# AI Agent Architecture Document

## 1. Components

- **Planner Agent:**
  Extracts key concepts from the input context (text or PDF) to determine what questions should be generated.
- **Executor Agent:**
  Generates questions (MCQ or short-answer) and answers for each concept, using a fine-tuned language model.
- **User Interface (CLI):**
  Allows users to input text or PDF files, select question type, and receive generated questions and answers.
- **Data Processing Scripts:**
  Prepare and format datasets (RACE, SQuAD) for training and evaluation.

## 2. Interaction Flow

1. **User Input:**
   User provides a context (text or PDF) and selects the type and number of questions to generate.
2. **Planning:**
   The planner agent analyzes the context and extracts the most important concepts.
3. **Execution:**
   For each concept, the executor agent generates a question (MCQ or short-answer) and its answer.
4. **Output:**
   The system displays the generated questions and answers to the user.

## 3. Models Used

- **Base Model:**
  Why I Used This Model
  I chose distilgpt2 as the base model for my project primarily due to memory and technical constraints. As a distilled version of GPT-2, distilgpt2 offers a significantly smaller model size and lower resource requirements, making it ideal for environments with limited RAM and storage. This allowed me to efficiently fine-tune and deploy the model on my local machine without the need for specialized hardware or cloud resources.

Additionally, by combining distilgpt2 with LoRA (Low-Rank Adaptation) via the PEFT library, I was able to further reduce the number of trainable parameters and storage overhead. This parameter-efficient approach enabled me to adapt the model for the specific task of exam question generation while staying within strict technical limits.

Overall, this choice ensured that my agent remained lightweight, fast, and practical for real-world use, while still delivering reliable and contextually appropriate outputs.

- **Fine-Tuning:**
  LoRA (Low-Rank Adaptation) via the PEFT library, targeting GPT-2's attention and projection layers for parameter-efficient adaptation.

- **Pipeline:**
  The pipeline of this project begins with the user providing input text or a PDF, which is processed to extract key concepts using the planner agent. For each concept, the executor agent—powered by a fine-tuned, memory-efficient distilgpt2 model—generates either a multiple-choice or short-answer question along with its answer. The system leverages prompt engineering and parameter-efficient fine-tuning (LoRA) to ensure outputs are relevant and well-structured. Finally, the generated questions and answers are presented to the user via a simple command-line interface, enabling fast, reliable exam question creation from any study material.

# 4. Reasons for Choices

- **distilgpt2:**
  Selected for its small size, fast inference, and suitability for resource-constrained environments.
- **LoRA/PEFT:**
  Enables efficient fine-tuning with minimal additional parameters, reducing storage and memory requirements.
- **Planner + Executor (Multi-Agent):**
  Improves modularity and allows for more controllable and interpretable question generation.
- **CLI Interface:**
  Ensures broad compatibility and ease of use without requiring a web or desktop GUI.
- **Dataset Choices (RACE, SQuAD):**
  Provide a mix of MCQ and short-answer formats, ensuring the model can handle both types of questions.

# 5. Why This Fine-Tuning Target Was Chosen

I selected `distilgpt2` as the base model for fine-tuning because it is a lightweight, efficient version of GPT-2, making it well-suited for fast inference and deployment in resource-constrained environments. Fine-tuning with LoRA (Low-Rank Adaptation) via the PEFT library allows for parameter-

efficient adaptation, enabling the model to specialize in generating exam-style questions (both MCQ and short-answer) without requiring large amounts of additional storage or compute.

This approach provides:

- **Task Specialization:** The model is adapted specifically for question generation using real exam datasets (RACE and SQuAD), improving its ability to generate relevant, high-quality questions and answers.
- **Improved Reliability:** Fine-tuning on a diverse, well-structured dataset ensures the model produces more accurate and contextually appropriate outputs compared to a generic, pre-trained model.
- **Adapted Style:** By training on actual exam questions, the model learns the expected format, language, and structure, resulting in outputs that closely match real-world requirements for educational assessment.

In summary, this fine-tuning target was chosen to balance efficiency, reliability, and output quality for the specific task of automated question generation.

# 6. Assignment Criteria Fulfillment

## Core Features (Mandatory)

- **Automate a manual task:**
  Automated the task of generating exam questions from study material.
- **At least one fine-tuned model:**
  Used distilgpt2 fine-tuned with LoRA/PEFT.
- **Explain fine-tuning target:**
  Chose distilgpt2 for efficiency and LoRA for parameter-efficient adaptation (see above).
- **Evaluation metrics:**
  Designed and implemented human and rule-based evaluation (see evaluation report).

## Optional Features (Bonus Points)

- **Multi-agent collaboration:**
  Implemented (Planner + Executor).
- **User interface:**
  Implemented as a CLI for broad compatibility.

---

This architecture fulfills all core requirements and several bonus criteria, providing a modular, efficient, and practical AI agent for question generation.