**Project Report: Datalytics**

*-Zainab Jambughodawala (web development intern)*

**1. Project Overview**

**Datalytics** is a full-stack web application designed to provide data analytics and visualization features for Excel files. Built using the MERN (MongoDB, Express.js, React, Node.js) stack, it allows users to upload .xlsx files, preview data, extract AI-based insights, and visualize data in both 2D and 3D chart formats.

**2. Technology Stack**

* **Frontend**: React.js (with Tailwind CSS for styling, Chart.js, Three.js for 3D)
* **Backend**: Node.js with Express
* **Database**: MongoDB (Mongoose ODM)
* **AI Insight Engine**: Custom rule-based logic
* **Visualization Libraries**: Chart.js, react-three-fiber (Three.js wrapper)

**3. Features**

* User authentication and role-based access (Admin vs User)
* Excel file upload and real-time preview
* AI-generated insights based on uploaded data
* Multiple chart types (Bar, Line, Pie, Radar)
* Optional 3D visualization (Bar3D using Three.js)
* Admin dashboard for user and upload management

**4. Frontend File Structure (React)**

Located inside /frontend/src

**/components**

* App.jsx: Main app component configuring routes (React Router).
* UploadFile.jsx: File upload UI, sends file to backend, and stores data context.
* Visualization.jsx: Handles rendering of selected chart type using Chart.js and Three.js.
* ExcelDataContext.jsx: React context provider to share uploaded Excel data across components.
* Dashboard.jsx: Landing page post-login with user options.
* AdminPanel.jsx: Restricted component displaying user and file upload data, with delete options.
* InsightSummary.jsx: Displays AI insights after file upload.
* AuthForm.jsx: Login and Signup Authentication screens.

**Other Files**

* main.jsx: ReactDOM entry point.
* App.css, index.css: Global styles.
* tailwind.config.js: Tailwind CSS configuration.
* vite.config.js: Configuration for Vite bundler.

**5. Backend File Structure (Express.js)**

Located inside /backend

**/routes**

* analyticsRoutes.js: Handles upload, fetch, preview, delete, and insight generation.
* adminRoutes.js: Admin-only endpoints for user and upload listing/deletion.

**/controllers**

* analyticsController.js: Contains logic for analyzing and saving Excel files to DB.

**/models**

* ExcelData.js: Schema for storing structured Excel data.
* Uploads.js: Schema for uploaded file metadata including preview rows and timestamps.

**/utils**

* summaryGenerator.js: Generates AI insights using simple summarization logic on sheet data.

**Other Files**

* index.js: Entry point of Express server. Sets up routes and middleware.
* .env: Holds sensitive configuration (DB URI, port).
* example.env: Template showing required environment variables.

**6. Database Design**

**Collections:**

* **Users**: Stores name, email, and hashed password. Admins are hardcoded based on email.
* **Uploads**: Stores file name, user email, upload time, and first few preview rows.
* **ExcelData**: Stores detailed content of uploaded Excel files.

**7. Admin Panel**

* Accessible only to hardcoded email your\_admin\_email@gmail.com
* View all users and uploaded files
* Delete any user or uploaded file

**8. Deployment & Setup**

**Frontend:**

cd frontend

npm install

npm run dev

**Backend:**

cd backend

npm install

npm run start

Ensure .env file is configured using example.env.

**9. Possible Enhancements**

* Role-based JWT Authentication
* Insight Engine powered by LLM (e.g., GPT)
* User-specific dashboards and history
* Export charts and insights as PDF/Excel

**10. Conclusion**

Datalytics offers a clean, modular, and extensible architecture to explore data analytics visually. It serves as a solid foundation for future feature expansion like predictive analytics, ML-based recommendations, and interactive dashboards.