**COLLEGE CODE: 8203**

**COLLEGE: AVC COLLEGE OF ENGINEERING**

**DEPARTMENT: INFORMATION TECHNOLOGY**

**STUDENT NM-ID: 4D2198FA90720AD8F834BBD4C69121C9**

**ROLL NO: 23IT100**

**DATE:08-09-2025**

**Completed the project named as Phase 1**

**TECHNOLOGY PROJECT NAME: Admin dashboard with charts**

**SUBMITTED BY,**

**NAME: Sivarajaganapathi S**

**MOBILE NO: 7845102808**

## Problem Understanding & Requirements

### Problem Statement

Managing and monitoring a complex application requires a centralized, real-time view of key performance indicators (KPIs) and operational metrics. Current practices often involve manual data extraction, disparate reporting tools, or static spreadsheets, leading to several challenges:

* Delayed Insights: Data is not updated in real-time, making timely decision-making difficult.
* Information Overload: Raw data from various sources is hard to interpret without visual aids.
* Lack of Centralization: Administrators must log into multiple systems to gather a complete operational picture.
* Security Risks: Granular control over which metrics certain roles can view is often insufficient.

There is a clear need for an Admin Dashboard that:

* Provides a centralized, secure platform for monitoring core business metrics (e.g., users, sales, views).
* Retrieves real-time data from the backend using Node.js/Express APIs.
* Visualizes data using Chart.js (line, bar, pie charts) for quick and easy analysis.
* Implements role-based access control to ensure data security and relevance.
* Updates charts automatically based on fresh data fetched from the API.

This solution aims to transform raw data into actionable intelligence for system administrators and business stakeholders.

### Users & Stakeholders

#### Users

These are the people who directly interact with the Admin Dashboard:

* System Administrators: Monitor system health, server load, and technical performance metrics.
* Business Managers: Track key business KPIs like sales, revenue, and customer acquisition rates.
* Marketing & Sales Teams: Analyze campaign performance, traffic sources, and conversion funnels.
* Product Managers: View feature usage, user engagement, and retention metrics.

#### Stakeholders

These are individuals or groups who have an interest in the system but may not directly use the dashboard:

* Project Development Team: Responsible for coding the backend (Node.js/Express/MongoDB) and integrating Chart.js on the frontend.
* Data Engineers: Ensure the integrity, availability, and structure of the underlying data (MongoDB).
* Security & Compliance Team: Define and audit the role-based access control (RBAC) rules.
* Investors / Company Leadership: Rely on the dashboard metrics for strategic planning and evaluating business health.

### User Stories & Stakeholder Stories

#### User Stories

* As a Business Manager, I want to see a line chart of daily sales over the last 30 days, so that I can quickly identify trends and peak periods.
* As a System Administrator, I want to view a real-time chart of server load/API calls, so that I can detect and respond to performance bottlenecks immediately.
* As a Marketing Manager, I want a pie chart showing user sign-ups by geographical region, so that I can allocate ad spend more effectively.
* As a Product Manager, I want to see a bar chart of feature usage, so that I can prioritize future development efforts.

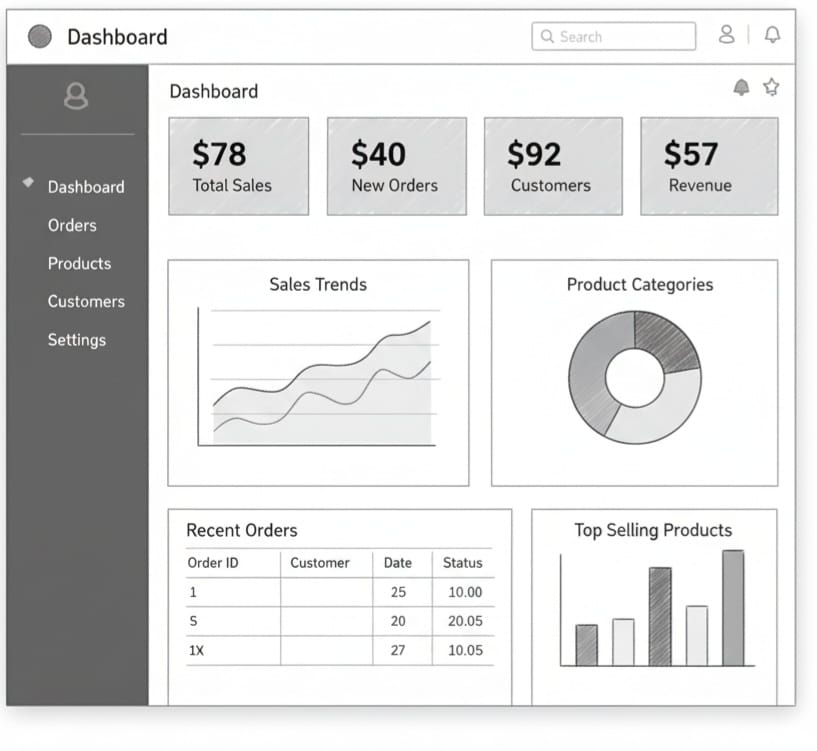
#### Stakeholder Stories

* As a Developer, I want to design clear API endpoints that deliver metric data in a consistent JSON format, so that the Chart.js integration is seamless.
* As a System Administrator, I want to implement MongoDB aggregation queries efficiently, so that complex metrics are computed and served quickly.
* As a Security Team Member, I want to enforce role-based access control on all API endpoints, so that sensitive data is only visible to authorized personnel.

### MVP [Minimum Viable Product] Features

1. Metric Computation Backend (Node.js/Express/MongoDB)
   * Backend logic to compute at least three core metrics (e.g., Total Users, Daily Sales, Total Views).
   * APIs must be protected by a basic authentication layer.
2. API Endpoint for Metrics
   * A single API endpoint (e.g., /api/metrics/summary) that returns all dashboard data in a structured JSON object.
3. Basic Chart Display (Frontend)
   * Display the three core metrics using Chart.js.
   * At least one chart must be a Bar Chart and one a Line Chart.
4. Auto-Update Functionality
   * The frontend charts automatically refresh/update by fetching new data from the API every 30 seconds.
5. Role-Based Data Filtering (Simplified)
   * A basic mechanism to show/hide one specific metric chart based on the logged-in user's role (e.g., only "Admin" role sees the "Sales" chart).

### Wireframes & API Endpoint List



#### API Endpoint List

| Endpoint | Method | Description | Request Parameters | Response |
| --- | --- | --- | --- | --- |
| /api/metrics/chartdata | GET | Fetch detailed data for a specific chart. | chartName (query) – e.g., 'salesTrend' | JSON: { labels: ["Mon", "Tue", ...], data: [100, 150, ...] } |
| /api/auth/login | POST | Authenticate a user and return their role. | username, password (body) | JSON: { token, role: "Admin" } |

### Acceptance Criteria

#### Data Computation & API

* The backend must successfully compute and return JSON data for the three core metrics.
* The API endpoint /api/metrics/summary must respond in under 2 seconds.
* The API must return a 403 Forbidden error if an unauthorized user attempts to access protected metric data.

#### Chart Visualization (Frontend)

* The frontend must use Chart.js to render the fetched data as a visually clear and readable Bar Chart and Line Chart.
* Chart axes, labels, and titles must be clearly defined and derived from the JSON data.
* Charts must accurately reflect the data received from the backend API.

#### Performance & Updates

* Charts must refresh data from the API automatically every 30 seconds.
* The auto-update process should be smooth and not disrupt the user interface.

#### Access Control

* When a user with the "Basic" role logs in, they should not see the "Sales" chart.
* When a user with the "Admin" role logs in, they must see all charts, including the "Sales" chart.