CreditSea Fullstack Engineer Assignment

Objective

Design and implement a fullstack application using the MERN stack (MongoDB, Express, React, Node.js) that processes XML files containing soft credit pull data from Experian. Your solution will include a backend API for file upload and data extraction, a storage mechanism for the processed data in MongoDB, and a frontend interface built in React that presents a comprehensive report.

Project Overview

You are provided with two sample XML files simulating soft credit pulls from Experian. Your task is to create an application with the following functionality:

1. **XML Upload Endpoint:**
    - Develop a RESTful API endpoint using Express and Node.js that accepts XML file uploads.
    - Ensure the endpoint validates the file format and handles errors gracefully.
2. **Data Extraction & Persistence:**
    - Parse the uploaded XML file and extract the following information:
        - **Basic Details:**
            - Name
            - Mobile Phone
            - PAN
            - Credit Score
        - **Report Summary:**
            - Total number of accounts
            - Active accounts
            - Closed accounts
            - Current balance amount
            - Secured accounts amount
            - Unsecured accounts amount
            - Last 7 days credit enquiries
        - **Credit Accounts Information:**
            - Credit Cards
            - Banks of Credit Cards
            - Addresses
            - Account Numbers
            - Amount Overdue
            - Current Balance
    - Store the extracted data in a well-designed schema in MongoDB.
3. **Reporting Frontend:**
    - Build a clean, user-friendly React frontend that retrieves and displays the stored report data.

    - The UI should clearly present the extracted information in sections (Basic Details, Report Summary, and Credit Accounts Information).

Technical Requirements

- **Backend (Node.js & Express):**
    - Create RESTful API endpoints for:
        - XML file upload.
        - Data extraction and persistence.
        - Data retrieval for the frontend.
    - Implement robust error handling and logging.
    - Use Node.js and Express as the server-side framework.
- **Frontend (React):**
    - Develop a responsive UI using React that consumes the backend APIs.
    - Ensure that the page is both aesthetically pleasing and user-friendly.
    - You may use additional libraries or tools (e.g., Redux, React Router) as needed.
- **Database (MongoDB):**
    - Use MongoDB for data persistence.
    - Document your schema design decisions.
- **Testing & Documentation:**
    - Include unit and/or integration tests where applicable.
    - Provide a README file with clear instructions on:
        - How to set up and run your application.

Evaluation Criteria

Your submission will be evaluated based on the following:

- **Functionality & Correctness:**
    - Accurate extraction and transformation of data from the XML files.
    - Robust handling of edge cases and errors.
- **API & Code Design:**
    - Clean, modular, and well-documented code.
    - Adherence to RESTful principles in API design.
    - Efficient and clear schema design for data persistence in MongoDB.
- **User Interface & Experience:**
    - A professional and intuitive React frontend that displays the report data clearly.
    - Responsiveness and usability.
- **Documentation & Testing:**
    - Clarity of the README and documentation.

Submission Guidelines

- Please host your project in a Git repository and share the link.
- Ensure that the repository contains:

    - All source code.
    - A README file with setup and run instructions.
    - Any additional documentation or test instructions.
- Please share a link to a short 3-5 minute video demo of your final product