📜 AI Agent Assignment – Final Submission Document

✅ SECTION 1: BASIC DETAILS

Name: Ritesh Kumar
AI Agent Title / Use Case: AI Agent to help students revise for an exam

---

🧠 SECTION 2: PROBLEM FRAMING

2.1 What problem does your AI Agent solve?
Students often feel overwhelmed before exams and don't know how to plan their revision. This agent breaks down a limited study timeframe into manageable daily tasks based on user input.

2.2 Why is this agent useful?
It helps students personalize their study schedule in seconds using natural language. This saves time, reduces stress, and improves focus by tailoring suggestions to specific topics and preferences.

2.3 Who is the target user?
Any student preparing for an exam with limited time — especially helpful for high school and college students studying complex subjects like biology or math.

2.4 What not to include?

- Detailed explanations of concepts (kept out for scope)
- Flashcards or quizzes
- Multi-day tracking across sessions (yet)

🧱 SECTION 3: 4-LAYER PROMPT DESIGN

🔹 3.1 INPUT UNDERSTANDING

Prompt:
"Extract the subject, topics (if any), number of days left to revise, and focus areas from the following student input: "{user_input}"

Respond ONLY in this JSON format:

```
{

  "subject": "string",
```

"topics": ["list", "of", "topics"],

"days": int,

"focus": ["list", "of", "focus areas"]

}”

Responsible For:
It extracts structured fields (subject, topics, days, focus) from a student’s natural language input and converts it into JSON for internal state tracking.

Example Input + Output:
Input: "I have 6 days to revise Chemistry, especially Organic and Thermodynamics. I want to focus on NCERT diagrams and reactions."
Output:

{

"subject": "Chemistry",

"topics": ["Organic", "Thermodynamics"],

"days": 6,

"focus": ["NCERT diagrams", "reactions"]

}

🔹 3.2 STATE TRACKER

Prompt:
“(Not an LLM prompt — this is implemented in Python using session_state and JSON file storage.)”

How it remembers:
It stores user-specific data in a global session_state dictionary and persists it to a JSON file under a unique user ID. When the user returns, the agent reloads this file to resume where they left off.

Simulated memory:
Yes — memory is simulated using:

- A Python dictionary to hold current session data
- A file-based system (sessions/{user_id}.json) to retain memory between runs
- No system prompts required because the memory is handled outside the LLM

🔹 3.3 TASK PLANNER

Prompt:
“The student is preparing for {subject}.

Topics: {topics}

Focus areas: {focus}

Days available: {days}

Format the study plan in Markdown with:

- Clear headings and bullet points

- Emojis and motivational tone

- Day-wise tasks for each day

- A friendly message at the end to keep the student motivated”

Internal Steps:

- Receives structured input from the parser.
- Saves input to session memory.
- Passes context to the LLM for generating a study plan.
- Optionally exports output as Markdown and PDF.

Chaining/Branching:
The output from **extract_user_intent()** feeds into **update_session_state()**, which then feeds into **generate_plan()** and optionally into the export functions.
Each function handles a clear layer of the agent's flow: parsing → memory → planning → output.

Branching is handled in the CLI through a menu-based interface that supports:

- Editing specific session fields
- Resetting sessions
- Exporting in different formats (Markdown or PDF)

This separation of concerns kept the system clean, modular, and easy to extend.

🔹 3.4 OUTPUT GENERATOR

Prompt:
“(Same as task planner, but focused on presentation)

Format the study plan in Markdown with:

- Clear headings and bullet points

- Emojis and motivational tone

- Day-wise tasks for each day

- A friendly message at the end to keep the student motivated”

Output Behavior:

- Clear **Markdown structure** (headers, bold text, bullet points)
- Encouraging, coach-like tone
- Plan readability across terminals and export formats
- Emojis to make the plan feel warm and personalized

Special Behavior:

Yes — the system supports exporting output to:

- Markdown export via .md file
- PDF export via markdown2 + pdfkit

🔍 SECTION 4: CHATGPT EXPLORATION LOG

Attempt #	Prompt Variant	What Happened	What You Changed	Why You Changed It
1	Used OpenAI API	Hit deprecation + quota errors	Switched to Groq API + Mixtral	Avoid API cost + errors
2	Used Mixtral with Groq	Model was decommissioned	Switched to LLaMA3	Stay current with supported model
3	Plain text output	Looked dull and hard to follow	Added markdown + emojis	Improve readability + engagement
4	Asked vague input ("Help me study")	Agent confused, generated generic response	Added fallback message to clarify input	Help users rephrase their query
5	No session memory across runs	Every restart reset the data	Added JSON-based persistent session using user ID	Enable multi-session continuity and personal study history

---

🧪 SECTION 5: OUTPUT TESTS

Test 1: Normal Input
Input: "I have 5 days to study math. Focus on calculus and probability."
Output:

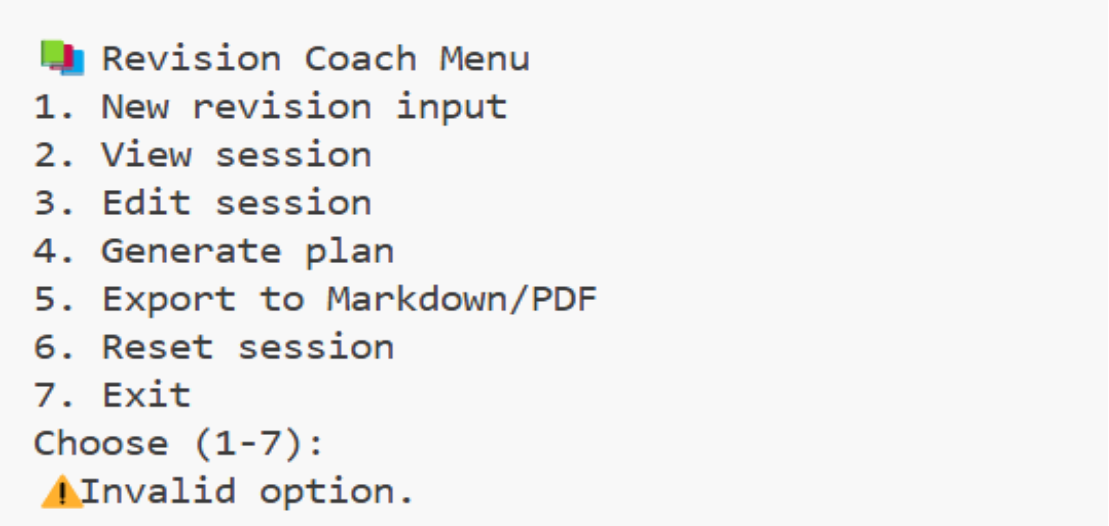
```
🐥 Enter your name to start or continue your session.
Your Name: Ritesh

📚 Revision Coach Menu
1. New revision input
2. View session
3. Edit session
4. Generate plan
5. Export to Markdown/PDF
6. Reset session
7. Exit
Choose (1-7): 1
Describe your revision goal: I have 5 days to study math. Focus on calculus and probability

📚 Revision Coach Menu
1. New revision input
2. View session
3. Edit session
4. Generate plan
5. Export to Markdown/PDF
6. Reset session
7. Exit
Choose (1-7): 4
📝 Your Revision Plan:

**Study Plan for 5 Days 🗓️**

**Calculus & Probability Study Schedule 🤓**

### Day 1: Calculus Warm-up 🌟

* Review basic calculus concepts: functions, limits, derivatives, and integrals 📝
* Go through example problems to refresh your memory 📊
* Take a 15-minute break to stretch and refuel 💪

### Day 2: Exploration of Probability 🎲

* Learn the basics of probability theory: definitions, rules, and formulas 🔮
* Practice calculating probabilities using real-world examples 👩‍🎤
* Watch a short video on probability concepts to reinforce learning 📺

### Day 3: Calculus Challenges 👊

* Practice solving calculus problems, focusing on derivatives and integrals 🔍
* Try solving problems that involve optimization, motion, and related rates 🚀
* Take a 10-minute break to stretch and refocus 🤸‍♀️
```

```
### Day 4: Probability Practice 🚀

* Complete a set of practice problems on probability, focusing on conditional probability and independence 👯
* Review and summarize key concepts in your own words 📝
* Take a 15-minute break to relax and recharge 💆‍♀️

### Day 5: Review & Reflection 📊

* Review all concepts learned during the past 4 days 📚
* Reflect on what you're doing well and what needs improvement 🤔
* Set specific goals for what you want to achieve before the next study session 🎯

**Remember, Consistency is Key! 🌟**

You've got this! Stay committed, and you'll be a calculus and probability master in no time. Take breaks, stay hydrated, and believe
in yourself. You're capable of achieving your goals. Keep it up, and you'll be celebrating your progress in no time! 🎉 Good luck, an
d happy studying! 💪

📚 Revision Coach Menu
1. New revision input
2. View session
3. Edit session
4. Generate plan
5. Export to Markdown/PDF
6. Reset session
7. Exit
Choose (1-7):
```

Test 2: Vague Input
Input: "Help me revise science."
Output:

```
📚 Revision Coach Menu
1. New revision input
2. View session
3. Edit session
4. Generate plan
5. Export to Markdown/PDF
6. Reset session
7. Exit
Choose (1-7): 1
Describe your revision goal: Help me revise science

📚 Revision Coach Menu
1. New revision input
2. View session
3. Edit session
4. Generate plan
5. Export to Markdown/PDF
6. Reset session
7. Exit
Choose (1-7): 4
⚠️Session incomplete.
```

Test 3: Empty Input
Input:
Output:



SECTION 6: REFLECTION

6.1 What was the hardest part of this assignment?
The hardest part was dealing with API instability — first OpenAI's model changes, then Groq model retirement. Each time, I had to refactor code, update prompts, and re-test flow. Ensuring the backend could consistently extract structured data from varied inputs while keeping session logic clean was also tricky.

6.2 What part did you enjoy the most?
The most fun part was seeing the AI generate a helpful, supportive study plan with Markdown formatting, emojis, and motivational tone. Watching the system translate vague input into a personalized, well-structured plan felt like real magic — and made the whole experience feel alive..

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

- Add support for refining existing plans based on user feedback.
- Enable tracking progress across multiple subjects and sessions.
- Add reminders or schedule syncing.
- Build a minimal web or mobile UI to expand usability.

6.4 What did you learn about ChatGPT or prompt design?

I learned that prompt design is closer to system design than conversation — clear roles, strict formatting, and intentional phrasing matter a lot. I also learned how to debug LLM output, update API calls under pressure, and think like a pipeline: input → session state → model task → final output.

6.5 Did you ever feel stuck? How did you handle it?
Yes — especially when models were decommissioned or API keys failed without clear messaging. I handled it by digging through docs, experimenting with fallbacks, and switching to Groq's models. Breaking issues into smaller parts and testing iteratively helped me stay focused and find solutions faster.

🧠 SECTION 7: HACK VALUE

Did you go beyond the brief in any way?

Yes, I went beyond the brief in several ways:

- **Simulated multiple users** by building a profile-based session system where each user's progress is saved and retrieved independently via a simple JSON store.
- **Added persistent memory** using file-based sessions so the assistant could remember each user's subject, topics, and focus areas across runs.
- **Extended export logic** with both Markdown and PDF formats, which required integrating external tools like wkhtmltopdf and handling formatting dynamically.
