

Using Pipfile and Dockerfile

For the dependence management of the project, the `pipenv` packaging tool is used. It will allow users of this repository to get exact versions of the packages that are used while developing it.

In order to install `pipenv`, type the line below to your terminal:

```
pip install pipenv
```

Once installing completed, move to the project folder. Since the Pipfile and Pipfile.lock files are under main folder, just type:

```
pipenv install
```

This command will create a virtual environment that is defined by Pipfile and Pipfile.lock files. Installation takes few seconds and you will get such a screen:

```
>~~~| ls
data documentation local_service notebooks Pipfile Pipfile.lock README.md web_service
>-----[base] chapar@pop-os ~/Desktop/midterm_project
>~~~| pipenv install
Creating a virtualenv for this project...
Pipfile: /home/chapar/Desktop/midterm_project/Pipfile
Using /usr/bin/python3.9 (3.9.5) to create virtualenv...
:: Creating virtual environment...created virtual environment CPython3.9.5.final.0-64 in 361ms
creator CPython3Posix(dest=/home/chapar/.local/share/virtualenvs/midterm_project-TBMLFfwK, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=/home/chapar/.local/share/virtualenv)
added seed packages: pip==21.2.4, setuptools==58.2.0, wheel==0.37.0
activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator

✓ Successfully created virtual environment!
Virtualenv location: /home/chapar/.local/share/virtualenvs/midterm_project-TBMLFfwK
Installing dependencies from Pipfile.lock (73ca8d)...
  27/27 - 00:00:11
To activate this project's virtualenv, run pipenv shell.
Alternatively, run a command inside the virtualenv with pipenv run.
>-----[base] chapar@pop-os ~/Desktop/midterm_project
>~~~|
```

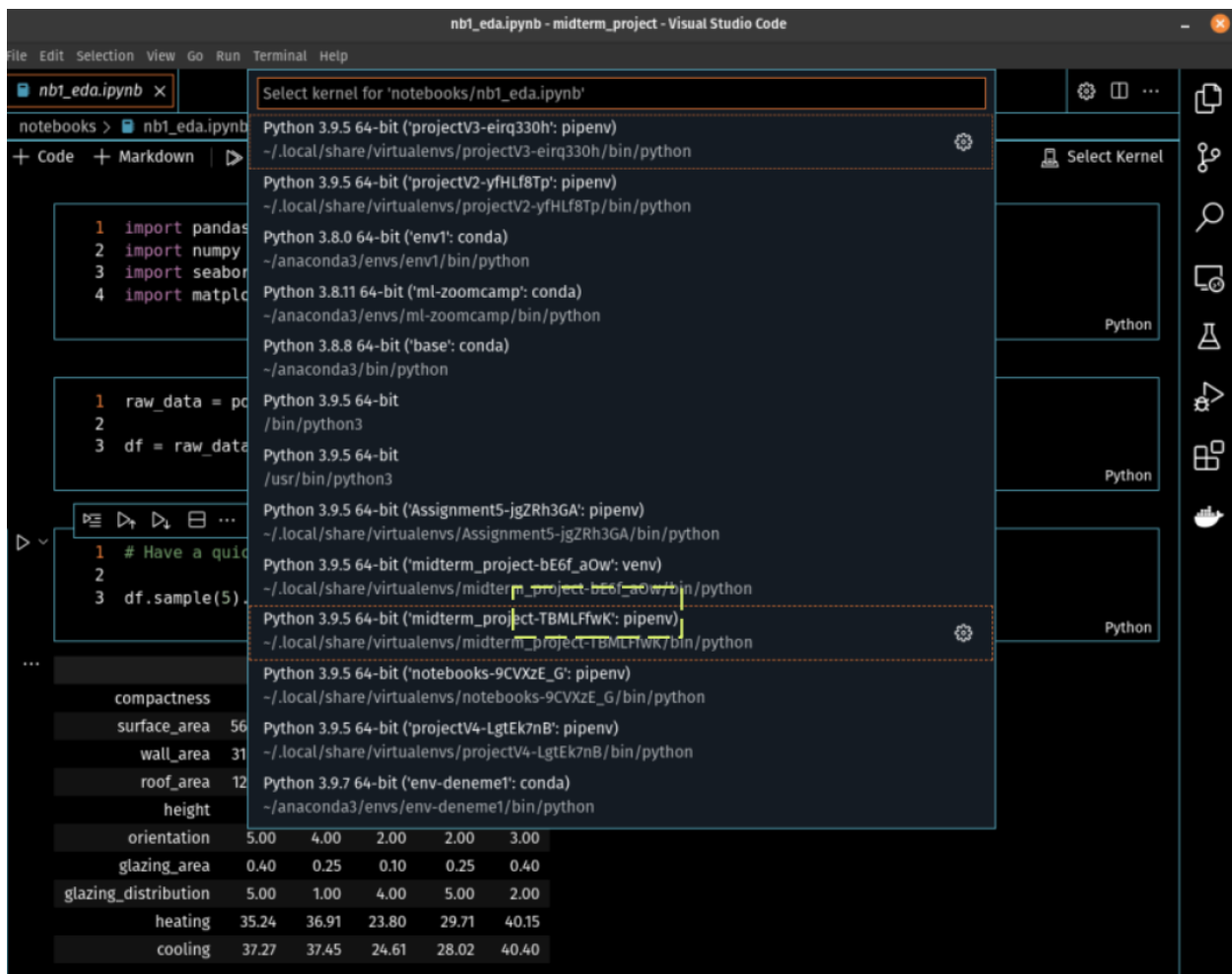
As the command line instructions mention, next step is to activate this virtual environment by typing:

```
pipenv shell
```

This yields such an output:

```
>-----[base] chapar@pop-os ~/Desktop/midterm_project
>~~~| pipenv shell
Launching subshell in virtual environment...
. /home/chapar/.local/share/virtualenvs/midterm_project-TBMLFfwK/bin/activate
>-----[base] chapar@pop-os ~/Desktop/midterm_project
>~~~| . /home/chapar/.local/share/virtualenvs/midterm_project-TBMLFfwK/bin/activate
>-----[base] chapar@pop-os ~/Desktop/midterm_project
>~~~|
```

Notice the identifier shown with rectangle. In your code editor, you will pick it. As an example, in VS Code, select the kernel with this identifier:



Note that you may need to restart your editor to see in in the list.

In order to use the Dockerfile, a similar procedure is needed: building (like `pipenv install`) and running (like `pipenv shell`). In order to activate the docker image, move to the `local_service` folder (under the main folder) in command line and type:

```
docker build -t give_name .
```

You can put any name instead of `give_name` term.

In case experiencing authorization problems due to your own Docker installation, this may help:

```
sudo chown $(whoami):$(whoami) /var/run/docker.sock
```

When building it successfully (would take few minutes), type the code below to activate (pay attention to use same name given in previous step):

```
docker run -it --rm -p 9696:9696 give_name
```

```
>-----[base] chapar@pop-os Desktop/midterm_project/local_service
>~~~| docker run -it --rm -p 9696:9696 zoomcamp
* 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://172.17.0.2:9696/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 298-838-788
```

The screen shows that the prediction service is up and running locally.

In order to use the service, follow the instructions in *Using_service_in_local.pdf* file.

Note that the `local_service` folder contains the same `Pipfile` and `Pipfile.lock` files. It is intended to keep this folder independent. In case the folder is moved out of the main folder, `Dockerfile` still will be able to reach those and run service.