Hector Gonzalez
Sonya Cates
Data Science
March 24, 2025

——--Final Project Report #2——---

Project Title:

Real-Time Credit Card Fraud Detection Web Application

Overview:

This project aims to build a full-stack web application that checks and detects potentially fraudulent transactions using real-time data and a third-party fraud detection service.

The frontend will be built using **React.js**, and the backend will be developed using **Python (Flask)** in **Visual Studio Code**. The app will communicate with the **FraudLabs Pro API**, which analyzes transaction data and returns a fraud risk score, status, and recommendation. The results will also be logged and viewable in the FraudLabs Pro dashboard.

Data Source:

We are using a third-party API to handle data collection and fraud scoring:

- **API Name:** FraudLabs Pro
- **API Link:** https://www.fraudlabspro.com
- **API Endpoint Used:** `https://api.fraudlabspro.com/v1/order/screen`
- **API Documentation:** https://www.fraudlabspro.com/developer/api

The transaction data submitted to the API includes:

- IP address

- Email address
- Phone number
- Billing address
- Card BIN (first 6 digits of credit card)
- Transaction amount and currency

The API responds with:

- A **fraud score** (0–100 scale)
- A **status** (APPROVE, REVIEW, REJECT)
- Additional metadata such as country match, blacklist flags, and transaction ID

Objective with the Data:

We will use the API to:

- **Submit user-entered transaction data**
- **Receive fraud analysis** in real time
- **Log results** to both the user interface and the FraudLabs Pro dashboard
- **Display the analysis visually** (e.g., fraud score, risk level)

This will **simulate a real-world credit card verification process** similar to what financial institutions use behind the scenes.

Tech Stack:

- **Frontend:** React.js (user input form, results display)

- **Backend:** Python + Flask (handles API requests)
- **IDE:** Visual Studio Code
- **External Service:** FraudLabs Pro API

---

2. Work Completed So Far

- Integrated Python backend using Flask
- Successfully connected to the FraudLabs Pro API
- Created a POST request to submit transaction data
- Parsed the **XML response** using `xml.etree.ElementTree`
- Confirmed that responses are **logged to the FraudLabs Pro dashboard**
- Debugged issues with JSON parsing (API returns XML by default)
- Built the base structure for a React frontend with Axios to submit form data

Challenges Encountered:

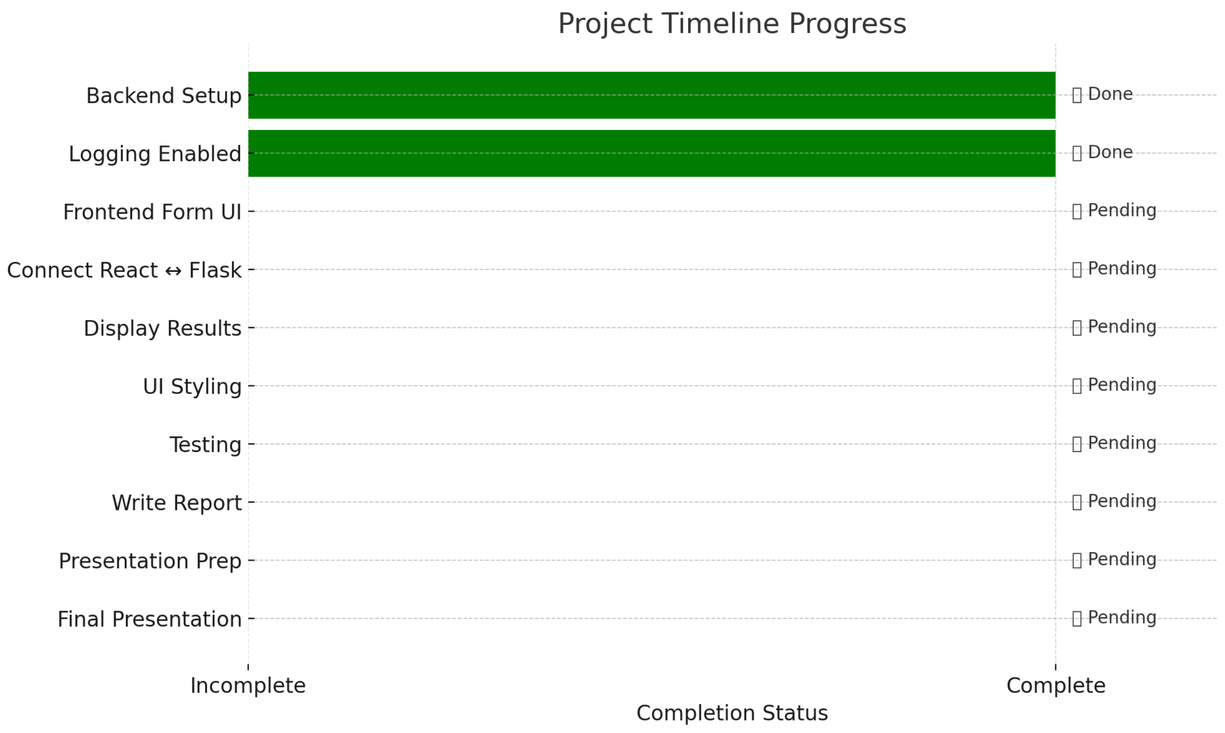
- The API initially returned XML instead of JSON, causing parsing failures
- Had to adjust from `GET` to `POST` with `action=CHECK` to properly log transactions
- Dealt with limited documentation on dashboard visibility conditions
- Required extra setup to enable cross-origin communication between Flask and React (CORS)

3. Completion Plan & Timeline

Detailed Steps to Completion:

Step	Description	Target Completion Date
Backend Setup	Connect Flask to FraudLabs Pro API and parse XML	**Mar 27**
Logging Enabled	Ensure transactions appear in dashboard via `POST` and `action=CHECK`	**April 4**
Frontend Form UI	Build dynamic React form for all required fields	**April 15**
Connect React ↔ Flask	Use Axios to post user data to Flask endpoint	**April 18**
Display Results	Show fraud score, status, recommendation in UI	**April 22**
UI Styling	Clean up with Tailwind CSS or basic CSS	**April 24**
Testing	Try different scenarios to test fraud scores	**April 25**
Write Report	Finalize written submission	**April 30**
Presentation Prep	Prepare slides/demo for in-class presentation	May 5
Final Presentation	Present working demo to class	May 7

What Else Can Be Added (Stretch Goals)

- Add authentication for user access
- Save transaction history in a local database (SQLite or MongoDB)
- Visualize fraud score trends over time
- Build a mobile-friendly version with React Native
- Add real-time alerts or risk notifications in the frontend
- 
Project Timeline Progress
Backend Setup
Logging Enabled
Frontend Form UI
Connect React ↔ Flask
Display Results
UI Styling
Testing
Write Report
Presentation Prep
Final Presentation
Done
Done
Pending
Pending
Pending
Pending
Pending
Pending
Pending
Pending
Incomplete
Complete
Completion Status