AI-Driven Hyper-Personalization & Recommendations

Team – SmartReconcilers

- Pavani Kondru
- Priyadharsini A
- Harini Vetrivelan
- Anjali Puthillath

Overview

- Problem Statement
- Our Approach & Key Features
- Model Selection
- Design Overview
- Detail Design & Tech Stack
- Tests
- Model Efficiency
- Challenges Faced

Problem Statement

- Modern customers expect highly personalized experiences that cater to their unique preferences
- Design and Develop a Gen-AI driven solution that enhances the hyper personalization by analysing customer profiles, social media activity, purchase history, sentiment data and demographic details.
- The system should generate personalized recommendations for products, services or content while also providing actionable insights for business to optimize customer engagement.

Our Approach & Key Features

Our approach to the problem statement is to consider bank customers, who will be availing one or more of the bank products and how we can personalize their experience and provide recommendations which optimizes their engagement.

- Categorize based on users transaction history and provide recommendations using AI
- Alert notification based on transaction limit
- Provide product recommendation to the user using user based collaborative filtering
- Provide recommendation based on social media data for the user
- A chat bot which provided recommendations based on key in text or uploaded voice message
- Investment Strategy using market trends

Model Selection

GPT-2 (gpt2)

- Preloaded for potential text generation.
- Could be used for AI-driven chatbot responses, though not explicitly used.

spaCy (en_core_web_sm)

- NLP-based keyword extraction.
- Helps extract relevant financial terms from tweets.

Hugging Face Transformers (pipeline("sentiment-analysis"))

- Sentiment classification of tweets.
- Determines if a user’s tweet is positive or negative, influencing product recommendations.

LightFM

- Collaborative filtering for personalized product recommendations.
- Trains on user interactions to suggest relevant financial products.

Sentence-Transformers (all-MiniLM-L6-v2)

- Embedding-based search for chat recommendations.
- Helps find relevant financial suggestions based on user input.

FAISS

- Efficient similarity search for recommendation queries.
- Speeds up AI-powered customer support.

VADER (SentimentIntensityAnalyzer)

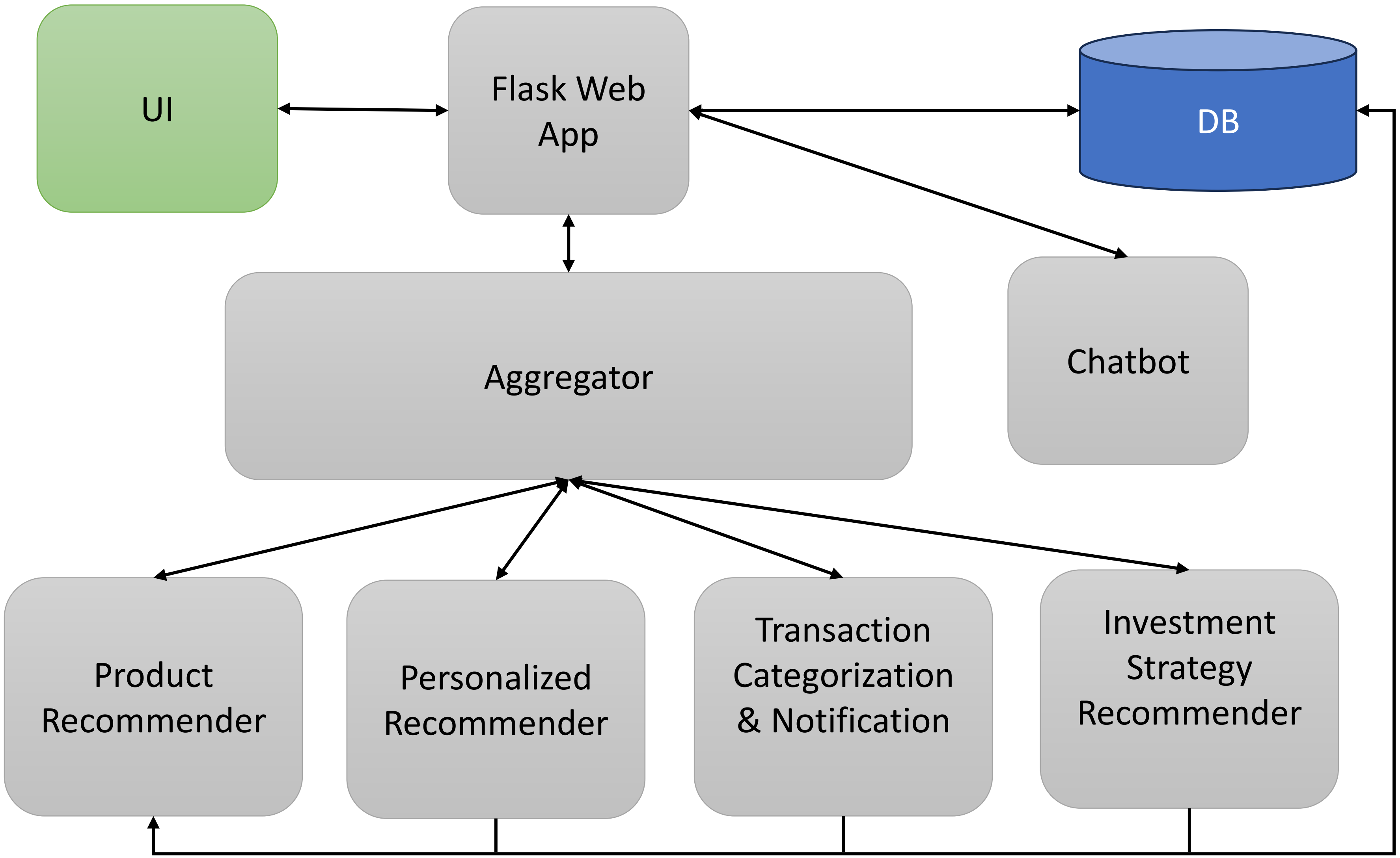
- Sentiment scoring for chatbot input.
- Provides an additional measure of user sentiment.

DistilBERT (distilbert-base-uncased)

- AI-based transaction categorization.
- Automatically classifies user transactions into categories like groceries, entertainment, and utilities.

Tensorflow Recommenders

Design Overview



Detail Design & Tech Stack

Flask-based web application with functionalities for financial transaction management, sentiment analysis, AI-driven recommendations, and user interaction handling. Key functionalities include:

- User Authentication: Login and session management.
- Database Operations: Fetching user data, transactions, savings, and loan details from DB.
- Sentiment Analysis: Extracting sentiments from tweets and using them for financial product recommendations.
- AI-Powered Recommendations:
  - Personalized product recommendations based on tweets (using LightFM).
  - Chat-based recommendations using Sentence-BERT and FAISS.
  - Transaction categorization using DistilBERT.
- Financial Alerts & Strategy: Notifications based on savings goals, transaction limits, and loan repayments.
- Visualization: Transaction category pie chart with Plotly.

Tech Stack:

- Python
- Flask
- SqlLiteDB
- HTML
- Docker

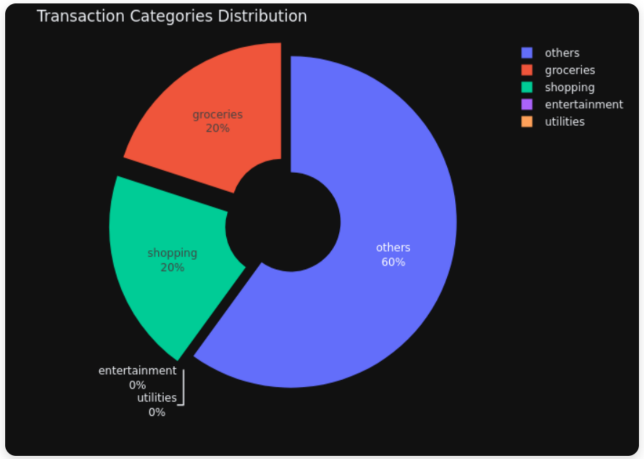
Model Efficiency

Model	Purpose	Efficiency Factors	Expected Performance
GPT-2 (Hugging Face)	Text Generation	Slow on CPU, faster with GPU, large memory usage	Moderate to slow
spaCy (en_core_web_sm)	NLP (Keyword Extraction)	Lightweight, optimized for CPU	Fast
Hugging Face Sentiment Analysis (pipeline("sentiment-analysis"))	Sentiment Analysis	Optimized for transformers, medium computational load	Fast (~30ms per sentence on GPU)
LightFM	Collaborative Filtering (Recommendations)	Sparse matrix computation, efficient for large datasets	Fast with sparse data
Sentence-Transformers (all-MiniLM-L6-v2)	Text Embedding for Recommendations	Compact model, optimized for inference speed	Moderate (~10ms per query on GPU)
FAISS	Vector Search for Similarity Matching	Highly efficient, optimized for large-scale retrieval	Very fast (~1ms for 1M embeddings)
VADER Sentiment Analyzer	Rule-based Sentiment Analysis	Lightweight, CPU-efficient	Very fast (~1ms per sentence)
DistilBERT (distilbert-base-uncased)	Transaction Categorization	Smaller BERT model, 60% faster than BERT	Moderate (~40-50ms per transaction on GPU)

Tests

Welcome,User!

Transaction Categories



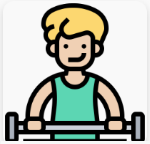
Notifications

Alert: You have exceeded your transaction limit of $5000.

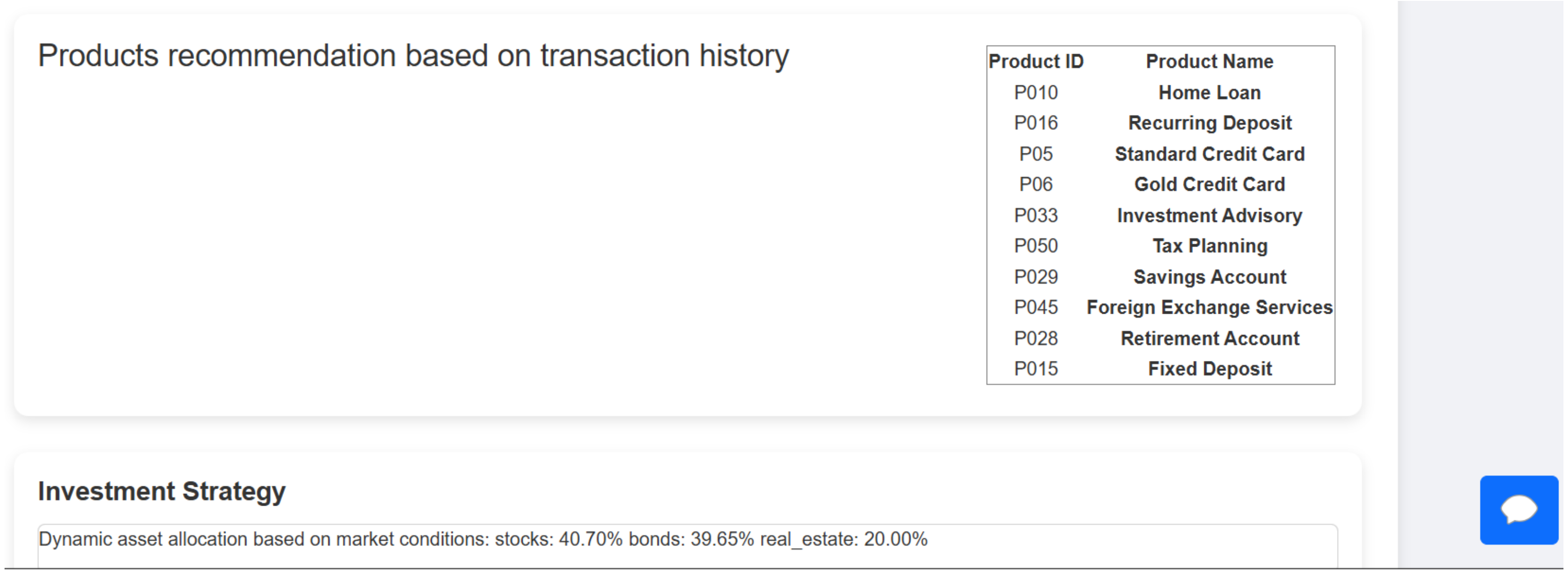
Recommendations based on social media interaction

Retirement Plan

Investment Plan



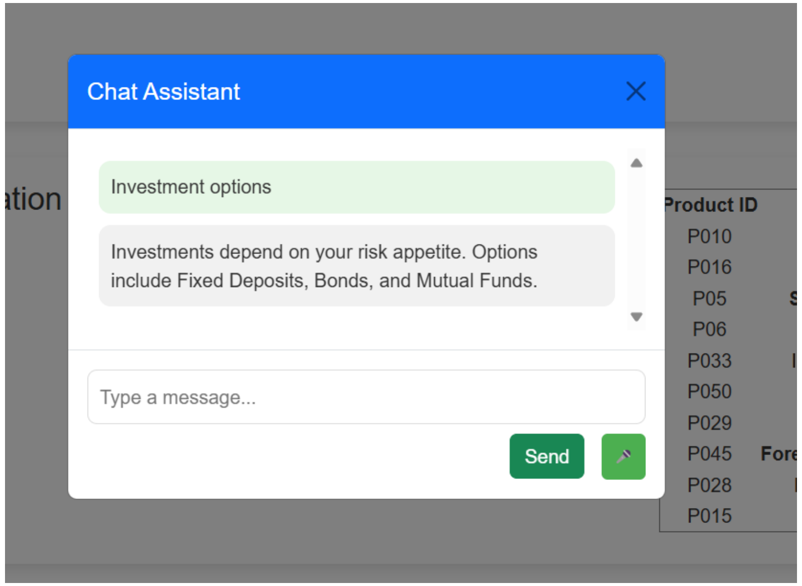
High-Yield Savings


Products recommendation based on transaction history

Product ID	Product Name
P010	Home Loan
P016	Recurring Deposit
P05	Standard Credit Card
P06	Gold Credit Card
P033	Investment Advisory
P050	Tax Planning
P029	Savings Account
P045	Foreign Exchange Services
P028	Retirement Account
P015	Fixed Deposit

Investment Strategy

Dynamic asset allocation based on market conditions: stocks: 40.70% bonds: 39.65% real_estate: 20.00%



Challenges Faced

- Open AI provided relevant recommendations but its services can be availed only through paid subscription
- Dependency resolution for different models while creating docker image

Thank You