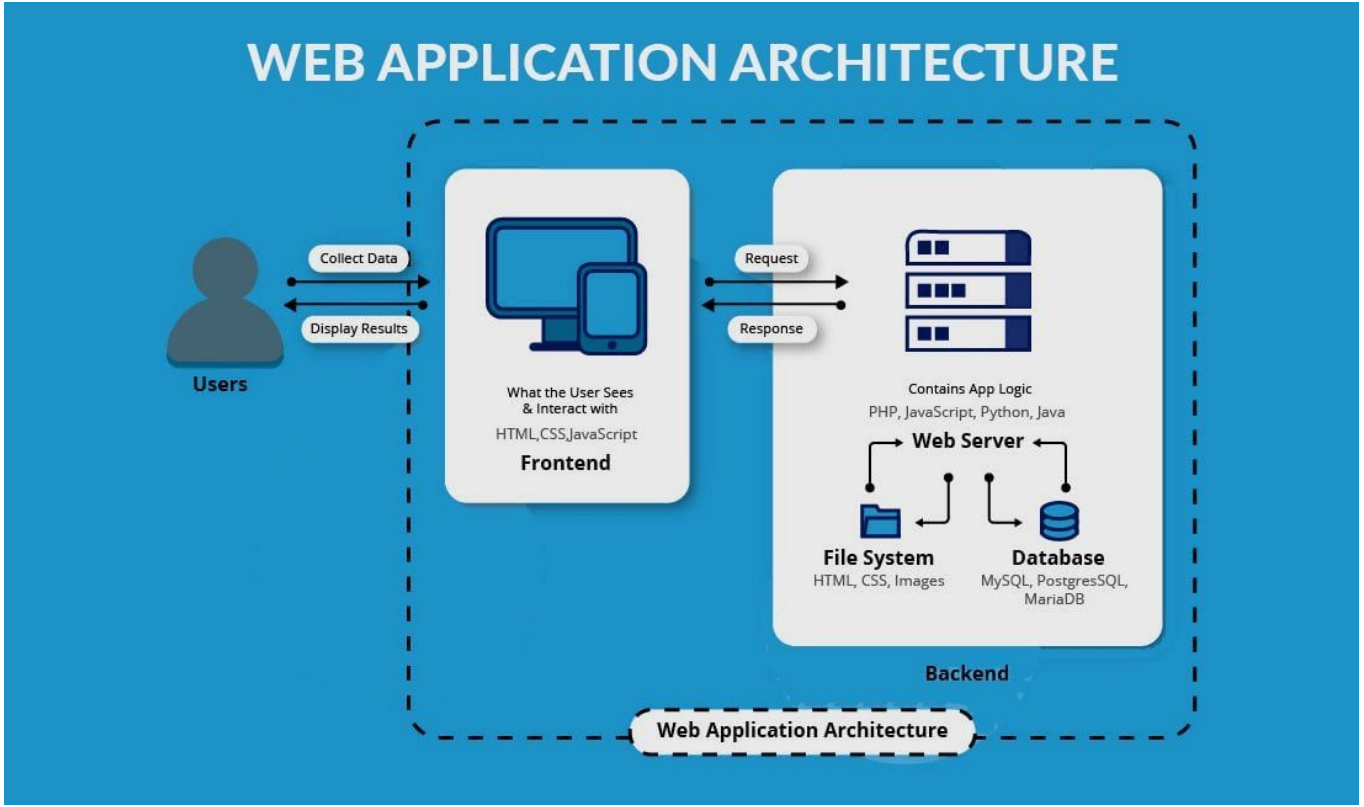
Model Serving Strategies

Web Application



Flask

Flask is a micro web framework written in Python.

This web app will be used to consume the created model.

For instance, a user can enter the feature values into the form and after submitting, the model will predict the outcome(churn or not).

Docs

Flask 101

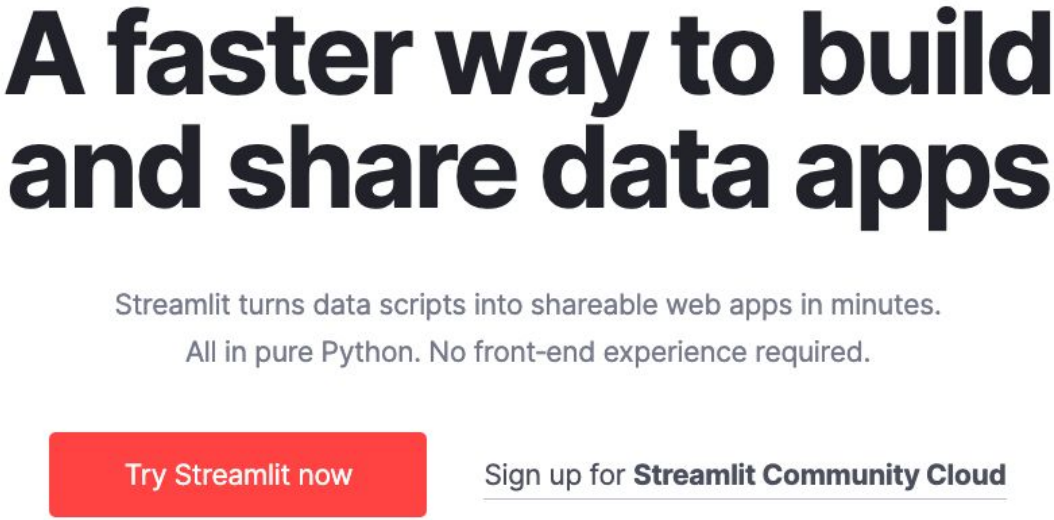
1) install Flask: *pip install flask*
2) create a **hello.py** file with

agiledatascience_labs > flask_lab > hello.py > hello_world

```
1   from flask import Flask
2
3   app = Flask(__name__)
4
5   @app.route("/")
6   def hello_world():
7       return "<p>Hello, World!</p>"
```

3) run the app in the server: *flask --app hello run --port=5002*

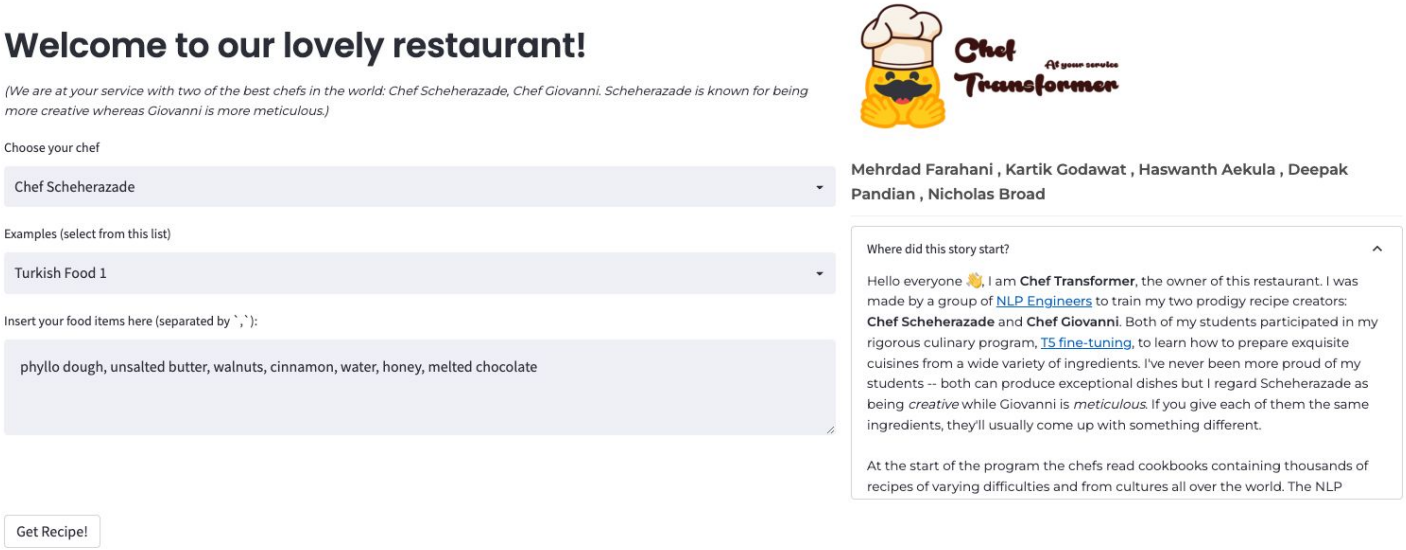
Streamlit - https://streamlit.io/



Docs

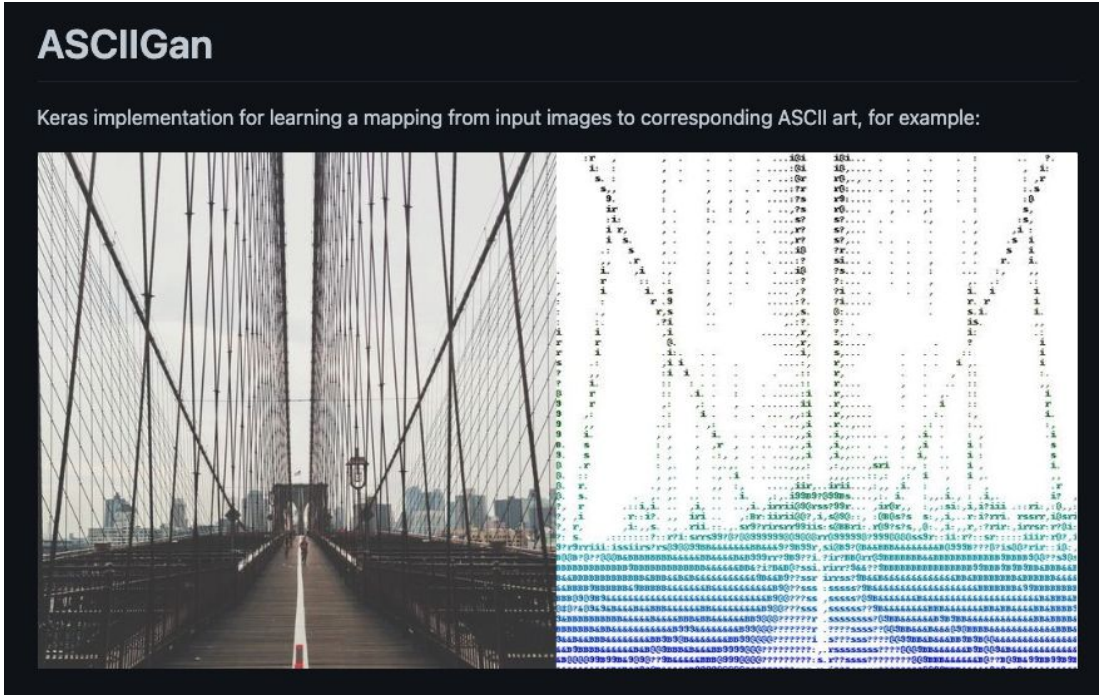
Streamlit - Chef Composer

Chef Transformer

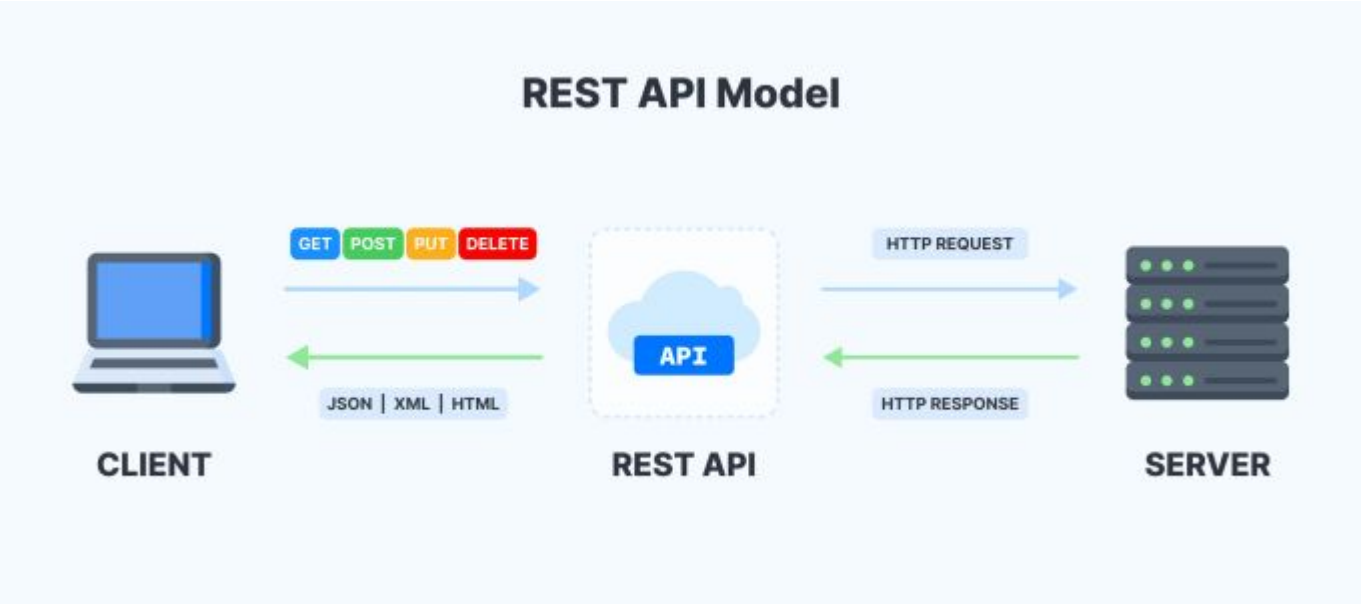


Streamlit - Pic2Ascii

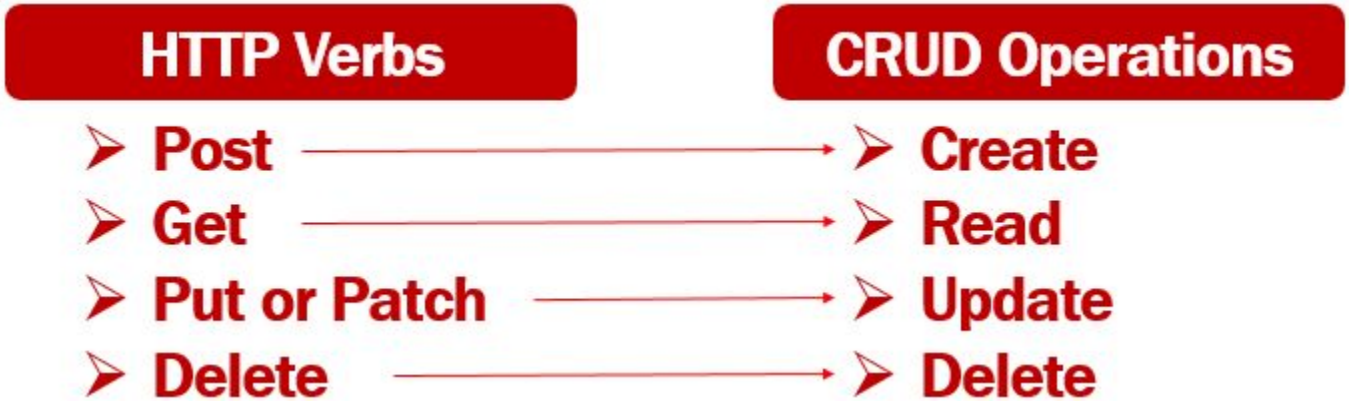
[Github]



API REST


REST API Model
GET
POST
PUT
DELETE
HTTP REQUEST
API
JSON | XML | HTML
HTTP RESPONSE
CLIENT
REST API
SERVER

API REST


HTTP Verbs
CRUD Operations
Post
Create
Get
Read
Put or Patch
Update
Delete
Delete

API REST


GET
/pet/{petId} Find pet by ID
PUT
/pet Update an existing pet
DELETE
/pet/{petId} Deletes a pet
POST
/pet/{petId}/uploadImage uploads an image

FAST API

FastAPI framework, high performance, easy to learn, fast to code, ready for production

Docs

FAST API - Example

1) go to **fastapi-lab** in the course github
2) install **fastapi** and **uvicorn** packages
3) train the model executing **python model/train.py**
4) run the API REST server with **uvicorn app:app --reload** under the project folder
5) review and test it in the Fast API docs: **http://127.0.0.1:8000/docs**

FAST API - Docker

1) Review the Dockerfile
2) run **docker build -t fastapiml .**
3) run **docker run -d --name heart-disease-serving -p 80:80 fastapiml**

Batch inference