**Personal Finance Advisor Project Report**

**1. Define the Objective**

**What to Do**

The primary objective of this project is to develop an AI-powered Personal Finance Assistant that provides personalized financial advice to users. The system is designed to:

* Offer budget management guidance
* Provide investment suggestions
* Handle finance-related queries exclusively
* Deliver personalized responses based on user input

**How to Approach**

Success metrics for this project include:

* Accuracy and relevance of financial advice
* Appropriate boundary setting for non-financial queries
* Response generation speed and consistency
* User satisfaction with personalized recommendations

**2. Gather and Understand Data**

**What to Do**

The project utilizes the Google Gemini API for natural language processing and response generation. Key data components include:

* User queries related to personal finance
* Pre-defined system prompts for financial context
* Chat history for conversation continuity

**How to Approach**

The project leverages:

* Google's Gemini 1.5 Flash model for optimal performance
* Structured conversation history management
* API integration for real-time response generation

**3. Data Preprocessing**

**What to Do**

The system implements several preprocessing steps:

* Query validation and formatting
* Chat history structuring
* Response text formatting for display

**How to Approach**

The implementation includes:

* Text wrapper utility for formatting responses
* Markdown conversion for better readability
* Clean input handling through the handle\_user\_query function

4. Exploratory Data Analysis (EDA)

**What to Do**

The system explores:

* Query patterns and types
* Response generation capabilities
* Model performance characteristics

**How to Approach**

Analysis includes:

* Model capability assessment through list\_models()
* Response time monitoring using %%time magic command
* Chat history structure validation

5. Feature Engineering

**What to Do**

The project implements several key features:

* Specialized chat initialization with finance-focused context
* Query handling system with finance domain expertise
* Response formatting for improved readability

**How to Approach**

Features are implemented through:

* Custom prompt engineering for financial context
* Structured chat history management
* Markdown-based response formatting

**6. Model Selection**

**What to Do**

The project utilizes Google's Gemini 1.5 Flash model, selected for its:

* Optimal performance in conversational AI
* Speed and efficiency in response generation
* Cost-effectiveness for deployment

**How to Approach**

Model selection process included:

* Evaluation of available Gemini models
* Selection of Flash variant for optimal performance
* Integration testing with the chat system

**7. Model Training**

**What to Do**

The system leverages pre-trained Gemini model with specialized configuration:

* Custom initialization for financial domain
* Specific prompt engineering for financial context
* Chat history management for consistent responses

**How to Approach**

Implementation includes:

* Model initialization with API key configuration
* Chat session management
* Response handling and formatting

**8. Hyperparameter Tuning**

**What to Do**

The system utilizes default model parameters with specific customizations:

* Chat history configuration
* Response formatting settings
* API integration parameters

**How to Approach**

Customization focuses on:

* Optimal chat history structure
* Response formatting for readability
* System prompt engineering

**9. Model Evaluation**

**What to Do**

Evaluation metrics include:

* Response relevance to financial queries
* System performance with non-financial queries
* Response generation time
* Output formatting quality

**How to Approach**

Testing includes:

* Financial query response validation
* Non-financial query boundary testing
* Performance timing analysis

**10. Deployment**

**What to Do**

The system is deployed as a Jupyter notebook with:

* Google Gemini API integration
* Interactive query handling
* Formatted response display

**How to Approach**

Deployment includes:

* API key configuration
* Dependencies management
* Interactive interface implementation

**11. Monitoring and Maintenance**

**What to Do**

The system includes:

* Response time monitoring
* Query handling validation
* Output formatting verification

**How to Approach**

Maintenance procedures include:

* Regular API key validation
* Performance monitoring
* Response quality assessment

**12. Documentation and Communication**

**What to Do**

Documentation includes:

* Comprehensive code comments
* Function docstrings
* Implementation details
* Usage instructions

**How to Approach**

Documentation is provided through:

* Inline code comments
* Markdown cells in notebook
* Function documentation
* System architecture description

**13. References**

1. Google Generative AI Documentation

* Gemini API Documentation
* Python SDK Guidelines

2. Implementation Resources:

* IPython Display Documentation
* Markdown Formatting Guidelines
  + Google Colab Documentation

[View on GitHub](https://github.com/raoalihassan228/personal_finance_advisor/blob/main/notebbok.ipynb)