Project Title

- **Personal Budgeting & Expense Forecaster** (Phase 3: Forecasting Engine and Goal Setting)

Introduction

- This phase is the most important part of the project. It moves the tool from just tracking past spending to actively **predicting the financial future**.
- This module implements the continuous revision of the financial roadmap, which is essential because projects without accurate forecasting often experience major problems.
- The goal is to provide users with predictive intelligence so they can actively control costs and reduce financial risks.

Objectives

- **Prepare Data:** Get the organized transaction data from Milestone 2 ready for the prediction model.
- **Integrate Prediction Model:** Fully connect the Prophet library to generate future forecasts for both specific categories and overall spending.
- **Enable Goal Tracking:** Build the system that lets users define, track, and measure their financial goals (like saving targets or spending reduction).
- **Visualize the Future:** Update the main screen to clearly show the predicted future expenses and savings alongside the historical data.

System Overview

- This phase connects the cleaned historical data to the advanced prediction engine.
- **Step 1: Data Aggregation:** The cleaned data (from M2) is summed up (aggregated) into time series (daily, weekly, or monthly totals).
- **Step 2: Prophet Execution:** The prepared time series data is fed into the Prophet model.
- **Step 3: Prediction & Goal Check:** Prophet creates the forecast, and the system compares these predictions against the user's defined goals.
- **Step 4: Dashboard Update:** The user's screen is updated with the new forecast charts and a status update on their goals.

Methodology / Workflow

- **Prediction Tool (Prophet):** We use the Prophet forecasting method because it is powerful and handles "messy" real-life financial data well.
- **Time Series Preparation:** The model requires historical data to be organized in a time series format, where all transactions are grouped (aggregated) by time unit (e.g., all grocery expenses summed up for each week).
- **Additive Model:** Prophet breaks the complex spending patterns into simple parts (like trend and yearly/weekly seasonality) to create an accurate prediction.
- **Risk Display:** The output will include the single best prediction, plus an upper and lower uncertainty range. This range is critical for budgeting responsibly and planning for potential worst-case scenarios.
- **Visualization Tool:** Seaborn and Matplotlib will be used to draw the line charts, ensuring the visual output is clear and professional.

System Design

- **Data Preparation Format:** The system must ensure the data is in the required two-column format: ds (datestamp) and y (the aggregated money value).
- **Goal Setting Logic:** The system needs logic to track simple, target-based goals, such as calculating if the forecast expense line is below the goal-setting expense line.
- **Forecast Display Design:** The main chart must show three elements together in a single

line chart for easy comparison:
    - The **Actual** historical spending line.
    - The **Forecasted** future expense line.
    - The **Projected Savings** line, based on the difference between income forecast and expense forecast.

Implementation

A. Environment Preparation (Setup)

- Ensure the Python environment has the necessary libraries installed from previous phases.
- The Prophet forecasting library must be installed using the command pip install prophet.
- The visualization libraries, Matplotlib and Seaborn, must be ready for drawing line charts.

B. Development Milestone: (Module 3: Forecasting Engine & Goal Setting)

- **Historical Data Preparation:** Implement the code to take transaction data and turn it into aggregated time series data (e.g., total monthly spending for "Transport").
- **Prophet Integration:** Integrate the Prophet library and run the training process on the prepared historical data to generate future expense forecasts for a defined period.
- **Financial Goal Setting:** Build the interface and logic to let users input specific goals, such as setting a savings amount or specifying a percentage reduction in a category.
- **Forecast Visualization:** Update the dashboard. Use Matplotlib/Seaborn to generate a line chart that overlays historical data with the new forecasted expenses. Display a calculation of expected future savings based on the prediction.

Testing & Results

- **Prediction Accuracy Testing:** Run the Prophet model and compare its predictions to the actual results after a short period. This checks the model's performance.
- **Goal Tracking Testing:** Test the goal-setting feature to ensure the system correctly calculates and reports whether the user is on track to meet their savings or reduction goals.
- **Visualization Testing:** Verify that the forecast line chart is clear and accurately includes the historical data and the projected savings calculation. Check that the chart is "attractive and informative.

Conclusion

- Milestone 3 delivers the core value proposition of the project: predictive power.
- By integrating the robust Prophet model, users can now look into the future, set actionable goals, and proactively adjust their financial plans, fulfilling the purpose of continuous financial revision.

Future Scope

- The system can be enhanced by allowing users to compare their forecasts against external benchmarks or industry-specific templates.
- Long-term plans include connecting the system to real-time bank data (via API) to automatically update the forecast instantly, eliminating manual data handling.