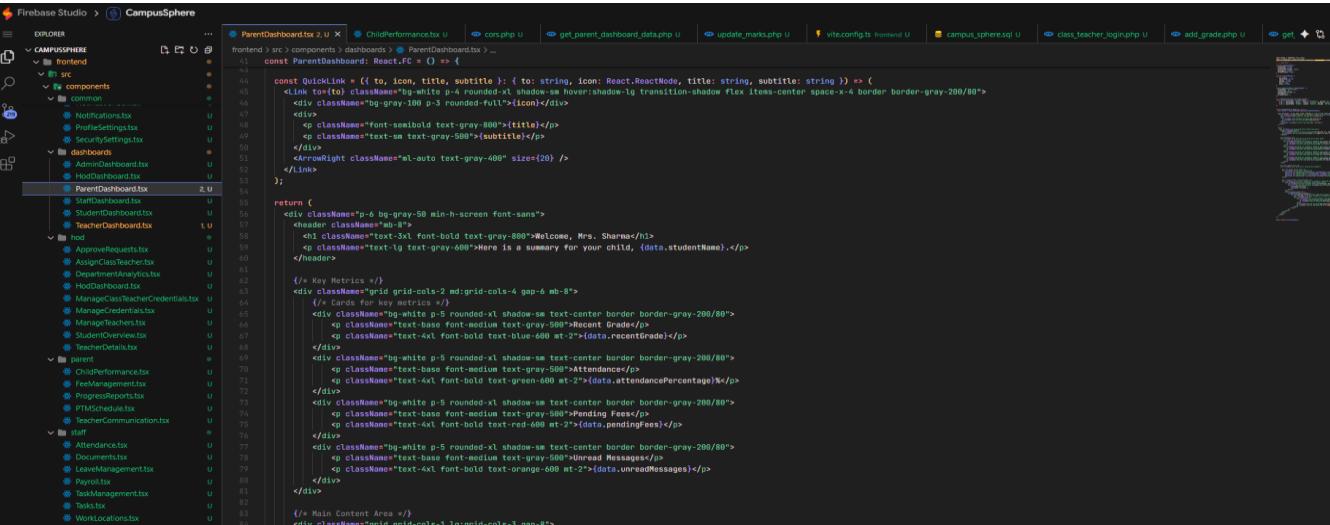
  const handleQuickAction = (action: string) => {
    switch (action) {
      case 'attendance':
        alert('Opening attendance marking interface...');
        break;
      case 'marks':
        alert('Opening marks upload interface...');
        break;
      case 'email':
        const subject = prompt('Enter email subject:');
        if (subject) {
          emailRef.sendEmail(subject);
          alert('Email composer opened!');
        }
        break;
      case 'schedule':
        alert('Opening class schedule...');
        break;
      case 'reports':
        alert('Generating student reports...');
        break;
      case 'materials':
        alert('Opening course materials...');
        break;
      default:
        console.log(`Unknown action: ${action}`);
    }
  };

  return (
    <div className="p-6 space-y-6">
      </> Header </div>
      <div className="bg-gradient-to-r from-green-600 to-green-800 rounded-xl p-6 text-white">
        <h1 className="text-2xl font-bold mb-2">Good morning, {user?.name}!</h1>
        <p className="text-green-100">Ready to inspire minds today? Here's your teaching overview.</p>
      </div>

      </> Stats Card </div>
      <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
        {stats.map((stat, index) => {
          const Icon = stat.icon;
          return (
            <div key={index} className="bg-white rounded-xl p-6 shadow-sm border border-gray-200 hover:shadow-md transition-shadow">
              <div className="flex items-center justify-between">
                <div>
                  <span>{stat.title}</span>
                  <span>{stat.value}</span>
                  <span>{stat.meta}</span>
                </div>
                <span>{Icon}</span>
              </div>
            </div>
          );
        })}
      </div>
    </div>
  );
};

export default TeacherDashboard;
```

PARENT DASHBOARD



The screenshot shows the Firebase Studio interface with the code editor open for the `ParentDashboard` component. The code is written in React.js and uses styled-components for styling. The component structure includes sections for the dashboard header, key metrics (attendance, fees), and main content areas for child performance, fee management, and reports.

```
const DashboardHeader = ({ to, icon, title, subtitle }) => {
  const QuestionLine = ({ to, icon, title, subtitle }) => (
    <div href={to} className="bg-white p-5 rounded-lg shadow-sm transition-shadow flex items-center space-x-4 border border-gray-200/80">
      <div>
        <img alt={icon} alt="Icon for the dashboard section" />
        <div>
          <p>{title}</p>
          <p>{subtitle}</p>
        </div>
      </div>
      <ArrowRight className="ml-auto text-gray-400" size={20} />
    </div>
  );
  return (
    <div>
      <div>
        <h1>Welcome, Mrs. Sharma!</h1>
        <p>Here is a summary for your child, {data.studentName}.</p>
      </div>
      <div>
        <h2>Key Metrics</h2>
        <div>
          <div>
            <h3>Attendance</h3>
            <p>Recent Grade:</p>
            <p>Attendance Percentage:</p>
          </div>
          <div>
            <h3>Fees Management</h3>
            <p>Pending Fees:</p>
            <p>Unread Messages:</p>
          </div>
        </div>
        <div>
          <h2>Main Content Area</h2>
          <div>
            <div>
              <h3>Child Performance</h3>
              <QuickLink to="/parent/child-performance" icon="barChart" title="Child Performance" subtitle="View detailed analytics" />
              <QuickLink to="/parent/fees" icon="creditCard" title="Fee Management" subtitle="Pay fees and view history" />
              <QuickLink to="/parent/reports" icon="dropdown" title="Program Reports" subtitle="Download term reports" />
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}
```

The screenshot shows a developer's environment with several open tabs and windows. The left sidebar lists project components like 'front end', 'components', 'dashboards', 'easboards', 'hoc', 'parent', 'staff', 'student', and 'teacher'. The main area has two code editors. The top editor shows 'ParentDashboard.tsx' with React code for a dashboard. The bottom editor shows 'ChildPerformance.tsx' with React code for a child performance page. A terminal window at the bottom shows command-line history. The status bar at the bottom right includes 'Preview', 'Gmails', 'Ln 125, Col 1', 'Spaces: 2', 'UTR-8', 'LF', 'TypeScript JSX', 'Layouts', and 'US'.

```
const ParentDashboard: React.FC<{}> = () => {
  return (
    </>
    </div>
  );
}

// Left Column: Quick Links
<div className="ui-grid-span-1-space-y-4">
  <QuickLink to="/parent/child-performance" icon="barChart" className="text-blue-500" /> title="Child Performance" subtitle="View detailed analytics"
  <QuickLink to="/parent/fees" icon="creditCard" className="text-red-500" /> title="Fee Management" subtitle="Pay fees and view history"
  <QuickLink to="/parent/attendance" icon="calendar" className="text-green-500" /> title="Progress Reports" subtitle="Download term reports"
  <QuickLink to="/parent/gpa" icon="calendar" className="text-purple-500" /> title="PTM Schedule" subtitle="Book parent-teacher meetings"
</div>

// Right Column: Recent Messages
<div className="ui-grid-span-4-space-y-4 shadow-sm border border-gray-200/80">
  <div className="flex justify-between items-center mb-4">
    <h2 className="text-xxl font-bold text-gray-800">Recent Messages</h2>
    <Link to="/parent/communication" className="text-sm font-medium text-blue-600 hover:underline flex items-center space-x-1">
      <span>View All</span>
      <ChevronRight size="16" />
    </Link>
  </div>
  <div className="flex space-y-4">
    <div key={msg.id} className="flex items-start space-x-4 p-4 bg-gray-50 rounded-lg hover:bg-gray-100 transition-colors">
      <div className="w-10 h-10 rounded-full bg-purple-600 text-white flex-shrink-0 flex items-center justify-center font-bold">{msg.avatar}</div>
      <div className="flex flex-grow justify-between items-baseline">
        <div className="font-semibold text-gray-900">{msg.teacherName}</div>
        <div className="text-sm text-gray-500">{msg.subject}</div>
        <div className="text-sm text-gray-500">{msg.message}</div>
      </div>
    </div>
  </div>
</div>
};

export default ParentDashboard;
```

STAFF DASHBOARD

```
frontend > src > components > dashboards > ParentDashboard.tsx > ...  
14 const Staffashboard: React.FC<{}> = () => {  
15   const handleLeaveAction = (action: string) => {  
16     switch (action) {  
17       case 'markAttendance':  
18         const now = new Date();  
19         const timeString = now.toLocaleTimeString(), { hour: '2-digit', minute: '2-digit' };  
20         alert(`Attendance marked successfully at ${timeString}`);  
21         break;  
22       case 'reportIssue':  
23         const issue = prompt('Describe the issue you want to report:');  
24         if (issue) {  
25           alert(`Issue reported successfully. Your supervisor will be notified.`);  
26         }  
27         break;  
28       case 'Tasks':  
29         alert('Opening task management interface...');  
30         break;  
31       case 'Leave':  
32         const reason = prompt('Enter reason for leave request:');  
33         if (reason) {  
34           alert(`Leave request submitted successfully.`);  
35         }  
36         break;  
37       case 'paystip':  
38         alert('Downloading latest paystip...');  
39         break;  
40       case 'Supervisor':  
41         const message = prompt('Enter message for supervisor:');  
42         if (message) {  
43           alert(`Message sent to supervisor successfully.`);  
44         }  
45         break;  
46       default:  
47         console.log('Unknown action:', action);  
48     }  
49   };  
50  
51   return (  
52     <div className="p-6 space-y-6">  
53       </> Header </>  
54       <div className="bg-gradient-to-r from-purple-600 to-purple-800 rounded-xl p-6 text-white">  
55         <h1 className="text-2xl font-bold mb-2">Welcome back, {user?.name}!</h1>  
56         <p className="text-purple-100">Here's your work overview and task assignments for today.</p>  
57       </div>  
58  
59       </> Stats Grid </>  
60       <div className="grid grid-cols-3 md:grid-cols-2 lg:grid-cols-4 gap-6">  
61         <div>({stat._index})</div>  
62       </div>  
63     </div>  
64   );  
65 }  
66  
67 export default ParentDashboard;
```

The screenshot shows the Firebase Studio interface with the project 'CampusSphere' selected. The code editor displays the 'StaffDashboard.tsx' file. The code is a React component that renders a dashboard for staff. It includes sections for recent activities, quick actions, and a grid of various management tasks. The code uses styled-components for styling and handles user interactions like clicking on quick actions.

```
const StaffDashboard: React.FC<{}> = ({recentActivities, mapAction, index}) => {
  return (
    <div>
      <div>
        <div>
          <div>
            <div>
              <div><h2>Recent Activities</h2></div>
              <div>
                {recentActivities.map((activity, index) => (
                  <div>
                    <div>{activity.title}</div>
                    <div>{activity.description}</div>
                    <div>{activity.time}</div>
                  </div>
                ))}
              </div>
            </div>
            <div>
              <h3>Quick Actions</h3>
              <div>
                <div>
                  <div>
                    <div>
                      <div>
                        {mapAction(index).map((action, index) => {
                          const Icon = action.icon;
                          return (
                            <button key={index} onClick={() => handleQuickAction(action.action)}>
                              <div>
                                <Icon color={action.color} mb-2 />
                                <span>{action.label}</span>
                              </div>
                            </button>
                          );
                        })}
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}

export default StaffDashboard;
```

HOD DASHBOARD

```
frontend/src/components/headers/HodDashboard.tsx
47 const HodDashboard: React.FC = () => {
48   return (
49     <div className="bg-white p-6 rounded-2xl shadow-lg border flex items-start space-x-4">
50       <div className="bg-yellow-500 p-4 rounded-full text-yellow-600">Icon</div>
51       <div>
52         <p>Text, string, value: string | number, icon: React.ReactNode</p>
53         <p>Text, string | number</p>
54         <p>Text, string | number</p>
55       </div>
56     </div>
57   );
58
59   return (
60     <div className="p-8 bg-gray-100 min-h-screen font-sans">
61       <div>
62         <h1>Welcome back, Head of Department!</h1>
63         <p>A summary of your department's performance and metrics.</p>
64       </div>
65
66       <div>
67         <StatCard title="Total Students" value={mockStats.totalStudents} icon={editors.size(24)} />
68         <StatCard title="Courses Offered" value={mockStats.coursesOffered} icon={bookOpen.size(24)} />
69         <StatCard title="Pass Percentage" value={`${mockStats.passPercentage}%`} icon={award.size(24)} />
70         <StatCard title="Faculty Count" value={mockStats.facultyCount} icon={users.size(24)} />
71       </div>
72
73       <div>
74         <div>
75           <h2>Teacher Performance</h2>
76           <Table>
77             <thead>
78               <tr>
79                 <th>Name</th>
80                 <th>Subject</th>
81                 <th>Workload</th>
82               </tr>
83             <tbody>
84               <tr>
85                 <td>John Doe</td>
86                 <td>Mathematics</td>
87                 <td>40 hours</td>
88               </tr>
89             </tbody>
90           </Table>
91         </div>
92       </div>
93     </div>
94   );
95 }

```

The screenshot shows the Firebase Studio interface with the code editor open for the `HodDashboard` component. The code is a React component that displays a list of recent activities and links to various management pages. The code includes imports for React, useState, useEffect, and several utility classes. It uses state management to track the number of pending tasks and updates the UI accordingly.

```
import React, { useState, useEffect } from 'react';
import { useAuth } from 'SecurityAccess';
import { TaskList } from 'TaskManagement';
import { LeaveManagement } from 'LeaveManagement';
import { WorkLocations } from 'WorkLocations';
import { Payroll } from 'Payroll';
import { Attendance } from 'Attendance';
import { Documents } from 'Documents';
import { TeacherCommunication } from 'TeacherCommunication';

const HodDashboard = () => {
  const [mockTeachers, setMockTeachers] = useState([]);
  const [activity, setActivity] = useState([]);
  const [pendingTasks, setPendingTasks] = useState(0);
  const [teacherSubject, setTeacherSubject] = useState('');

  useEffect(() => {
    const mockTeachersData = [
      {teacher_id: 1, teacher_name: 'John Doe', subject: 'Math', pending_tasks: 2, last_update: '2023-10-01T12:00:00Z'},
      {teacher_id: 2, teacher_name: 'Jane Smith', subject: 'Science', pending_tasks: 1, last_update: '2023-10-01T14:00:00Z'},
      {teacher_id: 3, teacher_name: 'Mike Johnson', subject: 'English', pending_tasks: 3, last_update: '2023-10-01T15:00:00Z'}
    ];
    setMockTeachers(mockTeachersData);
  }, []);

  useEffect(() => {
    const mockActivitiesData = [
      {activity_id: 1, activity_name: 'Approve Requests', description: 'Pending leave requests', time: '2023-10-01T12:00:00Z'},
      {activity_id: 2, activity_name: 'Assign Class Teacher', description: 'Pending class assignments', time: '2023-10-01T14:00:00Z'},
      {activity_id: 3, activity_name: 'Manage Budget', description: 'Pending budget reviews', time: '2023-10-01T15:00:00Z'}
    ];
    setActivity(mockActivitiesData);
  }, []);

  const handleLogout = () => {
    useAuth().signOut();
  };

  const handleTaskClick = (task) => {
    if (task.pending_tasks === 0) {
      return;
    }
    setPendingTasks(pendingTasks - 1);
  };

  const handleTeacherClick = (teacher) => {
    setTeacherSubject(teacher.subject);
  };

  const handleActivityClick = (activity) => {
    switch (activity.activity_name) {
      case 'Approve Requests':
        window.location.href = '/hod/approve-requests';
        break;
      case 'Assign Class Teacher':
        window.location.href = '/hod/assign-class-teacher';
        break;
      case 'Manage Budget':
        window.location.href = '/hod/manage-budget';
        break;
    }
  };
}

export default HodDashboard;
```

STUDENT DASHBOARD

The screenshot shows the Firebase Studio interface with the 'CampusSphere' project selected. The left sidebar displays the project structure under 'EXPLORER', including components like 'common', 'dashboards', and 'parent'. The main area shows the code for 'StudentDashboard.tsx' with syntax highlighting for React and TypeScript. The right side features a detailed code viewer with line numbers, file navigation, and a search bar.

```
import React, { useState, useEffect } from 'react';
import CreditCard, { BookOpen, Home, FileText, TrendingUp, Calendar, AlertTriangle, Chat, Phone, } from 'tcomb-react';
import { useAuth } from '../../../../../contexts/AuthContext';

const StudentDashboard: React.FC = () => {
  const [user, setUser] = useState<any>(null);
  const [recentActivities, setRecentActivities] = useState<any>[]();
  const [upcomingEvents, setUpcomingEvents] = useState<any>[]();
  const [loading, setLoading] = useState<boolean>(true);
  const [error, setError] = useState<string>(null);

  useEffect(() => {
    const fetchDashboardData = async () => {
      if (!user) return;
      try {
        setLoading(true);
        const [statsRes, notificationsRes, eventsRes] = await Promise.all([
          fetch(`http://localhost:${port}/api/get_student_dashboard.php?user_id=${user.id}`),
          fetch(`http://localhost:${port}/api/notifications.php?user_id=${user.id}`),
          fetch(`http://localhost:${port}/api/calender/get_events.php`),
        ]);
      } catch (err) {
        setError(err.message);
      }
    };
    fetchDashboardData();
  }, [parent]);
}

export default StudentDashboard;
```

Firebase Studio > CampusSphere

```

EXPLORER
CAMPUSPHERE
  -> frontend
    -> src
      -> components
        -> common
          Notifications.tsx
          ProfileSettings.tsx
          SecuritySettings.tsx
        -> dashboards
          AdminDashboard.tsx
          HodDashboard.tsx
          ParentDashboard.tsx
          StaffDashboard.tsx
          StudentDashboard.tsx
          TeacherDashboard.tsx
        -> hod
          ApproveRequests.tsx
          AssignClassTeacher.tsx
          DepartmentAnalytics.tsx
          HodDashboard.tsx
          ManageClassTeacherCredentials.tsx
          ManageCredentials.tsx
          ManageTeachers.tsx
          StudentOverview.tsx
          TeacherDetails.tsx
        -> parent
          ChildPerformance.tsx
          FeeManagement.tsx
          ProgressReports.tsx
          PTMSchedule.tsx
          TeacherCommunication.tsx
        -> staff
          Attendance.tsx
          Documents.tsx
          LeaveManagement.tsx
          Payroll.tsx
          TaskManagement.tsx
          Tasks.tsx
          WorkLocations.tsx
        -> student
          AcademicChatbot.tsx
          AcademicRecords.tsx
          DocumentRequest.tsx

```

```

StudentDashboard.tsx U
frontend > src > components > dashboards > StudentDashboard.tsx
14 const StudentDashboard: React.FC = () => {
15   return (
16     <div className="p-6 space-y-6">
17       </> Header
18       <div className="bg-gradient-to-r from-blue-600 to-blue-800 rounded-xl p-6 text-white">
19         <h1 className="text-2xl font-bold mb-2">Welcome back, {user?.name}!</h1>
20         <p className="text-blue-100">Here's what's happening with your academic journey today.</p>
21       </div>
22
23       </> Stats Grid
24       <div className="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6">
25         {statItems.map((stat, index) => {
26           const icon = stat.icon;
27           return (
28             <div key={index} className="bg-white rounded-xl p-6 shadow-sm border border-gray-200 hover:shadow-md transition-shadow">
29               <div className="flex items-center justify-between">
30                 <div>
31                   <p className="text-sm font-medium text-gray-600 mb-1">{stat.title}</p>
32                   <p className="text-2xl font-bold text-gray-900">{stat.value}</p>
33                   <p className="text-sm">
34                     stat.changeType === 'positive' ? 'text-green-600' :
35                     stat.changeType === 'warning' ? 'text-orange-600' : 'text-gray-600'
36                   </p>
37                   <span>{stat.change}</span>
38                 </div>
39                 <div className="p-3 rounded-full ${stat.color}">
40                   <Icon className="h-6 w-6 text-white" />
41                 </div>
42               </div>
43             </div>
44           );
45         });
46       </div>
47
48       </> Main Content Grid
49       <div className="grid grid-cols-1 lg:grid-cols-3 gap-0">
50         </> Recent Activities
51         <div className="lg:col-span-2 bg-white rounded-xl p-6 shadow-sm border border-gray-200">
52           <h2 className="text-lg font-semibold text-gray-900 mb-4">Recent Activities</h2>
53           <div className="space-y-4">
54             {recentActivities.map((activity, index) => {
55               <div key={index} className="flex items-center space-x-4 p-3 bg-gray-50 rounded-lg">
56                 <div className="w-2 h-2 rounded-full ${activity.type === 'success' ? 'bg-green-500' :
57                   activity.type === 'warning' ? 'bg-orange-500' : 'bg-blue-500' }/>
58                 <div>
59                   <p className="text-sm font-medium text-gray-900">{activity.title}</p>
60                 </div>
61               </div>
62             });
63           </div>
64         </div>
65
66       </div>
67
68       </> Upcoming Events
69       <div className="bg-white rounded-xl p-6 shadow-sm border border-gray-200">
70         <h2 className="text-lg font-semibold text-gray-900 mb-4">Upcoming Events</h2>
71         <div className="space-y-4">
72           {upcomingEvents.map((event, index) => {
73             <div key={index} className="p-3 border border-gray-200 rounded-lg">
74               <div className="flex items-center space-x-2 mb-1">
75                 <Calendar className="h-4 w-4 text-blue-600" />
76                 <span className="text-xs px-2 py-1 rounded-full bg-blue-100 text-blue-700" style={{ color: event.start.toLocaleDateString() }}>{event.title}</span>
77               </div>
78               <div>
79                 <span>{event.type}</span>
80               </div>
81             </div>
82           });
83         </div>
84
85       </div>
86
87       </> Quick Actions
88       <div className="bg-white rounded-xl p-6 shadow-sm border border-gray-200">
89         <h2 className="text-lg font-semibold text-gray-900 mb-4">Quick Actions</h2>
90         <div className="grid grid-cols-2 md:grid-cols-4 lg:grid-cols-6 gap-4">
91           {quickActions.map((action, index) => {
92             const icon = action.icon;
93             return (
94               <button
95                 key={index}
96                 className="flex flex-col items-center p-4 bg-gray-50 hover:bg-gray-100 rounded-lg transition-colors"
97               >
98                 <div className="p-3 rounded-full ${action.color} mb-2">
99                   <Icon className="h-5 w-5 text-white" />
100                 </div>
101                 <span className="text-xs font-medium text-gray-700 text-center">{action.label}</span>
102               </button>
103             );
104           });
105         </div>
106       </div>

```

Firebase Studio > CampusSphere

```

EXPLORER
CAMPUSPHERE
  -> frontend
    -> src
      -> components
        -> common
          Notifications.tsx
          ProfileSettings.tsx
          SecuritySettings.tsx
        -> dashboards
          AdminDashboard.tsx
          HodDashboard.tsx
          ParentDashboard.tsx
          StaffDashboard.tsx
          StudentDashboard.tsx
          TeacherDashboard.tsx
        -> hod
          ApproveRequests.tsx
          AssignClassTeacher.tsx
          DepartmentAnalytics.tsx
          HodDashboard.tsx
          ManageClassTeacherCredentials.tsx
          ManageCredentials.tsx
          ManageTeachers.tsx
          StudentOverview.tsx
          TeacherDetails.tsx
        -> parent
          ChildPerformance.tsx
          FeeManagement.tsx
          ProgressReports.tsx
          PTMSchedule.tsx
          TeacherCommunication.tsx
        -> staff
          Attendance.tsx
          Documents.tsx
          LeaveManagement.tsx
          Payroll.tsx
          TaskManagement.tsx
          Tasks.tsx
          WorkLocations.tsx
        -> student
          AcademicChatbot.tsx
          AcademicRecords.tsx
          DocumentRequest.tsx

```

```

ers.tsx U
get_teacher_details.php U
update_user.php U
ProfileSettings.tsx U
Layout.tsx U
StudentOverview.tsx U
DepartmentAnalytics.tsx U
ApproveRequests.tsx U
ManageCredentials.tsx U
StudentDashboard.tsx X
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
617
618
619
619
620
621
622
623
623
624
625
625
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1
```

Coding:

Testing is a crucial phase in the development of the Smart Campus Sphere project to ensure that all system modules function correctly, efficiently, and securely. A series of testing methodologies were applied to validate both the frontend (React + Vite) and backend (PHP–MySQL REST API) components. The following are the major tests conducted:

1. Unit Testing

Purpose:

- To verify that individual functions, modules, and components operate correctly.

Applied On:

- React Components
Testing UI elements such as login forms, dashboards, validation functions, and API request handlers.
- PHP Backend Functions
Verifying login authentication, database queries, request handling, and API responses.
- Database Queries (MySQL)
Checking individual CRUD operations (Insert, Update, Select, Delete).
- Example Checks:
- Login form validation
- API JSON response format
- Session and token verification
- Dashboard rendering for each role

2. Integration Testing

- **Purpose:**
- To ensure smooth communication and data flow between combined modules.
- **Integration Points Tested:**
- React frontend communicating with PHP API
- PHP API interacting with MySQL
- Authentication flow (Login → Token/Session → Dashboard)
- Document upload flow (Frontend → Backend → Storage → Display)

Example Scenarios:

Student login → fetch dashboard → display academic records

Admin upload notice → students receive notification

Faculty update profile → backend saves → frontend reflects change

3. System Testing

Purpose:

- To evaluate the system as a whole and verify that all modules work together as per requirements.
- Areas Covered:
- Complete functionality of student, admin, and faculty dashboards
- Navigation and routing across pages
- Data consistency across modules
- Error handling and fallback pages

Example:

Checking whether a student can request documents and track status end-to-end

4. Acceptance Testing

Purpose:

- To validate that the final system meets user expectations and institutional requirements.
- Conducted With:
- Students
- Faculty
- Administrative staff
- Focus Areas:
- Ease of usage
- Clarity of UI
- Response time
- Accuracy of displayed academic/fee data
- This ensures the system is ready for real-world deployment.

5. Performance Testing

Purpose:

To assess system stability and responsiveness under load.

Key Aspects Tested:

- Page load time using Vite-optimized frontend
- Backend API response time under high traffic
- Database query speed for large records (users, notices, requests)
- Example:
- Stress testing login API for multiple parallel users

6. Security Testing

Purpose:

To ensure the platform is protected against common security threats.

Vulnerabilities Checked:

- SQL Injection attempts
- Session hijacking
- Unauthorized dashboard access
- File upload validation (document uploads)
- Measures Verified:
- Sanitized inputs before SQL queries
- Secure password hashing (PHP)
- Role-based access control enforcement

7. Usability Testing

Purpose:

To ensure the UI/UX is intuitive and user-friendly for all campus users.

Focus Areas:

- Mobile responsiveness
- Dashboard layout and clarity
- Accessibility for non-technical users
- Color contrast and readability
- Outcome:
- Improvements made to navigation bar, table readability, and notification visibility.

8. Regression Testing

Purpose:

To confirm that new updates do not break existing features.

Trigger Events:

- Adding new pages or components
- Updating API endpoints
- Modifying database schema
- Example:
- After implementing profile photo upload, testing login, dashboard, and request modules again.

User Manual

The following User Manual provides a clear guide for students, faculty, and administrators on how to use the OrbitCampus platform. It explains how to navigate the system, access features, and perform essential actions. This manual ensures that all users can operate the system efficiently and without confusion.

1. Introduction to OrbitCampus

OrbitCampus is a web-based campus management platform designed to streamline academic and administrative processes within an educational institution. The system provides separate dashboards for Students, Faculty, and Administrators, ensuring role-based access and security.

This user manual explains how to log in, navigate the system, perform tasks, and manage data effectively.

2. System Requirements:

Hardware Requirements

- Minimum **4 GB RAM** (8 GB recommended)
- Processor: **Intel i3/i5/i7 or AMD equivalent**
- Stable **internet connection**

Software Requirements

- **Operating System:** Windows / Linux / macOS

- **Browser:** Google Chrome, Microsoft Edge, or Firefox
- **XAMPP** installed (for local hosting)
- **Node.js + npm** (for running the React frontend)
- **Vite** development server (bundled with the frontend)

3. Getting Started

3.1 Accessing OrbitCampus

- Open your browser.
- Enter the official platform URL (provided by your institution).
- You will be directed to the Login Page.

4. Login & Authentication:

Steps to Log In

- Enter your **Email/Username**.
- Enter your **Password**.
- Click **Login**.

If credentials are correct, the user is taken to their role-based dashboard.

Forgot Password

If available:

- Click **Forgot Password**, enter your registered email, and follow instructions.

5. Role-Based Dashboards:

- After login, each user sees a dashboard tailored to their role.

6. Student Module – User Guide

6.1 Home Dashboard

Shows:

- Recent announcements
- Pending requests
- Profile summary

6.2 Profile Management:

Students can:

- Update basic profile details
- Upload/change profile photo

6.3 Academic Information:

- Includes:
- Academic records
- Course details
- Attendance (if enabled)
- Fee details

6.4 Document Requests:

Students can:

- Request bonafide certificates
- Request transcripts / leaves / academic documents
- Track request status
- *Pending*
- *In Process*
- *Approved*
- *Rejected*

6.5 Notifications:

- Displays all official notices posted by faculty/admin.

7. Faculty Module – User Guide

7.1 Dashboard:

Displays:

- Total students
- Pending document approvals
- Recent communications

7.2 Document Verification:

Faculty can:

- View student document requests
- Approve, Reject, or Mark as Processing
- Add comments or notes

7.3 Profile Section:

Faculty can manage:

- Personal details
- Contact information

7.4 Announcements:

Faculty can:

- Create announcements for students
 - Edit or remove old notices
-

8. Admin Module – User Guide

Admins have full control over the system.

8.1 User Management:

Admin can:

- Add new students/faculty users
- Edit user details
- Reset passwords
- Remove inactive users

8.2 Document Management:

Admins can:

- Access all document requests
- Provide final approval
- Upload official documents when needed

8.3 Academic & Fee Data:

Admins can manage:

- Fee details
- Academic records
- Semester updates
- Department data

8.4 Notification System:

Admins post campus-wide notices and updates.

8.5 System Settings:

Admins can configure:

- Backend settings
- Modules visibility
- Security settings

9. Navigating the Interface:

Sidebar Menu

Contains:

- Dashboard
- Profile
- Academics
- Documents
- Notifications
- Settings (Admin)

Top Navigation Bar

Shows:

- User name
- Logout button
- Quick notification icon

10. How to Upload Documents:

- Go to **Document Upload** section.

- Click **Select File**.
- Choose a PDF/JPG/PNG file.
- Click **Upload**.
- The file is stored in the backend (PHP–MySQL).

11. How to Request a Document:

- Navigate to **Request Document**.
- Select the document type from the dropdown.
- Enter purpose (optional).
- Click **Submit Request**.
- Track progress in the **Request Status** section.

12. Logging Out:

- To safely log out:
- Click the **Logout** button in the top right corner.
- You will be redirected to the login page.

13. Troubleshooting Guide:

Login Issues

- Check email/password.
- Make sure Caps Lock is off.
- Contact Admin for password reset.

File Upload Error

- Ensure the file is within allowed size limit.
- Supported formats: PDF, JPG, PNG.

Page Not Loading

- Refresh the page.
- Clear browser cache.
- Try a different browser.

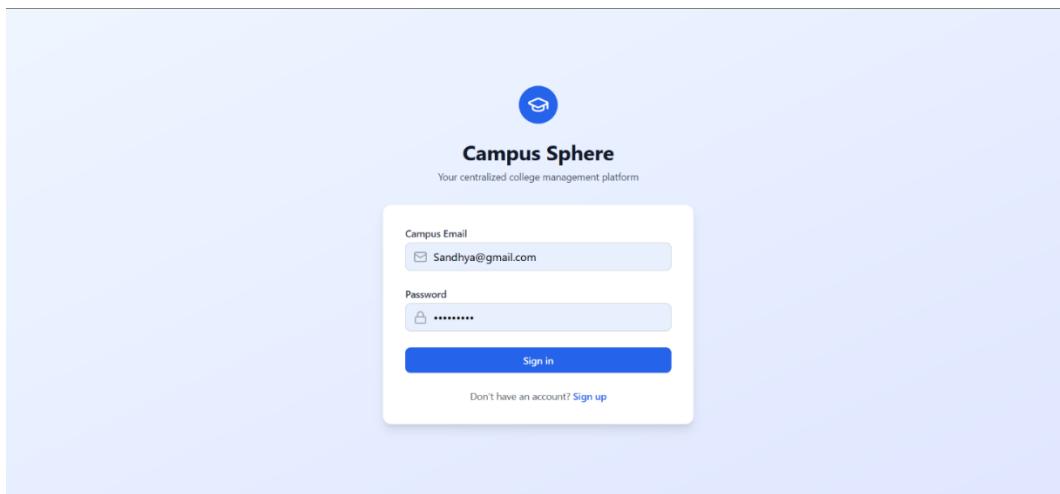
14. Safety & Security Tips:

- **Do not share your password with anyone.**

- Always logout after use.
- Keep personal details updated.
- Avoid accessing the system on public computers.

OUTPUT SCREEN'S

LOGIN PAGE



STUDENT SCREEN'S

Welcome back, Gara Mani Deepak!

Here's what's happening with your academic journey today.

Current CGPA	Pending Fees	Credits Earned	Attendance
9.2 +0.2	N/A Due: Jan 15	N/A 70% Complete	N/A Good Standing

Recent Activities

Upcoming Events

Quick Actions

- Pay Fees
- View Marks
- Download Books
- Request Document
- Apply Scholarship
- Academic Calendar

Campus Sphere

Guru Mani Deepak Student

Fee Management

Your financial overview for the academic year 2023-2024.

Total Annual Fees: ₹4,800

Amount Paid: ₹4,800

Amount Due: ₹0

Academic Year: 2023-2024
Due Date: 03/01/2024

All dues for this academic year have been cleared. Thank you!

Detailed Fee Breakdown

Campus Sphere

Guru Mani Deepak Student

Academic Records

Track your academic performance, view grades, and download official transcripts.

Current CGPA: 0.0 Out of 12.0

All Semesters

Subject-wise Performance

SUBJECT	SEMESTER	GRADE
No academic records found.		

Campus Sphere

Guru Mani Deepak Student

Study Materials

Access textbooks, PDFs, video lectures, lab manuals, and other academic resources.

Textbook 1 Lab Manual 1 Video Lecture 1

All types All Subjects

Search materials...

Textbook

Data Structures Textbook
Data Structures Semester 3 15.2 MB
Complete textbook for Data Structures course
Updated 1/1/2024

Lab Manual

Algorithm Analysis Lab Manual
Algorithms Semester 4 4.5 MB
Lab exercises and experiments
Updated 10/10/2024

Video Lecture

Database Design Video Lecture
Database Systems Semester 4 245 MB
Introduction to database design principles
Updated 1/1/2024

Download Download Download

Quick Access

Textbooks Core source textbooks
Video Lectures Recorded lectures
Lab Manuals Practical guides

Campus Sphere

Guru Mani Deepak Student

Document Requests

Request and download official documents.

New Request

DOCUMENT TYPE	REQUESTED AT	STATUS
Leave Certificate	11/01/2023	Pending
Leaving Certificate	11/02/2023	Ready for Pickup
Leaving Certificate	11/03/2023	Pending
No Due Certificate	11/10/2023	Ready for Pickup
Admin Notes: No		
Download Document		
Leaving Certificate	11/16/2023	Rejected

Campus Sphere

Guru Mani Deepak Student

Scholarships & Financial Aid

Explore opportunities to fund your education.

Search scholarships... All Levels

No Matching Scholarships Found
Try adjusting your search or filters.

Campus Sphere

Guru Mani Deepak Student

Class Time Table

View your class schedule, room assignments, and daily agenda.

Next Class: No more classes today

Today's Classes: Total Classes: 0 Lectures: 0 Labs: 0 Tutorials: 0

Weekly Overview: Total Classes/Week: 3 Active Days: 2 Subjects: 3 Teachers: 3

View Week Day Lecture Tutorial

Weekly Schedule

TIME	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY
08:00 - 10:30	Data Structures Room: 301 Dr. Seema Wilson					
10:00 - 12:30		Computer Networks Room: 202 Prof. Ashish Patel				
13:30 - 15:00						
15:30 - 17:00						

Gara Mani Deepak
Student

Dashboard
Fee Management
Academic Records
Study Materials
Document Requests
Scholarships
Time Table
Academic Support
Notifications
Campus Email
Profile

Academic Support Assistant

Get instant help with assignments, grades, schedules, and study guidance.

Quick Actions

- My Assignments
- Calculate CGPA
- Class Schedule
- Study Tips

Need Help?
Ask me anything about your academics, assignments, grades, or study tips!

Academic Assistant
Hello! I'm your Academic Support Assistant. I can help you with:

- Course information and schedules
- Assignment deadlines and requirements
- Study tips and resources
- Academic policies and procedures
- Fee and scholarship information

How can I assist you today?
08:13 PM

Show my assignments Calculate my CGPA Upcoming deadlines Study tips

Ask me anything about your academics...

Gara Mani Deepak
Student

Dashboard
Fee Management
Academic Records
Study Materials
Document Requests
Scholarships
Time Table
Academic Support
Notifications
Campus Email
Profile

Notifications

All

No Notifications
You're all caught up!

Gara Mani Deepak
Student

Dashboard
Fee Management
Academic Records
Study Materials
Document Requests
Scholarships
Time Table
Academic Support
Notifications
Campus Email
Campus Email
Profile

Campus Email

Secure communication within the campus community.

+ Compose

Inbox 2

Sent

Drafts

Trash

1 unread message

Inbox

Search emails...

From: sarah.wilson@campus.edu • 1/14/2024
Assignment Submission Reminder
Dear John, This is a reminder that your Data Structures assignment is due tomorrow. Please submit it...

From: admin@campus.edu 1/13/2024
Fee Payment Reminder
Your semester fee payment is due on January 15th. Please make the payment to avoid late fees...

Gara Mani Deepak
Student

Profile

Full Name: Gara Mani Deepak
Email: deepak@gmail.com

Phone: _____
Department: _____

Address: _____

Bio: Tell us about yourself...

Save Profile

Sign Out

Teacher screen's

Campus Sphere

Bindhu Teacher

Good morning, Bindhu!
Ready to inspire minds today? Here's your teaching overview.

Total Students: **156** (3 subjects)

Pending Evaluations: **23** Due this week

Class Average: **7.8** +0.5 from last sem

Office Hours: **12** This week

My Classes

- Data Structures** CS201
 - Time: 9:00 AM - 10:30 AM
 - Location: Room 301
 - Attendance: 89%
 - [View Details](#)
- Algorithms** CS301
 - Time: 11:00 AM - 12:30 PM
 - Location: Room 205
 - Attendance: 92%
 - [View Details](#)
- Database Systems** CS401
 - Time: 2:00 PM - 3:30 PM
 - Location: Room 401
 - Attendance: 85%
 - [View Details](#)

Today's Schedule

- 9:00 AM **Data Structures** Room 301 +48 students
- 11:00 AM **Algorithms** Room 205 +38 students
- 2:00 PM **Database Systems** Room 401 +73 students

Recent Activities

- Assignment Graded: Data Structures - Assignment 3 2 hours ago
- Attendance Marked: Algorithms - Morning session 4 hours ago
- Email Sent: Class announcement to CS201 students 1 day ago
- Marks Updated: Database Systems - Mid-term results 2 days ago
- Office Hours: 3 students attended consultation 3 days ago

Quick Actions

- [Mark Attendance](#)
- [Upload Marks](#)
- [Send Email](#)
- [Class Schedule](#)
- [Student Reports](#)
- [Course Materials](#)

Campus Sphere

Bindhu Teacher

My Assigned Students

Search by name or email...
Gara Mani Deepak

STUDENT NAME **EMAIL**

Marks

Campus Sphere

Bindhu Teacher

Enter Marks
Select a course and enter the marks for each student.

Course Selection

STUDENT	UNIT 1	UNIT 2	END TERM	FINAL EXAM
Deepak Kurur (ID: 1)	85	90	88	N/A
Nandu Kohare (ID: 2)	92	88	90	N/A
Jaswanth (ID: 3)	78	82	80	N/A
Bala Manohar (ID: 4)	88	91	89	N/A
Siddu (ID: 5)	95	94	95	N/A

Save All Marks

Submit Grades

No Students Assigned
You do not have any students assigned to you yet.

Take Attendance

Select a class and mark the attendance for each student.

Class Selection	Math
Deepak Kumar Student ID: 1	Select
Nandha Kishore Student ID: 2	Select
Jaswarth Student ID: 3	Select
Bala Manohar Student ID: 4	Select
Siddu Student ID: 5	Select

Save All Attendance

Course Materials

Select a Course: Introduction to Computer Science

Upload New Material

Choose File: No file chosen. Upload File

- Uploaded Materials for CS101
 - Lecture 1 - Intro to CS.pdf (Uploaded on: 2024-01-15)
 - Syllabus.pdf (Uploaded on: 2024-01-12)
 - Assignment 1.pdf (Uploaded on: 2024-01-10)

My Class Schedule

November 28, 2025 Friday

CS101	Project Presentations	10:00 - 12:30
Office Hour	Student Queries	15:00 - 16:00

My Office

My Payroll

Total Earnings: ₹65,000.00 Total Deductions: ₹8,000.00 Net Salary: ₹57,000.00

Salary Slip for June 2024

Earnings	Deductions
Basic Salary: ₹50,000.00	Provident Fund (PF): ₹4,000.00
Housing Rent Allowance: ₹10,000.00	Professional Tax: ₹1,500.00
Dearness Allowance: ₹5,000.00	Income Tax (ITD): ₹2,500.00
Net Salary Payable: ₹57,000.00	

Download Slip

My Documents

Upload New Document

Choose File: No file chosen.

Search documents...

Name	Category	Date	Size	Actions
Math_Material_Question_Paper.pdf	Question Papers	2024-05-10	1.2 MB	
Science_Project_Guidelines.docx	Project Guidelines	2024-04-22	0.8 MB	
Class_10_Syllabus.pdf	Syllabus	2024-03-15	2.5 MB	
Faculty_Meeting_Minutes_April.pdf	Meeting Minutes	2024-04-05	0.5 MB	
Professional_Development_Certificate.jpg	Certificates	2023-11-05	3.1 MB	

Parent screen's

Welcome, Mrs. Sharma
Here is a summary for your child, Ananya Sharma.

Recent Grade	Attendance	Pending Fees	Unread Messages
A-	95%	2	3

Recent Messages

- B Bindhu** Mathematics
No, a standard scientific calculator will be sufficient.
11 ago
- P Padmaja** Science
Anaya did a great job on her science project!
Yesterday
- S Sridevi** History
The field trip form is due tomorrow.
2 days ago

Child's Performance Overview

Academic Grades

SUBJECT	MARKS	GRADE
Mathematics	88	A
Science	92	A+
History	75	B+
English	81	A
Physical Education	95	A+

Attendance Records

Subject	Attended / Total
Mathematics	80 / 10
Science	90 / 12
History	80 / 10
English	80 / 10

Teacher Communication

Teachers

- B Bindhu** Mathematics
- P Padmaja** Science
- S Sridevi** History
- N PVL Narayana** English

Bindhu Mathematics

Hi, just a reminder about the math test on Friday
3:30 PM

Thanks for the reminder! Will Anaya need a special calculator?
5:00 PM

No, a standard scientific calculator will be sufficient.
5:05 PM

Fee Management

Total Amount Due ₹ 83,000
Across all pending invoices.

Pending Invoices

DESCRIPTION	DUE DATE	AMOUNT	ACTION
Term 1 Tuition Fee	2024-07-31	₹75,000	Pay Now
Bus Transportation Fee	2024-07-31	₹8,000	Pay Now

Payment History

INVOICE	DATE PAID	AMOUNT	RECEIPT
Annual Development Fee	2024-06-20	₹12,000	Download
Term 4 Tuition Fee (2023)	2024-03-15	₹72,000	Download

Progress Reports

Available Reports

REPORT TERM	DATE ISSUED	DOWNLOAD
Mid-Term, Fall 2023	2023-10-15	Download
Final, Fall 2023	2023-12-20	Download
Mid-Term, Spring 2024	2024-01-10	Download

Campus Sphere

Sridevi Live S

5 Sridevi Parent

- Dashboard
- Child Performance
- Fee Management
- Teacher Communication
- Progress Reports
- PTM Schedule**
- Notifications
- Campus Email
- Profile

Parent-Teacher Meetings

My Booked Slots

Mathematics with Mr. Davis
2024-05-21 11:00 - 11:15 Online

Book a New Slot

Mathematics with Mr. Davis
2024-05-20 10:00 - 10:15

History with Ms. Garcia
2024-05-20 10:30 - 10:45

HOD Screen's

Campus Sphere

Lakshmi Narayana Live L

L Lakshmi Narayana Hod

- Dashboard
- Manage Teachers
- Student Overview
- Department Analytics
- Approve Requests
- Assign Class Teacher
- Manage Credentials
- Assign Students
- Notifications
- Campus Email
- Profile

Welcome back, Head of Department!

Here's a summary of your department's performance and metrics.

Total Students **350**

Courses Offered **25**

Pass Percentage **92%**

Faculty Count **15**

Teacher Workload

Teacher	Subject	Workload
Dr. Evelyn Reed	Quantum Physics	8 hrs/week
Mr. Samuel Grant	Organic Chemistry	12 hrs/week
Ms. Olivia Chen	Linear Algebra	10 hrs/week

Manage Teachers →

Recent Activity

- New course 'Advanced Thermodynamics' approved. 2h ago
- Reminder: Departmental meeting at 3 PM today. 4h ago
- Dr. Reed requested a new lab equipment budget. Yesterday

Quick Actions

- Approve Leave/Budget Requests
- View Department Analytics
- Assign Class Teachers

Campus Sphere

Lakshmi Narayana Live L

L Lakshmi Narayana Hod

- Dashboard
- Manage Teachers**
- Student Overview
- Department Analytics
- Approve Requests
- Assign Class Teacher
- Manage Credentials
- Assign Students
- Notifications
- Campus Email
- Profile

Manage Teachers

ID	Name	Email	ACTIONS
5	Bindu	bindu@gmail.com	Manage
7	Rajendra Janchi	RajendraJanchi@gmail.com	Manage
11	Gopi	Gopi@gmail.com	Manage

Campus Sphere

Lakshmi Narayana Live L

L Lakshmi Narayana Hod

- Dashboard
- Manage Teachers
- Student Overview**
- Department Analytics
- Approve Requests
- Assign Class Teacher
- Manage Credentials
- Assign Students
- Notifications
- Campus Email
- Profile

Student Overview

Monitor all students within your department.

STUDENT NAME	COURSE	YEAR	ATTENDANCE	OVERALL GRADE
Rahul Marhar	Computer Science	2	85%	82%
Deepak	Computer Science	2	95%	91%
Jaswanth	Quantum Physics	3	99%	94%
Karunika	Linear Algebra	1	98%	88%
Kumar	Organic Chemistry	3	88%	85%
Nandha Kalan	Linear Algebra	1	92%	88%
Sidhu	Atmospheric Physics	2	98%	90%

Approve Requests

Review and manage requests from your department.

Search by applicant or details... All Statuses

Requester	Date	Description	Action Buttons
Dr. Evelyn Reed Leave Request	2023-10-26	Requesting 3 days of professional leave for a conference.	<button>✓ Approve</button> <button>✗ Reject</button>
Mr. Samuel Grant Budget Request	2023-10-25	Requesting \$500 for new lab equipment.	<button>✓ Approve</button> <button>✗ Reject</button>
Ms. Olivia Chen Leave Request	2023-10-24	Requesting 1 day of sick leave.	<button>✓ Approve</button> <button>✗ Reject</button>
Dr. Benjamin Carter Budget Request	2023-10-22	Requesting \$1,200 for research materials.	<button>✓ Approve</button> <button>✗ Reject</button>
Dr. Evelyn Reed Leave Request	2023-10-23	Requesting 2 days of personal leave.	<button>✓ Approve</button> <button>✗ Reject</button>

Manage Credentials

Oversee and secure faculty access and credentials.

Search for a teacher...

TEACHER	ROLE	LAST LOGIN	TWO-FACTOR AUTH	ACTIONS
Dr. Evelyn Reed T001	Teacher	2023-10-26 09:15 AM	Enabled	<input type="checkbox"/> Reset Password <input type="checkbox"/> Toggle 2FA
Mr. Samuel Grant T002	Teacher	2023-10-25 08:45 PM	Disabled	<input type="checkbox"/> Reset Password <input type="checkbox"/> Toggle 2FA
Ms. Olivia Chen T003	Teacher	2023-10-26 11:00 AM	Enabled	<input type="checkbox"/> Reset Password <input type="checkbox"/> Toggle 2FA
Dr. Benjamin Carter T004	Senior Teacher	2023-10-24 08:00 AM	Enabled	<input type="checkbox"/> Reset Password <input type="checkbox"/> Toggle 2FA

Assign Class Teacher

Semester: Semester 1

Currently Assigned: Sudhu

Teacher: Select a Teacher

Assign Students to Semester

STUDENT NAME	EMAIL	ASSIGNED SEMESTER
Gaurav Deepak	deepak@gmail.com	Semester 1
Kumar	kumar@gmail.com	Semester 4
Nandu	nandu@gmail.com	Semester 3

Department Analytics

In-depth metrics and performance of your department.

Total Students **350**

Courses Offered **25**

Average Pass Rate **92%**

Faculty Count **15**

Annual Pass Percentage

Year	Pass Rate (%)
2020	85
2021	80
2022	85
2023	85

Enrollment Trends

Month	New Students
Jan	30
Mar	45
Jun	60
Sep	50
Nov	70

Faculty Workload Distribution

Workload Segment	Percentage
1	40%
2	30%
3	20%
4	10%

NON Teaching Staff Screen's

Campus Sphere

Welcome back, Chiru!

Here's your work calendar and task assignments for today:

Tasks Assigned: 12 (3 pending)

Attendance Rate: 96% This month

Leave Balance: 18 Days remaining

Work Hours: 8.5 Today

Current Tasks:

- Library Maintenance @ Central Library - 2nd floor Due: Today, 3:00 PM
- AC Repair - Room 501 @ Computer Science Block Due: Tomorrow, 10:00 AM
- Garden Maintenance @ Main Campus Grounds Due: Jan 20, 2024

Today's Schedule:

- 9:00 AM Daily Safety Inspection Engineering Block High priority
- 11:00 AM Equipment Maintenance Admin Building Medium priority
- 2:00 PM Cleaning Supervision Administration Building Low priority
- 4:00 PM Security Round Campus perimeter High priority

Recent Activities:

- Task Completed: Electrical repair in Lab 200 (2 hours ago)
- Attendance Marked: Check-in at 8:00 AM (4 hours ago)
- Leave Request: Applied for medical leave (2 days ago)

Quick Actions:

- Mark Attendance
- Report Issue
- View Tasks
- Request Leave
- Download Paylip
- Contact Supervisor

Campus Sphere

My Tasks

Add New Task

Process pending admission applications (High Priority) Dept: Admissions Due: 2024-08-10

Compile monthly financial summary for July (High Priority) Dept: Finance Due: 2024-08-05

Update IT inventory list (Medium Priority) Dept: IT Due: 2024-08-15

Organize files for the upcoming audit (Low Priority) Dept: Administration Due: 2024-09-01

Respond to support tickets in the helpdesk queue (Medium Priority) Dept: IT Due: 2024-08-01

Campus Sphere

My Attendance

November 2025

DATE	STATUS	CHECK IN	CHECK OUT	LOCATIONS
No attendance records for this month.				

Campus Sphere

Leave Management

Annual Leave: 12 / 20 days | Sick Leave: 7 / 10 days | Casual Leave: 5 / 10 days

+ Apply for New Leave

dd-mm-yyyy dd-mm-yyyy Annual Leave Submit

My Leave Requests

TYPE	DATES	REASON	STATUS
Annual Leave	2024-08-10 to 2024-08-25	Family vacation	Approved
Sick Leave	2024-07-10 to 2024-07-16	Flu	Approved
Casual Leave	2024-08-01 to 2024-08-01	Personal appointment	Pending
Unpaid Leave	2024-06-10 to 2024-06-12	Extended travel	Rejected

Campus Sphere

My Payroll

Total Earnings: ₹43,000.00 | Total Deductions: ₹5,500.00 | Net Salary: ₹37,500.00

Salary Slip for July 2024

Earnings	Deductions
Basic Salary	₹30,000.00 Provident Fund (PF)
Travel Allowance	₹5,000.00 Employee State Insurance (ESI)
Food Allowance	₹3,000.00 Professional Tax
Net Salary Payable:	₹37,500.00

July 2024 Download Slip

Campus Sphere

Chiru Staff

Work Locations & Schedules

Search by building or department...

Main Administration Building
2nd Floor, Admissions Office
Mon: 09:00 - 17:00 Tue: 09:00 - 17:00 Wed: 09:00 - 17:00 Thu: 09:00 - 17:00 Fri: 09:00 - 13:00 Sat: Closed Sun: Closed

Library Complex
Ground Floor, Library Services
Mon: 08:00 - 18:00 Tue: 08:00 - 18:00 Wed: 08:00 - 18:00 Thu: 08:00 - 18:00 Fri: 08:00 - 18:00 Sat: 10:00 - 16:00 Sun: Closed

Engineering Block B
3rd Floor, IT Support Helpdesk
Mon: 08:30 - 17:30 Tue: 08:30 - 17:30 Wed: 08:30 - 17:30 Thu: 08:30 - 17:30 Fri: 08:30 - 17:30 Sat: Closed Sun: Closed

Campus Sphere

Chiru Staff

Staff Documents

Search documents...

NAME	CATEGORY	DATE	SIZE	ACTIONS
Finance_Report_July.xlsx	Financial Reports	2024-07-01	0.5 MB	
Admission_Policy_2024.pdf	Policy Documents	2024-07-10	1.8 MB	
IT_Maintenance_Schedule.pdf	Schedules	2024-07-25	0.3 MB	
Staff_Handbook_Reviewed.docx	Handbooks	2024-07-15	2.2 MB	
Event_Proposal_Annual_Day.pdf	Proposals	2024-07-05	1.1 MB	

ADMIN Screen's

Campus Sphere

Chiru Live Sandya Rani 5

Sandy Rani Admin

Dashboard

Total Users **7** Students **3** Teachers **1** Staff **1**

Welcome, Sandy Rani! Comprehensive overview of campus operations and management.

Administrative Tools

Assign Students to Semester
Manage student semester assignments.

Campus Sphere

Chiru Live Sandya Rani 5

Sandy Rani Admin

Dashboard

User Management

Name	Email	Role	Actions	
Bindhu	bindhu@gmail.com	teacher		
Chiru	Chiru@gmail.com	staff		
Gara Mani Deepak	deepak@gmail.com	student		
Gopi	Gopi@gmail.com	teacher		
Kumar	kumar@gmail.com	student		
Lakshmi Narayana	Naryana@gmail.com	hod		
Laxmi	Laxmi@gmail.com	parent		
Nandhu	nandhu@gmail.com	student		
Padmja Kanchi	Padmja@gmail.com	teacher		
Sandy Rani	Sandhya@gmail.com	admin		
Sridevi	Sridevi@gmail.com	parent		

Campus Sphere

Sandy Rani Admin · 5

Academic Calendar

Event Title: dd-mm-yyyy Event: dd-mm-yyyy

Event Description:

Add Event

Today Back Next

Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	31	01
Sports						
02	03	04	05	06	07	08
09	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	01	02	03	04	05	06

November 2025

Month Week Day Agenda

Campus Sphere

Sandy Rani Admin · 5

Document Workflow

REQUEST ID	STUDENT	DOCUMENT TYPE	STATUS	REQUESTED AT	ACTIONS
#6	Nandhu nandhu@gmail.com	Leaving Certificate	Processing	11/23/2025	 
#5	Gara Mani Deepak deepak@gmail.com	Bonafide Certificate	Pending	11/21/2025	
#4	Gara Mani Deepak deepak@gmail.com	Leaving Certificate	Ready for Pickup	11/20/2025	  
#3	Gara Mani Deepak deepak@gmail.com	Leaving Certificate	Pending	11/20/2025	
#2	Gara Mani Deepak deepak@gmail.com	No Dues Certificate	Ready for Pickup	11/19/2025	  
#1	Gara Mani Deepak deepak@gmail.com	Leaving Certificate	Rejected	11/19/2025	

Campus Sphere

Sandy Rani Admin · 5

Admin Fee Management

Generate Missing Records

Search by student name or ID... All Statuses

STUDENT	YEAR	TOTAL	PAID	DUUE	STATUS	ACTION
Gara Mani Deepak ID: 1	2023-2024	₹4,800	₹4,800	₹0	Paid	

Campus Sphere

Sandy Rani Admin · 5

Manage Parent-Student Links

Create New Link

Select Parent: Laxmi (Laxmi@gmail.com) Select Student: Nandhu (nandhu@gmail.com)

Link

Existing Links

Parent	Student	Actions
Laxmi (Laxmi@gmail.com)	Nandhu (nandhu@gmail.com)	
Sridevi (Sridevi@gmail.com)	Gara Mani Deepak (deepak@gmail.com)	

S Sandya Rani Admin +

- Dashboard
- User Management
- Analytics
- Academic Calendar
- Document Workflow
- Fee Management
- Parent-Student Links
- Communications
- Security & Access**
- Notifications
- Campus Email

Security & Access Control

Manage roles and permissions across the platform.

Roles

- Admin
- HOD**
- Teacher
- Student
- Parent

HOD Permissions

- Manage Department Teachers
- View Department Analytics
- Approve Department Requests
- Assign Courses to Teachers

S Sandya Rani Admin +

- Dashboard
- User Management
- Analytics
- Academic Calendar
- Document Workflow
- Fee Management
- Parent-Student Links
- Communications**
- Security & Access
- Notifications
- Campus Email
- Profile

Communications Hub

Send and manage announcements for all campus members.

Create Announcement

What's on your mind? Broadcast a message to the campus...

To: All To: Students To: Teachers To: Parents

Send Announcement

Sent Announcements

All Students Teachers Parents

Upcoming Holiday Schedule
To: All • 2024-07-28
Please be advised that the campus will be closed for the upcoming public holiday on Friday. All classes and administrative activities will be suspended.

Final Exam Timetable Released
To: Students • 2024-07-27
The final examination timetable for the current semester is now available on the student portal. Please review your schedules and report any clashes immediately.

Faculty Development Workshop
To: Teachers • 2024-07-26
A mandatory faculty development workshop on modern teaching methodologies will be held next Monday. All teaching staff are required to attend.

Parent-Teacher Meeting Invitation
To: Parents • 2024-07-25
We invite all parents to the scheduled Parent-Teacher Meeting this Saturday to discuss your child's academic progress and overall development.

S Sandya Rani Admin +

- Dashboard
- User Management
- Analytics**
- Academic Calendar
- Document Workflow
- Fee Management
- Parent-Student Links
- Communications
- Security & Access
- Notifications
- Campus Email
- Profile

Analytics Dashboard

Comprehensive insights into campus operations and performance metrics.

Total Users **3,101** +12.3% from last month

Revenue **₹12.4M** +8.7% from last month

Average CGPA **7.8** +0.2 from last semester

System Uptime **99.8%** Excellent performance

User Growth Trend

Month	Users
Aug	2,650
Sep	2,720
Oct	2,780
Nov	2,820
Dec	2,847

Department Performance

Department	CGPA	Attendance	Satisfaction	Students
Computer Science	8.2	89.5%	4.6/5	856 students
Electronics	7.9	87.2%	4.4/5	632 students
Mechanical	7.6	85.8%	4.2/5	734 students
Civil	7.8	88.1%	4.3/5	445 students
Chemical	8	90.2%	4.5/5	139 students

User Breakdown

User Type	Count	Change
Students	2,847	+12.3%
Teachers	156	+8.7%
Staff	98	+0.2%
Parents	1247	+11.6%
Admins	8	0%

Academic Performance

Metric	Value
Pass Rate	92.4%
Attendance Rate	87.6%
Completion Rate	89.3%

System Health

Metric	Value
Uptime	99.8%
Response Time	245ms
Error Rate	0.02%
Storage Usage	78.4%

BIBLIOGRAPHY

AND

CONCLUSION

Bibliography:

React Documentation – Official documentation for building interactive UIs.

<https://react.dev/>

Vite Official Guide – Documentation for fast frontend tooling and development server.

<https://vitejs.dev/>

PHP Manual – Reference for server-side scripting and backend logic.

<https://www.php.net/manual/>

MySQL Reference Manual – Official guide for relational database management and SQL querying.

<https://dev.mysql.com/doc/>

W3Schools Web Technologies Tutorials – General guidance for HTML, CSS, JavaScript.

<https://www.w3schools.com/>

MDN Web Docs – Standard reference for web technologies and frontend programming.

<https://developer.mozilla.org/>

Axios Documentation – For handling frontend-to-backend API communication.

<https://axios-http.com/>

Software Engineering Textbooks – For system design, SRS documentation, and testing methodologies:

Pressman, Roger. "Software Engineering: A Practitioner's Approach." McGraw-Hill.

Bootstrap & UI/UX Guidelines – For styling concepts referenced in frontend development.

<https://getbootstrap.com/>

General REST API Design References – Best practices for backend service structure.
<https://restfulapi.net/>

Conclusion

The *Orbit Campus* platform successfully demonstrates how modern web technologies can digitalize and streamline core academic and administrative operations within an educational institution. By integrating a **React + Vite frontend** with a **PHP–MySQL backend**, the system delivers a highly responsive, secure, and scalable environment for students, faculty, and administrators.

The platform not only centralizes essential campus services—such as profile management, document handling, fee information, academic data, and notifications—but also significantly reduces manual workload, paperwork, and dependency on traditional processes. The role-based access model ensures that every user interacts only with the information relevant to their responsibilities, improving usability and security.

Through structured APIs, optimized database storage, and a user-friendly interface, *Orbit Campus* enhances transparency, minimizes errors, and accelerates campus workflows. It succeeds in creating a unified digital ecosystem that improves communication, accessibility, and operational efficiency for all stakeholders.

Overall, *Orbit Campus* stands as a scalable and future-ready solution capable of supporting the evolving digital needs of modern educational institutions.

THANK YOU