

**AI Agent Title: Revise Mate**

***SECTION 1:***

**Use Case:**  
AI Agent to Help College/School Students Revise from PDFs using Summaries, Notes, MCQs, and Long Questions

Submitted by:

Adrija Ghosh

Email id: [adrijamethodist@gmail.com](mailto:adrijamethodist@gmail.com)

Whatsapp No: 9051145897

***SECTION 2: PROBLEM FRAMING***

**1.1What problem did my AI Agent solve?**

It helps students revise academic PDFs quickly by generating chapter-wise summaries and short notes.  
Along with that, it provides MCQs and long questions to boost active recall and test conceptual clarity.  
Students can also track their performance through interactive quizzes.  
It removes the need for manual note-making or searching for practice questions.  
This is your one-stop solution for study notes and chapter-wise mini exams.

**1.2. Why is this agent useful?**  
It helps users revise PDF notes quickly by generating summaries and key takeaways.  
MCQs and long questions are auto-generated to test active recall and understanding.  
Tracks performance to help students identify weak areas instantly.  
Eliminates the need to manually create quizzes or summaries from notes.  
A one-stop solution for studying, testing, and measuring chapter-wise progress.

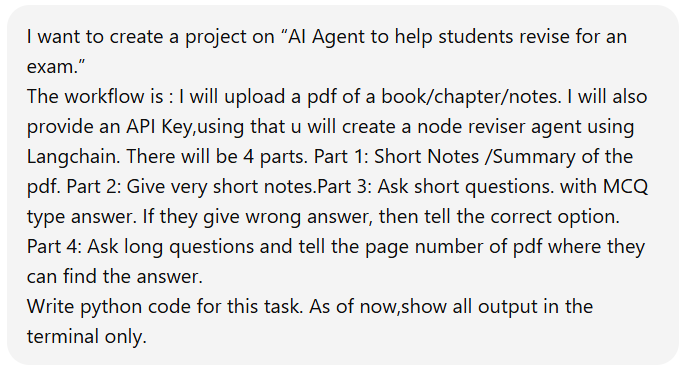
**1.3. Who is the target user?**.

Students and academicians.

**1.4. What not to include?**  
We did not include handwritten note generation or PDF annotation tools.  
No support for handwritten scanned PDFs or image-based books.  
Avoided complex dashboard analytics to keep the tool lightweight.  
Did not include external resource fetching to ensure focus on provided PDF.  
Skipped long setup or login processes to make it quick and usable instantly.

***SECTION 3: INPUT UNDERSTANDING***

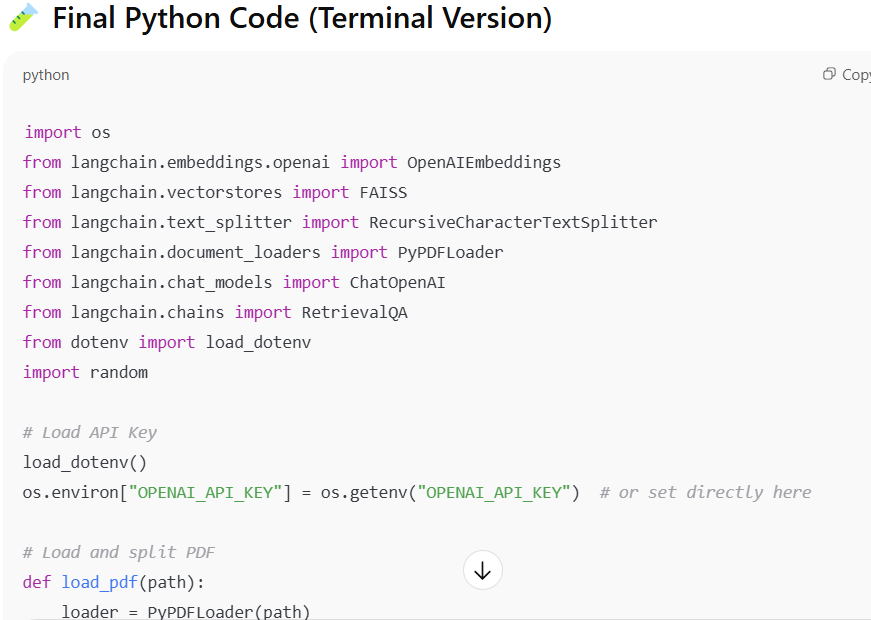
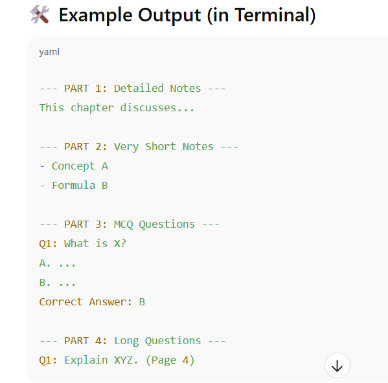
**3.1: Input Understanding**

**Prompt:** 

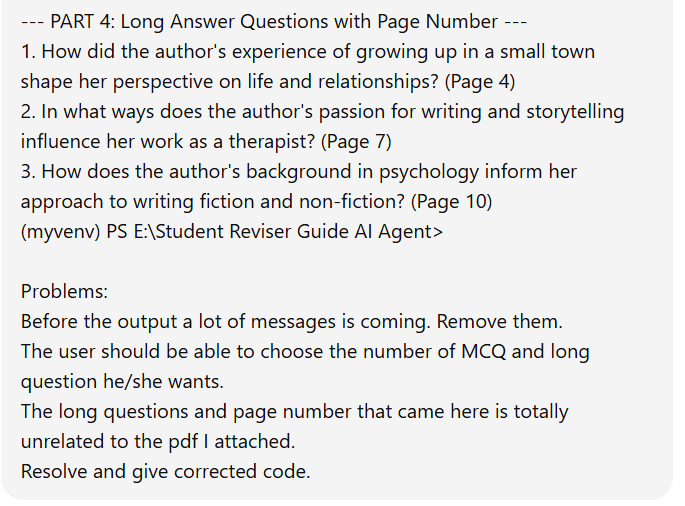
**What is this prompt responsible for?**

This prompt is giving the initial idea about the project.It is describing what is expected from the code. It divides the project into four parts and also explains the workflow of each part.

Output:



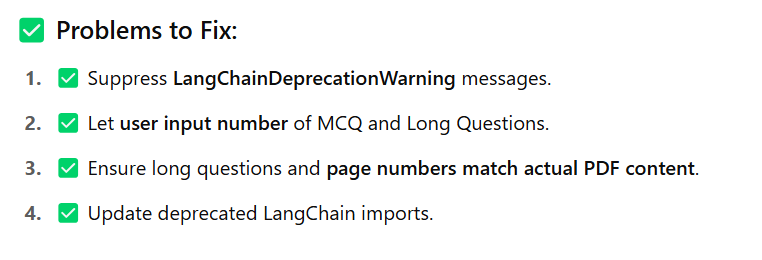
Prompt:



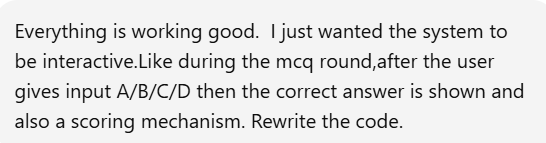
**What is this prompt responsible for?**

In the output a lot of unwanted messages was coming. And I have attached a pdf of the topic “Binary Trees” which is a part of Data Structures and Algorithms. In part 1,part 2 and part 3 the output was related to the topic that is “Binary Trees” but in part 4,the long questions were totally unrelated and chatgpt has hallucinated there. Also the page numbers of answers were incorrect. That is the reason I have prompted that addressing the problem.

**Output**



**3.2 State Tracker**

Prompt : 

**How does this help the agent “remember”?**  
Did you simulate memory with variables / system messages? If yes, how?

The agent simulates memory using **runtime variables** such as ‘score’, ‘question number’, ‘user answers’, and ‘parsed\_questions’ , which are retained throughout the session to track quiz progress and evaluation.  
This allows the system to deliver **interactive, stateful experiences**—it knows which question you're on, whether your answer was correct, and calculates your score accordingly.  
Although it doesn’t use persistent memory or databases, this form of in-session memory enables contextual continuity across inputs.  
System messages are **not explicitly used**, but the state is **preserved functionally** within Python function calls and data structures during a single run.

**3.3 Task Planner**

**What steps does your agent take internally to solve the problem?**

The agent follows these internal steps:

1. **Extracts text** from the uploaded PDF using OCR or text parsing tools
2. Parses and segments the content into topics, headings, and subtopics.
3. Generates MCQs and long questions using OpenAI API prompts based on extracted content.
4. Stores user answers, compares them with correct answers, and updates the score.
5. Displays evaluation metrics to help users assess performance.

**Did you use chaining? Branching? How did you manage complexity?**

Yes, we used **chaining** to manage task flow—each function feeds into the next (PDF ➝ Text ➝ Questions ➝ Quiz ➝ Evaluation).  
We avoided unnecessary **branching** to keep the logic linear and predictable.  
Complexity is managed by **modularizing** each task (like parsing, question generation, scoring) into clean, reusable functions.  
State (like score, current question) is tracked via **session variables**, making the experience interactive yet simple.  
This structure keeps the codebase scalable and easy to debug or expand.

**3.4 Output Generator**

**What kind of output formatting or phrasing did you aim for?**  
The output is designed to be **clear, interactive, and user-friendly**. Each stage (summary, questions, answers, scores) appears in visually distinct sections with headings and separators.

**Any special behavior?**

* Used **markdown formatting** for summaries and instructions
* Presented **MCQs with radio buttons** for easy selection
* Gave **instant feedback** after each answer (Correct/Incorrect)
* Tracked and displayed **final score** clearly
* Used a conversational tone to make it feel like a study assistant

This makes learning engaging and structured while keeping the UI clean.

***SECTION 4: CHATGPT EXPLORATION LOG***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attempt #** | **Prompt Variant** | **What Happened** | **What You Changed** | **Why You Changed it** |
| 1 | “I want to extract summary short notes from a PDF and generate MCQs and long questions using OpenAI.” | Received basic script to extract summary, short notes and generate MCQs using OpenAI API. | Prompted to list down all the required libraries in requirements.txt. | This is because it was showing errors due to unavailability of libraries. |
| 2 | |  | | --- | |  |  |  | | --- | | “Rewrite the code so that it does not show warnings in the output of terminal” | | Received a cleaner code with no warnings. | Changed the old code with the new one. | Because the old code were showing warning messages while running. |
| 3 | “The long questions and the page numbers are unrelated to the pdf. Re-write the code.” | Received a fresh code with no unrelated questions.  Each question is related to the topic. | Changed the old code with the new one. | Because in the old code the long questions were generic and not as per the topic discussed in the pdf. |
| 4 | “The long questions are coming from the front pages and is not randomized. Hence important topics present in last can be missed.” | Received a new code that generates long questions from random pages. | Changed the old code with the new one. | Because the old code was giving questions from front section of pdf and neglecting the end section. |
| 5 | “In the mcq section,the answers are already print.Do the code in this way that the user will first give the answer and then the correct answer is shown.” | Received a new code that takes user input of each mcq and then give the correct answer. | Changed the old code with the new one. | Because the old code was already giving the answers and the system is not interactive. |
| 6. | “Increase the length of short notes of part 1 into 1 pager format” | Received a new code with proper sized short notes. | Changed the old code with the final and properly working code. | Because the old code was giving too short notes which is not really helpful during revision. |

***SECTION 5: OUTPUT TESTS***

TEST CASE 1 : Testing my agent with a pdf file of binary trees .

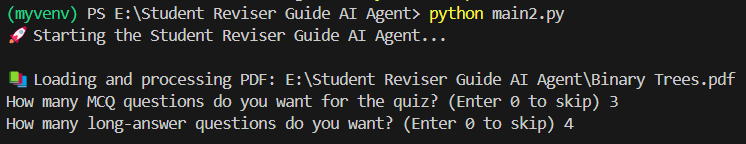


Fig 1: Asking user to specify the number of long questions and mcqs is needed.

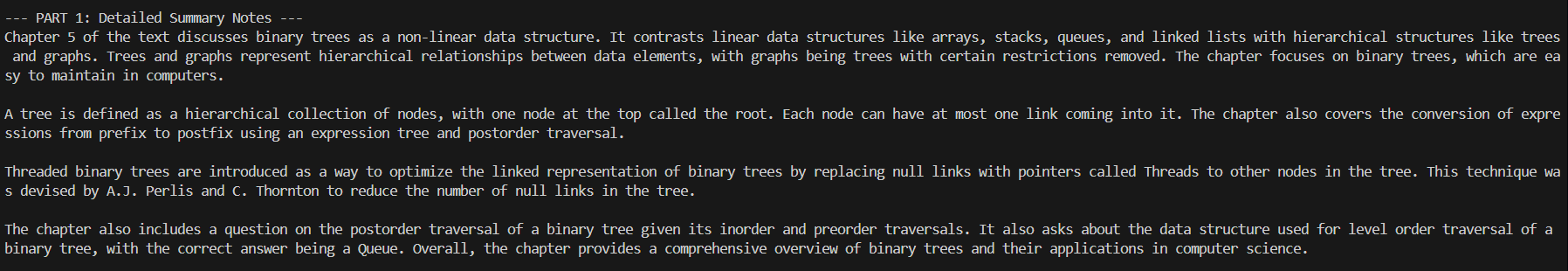


Fig 2: One page summary output

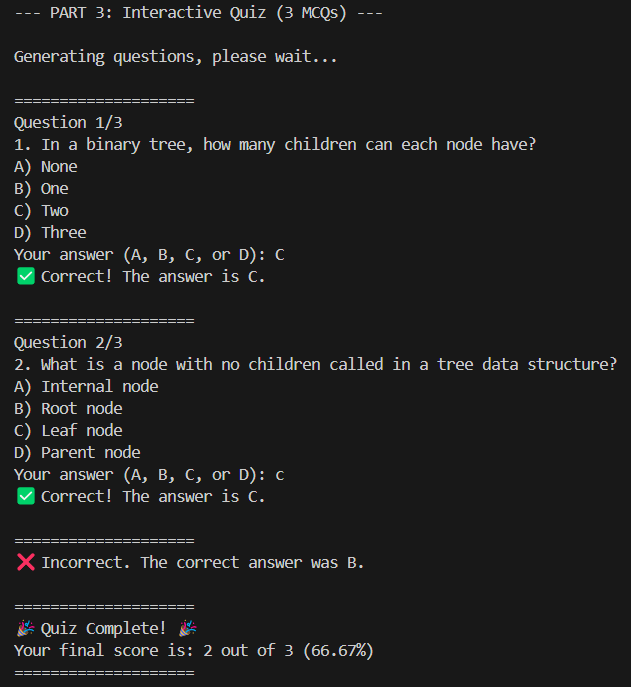


Fig 3: MCQ questions that is taking user input, matching it with correct answer and calculating final score.

TEST CASE 2: Trying to run the code without attaching the pdf file.

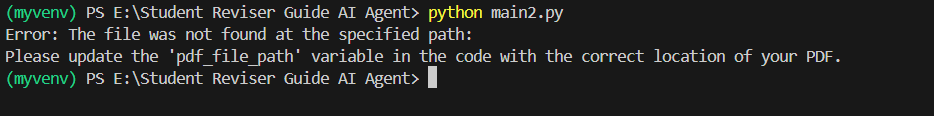


Fig 4: Error message showed with solution.

TEST CASE 3: Try giving wrong inputs instead of A/B/C/D

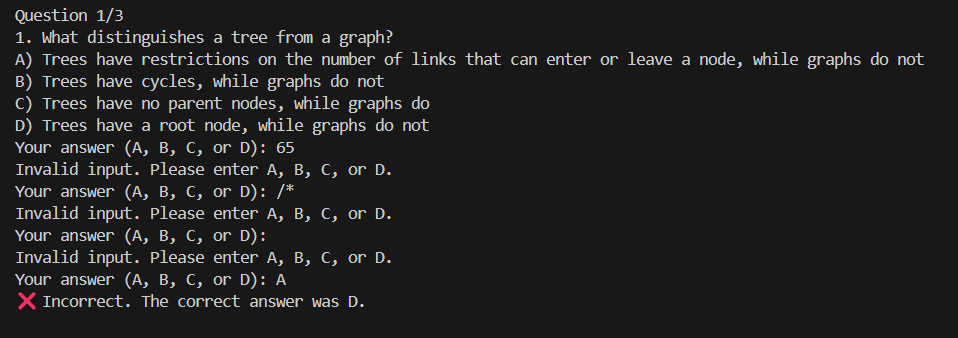


Fig 5: The loop continues till any of the 4 options is selected by user.

***SECTION 6: REFLECTION***

**6.1. What was the hardest part of this assignment?**

To make the AI Agent generate tough and academic level genuine questions for long question part i.e part 4 and also attach the correct page number to it.

**6.2. What part did you enjoy the most?**

I enjoyed playing/answering the scored mcq based questions.

**6.3. If given more time, what would you improve or add?**

I would have created a website and used my AI Agent in the backend. Also I would keep the option of uploading multiple pdfs. I would have also added the choice of selecting the difficulty level when the user will ask for summary/long questions.

**6.4. What did you learn about ChatGPT or prompt design?**

I learned that clear, specific prompts give better results. More iterations is needed to get a perfect code with every feature implemented. Giving format instructions generate better output.

**6.5. Did you ever feel stuck? How did you handle it?**

I felt stuck when the long questions of part 4 was vague and not as per the topic.

I prompted ChatGPT with pasting the error and it gave be working code .

***SECTION 7: HACK VALUE***

**Added logic for memory or role play?**

Designed the agent to simulate memory using stored extracted content and user preferences.

Added dynamic planning for generating quizzes and tracking progress.

**Explored chaining multiple agents?**

Explored **prompt chaining** (PDF → Summary → Quiz) to streamline the full learning loop.