AI Agent – Approach Document

Introduction

This document explains the complete journey of designing a 4 module AI Agent. It includes the use case selection, module prompts, challenges, fixes, and final execution flow.

Chosen Use Case

The goal was to build a **Study Assistant Agent** that understands user input, creates a study plan, generates learning sessions, and maintains memory across sessions.

4 Step Agent Architecture & Prompts

The system uses four clearly defined modules:

1. Input Understanding Module – Reads the user request and outputs structured JSON.

2. Task Planning Module – Converts the parsed request into actionable steps.

3. Session Generator Module – Generates the learning content based on the plan.

4. State Manager Module – Updates and stores the session state.

Prompts Used for Each Module

Each module uses strict prompts ensuring JSON safe output for planning modules and free text output for lesson generation.

InputUnderstanding Prompt:- Ensures JSON for intent, details, flags.

TaskPlanner Prompt:-Converts parsed request into 'task_type' and 'steps'.

SessionGenerator Prompt:- Creates actual learning material.

StateManager Prompt:** Updates memory in strict JSON format.

ChatGPT / Groq Interaction Snapshots

During testing, the LLM was instructed to output strict JSON. Errors occurred when: - The model produced empty responses. - The JSON had trailing commas or hallucinated text. - The model drifted into non study topics. These were fixed by adding validation, regex based JSON extraction, and restrictive system prompts.

Challenges & Fixes

1. **JSONDecodeError** – Solved by wrapping LLM output with a JSON extraction function.

2. **LLM hallucinations** – Solved with strict rules forcing domain specific output.

3. **Empty responses** – Fixed by rejecting empty strings and retrying.

4. **Groq output randomness** – Controlled by strong system messages and formatting rules.

Final Execution Flow

1. User enters a study request.

2. Previous state is loaded.

3. InputUnderstanding parses intent.

4. TaskPlanner creates the plan.

5. SessionGenerator produces the learning session.

6. StateManager updates state.json.

7. Output is shown to the user. This forms a complete cyclic loop allowing continuous learning sessions.

Conclusion

The AI Agent is capable of structured reasoning, planning, content generation, and memory management. This modular design mirrors real world autonomous agent architectures.