Git/Github, Dash, and Heroku

(IE 481, Lab #3 Configuration)

•••

Joonyoung Park

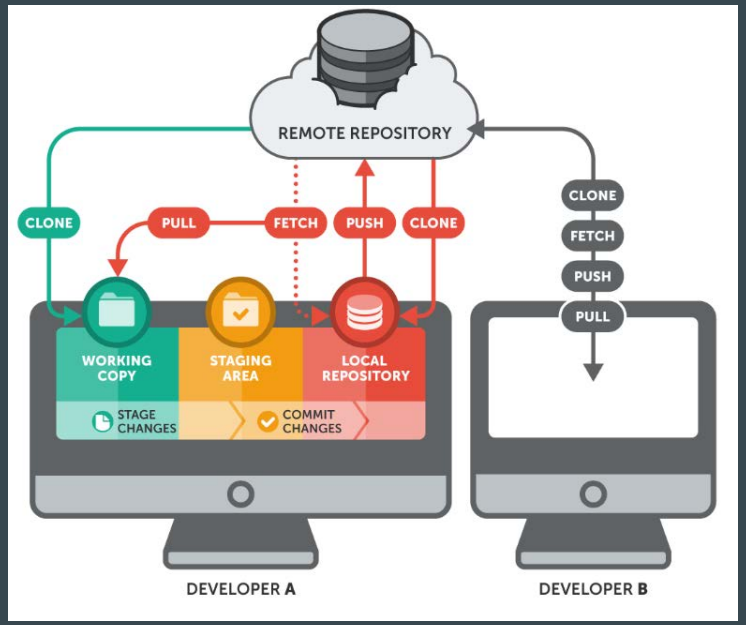
Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

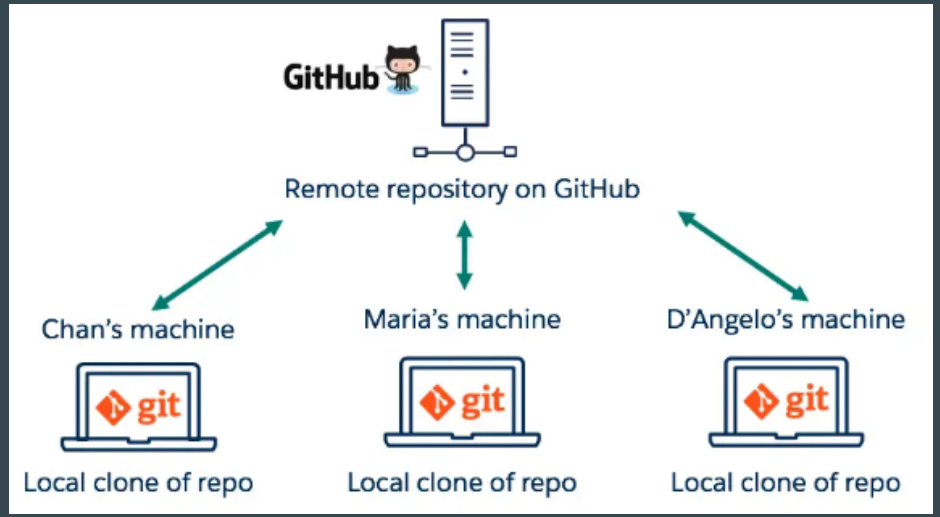
Git/Github

What is Git and Github?

- **Git:** A distributed version control system for tracking and managing history of changes of files
- **Github:** A remote repository to upload and share the files managed on the local machine by Git



An example of Git operation



Relationship between Git and Github

https://medium.com/@hitesh19arora/git-an-open-source-version-control-system-bd3768b0306e
https://trailhead.salesforce.com/en/content/learn/projects/develop-app-with-salesforce-cli-and-source-control/add-salesforce-dx-project-to-source-control

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Install and configure Git

- Go to the Git homepage: link
- Click ‘Download’ (Fig 1) → Select your OS version (Fig 2.) → Start downloading automatically (Fig 3.)
- In step Fig 3. , choose your own path to download executable file

Fig 1. Click ‘Download’

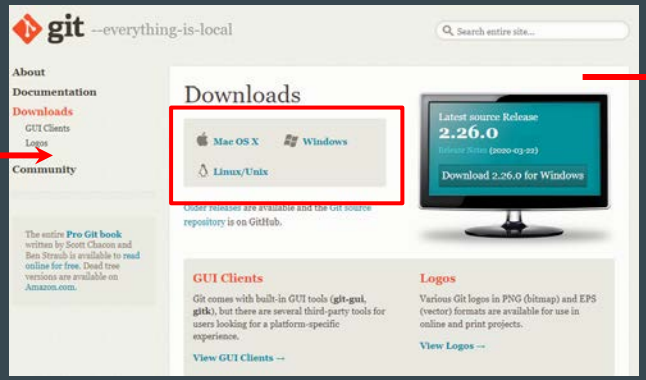


Fig 2. Select OS version

Fig 3. Start downloading automatically

Install and configure Git

- Iteratively click ‘Next’ and ‘Install’ button on the process of installation (Fig 4.)
- Uncheck the box of ‘view release Notes’ (Fig 5.)
- After installation, open a command line and Typing ‘git --version’ to check if the install completed successfully (Fig 6.)

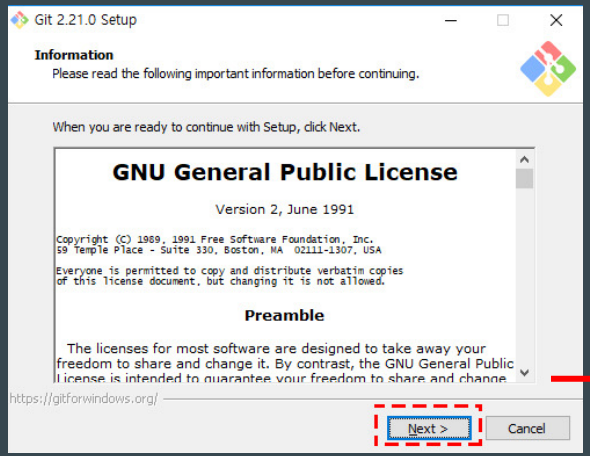


Fig 4. First step to install

(Some steps)

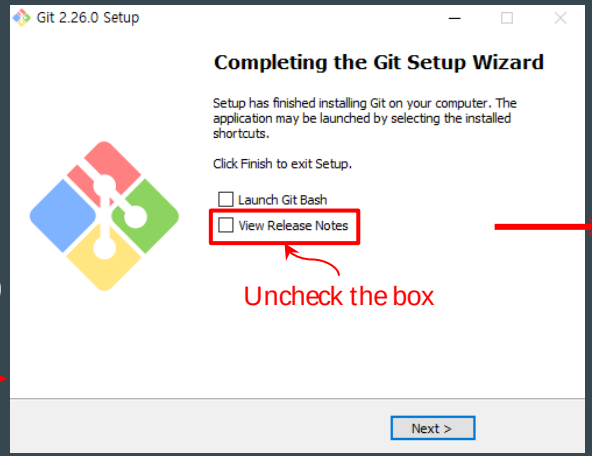


Fig 5. Uncheck the box of ‘view release Notes’

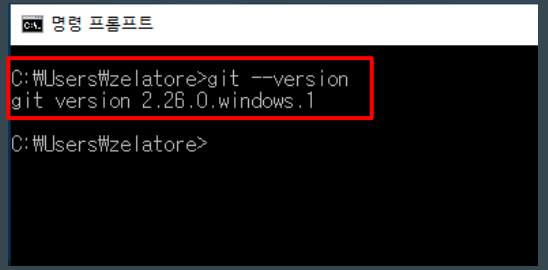


Fig 6. Checking your installation

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Create a Github account

- Go to the Github homepage: link
- Fill in your account information (Fig 4.) and choose free version of account in the sign-up process (Fig 5.)

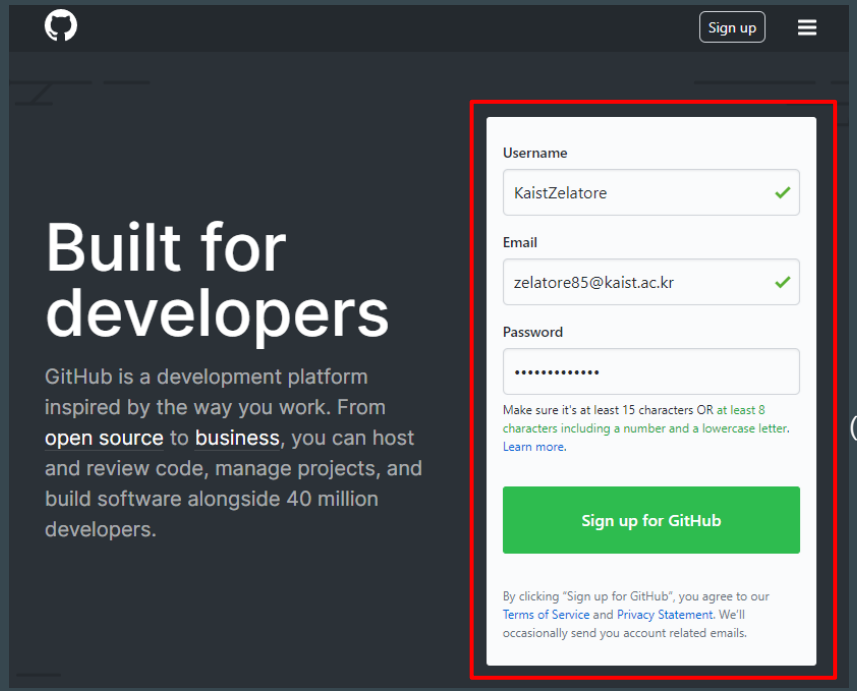


Fig 1. Creating your account

(Some steps)

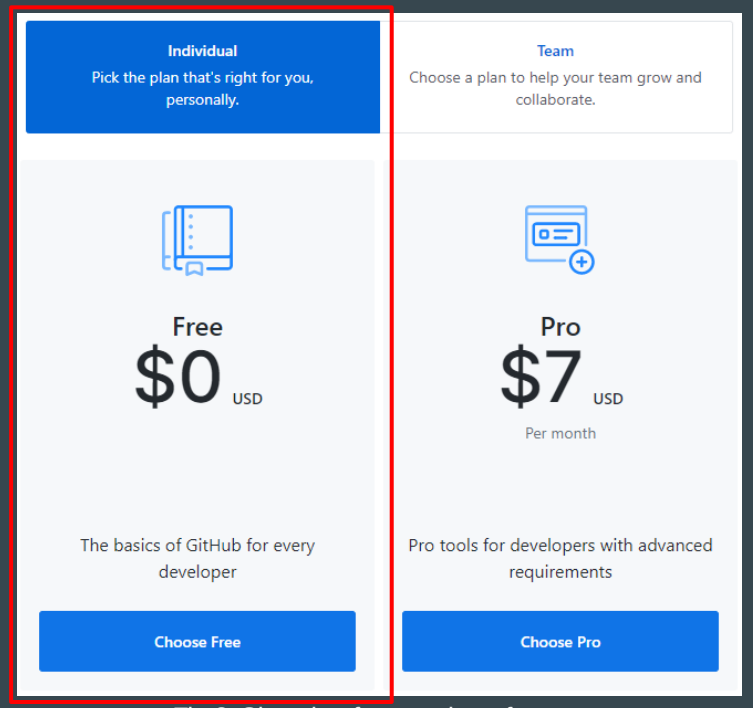


Fig 2. Choosing free version of account

Create a Github account

- Verify your account through email (Fig 3.) and create the first new repository (Fig 4.) after verification
- In step Fig 4. , choose ‘Public’ option and check the box of README initialization

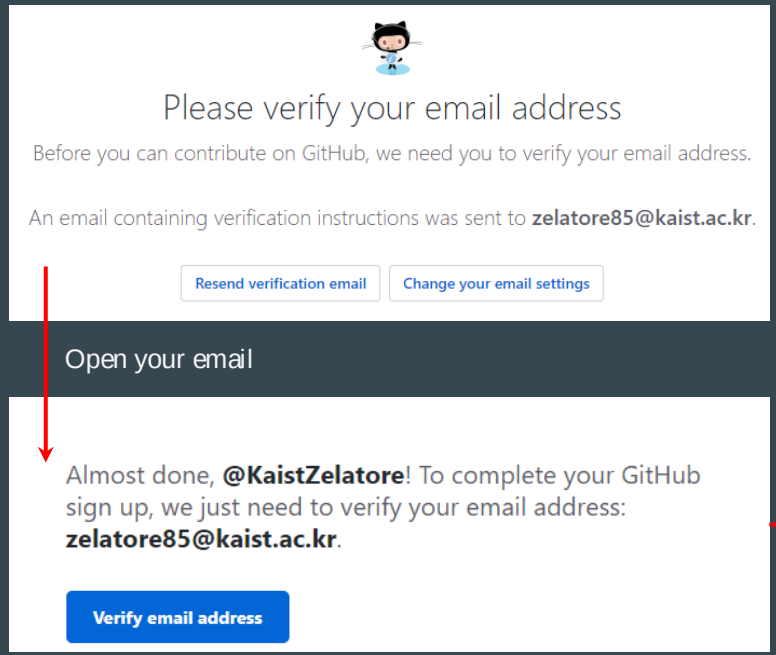


Fig 3. Verification process

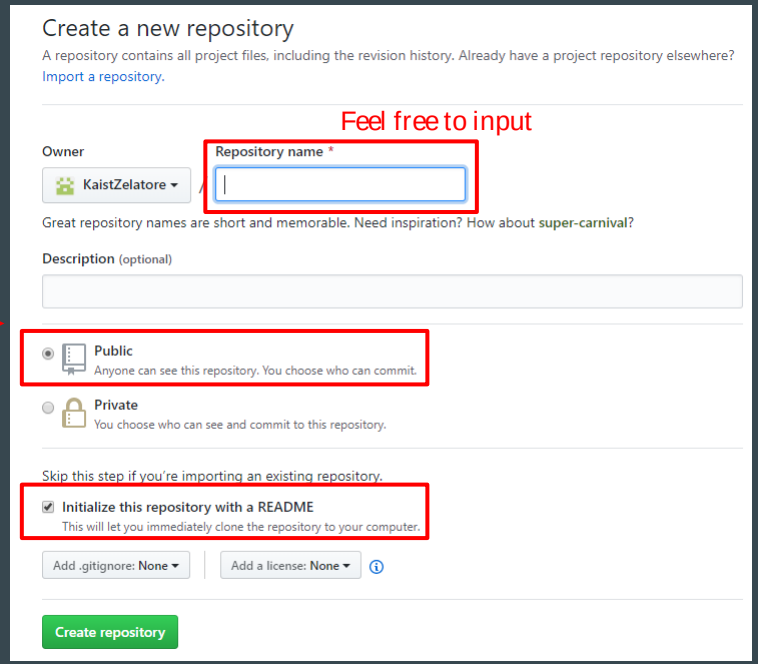


Fig 4. Create a new repository

Create a Github account

- Click the repository you made (Fig 5.) and see the repository to check if it completed successfully (Fig 6.)

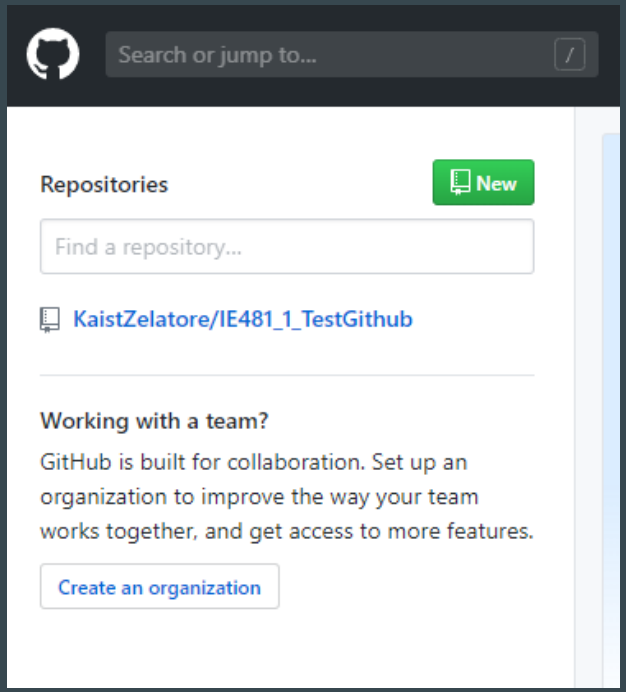


Fig 5. Your repository list

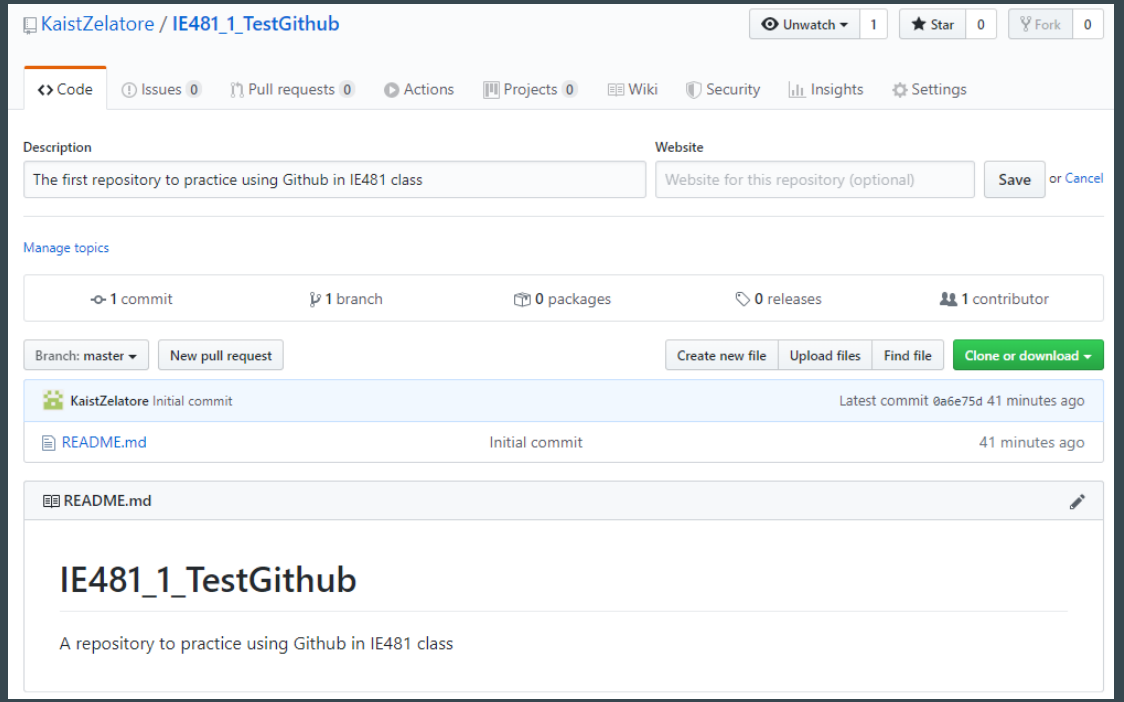


Fig 6. Selected repository view (No any files so far)

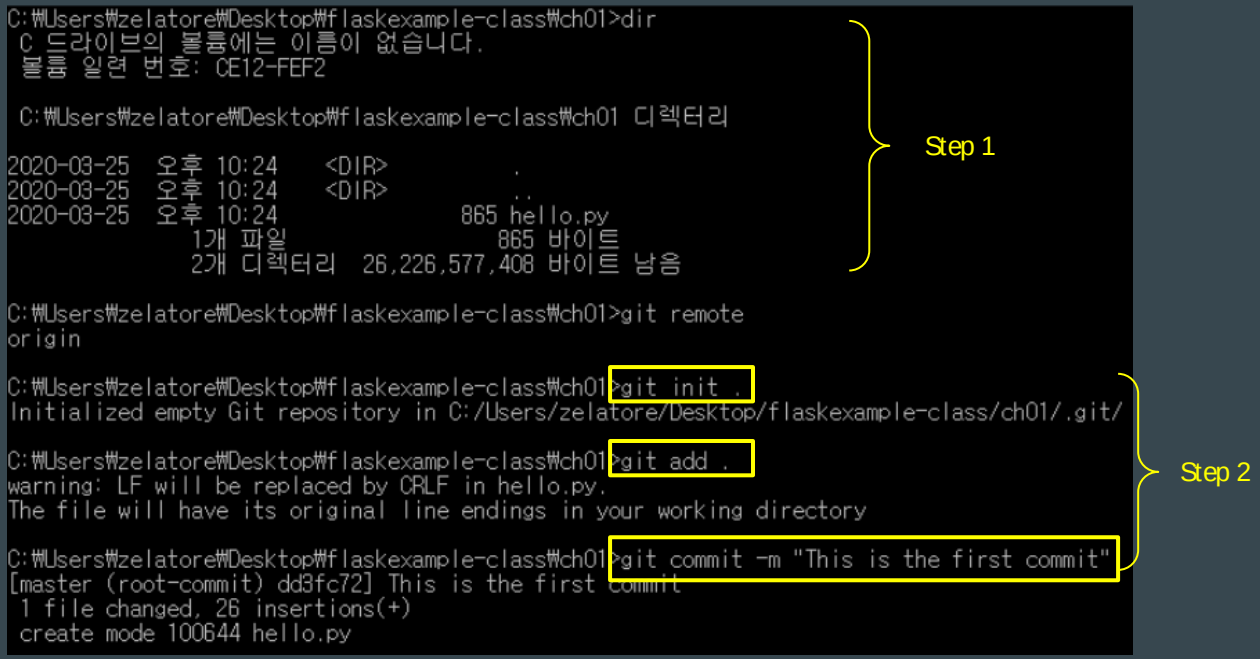
Now, Let’s practice Git and Github!

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Save files to remote server (Local → Remote)

- Step 1: Go to the project directory path to upload
- Step 2: Initialize Git and commit all files to the local repository



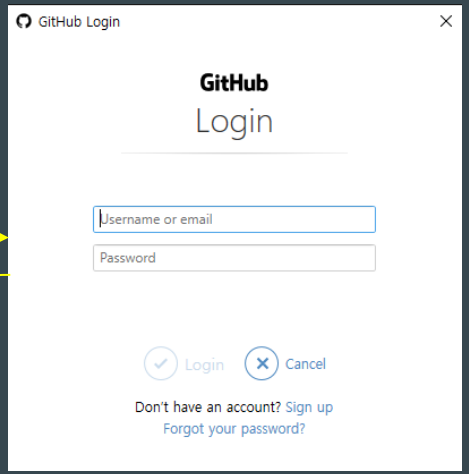
Save files to remote server (Local → Remote)

- Step 3: Choose a remote repository to upload your files
- Step 4: (Login credentials) → push all files

Choose a remote repository

```
C:₩Users₩zelatore₩Desktop₩flaskexample-class₩ch01>git remote add origin https://github.com/K
aistZelatore/TestGit

C:₩Users₩zelatore₩Desktop₩flaskexample-class₩ch01>git remote -v
origin  https://github.com/KaistZelatore/TestGit (fetch)
origin  https://github.com/KaistZelatore/TestGit (push)

C:₩Users₩zelatore₩Desktop₩flaskexample-class₩ch01>git push origin master
```

Login window popped up

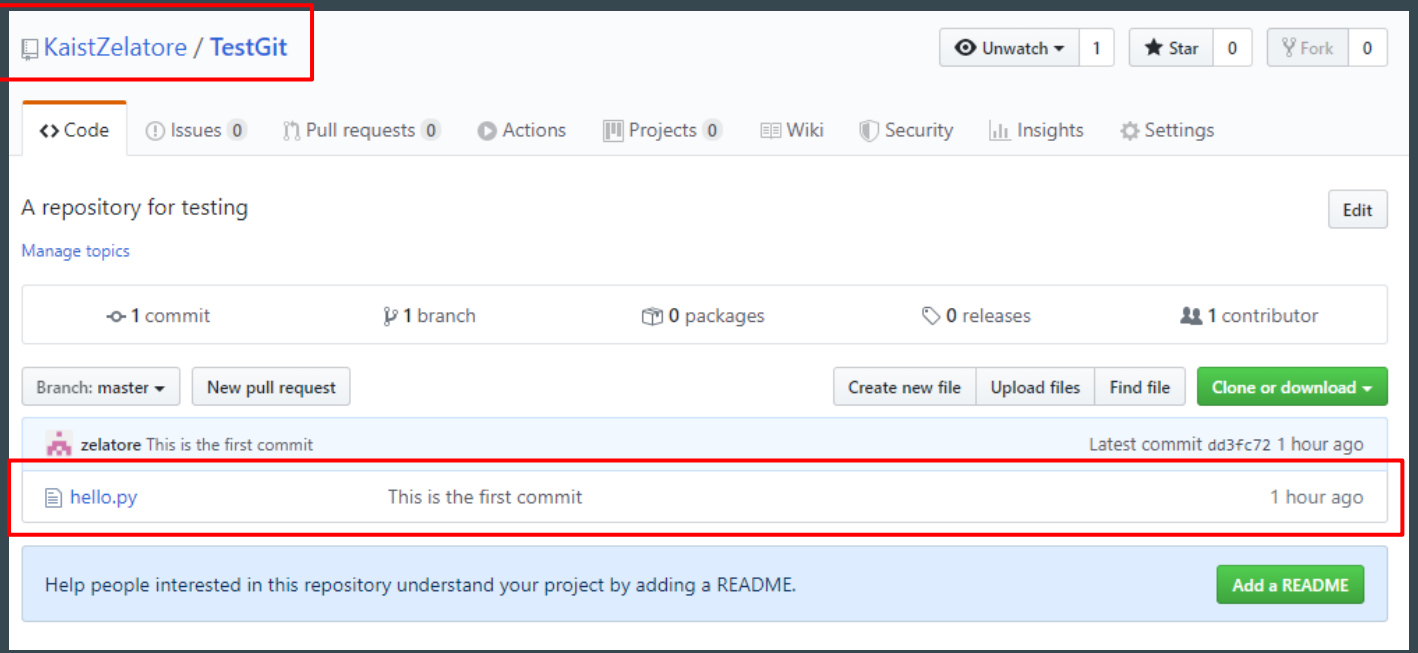



```
C:₩Users₩zelatore₩Desktop₩flaskexample-class₩ch01>git push -f origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 635 bytes | 635.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/KaistZelatore/TestGit
 + 8d8ff7e...dd3fc72 master -> master (forced update)

C:₩Users₩zelatore₩Desktop₩flaskexample-class₩ch01>
```

After login credentials,
Archiving process will be completed

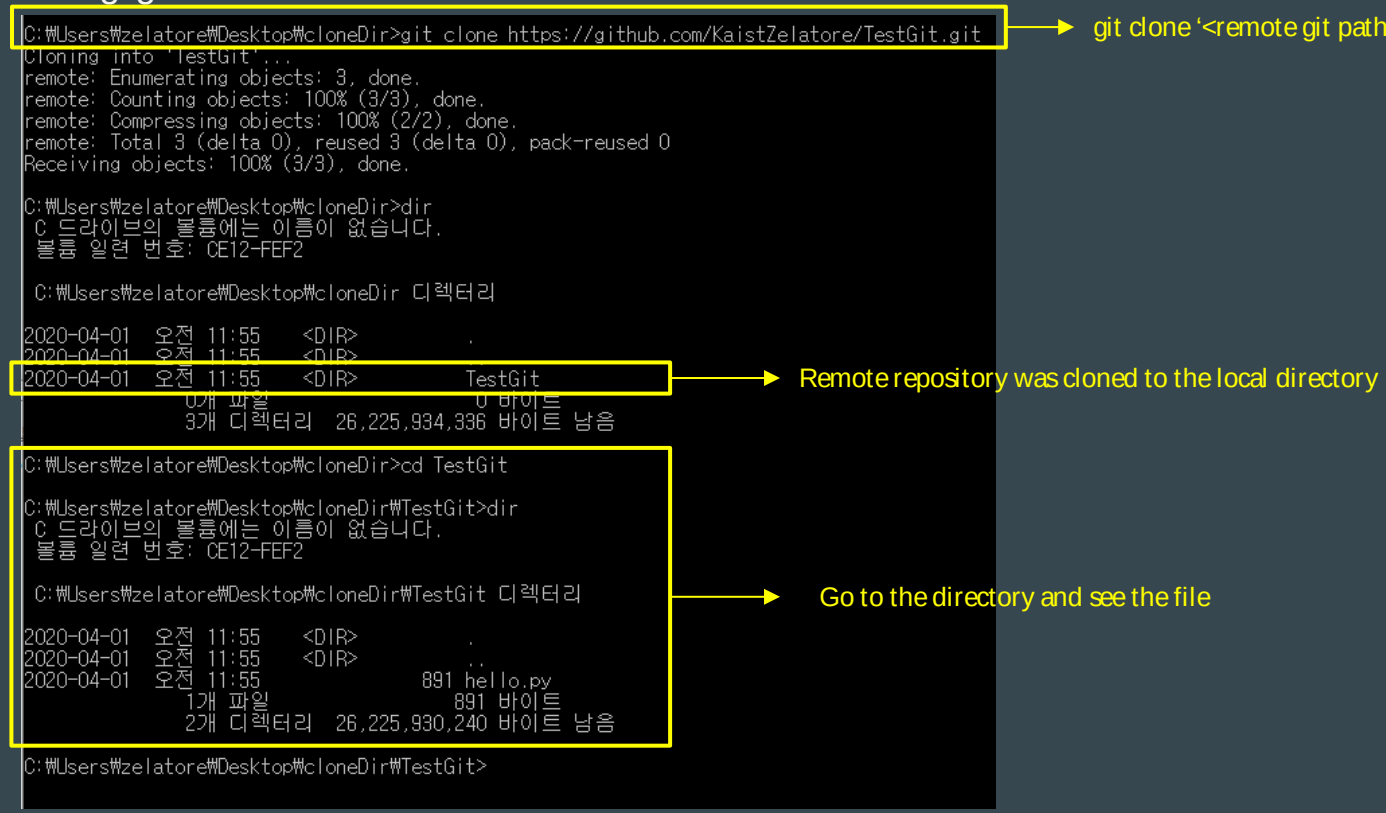
Save files to remote server (Local → Remote)

- Step 5: Check and see the uploaded files



Get project from remote server (Remote → Local)

- Using 'git clone' instructions



git clone '<remote git path>'

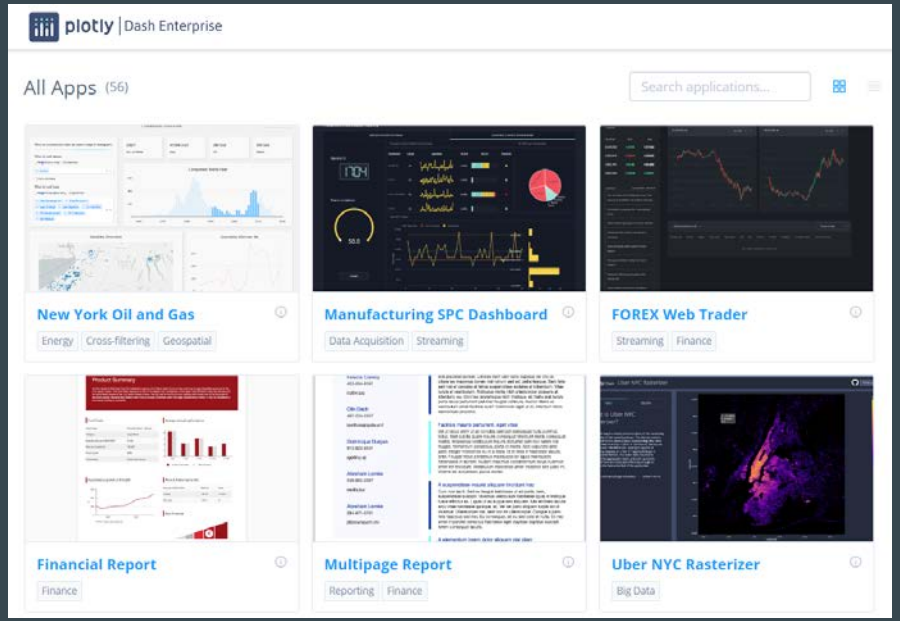
Remote repository was cloned to the local directory

Go to the directory and see the file

Dash

What is Dash?

- Open source python library for creating ‘reactive’ web application
- Integrating with Flask for building a variety of data-driven web application
- Usually used with Plotly library for data analytics app (e.g., data visualization dashboard application)



A variety of examples of Dashboard application using Dash with Plotly

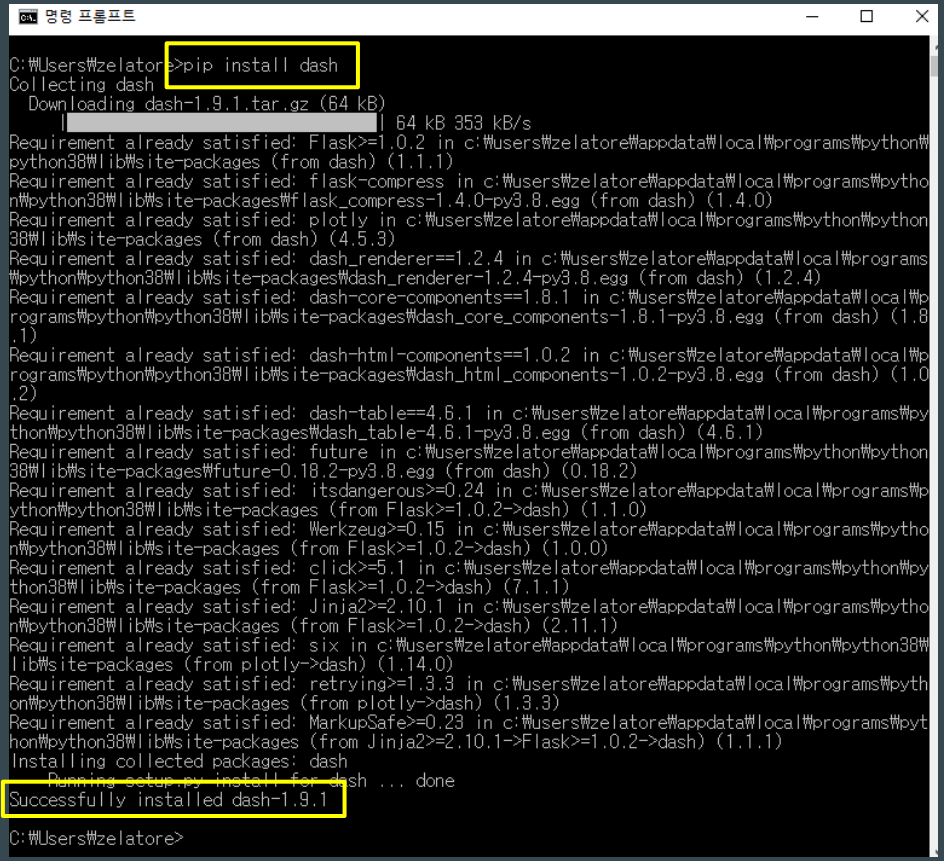
https://dash-gallery.plotly.host/Portal/
https://medium.com/plotly/introducing-dash-5ecf7191b503

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Install Dash

- Just typing the text below on the command line
- ‘pip install dash’



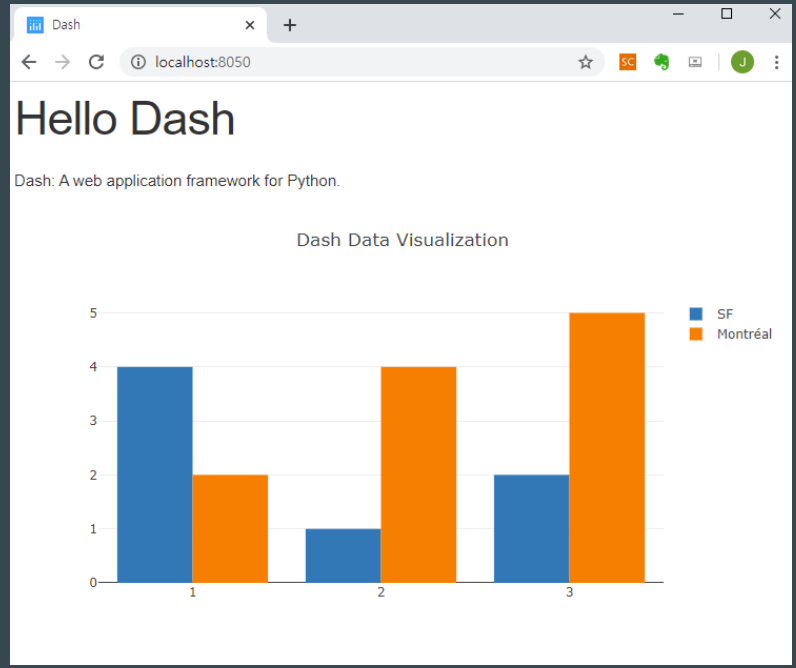
Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

An example of Dash application

- Using 'dash_core_components' to include and use Dash components (e.g., Graph) in a web page
- Using 'dash_html_components' to provide page layout information
- You may use the source code here

```python
# -*- coding: utf-8 -*-
import dash
import dash_core_components as dcc
import dash_html_components as html

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

app.layout = html.Div(children=[
    html.H1(children='Hello Dash'),

    html.Div(children='''
        Dash: A web application framework for Python.
    '''),

    dcc.Graph(
        id='example-graph',
        figure={
            'data': [
                {'x': [1, 2, 3], 'y': [4, 1, 2], 'type': 'bar', 'name': 'SF'},
                {'x': [1, 2, 3], 'y': [2, 4, 5], 'type': 'bar', 'name': u'Montréal'},
            ],
            'layout': {
                'title': 'Dash Data Visualization'
            }
        }
    )
])

if __name__ == '__main__':
    app.run_server(debug=True)
```

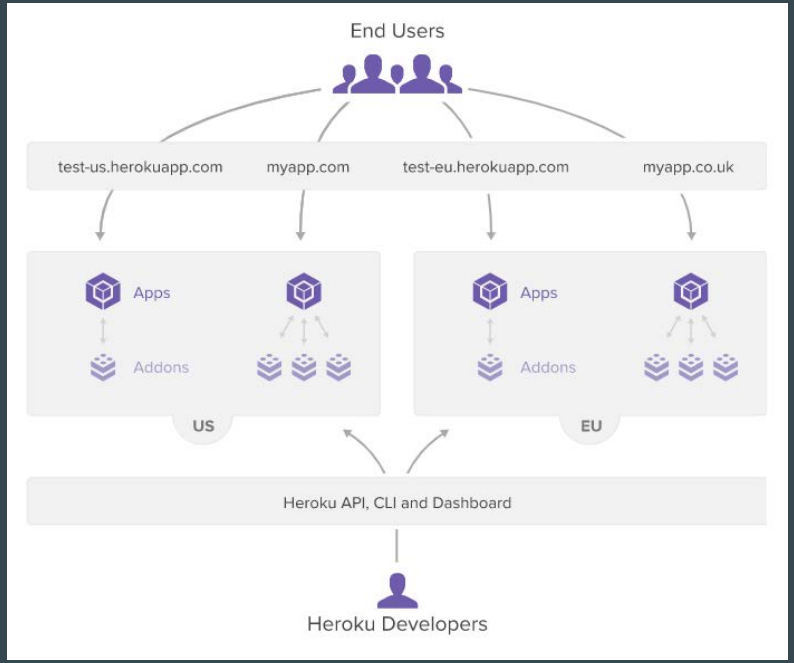


https://dash.plotly.com/layout

Heroku

What is Heroku?

- A cloud platform that helps users to deploy your app
- Support a free cloud web hosting services to run your web application (e.g., Flask app)



https://devcenter.heroku.com/categories/reference
https://blog.heroku.com/europe-region

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Install Heroku and create an account

- Go to the Heroku homepage: link
- Click the sign up button (Fig 1)
- Fill in your information (Fig 2.)

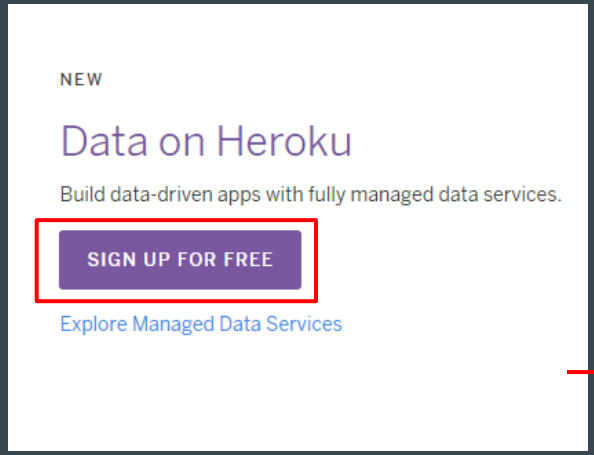


Fig 1. Click sign up button

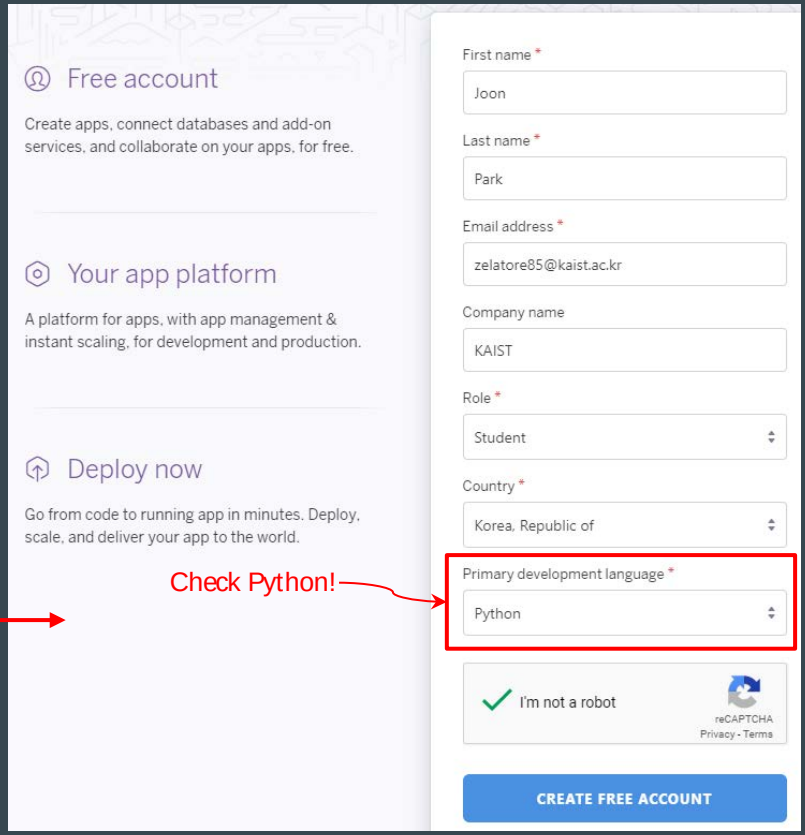


Fig 2. Create your account

Install Heroku and create an account

- Verify your account by email (Fig 3.) and finalize setting up your account (Fig 4.)

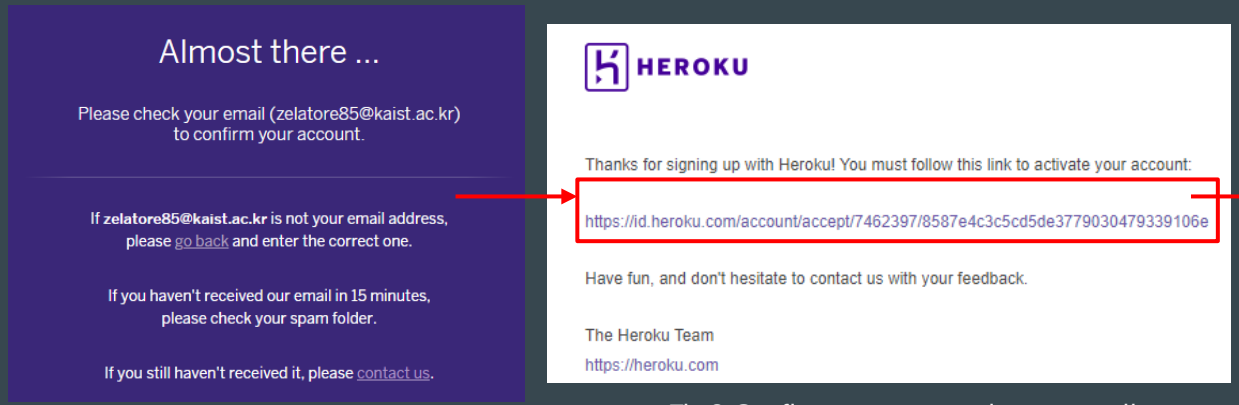


Fig 3. Confirm your account in your email

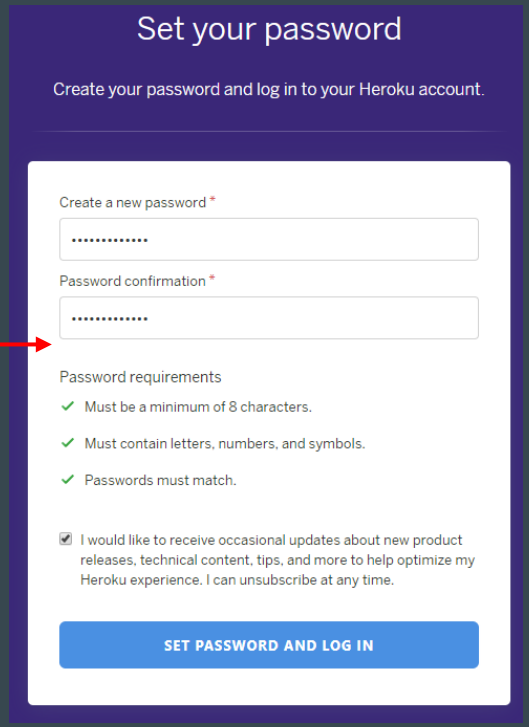


Fig 4. Finalize your account by creating your password

Install Heroku and create an account

- After logging in, install the Heroku (Go to the download page)
- In the download page, choose your OS version and download the executable file (Fig 5.)

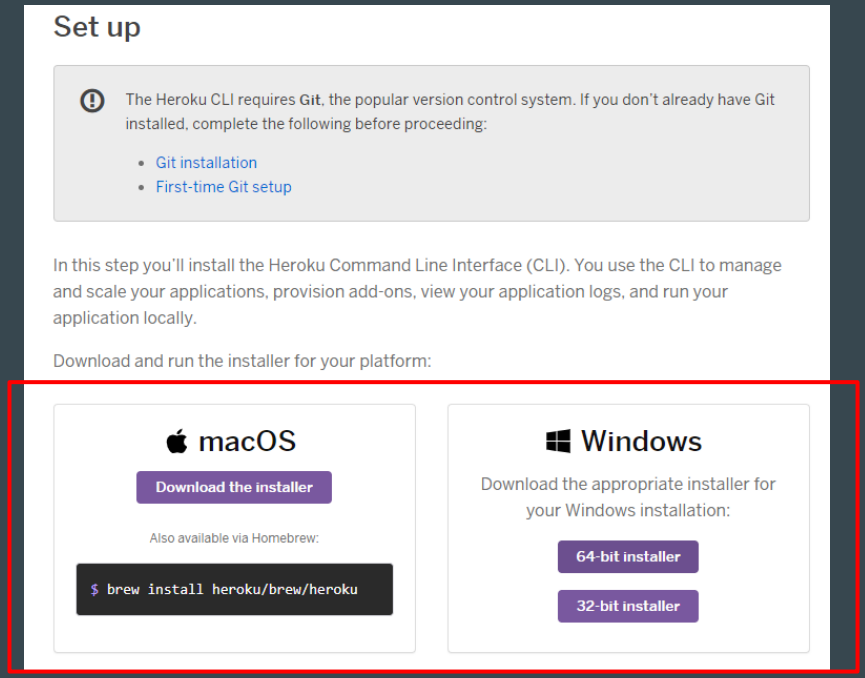
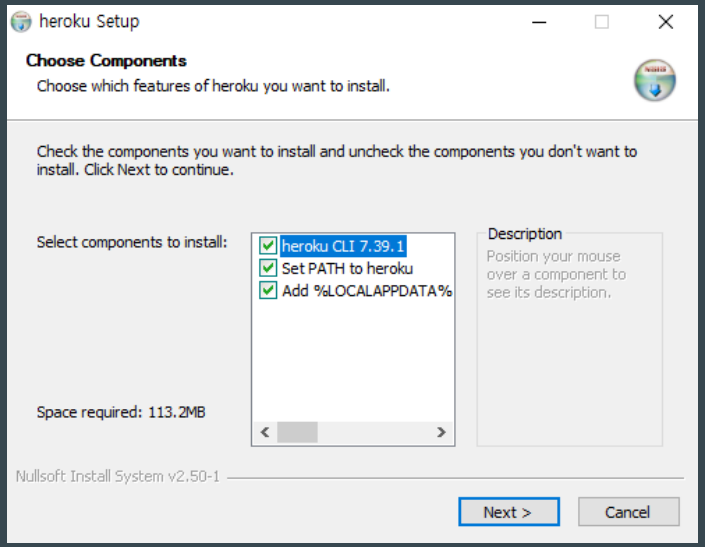


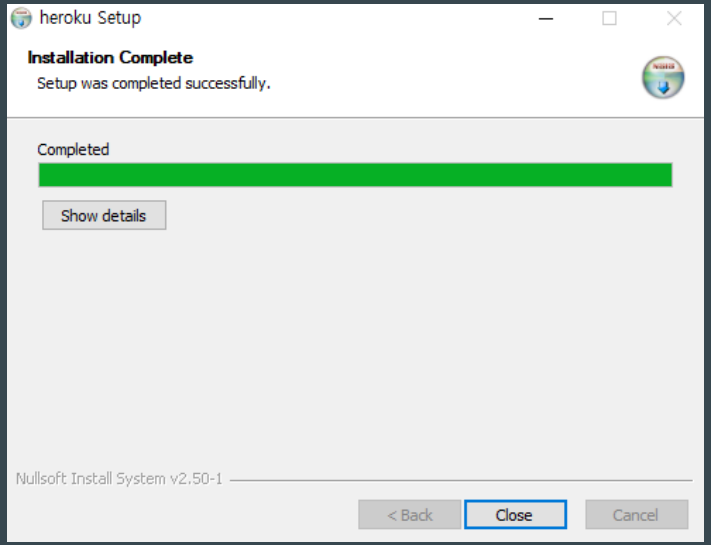
Fig 5. Choose the executable file based on your OS version

Install Heroku and create an account

- Iteratively click the ‘Next’ button, and complete your installation

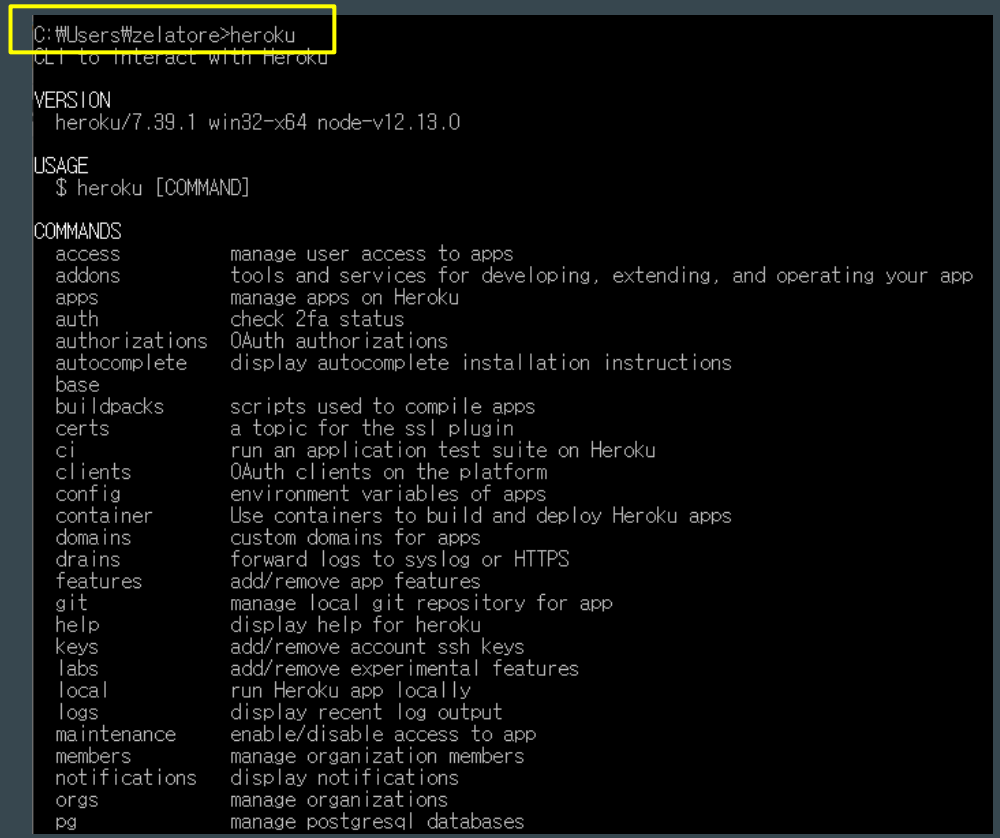


(Some steps)



Install Heroku and create an account

- Check your installation result
- Type ‘heroku’ in a command line (Fig 6.)



Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Configure web hosting for your app using Heroku

- If you complete creating account, you can see the Fig 1.
- Otherwise, you can see the same page (Fig 2.) by logging in at the login page

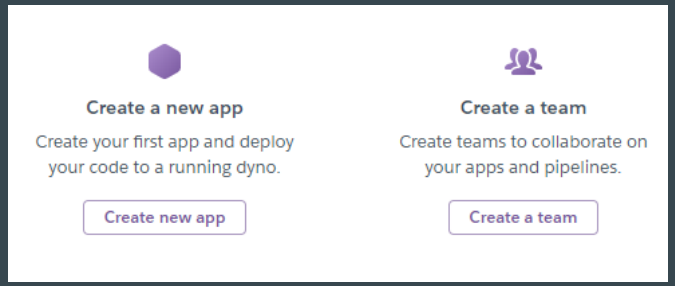


Fig 1 A web page for creating a new app

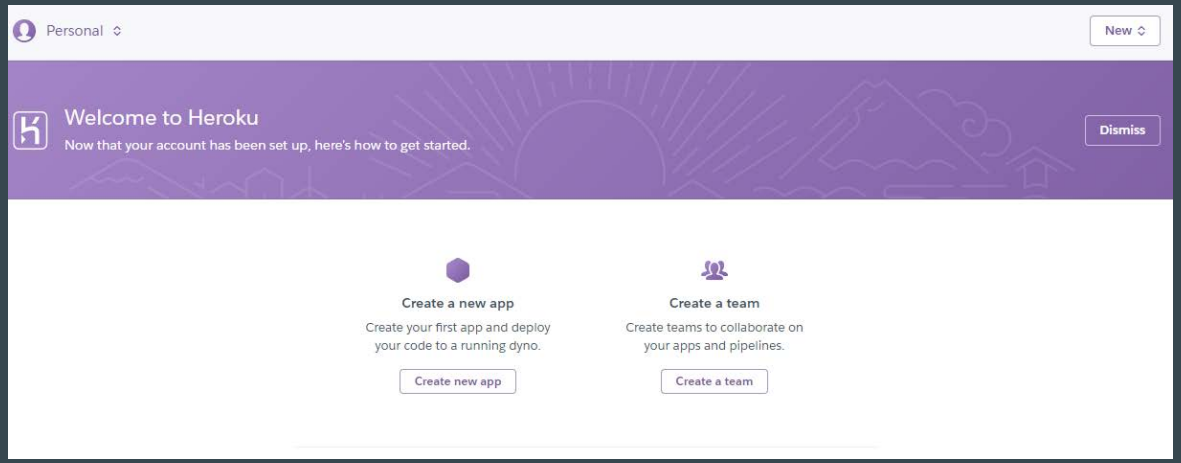


Fig 2. The same as the Fig 1 (Accessed by log in)

Configure web hosting for your app using Heroku

- Click the ‘create new app’ button (Fig 1), then fill in your app for your web hosting
- You are not able to change your region, just keep choosing the default option and click ‘Create App’ button

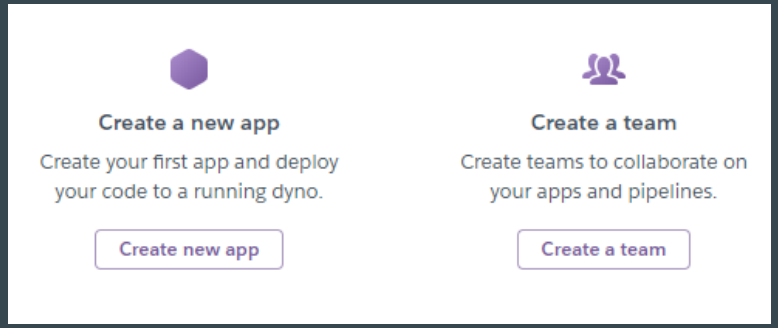


Fig 1. A web page for creating a new app

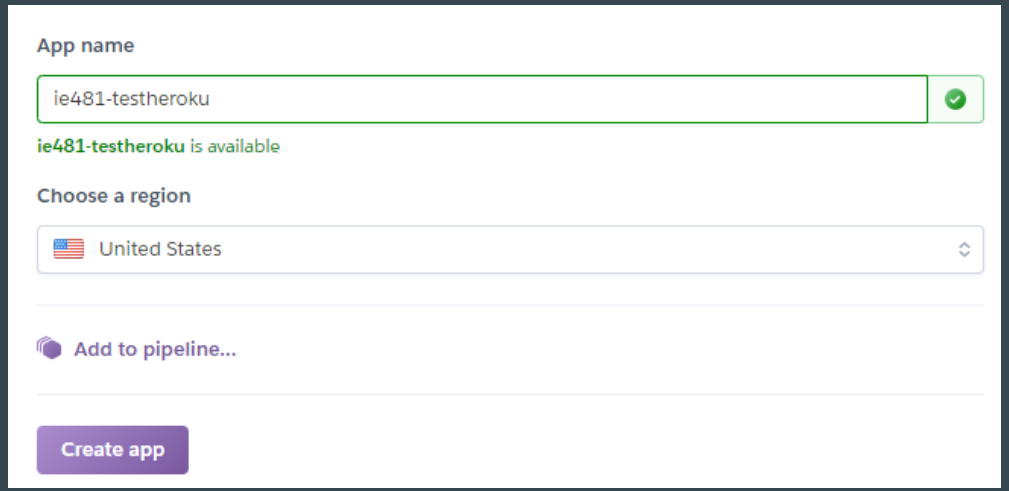


Fig 3. Finalize your web hosting by clicking the button above

Configure web hosting for your app using Heroku

- Now, you can see a dashboard web page configuring your hosting app (Fig 4.)
- By clicking 'Open app' button in Fig 4., you can actually see and confirm your hosted app
  - In the Heroku, all of the hosted app can be accessed by url like '[your app name].herokuapp.com'
- In heroku, you may create your own app up to five
  - For creating the 6th app, you must remove an existing app)

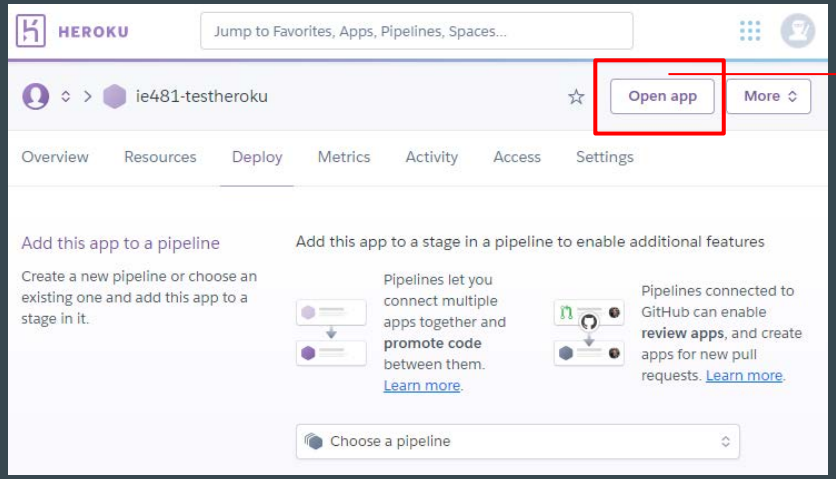


Fig 4. Dashboard page for configuring hosting app

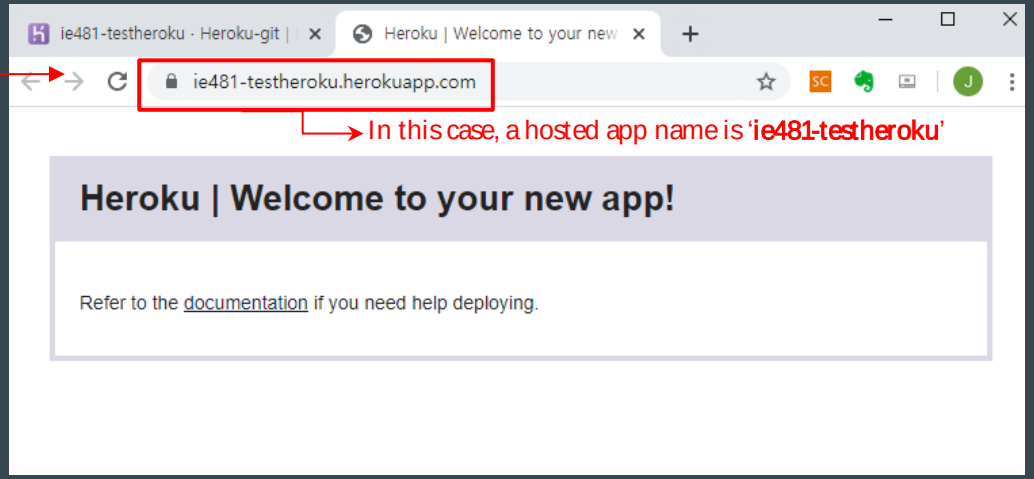


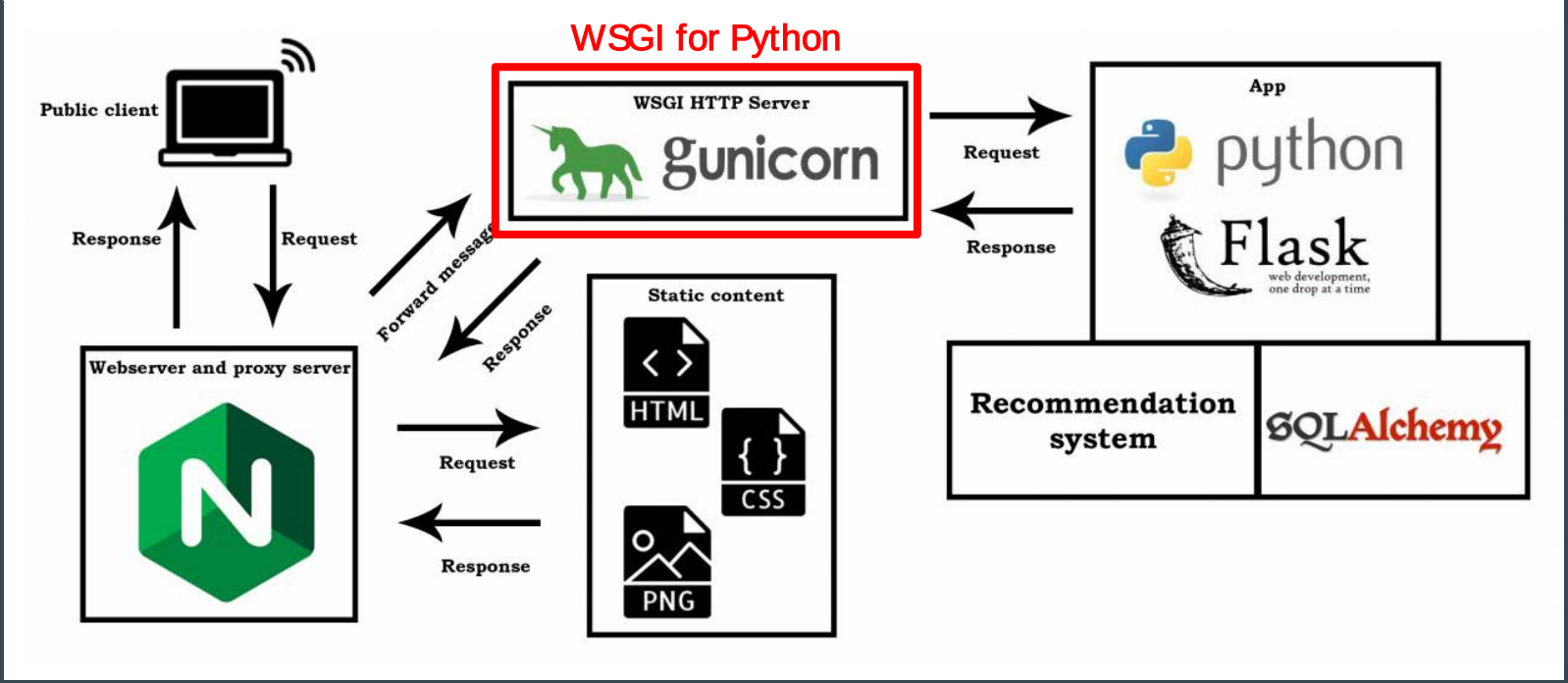
Fig 5. Hosted app service

Contents

- Git/Github
  - Install and configure Git
  - Create a Github account
  - Guideline for using Git and Github
- Dash
  - Install and configure Dash
  - Test a basic example of Dash to understand how it works (prerequisite learning for Lab #3)
- Heroku
  - Install Heroku and create an account
  - A sample example of web hosting using Heroku
  - Guideline for hosting web services using Python with Heroku

Process of python application web hosting

- Needs to have Web Server Gateway Interface(WSGI) to communicate between web server and python app
- One kind of WSGI for python is 'gunicorn' → Interpret the python script to communicate with web server



https://scripties.uba.uva.nl/document/659760

For python web application hosting, We need to have 3 modules

Web App
(Python code)

WSGI
(e.g., gunicorn)

Hosting Server
(e.g., Heroku)

Guideline for hosting web services using Python with Heroku

- Step 1: Install gunicorn modules → 'pip install gunicorn' in command line
- Step 2: Create a python code executed on the server (Here, we use a Dash example code)

```
C:₩Users₩zelatore>pip install gunicorn
Collecting gunicorn
  Using cached gunicorn-20.0.4-py2.py3-none-any.whl (77 kB)
Requirement already satisfied: setuptools>=3.0 in c:₩users₩zelatore₩appdata₩l
ocal₩programs₩python₩python38₩lib₩site-packages (from gunicorn) (41.2.0)
Installing collected packages: gunicorn
Successfully installed gunicorn-20.0.4

C:₩Users₩zelatore>
```

```python
# -*- coding: utf-8 -*-

import os
import flask
import dash
import dash_core_components as dcc
import dash_html_components as html

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

server = flask.Flask(__name__)
server.secret_key = os.environ.get('secret_key', str(randint(0, 1000000)))
app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

app.layout = html.Div(children=[
    html.H1(children='Hello Dash'),

    html.Div(children='''
        Dash: A web application framework for Python.
    '''),

    dcc.Graph(
        id='example-graph',
        figure={
            'data': [
                {'x': [1, 2, 3], 'y': [4, 1, 2], 'type': 'bar', 'name': 'SF'},
                {'x': [1, 2, 3], 'y': [2, 4, 5], 'type': 'bar', 'name': u'Montréal'},
            ],
            'layout': {
                'title': 'Dash Data Visualization'
            }
        }
    )
])

if __name__ == '__main__':
    app.run_server(debug=True)
```

Guideline for hosting web services using Python with Heroku

- Step 3: In command line, go to the directory including python source code that you created (Fig 1)
- Step 4: To host python application using Heroku, we need to have 2 files
  - **'Procfile'**: To let Heroku know the sequence of instructions for executing python code
  - **'requirements.txt'**: To let Heroku know which python packages are needed to run the application

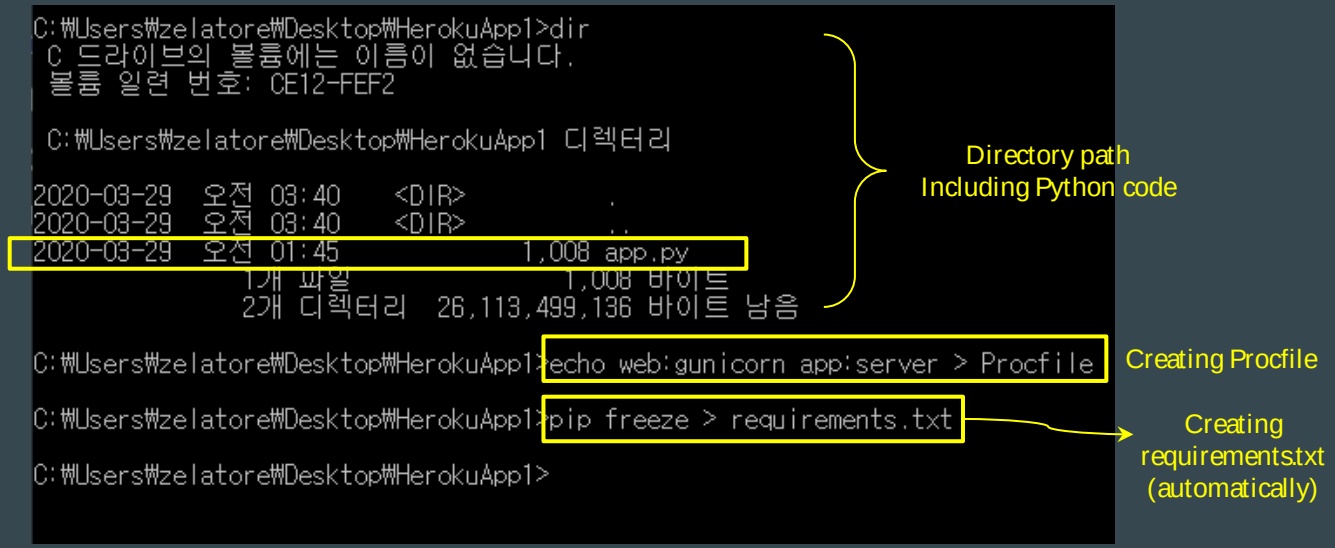


Fig 1 Guideline for creating 2 dependency files

```
Flask==1.1.1
flask-assets==2.0
flask-compress==1.4.0
future==0.18.2
gunicorn==20.0.4
itsdangerous==1.1.0
Jinja2==2.11.1
MarkupSafe==1.1.1
numpy==1.18.1
pandas==1.0.1
pathlib==1.0.1
plotly==4.5.4
plotly-flask-tutorial==0.0.1
python-dateutil==2.8.1
pytz==2019.3
retrying==1.3.3
six==1.14.0
```

An example of requirements.txt

Guideline for hosting web services using Python with Heroku

- Step 5: Initialize, add, and commit the python source code using Git
- Step 6: Create your hosting app
- Step 6: Push all resource files (Procfile, requirements.txt, and python code) to a Heroku remote github

```
C:₩Users₩zelatore₩Desktop₩HerokuApp1>git init
Initialized empty Git repository in C:/Users/zelatore/Desktop/HerokuApp1/.git/

C:₩Users₩zelatore₩Desktop₩HerokuApp1>git add .

C:₩Users₩zelatore₩Desktop₩HerokuApp1>git commit -m "The first dash app commit"
[master (root-commit) e9c46c4] The first dash app commit
 3 files changed, 63 insertions(+)
 create mode 100644 Procfile
 create mode 100644 app.py
 create mode 100644 requirements.txt

C:₩Users₩zelatore₩Desktop₩HerokuApp1>heroku create ie481-dashapptest
Creating ⬢ ie481-dashapptest... done
https://ie481-dashapptest.herokuapp.com/ | https://git.heroku.com/ie481-dashapptest.git

C:₩Users₩zelatore₩Desktop₩HerokuApp1>heroku git:remote -a ie481-dashapptest
set git remote heroku to https://git.heroku.com/ie481-dashapptest.git

C:₩Users₩zelatore₩Desktop₩HerokuApp1>git push heroku master
```

Archiving all files to repository using Git

Creating an Heroku app (hosting name)

Creating remote repository for Heroku app

Pushing all files to remote repository

Guideline for hosting web services using Python with Heroku

- Step 6: Push all resource files (Procfile, requirements.txt, and python code) to a Heroku remote github
- Step 7: Test your web hosting result through web browser (E.g., '[your app name].herokuapp.com')

Deploying process to the Heroku

```
remote:          Stored in directory: /tmp/pip-ephem-wheel-cache-bf1tsclk/wh
eels/ac/cb/8a/b27bf6323e2f4c462dcbf77d70b7c5e7868a7fbe12871770cf
remote:        Successfully built dash dash-core-components dash-html-compon
ents dash-renderer dash-table flask-compress future pathlib retrying
remote:        Installing collected packages: click, Werkzeug, MarkupSafe, J
inja2, itsdangerous, Flask, flask-compress, six, retrying, plotly, dash-rend
erer, dash-core-components, dash-html-components, dash-table, future, dash,
webassets, flask-assets, gunicorn, numpy, python-dateutil, pytz, pandas, pat
hlib
remote:        Successfully installed Flask-1.1.1 Jinja2-2.11.1 MarkupSafe-1
.1.1 Werkzeug-1.0.0 click-7.1.1 dash-1.9.1 dash-core-components-1.8.1 dash-h
tml-components-1.0.2 dash-renderer-1.2.4 dash-table-4.6.1 flask-assets-2.0 f
lask-compress-1.4.0 future-0.18.2 gunicorn-20.0.4 itsdangerous-1.1.0 numpy-1
.18.1 pandas-1.0.1 pathlib-1.0.1 plotly-4.5.4 python-dateutil-2.8.1 pytz-201
9.3 retrying-1.3.3 six-1.14.0 webassets-2.0
remote: -----> Discovering process types
remote:
remote: -----> Compressing...
remote:        Done: 97M
remote: -----> Launching...
remote:        Released v3
remote:        https://ie481-dashapptest.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/ie481-dashapptest.git
 * [new branch]      master -> master

C:₩Users₩zelatore₩Desktop₩HerokuApp1>
```