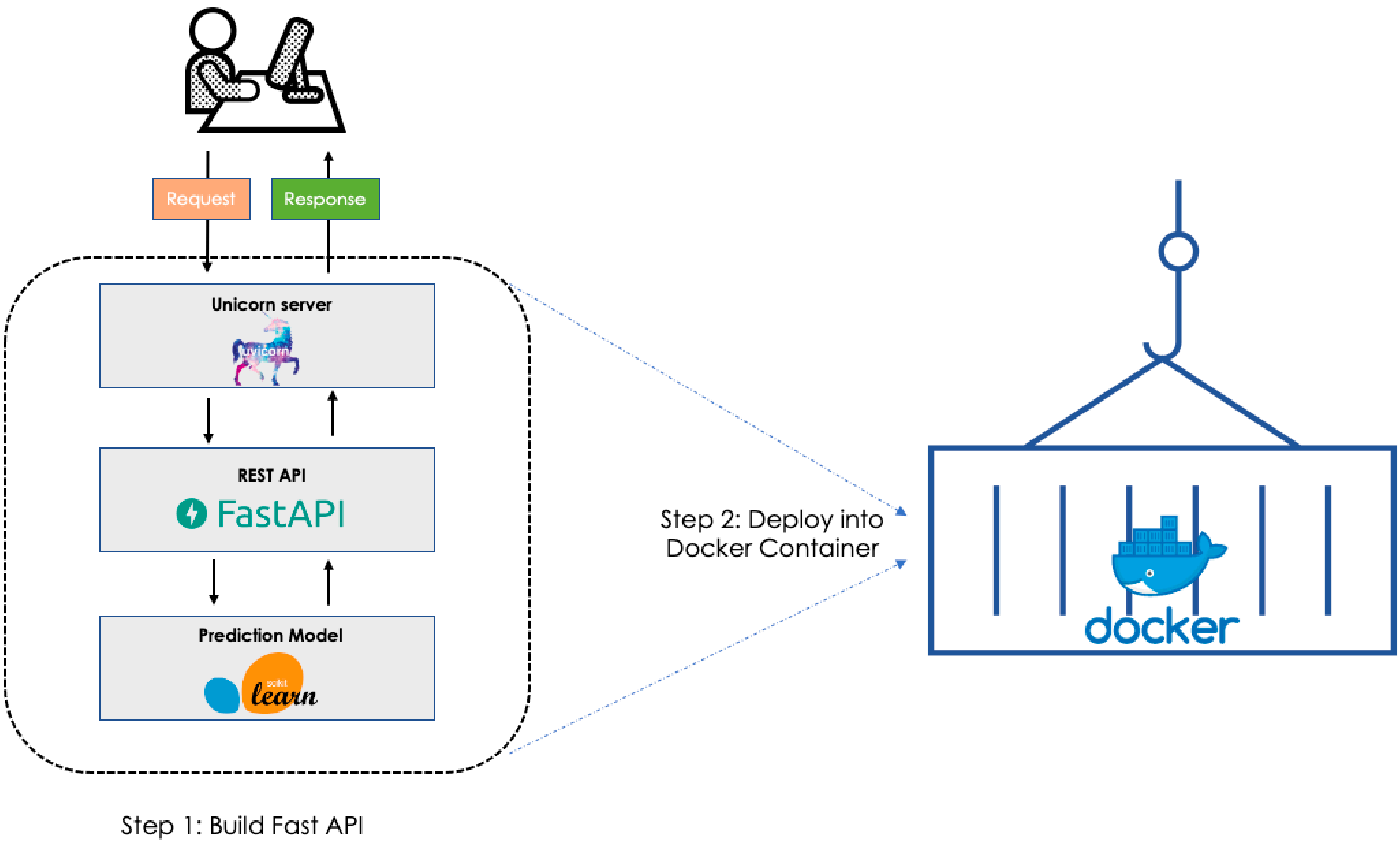
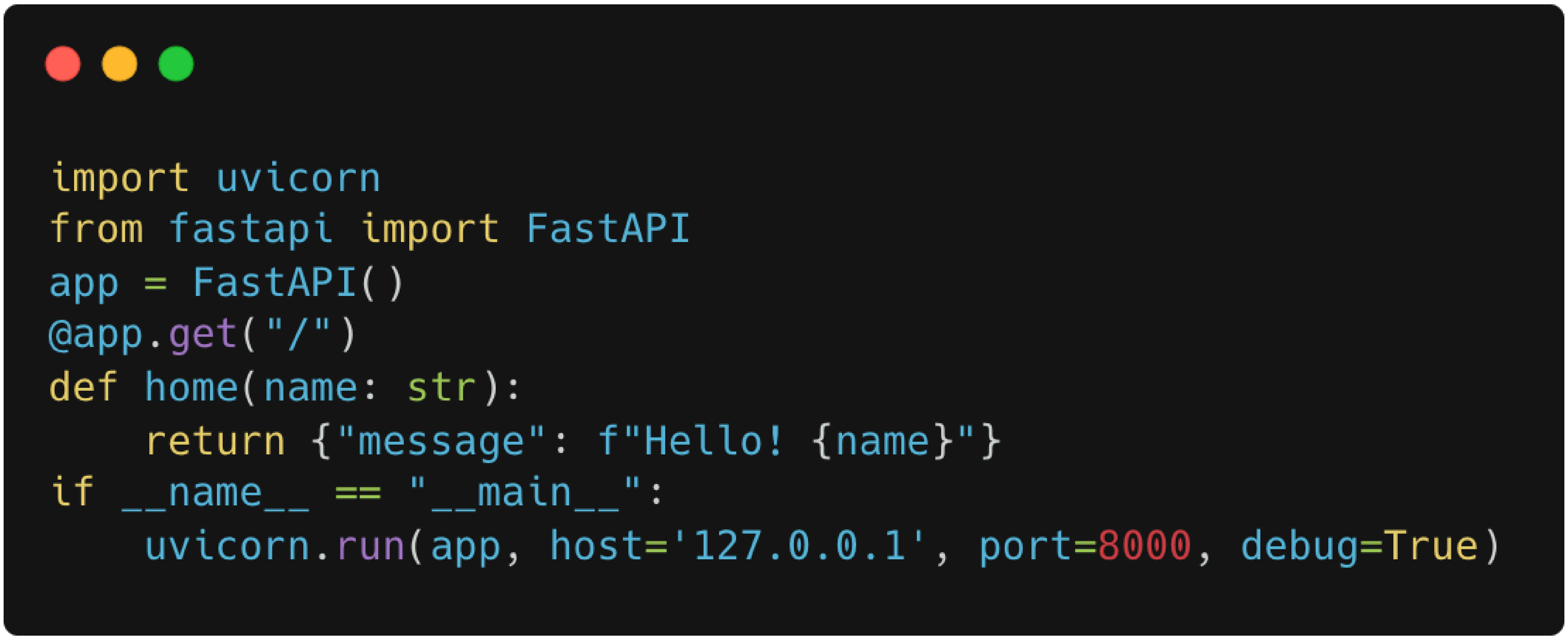
Analytics Vidhya

Model Deployment using FastAPI



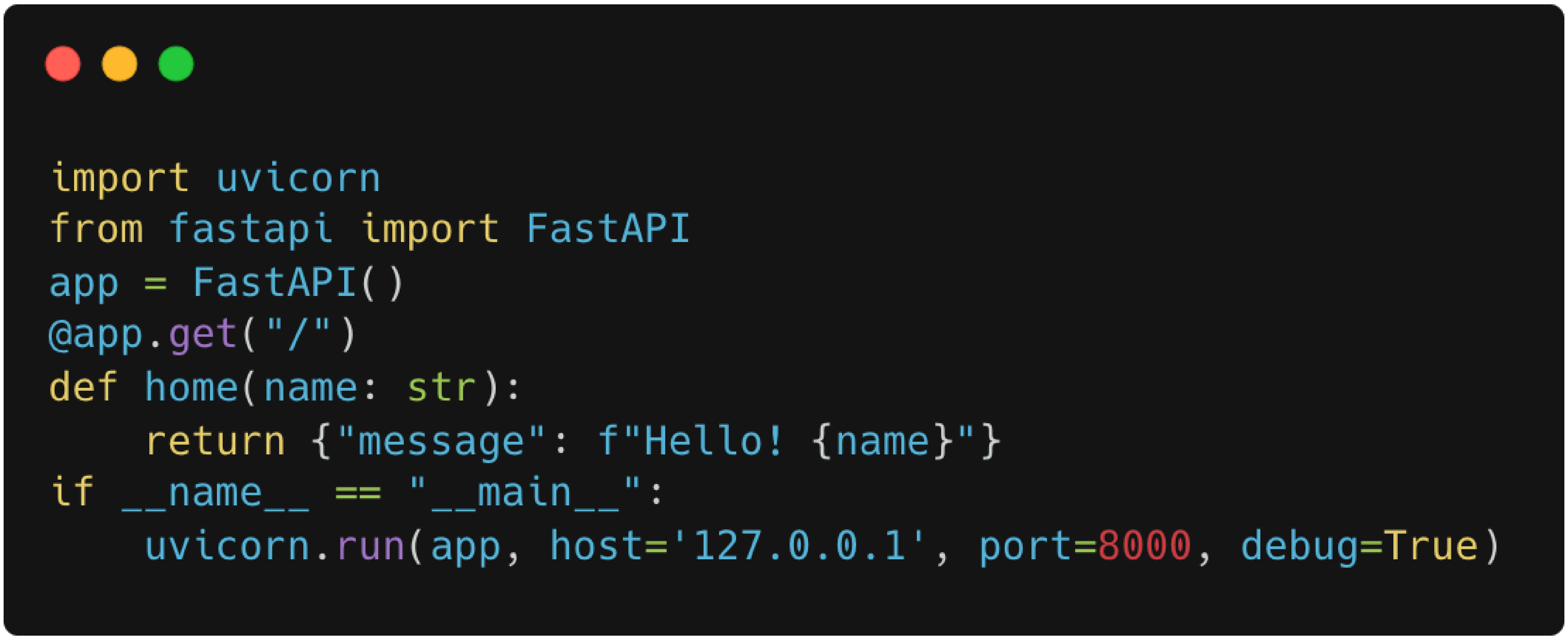


```python
import uvicorn
from fastapi import FastAPI
app = FastAPI()
@app.get("/")
def home(name: str):
    return {"message": f"Hello! {name}"}
if __name__ == "__main__":
    uvicorn.run(app, host='127.0.0.1', port=8000, debug=True)
```

It is a modern framework that allows you to build APIs seamlessly without much effort. It has the ability to separate the server code from the business logic increasing code maintainability. As the name itself has fast in it, it is much faster as compared to the flask because it’s built over ASGI (Asynchronous Server Gateway Interface) instead of WSGI (Web Server Gateway Interface). It has a data validation system that can detect any invalid data type at the runtime and returns the reason for bad inputs to the user in the JSON format only which frees developers from managing this exception explicitly.

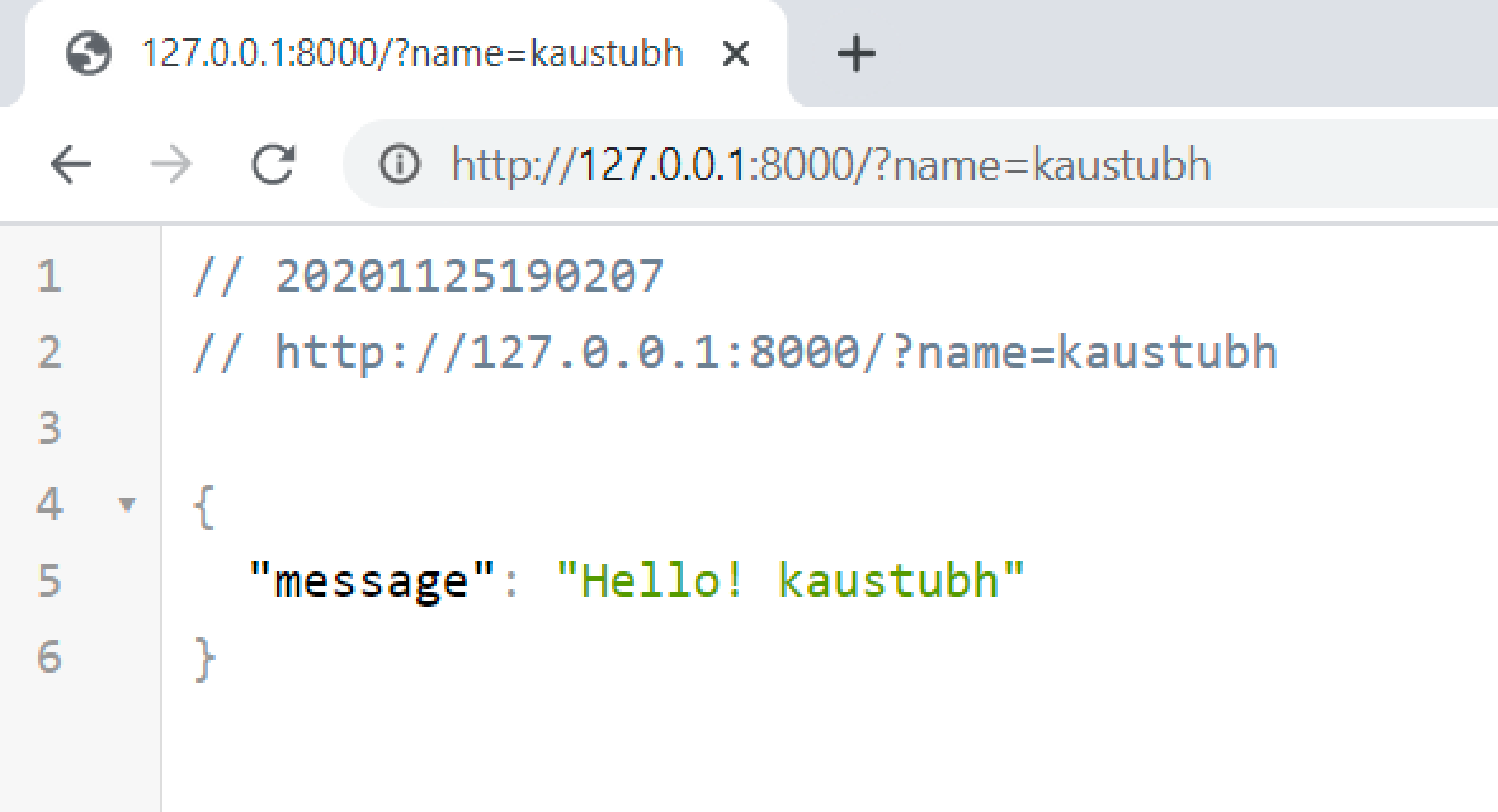
It generates the documentation on the go when you are developing the API which is the most requested thing from all the developers. Documentation is a great way for other developers to collaborate on a project as it presents them with everything that can be done with the necessary instructions. It also generates a nice GUI which solves everything that was missing in the flask.

It does all these things OpenAI specifications and Swagger for implementing these specifications. Being a developer, you are only focusing on the logic building part and the rest of the things are managed by the FastAPI. Let’s look at the same example which was created using Flask now implemented in FastAPI:import uvicorn from fastapi



```
import uvicorn
from fastapi import FastAPI
app = FastAPI()
@app.get("/")
def home(name: str):
    return {"message": f"Hello! {name}"}
if __name__ == "__main__":
    uvicorn.run(app, host='127.0.0.1', port=8000, debug=True)
```

On hitting the URL localhost/?name=AnyNameHere, you will be prompted with output such as:


127.0.0.1:8000/?name=kaustubh
http://127.0.0.1:8000/?name=kaustubh

```
// 20201125190207
// http://127.0.0.1:8000/?name=kaustubh

{
  "message": "Hello! kaustubh"
}
```

You can see that the code is very similar to flask but here we are using uvicorn server which is an ASGI implementation. Also, here we are not routing any endpoints and creating them directly using decorators which makes more sense. The function here simply takes the arguments required further which eliminates the need for the request object to be called.