AI Agent Architecture Document
Project: Course Suggestor AI Agent

1 Overview

This document outlines the architecture of the AI agent designed to automate the task of finding and recommending university courses based on natural language queries. The architecture follows a Planner-Tool-Executor (or "RAG") pattern, which leverages a fine-tuned model to call external tools and a base model to synthesize the final answer.

2 System Diagram

The data flows through the system as follows:

```
[User Prompt] -> [Agent (Planner)] -> [JSON Query] -> [Database Tool
(RAG)] -> [Course Data] -> [Agent (Executor)] -> [Final Answer]
```

3 Component Breakdown

1. **The Planner (Fine-Tuned Model)**
   - **Model:** `microsoft/Phi-3-mini-4k-instruct` (fine-tuned with QLoRA)
   - **Task:** This is the mandatory fine-tuned model. Its sole purpose is Task Specialization.
   - **Rationale:** A base LLM cannot reliably generate the specific JSON query format needed by our database. We fine-tuned this model on 100 examples to specialize it for one task: translating a user's prompt (e.g., "find me an easy class") into a machine-readable JSON query (e.g., `{"Branch": "CE", "sortBy": "AGP", "sortOrder": "desc"}`). This dramatically improves the reliability of the agent.
   - **Output:** A JSON string (e.g., `{"Branch": "CE", "AGP_gte": 8.0}`).

2. **The Database Tool (RAG Integration)**
   - **Type:** A custom Python function (`query_course_database`). This is an External Tool Integration.
   - **Knowledge Base:** `final_database.json` (a manually compiled database from schedule and grading PDFs).

- **Task:** This tool acts as the Retrieval-Augmented Generation (RAG) component. It receives the JSON query from the Planner, filters the 66 courses in the database, and retrieves the top 5 matching results.
- **Output:** A JSON list of course data.

3. **The Executor (Base Model)**
   - **Model:** The base `microsoft/Phi-3-mini-4k-instruct` model.
   - **Task:** This component fulfills the Multi-agent collaboration role. It receives the user's original prompt and the data retrieved by the tool. Its job is to synthesize this information into a polite, helpful, and natural-language answer for the user.
   - **Output:** The final chat response.