### **Project Overview: Credit Card Fraud Detection API**

The Credit Card Fraud Detection API is a web service that predicts whether a given credit card transaction is fraudulent or not. The API is built using Flask, a lightweight Python web framework, and it employs a pre-trained machine learning model to make predictions.

### **Key Components**

#### **1. Flask Application**

The core of the project is a Flask application. Flask is chosen for its simplicity and flexibility in building web services. The application defines an endpoint (/predict) that accepts transaction data and returns a prediction.

#### **2. Machine Learning Model**

The model used for prediction is pre-trained to identify fraudulent transactions. This model is saved in a file and loaded into the Flask application when it starts. The model uses features of the transactions (such as amount, location, time, etc.) to make its predictions.

#### **3. API Endpoint**

The main functionality of the API is provided through a single endpoint:

* **/predict**: This endpoint accepts POST requests with JSON payloads. The payload contains transaction features, which the model uses to predict if the transaction is fraudulent. The prediction is then returned as a JSON response.

#### **4. Requirements Management**

A requirements.txt file lists all the Python packages needed to run the application. This file ensures that the correct versions of the dependencies are installed, promoting consistency and reproducibility.

#### **5. Deployment on Render.com**

Render.com is used to host and deploy the Flask application. The process involves connecting a GitHub repository containing the project code to Render, configuring the web service, and deploying it. Render automatically builds the Docker image, runs the container, and provides a URL where the API is accessible.

### **Process Flow**

1. **User sends a request**: A client sends a POST request to the https://fraud-detection-3vxr.onrender.com endpoint with transaction data in JSON format.

**Code to be used in powershell to access the API**

$body = @{

features = @(133184,-1.212681701,-2.48482353,-6.397185815,3.670562448,-0.863375061,-1.855854731,1.01773158,-0.544703775,-1.70337805,-3.739659479,1.738124014,-2.844449336,0.765863961,-4.799737135,-0.011335412,-2.693168082,-3.166955158,-1.06780012,-0.559132202,2.908373946,1.396872063,0.092072872,-1.492882499,-0.204227396,0.532510949,-0.293871104,0.21266306,0.431094708,1335)

} | ConvertTo-Json

$headers = @{

"Content-Type" = "application/json"

}

$response = Invoke-RestMethod -Uri "https://fraud-detection-3vxr.onrender.com" -Method Post -Headers $headers -Body $body

$response

1. **Data Processing**: The Flask application extracts the transaction features from the request and formats them as required by the model.
2. **Prediction**: The model processes the features and predicts whether the transaction is fraudulent.
3. **Response**: The application sends back the prediction result in JSON format, indicating whether the transaction is likely to be fraudulent. ( If the result is 0, the transaction is legitimate. If it is 1, the transaction is fraudulent )

### **Benefits**

* **Scalability**: Using Docker allows the application to be easily scaled and deployed across different environments without worrying about inconsistencies.
* **Simplicity**: Flask provides a straightforward way to handle web requests and integrate with the machine learning model.
* **Reliability**: Hosting on Render.com ensures high availability and reliability, with automatic handling of deployments and scaling.

### **Conclusion**

The Credit Card Fraud Detection API is a robust and scalable solution for predicting fraudulent transactions. It leverages the simplicity of Flask, the power of pre-trained machine learning models, and the deployment ease provided by Docker and Render.com. This combination makes it an effective tool for real-time fraud detection in financial transactions.