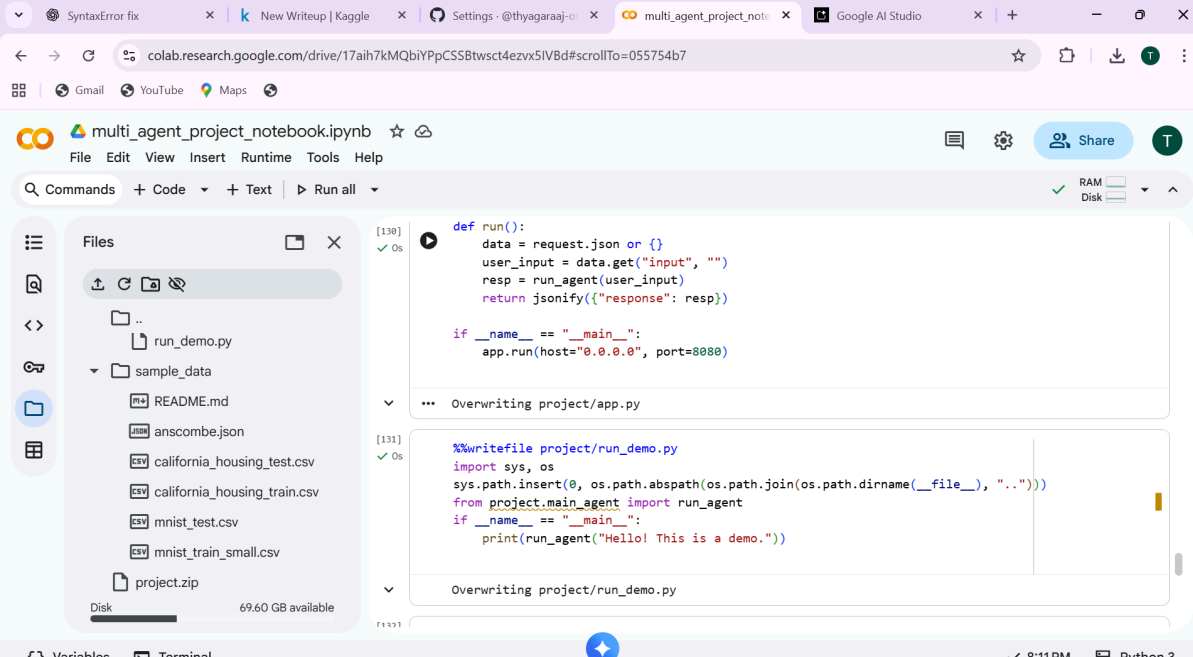


Kaggle Capstone Project Ideas – Diet & Nutrition Assistant Agents evidence

Part 1:



The screenshot shows a Google Colab notebook interface. The browser tabs include 'SyntaxError fix', 'New Writup | Kaggle', 'Settings - @thyagaraaj-o', 'multi_agent_project_note', and 'Google AI Studio'. The notebook title is 'multi_agent_project_notebook.ipynb'. The left sidebar shows a file explorer with a folder named 'sample_data' containing files like 'README.md', 'anscombe.json', 'california_housing_test.csv', 'california_housing_train.csv', 'mnist_test.csv', 'mnist_train_small.csv', and 'project.zip'. The main code area shows two code cells. The first cell (execution [130], 0s) defines a 'run()' function that takes a request, processes it through 'run_agent', and returns a JSON response. It also includes a main block to run the app on host '0.0.0.0' and port '8080'. The second cell (execution [131], 0s) overwrites 'project/app.py' and 'project/run_demo.py' with new code. The 'run_demo.py' code imports 'sys' and 'os', and defines a 'run_agent' function that prints a demo message. The bottom status bar shows '8:11 PM' and 'Python 3'.

```
def run():
    data = request.json or {}
    user_input = data.get("input", "")
    resp = run_agent(user_input)
    return jsonify({"response": resp})

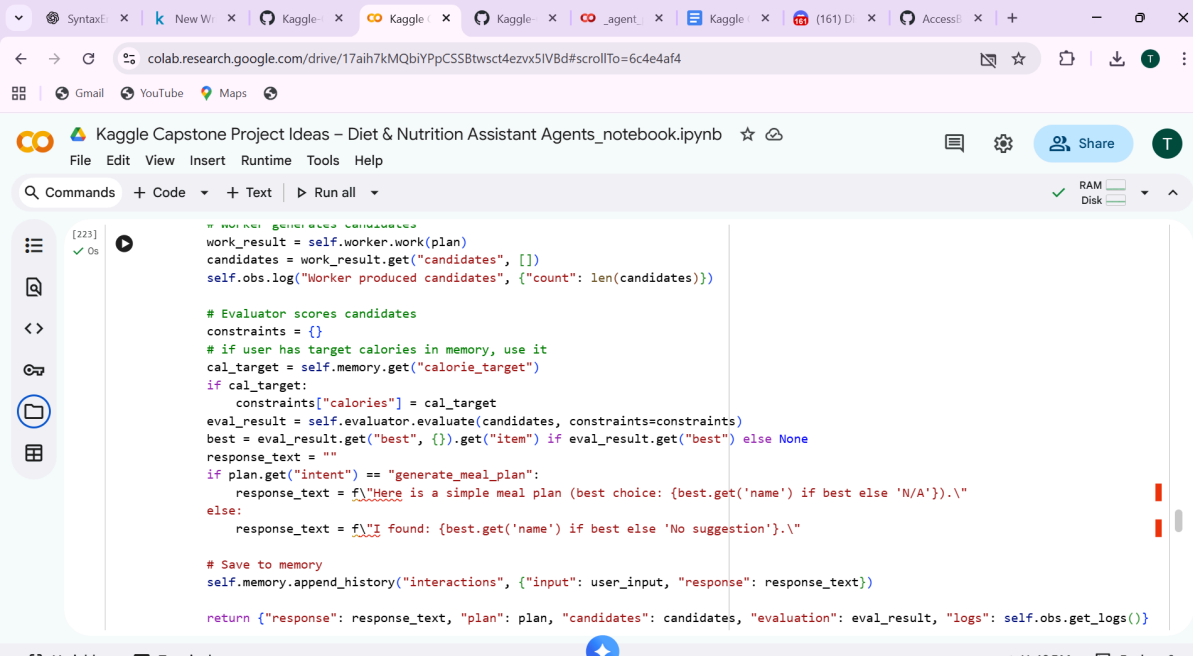
if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8080)
```

```
Overwriting project/app.py
```

```
Overwriting project/run_demo.py
```

```
%%writefile project/run_demo.py
import sys, os
sys.path.insert(0, os.path.abspath(os.path.join(os.path.dirname(__file__), "..")))
from project.main_agent import run_agent
if __name__ == "__main__":
    print(run_agent("Hello! This is a demo."))
```

Part 2 Diet & nutrition agents:



The screenshot shows a Google Colab notebook interface. The browser tabs include 'SyntaxE...', 'New Wi...', 'Kaggle -', 'Kaggle -', 'Kaggle -', 'Kaggle -', '(161) D...', 'Access...', and '+'. The notebook title is 'Kaggle Capstone Project Ideas – Diet & Nutrition Assistant Agents_notebook.ipynb'. The left sidebar shows a file explorer with a folder named 'sample_data' containing files like 'README.md', 'anscombe.json', 'california_housing_test.csv', 'california_housing_train.csv', 'mnist_test.csv', 'mnist_train_small.csv', and 'project.zip'. The main code area shows a single code cell (execution [223], 0s) containing Python code for a diet and nutrition agent. The code defines a 'Worker' class with methods 'work', 'evaluate', and 'generate_meal_plan'. The 'work' method generates candidates and returns them. The 'evaluate' method evaluates candidates based on constraints and returns the best result. The 'generate_meal_plan' method generates a meal plan based on the best result and returns it. The code also includes a main block to run the agent on host '0.0.0.0' and port '8080'. The bottom status bar shows '11:42 PM' and 'Python 3'.

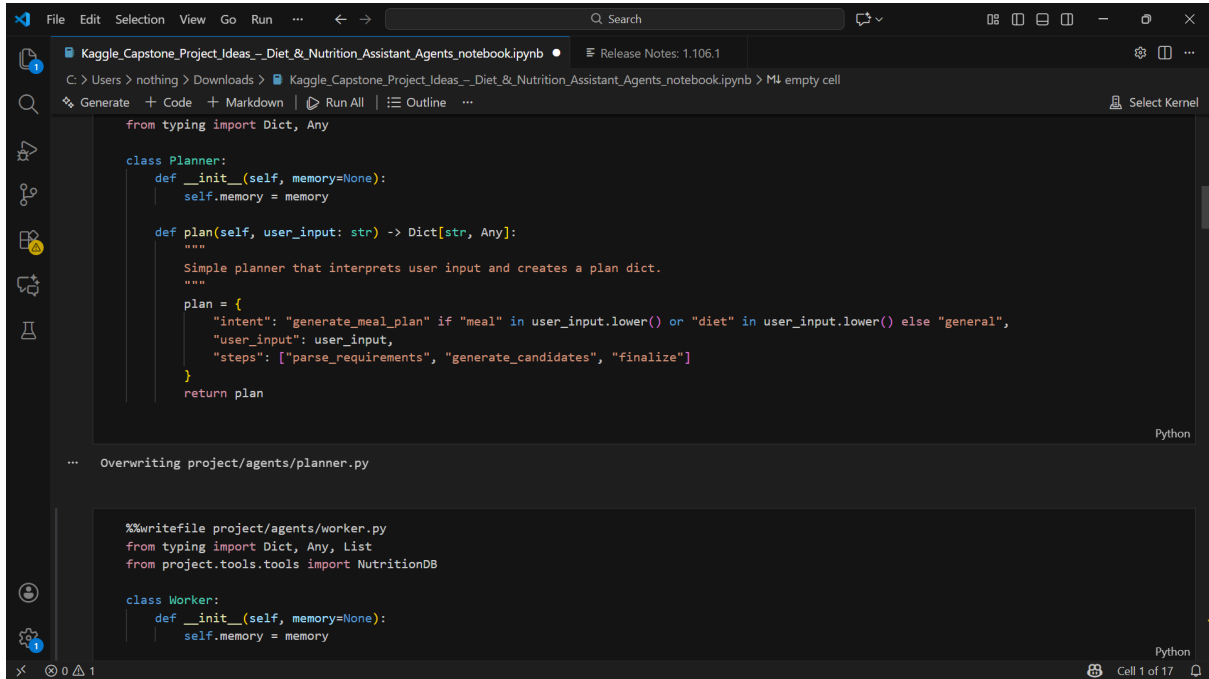
```
work_result = self.worker.work(plan)
candidates = work_result.get("candidates", [])
self.obs.log("Worker produced candidates", {"count": len(candidates)})

# Evaluator scores candidates
constraints = {}
# if user has target calories in memory, use it
cal_target = self.memory.get("calorie_target")
if cal_target:
    constraints["calories"] = cal_target
eval_result = self.evaluator.evaluate(candidates, constraints=constraints)
best = eval_result.get("best", {}).get("item") if eval_result.get("best") else None
response_text = ""
if plan.get("intent") == "generate_meal_plan":
    response_text = f"Here is a simple meal plan (best choice: {best.get('name') if best else 'N/A'})."
else:
    response_text = f"I found: {best.get('name') if best else 'No suggestion'}."

# Save to memory
self.memory.append_history("interactions", {"input": user_input, "response": response_text})

return {"response": response_text, "plan": plan, "candidates": candidates, "evaluation": eval_result, "logs": self.obs.get_logs()}
```

Visual studio using of the jupyter server and py



The image shows a Visual Studio Code window with a Jupyter notebook open. The notebook is titled "Kaggle_Capstone_Project_Ideas_-_Diet_& Nutrition_Assistant_Agents_notebook.ipynb". The code in the notebook is as follows:

```
from typing import Dict, Any

class Planner:
    def __init__(self, memory=None):
        self.memory = memory

    def plan(self, user_input: str) -> Dict[str, Any]:
        """
        Simple planner that interprets user input and creates a plan dict.
        """
        plan = {
            "intent": "generate_meal_plan" if "meal" in user_input.lower() or "diet" in user_input.lower() else "general",
            "user_input": user_input,
            "steps": ["parse_requirements", "generate_candidates", "finalize"]
        }
        return plan
```

Below the code, there is a message: "Overwriting project/agents/planner.py".

At the bottom of the notebook, there is another code cell with the following code:

```
%%writefile project/agents/worker.py
from typing import Dict, Any, List
from project.tools.tools import NutritionDB

class Worker:
    def __init__(self, memory=None):
        self.memory = memory
```

The status bar at the bottom indicates "Cell 1 of 17".