스파르타코딩클럽 7기 6주차 - 학습 자료

[수업 목표]

1. EC2 서버를 구매 & 접속한다.
2. 리눅스에서 파이썬 가상환경을 켜고/끌 수 있다.
3. EC2 서버에서 flask 서버를 돌릴 수 있다.

📢: 실습 시간

* 강의 상황에 따라, 시간은 유동적일 수 있습니다.

전반 3시간

[2시간]: 수업 (서버 구매부터 간단한 flask 서버 올리기까지)

▼ 1) 6~8주차 수업 설명: "웹서비스를 런칭하기 위한 작업들"

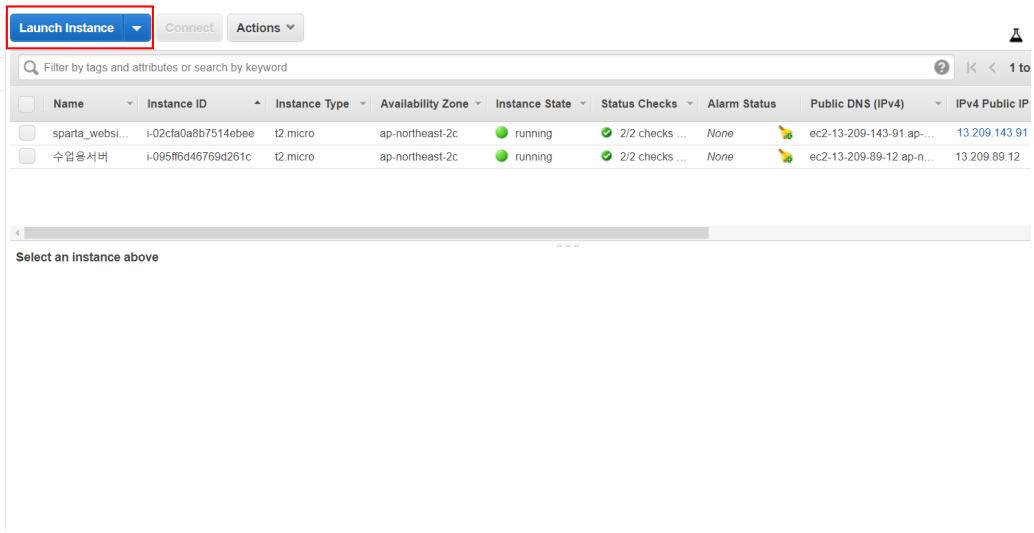
- 6주차(오늘): AWS에서 EC2 서버 구매하기~EC2 서버에서 간단한 flask 서버 실행해보기
- 7주차: mongoDB 설치하기+5주차 완성본을 서버에서 실행하기
- 8주차: 도메인 붙이기, 포트 번호 떼기

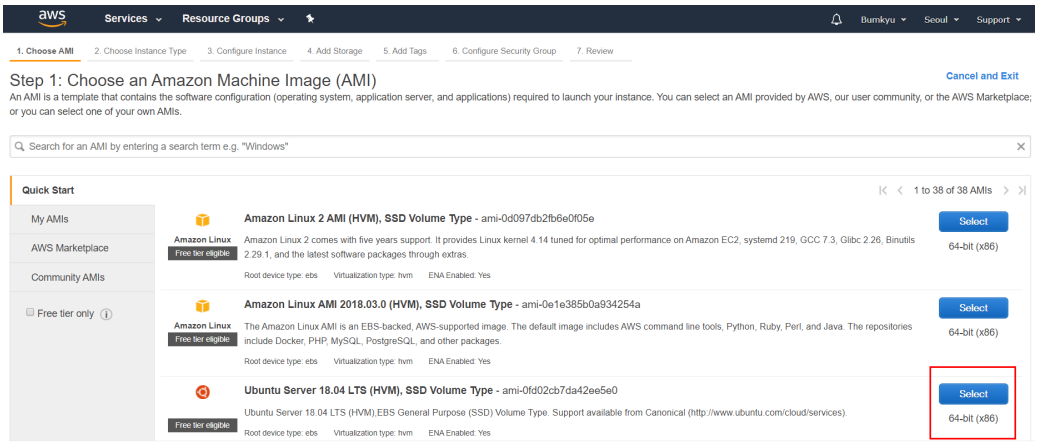
▼ 2) EC2 서버 구매하기

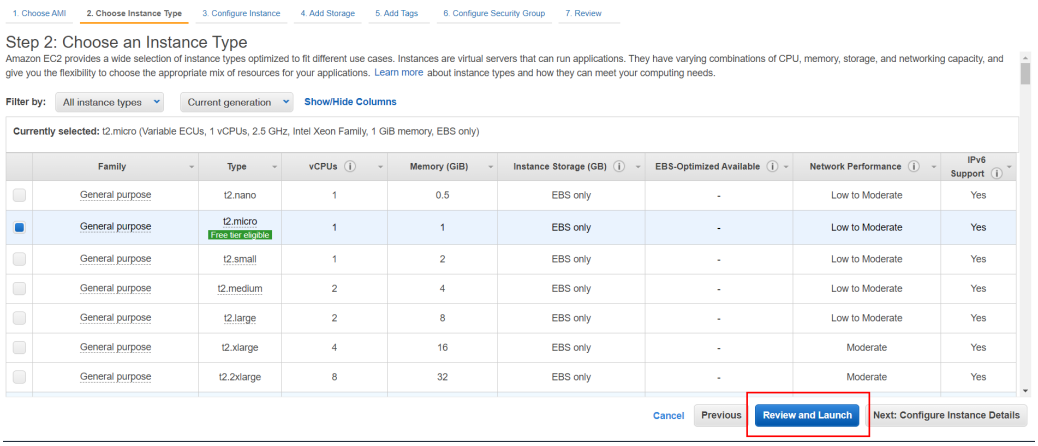
▼ AWS EC2 서버 사기

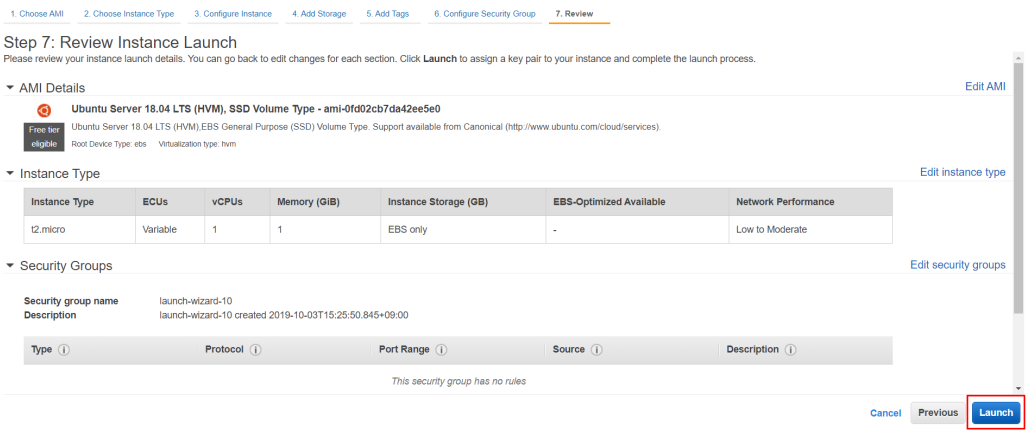
- https://ap-northeast-2.console.aws.amazon.com/ec2/v2/home?region=ap-northeast-2
  (Seoul Region으로 들어가기)

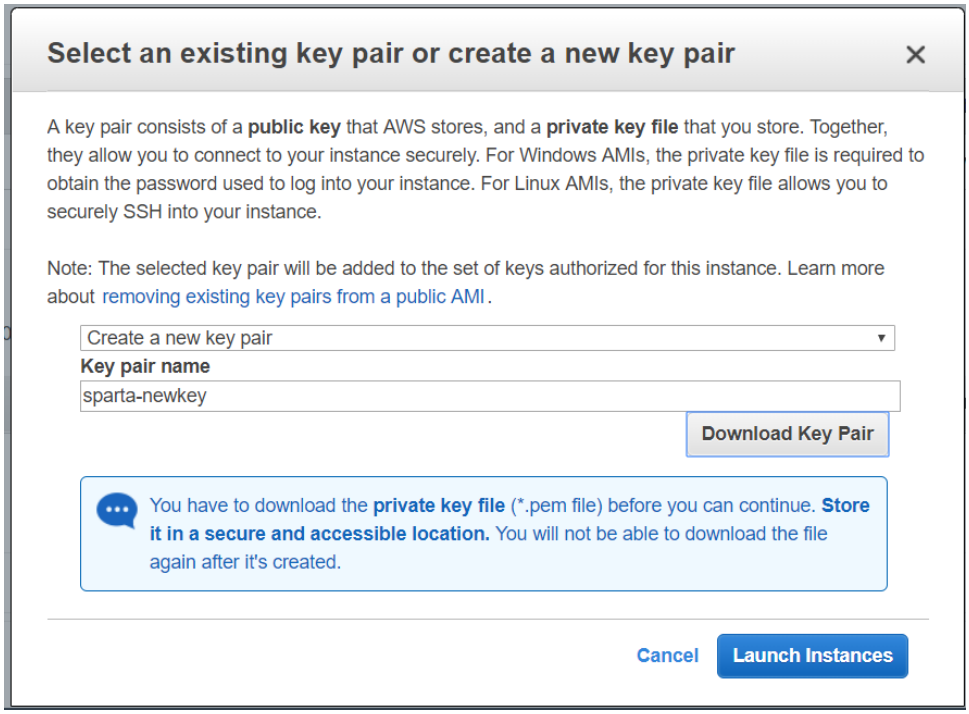
▼ 구매 화면들 따라하기


Launch Instance
Connect
Actions
Filter by tags and attributes or search by keyword
Name
Instance ID
Instance Type
Availability Zone
Instance State
Status Checks
Alarm Status
Public DNS (IPv4)
IPv4 Public IP
sparta_websi...
i-02cfa0a8b7514ebee
t2.micro
ap-northeast-2c
running
2/2 checks ...
None
ec2-13-209-143-91.ap-...
13.209.143.91
수업용서버
i-095ff6d46769d261c
t2.micro
ap-northeast-2c
running
2/2 checks ...
None
ec2-13-209-89-12.ap-n...
13.209.89.12
Select an instance above


Services
Resource Groups
Bumkyu
Seoul
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 1: Choose an Amazon Machine Image (AMI)
Cancel and Exit
An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.
Search for an AMI by entering a search term e.g. "Windows"
Quick Start
1 to 38 of 38 AMIs
My AMIs
AWS Marketplace
Community AMIs
Free tier only
Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0d097db2fb6e0f05e
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
Root device type: ebs
Virtualization type: hvm
ENA Enabled: Yes
Select
64-bit (x86)
Amazon Linux AMI 2018.03.0 (HVM), SSD Volume Type - ami-0e1e385b0a934254a
The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.
Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0fd02cb7da42ee5e0
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).
Free tier eligible


1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 2: Choose an Instance Type
Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.
Filter by:
All instance types
Current generation
Show/Hide Columns
Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)
Family
Type
vCPUs
Memory (GiB)
Instance Storage (GB)
EBS-Optimized Available
Network Performance
IPv6 Support
General purpose
t2.nano
1
0.5
EBS only
-
Low to Moderate
Yes
t2.micro
Free tier eligible
1
1
EBS only
-
Low to Moderate
Yes
t2.small
1
2
EBS only
-
Low to Moderate
Yes
t2.medium
2
4
EBS only
-
Low to Moderate
Yes
t2.large
2
8
EBS only
-
Low to Moderate
Yes
t2.xlarge
4
16
EBS only
-
Moderate
Yes
t2.2xlarge
8
32
EBS only
-
Moderate
Yes
Cancel
Previous
Review and Launch
Next: Configure Instance Details





▼ 3) EC2에 접속하기

▼ 4) 간단한 리눅스 명령어 연습하기

- 리눅스는 윈도우 같지 않아서, '쉘 명령어'를 통해 OS를 조작한다. (일종의 마우스 역할)

[가장 많이 쓰는 몇 가지 명령어]

```
ls: 내 위치의 모든 파일을 보여준다.

pwd: 내 위치(폴더의 경로)를 알려준다.

mkdir: 내 위치 아래에 폴더를 하나 만든다.

cd [갈 곳]: 나를 [갈 곳] 폴더로 이동시킨다.
```

```
cd .. : 나를 상위 폴더로 이동시킨다.

cp -r [복사할 것] [붙여넣기 할 것]: 복사 붙여넣기

rm -rf [지울 것]: 지우기

sudo [실행 할 명령어]: 명령어를 관리자 권한으로 실행한다.
sudo su: 관리가 권한으로 들어간다. (나올때는 exit으로 나옴)
```

- 함께 해보기: 임의의 폴더를 만들어보고, 지워보기

[이 밖에도 아주 많은 명령어들]

- 예 - https://zzsza.github.io/development/2017/12/04/linux-1/
- 역시 일일이 다 외울 필요 없고, 찾아보면 됩니다.

▼ Tips !

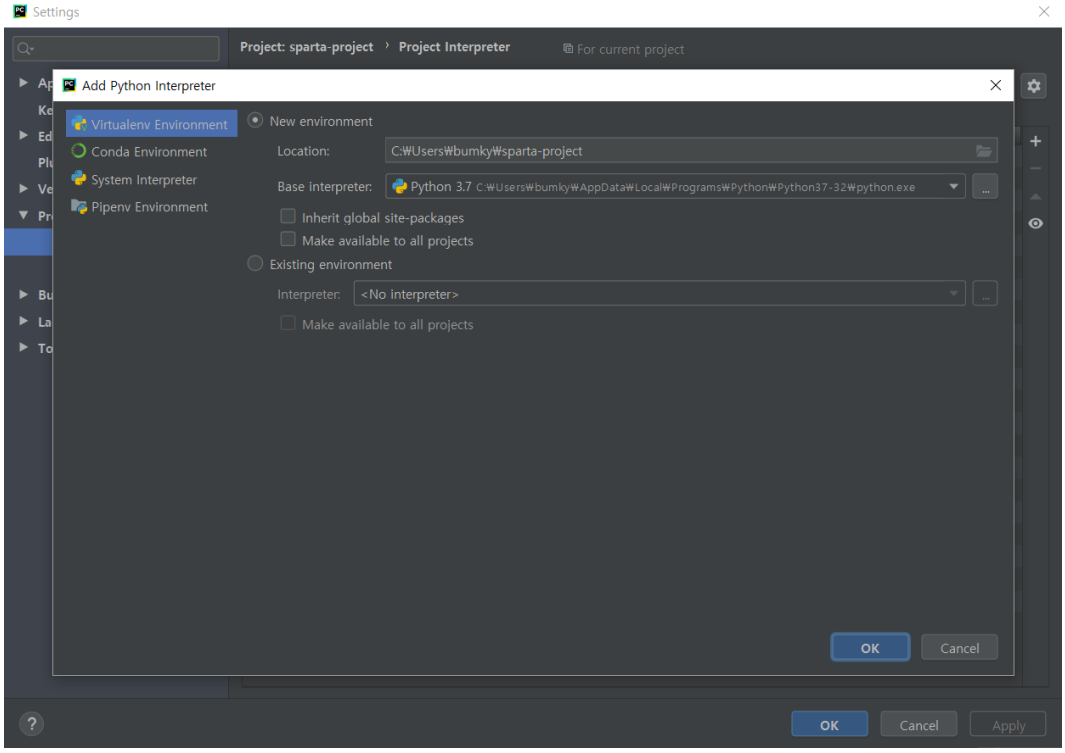
- 리눅스 커널에 붙여넣기를 할 때는 마우스 오른쪽 버튼을 이용하면 됩니다.
- 리눅스 커널에서 윗화살표를 누르면 바로 전에 썼던 명령어가 나옵니다.

▼ 5) 파이썬 가상환경 설정하기

- EC2에 Python이 설치되었는지 확인 (고맙게도 이미 설치되어있습니다!)

```
python3 --version
```

- 해야하는 것: 우리가 pycharm에서 쉽게 했던 이 부분을, 쉘 명령어로 해줘야한다!



- 예전엔 가상환경을 위한 패키지(virtualenv)를 따로 받아야 했지만, python3부터는 자체적으로 제공합니다. (참고 링크)

```
설치를 위한 사전 업데이트 (apt-get : 리눅스에서 미리 모아둔 설치 파일들을 받는 편리한 방법)
sudo apt-get update

설치하기
sudo apt-get install python3-venv

가상환경 구동폴더 만들기
python3 -m venv ./myenv

가상환경 구동하기
[중요1] 명령어 실행 위치 내 myenv 폴더 -> bin  폴더 -> activate 파일을 실행하는 것
[중요2] 명령어를 실행 할 때, 위치를 잘 고려하자!!
[중요3] . myenv/bin/activate 임! . <- 점을 꼭 어야한다!
. myenv/bin/activate

가상환경 끄기
deactivate

잘 구동되면 앞에 (myenv)가 붙은 것을 확인할 수 있음
```

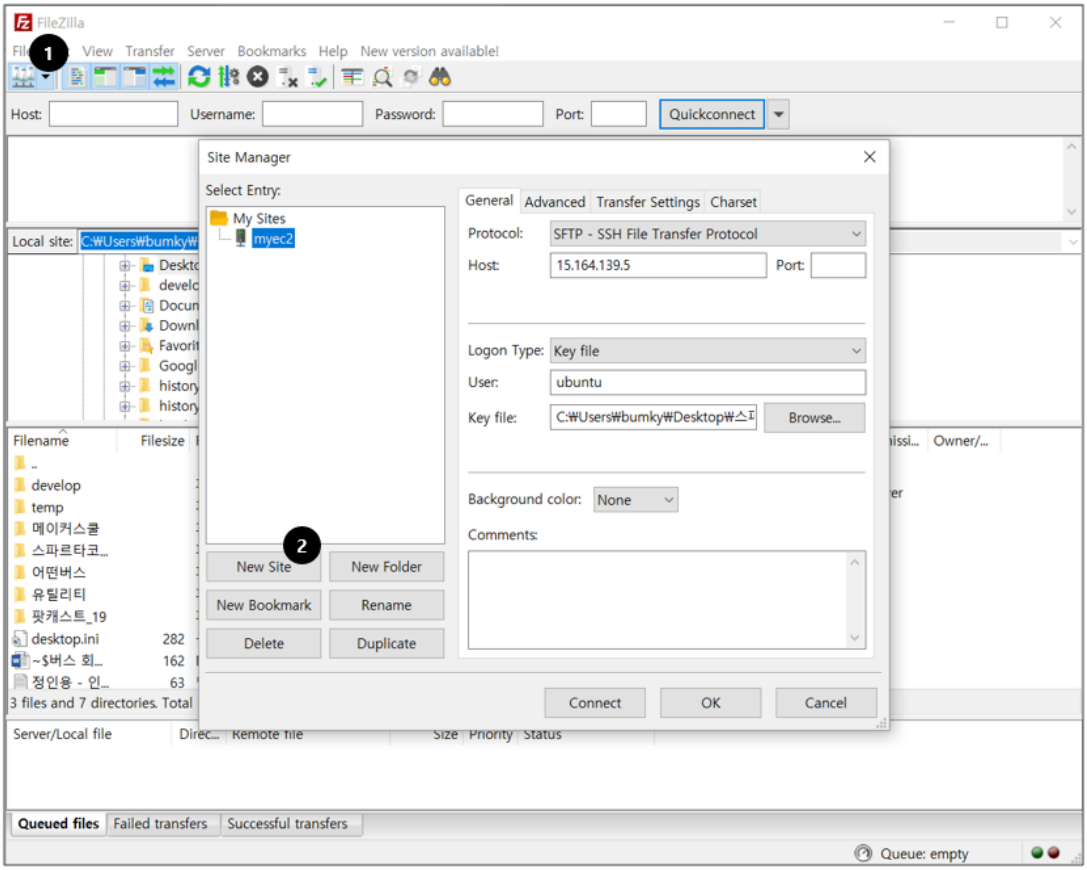
▼ 6) filezilla를 이용해서, 간단한 python 파일을 올려봅니다.

- 서버에 업로드 할 간단한 파일을 만듭니다.

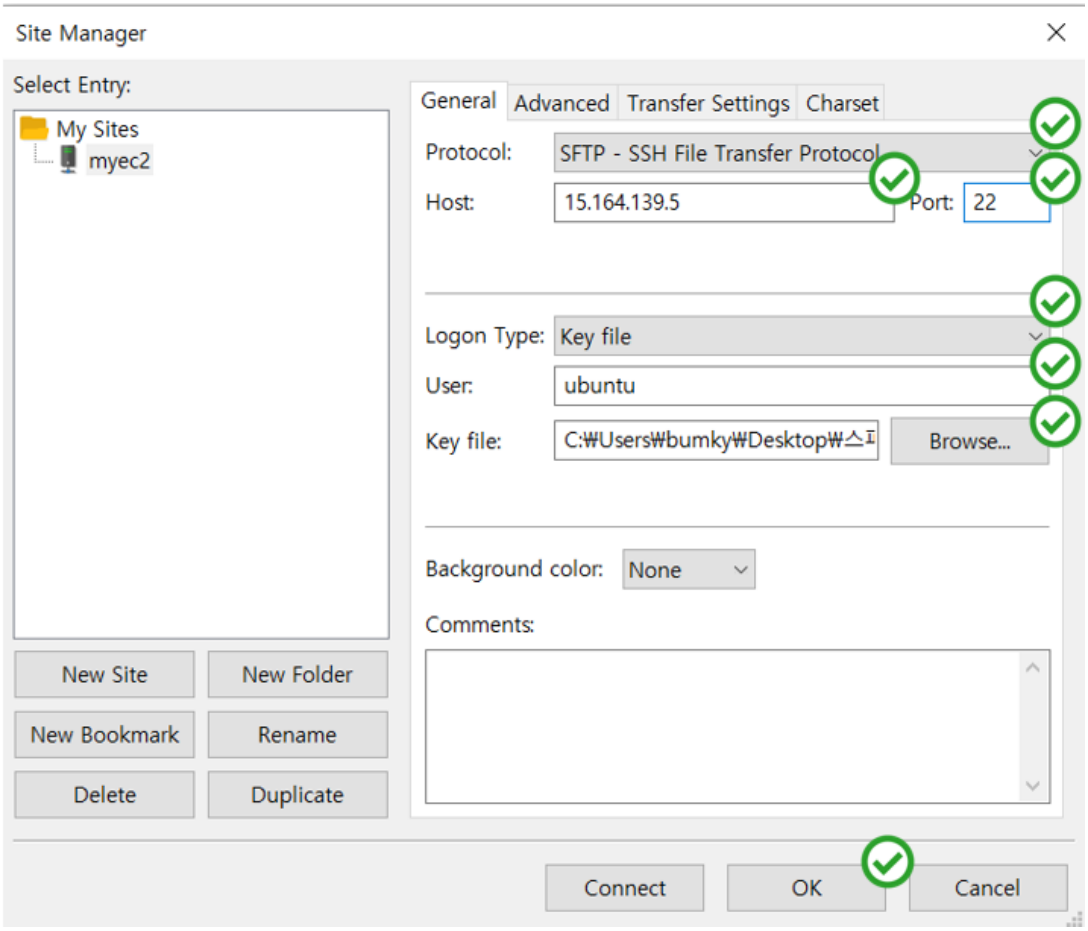
  예) test.py

```
# 아주 간단하게, 이 정도만 적어볼까요?
# 그리고 적당한 곳에 파일을 저장해봅니다.
print ('hello word!')
```

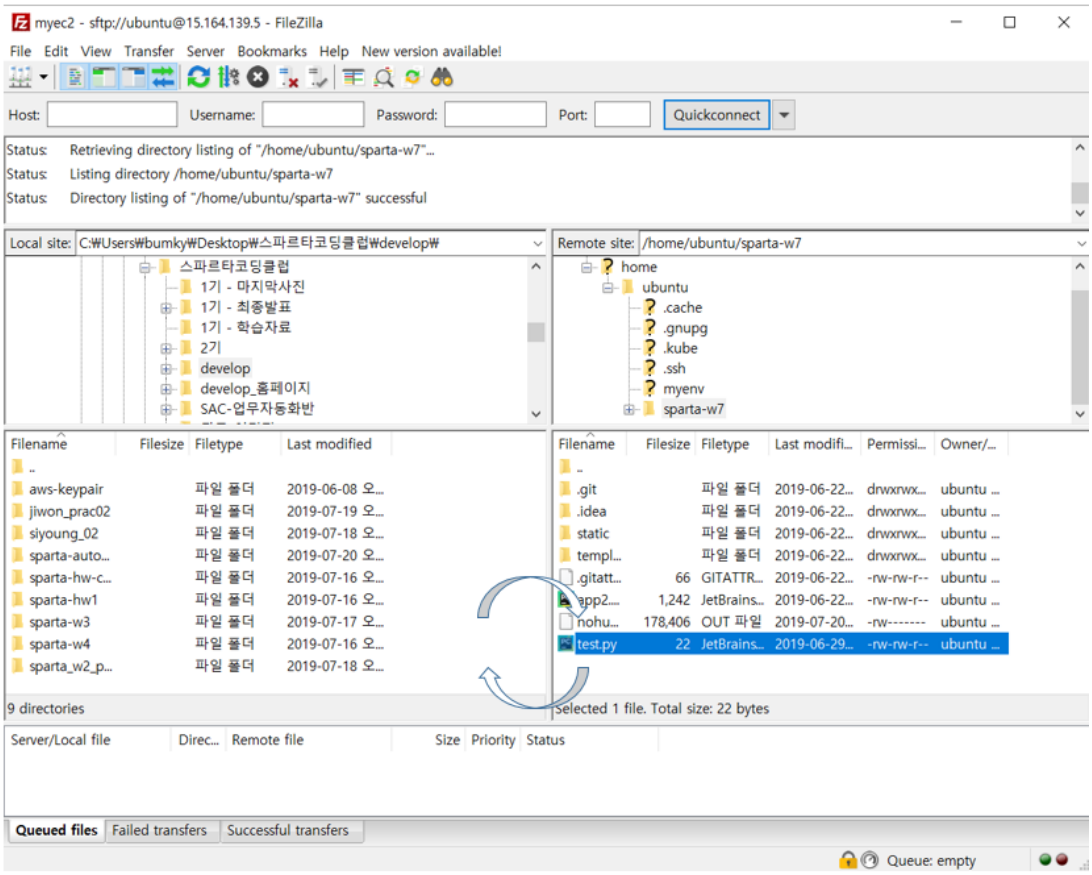
- 파일질라 실행, 다음과 같이 설정



- 정보들을 입력하고, ok 누르면 서버의 파일들을 볼 수 있음
(Host: 내 EC2서버의 ip // User: ubuntu 로 입력)



- 마우스로 드래그 해서 파일을 업로드/다운로드하면 됩니다!
  (자, 그럼 이제 간단한 파이썬 파일을 하나 만들어서 업로드 해볼까요?)



▼ 7) 파이썬 파일 실행해보기

- EC2 콘솔창에서 아래와 같이 입력합니다.

```
// home 디렉토리로 이동
cd ~

// 가상환경을 켠다 (myenv -> 본인이 설정한 가상환경의 폴더명을 입력)
. myenv/bin/activate

// 실행. 콘솔창에 hellow world!가 뜨는 것을 확인 할 수 있습니다.
python test.py
```

▼ 8) flask 서버를 실행해보기

- 기초적인 flask 서버 파일을 하나 만들어봅니다.

  예) app.py 파일

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
```

```
    return 'This is Home!'

if __name__ == '__main__':
    app.run('0.0.0.0',port=5000,debug=True)
```

- filezilla를 통해 EC2에 업로드한 다음, 실행해봅니다.

```
// 실행
python app.py
```

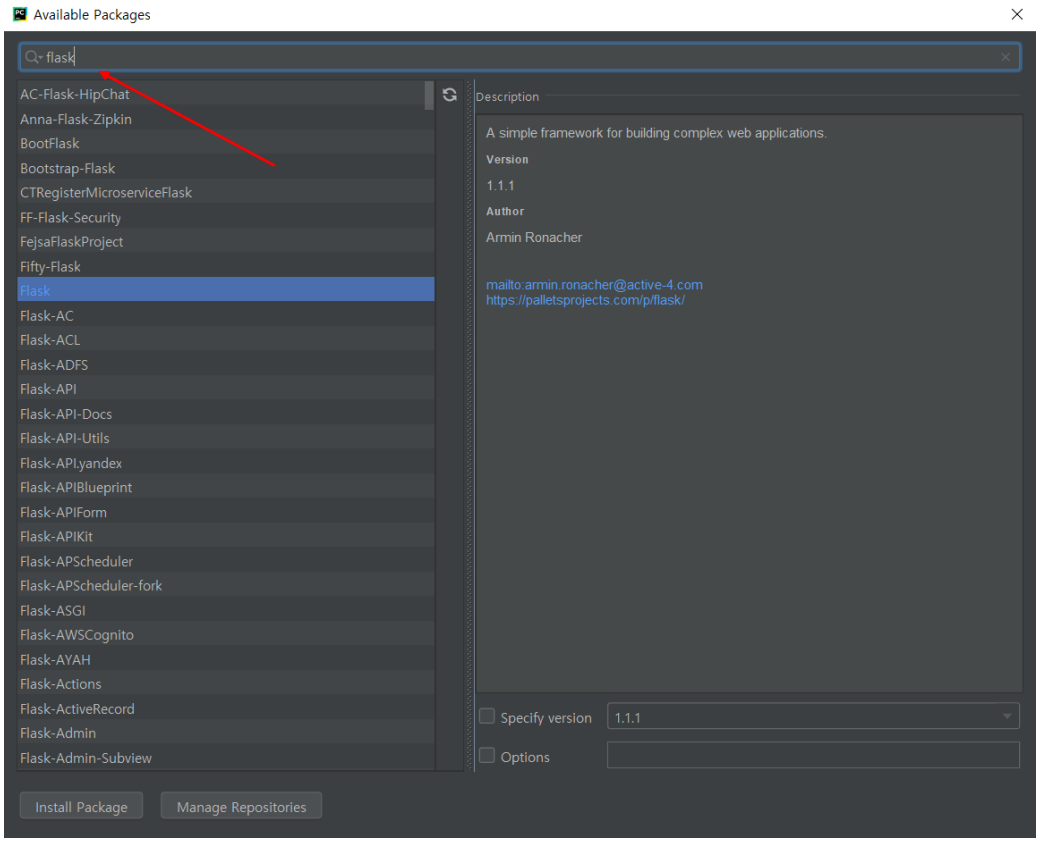
👉 에러가 납니다! 뭐라고 에러가 나는가요?

👉 정답: flask 패키지가 없는데? - 라는 에러입니다.
그럼, 패키지를 설치해볼까요?

▼ 9) 리눅스에서 패키지 설치하기

- 파이참으로 클릭→클릭해서 쉽게 설치했던 패키지를, 리눅스에선 콘솔 명령어를 통해 설치해줘야 합니다.

  ▼ 파이참에서 했던, 바로 이 부분!



- 다행히 파이썬은 친절하게도, 패키지를 쉽게 다운받고/업데이트하고/삭제할 수 있게 하는 패키지가 있습니다.(즉, 패키지 설치를 돕는 패키지!)
- 바로, "pip"라는 패키지입니다. (설명 링크)
- 가상환경을 켠 상태(=지금 그 상태에서!)에서 실행해보세요.
  고맙게도 이미 설치되어 있을 것입니다!

```
pip --version
```

- 설치한 pip를 이용해 해당 가상환경에 flask 설치

```
pip install flask
```

▼ 10) 다시 flask 서버를 실행해보기

- 아래 명령어로 flask 서버를 실행합니다.

```
python app.py
```

- 서버 실행이 되면, 크롬에서 접속을 해봅니다.

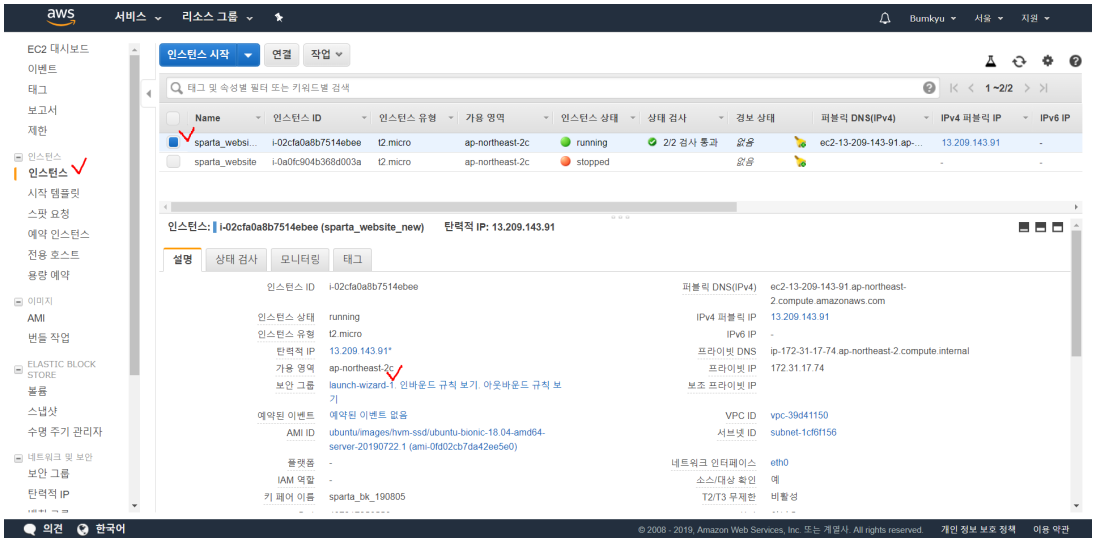
```
크롬 브라우저 창에 아래와 같이 입력합니다.

http://[내 EC2 IP]:5000/
```

> 👉 아직, 작동하지 않을 걸요! → AWS에서 약간의 설정이 더 필요합니다.

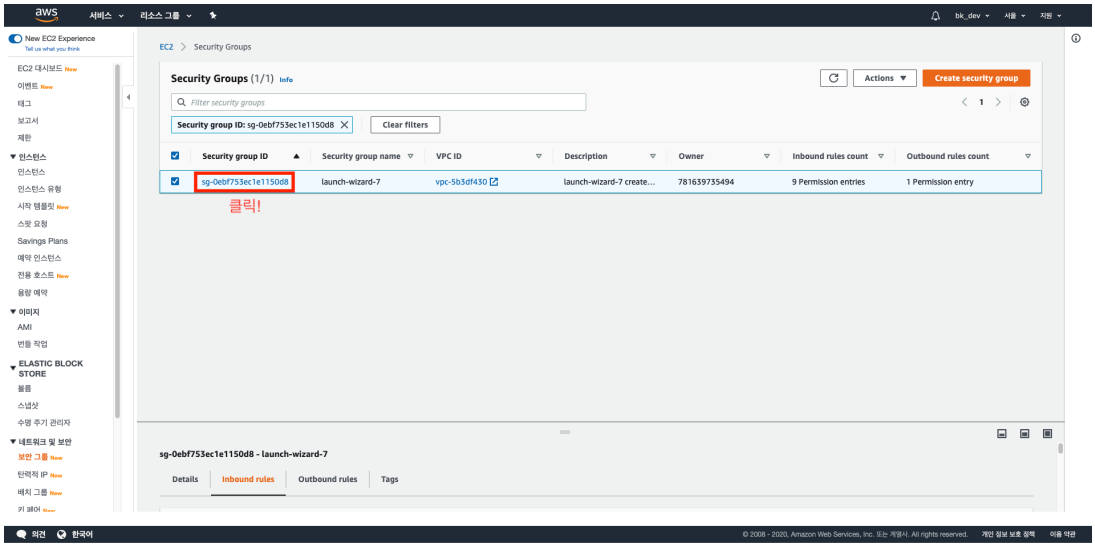
▼ 11) AWS에서 5000포트를 열어주기

- EC2 서버(=가상의 내 컴퓨터)에서 포트를 따로 설정하는 것 외에도,
  AWS EC2에서도 자체적으로 포트를 열고/닫을 수 있게 관리를 하고 있습니다.

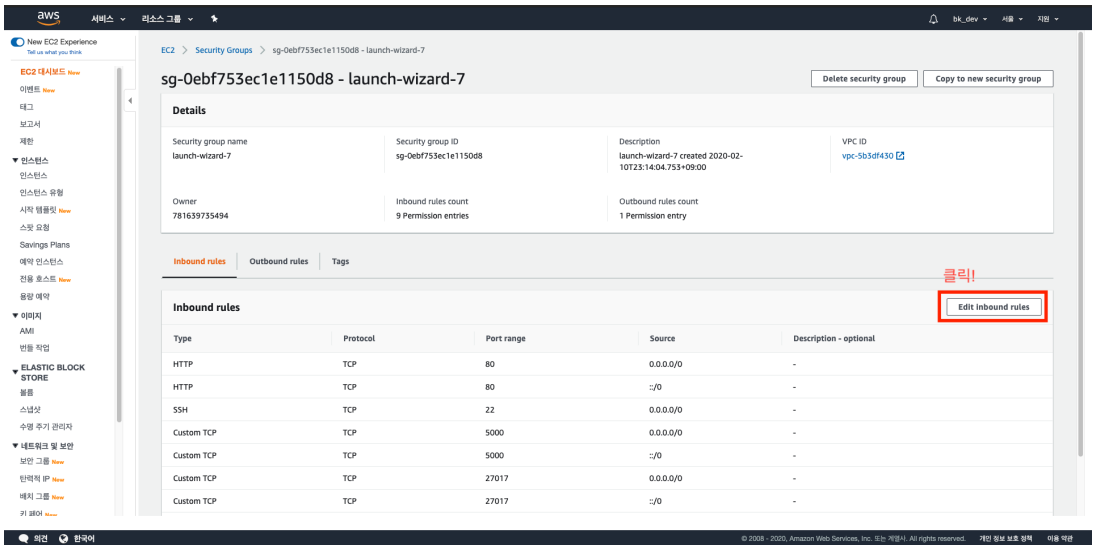
  → 그래서 AWS EC2 Security Group에서 인바운드 요청 포트를 열어줘야 합니다.
- 일단, EC2 관리 콘솔로 들어갑니다. 그리고 보안그룹(영문: Security Group)을 눌러 들어갑니다. 여기선 launch-wizard-1 이라고 쓰여 있네요



- 해당 보안그룹을 클릭합니다.



- Edit inbound rules를 선택합니다.

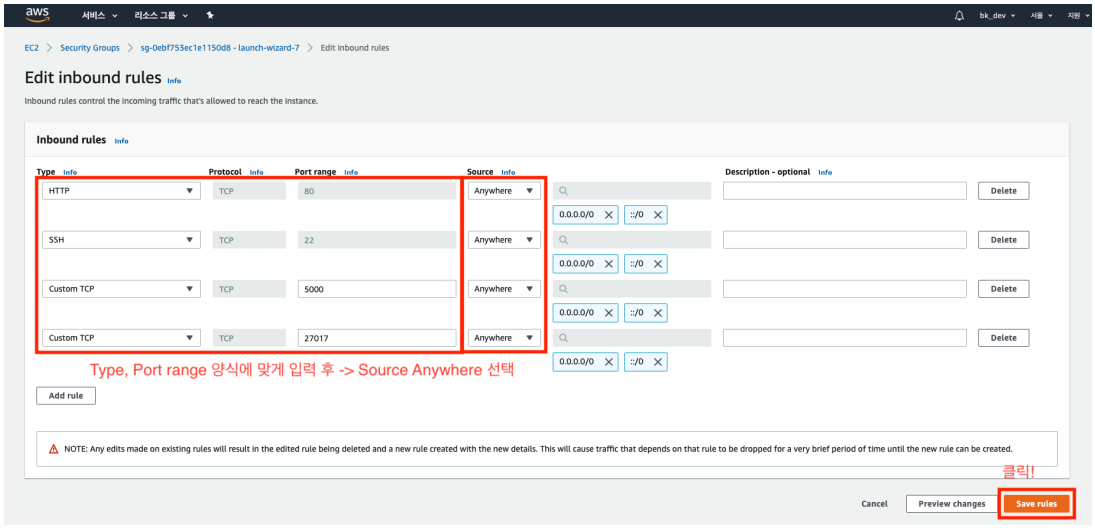


- 세 가지 포트를 추가해봅니다.

  → 80포트: HTTP 접속을 위한 기본포트

  → 5000포트: flask 기본포트

  → 27017포트: 외부에서 mongoDB 접속을 하기위한 포트



▼ 12) 다시 접속해봅니다!

- http://내아이피:5000

  → 잘 작동하는 것을 확인할 수 있습니다.

- 이제 app.py 파일을 수정하고, 다듬어서 올리면 진짜 프로젝트가 되겠죠?

[1시간]: 프로젝트 개요 & 목표 공유

▼ 각자의 프로젝트 진행 상황 / 목표를 공유합니다.

> 👉 **"개발일지" 작성해오셨나요?**
> 채팅방에 각자의 개발일지를 공유하고, 돌아가며 발표합니다.

- 발표에 꼭! 들어가야 하는 내용들
  1. 한 주 동안의 회고
  2. 한 주 동안의 배운 것들
  3. 이번주의 목표

▼ github repository의 주소를 톡방에 공유합니다.

- 튜터는 진행상황을 체크하고, 코드리뷰를 준비합니다

휴식시간 (30분)

후반 3시간

[3시간]: 개별 코딩 & 코드리뷰

- 집중 코딩을 합니다.
- 궁금한 것은 튜터에게 물어봅니다.
- 30분~1시간에 한번씩 튜터가 돌며 코드 리뷰를 합니다.

[숙제] - 다음 수업까지 개발일지 써오기

- Medium에 아래 내용이 포함된 개발일지를 작성해옵니다.

> 👉 개발일지는, 본인을 위해 작성하는 것입니다! 꼭 일주일에 한 편씩만 작성하지 않아도 되며, 사실은 개발을 할 때마다 매일 작성하는 것이 가장 좋은 방법입니다.
>
> TIL (today I learned)를 검색하면, 많은 분들의 개발일지를 볼 수 있습니다.

1. 한 주 동안의 회고
2. 한 주 동안의 배운 것들
3. 이번주의 목표

[설치] - 다음 시간을 위해 미리 설치해와야 할 것들

(없음!)