**AI Agents creation**

Building Agents from scratch.

I decided to use the ReAct Pattern(Reasoning + Acting) in this particular project.

In this pattern, the LLM first thinks about what to do and then decides on an action to take.

* An action is executed in an environment, and then decides on an action to take.

Day 1: July 9, 2024

1. Watched numerous tutorials, and read articles on LangChain and LangGraph for inspiration.
2. Being that fitness plays an important part of my life, I decided to create a “fitness inspired” application.
3. I created three AI agents: Workout Planner, Nutrition Advisor, Progress Tracker.

Day 2: July 11, 2024

1. Created a folder(**AI\_Fitness\_Agents**) and repository and initialized folder to Github.
2. Installed langgraph and langchain in vs code.
3. Created 4 files, one for each agent and of course a main.py file in order to run the application in main\_workflow.py.

Day 3: July 13, 2024

1. In each agent file, I created classes. The **WorkoutPlanner class** will be responsible for creating a workout plan, based on user **goal** and **experience level.**
2. The **NutritionAdvisor class** is responsible for creating a nutrition plan, based on **goal**, **dietary restrictions**, and **meal preferences**.
3. And finally, the **ProgressTracker class** is responsible for **goal**, **completed workouts**, **dietary adherence**, and *user’s* weekly summary.
4. “Ran” this terminal to ensure all working properly. I ran into an error message, but realized it was a simple issue dealing with a directory path.

Day 4: July 14, 2024

1. I wanted to create simple, yet interesting and effective prompts for each agent. As mentioned before, the workout\_planner prompt is asked to create a personalized workout plan according to user’s experience (ie. beginner, intermediate, etc). The nutrition\_advisor prompt is asked to create a dietary plan according to the user’s workout plan. The progress\_tracker prompt will track the user’s progress, depending on 4 categories.
2. Created said prompts: []
3. Continued to watch tutorials and additional text reading for continued inspiration and learning.
4. Updated main\_workflow.py. Now Each agent now uses prompts to generate responses based on user preferences.

Day 5: July 15, 2024

1. Retrieved both OPENAI\_API key as well as TAVILY\_API key. Created .env file.
2. Continuous debugging regarding API\_KEY retrieval, as well as issues regarding scripting prompts.

Day 6: July 16, 2024

1. Continuously updating script. I’m not sure why, but falling into one error after another regarding imports and prompting.