System Design for Campus Management System using MERN Stack

1 Introduction

This document outlines the architectural design and implementation strategy for a Campus Management System built using the MERN stack (MongoDB, Express.js, React.js, Node.js). The system supports multiple user roles and provides modules for grievance management, academic resource tracking, and internship opportunities.

2 Technology Stack

2.1 Frontend

- React.js
- Context API or Redux for state management
- Axios for API communication
- Tailwind CSS for UI

2.2 Backend

- Node.js with Express.js
- JWT for authentication
- bcrypt for password hashing
- Role-Based Access Control (RBAC)

2.3 Database

- MongoDB
- Mongoose for schema modeling

3 High-Level Architecture

The system follows a three-tier architecture:

- Presentation Layer – React frontend
- Application Layer – Express APIs
- Data Layer – MongoDB

Flow:
User → React Client → Express Server → MongoDB

4 Role-Based Authentication

The platform implements RBAC to control system access.

4.1 Roles

- Student
- Faculty
- Authority
- Admin

4.2 Authentication Flow

1. User logs in with credentials.
2. Server validates password using bcrypt.
3. JWT token is generated containing:
    - userId
    - role
    - token expiry
4. Token is stored in an HTTP-only cookie.
5. Protected routes verify the token via middleware.

4.3 Authorization

Middleware checks user roles before granting access:

- Only Admin can create users.
- Authority handles grievance approvals.
- Faculty uploads academic resources.
- Students submit grievances and access materials.

5 Database Design

MongoDB collections:

5.1 Users

- id
- name
- email
- passwordHash
- role
- createdAt

5.2 Grievances

- id
- studentId
- title
- description
- status (Pending, In Progress, Resolved)
- timestamps

5.3 Courses

- id
- courseName
- instructorId
- credits

5.4 Resources

- id
- courseId
- fileURL
- uploadedBy

5.5 Internships

- id
- title
- organization
- description
- eligibility
- deadline
- postedBy

6 Module Design

6.1 Grievance Management System

Workflow:
Student → Submit grievance → Authority review → Status update
Status tracking improves transparency and accountability.

6.2 Academic Resource and Course Tracking

Faculty members can:

- Create courses
- Upload lecture notes
- Share assignments

Students can view and download resources via the dashboard.

6.3 Internship / Research Portal

Admins or faculty post opportunities. Students can browse and apply.
Future enhancement: automated email notifications.

7 API Structure

- /auth/login
- /auth/register
- /grievances
- /courses
- /resources

- /internships

Each route is protected using authentication middleware.

8 Security Considerations

- Password hashing using bcrypt
- JWT expiration
- HTTPS enforcement
- Rate limiting to prevent brute-force attacks
- Input validation against injection attacks

9 Scalability Recommendations

- Use Redis for caching sessions
- Implement refresh tokens
- Store files in AWS S3 instead of the local server
- Add logging and monitoring

10 Future Enhancements

- Microservices architecture
- AI-based grievance categorization
- Real-time notifications using WebSockets
- Mobile application support

11 Conclusion

The proposed MERN-based architecture provides a secure, modular, and scalable foundation for a campus management platform. By implementing RBAC, structured APIs, and modular services, the system is capable of supporting institutional-scale workloads while remaining extensible for future features.