💻 The NHIS Fraud Auditor Dashboard Assignment

🎯 Goal

Your goal is to build a minimal, functional web application (MVP) that serves as a Claims Auditor Dashboard for the NHIS Healthcare Claims and Fraud Dataset provided.

You must complete the core features, deploy the application, and submit your work within 8 hours (simulating a workday). You are expected to use AI coding assistants and LLMs extensively.

📊 Dataset

You must use the data from the following Kaggle source:

- **Dataset: NHIS Healthcare Claims and Fraud Dataset**
- **File to Use: You should primarily focus on the main claims data file(s) for the core task.**

📝 Problem Statement

You are a data developer tasked with rapidly prototyping a dashboard for a team of claims auditors. The primary function of this application is to process the NHIS claims data and use a simple heuristic to automatically flag claims with a high likelihood of fraud.

Your application must demonstrate data loading, basic analysis, a simple predictive heuristic, and a user interface for review.

🛠️ Requirements and Deliverables

Part 1: Technical Implementation (Core Features)

You may use any language, framework, database, and hosting solution (e.g., React/Node/Postgres, Python/Flask/SQLite, or serverless functions with tools like Vercel or Replit).

1. **Data Ingestion and Setup:**
    - **Create a method to load the data from the provided CSV file(s) into your application's data layer (e.g., local database, in-memory store, etc.).**
2. **The "Fraud Likelihood Score" Engine:**
    - **Design a simple, explainable, and non-ML-based heuristic to assign a "Fraud Likelihood Score" (an integer from 0 to 100) to every single claim in the dataset.**

- ***Example Heuristic Idea (You must create your own):*** **Score = (Amount billed/ Avg_Charge_for_Procedure) * 100. The complexity is up to you, but it must be functional and defendable.**

3. **Dashboard View (/):**
    - **Display three key overall metrics (e.g., Total Number of Claims, Average Claim Charge, Number/Percentage of Claims Flagged with a score > 75).**
    - **Display a simple chart (e.g., a bar chart or pie chart) visualizing the distribution of claims by their assigned Fraud Likelihood Score Category (e.g., Low: 0-25, Medium: 26-75, High: 76-100).**
4. **Claims Review View (/claims):**
    - **Display a paginated or scrollable list of all claims.**
    - **For each claim, prominently display the assigned Fraud Likelihood Score.**
    - **Include a functional search/filter bar that allows the auditor to search claims by different parameters like diagnosis, patient , etc.,**

Part 2: AI Tool Fluency and "Vibe Coding" Mandate

You are required to use LLMs/AI tools for acceleration and must document this use.

1. **Prompt Journal: In your final submission's README, include a section titled "AI Prompt Journal." This must contain:**
    - **Your five most effective prompts used for the assignment (e.g., generating a specific function, debugging a piece of code, defining the database schema, or suggesting the Fraud Heuristic).**
    - **A brief note on what the AI output was used for.**
2. **AI for Testing/Documentation: Use an LLM/AI assistant to generate one code snippet for testing (e.g., a unit test for your Fraud Likelihood Score function or a test for your API endpoint). Include the generated code and the prompt used.**
3. **Architectural Decision: Use an LLM/AI assistant to quickly compare two specific technical choices relevant to your project (e.g., "Compare using Postgres vs. SQLite for this specific application," or "Compare React vs. Vue for the frontend dashboard"). Include the prompt and a brief summary of the resulting decision you made.**

Submission & Evaluation

Evaluation Focus

Your solution will be assessed based on the following equally weighted criteria:

Criterion	What We Are Looking For
Technical Correctness	All Core Features are implemented and functional. The application is stable and performs basic CRUD/filtering operations correctly.
Code Quality	Clean, readable, well-structured, and appropriately modularized code.
AI Tool Fluency	Demonstrated ability to leverage AI effectively to accelerate development. The submitted prompts show strategic thinking and complex problem-solving.
Rapid Prototyping	The scope of the solution reflects strong prioritization and scoping to deliver a functional MVP within the time limit.