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

Stay Ahead

Student Name : **ARAVINDHAN V**

Seat No : **140**

Project ID : **04**

Project Title : **Student Ranking Dashboard**

STUDENT RANKING DASHBOARD

1. Problem Statement

The challenge is to develop a Student Ranking Dashboard that provides a comprehensive platform for students and faculty to view and analyze student performance rankings based on various academic criteria within the college.

2. Introduction

2.1 Purpose

The purpose of this document is to describe the workflow and key components of the Student Ranking Dashboard project.

2.2 Scope

The Student Ranking Dashboard will provide students and faculty with a platform to view and analyze student performance rankings based on various academic criteria within the college.

2.3 Definitions and Abbreviations

- SRS: Software Requirements Specification
- UI : User Interface

3. Overall Description

3.1 Product Perspective

The Student Ranking Dashboard is a web application that interacts with a database to manage and display student rankings.

3.2 User Classes and Characteristics

- **Students:** College students who use the dashboard to view their rankings.
- **Faculty:** College faculty members who manage and analyze student rankings.

3.3 Operating Environment

The application will be accessible through any web browser on both desktop and mobile devices.

3.4 Design and Implementation Constraints

The application will be developed using the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js.

4. Specific Requirements

4.1 Functional Requirements

- ➢ **User Registration:** Students and faculty can create accounts.
- ➢ **User Login:** Registered users can log in to the system.
- ➢ **View Rankings:** Users can view rankings based on selected criteria.
- ➢ **Criteria Selection:** Users can filter rankings by grades, attendance, etc.
- ➢ **Admin Panel:** Admins can manage ranking criteria and update student data.
- ➢ **Notifications:** Users receive updates and notifications regarding rankings.
- ➢ **Data Visualization:** Rankings are displayed using charts and graphs.

4.2 Non-functional Requirements

- **Performance:** The application should perform efficiently even with multiple users.
- **Security:** Ensure that all user data is securely stored and protected.
- **Usability:** The UI should be intuitive and user-friendly.
- **Reliability:** The application must be consistently available for users.

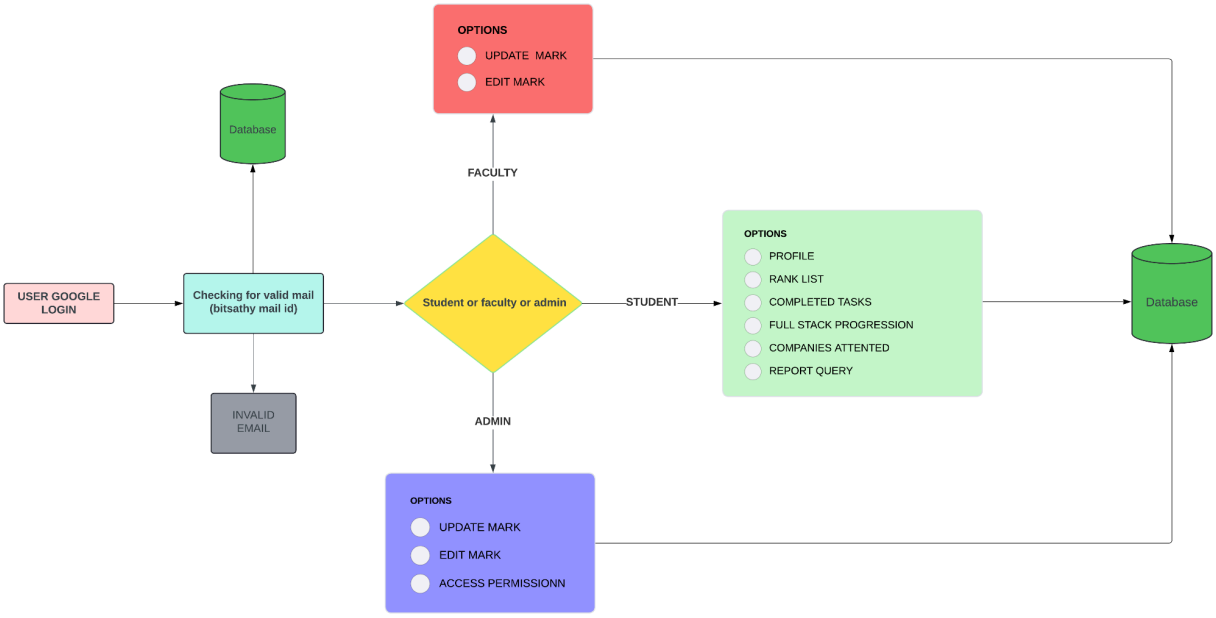
4.3 User Interfaces

- ❖ **Login Page:** For user authentication.
- ❖ **Dashboard Page:** Displays ranking data and allows interaction.
- ❖ **Admin Panel:** For administrators to manage data and criteria.

4.4 System Interfaces

- **Database:** Connects with MongoDB for storing and retrieving data.

5. Flow Chart



6. Workflow

6.1 User Registration

Users (students and faculty) register on the platform, providing necessary details and creating accounts.

6.2 User Login

Registered users log in to access their personalized dashboards where they can view and analyze rankings.

6.3 Data Input

Admins upload or input student performance data into the system, including grades, attendance, and extracurricular activities.

6.4 Ranking Calculation

The system automatically calculates rankings based on predefined criteria such as academic performance, attendance, and extracurricular participation.

6.5 Data Visualization

The dashboard displays the rankings in various visual formats, such as lists, graphs, and charts, enabling users to easily analyze performance trends.

7. Application Features

7.1 User-side Features

- **Sort and Filter:** Users can organize rankings based on different criteria.
- **Historical Data:** Users can view past ranking data.
- **Comparison:** Users can compare rankings of different students.
- **Personalized Notifications:** Alerts for ranking updates.
- **Performance Insights:** Suggestions to improve student rankings.
- **Export Data:** Ability to download ranking data.

7.2 Admin-Side Features

- **Criteria Management:** Enable admins to manage and update criteria for ranking students.
- **Data Management:** Provide admins with the ability to update and manage student performance data.
- **User Management:** Allow admins to manage user accounts, including registration approvals and role assignments.
- **Audit Trail:** Maintain an audit trail of changes made to the ranking criteria and student data.
- **Bulk Data Upload:** Enable admins to upload student performance data in bulk using CSV or Excel files.

8. Technology Stack

8.1 Backend

- **Node.js:** Used for server-side logic and handling HTTP requests.
- **Express.js:** Framework for building the backend API and handling routing.

8.2 Frontend

- **React.js:** Used for building the user interface, providing a dynamic and responsive experience for users.

8.3 Database

- **MongoDB:** NoSQL database used for storing student information and ranking data.

8.4 Additional Libraries and Components

- **React Router:** For client-side routing in the React.js application, enabling navigation between different views.
- **Material-UI:** React component library for building a modern and visually appealing user interface.
- **Chart.js/D3.js:** JavaScript libraries for creating interactive and visually appealing charts and graphs.