16/06/2025

Savvy

AI - Powered Personal Finance Management Application

Graduation Project II - A.Y. 2024-2025

Prepared By:

Basmala Salama Hassan

Elham Hamed Mahmoud

Irinie Magued Sabry

Menna Allah Samy

Menna Allah Saeed

Supervised By:

Dr. Reem Essameldin

SAVVY

Savvy

AI - Powered Personal Finance Management Application

SAVVY

Overview

- **Key Features Overview**
  - Automated Transaction Tracking
  - Budget Planning
  - Goal Management
  - Chatbot & Spending Forecast (AI Features)
- **Tech Stack:**
  - Frontend: Flutter
  - Backend: FastAPI + Supabase
  - ML Models: LSTM & LangChain RAG &Transactions chatbot

- **Project Objectives**
  - Help users track income and expenses
  - Provide smart financial insights using AI
  - Enable personalized goal setting and budgeting
  - Predict future spending trends

Savvy

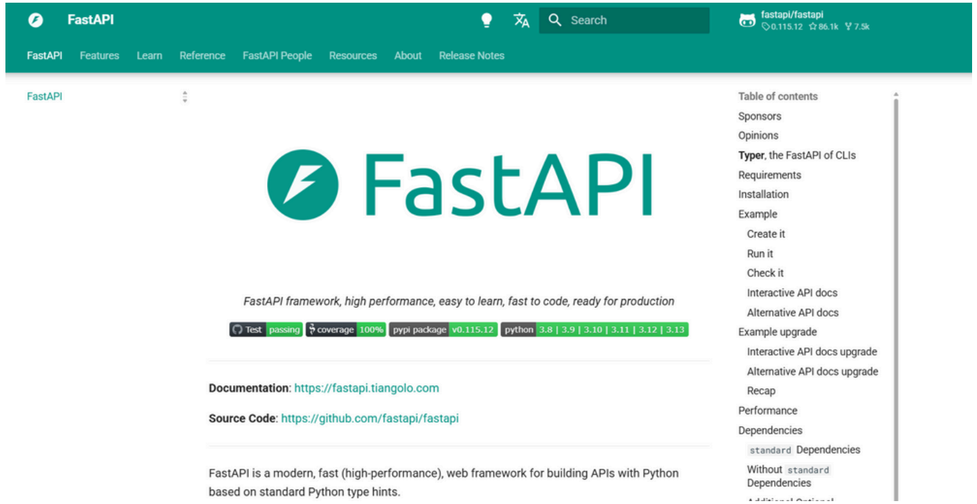
Backend Development

Backend Development

FastAPI

FastAPI is a modern web framework that is relatively fast and used for building APIs with Python

- High performance using asynchronous Python
- Automatic documentation with Swagger & ReDoc
- Type-safety with Pydantic
- Fast development cycle

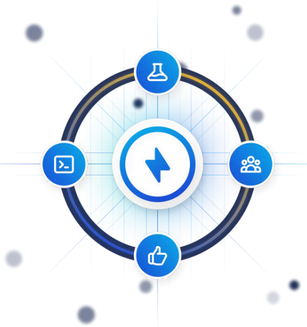


What is Supabase and Why We Chose It?

Supabase is an open-source Backend-as-a-Service (BaaS) platform that provides a full backend solution powered by PostgreSQL — the world's most advanced open-source relational database.

Row-Level Security (RLS)

nsures secure, per-user data access using Row-Level Security—so users only see their own data.



Real-Time Updates

Enables dynamic updates to transactions and budgets without page refreshes, improving UX.

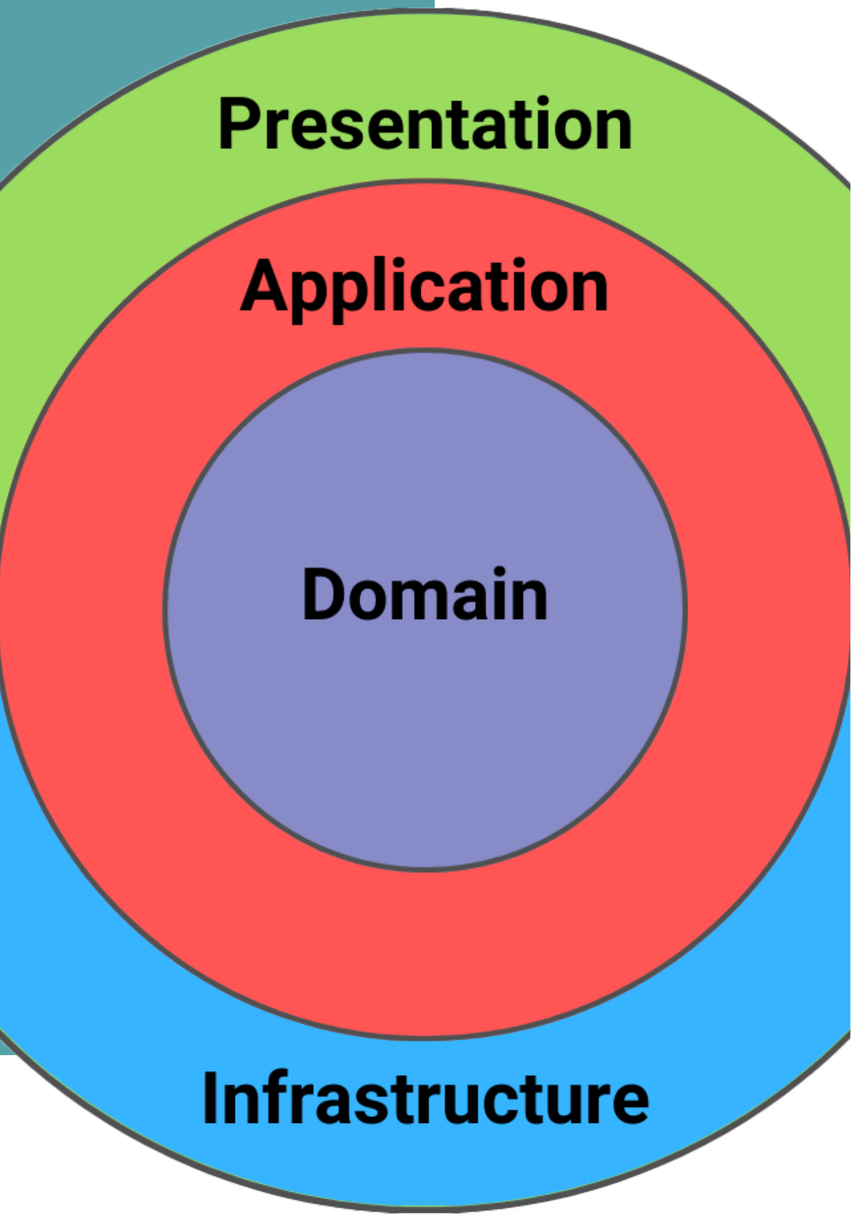
SQL Flexibility

Advanced querying, full-text search, and time-series support helped us build powerful analytics and prediction features.

User Authentication & Authorization

- Fully integrated sign-up / login / session handling
- JWT-based authentication with Row-Level Security (RLS) to enforce per-user access to data.

Clean Architecture



Clean Architecture is a software design approach that emphasizes long-term modularity, and
testability by enforcing a strict separation of concerns, particularly **between core business logic and external systems**

Presentation Layer
(API endpoints)

Service Layer
(Business logic)

Infrastructure Layer
(Data access)

Domain Layer
(Entities, Models)

Repository Design Pattern

A structural design pattern that

- Abstracts how data is fetched, saved, updated, or deleted.
- Acts like a controlled in-memory collection for domain entities.
- Hides the details of data persistence from the application logic.
- Promotes modular, testable, and maintainable code.

What app does VS How data stored

Frontend

Flutter

Flutter

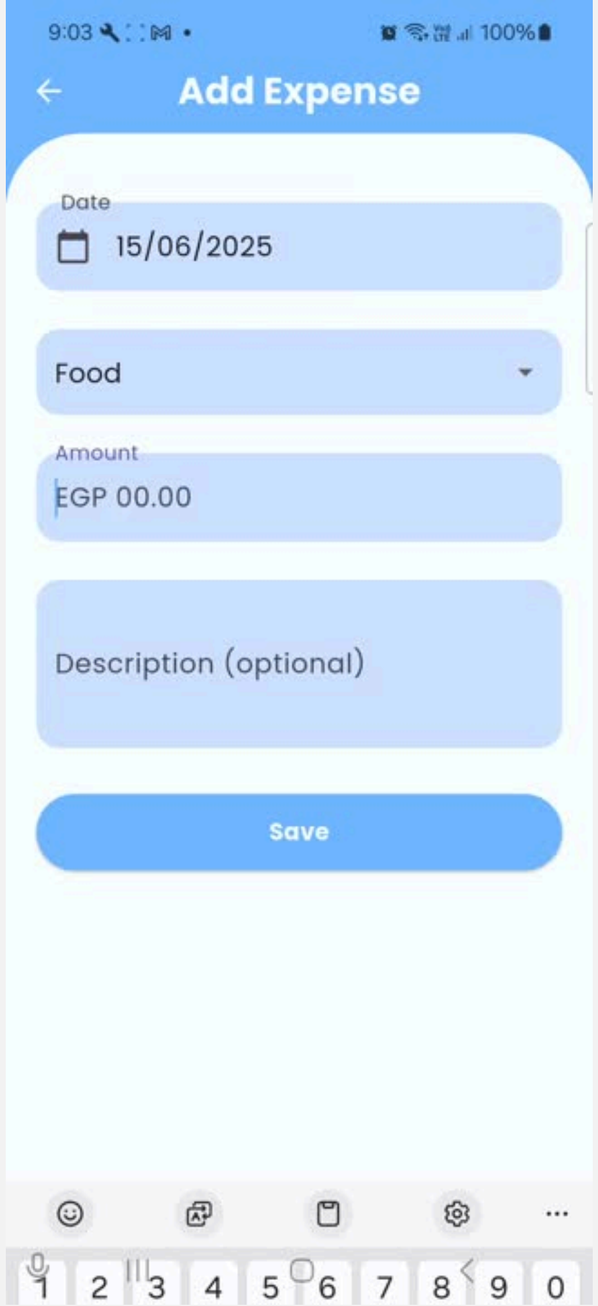
App Development

Google

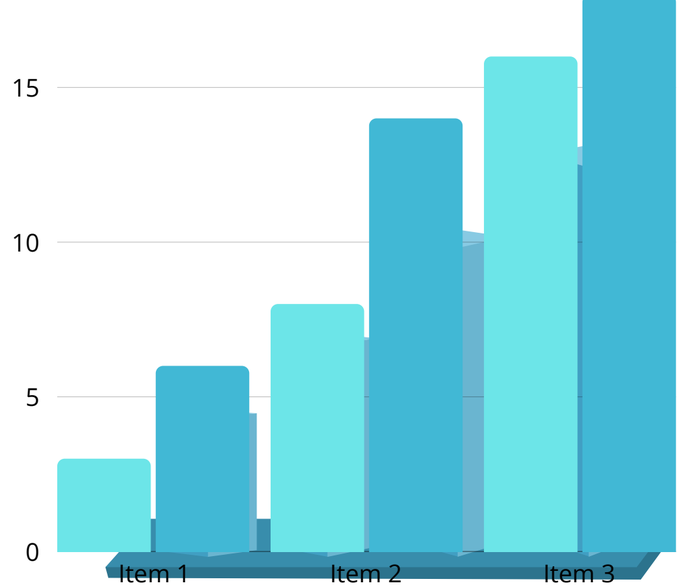
Where is my money??!

When will I reach my goal?

Demo Time

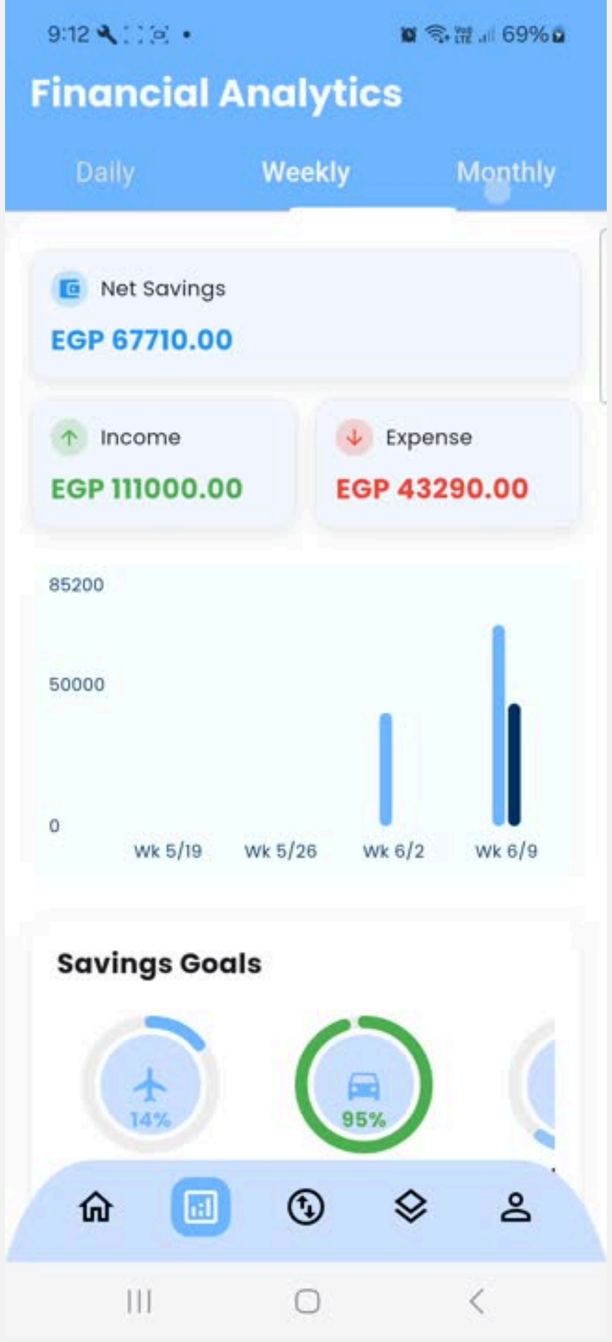

9:03
100%
Add Expense
Date
15/06/2025
Food
Amount
EGP 00.00
Description (optional)
Save
1 2 3 4 5 6 7 8 9 0

Analytics

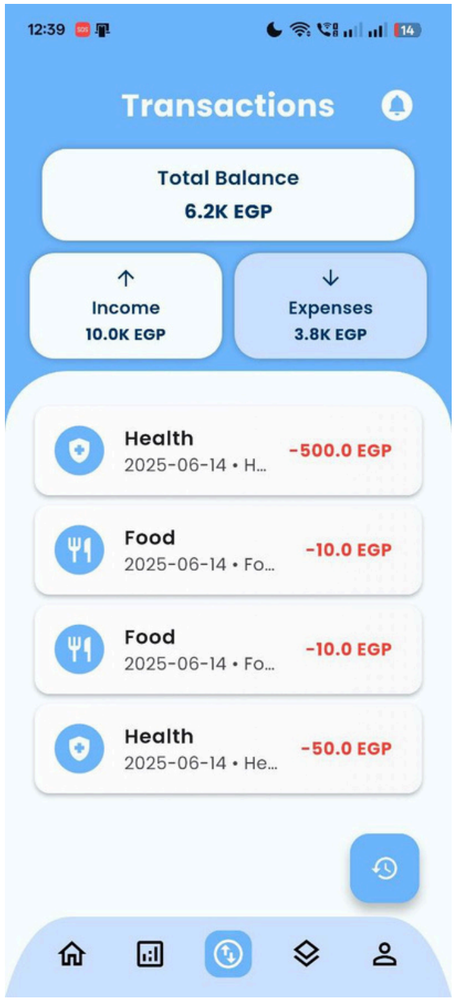


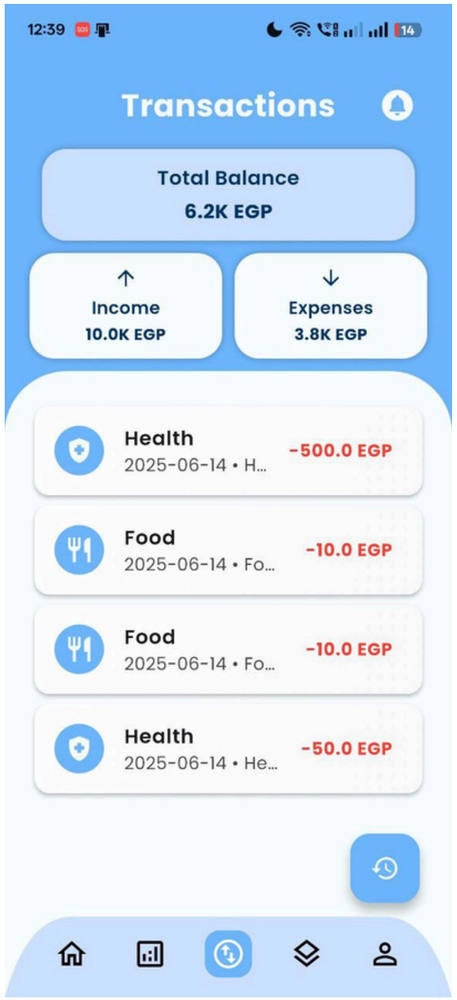
fl_chart

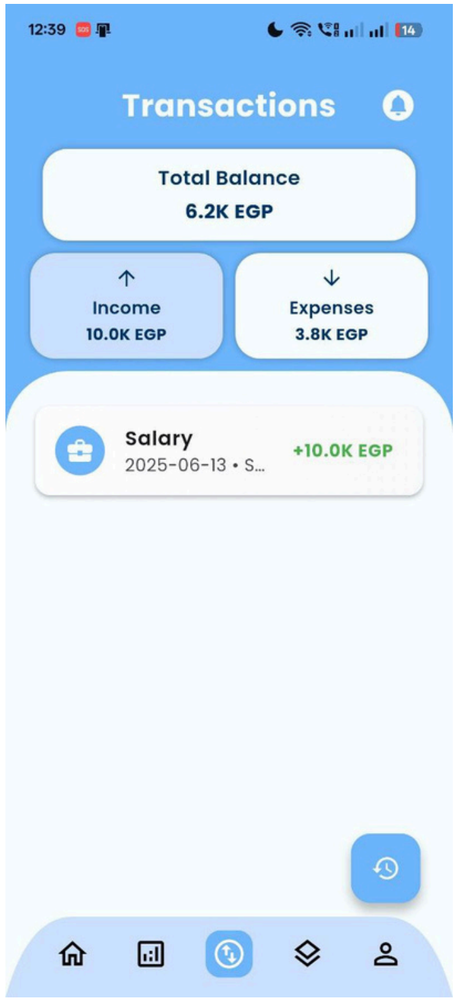
We made sure the analytics update in real time by using **state management with Riverpod**. Any time the user adds a transaction or saves money toward a goal, our providers automatically recalculate the totals and trigger a UI update.


9:12
69%
Financial Analytics
Daily
Weekly
Monthly
Net Savings
EGP 67710.00
Income
EGP 111000.00
Expense
EGP 43290.00
85200
50000
0
Wk 5/19
Wk 5/26
Wk 6/2
Wk 6/9
Savings Goals
14%
95%

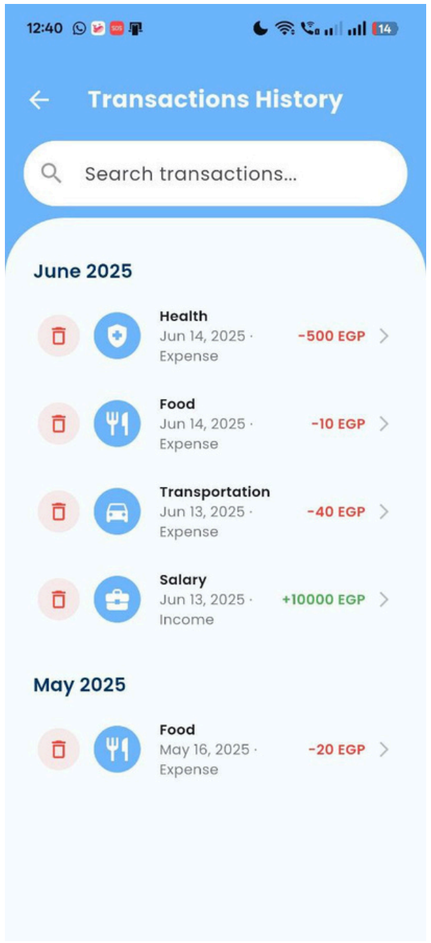
Transactions

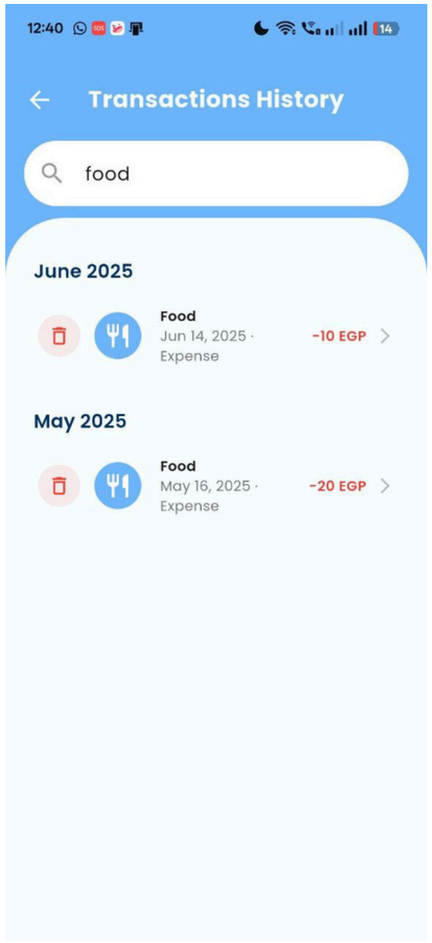


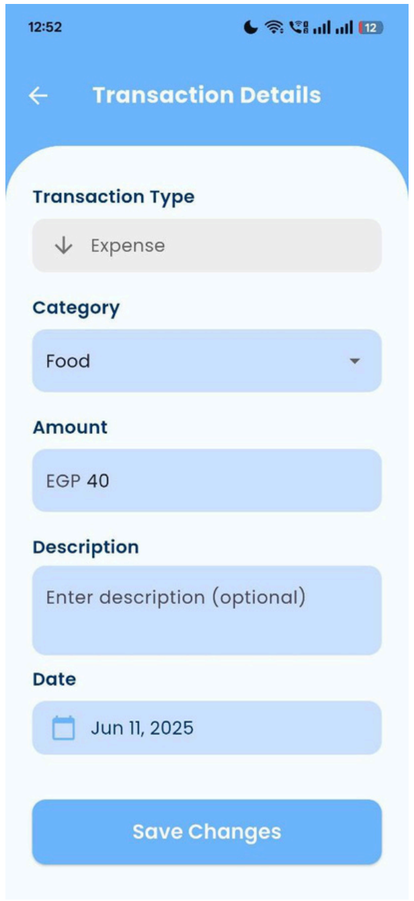


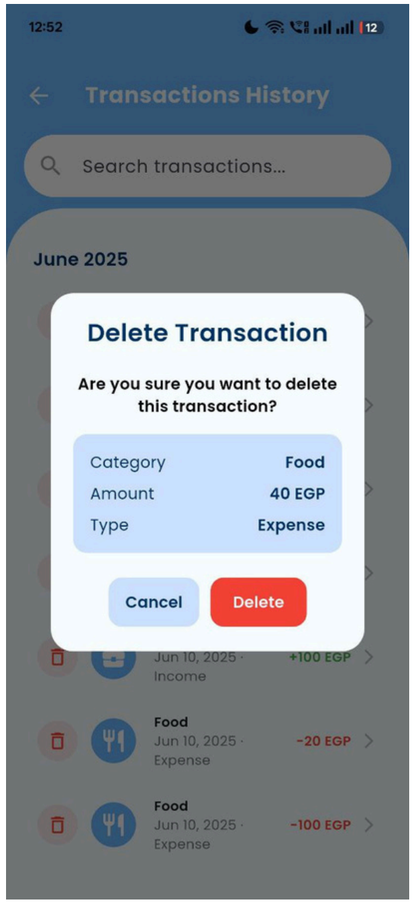


Transactions history

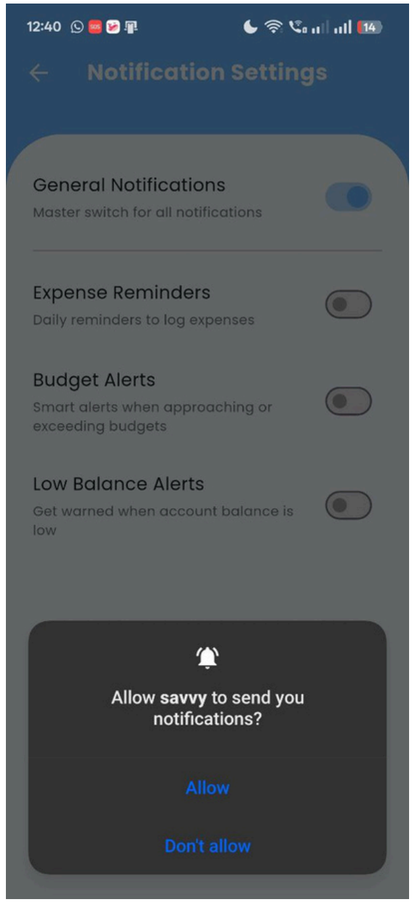


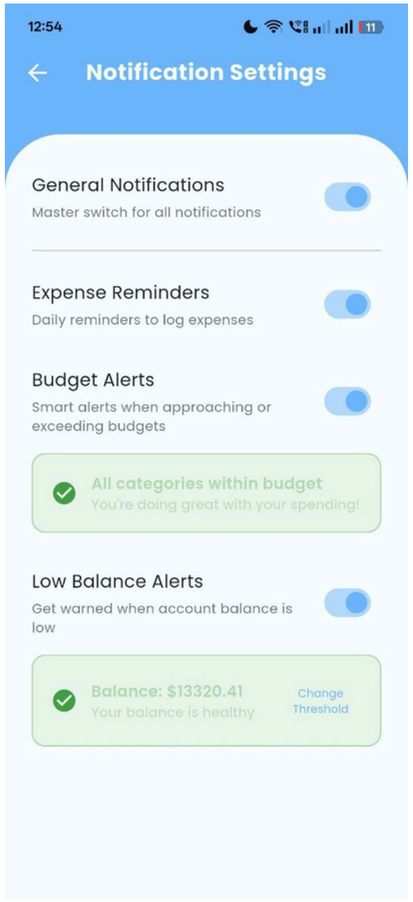


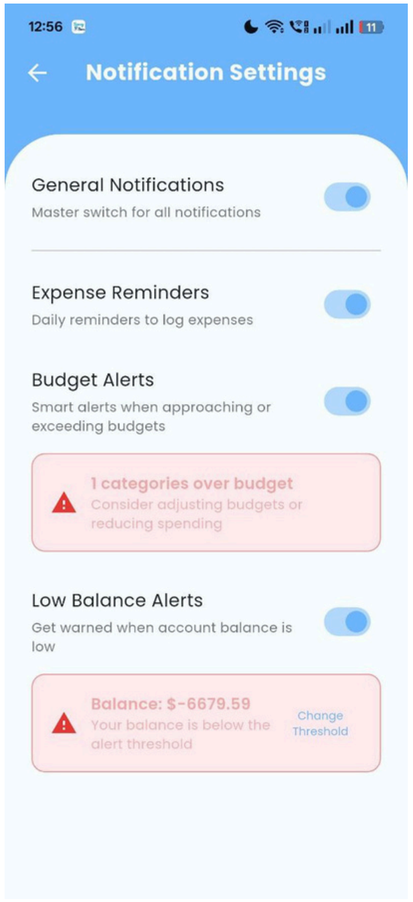


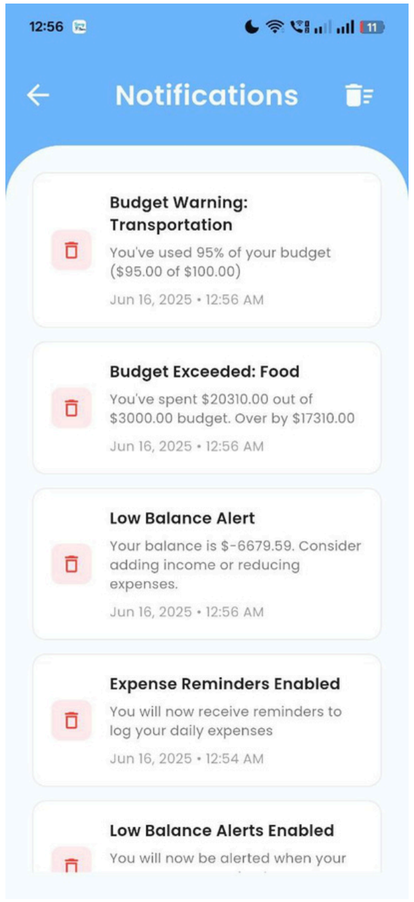


Notifications

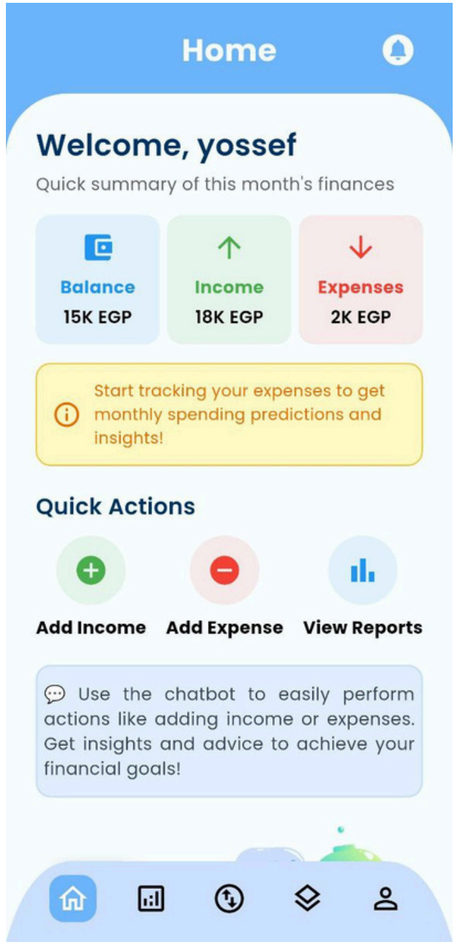


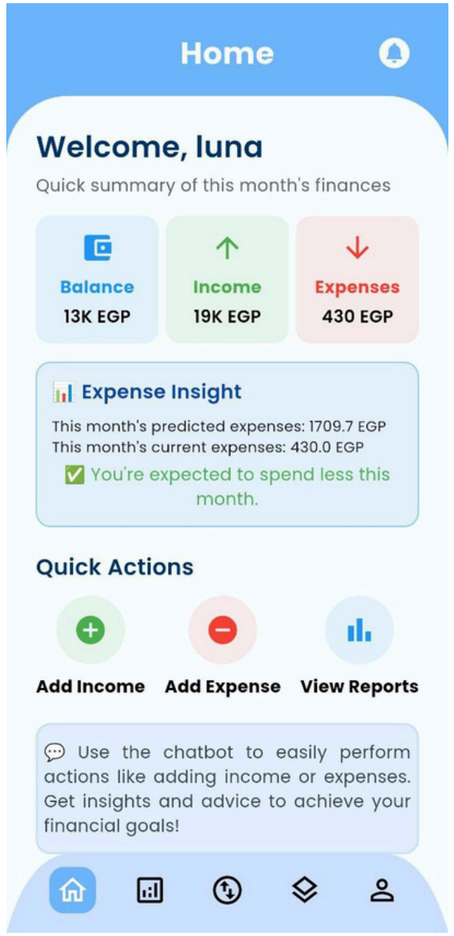


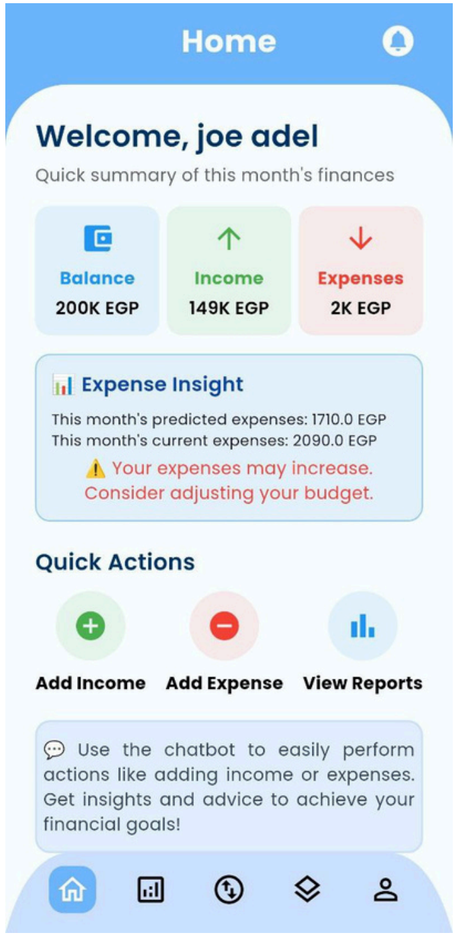




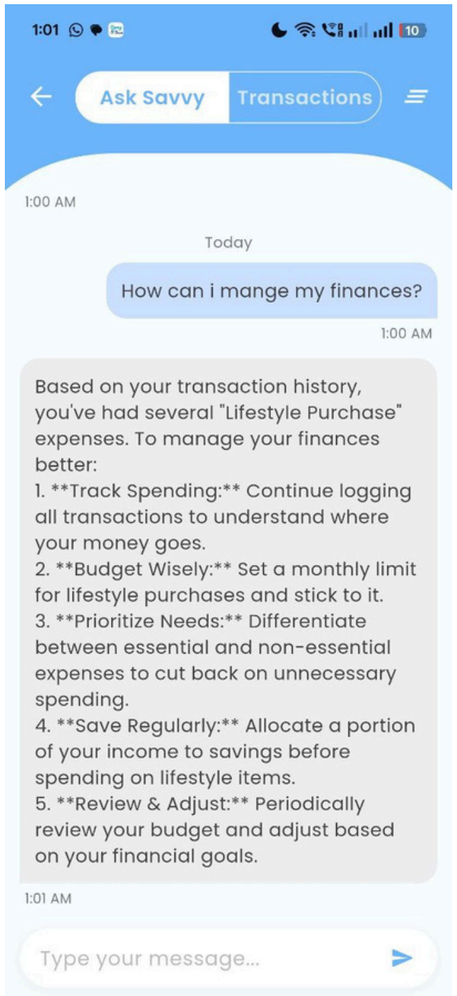
Home screen

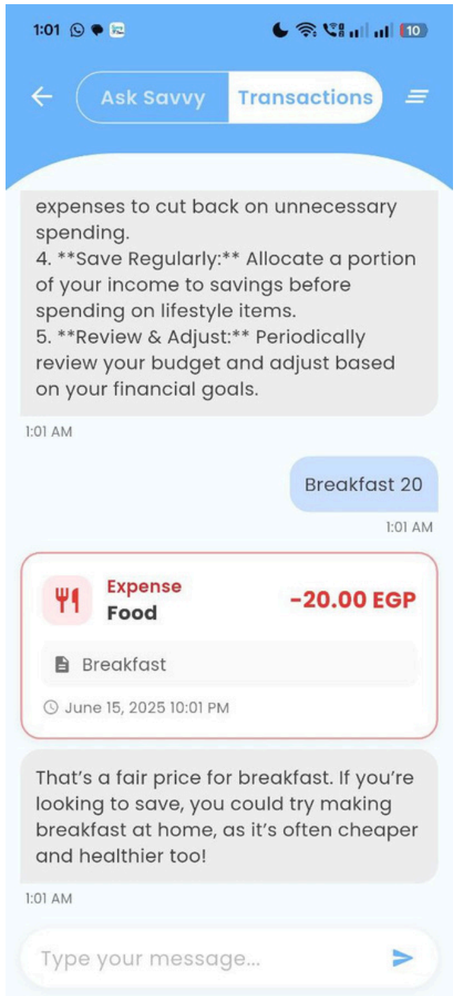

Home
Welcome, yossef
Quick summary of this month's finances
Balance
15K EGP
Income
18K EGP
Expenses
2K EGP
Start tracking your expenses to get monthly spending predictions and insights!
Quick Actions
Add Income
Add Expense
View Reports
Use the chatbot to easily perform actions like adding income or expenses. Get insights and advice to achieve your financial goals!

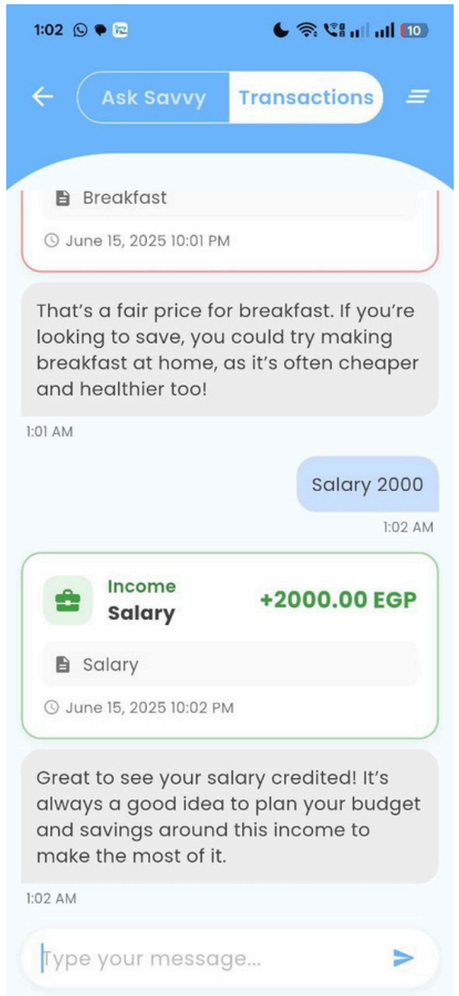

Home
Welcome, luna
Quick summary of this month's finances
Balance
13K EGP
Income
19K EGP
Expenses
430 EGP
Expense Insight
This month's predicted expenses: 1709.7 EGP
This month's current expenses: 430.0 EGP
You're expected to spend less this month.
Quick Actions
Add Income
Add Expense
View Reports
Use the chatbot to easily perform actions like adding income or expenses. Get insights and advice to achieve your financial goals!

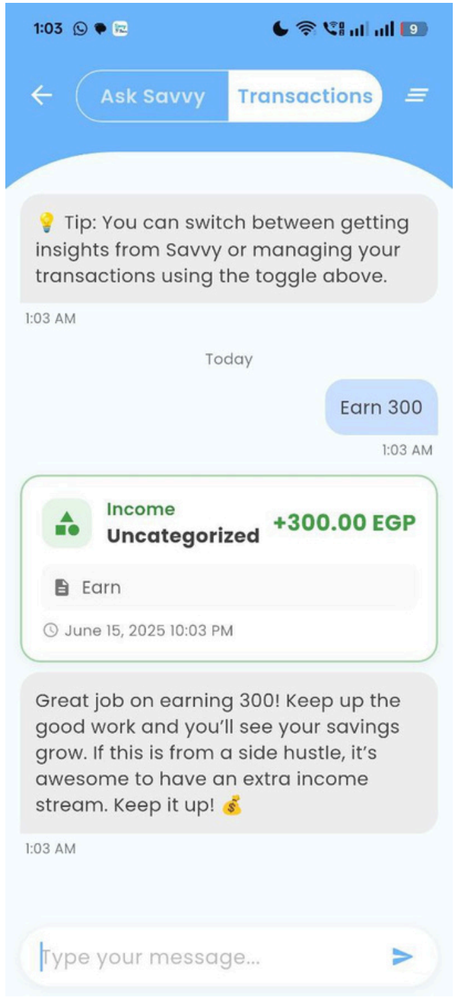

Home
Welcome, joe adel
Quick summary of this month's finances
Balance
200K EGP
Income
149K EGP
Expenses
2K EGP
Expense Insight
This month's predicted expenses: 1710.0 EGP
This month's current expenses: 2090.0 EGP
Your expenses may increase. Consider adjusting your budget.
Quick Actions
Add Income
Add Expense
View Reports
Use the chatbot to easily perform actions like adding income or expenses. Get insights and advice to achieve your financial goals!

Chatbot screen









Demo Time


2:15
Transactions
Total Balance
69.6K EGP
Income
272.2K EGP
Expenses
177.9K EGP
Food
2025-06-15 • F...
-10.0K EGP
Education
2025-06-15 • Ed...
-5.0K EGP
Transportation
2025-06-15 • Tra...
-1.0K EGP
Transportat...
2025-06-15 • T...
-300.0 EGP

Time Series Analysis

Data selection

Goal: Predict future monthly expenses to help users plan their budgets.

Dataset Overview: Contains:

1. **Date:** (from 9-2022 to 1-2025)
2. **Amount:** Expense amount.
3. **Category:** food, Transport,...etc.

Benefits for Modeling:

- Enables multivariate forecasting (use categories as features).
- Easy to preprocess: can group by month and category.
- Reflects personal budgeting behavior, aligned with app goals.

Data preprocessing

- **Category** Mapping Aligned with App Design.

- **Outlier Detection & Removal (Ensemble Method)**
  - Combined KNN, Local Outlier Factor (LOF), and Isolation Forest to remove anomalous transactions.
  - Prevents skewing of predictions due to extremely large or rare transactions.

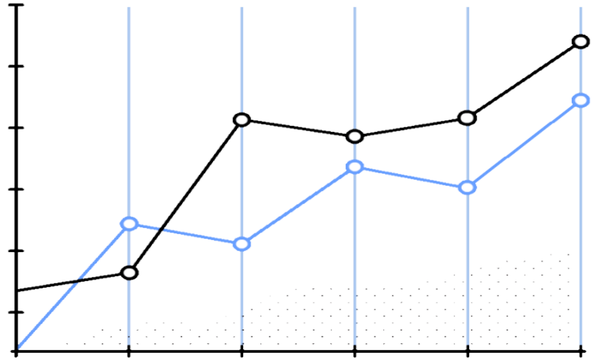
- **Monthly Aggregation**
  - Grouped daily transactions into monthly totals per app category.

Data Preparation

- **Features:**
  - "Education", "Entertainment", "Fashion", "Food", "Lifestyle", "Transportation", "Health"
- **Target:**
  - Expenses (total monthly spending).

Preprocessing Steps:

- Standardization:
  - Applied Standard Scaler to all features.
  - Ensures consistent scale across different spending categories.
- Train/Test Split:
  - 85% of data for training.
  - The remaining 15% for testing.
  - No shuffling — preserves time order.



Data Preparation

Sequence Design:

- Sliding Window Size: 3 months
  - For each prediction, use the last 3 months of all 7 features.

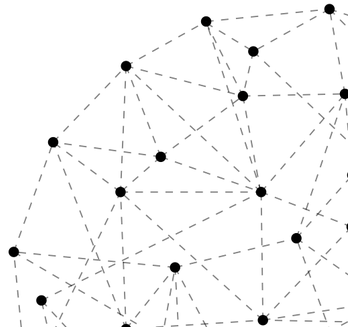
Testing multiple window sizes (e.g., 1, 3, and 6) showed that 3 months consistently led to better predictive performance, as measured by lower error metrics (RMSE).

Model Architecture

- **Input Layer**
  - Shape: 3 time steps × 7 features.
  - Each sample represents 3 months of spending across 7 categories.
  - Enables the model to learn temporal dependencies in multivariate data.
- **LSTM Layer 1 – 64 Units**
  - Processes the full 3-month sequence.
  - Learns short-term temporal patterns in each spending category.
  - Passes a sequence of hidden states to the next layer.
- **LSTM Layer 2 – 64 Units**
  - Takes output from LSTM 1 and condenses it into a single vector.
  - Captures higher-level sequential patterns across all categories.
  - Outputs a fixed-length representation of the input sequence.

Model Architecture

- **Dense Layer – 64 Neurons (ReLU Activation)**
  - Transforms the learned temporal features into a richer nonlinear space
  - Allows the model to understand complex relationships between categories and expenses
- **Dropout Layer – 20%**
  - Randomly disables 20% of neurons during training
  - Helps prevent overfitting by making the model less reliant on specific paths
- **Output Layer – 1 Neuron**
  - Outputs a single continuous value
  - Represents the predicted total expenses for the next month.



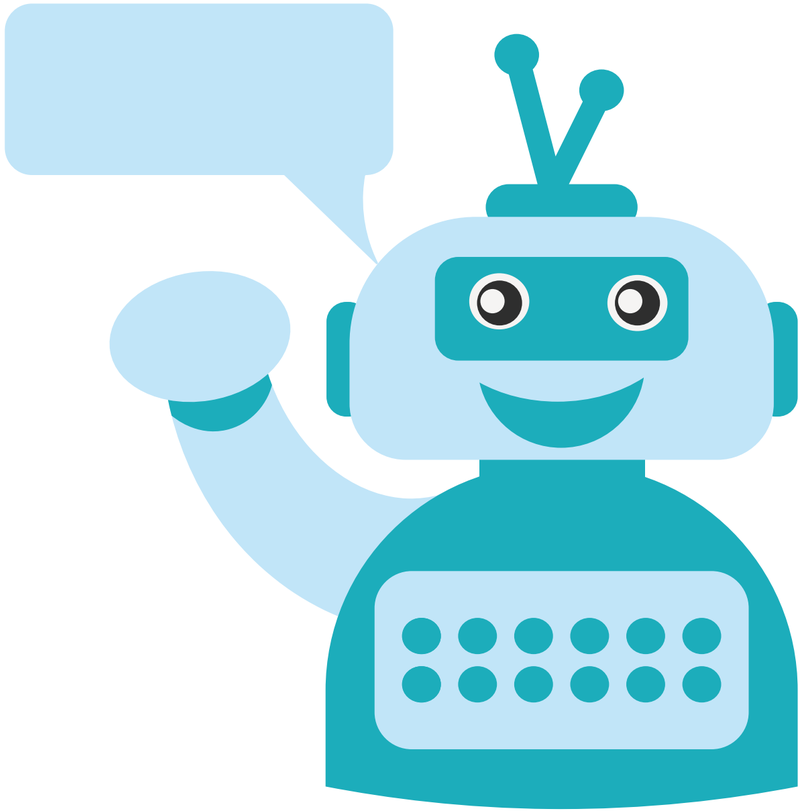
Model Evaluation – Results

- **Test Setup:**
  - Input: Past 3 months of spending data across 7 categories.
  - Output: Predicted total Expenses for the next month.
  - Test Size: Final 15% of the dataset.

- **Evaluation Metric:**
  - Root Mean Squared Error (RMSE): 312.49

Models Comparisons

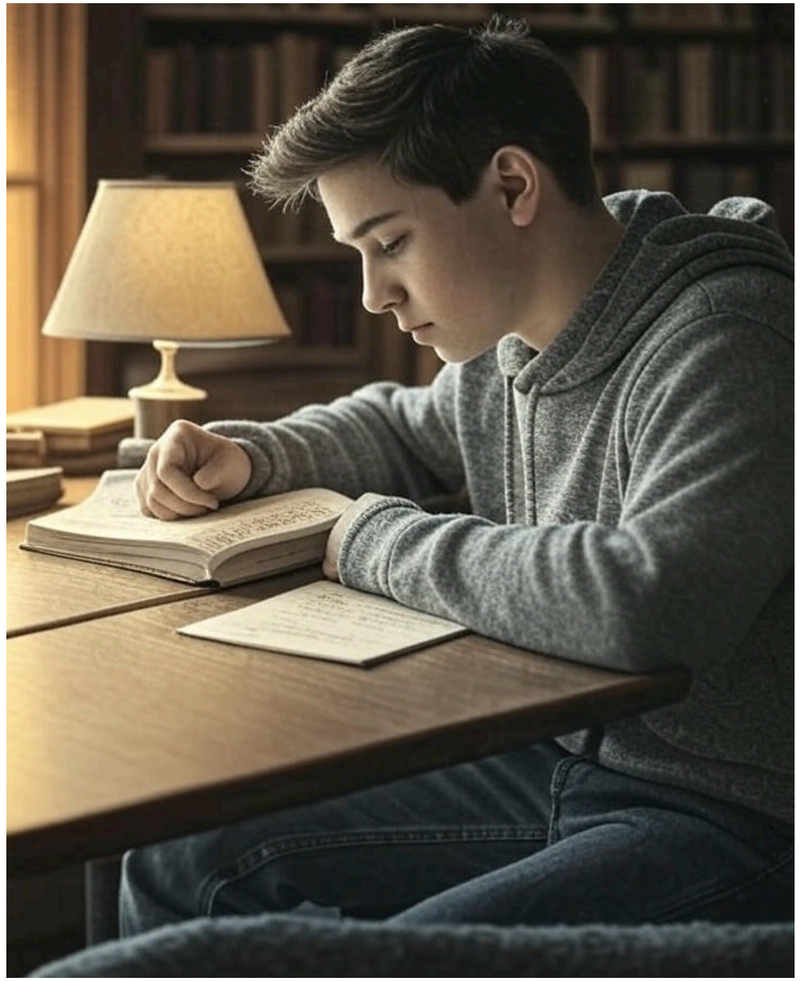
Model	Input Type	RMSE
SARIMA	Univariate (past Expenses)	**840.15**
Univariate LSTM	Past 3 months of Expenses	**510.90**
Multivariate LSTM	Past 3 months Expenses of 7 Categories	**312.49**



AI-Chatbot

What is RAG?

Retrieval-Augmented Generation is the process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response.



RAG Architecture Overview

Step 1 : Documents Loading

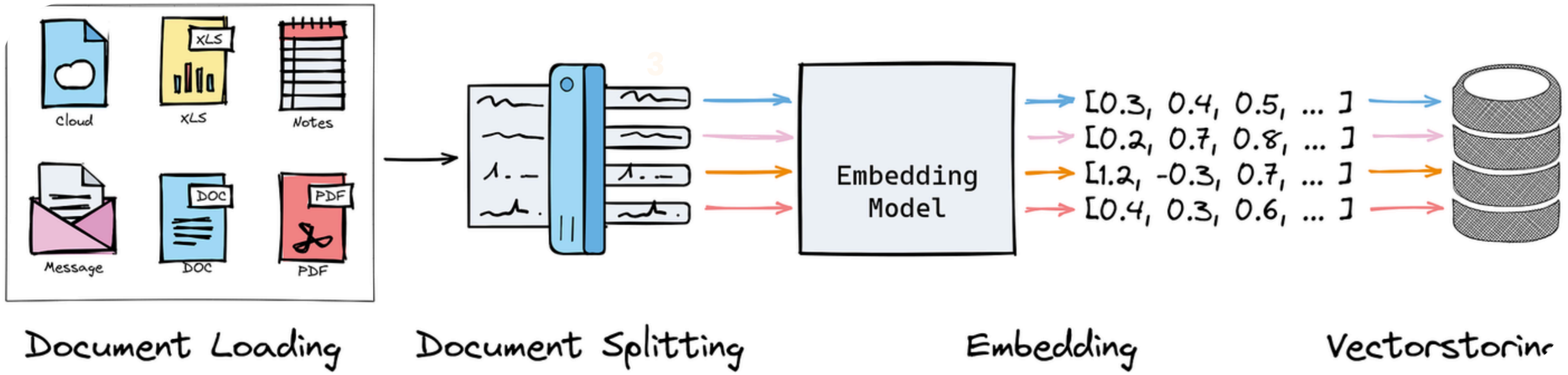
Load data from various sources, including PDFs, text files, Web Pages, databases, CSV, JSON, Unstructured data, Research papers, and more.

Step 2 :Embedding generation

When the documents are loaded successfully, they are broken down into smaller parts or chunks and enable further processing by the language models.

Step 3 : Vector storing

- Vector stores are specialized data stores that enable indexing and retrieving information based on vector representations.
- Vector stores are frequently used to search over unstructured data, such as text, images, and audio, to retrieve relevant information based on semantic similarity rather than exact keyword matches.

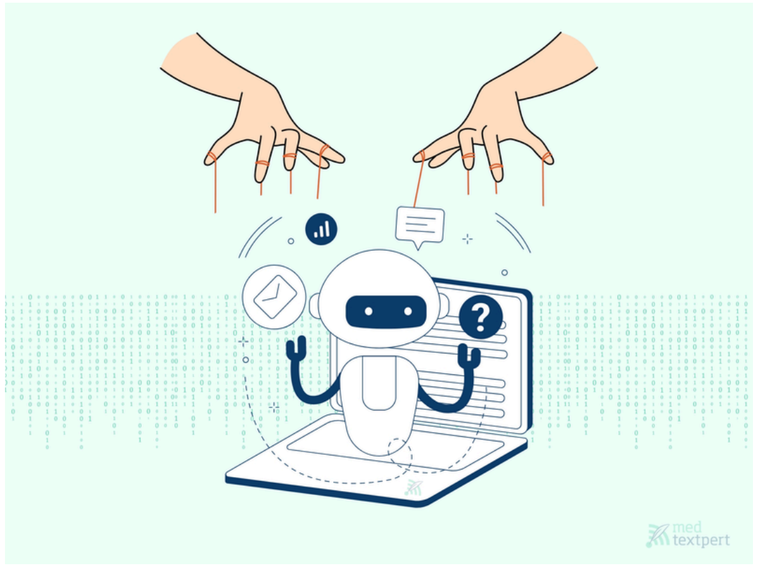


Step 4 :Semantic Retrievers

- Semantic Retrievers focus on understanding the underlying context of a query and documents in order to retrieve the relevant information from the database.

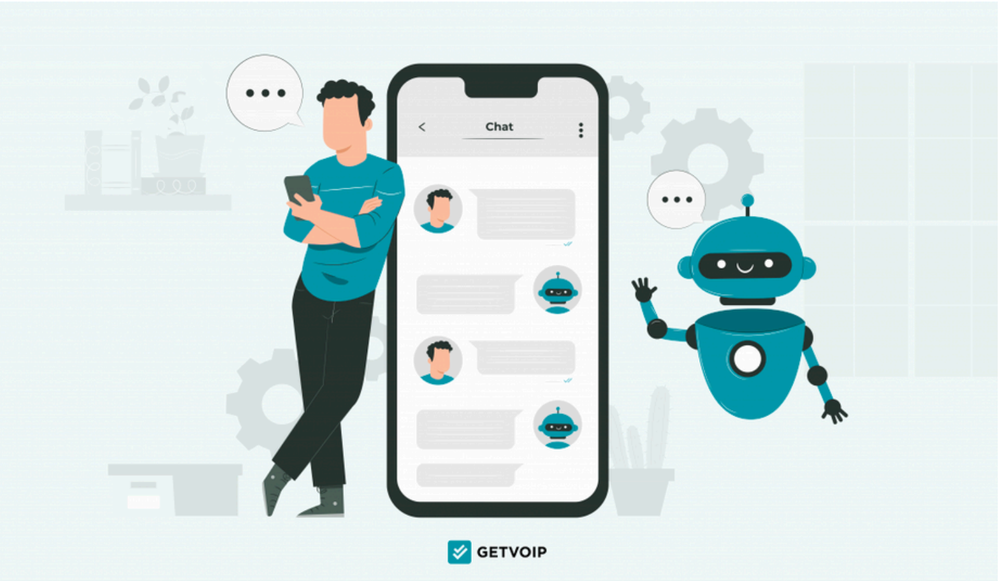
Step 5 :Prompt Construction

- Providing a clear set of instructions and examples for LM's responses.

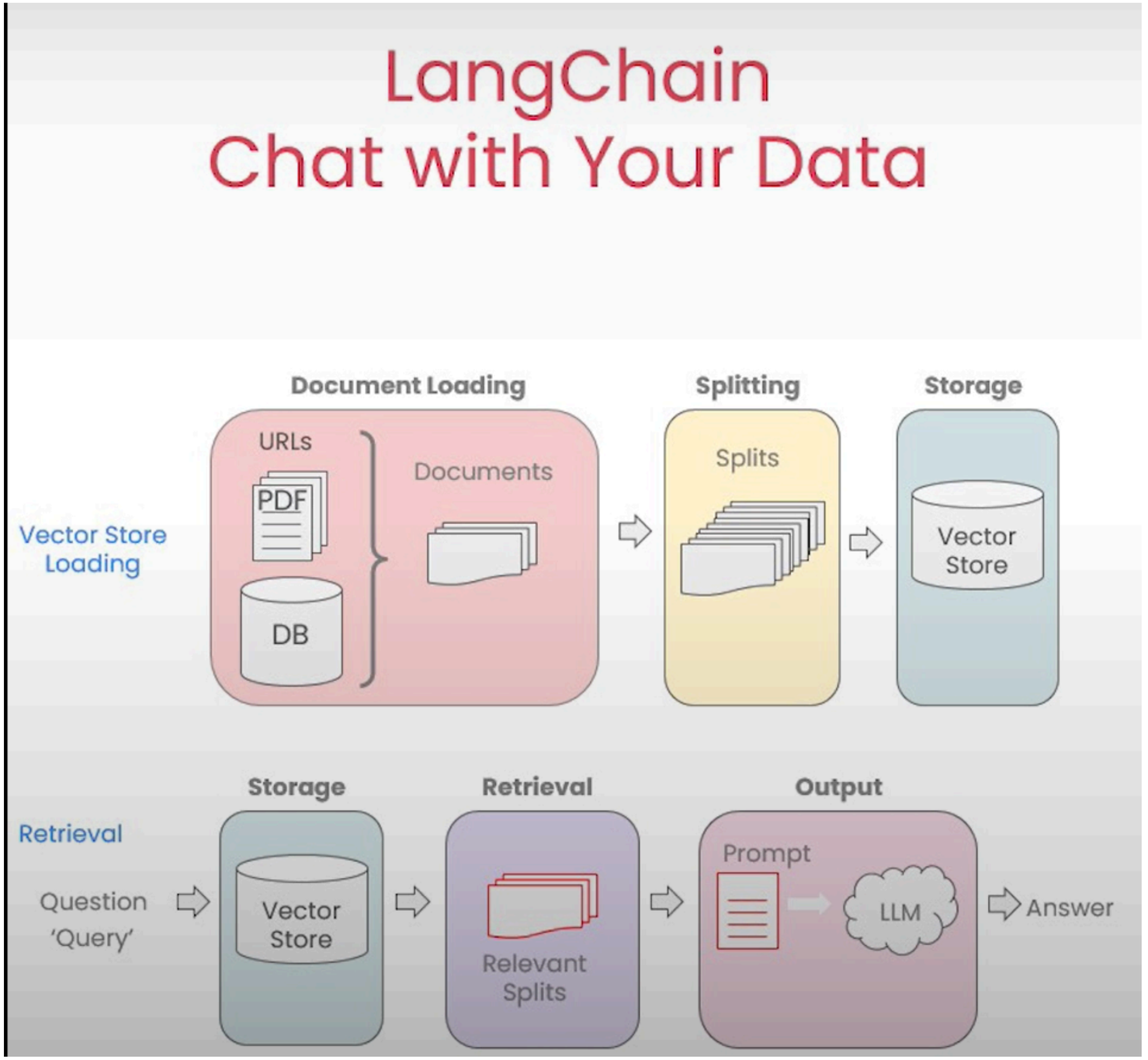


Step 6 : Text generation

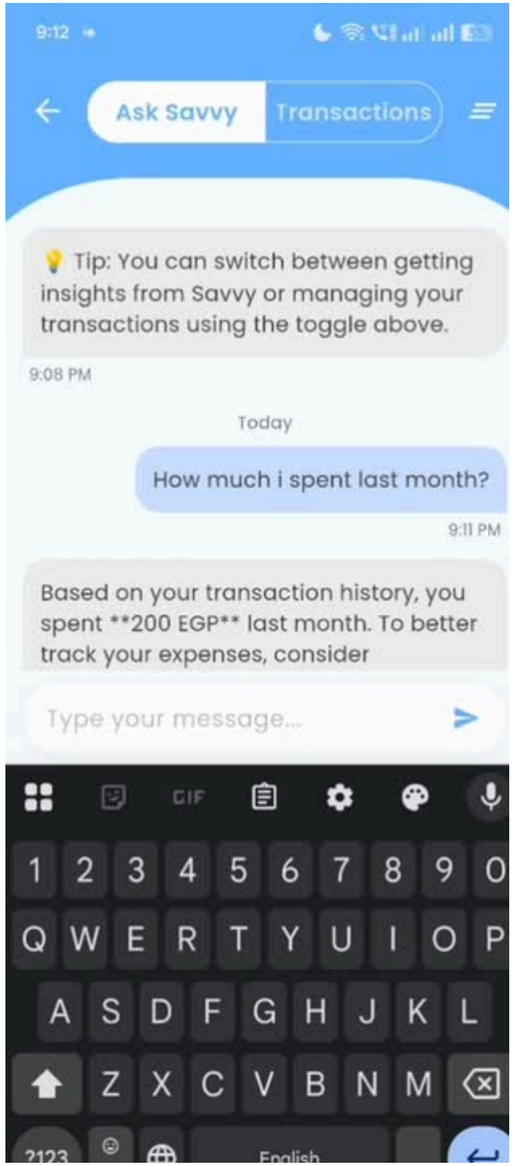
- Transformer model produces responses based on context.
- Deliver accurate, context-rich generated text.



LangChain
Chat with Your Data

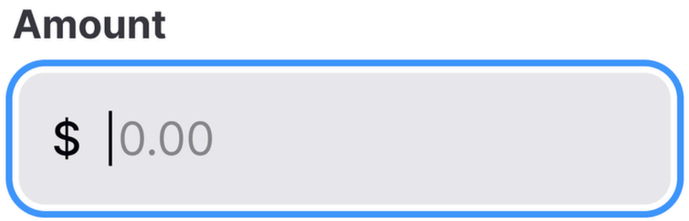


Demo Time


9:12
Ask Savvy
Transactions
Tip: You can switch between getting insights from Savvy or managing your transactions using the toggle above.
9:08 PM
Today
How much i spent last month?
9:11 PM
Based on your transaction history, you spent **200 EGP** last month. To better track your expenses, consider
Type your message...
GIF
1 2 3 4 5 6 7 8 9 0
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
?123
English

Why Transactions Chatbot?

People forget to track daily expenses.



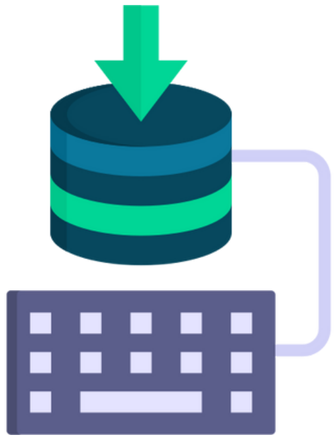
Budgeting apps require manual input , boring and time-consuming.

Users crave real feedback, not just data storage.

Transactions Chatbot as a Solution

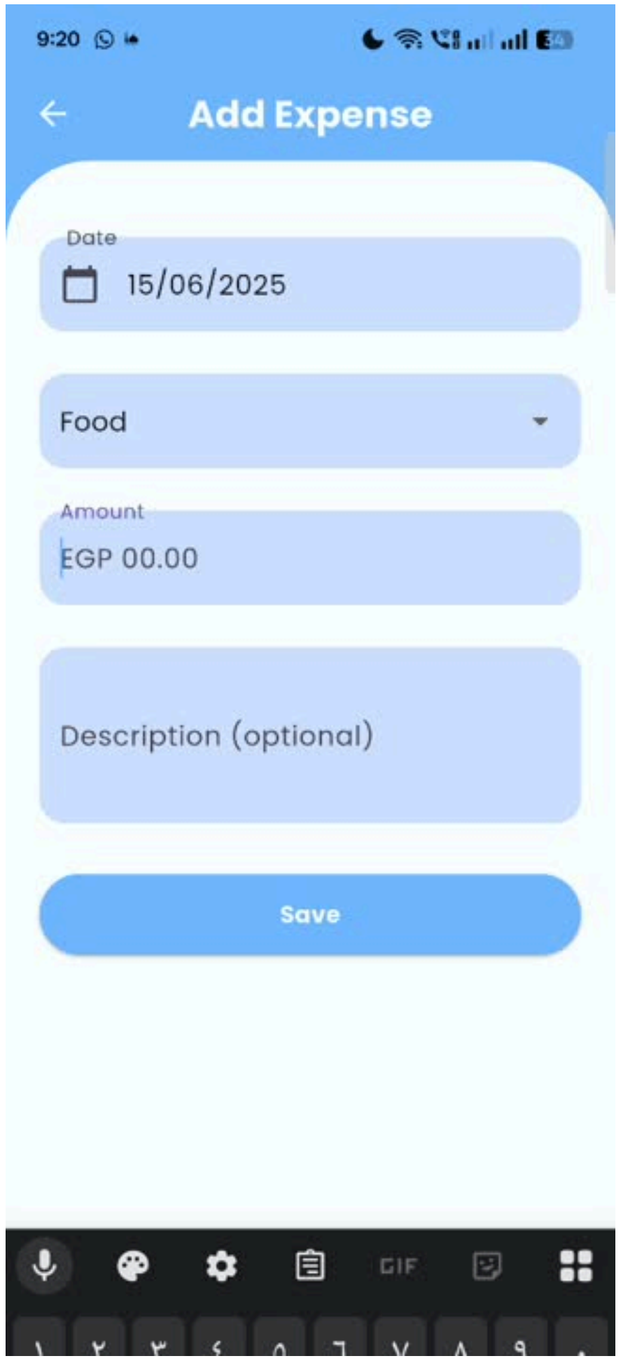
Extract data from user input

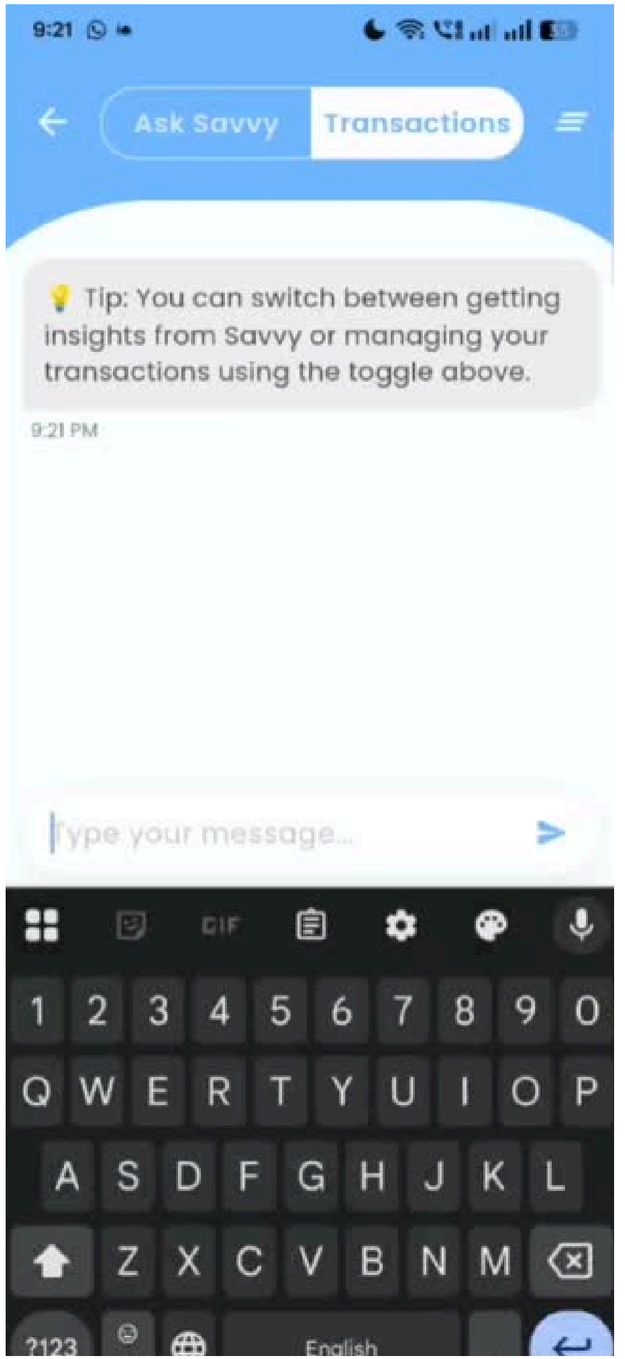
Smart Categorization



Add transaction to Database

Provide Personalized feedback


9:20
Add Expense
Date
15/06/2025
Food
Amount
EGP 00.00
Description (optional)
Save


9:21
Ask Savvy
Transactions
Tip: You can switch between getting insights from Savvy or managing your transactions using the toggle above.
9:21 PM
Type your message...

The Journey

BERT-based System (Phase 1)

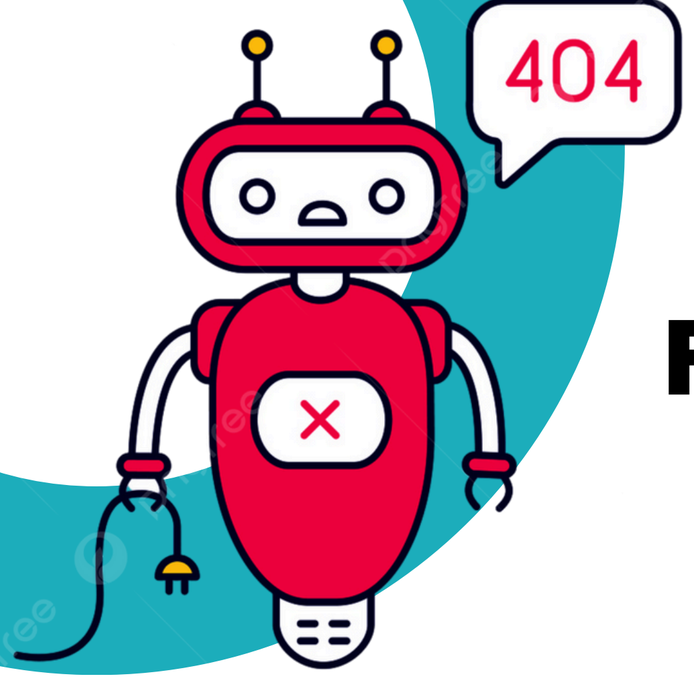
- Limited NLU
- Decent Classification
- Robotic & bland feedback

Hybrid Prompting (Phase 2)

- Intent collision between flows
- Fuzzy boundaries, unclear outputs

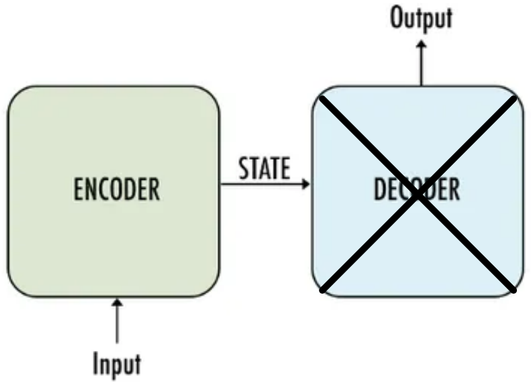
Custom LLM Prompting (Phase 3)

- Conversational, context-aware, robust
- Seamless entity extraction, smart categorization
- Feedback that feels human

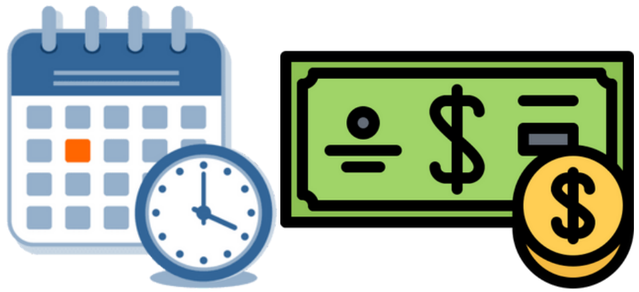


From BERT to Breakthrough

Encoder Model



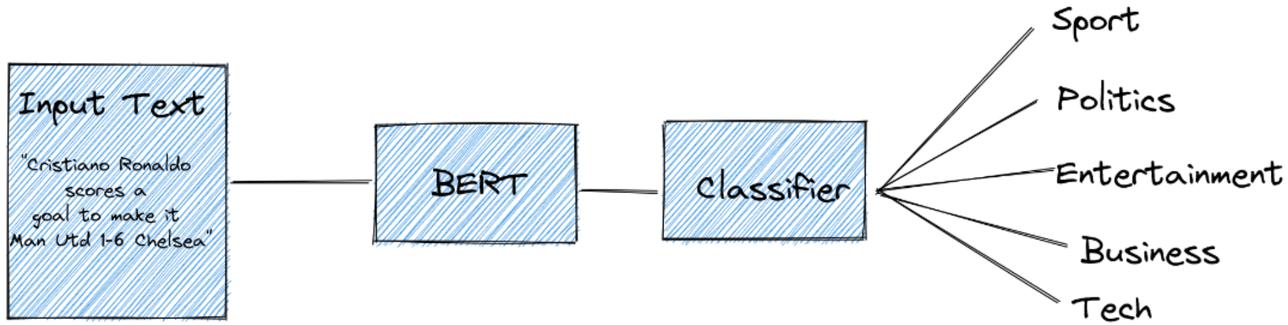
Limited Natural language understanding



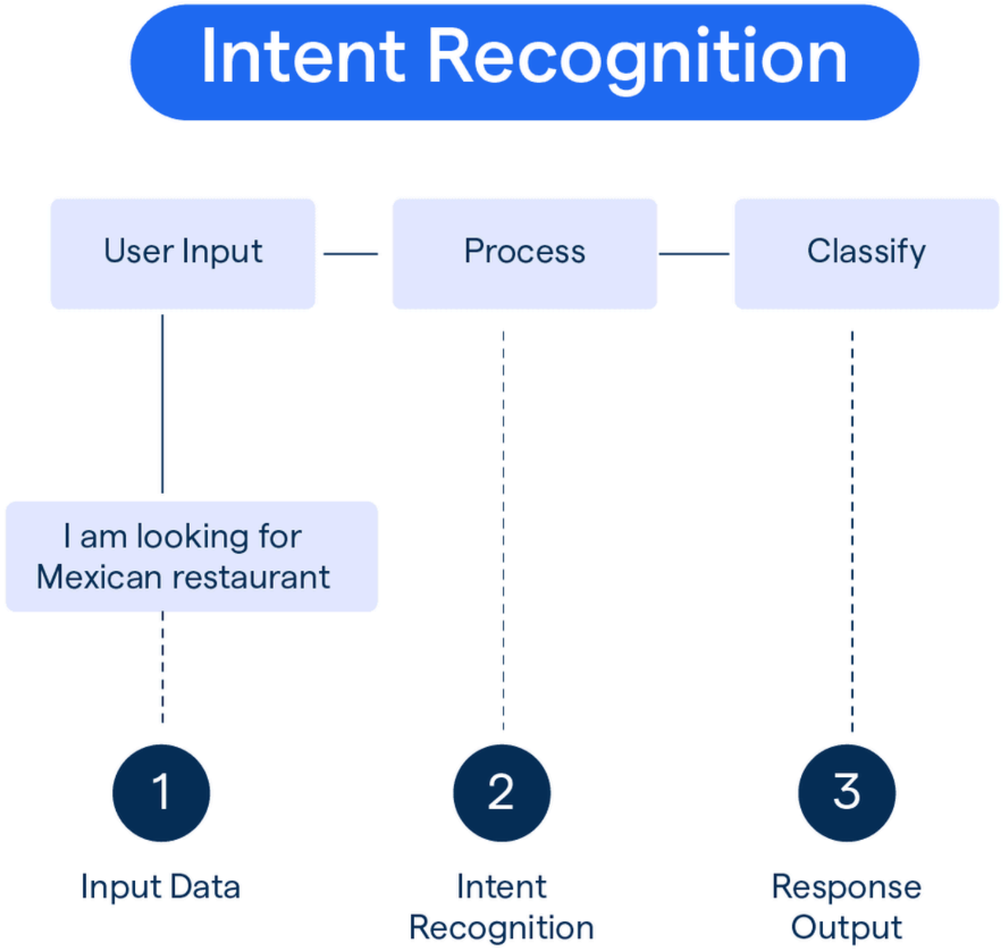
Date and time

Amount

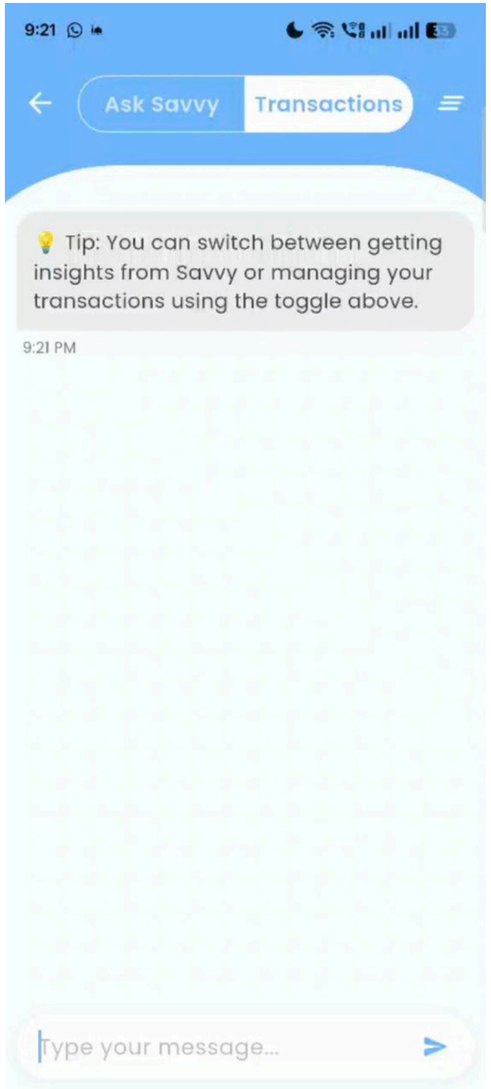
Decent Classification



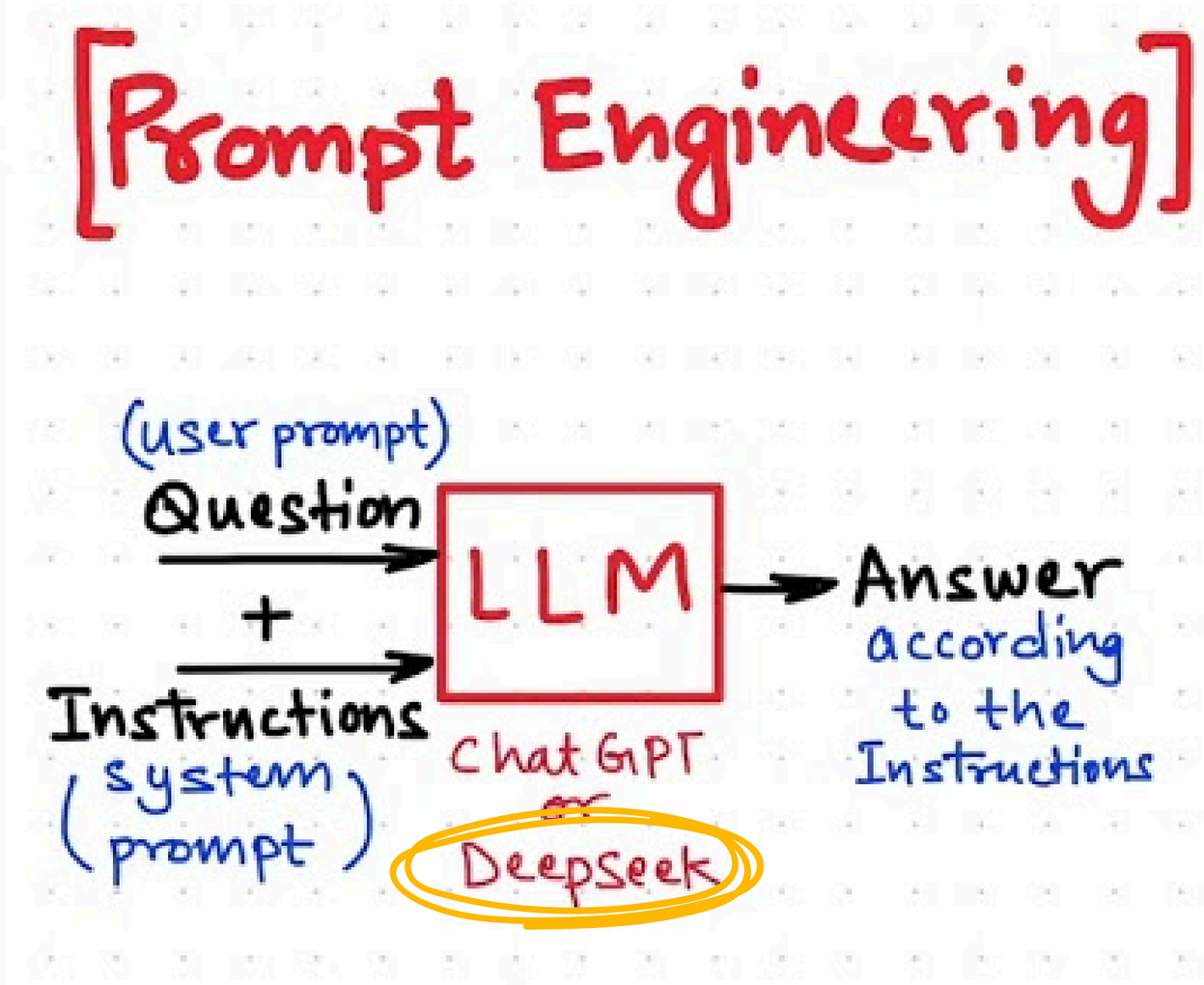
Combined Prompt



Separated Tasks



The Breakthrough: Custom LLM Prompting



deepseek

Prompt Magic: Design Principles

1

Structured Output

Enforced template (CREATED_AT, AMOUNT, etc.)

2

Natural Date Parsing

Understands “yesterday,” “this morning,” etc.

3

Controlled Categories

Only allowed options, always consistent

4

Conversational Feedback

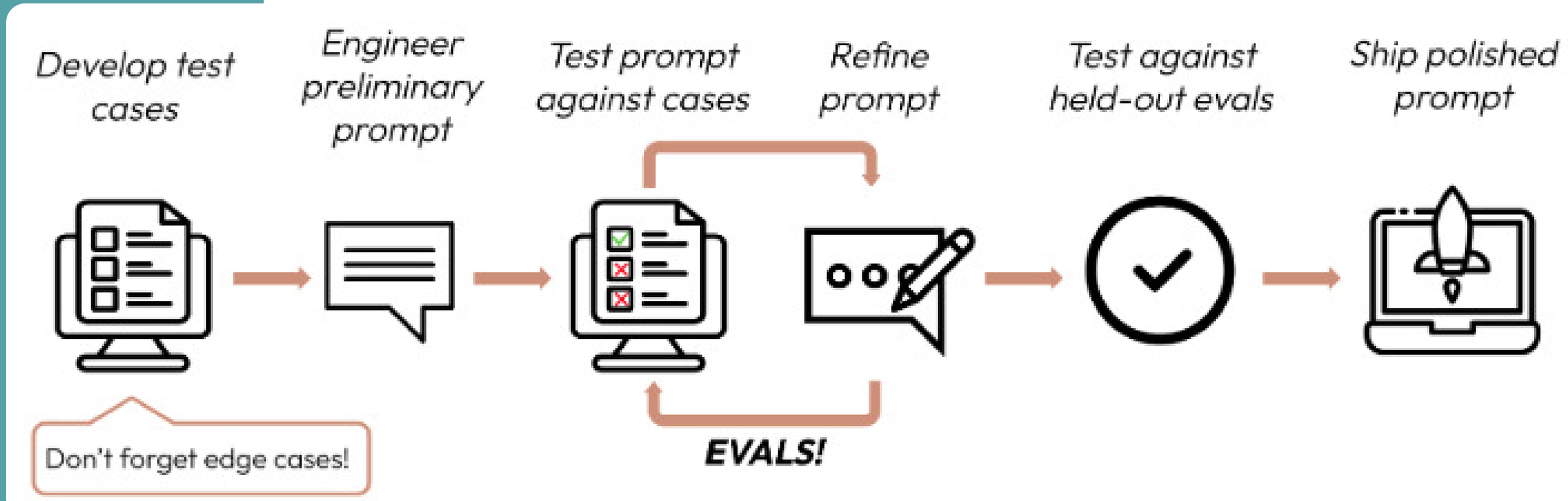
Empathetic, actionable advice (not just “OK!”)

5

Few-Shot Examples

Guides LLM for clarity & reliability

Final Step : Test and Evaluation



Putting It All Together

By implementing these strategies, you can achieve financial success. Creating a budget, saving, and investing are all important steps. Take control of your finances and work towards your goals.

Future Work

User can custom their own categories

Goals Improvements

Custom Date Range Filtering

Future Work

Geofenced Notifications

app can notify users when they enter high-spending zones
(like shopping malls) with motivational or humorous alerts.

Thank You