* Create python environment
  + python3 -m venv env
  + source env/bin/activate
* install packages
  + pip install django django-filter djangorestframework djangorestframework-gis
* upgrade pip
  + pip install –upgrade pip
* install linters
  + pip install flake8 (to verify pep8)
  + pip install black (code formatter)
* pip freeze > requirements.txt
* create project : django-admin startproject hospitals\_api .
* Open vscode
* add a setup,cfg configuration file
* aller dans settings.py
  + renommer Installed\_apps en DJANGO\_APPS
  + créer THIRD\_PARTY\_APPS = []
  + PROJECTY\_APPS = []
  + INSTALLED\_APPS = DJANGO\_APPS + THIRD\_PARTY\_APPS + PROJECTY\_APPS
  + Dans DJANGO\_APPS ajouter :
    - ‘django.contrib.gis’
  + Dans THIRD\_PARTY\_APPS :
    - ‘rest\_framework’, ‘rest\_framework\_gis’
  + Change Time Zone :
    - Ex : Africa/Kigali
  + python3 manage.py runserver
* create new github repo online
* go to terminal and create a .gitignore file
  + npx gitignore python (only work if you have nodejs)
  + go to [www.toptal.com/developers/gitignore/api/python](http://www.toptal.com/developers/gitignore/api/python) and copy the gitignore
* initialize a git project : git init
* add remote origin
* commit and push
* add a dockerfile
* add a .dockerignore
* create an .env file to add ENVIRONMENT Variables
* create a docker-compose.yaml
* docker compose config
* installer pyscopg2-binary et django-environ
* Aller dans settings et copier les valeurs de var, d’environnement SECRET\_KEY, DEBUG et ALLOWED\_HOSTS dans .env
* configurer la partie DATABASES
* créer un makefile pour éviter d ‘écrire de trop longues commandes
* créer de nouvelles applications :
  + docker-compose exec api python3 manage.py startapp hospitals
  + docker-compose exec api python3 manage.py startapp boundaries
  + ajouter ces applications dans PROJECTY\_APPS
  + ajouter au dossier de chaque appli, un dossier data contenant les shapefiles
  + pour inspecter ces data localement, il faut installer gdal (déjà installé grâce au dockerfile)
  + Ex : ogrinfo hospitals /data/Hospitals.shp
  + ogrinfo -so hospitals /data/Hospitals.shp Hospitals
  + ogrinfo -al hospitals /data/Hospitals.shp Hospitals
  + Utiliser gdal peut nous aider à observer les données et voir ce qu’on mettra dans nos modèles
* Aller dans hospitals/models.py
* srid : spatial reference system identity
* make makemigrations
* make migrate
* créer un fichier load,py dans hospitals afin de faire un layermapping :
  + permet de lier les champs créés dans models,py avec les champs observés au niveau des donées (ogrinfo)
* Pour mettre les données dans la base :
  + make shell
  + from hospitals import load
  + load,run()
  + les données des shapefiles seront saved
* Faire la même chose pour les boundaries :
  + on peut aller plus rapidement avec une seule commande pour nous générer le model et le layerMapping qu’on copiera ensuite dans nos fichiers
  + docker-compose exec api python3 manage,py ogrinspect boundaries/data/Boundary.shp Boundary –srid=4326 –mapping –multi
  + en copiant dans nos fichiers on peut modifier les noms qu’on ne veut pas
  + make makemigrations
  + make migrate
  + make shell
  + from boundaries import load
  + load.run()
* commit & push les résultats sur github