BANNARI AMMAN<br>INSTITUTE OF TECHNOLOGY
Stay Ahead

# BANNARI AMMAN INSTITUTE OF TECHNOLOGY

**An Autonomous Institution Affiliated to Anna University Chennai, Accredited by NAAC with A+ Grade**

**Sathyamangalam 638401 Erode District, Tamil Nadu,India**

| | |
|---|---|
| **Name** | Shrigayathri S |
| **Roll no** | 7376221CS309 |
| **Seat no** | 367 |
| **Project ID** | 4 |
| **Problem Statement** | Student Ranking Dashboard (PS) |

**Technical Components→**

| Component | Tech Stack |
|---|---|
| Frontend | React.js,React Router,Redux |
| Backend | Node.js, Express.js |
| Database | MongoDB |

# Project Flow

## Problem Statement

⇨ In educational institutions, managing student information, conducting assessments, and facilitating placement processes can be challenging due to the manual nature of many tasks. There is a need for a streamlined system that automates user management, performance tracking, and placement processes while providing real  time updates and insights.

## Solution

⇨The PS Portal is designed to address these challenges by providing a centralized platform with distinct functionalities for admins, faculty, and students. The portal will offer user management, performance tracking, course registration, Student Ranking Dashboard, and placement management in a seamless and efficient manner.

**Scope**

◦ Creating a web application using the MERN stack (MongoDB, Express.js, React.js, Node.js).

◦ Implementing user authentication and authorization.

◦ Developing a personalized dashboard for students, faculty, and administrators.

◦ Enabling course management, test bookings, and performance tracking.

◦ Facilitating placement processes and maintaining student placement details.

◦ Providing bulk upload and export functionalities for student data.

**Dependencies**

➢ Node.js and npm for backend development.

➢ React.js for frontend development.

➢ MongoDB for database management.

➢ Express.js for handling server side logic.

➢ JSON Web Tokens (JWT) for authentication.

➢ Multer and XLSX for file handling and bulk data operations.

**Functional Requirements**

**I. Admin Features**

**User Management**:

o Create, update, and delete user accounts for Admin , Faculty and Students.

**Bulk Upload/Export:**

o Provide bulk upload and export options for student data.

**Schedule Interviews:**

o Schedule interviews for students and manage placement processes.

**Filter and Sort:**

o Filter and sort student data based on various criteria like overall top 100 based ps level, ds level, Fa percentage, cgpa, Full stack level.

**Access Control:**

o Assign students to specific faculty members for progress tracking and review.

**Course Management:**

o Add / Remove / update new course category, course content , Schedule Test, Review Test

## II. Faculty Features

**Review Scheduling:**

o Schedule reviews for students and update their marks.

**Student Management:**

**o** View and manage the list of students allotted by the admin.

**Performance Tracking:**

o Track student performance in courses, daily challenges, and activities.

**III. Student Features:**

**Dashboard:**

o Display personalized dashboard with.

o Daily challenges and task progress.

o Course completion levels (programming, data structures, full stack project, Git Points).

o Formative assessment marks.

o Training Attendance

o Rank display and top 100 students' ranks.

**Course Registration:**

o Available Courses ▻ View and register for available courses.

o My Courses ▻ View registered courses and book test slots.

o Eligible Courses ▻ View and register for courses based on eligibility.

**Placement Details:**

o View placement details, including the number of companies attended and interview levels completed, Eligible company details .

**Resume Upload:**

o Upload and manage resumes.

**Rank Display:**

o Display ranks based on course completion and performance.

**Technology Used**

**Frontend** ➤ React.js, Redux for state management, React Router for seamless navigation.

**Backend** ➤ Node.js, Express.js for server side logic.

**Database** ➤ MongoDB for data storage and management.

**Authentication** ➤ JSON Web Tokens (JWT) for secure authentication.

**File Handling** ➤ Multer for file uploads, XLSX for Excel file handling.

**Deployment** ➤ Heroku or AWS for deploying the application.

## Admin Features

### 1. User Management

**Database**

◦ Use MongoDB to store user data (students, faculty, and admin).
Students-Mail id, Password, Name, Roll NO, Department, Year
Faculty-Mail id, Password, Name,Faculty Id, Department, Domain
Admin-Mail id , Password

**Backend**

◦ Use Express.js to handle CRUD operations for user management.
To update User Details like Delete, Add, Edit

**Authentication**

◦ Implement JWT (JSON Web Tokens) for secure authentication and role based access control, based on College mail Id.

## 2. Bulk Upload/Export

**File Handling**

- Use libraries like multer for handling file uploads in Express.js.

Excel sheets: Formative assessment mark sheet, ps level 1 to 7 mark sheet, ds level 1 to 3 mark sheet, cgpa sheet

**Data Parsing**

- Use xlsx or csv parse to read and parse Excel or CSV files.

**Database Operations**

- Insert multiple records into MongoDB in a single operation for bulk upload.

## 3. Schedule Interviews

**Database**

- Store interview schedules and related data in MongoDB.

Entity : Roll no, Name, Vertical, Project title, Timing, Schedule or Reschedule

**Backend**

- Create APIs to manage interview schedules.

## 4. Access Control

**Database**

- Maintain relationships between students and faculty members.

**Backend**

- Create APIs to assign students to faculty and manage access control.

## 5. Filter and Sort

**Database Queries**

- Use MongoDB's query capabilities to filter and sort data.

**Frontend**

- Implement sorting and filtering options in the admin interface.

**Faculty Features**

**1. Review Scheduling**

**Database**

- Store review schedules and marks.

**Backend :**

- Create APIs to manage review scheduling and marks updates.

**2. Student Management**

**Database**

- Store and retrieve student information assigned to faculty.

**Backend**

- Create APIs for faculty to manage assigned students.

**3. Performance Tracking**

**Database**

- Store performance data for various courses and activities.

**Backend**

- Create APIs to update and retrieve performance data.

**Student Features**

**1. Dashboard**

**Frontend**

- Use React.js to create a dynamic and interactive dashboard.

**Backend**

- Create APIs to fetch student  specific data for the dashboard.

**2. Course Registration**

**Database**

- Store course registration and test slot booking data.

**Backend**

- Create APIs to handle course registration and test booking.

**3. Placement Details**

**Database**

- Store placement details and interview records.

**Backend**

- Create APIs to manage and retrieve placement data.

**4. Resume Upload**

**File Handling**

- Use multer for handling resume file uploads.

**Storage**

- Store resume files in a directory or a cloud storage service.

**Backend**

- Create APIs to manage resume uploads.

**5. Rank and Performance**

**Database**

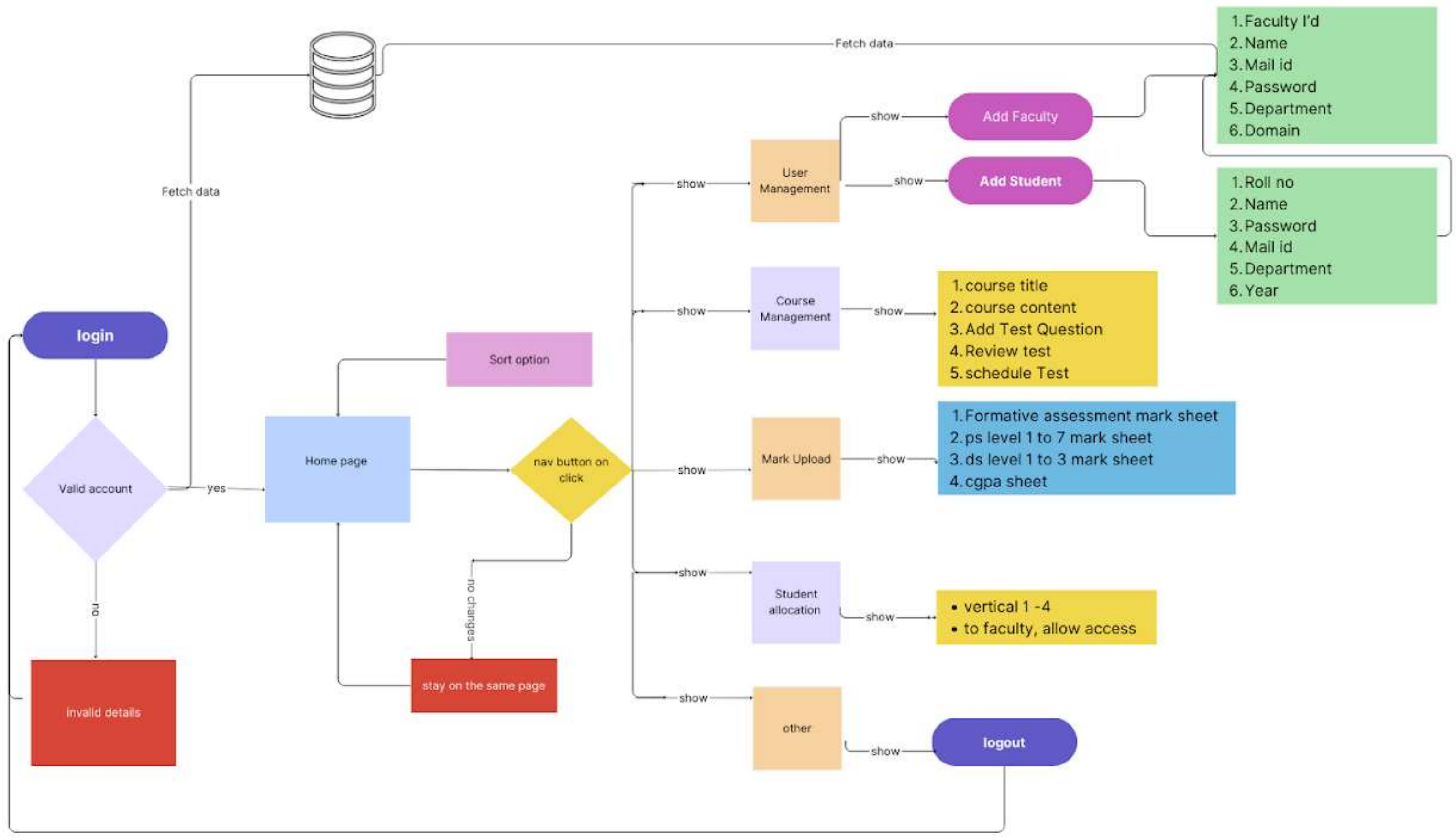
- Store ranking and performance metrics.

**Backend**

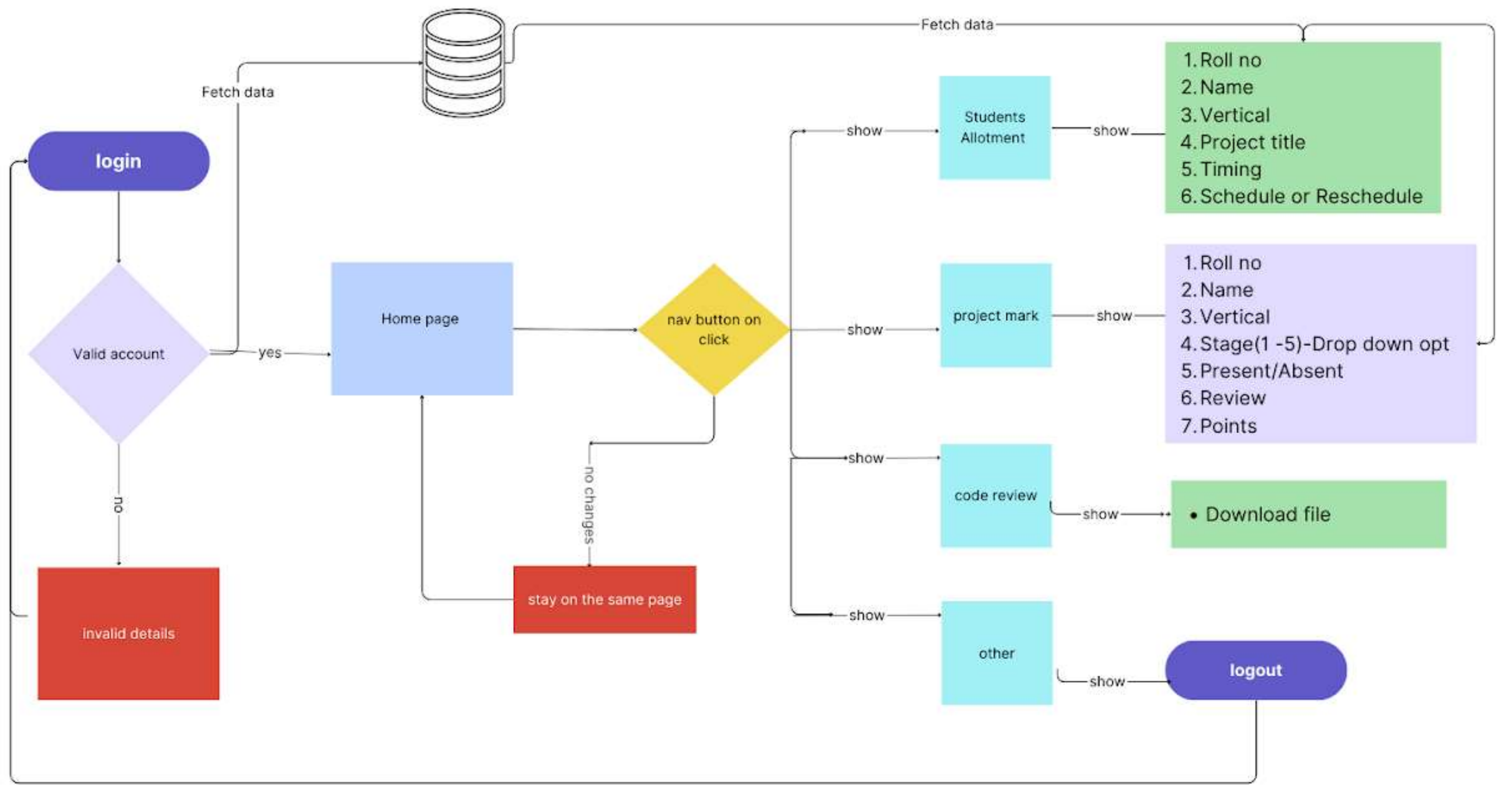
- Create APIs to calculate and retrieve ranks.

# Flowchart

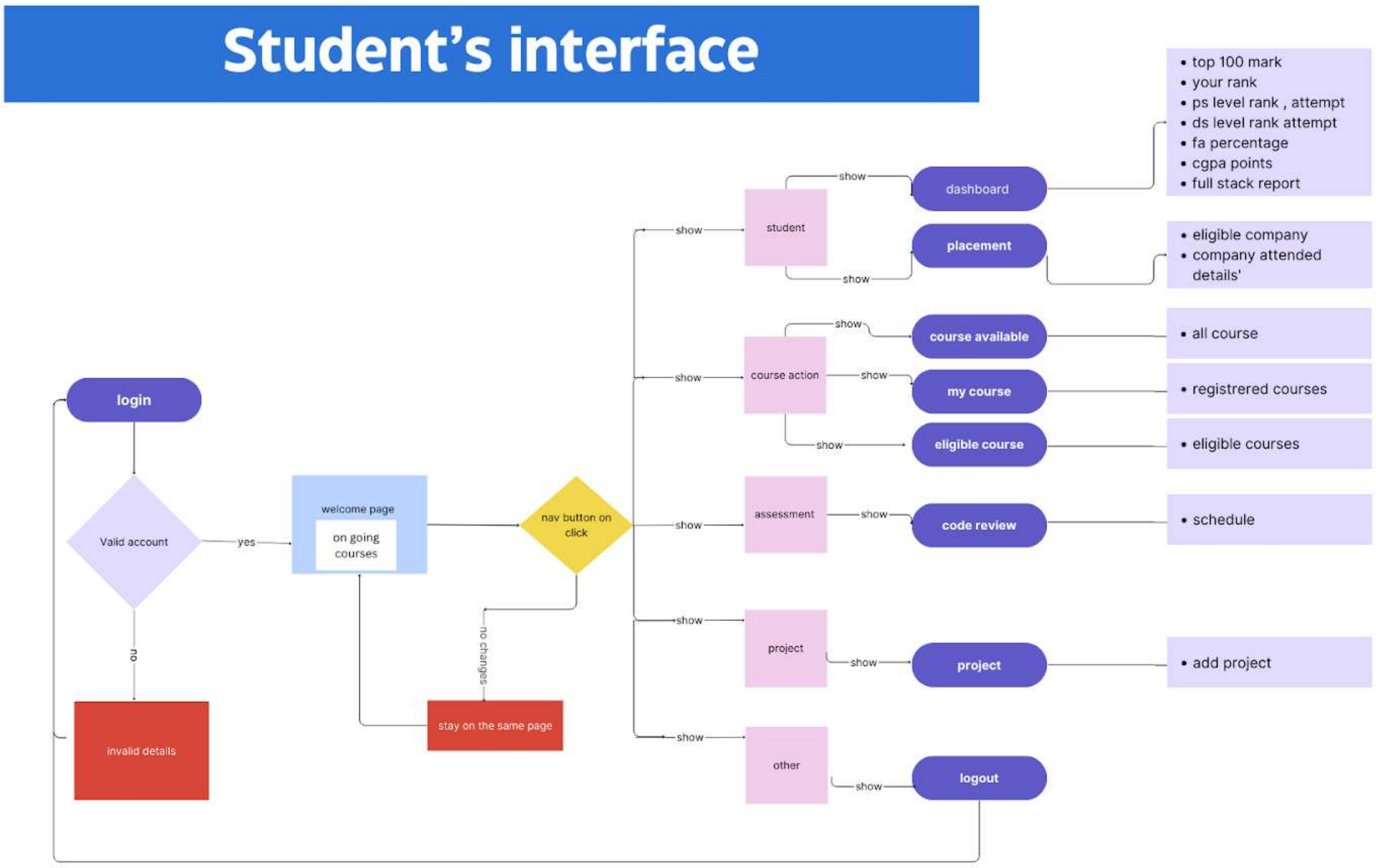
## Admin interface



## Faculty interface



# Student's interface


login
Valid account
yes
no
invalid details
welcome page
on going courses
nav button on click
no changes
stay on the same page
show
student
show
dashboard
show
placement
course action
show
course available
show
my course
show
eligible course
assessment
show
code review
project
show
project
other
show
logout
• top 100 mark
• your rank
• ps level rank , attempt
• ds level rank attempt
• fa percentage
• cgpa points
• full stack report
• eligible company
• company attended details'
• all course
• registrered courses
• eligible courses
• schedule
• add project