# 🧠 AI Agent Assignment – Approach Document

## 🧾 SECTION 1: BASIC DETAILS

Name: Krish Lodha

AI Agent Title / Use Case: AI Agent to help students revise for an exam (using Gemini API for quiz generation)

## 🧠 SECTION 2: PROBLEM FRAMING

2.1 What problem does your AI Agent solve?  
It helps students actively revise for upcoming exams by generating subject-specific quizzes and explanations on request.

2.2 Why is this agent useful?  
Students can quickly test their knowledge and fill gaps without manually searching for quizzes or notes. The agent acts like a revision tutor on demand.

2.3 Who is the target user?  
High school or college students preparing for exams in subjects like biology, physics, chemistry, etc.

2.4 What NOT to include?  
Full lecture content or detailed syllabus breakdowns. No long explanations or personalized academic advice.

## 🧱 SECTION 3: 4-LAYER PROMPT DESIGN

### 🔹 3.1 INPUT UNDERSTANDING

Prompt:  
You are an AI exam revision assistant. Extract the user's intent and the subject/topic from this message:  
Message: "{{user\_message}}"  
Return:  
{  
 "intent": "",  
 "subject": "",  
 "topic": "",  
 "question\_type": ""  
}

What is this prompt responsible for?  
Classifying the user's request and identifying subject/topic to trigger the correct response type.

Example Input + Output:  
Input: "Can you quiz me on photosynthesis in biology?"  
Output:  
{  
 "intent": "quiz\_me",  
 "subject": "biology",  
 "topic": "photosynthesis",  
 "question\_type": "quiz"  
}

### 🔹 3.2 STATE TRACKER

Prompt / Logic Description:  
The agent uses a Python class `StateTracker` to store and update the last subject, topic, and revision progress such as past quizzes.

How does this help the agent “remember”?  
It lets the agent maintain continuity across turns (e.g., generating a follow-up question, avoiding repetition).

Simulated Memory Example:  
{  
 "subject": "biology",  
 "topic": "photosynthesis",  
 "progress": {  
 "quizzes\_taken": ["photosynthesis"]  
 },  
 "last\_question\_type": "quiz"  
}

### 🔹 3.3 TASK PLANNER

Prompt or Logic:  
Based on the parsed intent, subject, and topic, the agent selects one of the following actions:  
- generate\_quiz  
- explain\_topic  
- summarize\_topic  
- ask\_followup

What steps does your agent take internally?  
Uses simple if-else logic in Python to decide what action to trigger next. Branching is used based on `intent`.

### 🔹 3.4 OUTPUT GENERATOR

Prompt or Generation Logic:  
If action == "generate\_quiz":  
 return generate\_quiz\_with\_gemini(subject, topic)

What kind of output formatting or tone did you aim for?  
Quiz-style multiple-choice questions with clear options and the correct answer at the end.

Any special behavior?  
Uses the Gemini API (Gemini Pro) to dynamically generate 3–5 MCQs for the chosen topic.

## 🔍 SECTION 4: CHATGPT EXPLORATION LOG

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Prompt Variant | What Happened | What You Changed | Why You Changed It |
| 1 | Generate quiz on biology | Very generic questions | Added topic and grade | To make quiz relevant |
| 2 | Quiz me on photosynthesis in biology | Good result | No changes | Prompt was already clear |
| 3 | Can you test me on Newton's laws? | Output lacked formatting | Added format prompt | To ensure MCQs had options and answers |

## 🧪 SECTION 5: OUTPUT TESTS

Test 1 – Normal Input:  
Input: "Quiz me on Newton's laws in physics"  
Output:  
Q1. Which law explains inertia?  
a) First law  
b) Second law  
c) Third law  
d) Fourth law  
Answer: a

Test 2 – Vague Input:  
Input: "Give me a quiz"  
Output:  
I can generate a quiz! What subject and topic would you like me to focus on?

Test 3 – Invalid Input:  
Input: ""  
Output:  
I'm not sure what you're asking. Could you please provide a subject or topic to quiz you on?

## 🔄 SECTION 6: REFLECTION

6.1 What was the hardest part?  
Deciding how to split responsibilities between prompts and code. Also tuning Gemini to give good multiple-choice formatting.

6.2 What part did you enjoy the most?  
Watching Gemini dynamically generate high-quality quizzes from just a topic prompt.

6.3 If given more time, what would you improve or add?  
Add scoring, explanations for each answer, and memory of past mistakes to help students improve.

6.4 What did you learn about ChatGPT or prompt design?  
That each step (understanding, tracking, planning, output) must be treated as its own mini-function. Modular prompts work best.

6.5 Did you ever feel stuck? How did you handle it?  
Yes, when Gemini responses were too long or freeform. I asked ChatGPT how to make the prompt more structured.

## 🧠 SECTION 7: HACK VALUE (Optional)

Integrated Google Gemini API to generate live quiz content.  
Used a memory class in Python to simulate state tracking across multiple turns.