**AI Engineer Assessment – Fintech Transaction Risk Intelligence System**

**Assessment Overview**

As an AI Engineer candidate, you are required to build a production-ready Artificial Intelligence solution that detects potentially fraudulent or high-risk financial transactions. This system must simulate a real-world financial intelligence engine capable of continuously adapting to evolving fraud patterns using obscured labels and noisy features.

The task reflects a real-life FinTech setting where fraudulent behavior evolves rapidly, and data patterns are not always easily separable. Your solution must include a trained AI model, an inference API, and a Streamlit-based interface to demonstrate how the model would be used in production.

**Assessment Objectives**

You are expected to design and implement an end-to-end AI solution with the following components:

**1. Risk Scoring Model**

Develop an AI model that predicts the probability of a transaction being fraudulent or anomalous. Use the label\_code column as your target variable. Note that this column has been intentionally obfuscated to simulate label ambiguity and real-world noise.

**2. Temporal Pattern Analysis**

Analyze whether fraud or risk signals correlate with temporal trends such as:

* Time of day (time\_of\_day)
* Day of the week (day\_of\_week)
* Timestamp patterns (transaction\_time)

Provide visual or analytical insights as part of your solution.

**3. Feature Engineering Strategy & Justification**

Perform and document advanced feature engineering including:

* Interaction features (e.g., amount × device\_type, is\_foreign\_transaction × location risk)
* Temporal transformations
* One-hot encoding, scaling, or embeddings (where applicable)

Explain your rationale for each transformation and interaction term used.

**4. Imbalanced Classification Strategy**

You are to conduct a thorough analysis of the target variable label\_code to determine whether the dataset exhibits class imbalance—a common characteristic in fraud detection tasks.

* **If the dataset is imbalanced:**  
  Apply specialized strategies to mitigate the risk of the model becoming biased toward the majority class. Consider using:
  + **Synthetic Oversampling** (e.g., SMOTE, ADASYN) to enrich the minority class.
  + **Undersampling** of the majority class while preserving predictive patterns.
  + **Cost-sensitive learning**, where higher penalties are assigned to misclassifying minority class instances.
  + **Anomaly detection approaches** as an alternative for identifying rare fraudulent transactions.
* **If the dataset is balanced:**  
  Proceed with standard classification techniques but still remain vigilant of other potential challenges such as:
  + **Label noise or ambiguity**, particularly since the label\_code has been intentionally obscured to simulate real-world uncertainty.
  + **Temporal leakage or data drift**, especially if patterns change over time.
  + **Overfitting**, especially if the model is overly complex or tuned on a static dataset without proper validation.

Regardless of the balance status, ensure that model evaluation goes beyond accuracy. Emphasize metrics that reflect the business priorities of fraud detection—**precision, recall, F1-score, and AUC-PR**—and discuss any trade-offs observed.

**5. Unsupervised Transaction Clustering (Bonus)**

Apply unsupervised clustering (e.g., K-Means, DBSCAN, Gaussian Mixture) to segment users or transactions into behavioral clusters like:

* High-frequency low-value users
* Foreign frequent travelers
* Infrequent but large-amount users  
  Discuss the potential business use case of each cluster.

**6. Model Explainability & Trade-offs**

Use interpretability tools like SHAP or LIME to:

* Visualize key feature contributions per transaction
* Discuss trade-offs between precision vs. recall
* Outline potential business impact of false positives and false negatives

**7. Deployment & Production Simulation**

You must simulate the deployment of your AI system using the following two channels:

**a. Streamlit Dashboard**

Create a user-friendly dashboard to:

* Input a new transaction
* Display fraud risk score
* Visualize key features and predictions
* Show daily/weekly fraud trend analysis

**b. REST API Endpoint**

Build and expose a REST API (e.g., using FastAPI or Flask) that:

* Accepts transaction JSON payloads
* Returns fraud probability and explanation

**Deliverables**

The following items must be completed and submitted:

| **Component** | **Description** |
| --- | --- |
| risk\_model.ipynb/.py | Full AI model implementation with EDA, training, and evaluation |
| streamlit\_app.py | Streamlit dashboard that simulates real-time fraud scoring |
| api\_server.py (FastAPI/Flask) | REST API for model serving with fraud scoring and explanations |
| utils/ | Any feature engineering or transformation utilities |
| models/ | Serialized model files (.pkl, .joblib, etc.) |
| README.md | Detailed project documentation (see format below) |
| requirements.txt | All dependencies required to run the app and API |
| GitHub Repository | Push your solution to a public GitHub repo and include the link |

**README File – Required Structure**

Your README.md should clearly describe the following:

* **Project Overview**
* **Data Understanding and Preprocessing**
* **Feature Engineering Approach**
* **Model Selection and Training Strategy**
* **Temporal Analysis Results**
* **Imbalance Handling Strategy**
* **Explainability Insights**
* **Deployment Instructions**
  + How to run the Streamlit app
  + How to start the API
* **Assumptions and Limitations**
* **Future Improvements**

**Tools and Frameworks Allowed**

You may use any of the tools listed below:

* **Programming Language:** Python
* **Modeling Libraries:** Scikit-learn, XGBoost, LightGBM, CatBoost, PyCaret (optional)
* **Explainability Tools:** SHAP, LIME
* **Visualization:** Matplotlib, Seaborn, Plotly
* **Dashboarding:** Streamlit (mandatory)
* **API Serving:** FastAPI or Flask (mandatory)
* **Development Environment:** Jupyter Notebook or .py scripts

**Dataset**

Download the dataset from the link below:  
fintech\_sample\_fintech\_transactions.csv  
(This dataset simulates real-world anonymized FinTech transactions.)

**Submission Guidelines**

* **Format:** Zip file or GitHub repository
* **Repository Name:** fintech-fraud-ai-<YourName>
* **Content:** All source code, models, documentation, and setup files
* **Delivery:** Share the GitHub repository link with the assessment panel

**Evaluation Criteria**

| **Criteria** | **Weight** |
| --- | --- |
| End-to-End System Functionality | 30% |
| Model Accuracy and Robustness | 20% |
| Code Structure and Modularity | 15% |
| Streamlit Dashboard Design | 10% |
| API Design and Response Quality | 10% |
| Documentation and Explanation Clarity | 10% |
| Creativity in Feature Engineering | 5% |

**Additional Notes**

* The solution must be executable and complete.
* Originality and integration skills will be assessed.
* Avoid hardcoding and build reusable components wherever possible.