

Using the service in local

In order to use prediction service locally, it should be activated in your machine. A way to do is activating the container as described in *Using_Pipfile_and_Dockerfile.pdf*. An alternative is to move into the `local_service` folder in command line and type:

```
python predict.py
```

that will yield such a screen:

```
>-----[base] chapar@pop-os Desktop/midterm_project/local_service
>~~~| ls
Dockerfile model.bin Pipfile Pipfile.lock predict.py predict-test.ipynb predict-test.py train.py
>-----[base] chapar@pop-os Desktop/midterm_project/local_service
>~~~| python predict.py
* Serving Flask app 'energy' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses.
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://192.168.0.11:9696/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 138-769-174
```

Next step is to reach this service. For it, use the `predict-test.ipynb` file that is in `local_service` folder.

```
+ Code + Markdown ▶ Run All ... Python 3.9.5 64-bit ('midterm_project-TBMLFfwK': pipenv)

[1] ✓ 0.6s Python
1 import requests

[2] ✓ 0.2s Python
1 url = 'http://localhost:9696/predict'

[3] ✓ 0.1s Python
1 building = {"compactness": 0.9,
2             "surface_area": 563.5,
3             "wall_area": 318.5,
4             "roof_area": 122.5,
5             "height": 7.0,
6             "orientation": 5.0,
7             "glazing_area": 0.4,
8             "glazing_distribution": 4.0}

[4] ✓ 0.9s Python
1 response = requests.post(url, json=building).json()
2
3 print(response)

... {'cooling': 35.88, 'heating': 36.23}
```

Alternatively, you can modify the `predict-test.py` file (in the same folder) by setting building parameters in your editor as you wish and run it with:

```
python predict-test.py
```