**📑 Project Report: Personal Finance Dashboard with AI Insights**

**1. 🎯 Project Goal**

We built an interactive **Personal Finance Dashboard** using **Streamlit + AI/ML**.  
The app helps users:

* Track their spending over time
* Visualize expenses by category
* Discover typical spending patterns using clustering (KMeans)
* Detect unusual transactions using anomaly detection (IsolationForest)
* Generate simple AI-powered **insights** like “Your food spending increased by 20% last month”

This project is a great exercise in **data science + machine learning + app development**.

**2. 📂 Project Structure**

We organized the project into a **clean folder structure**:

Personal-Finance-Dashboard-with-AI-Insights-1/

│

├── data/ # Sample and user data

│ └── transactions.csv # Generated dataset

│

├── src/ # Source code

│ ├── app.py # Main Streamlit dashboard

│ ├── preprocess.py # Data cleaning + feature engineering

│ ├── ai\_models.py # Clustering + anomaly detection

│ ├── analysis.py # Trend analysis + insights

│ └── generate\_mock.py # Script to generate fake transactions

│

├── requirements.txt # Dependencies

└── README.md # Documentation

**3. 🛠️ Components Explained**

**(a) Data: transactions.csv**

* Each row is a financial transaction.
* Columns:
  + date: when the expense happened
  + category: type of expense (Food, Transport, Shopping, etc.)
  + amount: money spent
  + description: short text label

This dataset can come from:

* A real CSV upload by the user
* A generated mock dataset (generate\_mock.py)

**(b) Preprocessing: preprocess.py**

Purpose: clean and prepare the raw data so ML models can use it.  
Steps:

1. **Standardize categories** (fill missing values, make them consistent).
2. **Feature engineering**:
   * Add amount\_log (log-transformed amount, helpful for ML).
   * Add amount\_bucket (categorical bucket: small, medium, large, very large).

This makes the data more useful for both visualization and ML models.

**(c) Machine Learning Models: ai\_models.py**

Two key AI features:

1. **Clustering with KMeans**
   * Groups transactions into “spending clusters” (e.g., daily small expenses, big bills, shopping sprees).
   * Helps understand *types* of spending behavior.
2. **Anomaly detection with IsolationForest**
   * Finds unusual transactions (e.g., abnormally large purchases).
   * Useful for spotting mistakes, fraud, or out-of-pattern spending.

**(d) Analysis: analysis.py**

This module extracts meaningful patterns:

* **Monthly spending trend**: sum of expenses per month.
* **Spending by category per month**: breakdown by type.
* **AI insights**:
  + Compare last month vs previous month.
  + Find the top spending category.
  + Warn if anomalies are detected.

**(e) Dashboard App: app.py**

The heart of the project, built with **Streamlit**.

Features:

1. **Sidebar controls**
   * Upload your own CSV or use sample data
   * Select date range filter
   * Adjust clustering (number of groups)
   * Adjust anomaly detection sensitivity
2. **KPIs (Key Performance Indicators)**
   * Total spending last month
   * % change compared to previous month
   * Total number of transactions
3. **Visualizations**
   * Line chart: monthly spending trend
   * Bar chart: category breakdown
   * Cluster summaries (group counts, sample transactions)
   * Anomaly table (flagged unusual transactions)
4. **AI Insights**
   * Human-readable insights, like “⚠️ Detected 3 unusual transactions”
5. **Full transaction table**
   * Expandable view with all filtered transactions + cluster labels.

**(f) Data Generator: generate\_mock.py**

Because we didn’t have real financial data, we created a script that:

* Generates **300 fake transactions** across categories
* Randomly assigns dates, amounts, and categories
* Saves them into data/transactions.csv

This ensures the dashboard always has a sample dataset to load.

**4. ⚙️ Workflow**

Here’s what happens step by step when you run:

streamlit run src/app.py

1. **Load Data**
   * If you uploaded a CSV → load it.
   * Else, load data/transactions.csv.
2. **Preprocess Data**
   * Clean categories, add engineered features.
3. **Filter Data**
   * Apply date range filter (from sidebar).
4. **Run Models**
   * KMeans → cluster transactions.
   * IsolationForest → detect anomalies.
5. **Compute Analysis**
   * Monthly totals, category totals, insights.
6. **Visualize in Streamlit**
   * Show KPIs, charts, clusters, anomalies, insights, full table.

**5. 📊 Example Outputs**

* **KPI:** “Total (last month): 1200.50”
* **Insight:** “You spent 15.3% more last month compared to the month before.”
* **Insight:** “Your top spending category is **Food**.”
* **Anomaly Table:** Lists unusual transactions with scores.
* **Cluster Summary:** Groups typical expenses like:
  + Cluster 0 → many small food purchases
  + Cluster 1 → medium shopping amounts
  + Cluster 2 → rare very large bills

**6. 💡 Skills Learned**

By building this project, you practiced:

* **Python data handling** with pandas
* **Feature engineering**
* **Machine learning basics** (clustering, anomaly detection)
* **Data visualization** with Plotly
* **Building web apps** with Streamlit
* **Project structuring** (clean folders, modular code)

These are exactly the skills useful for AI + Digital Health later — only the data will be medical instead of financial.

✅ That’s the full explanation of what we built and how it works.