AIMS
AMMI

# Lab 04: Serving Models with REST

The MLOps Course

25. September 2025

# Table of Contents

# Batch vs Real-time

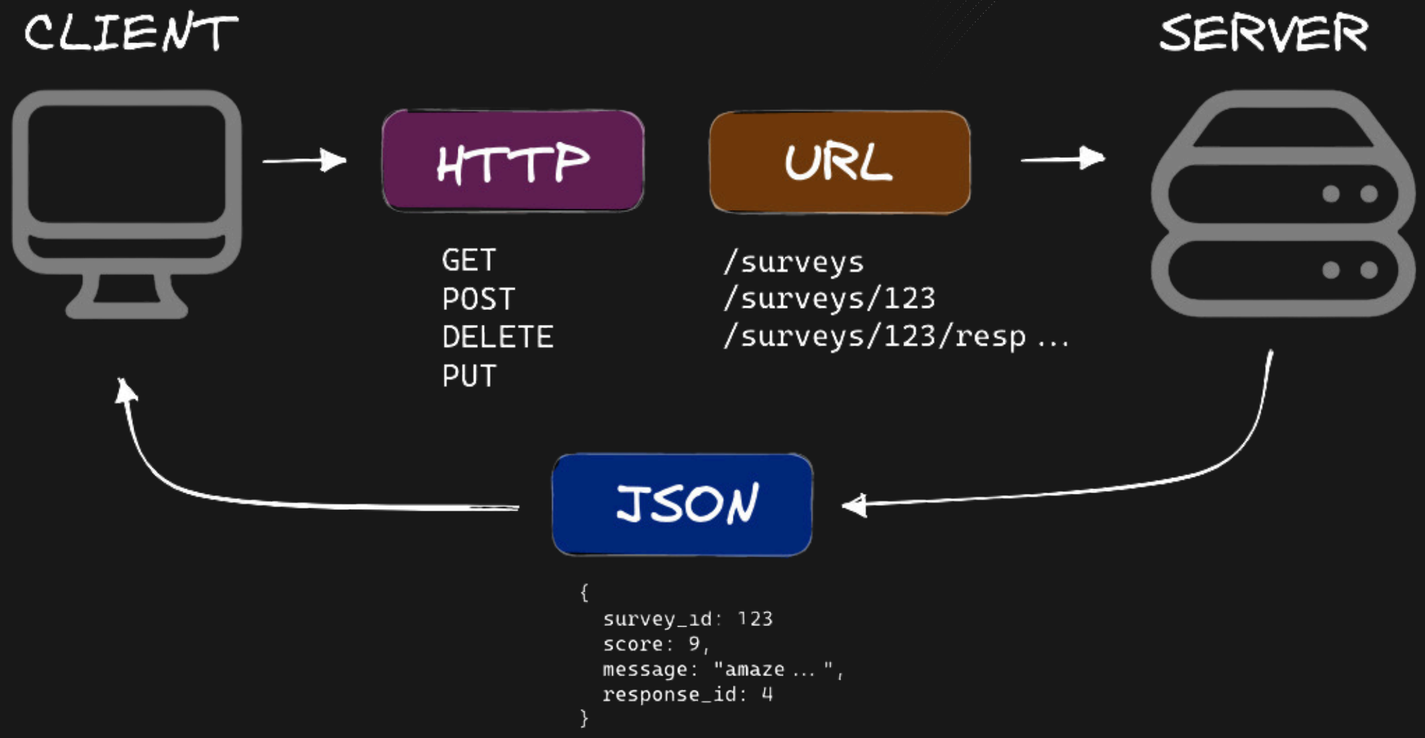
python Copy

```python
import pandas as pd
from sklearn.externals import joblib

# Load pre-trained model
model = joblib.load('model.pkl')

# Batch data for inference
data = pd.read_csv('new_data.csv')

# Batch prediction
predictions = model.predict(data)
```

python Copy

```python
from fastapi import FastAPI
import joblib

app = FastAPI()
model = joblib.load('model.pkl')

@app.post("/predict")
def predict(data: dict):
    input_data = [data['feature1'], data['feature2']]
    prediction = model.predict([input_data])
    return {"prediction": prediction[0]}
```

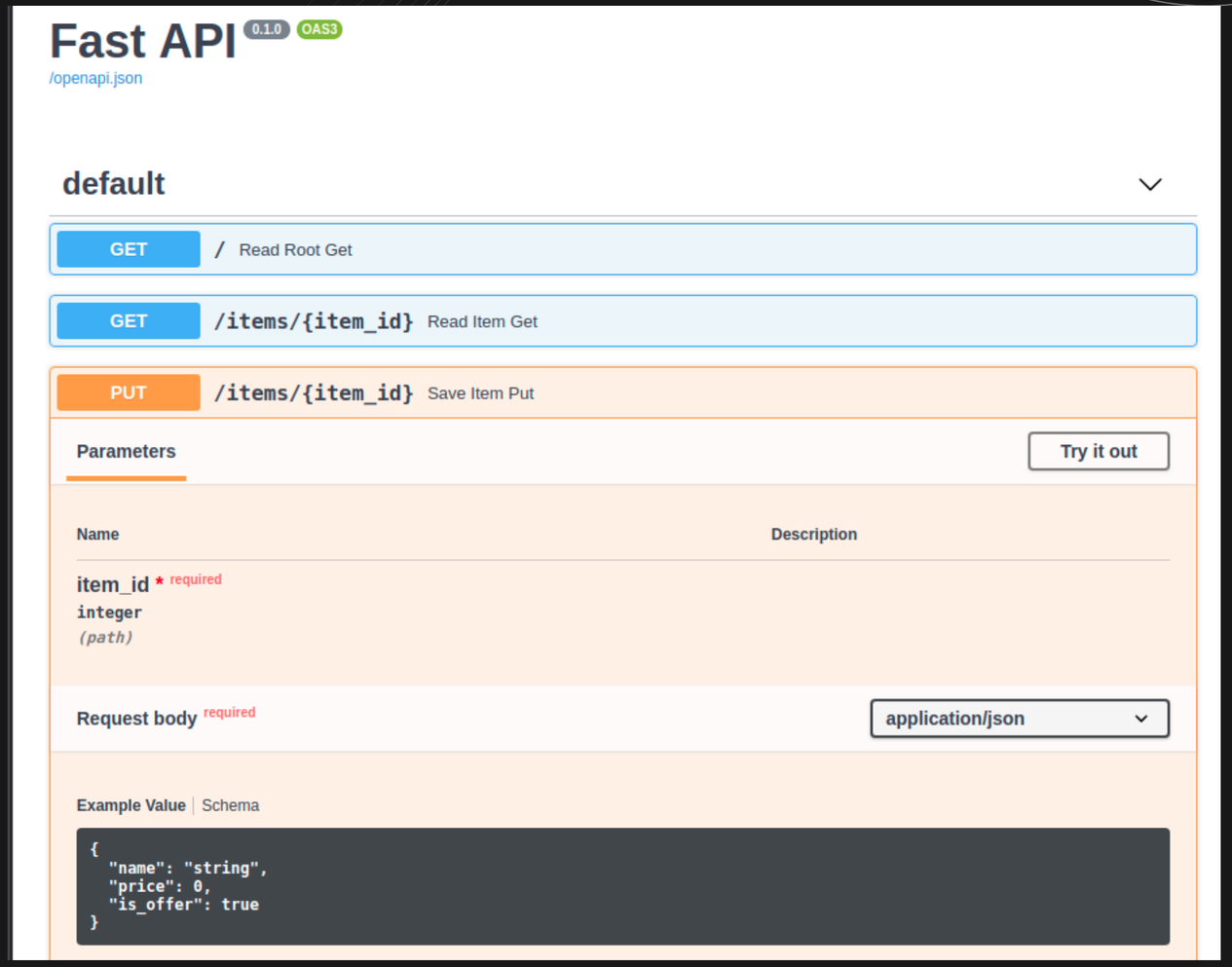
# Communication standard



# Fast API

- framework for building APIs with Python
- high performance
- fast to code
- Pydantic features

```python
from fastapi import FastAPI
from pydantic import BaseModel

app = FastAPI()


class Item(BaseModel):
    name: str
    price: float
    is_offer: bool = None


@app.get("/")
def read_root():
    return {"Hello": "World"}


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


@app.put("/items/{item_id}")
def save_item(item_id: int, item: Item):
    return {"item_name": item.pr, "item_id": item_id}
```

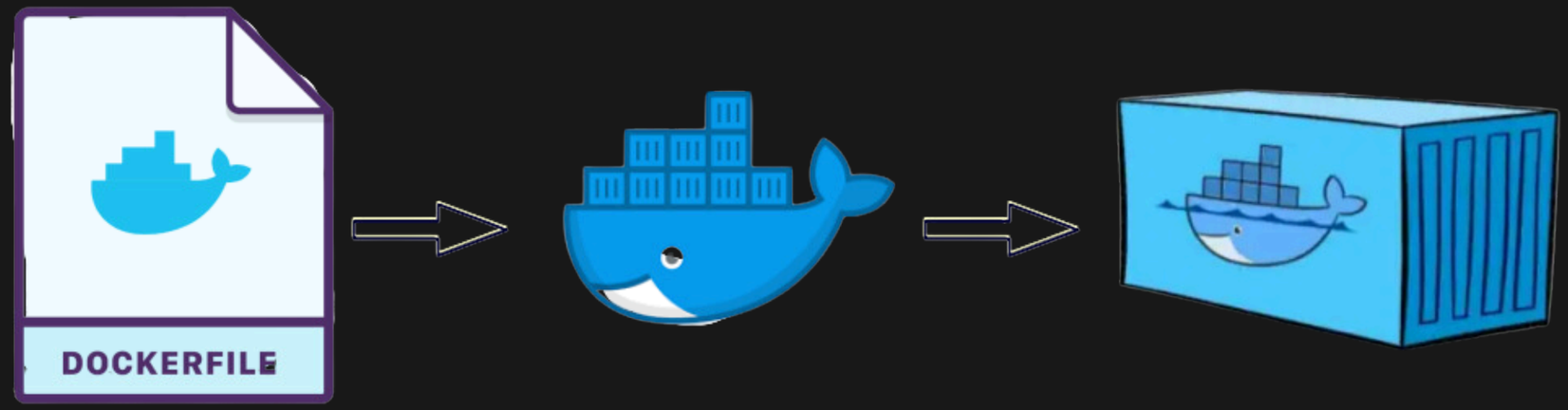


# Why docker?

- isolation
- lightweight
- portable

=> shipping container system for code

# From Dockerfile to Container



# From Dockerfile to Container

# Dockerfile

- text file with instructions (layers)
- perform actions on base image

  ==> create new docker image

Dockerfile

```
# Use a base image
FROM python:3.9

# Set working directory
WORKDIR /app

# Copy requirements and install
COPY requirements.txt .
RUN pip install -r requirements.txt

# Copy model and app files
COPY . .

# Expose port for Flask
EXPOSE 5000

# Run the Flask app
CMD ["python", "app.py"]
```

# Docker Image

- static component (snapshot)
- contains everything an application needs to run

# Docker Container

- isolated from other container
- packages app and dependencies together
- standard unit in which the application service resides and runs

# Docker Hub

- repository
- storage and distribution service for images
- `> docker **push**`

# Key Takeaways

serving models

FastAPI

containerization

Docker