How to Build and Document a Simple API Using FastAPI and OpenAPI

1. Introduction

FastAPI is a modern, fast (high-performance) web framework for building APIs with Python. It is designed to be easy to use while ensuring robust performance. One of its key features is automatic OpenAPI and Swagger documentation generation, making it ideal for API development.

Why Use FastAPI?

- **Automatic API Documentation** with Swagger UI and Redoc.
- **Fast Performance** powered by Starlette and Pydantic.
- **Ease of Use** with Python type hints and async support.
- **Built-in Validation** using Pydantic models.

Prerequisites

To follow this guide, ensure you have:

- **Python 3.7+** installed.
- Basic knowledge of REST APIs and Python.
- A terminal or command-line interface (CLI).

2. Setting Up the Environment

Installing FastAPI and Uvicorn

First, install FastAPI and Uvicorn (an ASGI server for running FastAPI applications):

```
pip install fastapi uvicorn
```

Creating a Basic FastAPI App

Create a new file main.py and add the following code:

```
from fastapi import FastAPI


app = FastAPI()


@app.get("/")
def read_root():
    return {"message": "Welcome to FastAPI!"}
Run the application using Uvicorn:
uvicorn main:app --reload
```

Now, open a browser and go to http://127.0.0.1:8000/docs to see the auto-generated Swagger UI.

3. Building a Simple API

Let's extend our API with CRUD operations for a **To-Do List API**.

Defining the Data Model

```
from pydantic import BaseModel

class TodoItem(BaseModel):
    id: int
    title: str
    description: str
    completed: bool
```

Creating Endpoints

```
todo_items = []

@app.post("/todos/")
def create_todo(todo: TodoItem):
    todo_items.append(todo)
    return todo

@app.get("/todos/")
def get_todos():
    return todo_items

@app.get("/todos/{todo_id}")
def get_todo(todo_id: int):
    for todo in todo_items:
        if todo.id == todo_id:
            return todo
    return {"error": "To-do item not found"}
```

4. Automatic API Documentation with OpenAPI

FastAPI automatically generates **Swagger UI** and **Redoc documentation**.

- Visit **http://127.0.0.1:8000/docs** for Swagger UI.
- Visit **http://127.0.0.1:8000/redoc** for Redoc documentation.

Customizing OpenAPI Metadata

```
Modify FastAPI() initialization:

app = FastAPI(
    title="To-Do List API",
    description="A simple API to manage To-Do items using FastAPI",
    version="1.0.0"
)
```

5. Testing & Deploying the API

Testing with Postman or Curl

To test the API:

```
curl -X GET "http://127.0.0.1:8000/todos/" -H "accept:
application/json"
```

Deploying with Uvicorn

Run the following command to deploy:

```
uvicorn main:app --host 0.0.0.0 --port 8000
```

6. Conclusion & Next Steps

- You have successfully built a simple API using **FastAPI**.
- The API is automatically documented using **OpenAPI**.
- Explore more features like authentication, middleware, and database integration.

Next Steps:

- **Deploy to a cloud platform** like AWS, Azure, or Google Cloud.
- **Integrate a database** (PostgreSQL, MongoDB, or SQLite).
- **Enhance security** using OAuth2 or JWT authentication.

By following this guide, you are now equipped to create and document APIs efficiently using FastAPI!