```
Import numpy as np
Import pandas as pd
From sklearn.model_selection import train_test_split
From sklearn.ensemble import RandomForestClassifier
From sklearn.metrics import classification_report, confusion_matrix

# 1. Simulate Transaction Data
Def generate_transaction(account_id, is_fraud=False):
    Amount = np.random.uniform(10, 500)
    Transaction_time = pd.to_datetime('now')
    Location = np.random.choice(['Online', 'Physical Store'])
    Ip_address = f"192.168.1.{np.random.randint(1, 255)}" if location == 'Online' else None
    Device = np.random.choice(['Mobile', 'Desktop', 'Tablet']) if location == 'Online' else None
    Transaction = {
        'account_id': account_id,
        'amount': amount,
        'transaction_time': transaction_time,
        'location': location,
        'ip_address': ip_address,
        'device': device,
        'is_fraud': is_fraud
    }
    Return transaction

Def generate_synthetic_data(num_samples=1000, fraud_ratio=0.05):
```

```
    Data = []
    Num_fraud = int(num_samples * fraud_ratio)
    Num_genuine = num_samples – num_fraud
    For i in range(num_genuine):
        Data.append(generate_transaction(f”user_{i}”))
    For i in range(num_fraud):
        Data.append(generate_transaction(f”fraudster_{i}”, is_fraud=True))
    Return pd.DataFrame(data)

# 2. Load and Preprocess Data
Data = generate_synthetic_data(num_samples=2000, fraud_ratio=0.03)

# Simple Feature Engineering
Data[‘transaction_hour’] = data[‘transaction_time’].dt.hour
Data[‘is_online’] = data[‘location’].apply(lambda x: 1 if x == ‘Online’ else 0)
Data = pd.get_dummies(data, columns=[‘location’, ‘device’], drop_first=True)
Data = data.drop([‘account_id’, ‘transaction_time’, ‘ip_address’], axis=1, errors=’ignore’)

# Separate features (X) and target (y)
X = data.drop(‘is_fraud’, axis=1)
Y = data[‘is_fraud’]

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 3. Train a Fraud Detection Model (Random Forest as an example)
```

```
Model = RandomForestClassifier(random_state=42)
Model.fit(X_train, y_train)

# 4. Fraud Scoring and Prediction
Def predict_fraud(transaction_data, model):
    Transaction_df = pd.DataFrame([transaction_data])
    # Preprocess the new transaction data consistently with the training data
    Transaction_df[‘transaction_hour’] = transaction_df[‘transaction_time’].dt.hour
    Transaction_df[‘is_online’] = transaction_df[‘location’].apply(lambda x: 1 if x == ‘Online’ else 0)
    Transaction_df = pd.get_dummies(transaction_df, columns=[‘location’, ‘device’], drop_first=True)
    Transaction_df = transaction_df.drop([‘account_id’, ‘transaction_time’, ‘ip_address’], axis=1, errors=’ignore’)

    # Ensure the new data has the same columns as the training data
    Missing_cols = set(X_train.columns) – set(transaction_df.columns)
    For c in missing_cols:
        Transaction_df[c] = 0
    Transaction_df = transaction_df[X_train.columns] # Ensure correct column order

    Probability = model.predict_proba(transaction_df)[:, 1]
    Prediction = model.predict(transaction_df)[0]
    Return probability[0], prediction

# 5. Transaction Guarding Function
Def guard_transaction(transaction):
```

```
    Fraud_probability, is_fraud = predict_fraud(transaction, model)
    Print(f”Transaction Details: {transaction}”)
    Print(f”Fraud Probability: {fraud_probability:.4f}”)
    If is_fraud == 1:
        Print(“Potential Fraud Detected! Transaction blocked.”)
        Return “Blocked”
    Else:
        Print(“Transaction Approved.”)
        Return “Approved”

# 6. Evaluate the Model
Y_pred = model.predict(X_test)
Print(“\n--- Model Evaluation ---“)
Print(“Confusion Matrix:\n”, confusion_matrix(y_test, y_pred))
Print(“\nClassification Report:\n”, classification_report(y_test, y_pred))

# 7. Simulate a New Transaction and Guard It
New_transaction = generate_transaction(“new_user”, is_fraud=False)
Guard_transaction(new_transaction)

Suspicious_transaction = {
    ‘account_id’: ‘suspicious_user’,
    ‘amount’: 450,
    ‘transaction_time’: pd.to_datetime(‘now’),
    ‘location’: ‘Online’,
    ‘ip_address’: ’10.0.0.1’,
```

```
    ‘device’: ‘Unknown’
}
Guard_transaction(suspicious_transaction)
```