University Explorer

# Group Details

## Group 1

|  |  |
| --- | --- |
| Registration Number | Name |
| 220911470 | Vannela Harshavardhan Reddy |
| 220911440 | Vanshaj Rai |
| 220911536 | Prashast Saxena |

# Project Abstract

# University Explorer is a comprehensive web application designed to help students discover and evaluate universities worldwide. Built using the MERN stack (MongoDB, Express.js, React, Node.js), the platform provides an intuitive interface for exploring universities, their academic programs, and user reviews.

# The application features a robust authentication system with role-based access control, allowing regular users to browse universities, save favorites, and submit reviews, while administrators can manage universities, programs, and users. The platform enhances the university selection process by consolidating information about institutions, their rankings, and academic offerings in one accessible location.

Key functionalities include:

User authentication and profile management

University and program browsing with advanced filtering options

Favorites management for registered users

User reviews and ratings for universities

Comprehensive admin dashboard for content management

Responsive design for seamless mobile and desktop experience

# High Level Design

System Architecture

The application follows a classic client-server architecture with separated frontend and backend codebases:

University Explorer

├── Frontend (React.js)

│ ├── Public Assets

│ └── Source Code

│ ├── Components

│ ├── Pages

│ ├── Context

│ └── API Services

└── Backend (Node.js/Express)

├── API Routes

├── Controllers

├── Models

├── Middleware

└── Database Connection

API Structure

The backend exposes several RESTful API endpoints:

Authentication API (auth.js)

* /api/auth/register - Register new users
* /api/auth/login - Authenticate users
* /api/auth/me - Get current user profile

Universities API (universities.js)

* GET, POST, PUT, DELETE operations for university records
* Filtering, sorting, and pagination capabilities

Programs API (programs.js)

* Full CRUD operations for academic programs
* Links programs to specific universities

User API (user.js)

* Profile management
* Favorites management

Admin API (admin.js)

* User management
* System administration features

Database Architecture

The MongoDB database uses the following data models:

User Model (User.js)

* Personal information (name, email)
* Authentication details (password - hashed)
* Role (user/admin)
* Favorites (references to universities)

University Model (University.js)

* Basic information (name, location, ranking)
* Description and website
* References to associated programs

Program Model (Program.js)

* Program details (name, duration, degree type)
* Reference to parent university

Review Model (Review.js)

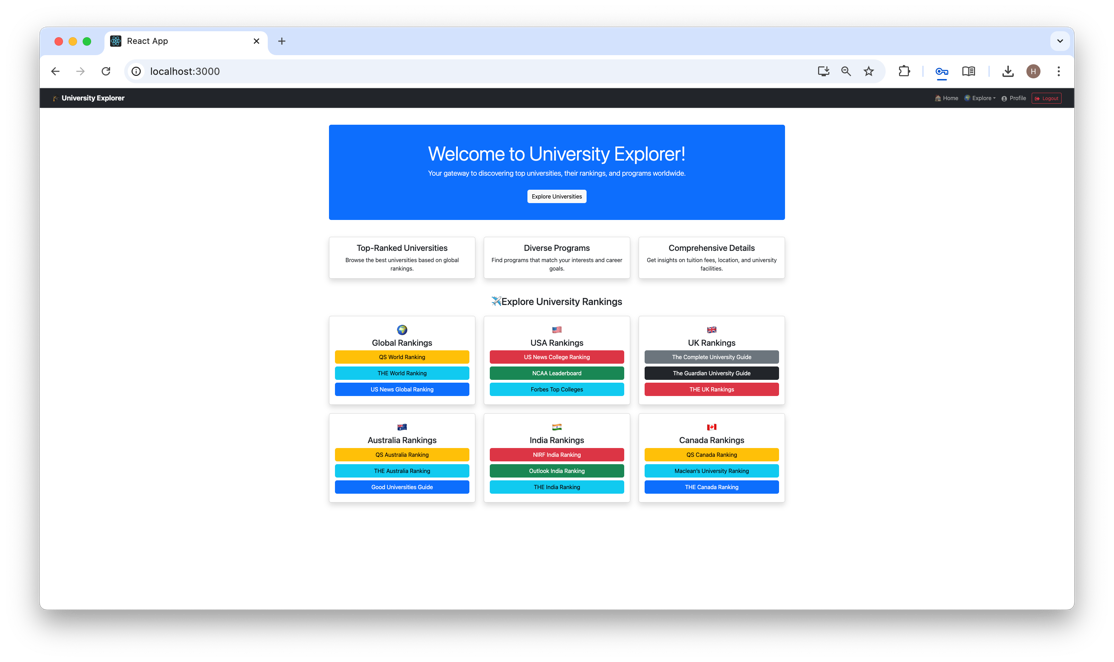
* Rating and comments
* References to university and user

# Components & Concepts Used

| **Component/Concept** | **Description** | **Purpose in Application** |
| --- | --- | --- |
| JWT Authentication | JSON Web Token based auth system | Secure user authentication and authorization |
| MongoDB/Mongoose | NoSQL database and ODM | Data storage and management with flexible schema |
| Express.js | Web framework for Node.js | Backend API routing and middleware management |
| React | Frontend library | Building responsive UI components |
| Context API | State management | Managing application-wide state (auth context) |
| React Router | Client-side routing | Navigation between different views/pages |
| Axios | HTTP client | API communication between frontend and backend |
| Bcrypt.js | Password hashing library | Secure storage of user passwords |
| Express Validator | Input validation | Sanitizing and validating user inputs |
| Reactstrap | UI component library | Styled UI components based on Bootstrap |
| Middleware Pattern | Request processing pipeline | Authentication, error handling, input validation |
| Role-based Access Control | Permission management | Securing admin-only routes and features |
| RESTful API Design | API architecture pattern | Standardized communication between client and server |

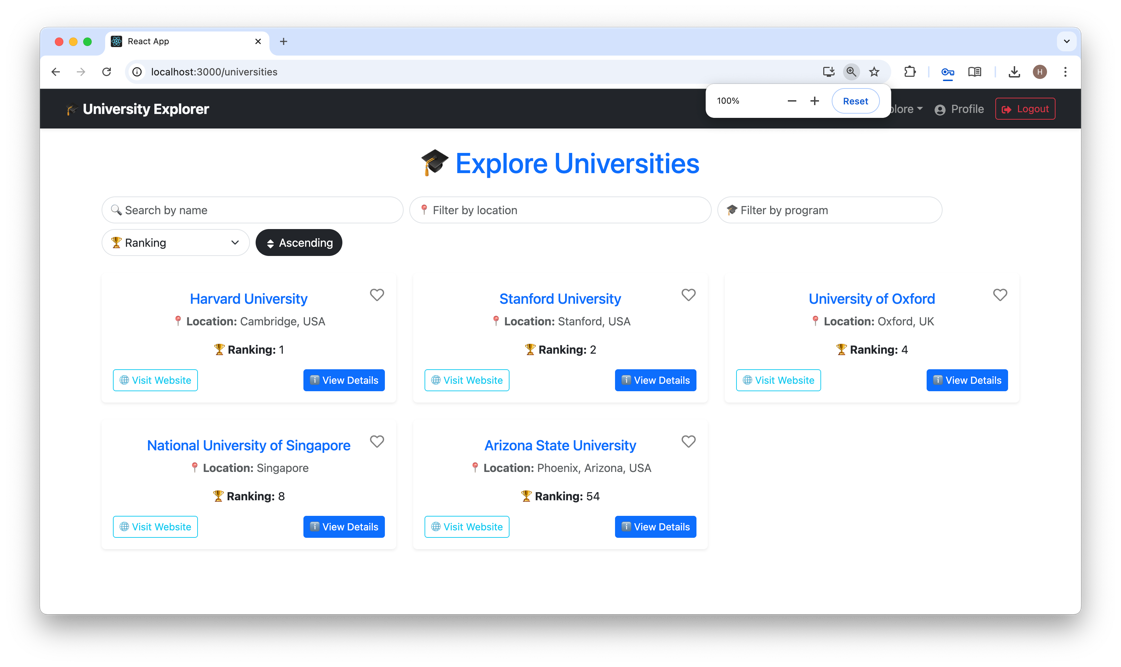
# Screenshots & Descriptions

## Home Page



Home Page opens on launching the website. We can click on the links which redirect to those websites. Clicking on Explore Universities redirects to login.

# User Universities



This is the User View of the Universities Page. Sorting by various methods are implemented. Clicking on the buttons takes you to the respective pages.

## User Favorites

A screenshot of a computer

AI-generated content may be incorrect.

User Favourites are displayed. The Remove From Favourites removes the University and updates the Database as well.

## University Details

A screenshot of a computer

AI-generated content may be incorrect.

Specific University Details for each university is a page. It is enabled for both admin type and user type logins.

## Admin Dashboard

A screenshot of a computer

AI-generated content may be incorrect.

This page is enabled only for Admin Logins. The respective links take you to the respective pages. Delete is enabled for all the pages.

## Add Program

A screenshot of a computer

AI-generated content may be incorrect.

Adding Programs for a particular University is enabled only for admin logins. Similar pages for adding universities and users are also present.

## Manage University

A screenshot of a computer

AI-generated content may be incorrect.

Admin only feature. Similar pages for users and programs is also enabled. Delete functionality is working.

# Individual Contribution Details

|  |  |  |
| --- | --- | --- |
| Registration Number | Name | Individual Contirbution |
| 220911470 | Vannela Harshavardhan Reddy | Backend Code and Database Design |
| 220911440 | Vanshaj Rai | Frontend Code and Design |
| 220911536 | Prashast Saxena | Report, Research and Documentation |

# Conclusion & Future Scope

University Explorer is a comprehensive web application designed to help users explore universities and their programs worldwide. By leveraging the MERN stack (MongoDB, Express.js, React, Node.js) and utilizing Bootstrap/ReactStrap for styling, the application provides a robust and scalable solution for managing and displaying university data. The application allows users to search for universities based on various criteria, view detailed information about each university, and manage their favorite universities. The use of modern web development technologies and best practices ensures a responsive and interactive user experience. Overall, University Explorer successfully meets its goal of providing a user-friendly platform for exploring universities and their programs.

## Future Scope

* Enhanced Search Functionality
  + - Description: Improve the search functionality by adding more filters and advanced search options.
    - Implementation: Implement additional filters such as tuition fees, acceptance rates, and campus facilities. Integrate fuzzy search algorithms to handle misspellings and partial matches.
* User Reviews and Ratings:
  + - Description: Allow users to leave reviews and ratings for universities.
    - Implementation: Create a review and rating system where users can submit their feedback on universities. Display average ratings and user reviews on the university details page.
* Recommendation System:
  + - Description: Implement a recommendation system to suggest universities based on user preferences and search history.
    - Implementation: Use machine learning algorithms to analyze user behavior and preferences. Provide personalized university recommendations on the home page and search results page.
* Mobile Application:
  + - Description: Develop a mobile application to provide a seamless experience on mobile devices.
    - Implementation: Use React Native to build a cross-platform mobile application that mirrors the functionality of the web application. Ensure the mobile app is optimized for performance and usability.
* Admin Dashboard Enhancements:
  + - Description: Enhance the admin dashboard with more features and analytics.
    - Implementation: Add features such as user management, detailed analytics on user activity, and university performance metrics. Provide visualizations and reports to help administrators make informed decisions.
* Internationalization and Localization:
  + - Description: Support multiple languages and regional settings to cater to a global audience.
    - Implementation: Implement internationalization (i18n) and localization (l10n) to support multiple languages. Allow users to select their preferred language and regional settings.
* Integration with External APIs:
  + - Description: Integrate with external APIs to provide additional data and services.
    - Implementation: Integrate with APIs such as Google Maps for campus locations, LinkedIn for alumni information, and government databases for accreditation and ranking data.

By focusing on these future enhancements, University Explorer can continue to evolve and provide even more value to its users. These improvements will help the application stay relevant and competitive in the ever-changing landscape of educational technology.