

Special Topic

Docker & AWS EC2

2021 Winter ABC Program

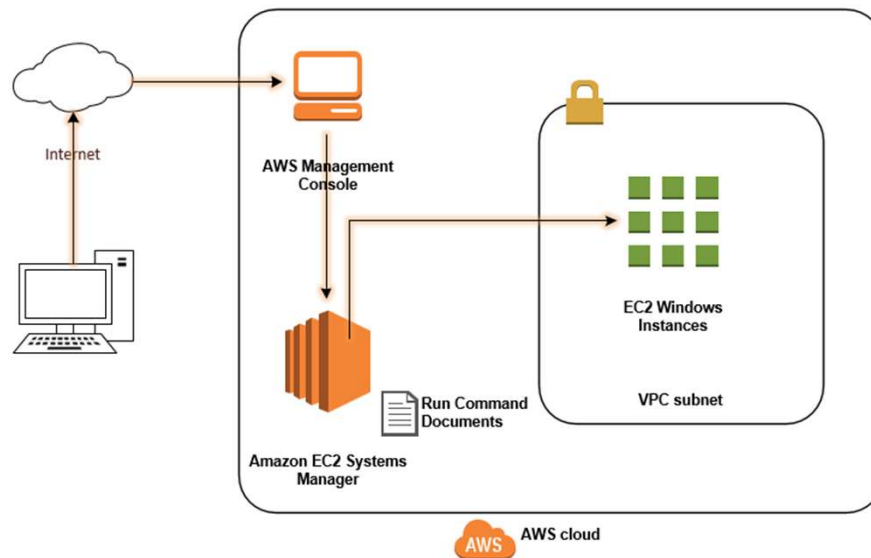
Mentor 정현준

2021 / 02 / 18



AWS EC2

- IaaS의 대표적인 예시입니다.
- 사용자는 AWS로부터 “인스턴스”라고 불리는 가상의 서버를 할당 받아서 그 위에 사용자가 원하는 응용 프로그램을 실행할 수 있도록 합니다.
- 실행 중인 인스턴스의 사용량이나 실행 시간에 대해 비용을 지불해야 합니다.



AWS 가입

- <https://signin.aws.amazon.com>
- 위 링크에서 콘솔에 로그인 -> AWS 계정 새로 만들기 버튼을 통해 계정을 만들 수 있습니다.
- AWS를 이용하기 위해서는 해외 결제가 가능한 VISA나 Mastercard가 필요합니다.





The image shows the AWS account creation form. On the left, a grey box contains the text: "12개월 프리 티어 액세스 포함 AWS 계정" and "Amazon EC2, Amazon S3 및 Amazon DynamoDB 사용 포함. 제안 약관 전문은 aws.amazon.com/free 참조". The main form is titled "AWS 계정 생성" and includes the following fields: "이메일 주소", "암호", "암호 확인", "AWS 계정 이름", "전화번호" (with a placeholder "+1 222-333-4444"), "국가/리전" (a dropdown menu currently showing "미국"), "주소" (with a placeholder "읍/면/동 우편번호, 회사 이름, 수신"), "구/군/시", "시/도 또는 리전", and "우편 번호".

AWS 가입

- 그리고 반드시 마지막에 “기본 플랜”으로 설정을 해주셔야 불필요한 결제를 막을 수 있습니다.
- 이 후 등록한 신용카드 계좌가 유효한 계좌인지 확인을 하게 되면 가입이 완료 됩니다.

AWS는 귀하의 요구를 충족할 수 있는 선별된 지원 플랜을 제공합니다. 귀하의 AWS 사용량에 맞는 지원 플랜을 선택하십시오. [자세히 알아보기](#)

		
기본 플랜	개발자 플랜	비즈니스 플랜
무료	월 \$29부터	월 \$100부터

AWS EC2 콘솔 대시보드

- AWS에서 제공하는 EC2를 웹 페이지를 통해 관리할 수 있는 서비스 입니다.
- 루트 사용자 로그인 -> 전체 서비스에서 EC2 선택

로그인

☒ 루트 사용자
무제한 액세스 권한이 필요한 작업을 수행하는 계
정 소유자입니다. 자세히 알아보기

☐ IAM 사용자
일일 작업을 수행하는 계정 내 사용자입니다. 자세
히 알아보기

루트 사용자 이메일 주소

다음

전체 서비스

컴퓨팅
EC2
Lightsail
Lambda
Batch

The screenshot displays the AWS Management Console interface. On the left, a navigation menu lists various services including IAM, EC2, and Lambda. The main panel is titled '리소스' (Resources) and shows a table of EC2 instances with columns for Name, State, and Availability Zone. The right sidebar contains sections for '계정 속성' (Account Attributes) and '추가 정보' (Additional Information), providing details about the account and the current instance.

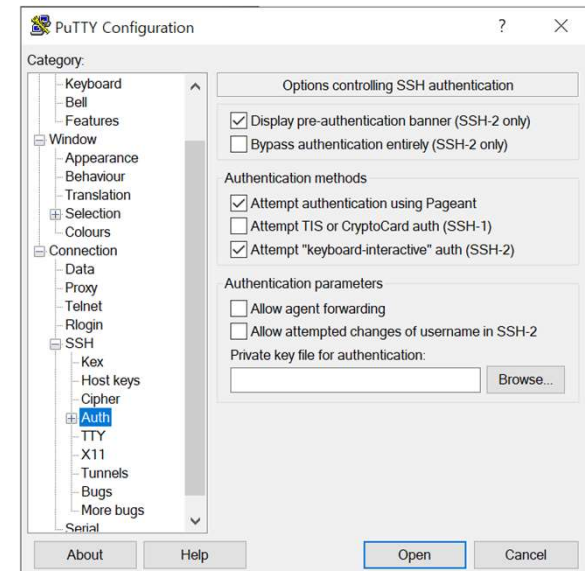
키 페어 생성하기

- 본격적으로 EC2 인스턴스를 생성하기 전에, 키 페어가 필요합니다.
- 키 페어는 생성한 EC2 인스턴스 서버에 관리자로 접속할 수 있는 권한을 확인하는 용도로 사용 됩니다.
- SSH의 private key – public key와 같습니다.
- AWS 대시보드 -> 리소스에서 키 페어 -> 키 페어 생성
- 접속하면 오른쪽 그림처럼 나옵니다.






키 페어 생성하기

- 키 페어 이름을 입력한 뒤, pem 형식과 ppk 형식 중에 선택해야 합니다.
- Pem 형식의 파일은 일반적인 SSH를 사용할 때, 다음과 같이 사용합니다.
 - `ssh -i [pem 파일 이름] [ID]@[인스턴스 ip 주소]`
- Ppk 형식의 파일은 putty를 통해 접속을 할 때 사용 됩니다.
 - Connection -> SSH -> Auth -> Browse
 - 이후 ppk 파일 선택
- 키 페어 생성을 누르면 자동으로 다운로드 됩니다.



AWS EC2 인스턴스 생성하기

- EC2 대시보드 -> 인스턴스 시작 버튼 클릭
- 또는 왼쪽 메뉴 -> 인스턴스 -> 인스턴스 시작 버튼 클릭

<div></div> <div>Amazon Linux</div> <div>프리 티어 사용 가능</div>	<div>Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-09282971cf2faa4c9 (64비트 x86) / ami-094d838e7f8147d58 (64비트 Arm)</div> <div>Amazon Linux 2는 5년간 지원을 제공합니다. Amazon EC2에 성능 최적화된 Linux kernel 4.14와 systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, 최신 소프트웨어 패키지를 추가적으로 제공합니다.</div> <div>루트 디바이스 유형: ebs 가상화 유형: hvm ENA 활성화: 예</div>	<div>선택</div> <div><input checked="" type="radio"/> 64비트(x86)</div> <div><input type="radio"/> 64비트(Arm)</div>
<div></div> <div>Ubuntu Server</div> <div>프리 티어 사용 가능</div>	<div>Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-067abcae434ee508b (64비트 x86) / ami-0e59aef3dbf4e4 (64비트 Arm)</div> <div>Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).</div> <div>루트 디바이스 유형: ebs 가상화 유형: hvm ENA 활성화: 예</div>	<div>선택</div> <div><input checked="" type="radio"/> 64비트(x86)</div> <div><input type="radio"/> 64비트(Arm)</div>
<div></div> <div>Ubuntu Server</div> <div>프리 티어 사용 가능</div>	<div>Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b50511490117e709 (64비트 x86) / ami-0ee0bf6766e2dbdca (64비트 Arm)</div> <div>Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).</div> <div>루트 디바이스 유형: ebs 가상화 유형: hvm ENA 활성화: 예</div>	<div>선택</div> <div><input checked="" type="radio"/> 64비트(x86)</div> <div><input type="radio"/> 64비트(Arm)</div>
<div></div> <div>SUSE Linux</div> <div>프리 티어 사용 가능</div>	<div>SUSE Linux Enterprise Server 15 SP2 (HVM), SSD Volume Type - ami-097fc5cd098dd20d5 (64비트 x86) / ami-06fb0a38fed3dcdec (64비트 Arm)</div> <div>SUSE Linux Enterprise Server 15 Service Pack 2 (HVM), EBS General Purpose (SSD) Volume Type. Amazon EC2 AMI Tools preinstalled; Apache 2.2, MySQL 5.5, PHP 5.3, and Ruby 1.8.7 available.</div> <div>루트 디바이스 유형: ebs 가상화 유형: hvm ENA 활성화: 예</div>	<div>선택</div> <div><input checked="" type="radio"/> 64비트(x86)</div> <div><input type="radio"/> 64비트(Arm)</div>
<div></div> <div>Red Hat</div> <div>프리 티어 사용 가능</div>	<div>Red Hat Enterprise Linux 8 (HVM), SSD Volume Type - ami-07270d166cdf39adc (64비트 x86) / ami-05de3e74a170005d0 (64비트 Arm)</div> <div>Red Hat Enterprise Linux version 8 (HVM), EBS General Purpose (SSD) Volume Type</div> <div>루트 디바이스 유형: ebs 가상화 유형: hvm ENA 활성화: 예</div>	<div>선택</div> <div><input checked="" type="radio"/> 64비트(x86)</div> <div><input type="radio"/> 64비트(Arm)</div>

반드시 "프리 티어 사용 가능"이 적힌 OS를 선택해야 합니다.

AWS EC2 인스턴스 생성하기

- 이 후 2단계 인스턴스 유형 선택에서 “프리 티어 사용 가능”이 적힌 유형을 선택한 후, “보안 그룹 구성” 탭까지 다음 버튼을 눌러 줍니다.

단계 6: 보안 그룹 구성

보안 그룹은 인스턴스에 대한 트래픽을 제어하는 방화벽 규칙 세트입니다. 이 페이지에서는 특정 트래픽을 인스턴스에 도달하도록 허용할 때에 대한 무제한 액세스를 허용하는 규칙을 추가합니다. 새 보안 그룹을 생성하거나 아래에 나와 있는 기존 보안 그룹 중에서 선택할 수 있습니다.

보안 그룹 할당: ☒ 새 보안 그룹 생성
☐ 기존 보안 그룹 선택

보안 그룹 이름:
설명:

- 보안 그룹 구성 탭에서 보안 그룹 이름과 설명을 바꾼 뒤, 검토 및 시작하기 버튼을 누르면 마지막 검토 페이지로 이동합니다.
- 검토 페이지에서 오른쪽 하단에 시작하기 버튼을 눌러 줍니다.

AWS EC2 인스턴스 생성하기

- 키 페어 선택 창이 나오면 이전 슬라이드에서 생성한 키 페어를 선택합니다.
- 이후 체크박스를 체크 해준 뒤, 인스턴스 시작 버튼을 눌러 줍니다.

기존 키 페어 선택 또는 새 키 페어 생성

×

키 페어는 AWS에 저장하는 **퍼블릭 키**와 사용자가 저장하는 **프라이빗 키 파일**로 구성됩니다. 이 둘을 모두 사용하여 SSH를 통해 인스턴스에 안전하게 접속할 수 있습니다. Windows AMI의 경우 인스턴스에 로그인하는 데 사용되는 암호를 얻으려면 프라이빗 키 파일이 필요합니다. Linux AMI의 경우, 프라이빗 키 파일을 사용하면 인스턴스에 안전하게 SSH로 연결할 수 있습니다.

참고: 선택한 키 페어가 이 인스턴스에 대해 승인된 키 세트에 추가됩니다. 퍼블릭 AMI에서 기존 키 페어 제거에 대해 자세히 알아보십시오.

기존 키 페어 선택

▼

키 페어를 선택하십시오

abc_mentoring

▼

☒ 선택한 프라이빗 키 파일(abc_mentoring.pem)에 액세스할 수 있음을 확인합니다. 이 파일이 없으면 내 인스턴스에 로그인할 수 없습니다.

취소

인스턴스 시작

AWS EC2 인스턴스 생성하기

- 인스턴스가 생성이 되면 EC2 대시보드에 인스턴스 메뉴에서 생성된 인스턴스를 볼 수 있습니다.

The screenshot displays the AWS Management Console interface for EC2 instances. At the top, there's a search bar and navigation buttons like '연결', '인스턴스 상태', '작업', and '인스턴스 시작'. Below this is a table listing instances. The table has columns for Name, 인스턴스 ID, 인스턴스 상태, 인스턴스 유형, 상태 검사, 정보 상태, 가용 영역, 퍼블릭 IPv4 DNS, and 퍼블릭 IPv4 One instance is listed with ID i-0ebec45d88b7f71df, status '실행 중' (Running), type 't2.micro', and availability zone 'ap-northeast-2c'. Below the table, the details for the selected instance '인스턴스: i-0ebec45d88b7f71df' are shown. The details are organized into sections: '인스턴스 요약 정보' (Instance Summary Information) and '인스턴스 상세 정보' (Instance Detailed Information). The summary section includes fields for 인스턴스 ID (i-0ebec45d88b7f71df), 인스턴스 상태 (실행 중), 인스턴스 유형 (t2.micro), and a link to 'AWS Compute Optimizer 찾기'. The detailed information section is divided into three columns: 퍼블릭 IPv4 주소 (52.79.234.227), 퍼블릭 IPv4 DNS (ec2-52-79-234-227.ap-northeast-2.compute.amazonaws.com), and 프라이빗 IPv4 주소 (172.31.40.165). Other details include 프라이빗 IPv4 DNS (ip-172-31-40-165.ap-northeast-2.compute.internal), VPC ID (vpc-0c1aac67), and 서브넷 ID (subnet-ec190ea0).

Name	인스턴스 ID	인스턴스 상태	인스턴스 유형	상태 검사	정보 상태	가용 영역	퍼블릭 IPv4 DNS	퍼블릭 IPv4 ...
-	i-0ebec45d88b7f71df	실행 중	t2.micro	초기화	경보 없음	ap-northeast-2c	ec2-52-79-234-227.ap...	52.79.234.227

인스턴스: i-0ebec45d88b7f71df

세부 정보 | 보안 | 네트워킹 | 스토리지 | 상태 검사 | 모니터링 | 태그

▼ 인스턴스 요약 정보

인스턴스 ID i-0ebec45d88b7f71df	퍼블릭 IPv4 주소 52.79.234.227 개방 주소법	프라이빗 IPv4 주소 172.31.40.165
인스턴스 상태 실행 중	퍼블릭 IPv4 DNS ec2-52-79-234-227.ap-northeast-2.compute.amazonaws.com 개방 주소법	프라이빗 IPv4 DNS ip-172-31-40-165.ap-northeast-2.compute.internal
인스턴스 유형 t2.micro	탄력적 IP 주소 -	VPC ID vpc-0c1aac67
AWS Compute Optimizer 찾기 권장 사항을 위해 AWS Compute Optimizer에 옵트인합니다. 자세히 알아보기	IAM 역할 -	서브넷 ID subnet-ec190ea0

AWS EC2 인스턴스 접속하기

- 인스턴스에 접속하기 위해서는 아래 3가지를 알아야 합니다.
 - 인스턴스에 접속할 IP
 - 인스턴스 계정 이름
 - pem 또는 ppk 파일
- 여기서 pem 파일이나 ppk 파일은 이전에 키 페어를 생성하면서 다운로드 받은 파일을 의미합니다.
- 또한 ubuntu 인스턴스를 만들었다면 처음 인스턴스 접속을 위한 계정 이름은 ubuntu라는 이름으로 초기화 되어 있습니다.







AWS EC2 인스턴스 접속하기

- 인스턴스 접속에 필요한 인스턴스의 IP는 EC2 대시보드에 인스턴스 탭에서 확인할 수 있습니다.

인스턴스: i-0ebec45d88bff71df

세부 정보 | 보안 | 네트워킹 | 스토리지 | 상태 검사 | 모니터링 | 태그

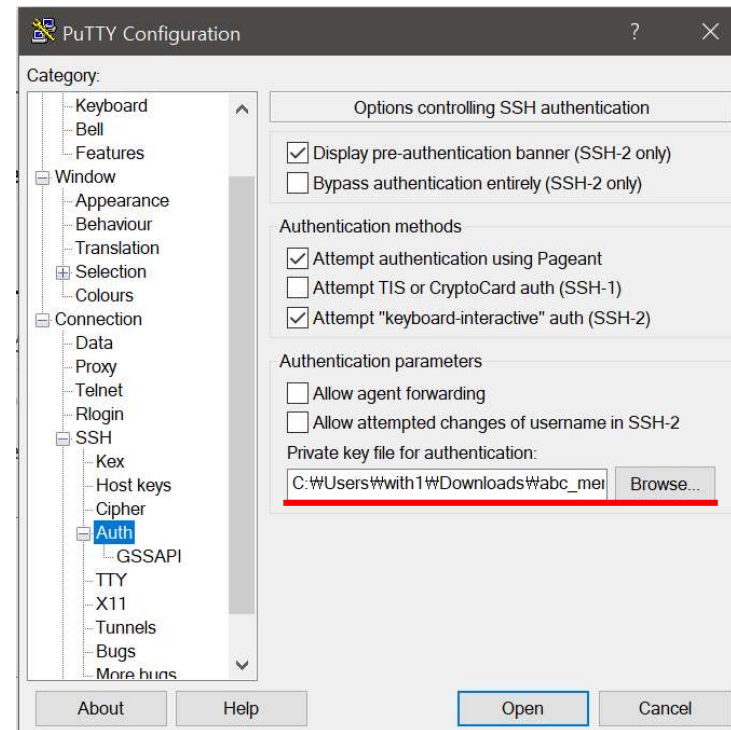
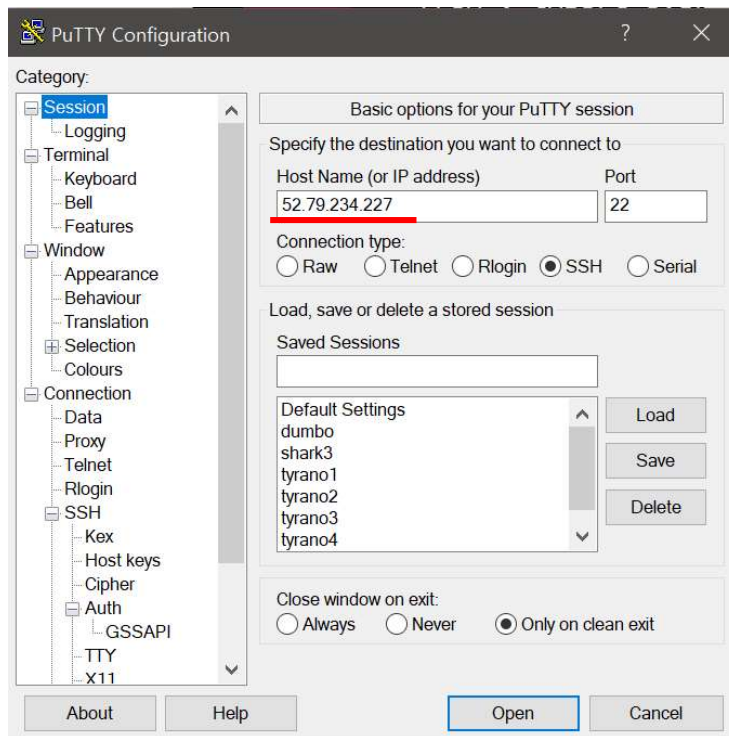
▼ 인스턴스 요약 정보

인스턴스 ID	퍼블릭 IPv4 주소
 i-0ebec45d88bff71df	 <u>52.79.234.227</u> 개방 주소법 
인스턴스 상태	퍼블릭 IPv4 DNS
 실행 중	 <u>ec2-52-79-234-227.ap-northeast-2.compute.amazonaws.com</u> 개방 주소법 
인스턴스 유형	탄력적 IP 주소
t2.micro	-

- 퍼블릭 IPv4 주소에 IP 주소나 DNS 주소를 통해 접속할 수 있습니다.

AWS EC2 인스턴스 접속하기

- Putty의 경우, Host name에 퍼블릭 DNS나 IP 주소를 입력하고 Connection -> SSH -> Auth 탭에서 ppk 파일을 Browse 합니다.



AWS EC2 인스턴스 접속하기

- 이 후, open 버튼을 누른 후, ubuntu 이름으로 로그인하면 EC2 인스턴스 서버를 이용할 수 있습니다.

```
ubuntu@ip-172-31-40-165: ~  
login as: ubuntu  
Authenticating with public key "abc_mentoring"  
Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 5.4.0-1037-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/advantage  
  
System information as of Wed Feb 17 09:44:10 UTC 2021  
  
System load:  0.0               Processes:           92  
Usage of /:   14.8% of 7.69GB   Users logged in:    0  
Memory usage: 18%              IP address for eth0: 172.31.40.165  
Swap usage:   0%  
  
0 packages can be updated.  
0 of these updates are security updates.  
  
New release '20.04.2 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Wed Feb 17 09:32:42 2021 from 114.70.9.219  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-40-165:~$
```

AWS EC2 설정

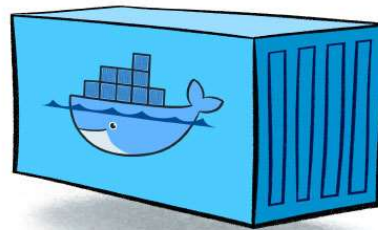
- EC2 접속이 완료되면 다음 명령어를 통해 EC2 환경을 업데이트 해줍니다.
- `sudo apt-get update`
- `sudo apt-get upgrade`
- 이 후 필요에 따라 vim이나 bash 또는 기타 환경 설정을 세팅해주면 됩니다.

Docker

- Container를 이용해 각종 응용 프로그램을 빠르게 구축, 테스트, 배포할 수 있도록 도와주는 SW 플랫폼 입니다.
- Linux 환경에서 Container를 생성하고 사용할 수 있도록 도와줍니다.



응용 프로그램



도커 컨테이너



서버

Docker 설치하기

- 우분투 환경에서 다음 명령어를 사용하면 docker가 설치 됩니다.
- `sudo snap install docker`
- `sudo apt-get install docker.io`
- 이 후 터미널에 docker를 입력했을 때, 다음과 같은 화면이 나타나면 docker 설치가 성공적으로 완료 된 것입니다.

```
ubuntu@ip-172-31-40-165:~$ docker
Usage: docker [OPTIONS] COMMAND
A self-sufficient runtime for containers

Options:
  --config string      Location of client config files (default "/home/ubuntu/snap/docker/796/.docker")
  -c, --context string  Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var
                        and default context set with "docker context use")
  -D, --debug           Enable debug mode
  -H, --host list       Daemon socket(s) to connect to
  -l, --log-level string Set the logging level ("debug"|"info"|"warn"|"error"|"fatal") (default "info")
  --tls                Use TLS; implied by --tlsverify
  --tlscacert string    Trust certs signed only by this CA (default "/home/ubuntu/snap/docker/796/.docker/ca.pem")
  --tlscert string      Path to TLS certificate file (default "/home/ubuntu/snap/docker/796/.docker/cert.pem")
  --tlskey string       Path to TLS key file (default "/home/ubuntu/snap/docker/796/.docker/key.pem")
  --tlsverify           Use TLS and verify the remote
  -v, --version         Print version information and quit

Management Commands:
  builder      Manage builds
  config       Manage Docker configs
  container    Manage containers
  context      Manage contexts
  engine       Manage the docker engine
  image        Manage images
  network      Manage networks
  node         Manage Swarm nodes
  plugin       Manage plugins
  secret       Manage Docker secrets
  service      Manage services
  stack        Manage Docker stacks
  swarm       Manage Swarm
  system       Manage Docker
  trust        Manage trust on Docker images
  volume       Manage volumes

Commands:
  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
```

Docker image

- Docker image는 다른 사람들이 사전에 미리 만들어 놓은 환경 입니다.
- Github처럼 docker hub를 통해 다운 받을 수 있습니다.
- Docker는 이미지를 실행하는 것으로 컨테이너에 해당 이미지에 내포된 환경을 올려서 서비스를 구축/실행 합니다.
- Scratch image는 docker에서 아무것도 없는 비어 있는 베이스 이미지를 의미합니다.
- Scratch image의 경우 아무것도 없기 때문에 기본 이미지로 많이 사용합니다.

Docker 명령어

- Docker는 os 커널을 사용하기 때문에 반드시 sudo를 붙여서 사용해야 합니다.
 - docker ps
 - docker search [이미지 이름]
 - docker pull [이미지 이름]
 - docker images
 - docker run [이미지 이름]
 - docker build [Dockerfile 경로]
 - docker rm [컨테이너 이름]
 - docker rmi [이미지 이름]
 - docker attach [컨테이너 이름]

Docker file

- Docker를 사용하기 위해서는 직접 docker 명령어를 입력하는 방법도 있지만 docker file을 이용한 방법도 있습니다.
- Make의 Makefile처럼 docker는 Dockerfile이라는 이름을 가진 파일을 통해 사전에 준비한 docker 명령어를 순차적으로 실행하고 필요한 환경을 자동으로 설정할 수 있습니다.
- Dockerfile은 항상 이름이 Dockerfile이며, 확장자는 없습니다.
- 완성된 Dockerfile은 아래와 같은 명령어로 실행이 가능합니다.
 - `docker build [option] [Dockerfile 경로]`

Docker file 작성 방법

- Dockerfile은 아래와 같은 여러 명령어들을 이용해 작성합니다.
 - FROM : 컨테이너의 기반이 되는 이미지
 - MAINTAINER : 작성자 정보
 - RUN : shell script나 명령어를 실행
 - CMD : 컨테이너가 실행되었을 때 사용할 명령어
 - LABEL : 라벨 작성 (docker inspect 명령어로 확인 가능)
 - ENV : 컨테이너 내부에서 환경변수 설정
 - ADD : 파일 / 디렉토리 추가
 - COPY : 파일 복사
 - ENTRYPOINT : 컨테이너가 시작되었을 때 딱 한 번 실행할 스크립트 실행
 - VOLUME : docker volume 마운트
 - WORKDIR : RUN, CMD, ENTRYPOINT가 실행 될 작업 디렉토리

Docker file 예시

```
Dockerfile
1 FROM ubuntu:16.04
2
3 MAINTAINER Hyunjoon_Jeong "with1015@unist.ac.kr"
4
5 RUN apt-get update -y
6 RUN apt-get install -y python-pip python-dev build-essential
7
8 COPY . /app
9
10 WORKDIR /app
11
12 RUN pip install -r requirements.txt
13
14 ENTRYPOINT ["python"]
15 CMD ["hello_world.py"]
```

- 파일 저장 후 **sudo docker build --tag abc_mentoring:abc .** 입력

Docker image 실행

- Dockerfile이 빌드 되면 docker images 명령어를 통해 이미지가 생성된 것을 확인할 수 있습니다.

```
with1015@ubuntu:~/Docker_study$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
abc_mentoring       abc                5d6cfd8021e6       43 minutes ago     429MB
ubuntu              16.04             8185511cd5ad       4 weeks ago        132MB
with1015@ubuntu:~/Docker_study$
```

- 생성된 이미지로 컨테이너를 만들고 실행하기 위해서는 다음 명령어를 사용해야 합니다.
 - sudo docker run REPOSITORY:TAG
 - 위 예시의 경우, sudo docker run abc_mentoring:abc

```
with1015@ubuntu:~/Docker_study$ sudo docker run abc_mentoring:abc
hello world!
with1015@ubuntu:~/Docker_study$
```


Docker container 관리

- `sudo docker ps -a`를 사용해 실행/종료/대기 중인 컨테이너를 볼 수 있습니다.
- `sudo docker rm [컨테이너 ID 또는 이름]` 으로 종료된 컨테이너를 삭제할 수 있습니다.

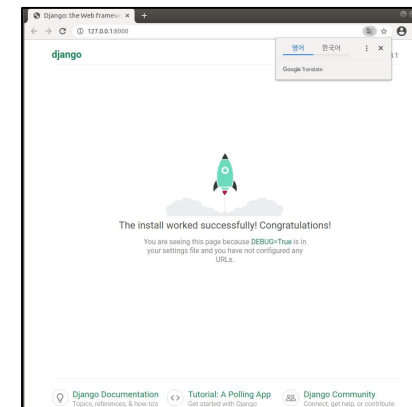
```
with1015@ubuntu:~/Docker_study$ sudo docker run abc_mentoring:abc
hello world!
with1015@ubuntu:~/Docker_study$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS          PORTS
NAMES
c891872f05f0   abc_mentoring:abc  "python hello_world...."  4 minutes ago  Exited (0) 4 minutes ago
vibrant_greider
340cd5bdc049   5d6cfd8021e6     "python hello_world...."  52 minutes ago  Exited (0) 52 minutes ago
focused_shockley
with1015@ubuntu:~/Docker_study$ sudo docker rm 340cd5bdc049
340cd5bdc049
with1015@ubuntu:~/Docker_study$ sudo docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS          PORTS
NAMES
c891872f05f0   abc_mentoring:abc  "python hello_world...."  5 minutes ago  Exited (0) 5 minutes ago
vibrant_greider
with1015@ubuntu:~/Docker_study$
```

Docker를 통한 실제 서비스 배포

- 아래 링크는 Django라는 웹 프레임워크를 이용한 간단한 웹 페이지 서비스 프로그램입니다.
- https://github.com/with1015/web_benchmark.git
- Python3와 pip가 있는 경우 다음 명령어를 통해 실행할 수 있습니다.
 - pip3 install Django
 - python3 manage.py runserver

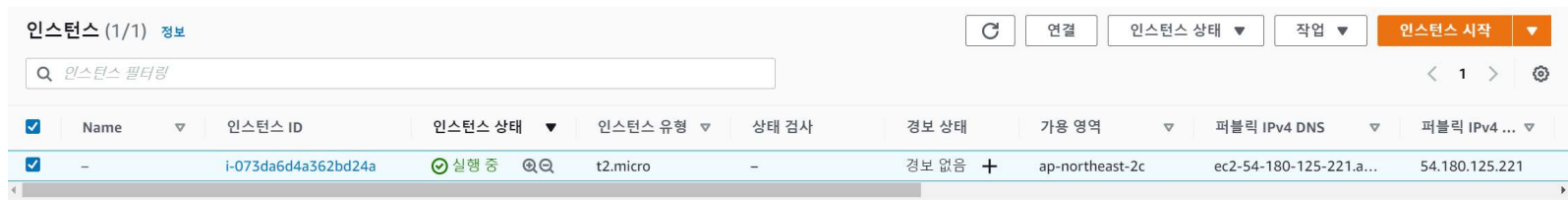
```
with1015@ubuntu:~/Docker_study/web_service/web_benchmark$ ls
db.sqlite3  manage.py  web_benchmark
with1015@ubuntu:~/Docker_study/web_service/web_benchmark$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
February 18, 2021 - 06:24:00
Django version 3.1.6, using settings 'web_benchmark.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```



Docker를 통한 실제 서비스 배포

- 하지만 이는 프로그램을 실행한 로컬 환경에서만 접속이 가능합니다.
- 실습을 통해 다른 사람들도 웹 페이지를 볼 수 있도록 docker와 AWS를 이용해 웹 서비스 배포를 해봅시다.
- 우선 배포하기에 앞서 AWS EC2에 인스턴스 하나를 만들어 준비하고, 인스턴스 내부에 docker를 설치 합니다.



The screenshot shows the AWS Management Console interface for an EC2 instance. At the top, there's a header with '인스턴스 (1/1) 정보' and several action buttons like '연결', '인스턴스 상태', '작업', and '인스턴스 시작'. Below this is a search bar and a table listing the instance details.

<input checked="" type="checkbox"/>	Name	인스턴스 ID	인스턴스 상태	인스턴스 유형	상태 검사	경보 상태	가용 영역	퍼블릭 IPv4 DNS	퍼블릭 IPv4 ...
<input checked="" type="checkbox"/>	-	i-073da6d4a362bd24a	실행 중	t2.micro	-	경보 없음	ap-northeast-2c	ec2-54-180-125-221.a...	54.180.125.221

Docker를 통한 실제 서비스 배포

- 그 다음 github에서 web benchmark 파일을 git clone을 통해 EC2 인스턴스에 복사합니다.

```
ubuntu@ip-172-31-36-224:~$ git clone https://github.com/with1015/web_benchmark.git
Cloning into 'web_benchmark'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 24 (delta 9), reused 20 (delta 5), pack-reused 0
Unpacking objects: 100% (24/24), done.
ubuntu@ip-172-31-36-224:~$ ls
web_benchmark
ubuntu@ip-172-31-36-224:~$
```

Docker를 통한 실제 서비스 배포

- 다음과 같이 EC2 인스턴스의 퍼블릭 IPv4 주소를 web_benchmark 폴더 내부에서 settings.py에 ALLOWED_HOSTS 리스트에 추가합니다.

인스턴스: i-073da6d4a362bd24a

세부 정보 | 보안 | 네트워킹 | 스토리지 | 상태 검사 | 모니터링 | 태그

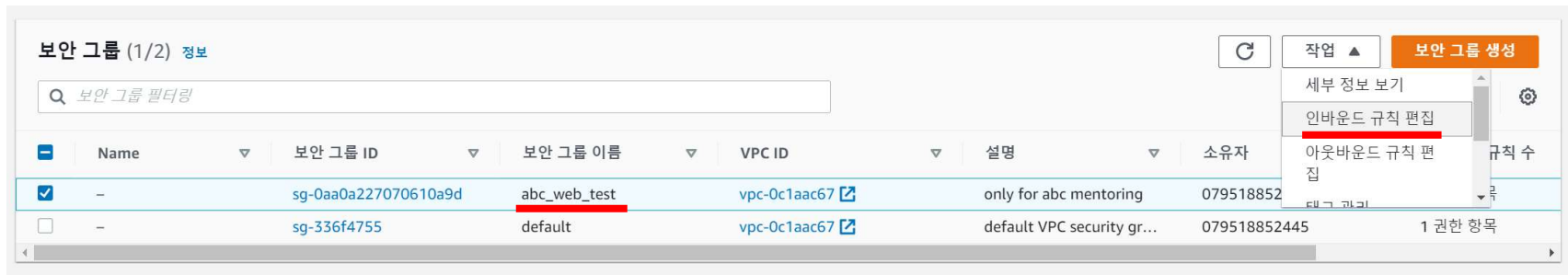
▼ 인스턴스 요약 정보

인스턴스 ID i-073da6d4a362bd24a	퍼블릭 IPv4 주소 54.180.125.221 개방 주소법
인스턴스 상태 실행 중	퍼블릭 IPv4 DNS ec2-54-180-125-221.ap-northeast-2.compute.amazonaws.com 개방 주소법

```
27
28 ALLOWED_HOSTS = ['127.0.0.1', '54.180.125.221']
29
30
```

Docker를 통한 실제 서비스 배포

- EC2 대시보드 왼쪽 메뉴 -> 네트워크 및 보안 -> 보안그룹
- 생성한 EC2 인스턴스 이름을 가진 그룹을 선택하고 작업 -> 인바운드 규칙 편집을 클릭해줍니다.



보안 그룹 (1/2) 정보

Q 보안 그룹 필터링

	Name	보안 그룹 ID	보안 그룹 이름	VPC ID	설명	소유자	
<input checked="" type="checkbox"/>	-	sg-0aa0a2270610a9d	abc_web_test	vpc-0c1aac67	only for abc mentoring	079518852	
<input type="checkbox"/>	-	sg-336f4755	default	vpc-0c1aac67	default VPC security gr...	079518852445	1 권한 항목

작업 ▲

- 세부 정보 보기
- 인바운드 규칙 편집
- 아웃바운드 규칙 편집
- 태그 관리

보안 그룹 생성

Docker를 통한 실제 서비스 배포

- 인바운드 규칙 편집에서 **규칙 추가** 버튼을 누른 뒤, **사용자 지정 TCP -> 포트 범위 8000 -> 소스는 위치 무관** 으로 설정해주고 규칙 저장 버튼을 누릅니다.

인바운드 규칙 편집 정보

인바운드 규칙은 인스턴스에 도달하도록 허용된 수신 트래픽을 제어합니다.

인바운드 규칙 정보

유형 <small>정보</small>	프로토콜 <small>정보</small>	포트 범위 <small>정보</small>	소스 <small>정보</small>
SSH ▼	TCP	22	사용자 지정 ▼
사용자 지정 TCP ▼	TCP	8000	위치 무관 ▼

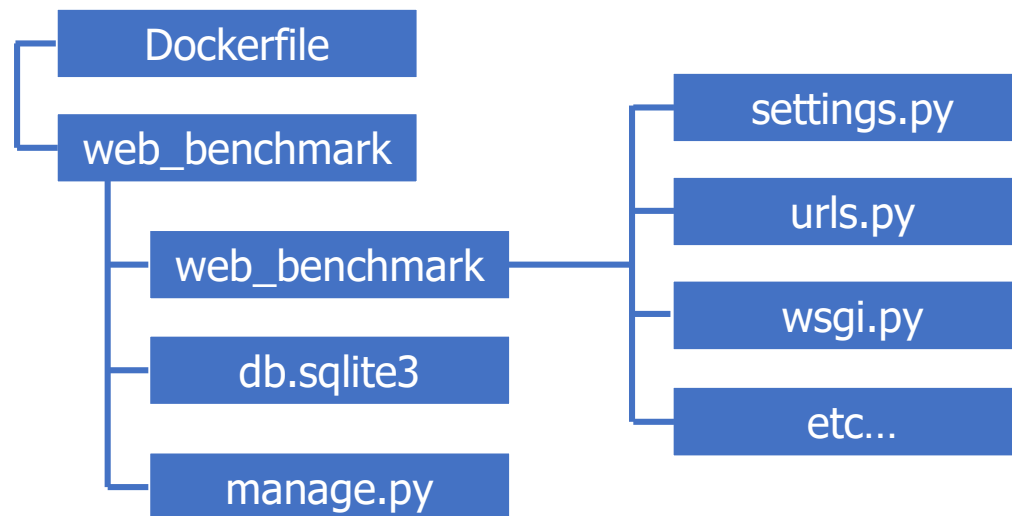
Docker를 통한 실제 서비스 배포

- EC2 인스턴스에 Dockerfile을 만든 후, 다음과 같이 작성해줍니다.

```
Dockerfile+
1 FROM python:3.6
2 MAINTAINER Hyunjoon_Jeong "with1015@unist.ac.kr"
3
4 RUN apt-get update
5 COPY . /app
6
7 WORKDIR /app/web_benchmark
8
9 RUN pip3 install Django
10
11 EXPOSE 8000
12
13 CMD ["python", "manage.py", "runserver", "0:8000"]
14
```


Docker를 통한 실제 서비스 배포

- Dockerfile과 web_benchmark 폴더는 반드시 다음과 같은 디렉토리 구조를 유지해야 합니다.



Docker를 통한 실제 서비스 배포

- Dockerfile이 완성되었으면, 다음과 같은 명령어를 통해 Dockerfile을 빌드하여 배포 이미지를 생성합니다.
 - `sudo docker build -t abc_mentoring:abc .`
- 성공적으로 build 되었을 경우 다음과 같이 docker images 명령어를 통해 생성된 이미지를 확인할 수 있습니다.

```
ubuntu@ip-172-31-36-224:~$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
abc_mentoring	abc	8102017c9943	30 seconds ago	933MB
python	3.6	1ce843b4cf8c	35 hours ago	875MB

```
ubuntu@ip-172-31-36-224:~$
```

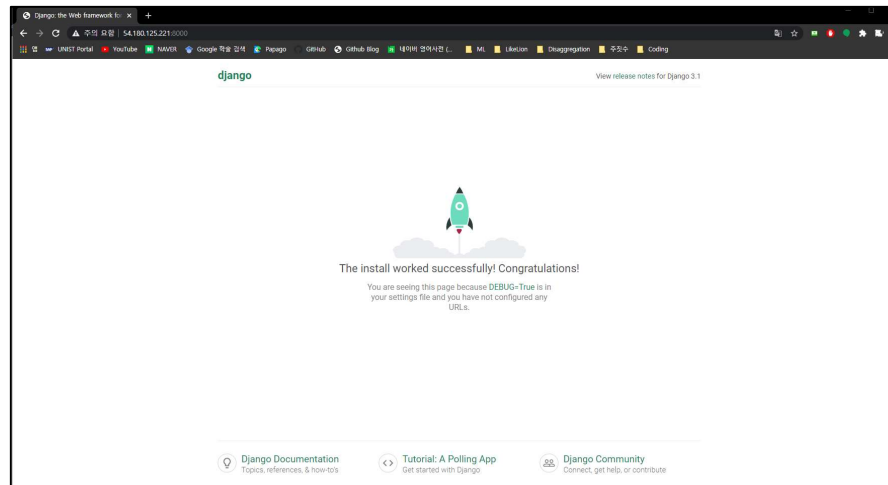
Docker를 통한 실제 서비스 배포

- 그 다음, 생성한 이미지를 컨테이너로 만들어야 합니다.
- 다음 명령어를 통해 이미지를 컨테이너로 만들고 실행할 수 있습니다.
 - `sudo docker run -p 8000:8000 --name abc_web abc_mentoring:abc`
 - 또는 `sudo docker run -d -p 8000:8000 --name abc_web abc_mentoring:abc`
- -d 옵션은 docker 컨테이너를 백그라운드로 실행하는 것을 의미합니다.
- -p 옵션은 컨테이너를 포트를 통해 접속할 수 있도록 열어줍니다.
- --name 옵션은 컨테이너에 이름을 부여합니다.

Docker를 통한 실제 서비스 배포

- 컨테이너가 성공적으로 실행되면 크롬을 열고 [EC2 IPv4 주소]:8000를 주소창에 입력하면 다음과 같이 웹 화면이 나타나게 됩니다.

```
ubuntu@ip-172-31-36-224:~$ docker run -p 8000:8000 --name abc_test abc_mentoring:abc
watching for file changes with StatReloader
[18/Feb/2021 07:34:09] "GET / HTTP/1.1" 200 16351
[18/Feb/2021 07:34:09] "GET /static/admin/css/fonts.css HTTP/1.1" 304 0
[18/Feb/2021 07:34:10] "GET /static/admin/fonts/Roboto-Bold-webfont.woff HTTP/1.1" 304 0
[18/Feb/2021 07:34:10] "GET /static/admin/fonts/Roboto-Regular-webfont.woff HTTP/1.1" 304 0
[18/Feb/2021 07:34:10] "GET /static/admin/fonts/Roboto-Light-webfont.woff HTTP/1.1" 304 0
```



Docker를 통한 실제 서비스 배포

- 한번 docker 컨테이너를 통해 서버에 올라가게 되면 컨테이너를 종료하지 않는 이상 터미널을 종료해도 웹 서버가 꺼지지 않습니다.
- 다음 명령어를 이용해 생성한 컨테이너와 빌드한 이미지를 제거할 수 있습니다.
- `sudo docker kill [웹 서비스 컨테이너 이름]`
- `sudo docker rm [웹 서비스 컨테이너 이름]`
- `sudo docker rmi [빌드했던 이미지 이름]`

참고 자료

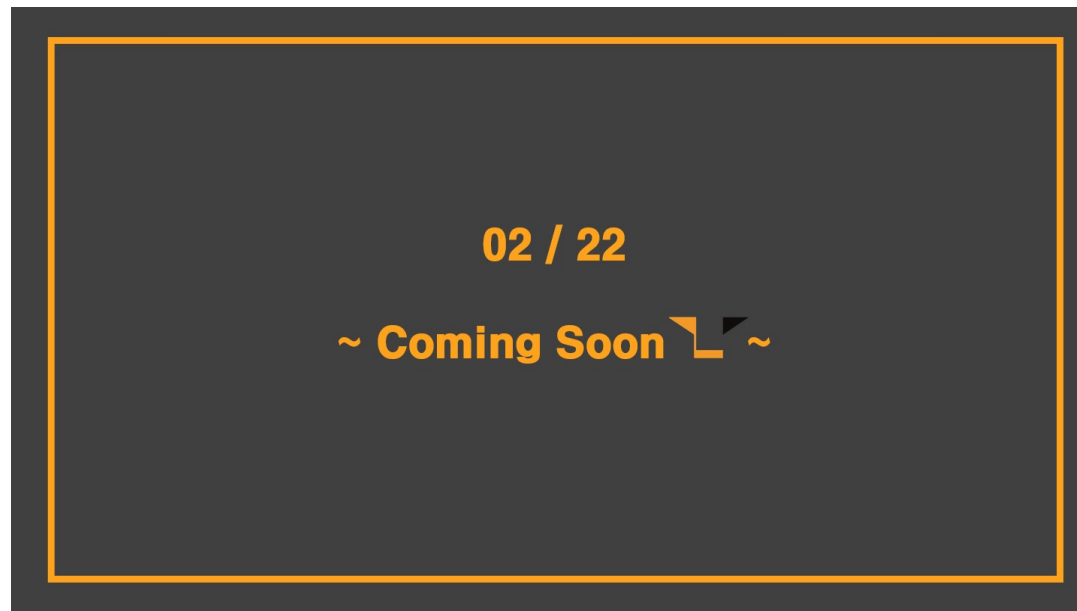
- Docker 명령어 관련
- <https://docs.docker.com/engine/reference/commandline/docker/>
- https://rampart81.github.io/post/docker_commands/

홍보

- 멋쟁이사자처럼(LikeLion)이란?
- 컴퓨터를 전혀 모르는 학생들이 코딩 교육을 통해 원하는 IT 서비스를 직접 런칭하는 것을 목표로 하는 동아리 입니다.
- 전국에 많은 대학에 동아리가 있고 매년 해커톤을 개최합니다.
 - (2020년 해커톤에서는 저희 학교 출신 팀이 은상 받았어요!)
- 주로 웹 서비스 구현에 초점이 맞춰져 있고, 전공에 상관 없이 다양한 사람들을 뽑는 것을 추구합니다.

홍보

- 2/22 ~ 3/12 UNIST 멋쟁이사자처럼 모집 예정



- 지원 방법은 아래 페이스북 참고 해주세요 (혹은 에타에 22일 공고 예정)
- <https://www.facebook.com/likelion.UNIST/>

Thank you

