Step 1: Set Up Your Environment

1. **Install FastAPI and Uvicorn**

Make sure you have Python installed. You can install FastAPI and Uvicorn (ASGI server) using pip:

pip install fastapi uvicorn

1. **Create a Project Directory**

Create a directory for your FastAPI project:

mkdir fastapi_example

cd fastapi_example

2. **Create a Python File**

Create a file named main.py inside your project directory:

touch main.py

Step 2: Develop the FastAPI Application

Open main.py and add the following code:

```
from fastapi import FastAPI


app = FastAPI()


@app.get("/")
def read_root():
    return {"message": "Welcome to the FastAPI application!"}


@app.get("/hello")
def read_hello():
    return {"message": "Hello, World!"}
```

```
@app.get("/items/{item_id}")
def read_item(item_id: int, q: str = None):
    return {"item_id": item_id, "q": q}

@app.post("/items/")
def create_item(item: dict):
    return {"received_item": item}

@app.put("/items/{item_id}")
def update_item(item_id: int, item: dict):
    return {"item_id": item_id, "updated_item": item}

@app.delete("/items/{item_id}")
def delete_item(item_id: int):
    return {"message": f"Item with ID {item_id} deleted"}
```

Step 3: Run the FastAPI Application

Use Uvicorn to run the FastAPI application:

```
uvicorn main:app --reload
```

Here's what each part of the command means:

- main: Refers to the Python file (main.py).
- app: Refers to the FastAPI instance inside main.py.
- --reload: Automatically reloads the server on code changes (useful during development).

Step 4: Test the Endpoints

Once the server is running, you can test the endpoints locally by navigating to http://127.0.0.1:8000 in your web browser or using tools like curl or Postman.

- **Home Route:**

    - URL: http://127.0.0.1:8000/
    - Response: {"message": "Welcome to the FastAPI application!"}
- **Hello Route:**
    - URL: http://127.0.0.1:8000/hello
    - Response: {"message": "Hello, World!"}
- **Read Item Route:**
    - URL: http://127.0.0.1:8000/items/{item_id}?q={query}
    - Example: http://127.0.0.1:8000/items/5?q=test
    - Response: {"item_id": 5, "q": "test"}
- **Create Item Route:**
    - Method: POST
    - URL: http://127.0.0.1:8000/items/
    - Body: {"name": "item_name"}
    - Response: {"received_item": {"name": "item_name"}}
- **Update Item Route:**
    - Method: PUT
    - URL: http://127.0.0.1:8000/items/{item_id}
    - Body: {"name": "new_item_name"}
    - Response: {"item_id": {item_id}, "updated_item": {"name": "new_item_name"}}
- **Delete Item Route:**
    - Method: DELETE
    - URL: http://127.0.0.1:8000/items/{item_id}
    - Response: {"message": "Item with ID {item_id} deleted"}

Step 5: Documentation and Interactive API

FastAPI automatically generates interactive API documentation:

- **Swagger UI:** http://127.0.0.1:8000/docs
- **ReDoc:** http://127.0.0.1:8000/redoc

These interfaces allow you to interact with your API directly from your browser.

Summary

You have developed a FastAPI application with 6 services and deployed it locally. The services include basic CRUD operations, and you can test them using the provided endpoints. FastAPI's auto-generated documentation makes it easy to explore and interact with your API.

Feel free to adjust the endpoints and functionality as needed for your specific use case.