trained\_models = save\_trained\_model\_elements

pipelines\_training\_and\_saving = training\_model

* train\_pipeline = train

prediction\_with\_model\_trained = new\_prediction\_with\_model\_trained

* new\_predict = predict

pipelines\_building = pipelines\_building

* build\_pipelines = pipelines

**for house-prices-api folder (to build fastapi (web frameworks) interface for users)**

* app = building\_fastapi\_app

api.py = prepare\_fastapi\_app.py

main.py = create\_fastapi\_app.py

config.py = config\_fastapi\_app.py

* schemas = visual\_presentation\_on\_fastapi\_app

health.py = health\_config\_fastapi\_app.py (config for health part in users interface)

predict.py = predict\_config\_fastapi\_app.py (config for predict part in users interface)

* tests = tests\_model\_on\_fastapi\_app

conftest.py = for pytest and variable used by any python file beginning by test\_

test\_api.py = test\_new\_prediction\_on\_fastapi\_app

**Requirements of deployment ML model in production**

* Web Frameworks = FastAPI (but you also have Django and Flask)
* Web Server = uvicorn
* Plateforme as a service (Paas) = Railway

Basically, a platform as a service simplifies some of the infrastructure maintenance that we typically would have to do when developing and launching an app. Another goal to use Paas in the deployment of ML models is to make the API built (ML models ) available to be called by mobile app, or front-end or web app and display the results in a customers interface.

You can also use Infrastructure as a service (Iaas), or Plateforme as a service (Paas) = Railway (500 hours for free), or Software as a service (Saas)