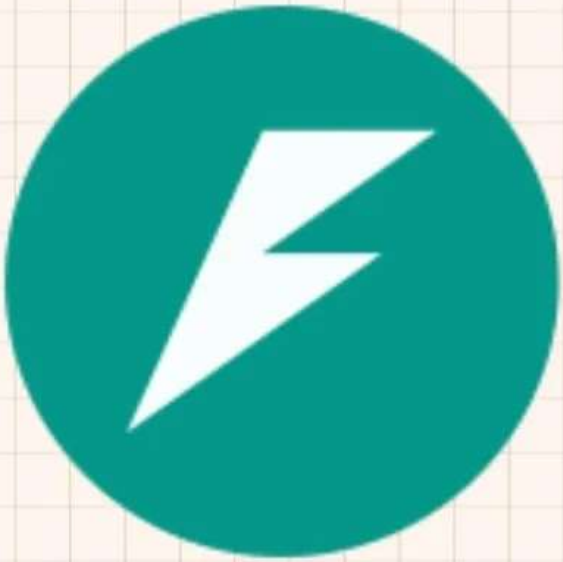
Akash • Python & Tech Enthusiast 🚀
@pycode.hubb

legitpython.com



FASTAPI EXPLAINED

by Pycode.Hubb

python™

NEXT

What is FastAPI?

FastAPI is a **modern, fast (high-performance)** web framework for building APIs with **Python 3.7+**. It's built on top of **Starlette** and uses **Pydantic** for data validation.

It's designed to help you create **RESTful APIs quickly**, with less code and better performance — all while keeping your code **clean and readable**.

Key highlights:

- Asynchronous support out of the box (thanks to async/await)
- Smart data validation using Python type hints
- Auto-generated docs (Swagger UI & ReDoc)
- Ideal for modern web apps, machine learning APIs, and microservices

Akash • Python & Tech Enthusiast 🚀
@pycode.hubb

legitpython.com

Why FastAPI over Flask/Django?

While Flask and Django are powerful, FastAPI brings some **modern features** and **developer-friendly tools** that set it apart:

Performance:

FastAPI is one of the fastest Python frameworks — on par with Node.js and Go, thanks to async support.

Type Hints = Fewer Bugs:

Uses Python type hints to validate data automatically. Less guesswork, more accuracy.

Built-in Docs:

Automatically generates interactive API documentation (Swagger & ReDoc) — no extra setup needed.

Asynchronous by Default:

Built for modern async programming. Makes handling concurrent requests super efficient.

Less Boilerplate, More Code:

Write clean APIs with fewer lines of code — no need for manual validation or decorators everywhere.

How to Install FastAPI + Uvicorn

To get started with FastAPI, you need two things:

- **FastAPI:** the main framework
- **Uvicorn:** the ASGI server to run your app

To get started with FastAPI, you need two things:

```
pip install fastapi uvicorn
```

You can also add **[all]** to include extra features like docs support:

```
pip install fastapi[all] uvicorn
```

- Works with Python 3.7+
- Make sure you're using a virtual environment (recommended)

Basic FastAPI App (Hello World)

Let's create your first FastAPI app in just a few lines!

Create a file named **main.py:**

```python
from fastapi import FastAPI

app = FastAPI()

@app.get("/")
def read_root():
    return {"message": "Hello, FastAPI!"}
```

To run the app using Uvicorn:

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

What this does:

- ***main*** → your filename (main.py)
- ***app*** → your FastAPI instance
- ***--reload*** → enables auto-reload on code change (great for dev)
- **Open** http://127.0.0.1:8000 in your browser
- **Visit** http://127.0.0.1:8000/docs for interactive API docs

Handling GET & POST Requests

FastAPI makes it super easy to handle different HTTP methods like GET and POST.

📌 GET Request Example

```python
@app.get("/greet")
def greet(name: str):
    return {"message": f"Hello, {name}!"}
```

Just visit:

```
http://127.0.0.1:8000/greet?name=Akash
```

📌 POST Request Example

```python
from pydantic import BaseModel

class User(BaseModel):
    name: str
    age: int

@app.post("/create-user")
def create_user(user: User):
    return {"message": f"User {user.name} added!"}
```

Test it in the Swagger UI at:

```
http://127.0.0.1:8000/docs
```

FastAPI automatically:

- Parses JSON request body
- Validates data using Pydantic
- Returns helpful error messages

Using Pydantic for Request Validation

FastAPI uses Pydantic models to validate and structure incoming request data. This means no more manual checks — it's all automatic!

📌 **Define a data model:**

```
from pydantic import BaseModel

class Item(BaseModel):
    name: str
    price: float
    in_stock: bool
```

📌 **Use it in a POST route:**

```
@app.post("/add-item")
def add_item(item: Item):
    return {"message": f"{item.name} added!"}
```

FastAPI will:

- Validate types (str, float, bool, etc.)
- Return automatic 422 errors if data is invalid
- Generate API docs with field details

Example request body:

```
{
  "name": "Notebook",
  "price": 9.99,
  "in_stock": true
}
```

Built-in API Docs (Swagger UI)

One of FastAPI's coolest features is its auto-generated, interactive API docs — with zero setup required!

📌 **Just run your app and visit:**

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

📌 **What you get:**

- Test your endpoints right from the browser
- See required parameters, request/response formats
- Useful for frontend devs, testers, or anyone using your API

FastAPI uses **OpenAPI standard** under the hood to generate these docs based on your code, routes, and Pydantic models.