How to run CUPID API

Process to run:

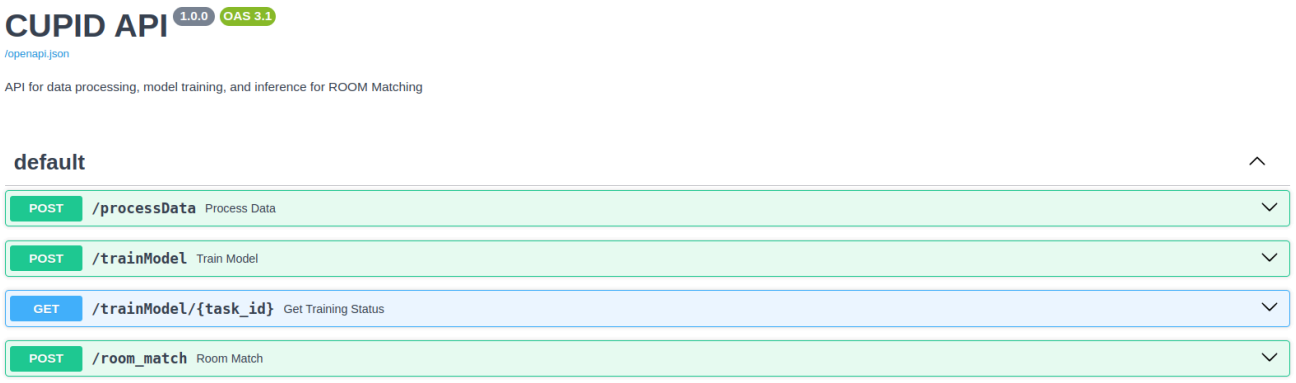
1. Clone the repo: `git clone https://github.com/shekharkoirala/CUPIDApi.git`
2. Setup data:
   create a data folder inside the cloned CUPIDApi folder `cd CUPIDApi && mkdir data`
3. Download two file inside the `data` folder.
   https://drive.google.com/file/d/1hKgQ3JwGlMVxdx3c41oas89QoXrsez56/view?usp=drive_link
   https://drive.google.com/file/d/1mLQ3O0ybTfNzJF_eC2nC1PlMBUlsFJl7/view?usp=drive_link

other necessary documents like spacy cache is here: spacy cache
3. create .env file from .env.sample : `touch .env`

4. update the .env file with the keys: API_KEY=<insert_key>
5. Build the docker: `docker build -t api .`
6. Run the docker: `docker run -itd -p 8000:8000 api`
7. Verify data (pdf files are in) in docker pod (Optional)

```
docker exec -it < docker container id> bash
cd data
ls
exit
```

7. check the url in any local browser: http://localhost:8000/docs



Important step : Don't trigger the processData as it the final data is created manually,

You can either use, web UI, or curl to trigger the endpoints.

If you only want to test the model, you can directly post the /room_match without triggering any other endpoints.

1. Post request to /processData

```
There are two method to processData, documented in two different branches.
The final data used to train the model is combination of both dataset.

A) for the process documented in Main branch, using only cosine
similarity:
- It takes around 50-100 seconds to process the data.
- In Production, A queue will be implemented instead of API for the long
processes. ( a small demonstration is shown in trainModel endpoint. )

B) for the process documented in newData branch, using spacy to find the
text similarity.
```

Both methods will produce a 4000 pair of matched training data with 0,1 label.
I have used the sample of class 1 from the spacy method (more accurate) and sample of class 0 from cosine similarity method.
The data is stored at ./data/new_training_data.json

2. Post request to /trainModel

- You might need to wait few minutes to download the sentence Transformer for the first time.
- You should expect to see a task_id returned by the endpoint. The task_id will be used to know the status of model training.

```
curl -X 'POST' \
  'http://localhost:8000/trainModel' \
  -H 'accept: application/json' \
  -H 'X-API-Key: e2R4t6Y8u0i3O5p7A9s1D3f5G7h9J2k4' \
  -H 'Content-Type: application/json' \
  -d '{
  "parameter_tuning": false
}'
```

expected result will be :

```
{
  "status": "accepted",
  "message": "Model training started",
```

```
  "task_id": "3d799eff-2054-42be-9f58-1eff44c476e8"
}
```

3. Get request to /trainModel/{task_id}

```
curl -X 'GET' \
  'http://localhost:8000/trainModel/3d799eff-2054-42be-9f58-1eff44c476e8'
\
  -H 'accept: application/json' \
  -H 'X-API-Key: e2R4t6Y8u0i3O5p7A9s1D3f5G7h9J2k4'
```

expected result will be:

```
{
  "status": "completed",
  "message": "Training status retrieved",
  "model_path": "mlmodels/xgb_model.json",
  "error": null
}
```

This is just an example of creating API for model training (heavy task load). A task_id (token) will be forwarded and the end user should keep on polling the status / log message of the process.

Important Note: Make sure you have completed this before testing the inference.

4. Post request to /room_match

```
curl -X 'POST' \
  'http://localhost:9000/room_match' \
  -H 'accept: application/json' \
  -H 'X-API-Key: e2R4t6Y8u0i3O5p7A9s1D3f5G7h9J2k4' \
  -H 'Content-Type: application/json' \
  -d '{
  "debug": true,
  "inputCatalog": [
    {
      "supplierId": "nuitee",
      "supplierRoomInfo": [
        {
          "supplierRoomId": "2",
          "supplierRoomName": "Classic Room - Olympic Queen Bed - ROOM
ONLY"
        },
        {
```

```
          "supplierRoomId": "3",
          "supplierRoomName": "CLASSIC ROOM ADA - ROOM ONLY"
        },
        {
          "supplierRoomId": "5",
          "supplierRoomName": "SUPERIOR ROOM ADA - ROOM ONLY"
        },
        {
          "supplierRoomId": "10",
          "supplierRoomName": "Superior Room - Olympic Queen Bed - ROOM
ONLY"
        },
        {
          "supplierRoomId": "6",
          "supplierRoomName": "Superior City View - Olympic Queen Bed -
ROOM ONLY"
        },
        {
          "supplierRoomId": "7",
          "supplierRoomName": "Balcony Room - Olympic Queen Bed - ROOM
ONLY"
        }
      ]
    }
  ],
  "referenceCatalog": [
    {
      "propertyId": "5122906",
      "referenceRoomInfo": [
        {
          "roomId": "512290602",
          "roomName": "Classic Room"
        },
        {
          "roomId": "512290603",
          "roomName": "Superior Room"
        },
        {
          "roomId": "512290604",
          "roomName": "Superior Room with City View"
        },
        {
          "roomId": "512290605",
          "roomName": "Balcony Room"
        },
        {
          "roomId": "512290608",
          "roomName": "Classic Room - Disability Access"
        },
        {
```

```
          "roomId": "512290609",
          "roomName": "Superior Room - Disability Access"
        },
        {
          "roomId": "512290610",
          "roomName": "Junior Suite - Disability Access"
        }
      ]
    }
  ],
  "threshold": 0.95
}'
```

Expect to change the threshold value as per your need.
The threshold value is also mentioned in the config.yaml.

The expected answer would be (Matching the official cupid API).
`

```
{
  "Results": [
    {
      "cleanRoomName": "Classic Room",
      "mappedRooms": [
        {
          "cleanSupplierRoomName": "CLASSIC ROOM ADA - ROOM ONLY",
          "score": 0.9953511953353882,
          "supplierId": "nuitee",
          "supplierRoomId": "3",
          "supplierRoomName": "CLASSIC ROOM ADA - ROOM ONLY"
        }
      ],
      "propertyId": "5122906",
      "roomId": "512290602",
      "roomName": "Classic Room"
    },
    {
      "cleanRoomName": "Superior Room with City View",
      "mappedRooms": [
        {
          "cleanSupplierRoomName": "SUPERIOR ROOM ADA - ROOM ONLY",
          "score": 0.9944533705711365,
          "supplierId": "nuitee",
          "supplierRoomId": "5",
          "supplierRoomName": "SUPERIOR ROOM ADA - ROOM ONLY"
        }
      ],
      "propertyId": "5122906",
      "roomId": "512290604",
```

```
      "roomName": "Superior Room with City View"
    }
  ],
  "UnmappedRooms": [
    {
      "cleanSupplierRoomName": "Classic Room - Olympic Queen Bed - ROOM
ONLY",
      "supplierId": "nuitee",
      "supplierRoomId": "2",
      "supplierRoomName": "Classic Room - Olympic Queen Bed - ROOM ONLY"
    },
    {
      "cleanSupplierRoomName": "Superior Room - Olympic Queen Bed - ROOM
ONLY",
      "supplierId": "nuitee",
      "supplierRoomId": "10",
      "supplierRoomName": "Superior Room - Olympic Queen Bed - ROOM ONLY"
    },
    {
      "cleanSupplierRoomName": "Superior City View - Olympic Queen Bed -
ROOM ONLY",
      "supplierId": "nuitee",
      "supplierRoomId": "6",
      "supplierRoomName": "Superior City View - Olympic Queen Bed - ROOM
ONLY"
    },
    {
      "cleanSupplierRoomName": "Balcony Room - Olympic Queen Bed - ROOM
ONLY",
      "supplierId": "nuitee",
      "supplierRoomId": "7",
      "supplierRoomName": "Balcony Room - Olympic Queen Bed - ROOM ONLY"
    }
  ]
}
```