Test Assignment – Fullstack Development

Frontend (React + Redux)

Create a local React frontend application with Redux for state management.

Requirements:

- File upload for `.csv` files

- Status display for each uploaded file:

  - "Processing" immediately after upload

  - "Error" if the CSV file contains 0 rows

  - "Done" if the file contains more than 0 rows (after a 3-second delay)

- Display all uploaded files in a table with the following columns:

  - File name

  - Upload timestamp

  - Status

- Use Redux to manage application state

Backend (local simulation of AWS Lambda + S3 + DynamoDB + Parquet)

Simulate the backend behavior of an AWS-based system using a local Python server (e.g., FastAPI or Flask).

Requirements:

1. File Upload

  - Implement a POST /upload endpoint that accepts a CSV file (e.g., via multipart/form-data)

  - Save the file locally, e.g., to ./uploads/

2. Processing

  - After the file is uploaded, automatically convert it to Parquet format

  - Store the .parquet file locally, e.g., in ./parquet/

3. Metadata Storage

  - Store metadata in a local SQLite database (metadata.db) with a table containing at least:

- File name

- Upload timestamp

- Number of rows

- Path to the Parquet file

- Processing status

4. Retrieve Metadata

- Provide a GET /files endpoint that returns all metadata entries from the database as JSON

Extension: Login System

Add a simple login mechanism to the application to restrict access to authenticated users.

Requirements:

- A login screen where users can enter username and password

- After successful login, redirect the user to the main area (upload + file list)

- Use Redux to manage authentication state globally

- Backend provides a POST /login endpoint that:

  - Accepts fixed credentials (username: test, password: password)

  - On success, returns a dummy token (e.g., "abc123") that is stored in the frontend

- Access to /upload and /files endpoints should be protected using the token (basic validation is sufficient)

Submission

Please submit either a ZIP archive or a GitHub repository containing:

- The React frontend (including Redux setup)

- The Python backend (e.g., FastAPI or Flask)

- A README.md file with brief instructions for running the application locally