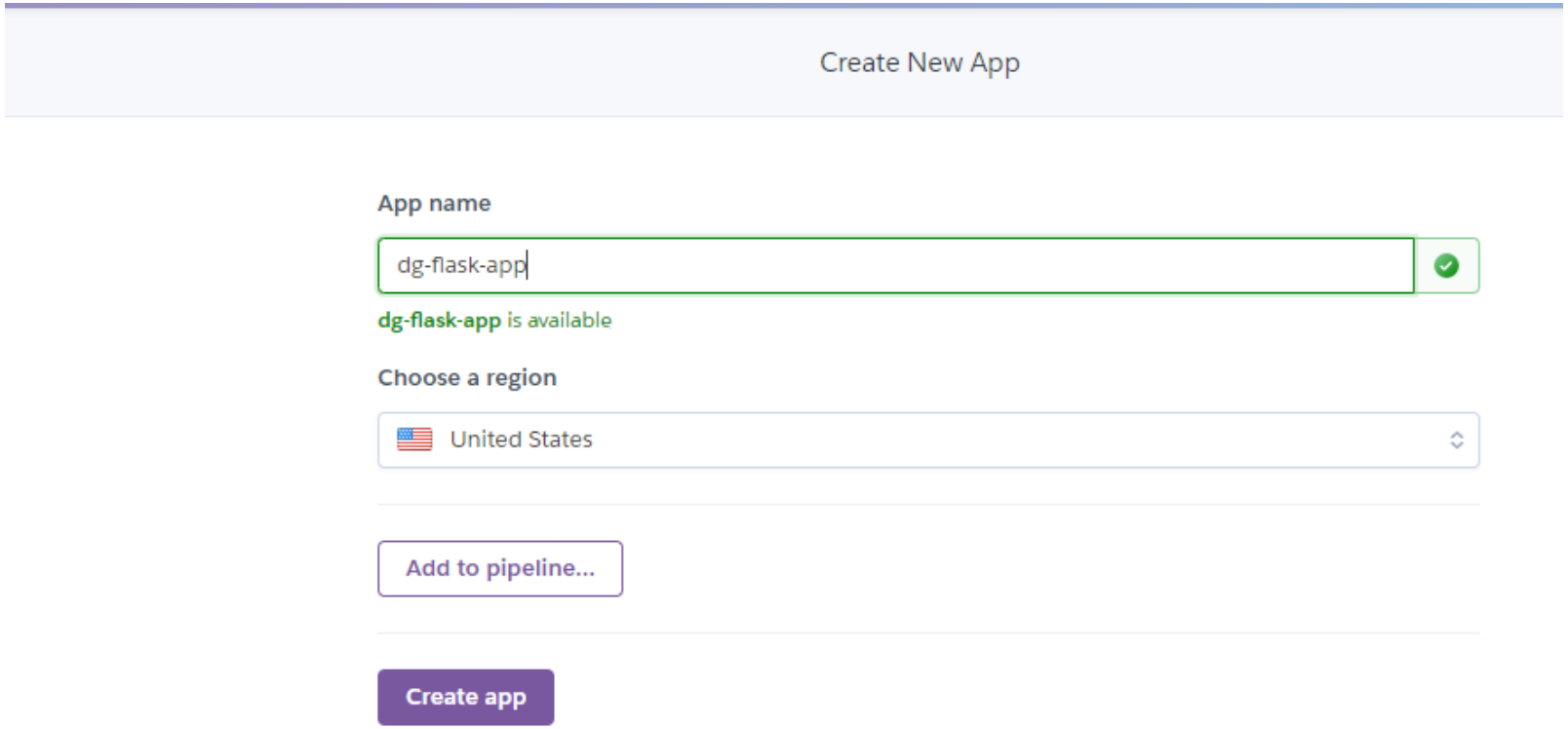
Deployment on Heroku

- Name : Fawzi El Khatib
- Batch code: LISP01
- Submitted to: Data Glacier Team
- Date: 01-April-2021

1. Create new app on Heroku

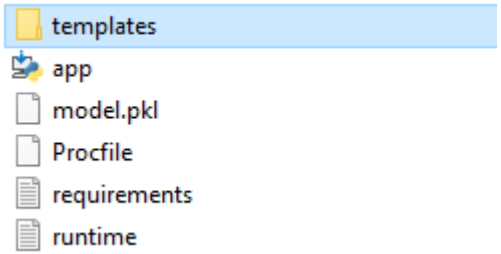


1. Create account on Heroku – Open source cloud –
2. Create our Heroku app, which we will track on git

1. Requirements:

(using flask app of week 4)

- 1.create our root folder:
- This is how the root folder looks like



- *Check carefully the types of the files from their properties*
- Procfile: *web: gunicorn app:app*

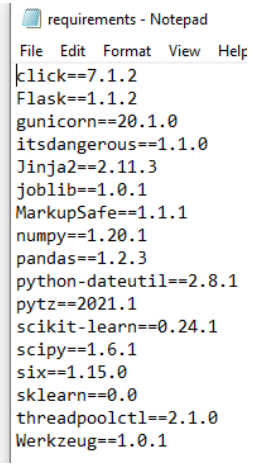
- Run:

```
$ pip freeze > requirements.txt
```

To lists the app dependencies together

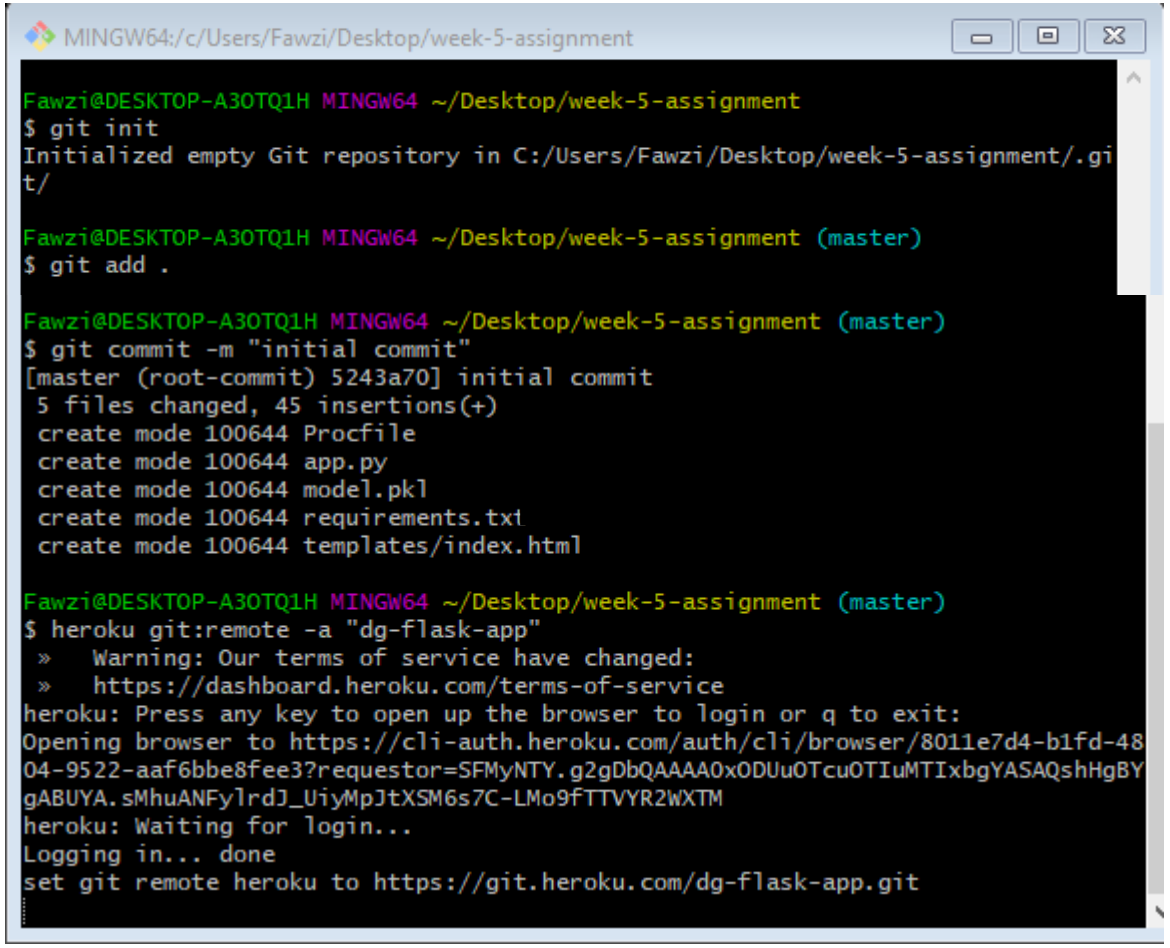
```
$ pip install -r requirements.txt
```

This command will make Heroku reads the file and installs the appropriate Python dependencies



- This is how the *requirements.txt* looks like

2. Create Heroku-hosted remote to our app



1. Initialize an empty git repo in our root folder that contains our app files

2. Track our app in the local git repository by adding and committing

3. Adding a remote to our local repo.

Here we have already created our Heroku app so we can easily add a remote using just the name of our Heroku app and the corresponding command.

➔Heroku Git repo associated with our app.

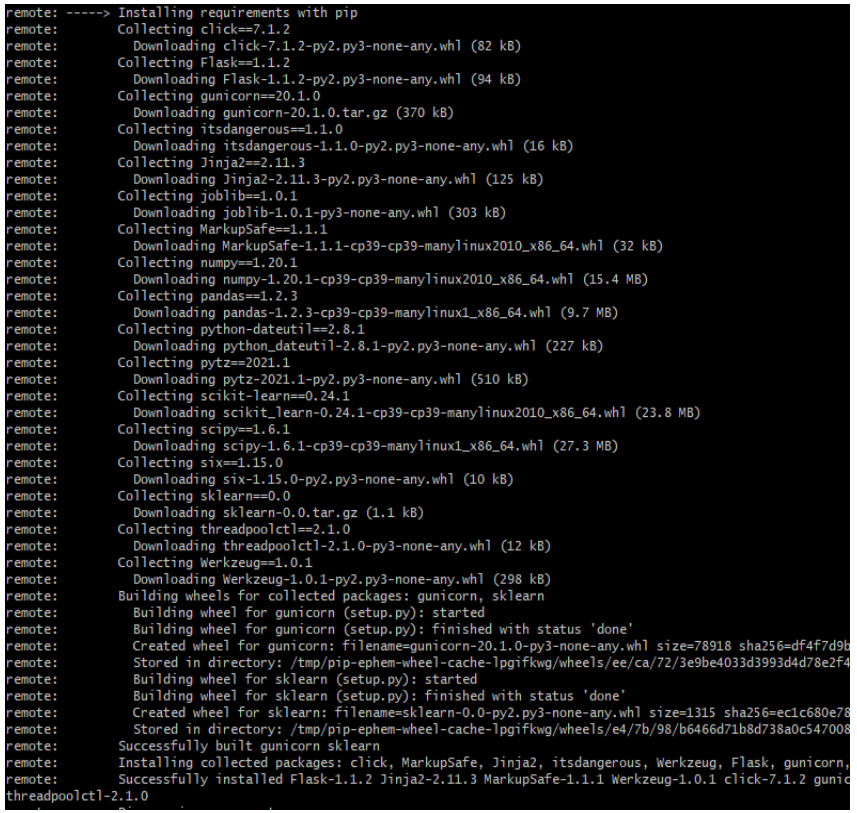
We can create a Heroku remote in many ways.
One of them is manually or use Heroku create…

Ps. I created a new folder name runtime.txt in order to tell Heroku which version of python should install. Because Heroku will install python-3.6 by default and it caused an error

3. Push the code to Heroku remote

```
Fawzi@DESKTOP-A3OTQ1H MINGW64 ~/Desktop/week-5-assignment (master)
$ git push heroku master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 233 bytes | 116.00 KiB/s, done.
Total 2 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Building on the Heroku-20 stack
remote: -----> Using buildpack: heroku/python
remote: -----> Python app detected
remote: -----> No change in requirements detected, installing from cache
remote: -----> Installing pip 20.1.1, setuptools 47.1.1 and wheel 0.34.2
remote: -----> Installing SQLite3
remote: -----> Installing requirements with pip
remote: -----> Discovering process types
remote:        Procfile declares types -> web
remote:
remote: -----> Compressing...
remote:        Done: 137.5M
remote: -----> Launching...
remote:        Released v12
remote:        https://dg-flask-app.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/dg-flask-app.git
   dba6525..b9748b4  master -> master
```

Our app is released in the web showing above



Successfully installed the requirements

Now we deployed our app to Heroku by pushing the code from our local repository using:

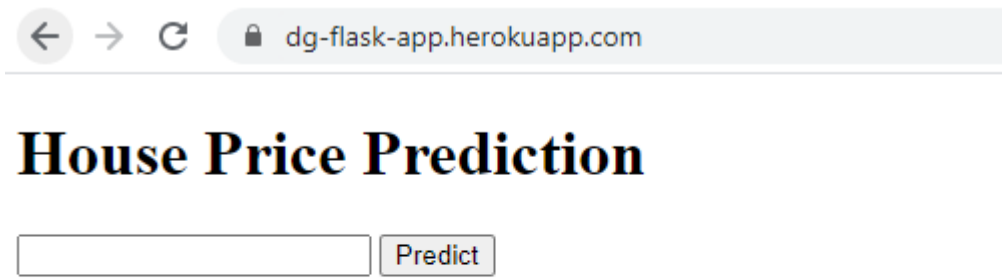
`$ git push heroku master`

4. Check the logs to confirm that everything is running without errors

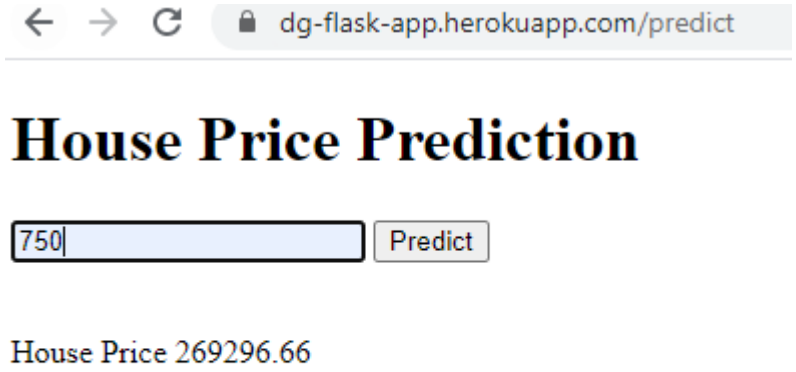
We can check the logs in our app in Heroku or by typing `$ heroku logs --tail`

Our Heroku app has been successfully built.

5. Clone the link we got in chrome



Finally our app is running smoothly.



Testing the app for a house of 750 sq.ft.

Ps. We can shut down the app by typing:
$ heroku ps:scale web=0

These are the steps to deploy our python model and publish it on open source cloud -Heroku–

We can follow the same logic in order to deploy our model in an open source cloud with considering changes needed upon your model requirements, coding language,…