Project Name: Financial Indicator Calculator with REST API Communication

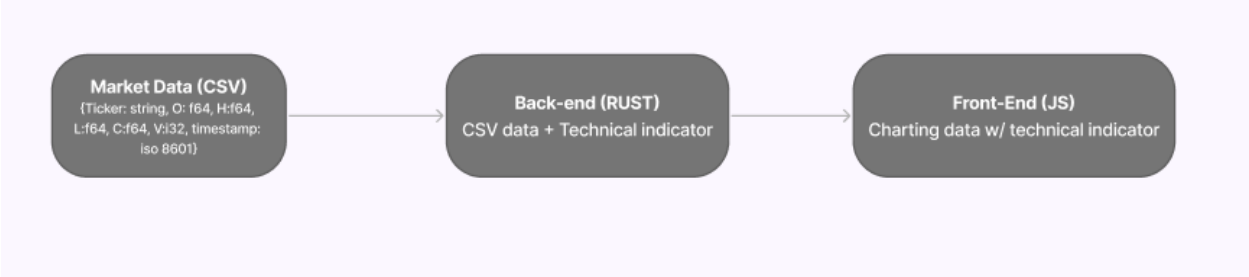
Project Overview: Create a simple financial web app that calculates a technical indicator from CSV data and displays it on a frontend. The project will involve creating a backend to read transaction data from a CSV file, calculate a financial metric (e.g., total spending or average spending by category), and expose this data via REST endpoints. The frontend will consume this data and visualize it using charts.

Key Features:

1. **Backend CSV Data Processing:** The backend will read transaction data from a CSV file. Each record in the CSV should include fields like "amount," "category" (e.g., groceries, bills), and "date."
2. **Financial Indicator Calculation:** Calculate a simple financial metric such as:
    - **Total Spending**: Calculate the total amount spent.
    - **Average Spending by Category**: Calculate the average amount spent in each category.
3. **REST Requests:** Implement basic REST API endpoints to expose the calculated metrics:
    - **GET**: Retrieve total spending or average spending by category.
4. **Frontend Data Visualization:** Use JavaScript and Chart.js to create a simple chart that visualizes the data retrieved from the backend. For example, display a bar chart showing spending per category.

Tech Stack for AI role:

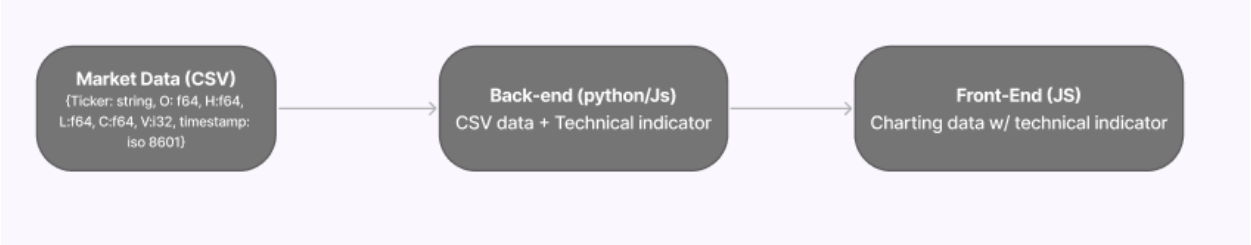
- **Backend**: Python or Node.js (Express)
- **Frontend**: HTML/CSS/JavaScript
- **Data Source**: CSV file for transaction data
- **API**: REST API to serve calculated financial metrics
- **Dockerized:** Must dockerize front and backend



Tech Stack for SWE role:

- **Backend**: Rust
- **Frontend**: HTML/CSS/JavaScript

- **Data Source**: CSV file for transaction data
- **API**: REST API to serve calculated financial metrics
- **Dockerized:** Must dockerize front and backend



Expected Learning Outcomes:

- Understand how to read and process CSV data.
- Learn to make REST requests and handle API responses.
- Practice building a simple backend to communicate with a frontend.
- Visualize data using charts.

Stretch Goals:

- Add more technical indicators
- Allow the user to upload their own CSV file for custom analysis.

Estimated Time Frame: 2-3 days for a basic version.

Resources to Get Started:

- Flask Tutorial for Beginners
- JavaScript Fetch API for Making REST Requests
- Chart.js Documentation for Visualizing Data
- Python CSV Module Documentation
- 7 Technical Indicators to Build a Trading Toolkit
- Rust for Backend Development - Mastering Backend
- Docker for Beginners: Everything You Need to Know
- pandas-ta | Technical Analysis Indicators - Pandas TA is an easy to use Python 3 Pandas Extension with 150+ Indicators
- ta - Rust