**How to push files to Github.**

1. Open Git Bash
2. cd Desktop
3. then go to working directory with cd folder name (eg. cd Python\_Classes).

* dir: helps to update which all files are present in folder.
* touch: helps to create file in folder (eg. touch demo.txt, will create demo.txt file in folder).
* git status: helps to check status of git of files

1. Initiate and track modification to the files using git init.
2. Track all untracked files in the folder using git Status.
3. Add the finalized file to Git Hub repository using

* git add file name to add individual file (upon typing git status, command prompt will make file green colored in git bash) or
* git add . will add all files present in that folder (upon typing git status, command prompt will make all file green colored in git bash).
* git clone is use to clone teammates codes by pasting link with git clone command.

1. commit the files by using git commit -m “write reason for committing”
2. open git hub and create new repository where all the files will be saved under repository.
3. Once you create the repository, you copy the path.
4. Go back to git bash and type git remote add origin and paste the path.
5. git push -u origin master will push all the files to created repository in github

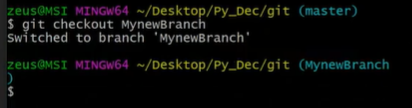
**How to Create Folder.**

1. Open Git Bash
2. cd Desktop
3. Mkdir folder name
4. cd folder name

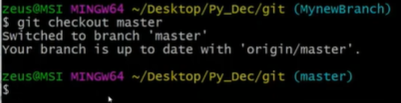
* cd stands for change directory
* cd.. to back to previous folder
* cat file name to read the file
* vi file name helps to write inside this file

**How to Create Branching.**

1. Open Git Bash
2. cd Desktop
3. then go to working directory with cd folder name (eg. cd Python\_Classes).
4. git branch folder name
5. to see what all branch are there, type git branch
6. git checkout folder name, then you will enter into branch from master



1. git checkout master or newbranch will show what all files are present and it will switch the path from master to newbranch or vice versa.



1. once branch is done push the branch to github
2. git merge newbranch name
3. followed by git push command
4. git branch -d folder name will delete the branch from local folder or git push origin –delete folder name will delete the branch from github repository.

**How to Create Tags.**

1. Open Git Bash
2. cd Desktop
3. then go to working directory with cd folder name (eg. cd Python\_Classes).
4. git tag will show all the available tags
5. git tag v1.0 will create tag named 1.0
6. git tag -a v1.1 -m “ver 1.1” will create subtag 1.1 under tag 1.0
7. git show version name, will show who created that version, creation date and commit number.
8. git log will show all the commits which had happened until now.
9. git tag -l “v1.\*” will show all the tags which starts with 1.
10. git push origin v1.0 will push version v1.0 tag, and if you want to push all the tags then use git push origin –tags
11. git tag -d version number will delete that tag from local folder or git push origin –d version name will delete the branch from github repository.

**How to create Virtual Environment**

1. Open command prompt

C:\Users\HOME>

1. Navigate to desktop, cd desktop.

C:\Users\HOME\Desktop>

* dir will show all folders and files which are on desktop

1. Navigate to folder, cd Python\_Classes

C:\Users\HOME\Desktop\Python\_Classes>

1. mkdir will help to create folder so type mkdir First\_Project

C:\Users\HOME\Desktop\Python\_Classes>mkdir First\_Project

1. navigate inside the project folder, cd First\_Project

C:\Users\HOME\Desktop\Python\_Classes\First\_Project>

1. to see all the packages which are installed on your machine type pip list

C:\Users\HOME\Desktop\Python\_Classes\First\_Project>pip list

1. if you want to install any package, type pip install package name to install that package on your machine.
2. C:\Users\HOME\Desktop\Python\_Classes\First\_Project>pip install virtualenv will help to create virtual environment.
3. C:\Users\HOME\Desktop\Python\_Classes\First\_Project>virtualenv Project\_A to setup the environment for subproject A.
4. In activate the virtual environment and install all the packages from local machine to virtual environment use Scripts generated in the folder using command below

* C:\Users\HOME\Desktop\Python\_Classes\First\_Project>cd Project\_A
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A>cd Scripts
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A\Scripts>activate
* (Project\_A) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A\Scripts>

1. Pip list in virtual environment will show all the packages which are present on that virtual folder.
2. (Project\_A) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A\Scripts>pip install pandas will install pandas on this virtual machine
3. To deactivate virtual environment use (Project\_A) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A\Scripts>deactivate.bat

This command give move you out of virtual environment C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A\Scripts>

1. Pip list or pip freeze both will show which all packages are present on this virtual environment.
2. (Project\_A) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_A>pip freeze --local >Packages.txt will create file named with Packages where all packages will be installed.

**How to read package file from one Virtual Environment to another**

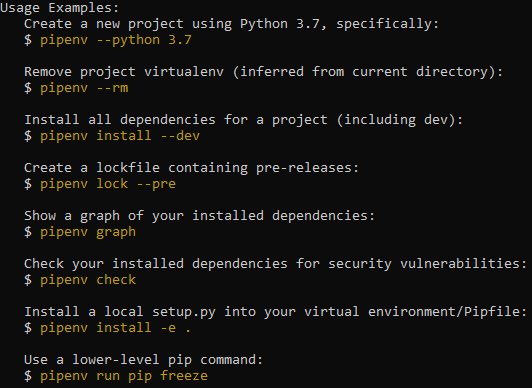
1. Open command prompt

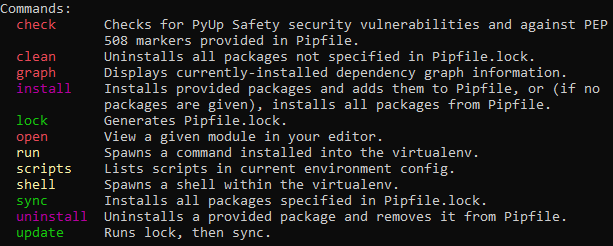
* C:\Users\HOME>
* C:\Users\HOME\Desktop>
* C:\Users\HOME\Desktop\Python\_Classes>
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project>
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project>virtualenv Project\_B
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project>cd Project\_B
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_B>cd Scripts
* C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_B\Scripts>activate
* (Project\_B) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_B\Scripts>

1. Copy the package file from one environment and paste to another environment.
2. (Project\_B) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_B>pip install -r Packages.txt will read the package file and start installing all the packages in this new virtual environment.

**How to manage the package**

1. C:\Users\HOME\Desktop\Python\_Classes>pip install pipenv
2. C:\Users\HOME\Desktop\Python\_Classes>pipenv will show all subcommands





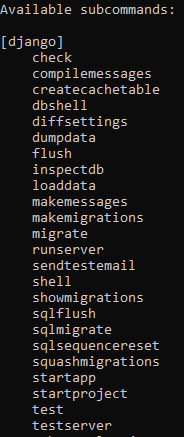
1. C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>pipenv install django will first create virtual environment and then install django in it.
2. C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>pipenv install pandas will first create virtual environment and then install pandas in it.
3. C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>pipenv shell will activate the environment as

(Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>.

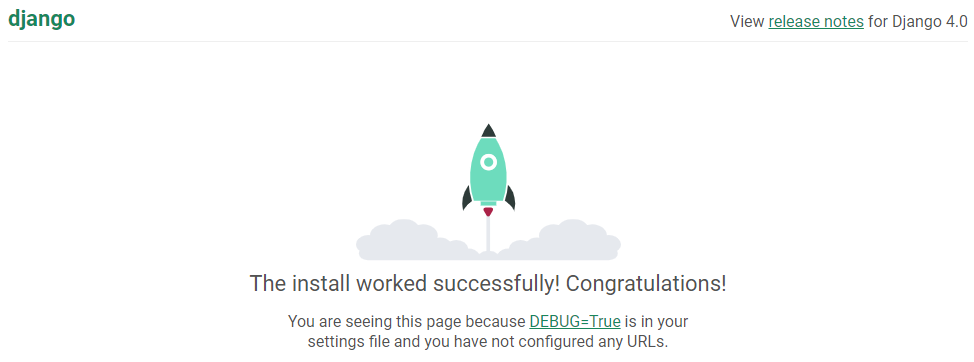
1. (Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>pipenv graph shows which all packages are included as function of django and panda installation as dependencies.

**How to create Project**

1. C:\Users\HOME\Desktop\Python\_Classes First\_Project\Project\_C >pipenv shell
2. (Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>
3. Ensure django is installed on this virtual environment by checking pip freeze command.
4. All the command for django starts with django, like django-admin shows all subcommands



1. django-admin startproject Python\_Website1 will create subfolder under Project C and start the project.
2. Once you start the project you have to run the project by navigating manage.py file in the folder using python manage.py runserver to run localhost:8000 on browser.

* (Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C>cd Python\_Website1
* (Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C\Python\_Website1>
* (Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C\Python\_Website1>python manage.py runserver
* 

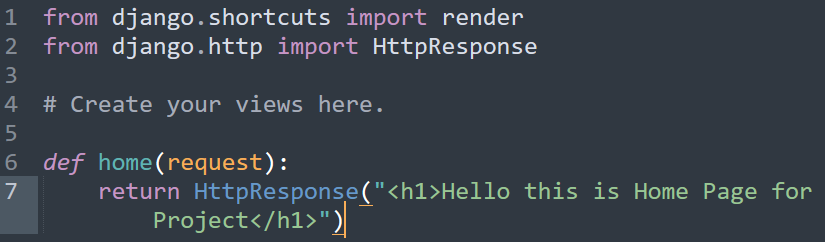
1. To stop the server press control C
2. Under the application, create an app using python manage.py startapp HomePage.

(Project\_C-FeboVhhN) C:\Users\HOME\Desktop\Python\_Classes\First\_Project\Project\_C\Python\_Website1>python manage.py startapp HomePage

1. Once app is created, install the app by

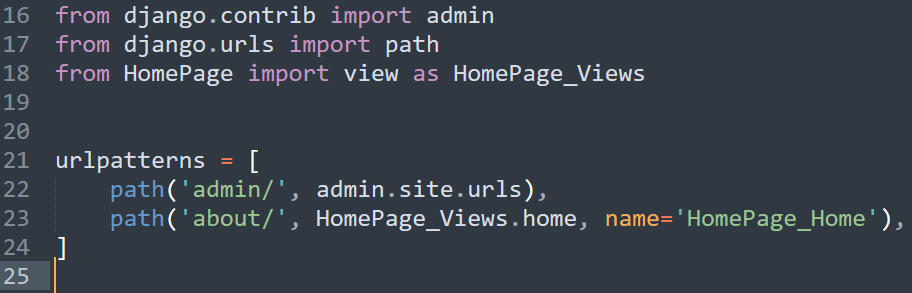
* Opening app.py file
* Copying the class
* Under setting.py file from main project folder, paste class from app under INSTALLED\_APPS list “HomePage.apps.HomepageConfig”.

1. Once its installed, create view using

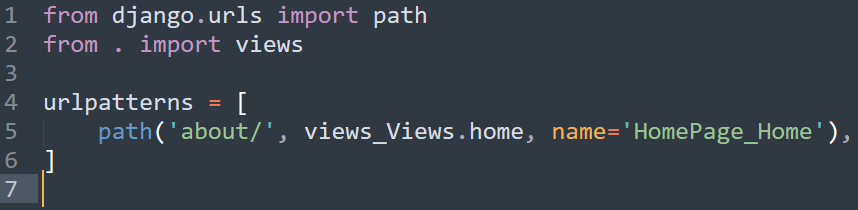
* View.py in HomePage folder
* Add http code here
* 

1. Once view is created, routing the code for website display by,

OPTION 1:

* Import the view.py file in url.py under Main Project folder as “from HomePage import view as HomePage\_Views”
* Add path in urls.py under Main Project folder as below
* 

OPTION 2:

* Create new file under HomePage app as urls.py
* Copy import & path code of Main Project’s urls.py file and paste it under HomePage’s urls.py file.
* 
* Add path in url.py under Main Project folder as below
* 