Problem Statement: Build a Dockerized FastAPI Backend for Train Schedule Management

📝 Context

You are provided with:

trains.json – Schedules of all trains including their stations, arrival/departure times, and days of operation.

stations.json – Schedule information indexed by station.

---

✅ Functional Requirements

📂 Train APIs

Get All Trains

GET /trains

→ Returns all train IDs with brief schedule info.

Get Train Schedule

GET /trains/{train\_id}

→ Returns the full schedule for a specific train.

➕ Add New Train

POST /trains

→ Adds a new train and its schedule.

Request body:

{

"train\_id": "12345",

"schedule": {

"StationA": {

"arrival": "08:00",

"departure": "08:10",

"day": "1",

"days": ["M", "W", "F"]

},

"StationB": {

"arrival": "12:00",

"departure": "--",

"day": "1",

"days": ["M", "W", "F"]

}

}

}

---

🏢 Station APIs

Get All Stations

GET /stations

→ Returns all station names.

Get Station Schedule

GET /stations/{station\_name}

→ Returns schedule for all trains stopping at the given station.

➕ Add or Update Station Schedule

POST /stations/{station\_name}

→ Adds a train to the station’s schedule.

Request body:

{

"train\_id": "12345",

"arrival": "08:00",

"departure": "08:10",

"days": ["M", "W", "F"]

}

---

🔎 Search APIs

Find Trains Between Stations

GET /search?from=StationA&to=StationB

→ Returns trains that go from StationA to StationB in the correct order with timings and operational days.

---

📦 Dockerization Requirements

Create a Dockerfile:

Based on python:3.11

Install FastAPI + Uvicorn

Start the server on port 8000

Optional: docker-compose.yml for dev setup

---

🧠 Data Handling

Data will be loaded from JSON into memory (dicts).

New data added via POST should be stored in memory (no persistence required).

---

⚙️ Tech Stack

FastAPI backend

Dockerized application

In-memory data (no database)