Full Stack Development with MERN

Project Documentation format

1. Introduction

- **Project Title:** *Cosmetic Insights: Navigating Cosmetics Trends and Consumer Insights with Tableau*
- **Team Members & Roles:**
  - Member 1 – Frontend Developer (React)
  - Member 2 – Backend Developer (Node.js, Express.js)
  - Member 3 – Database & ETL Integration (MongoDB, APIs)
  - Member 4 – Tableau Designer & Data Analyst

2. Project Overview

- **Purpose:**
  To provide cosmetic brands and marketers with an intuitive platform that visualizes consumer trends and sentiment insights using data analytics and interactive Tableau dashboards.
- **Features:**
  - User registration and login system
  - Upload and integration of CSV/API data
  - Real-time trend analysis using Tableau visualizations
  - Interactive dashboards (bar, pie, map, line)
  - Admin panel for managing data and stories
  - Secure API with JWT authentication

3. Architecture

- **Frontend (React):**
  - Built using React with functional components and hooks
  - React Router for navigation
  - Axios for API calls
  - Redux (optional) for global state management
- **Backend (Node.js + Express.js):**

    - RESTful APIs to handle CRUD operations, user management, and data uploads
    - Middleware for validation, authentication, and error handling
- **Database (MongoDB):**
    - Collections:
        - users: stores user profiles and auth details
        - datasets: uploaded cosmetic trend data
        - dashboards: links to Tableau dashboard metadata
        - stories: stored visual narrative content

4. Setup Instructions

- **Prerequisites:**
    - Node.js (v18+)
    - MongoDB Community Edition or Atlas
    - Git
- **Installation Steps:**

bash

CopyEdit

```
git clone https://github.com/your-repo/cosmetic-insights-mern.git

cd cosmetic-insights-mern
```

- Frontend:

bash

CopyEdit

```
cd client

npm install
```

- Backend:

bash

CopyEdit

```
cd server
```

```
npm install
```

- **Environment Variables (.env example):**

ini

CopyEdit

```
PORT=5000
MONGO_URI=your_mongodb_connection_string
JWT_SECRET=your_jwt_secret
TABLEAU_API_KEY=your_tableau_key_if_needed
```

5. Folder Structure

- **Client (React):**

bash

CopyEdit

```
/client
 /public
 /src
  /components
  /pages
  /services
  /utils
  App.js
  index.js
```

- **Server (Node.js + Express):**

bash

CopyEdit

```
/server
 /controllers
 /models
 /routes
 /middleware
 /config
 server.js
```

6. Running the Application

- **Frontend:**

```
bash
CopyEdit
cd client
npm start
```

- **Backend:**

```
bash
CopyEdit
cd server
npm start
```

7. API Documentation

Endpoint	Method	Description	Request Body/Params	Response Example
/api/register	POST	Register a new user	{ email, password }	{ token, user }
/api/login	POST	Authenticate user	{ email, password }	{ token, user }
/api/data/upload	POST	Upload data file	FormData (CSV file)	{ message: "Uploaded" }
/api/dashboard	GET	Get Tableau dashboard info	JWT token in header	Dashboard metadata

8. Authentication

- JWT-based authentication
- Secure routes using middleware
- Tokens stored in HTTP-only cookies or localStorage

- Role-based access (admin, analyst, viewer)

9. User Interface

- Responsive and modern UI using Material UI / Bootstrap
- Dashboard View:
  - Embedded Tableau dashboards
  - Interactive filters (dropdowns, sliders)
- Upload Page:
  - Drag-and-drop CSV uploader
- Admin Panel:
  - Manage datasets, users, and Tableau links

(Insert UI screenshots or GIFs)

10. Testing

- **Tools Used:** Jest, React Testing Library, Postman
- **Testing Strategy:**
  - Unit tests for React components and utility functions
  - Integration tests for Express APIs
  - Manual test cases for dashboard interactions

11. Screenshots or Demo

- 📸 Screenshots:
  - Home page
  - Data upload panel
  - Embedded Tableau dashboard
  - User login page
- 🔗 Demo Link: *https://drive.google.com/file/d/1quTu-BEOgwqqOI0Kx45MYPkLDDEYX6iS/view?usp=sharing*

12. Known Issues

- API integration for real-time social media data is limited to demo tokens
- Tableau Public embedding may have viewer restrictions

- Some features may break in mobile view (under testing)

13. Future Enhancements

- Real-time data stream integration (Kafka or WebSockets)
- AI-based trend predictions using ML models
- Full Tableau API integration for dynamic dashboard creation
- Integration with external e-commerce platforms (Amazon, Sephora)
- Mobile-friendly Progressive Web App (PWA) version