AWS EC2 Linux 환경에 Flask 웹서버 구축하기

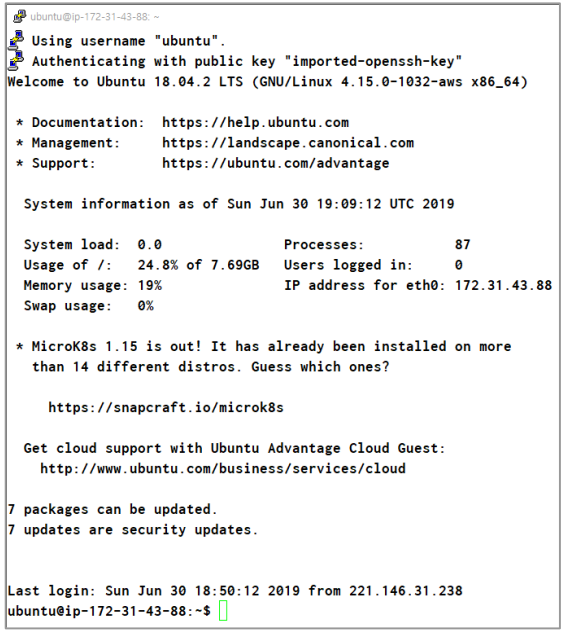
Jin Won,

School of Computer Engineering,

Hansung University

Contents

- PuTTY를 사용하여 Windows PC에서 AWS EC2 Linux 서버로 접속한다.
- 터미널에 다음 페이지부터 등장하는 명령어들을 입력한다.



Contents

- 기설치된 서비스 업데이트 및 업그레이드 실시

```
ubuntu@ip-172-31-43-88:~$ sudo apt-get update
ubuntu@ip-172-31-43-88:~$ sudo apt-get upgrade
```

- python 설치

```
ubuntu@ip-172-31-43-88:~$ sudo apt-get install python3.7
```

- pip(각종 python 라이브러리 설치를 도와주는 도구) 설치

```
ubuntu@ip-172-31-43-88:~$ sudo apt-get install python3-pip
```

Contents

- python3 가상 환경 만들기

```
ubuntu@ip-172-31-43-88:~$ sudo apt-get install python3-venv
ubuntu@ip-172-31-43-88:~$ mkdir OnlineDataAnalyzer
ubuntu@ip-172-31-43-88:~$ cd OnlineDataAnalyzer/
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ python3 -m venv godjin
```

Contents

- Flask 설치

```
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ pip install Flask
```

- boto3 설치

```
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ pip install boto3
```

- nltk 설치

```
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ pip install -U nltk
```

Contents

- FileZilla를 통해 Windows PC에서 작업한 코드들을 Linux 서버로 가져온다.

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ ls
__main__.py  __pycache__  application  godjin  venv
```

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer/application$ ls
__init__.py  __pycache__  nlp  static  templates
```

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer/application/nlp$ ls
EntityDetector.py   PhraseExtractor.py    __init__.py  accessKeys.csv
NGramsExtractor.py  WordPairExtractor.py  __pycache__
```

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer/application/static$ ls
download.js  main.js
```

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer/application/templates$ ls
index.html
```

Contents

- Flask 웹서버를 실행하는 방법은 두 가지가 있다.
- 첫번째 방법: user가 PuTTY로 접속하고 있는 동안에만 프로세스가 살아있다.

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ source godjin/bin/activate
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ python __main__.py
 * Serving Flask app "application" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8000/ (Press CTRL+C to quit)
```

Contents

- 두번째 방법: user 접속 유무와 상관없이 프로세스가 살아있다.
  백그라운드에서 돌아간다.

```
ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ source godjin/bin/activate
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$ python __main__.py & disown
[1] 2394
(godjin) ubuntu@ip-172-31-43-88:~/OnlineDataAnalyzer$  * Serving Flask app "application" (lazy loadi
ng)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:8000/ (Press CTRL+C to quit)
```

Contents

- 두번째 방법을 채택한 경우, 프로세스를 종료하려면 아래를 참고한다.

```
ubuntu@ip-172-31-43-88:~$ ps -aux | grep python
root         840  0.0  1.7 170880 17396 ?        Ssl  18:14   0:00 /usr/bin/pytho
3 /usr/bin/networkd-dispatcher --run-startup-triggers
root         865  0.0  2.0 187712 20208 ?        Ssl  18:14   0:00 /usr/bin/pytho
3 /usr/share/unattended-upgrades/unattended-upgrade-shutdown --wait-for-signal
ubuntu      2251  0.8  4.8 197528 49292 ?        S    18:49   0:00 python __main__
.py
ubuntu      2389  0.0  0.0  14856  1004 pts/1    S+   18:50   0:00 grep --color=au
to python
ubuntu@ip-172-31-43-88:~$ kill -9 2251
```

Contents

- 웹 페이지 접속

  ‘퍼블릭 DNS(IPv4):포트번호’ or ‘IPv4 퍼블릭 IP:포트번호’