Flask:

1. Create virtual-env and note that, start virtual environment with ps1, if in windows. like following.

* .\flask\_env\Scripts\activate.ps1

1. Then install flask:

* pip install flask

1. create “app.py” it can be anything name.
2. Create a Data Base, for that we will need a package named “flask.sqlalchemy”

* Pip install flask.sqlalchem
* Make some configrations in apps.py
* Create class for database
* Open python shell in terminal
  + from apps import db
  + db.create\_all()

**Similar Points in d-jango:**

@app.route("/", methods=['GET', 'POST'])

def hello\_world():

    if request.method == "POST":

        add\_todo = request.form['add\_todo']

* we have to give methods we are going to use in our methods, in app.route
* we are using form instead POST or Get method to get the input field value
* we have to import request here instead of passing it in method.

**Deployment Phase:**

We will deploy our todo app on “Heroku”

We are downloading Heroku cli for windows.

* Now install gunicorn using

“pip install gunicorn

* ”

Gunicorn:

The Gunicorn "Green Unicorn" is a Python Web Server Gateway Interface HTTP server. It is a pre-fork worker model, ported from Ruby's Unicorn project. The Gunicorn server is broadly compatible with a number of web frameworks, simply implemented, light on server resources and fairly fast.

Create requirement.txt

* pip freeze > requirements.txt

Create File “Procfile”:

* add this line: “web: gunicorn app:app”
* in terminal if Heroku cli is installed: them
* heroku
* heroku login
* git init
* git add .
* git commit -m “message”
* git remote -v //it should show remotes else repete the process
* heroku create pro\_name
* We will get a link click on it add open
* git push heroku master

if any Error is showing then solve it and again :

* git add .
* commit
* and push Heroku master.

Done

================================================