Bank Transaction Risk & Anomaly Analyzer

Course Project Specifications
(Console Application + OOP Design)

1. Project Objective

Develop a Python-based console application that analyzes banking transaction data to detect risky customers and anomalous transactions using data analysis and statistical techniques.

2. Mandatory Libraries

- Pandas: Data loading, cleaning, aggregation, and feature engineering
- NumPy: Numerical operations, vectorization, and statistical calculations
- SciPy: Statistical analysis and anomaly detection (z-score, distributions)

3. Input Data

• Transaction dataset (CSV) obtained from Kaggle
• Optional customer or profile dataset (CSV)

Recommended Kaggle Datasets

1. PaySim – Mobile Money Transactions
   https://www.kaggle.com/datasets/ealaxi/paysim1

2. Credit Card Fraud Detection
   https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

4. Application Type (Console Application)

The project must be implemented as a console-based application (CLI). Users interact with the system through a menu-driven interface.

Minimum required menu options:
1. Load dataset(s)
2. Clean and validate data
3. Build features

4. Score customers
5. Flag suspicious transactions
6. Export reports
7. Display summary in console
0. Exit application

5. Object-Oriented Design Requirements

The application must follow Object-Oriented Programming principles. The code should be modular, reusable, and easy to maintain.

Required classes:
- DataManager: Handles file loading and validation
- TransactionCleaner: Cleans and prepares data
- FeatureBuilder: Builds customer-level features
- RiskScorer: Computes risk scores using SciPy
- TransactionFlagger: Flags suspicious transactions
- ReportGenerator: Generates CSV and text reports
- ConsoleApp: Manages menu and application flow

6. Core Functional Requirements

Data Cleaning:
• Handle missing and invalid values
• Convert timestamps
• Remove duplicates

Feature Engineering:
• Transaction count per customer
• Total, average, and maximum transaction amounts
• Daily transaction velocity
• Rolling statistics

Risk & Anomaly Detection:
• Z-score based detection
• Risk band classification (Low / Medium / High / Critical)

7. Outputs

The application must generate the following files:
- flagged_transactions.csv
- customer_risk_summary.csv
- report.txt summarizing top risky customers and findings

8. Student Deliverables

- Source code organized as a Python package
- Console application entry point (main.py)
- Generated reports
- README explaining methodology and results

9. Deadline

- 28 Dec. 2025