+-----------------+  
| Data Sources |  
+--------+--------+  
 | (CSV Data,  
 | Real-time Stream - Optional)  
 v  
+--------+--------+  
| Data Ingestion |  
| (data\_ingestion.py)|  
+--------+--------+  
 | (Raw Data)  
 v  
+--------+--------+  
| Feature |  
| Engineering |  
| (feature\_ |  
| engineering.py)|  
+--------+--------+  
 | (Features)  
 v  
+--------+--------+  
| Data Storage |  
| (Feature Store/|  
| Database) |  
+--------+--------+  
 | (Stored Features)  
 v  
+--------+--------+  
| Model |  
| Training |  
| (training.py) |  
+--------+--------+  
 | (Model, Metrics)  
 v  
+--------+--------+  
| Trained Model |  
| & Data |  
| (models/) |  
+--------+--------+  
 | (Model for Inference)  
 v  
+--------+--------+  
| REST API |  
| (Flask) |  
| (api.py) |  
+--------+--------+  
 | (Transaction Data)  
 v  
+--------+--------+  
| Feature |  
| Engineering |  
| (feature\_ |  
| engineering.py)|  
+--------+--------+  
 | (Features)  
 v  
+--------+--------+  
| Model |  
| Inference |  
| (inference.py)|  
+--------+--------+  
 | (Prediction, Score)  
 v  
+--------+--------+  
| Prediction & |  
| Explanation |  
| (inference.py)|  
+--------+--------+  
 | (SHAP Values)  
 v  
+--------+--------+  
| SHAP Explainer|  
| (inference.py)|  
+--------+--------+

**Component Breakdown:**

1. **Data Sources:**
   * **Description:** Where the transaction data originates.
   * **Contents:**
     + CSV File (creditcard.csv): Historical transaction data for training.
     + Real-time Stream (Kafka/Kinesis): Optional, for streaming new transactions.
2. **Data Ingestion (data\_ingestion.py):**
   * **Description:** Loads data from the sources, performs basic cleaning, and splits data into training and testing sets.
   * **File:** data\_ingestion.py
3. **Feature Engineering (feature\_engineering.py):**
   * **Description:** Transforms raw data into features for the model. Creates features like Hour, Amount\_relative\_to\_mean, etc. Handles training and inference modes to prevent data leakage.
   * **File:** feature\_engineering.py
4. **Data Storage (Feature Store/Database):**
   * **Description:** Stores feature-engineered data. Could be a relational database (PostgreSQL, MySQL) or a specialized feature store (Feast).
5. **Model Training (training.py):**
   * **Description:** Trains the fraud detection model on the feature-engineered data. Evaluates the model using metrics like precision, recall, F1-score, and Average Precision.
   * **File:** training.py
6. **Trained Model & Data (models/):**
   * **Description:** Stores the trained model file (e.g., fraud\_detection\_model.joblib) and other necessary data (e.g., mean\_amount.joblib, mean\_amount\_by\_hour.joblib).
   * **Directory:** models/
7. **REST API (Flask) (api.py):**
   * **Description:** A Flask web application that provides a REST API endpoint (/predict) for real-time fraud prediction.
   * **File:** api.py
8. **Feature Engineering (feature\_engineering.py):**
   * **Description:** (Same as step 3) Applies the same feature engineering logic to new transactions received by the API.
9. **Model Inference (inference.py):**
   * **Description:** Loads the trained model and makes predictions on new, feature-engineered transaction data.
   * **File:** inference.py
10. **Prediction & Explanation (inference.py):**
    * **Description:** Generates the prediction (fraud/not fraud), the anomaly score, and uses the SHAP explainer to provide feature contributions.
11. **SHAP Explainer (inference.py):**
    * **Description:** Calculates and returns SHAP values to explain individual predictions.

**Data Flow:**

* **Training:** Data Sources -> Data Ingestion -> Feature Engineering -> Data Storage -> Model Training -> Trained Model & Data
* **Inference:** REST API -> Feature Engineering -> Model Inference -> Prediction & Explanation -> SHAP Explainer