COLLEGE CODE:8203

COLLEGE NAME:AVC COLLEGE OF ENGINEERING

DEPARTMENT:B.E-CSE

STUDENT NM ID: 7F0D98980E33FA58E37451488740DBF5

ROLL NO:23CS98

DATE:15.09.2025

Completed the project named as

Phase2

TECHNOLOGY PROJECT NAME: ADMIN DASHBOARD WITH CHARTS

SUBMITTED BY,

NAME: T.SIVARANJANI

MOBILE NO:7871098619

Tech Stack Selection

To build the IBM-NJ Admin Dashboard with charts, the right technology stack is crucial for scalability, performance, and integration.

- **Frontend:**
  - **React.js** (for dynamic UI rendering, reusable components, and performance optimization)
  - **Tailwind CSS / Bootstrap** (for responsive and modern UI styling)
  - **Charting Libraries:** Recharts, Chart.js, or D3.js (to visualize data effectively with interactive charts)
- **Backend:**
  - **Node.js + Express.js** (lightweight and scalable backend framework for APIs)
  - **Spring Boot (Java)** or **Flask/Django (Python)** can also be considered based on team expertise.
- **Database:**
  - **PostgreSQL / MySQL** (for structured relational data)
  - **MongoDB** (if dashboard requires flexible schema for analytics)
- **Authentication & Security:**
  - JWT (JSON Web Tokens) for secure login/session handling
  - OAuth 2.0 for third-party integrations
- **Deployment & Hosting:**
  - IBM Cloud, AWS, or Azure for scalable deployment
  - Docker + Kubernetes for containerization and orchestration

UI Structure / API Schema Design

UI Structure:

- **Login Page:** Secure authentication for admins
- **Dashboard Overview:** High-level metrics (e.g., number of users, tasks, logs, performance KPIs)

- **Navigation Panel:** Sidebar menu for navigating modules (Users, Reports, Charts, Settings, etc.)
- **Charts & Reports Section:**
  - Line charts (trend over time)
  - Bar charts (comparisons across categories)
  - Pie/Donut charts (proportions and distribution)
  - Heatmaps (activity patterns)
- **Settings:** Role management, API configurations, theme customization

API Schema Design:

- **/api/auth/login** → Authentication & session token generation
- **/api/users** → Manage user data (CRUD operations)
- **/api/dashboard/stats** → Fetch metrics for dashboard summary
- **/api/reports** → Generate and fetch reports in JSON/CSV format
- **/api/charts** → Provide chart-ready datasets (grouped, aggregated data)

Data Handling Approach

- **Data Sources:** Extracted from IBM-NJ system logs, user databases, and operational metrics
- **ETL (Extract, Transform, Load):**
  - Data Cleaning (removing duplicates, missing values)
  - Transformation (grouping, aggregating, normalizing data for charts)
  - Storage in optimized DB tables or NoSQL collections for faster queries
- **Caching:** Use Redis/Memcached to speed up frequently accessed metrics
- **Real-Time Data Handling:** Implement WebSockets or Kafka for real-time updates on charts
- **Security Measures:** Data encryption (AES-256), role-based access control (RBAC), logging & monitoring

Component / Module Diagram

Modules of the Dashboard:

- **Authentication Module** – login, signup, user verification
- **User Management Module** – admin control over users & roles
- **Dashboard Module** – charts, KPIs, analytics
- **Reporting Module** – export reports (PDF, CSV, Excel)
- **Settings Module** – configuration, themes, access control

Component Diagram Example (high-level):

- **Frontend Components:** Navbar, Sidebar, Chart Components, Report Viewer, User Table
- **Backend Components:** API Gateway, Authentication Service, User Service, Chart Service, Report Service
- **Database Layer:** User DB, Metrics DB, Logs DB

Basic Flow Diagram

Workflow:

1. **Admin Login** - Validates credentials via Authentication Service
2. **Dashboard Load-** Backend fetches stats & sends chart-ready JSON data
3. **Charts Render-**UI components display data in real-time with Chart.js/Recharts
4. **User Management** -Admin can add/remove/update users via User Service
5. **Reports Generation-** Data processed & exported in chosen format
6. **Settings Update** - Admin modifies configurations, which update API behavior