

# Github에 있는 프로젝트를 AWS EC2에 git clone을 통해 내려받아 jar 파일 빌드하기

## 배포용 버전

```
Gradle-groovy
Java : 17
Gradle : 8.1
Spring Boot 플러그인 : 3.0.5

Thymeleaf : implementation 'org.thymeleaf.extras:thymeleaf-extras-springsecurity6'

// MySQL 대신 h2 DB 로 대체 (추후에 MySQL 연동)
DB : H2 (runtimeOnly 'com.h2database:h2')
```

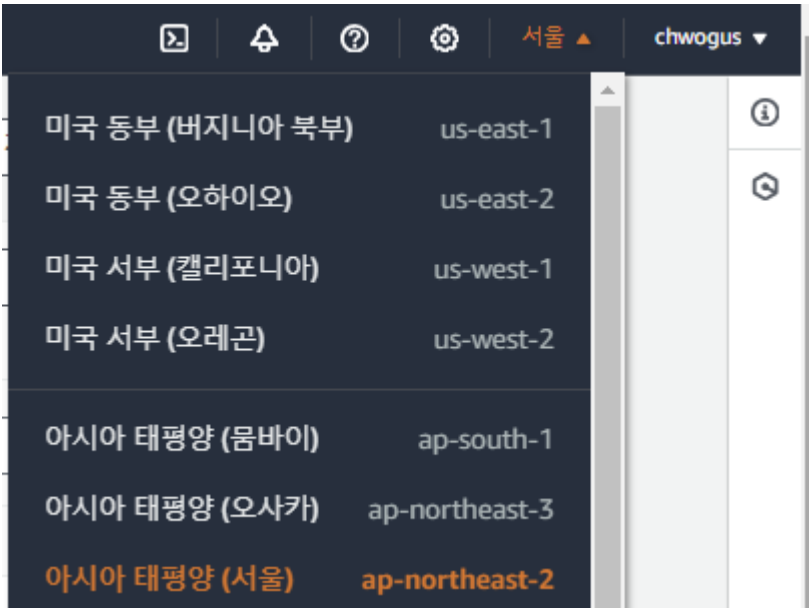
## 배포전 주의사항

- 만약 테스트코드가 통과되지 않는다면 배포가 안될 수 있기 때문에 배포 전에 확인해봐야한다.
- 특히 SpringBootTest에서 contextLoads 함수가 비어있으면 jar 파일을 만들지 못해 배포가 불가능하니 출력문이라도 추가해 주자!

## 1. AWS 세팅하기

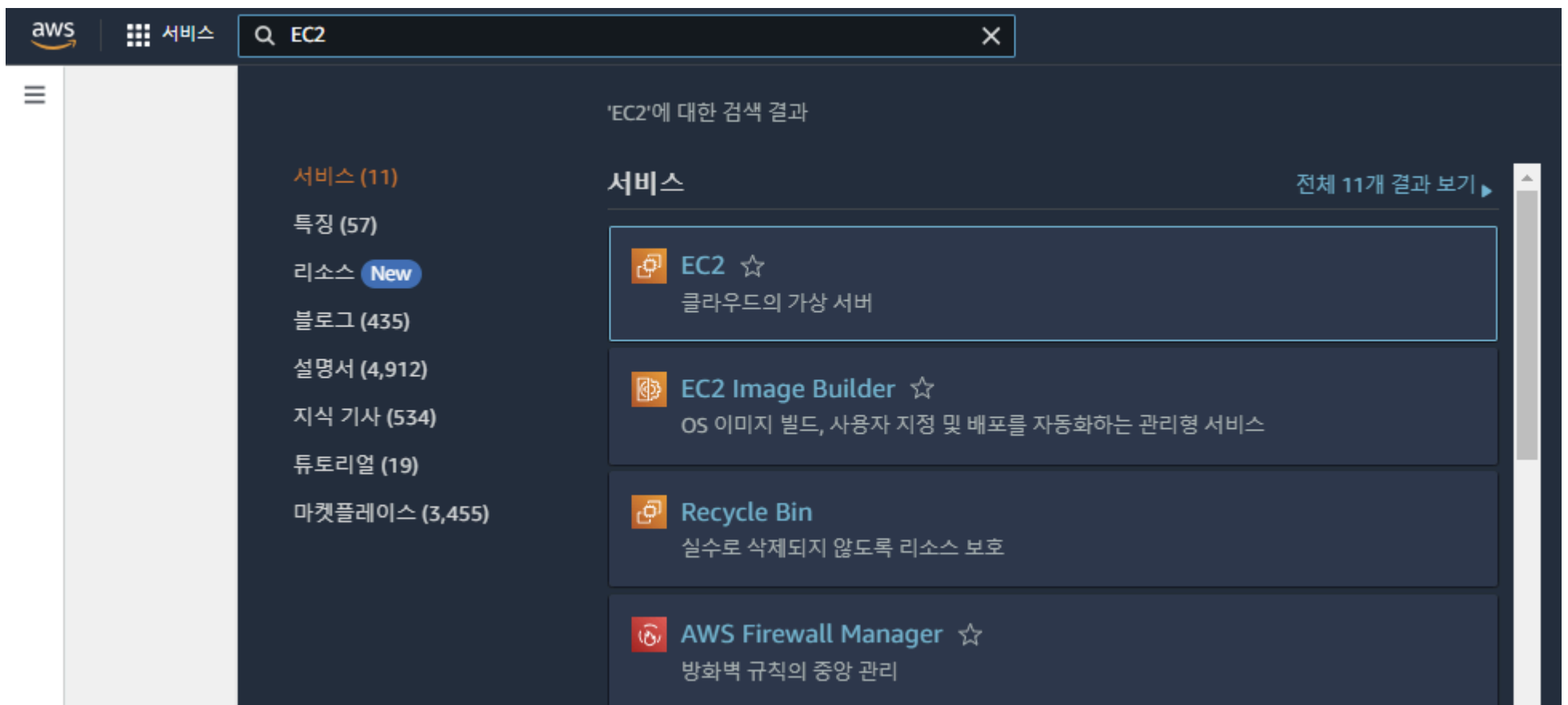
### AWS Region 설정

- 우측에서 지역을 서울로 설정해주자!
- 만약 만들어둔 인스턴스가 없다면 지역을 확인해보자!



### 인스턴스 생성하기

- 좌측 상단에 EC2 를 입력해서 EC2 대시보드 로 이동하자



- EC2 대시보드에서 **인스턴스 시작** 을 눌러 인스턴스를 생성해보자

리소스

[EC2 글로벌 보기](#)

아시아 태평양 (서울) 리전에서 다음 Amazon EC2 리소스를 사용하고 있음:

인스턴스(실행 중)	0	로드 밸런서	0	배치 그룹	0
보안 그룹	3	볼륨	2	스냅샷	0
인스턴스	2	전용 호스트	0	키 페어	1
탄력적 IP	0	Auto Scaling 그룹	0		

인스턴스 시작

시작하려면 클라우드의 가상 서버인 Amazon EC2 인스턴스를 시작하십시오.

인스턴스 시작

서버 마이그레이션

서비스 상태

[AWS Health 대시보드](#)

리전  
아시아 태평양 (서울)

상태  
이 서비스가 정상적으로 작동 중입니다.

## 원하는 인스턴스 명 지어주기

이름 및 태그 정보

이름

예: 내 웹 서버

추가 태그 추가

## OS 설정

- 여기서는 Ubuntu 64비트(x86) 으로 진행을 할 것이다.
- AMI 의 경우 프로젝트의 크기에 따라 자유롭게 설정하면 되며, 여기서는 프리티어 사용 가능한 Ubuntu Server 22.04 LTS (HVM), SSD Volumn Type 을 사용할 것이다.
  - 프리티어가 아닌 다른 AMI 는 과금이 발생할 수 있으니 주의하자.

## ▼ 애플리케이션 및 OS 이미지(Amazon Machine Image) 정보

AMI는 인스턴스를 시작하는 데 필요한 소프트웨어 구성(운영 체제, 애플리케이션 서버 및 애플리케이션)이 포함된 템플릿입니다. 아래에서 찾고 있는 항목이 보이지 않으면 AMI를 검색하거나 찾아보세요.

🔍 수천 개의 애플리케이션 및 OS 이미지를 포함하는 전체 카탈로그 검색

### Quick Start

Amazon Linux  
aws


macOS  
Mac

Ubuntu  
ubuntu®

Windows  
Microsoft

Red Hat  
Red Hat

SUSE L  
SUS

  
더 많은 AMI 찾아보기  
AWS, Marketplace 및 커뮤니티의 AMI 포함

### Amazon Machine Image(AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type  
ami-09a7535106fbd42d5 (64비트(x86)) / ami-0e4b1df799f55b8bb (64비트(Arm))  
가상화: hvm    ENA 활성화됨: true    루트 디바이스 유형: ebs

프리 티어 사용 가능 ▼

### 설명

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2024-03-01

### 아키텍처

64비트(x86) ▼

### AMI ID

ami-09a7535106fbd42d5

확인된 공급 업체

## 인스턴스 유형 선택

- 인스턴스 유형 또한 프리티어에서 사용가능한 t2,micro 를 사용할 예정이다.

## ▼ 인스턴스 유형 정보 | 조언 받기

### 인스턴스 유형

t2.micro    프리 티어 사용 가능

패밀리: t2    1 vCPU    1 GiB 메모리    현재 세대: true  
온디맨드 RHEL 기본 요금: 0.0744 USD 시간당  
온디맨드 Linux 기본 요금: 0.0144 USD 시간당  
온디맨드 SUSE 기본 요금: 0.0144 USD 시간당  
온디맨드 Windows 기본 요금: 0.019 USD 시간당

☒ 모든 세대  
인스턴스 유형 비교

소프트웨어가 사전 설치된 AMI에는 추가 비용이 적용됩니다.

## 키 페어 생성하기

키 페어는 EC2 인스턴스에 원격으로 접속하기 위해 필수적으로 필요하기 때문에 생성해줘야한다. 또한, 키 페어는 한번 생성하면 두번다시 다운로드 받을 수 없기 때문에 잘 보관하고 있어야한다.

- 우측의 새 키 페어 생성 혹은 이미 존재한다면 선택

## ▼ 키 페어(로그인) 정보

키 페어를 사용하여 인스턴스에 안전하게 연결할 수 있습니다. 인스턴스를 시작하기 전에 선택한 키 페어에 대한 액세스 권한이 있는지 확인하세요.

키 페어 이름 - 필수

선택 ▼

🔄 새 키 페어 생성

- 다음과 같이 키 페어 이름을 설정해주고 생성해주고 다운을 받아두자.

키 페어 생성

키 페어 이름  
키 페어를 사용하면 인스턴스에 안전하게 연결할 수 있습니다.  
sshKey  
이름에는 최대 255개의 ASCII 문자가 포함됩니다. 앞 또는 뒤에 공백을 포함할 수 없습니다.

키 페어 유형  

☒ RSA  
RSA 암호화된 프라이빗 및 퍼블릭 키 페어

☐ ED25519  
ED25519 암호화된 프라이빗 및 퍼블릭 키 페어

프라이빗 키 파일 형식  

☒ .pem  
OpenSSH와 함께 사용

☐ .ppk  
PuTTY와 함께 사용

⚠

 메시지가 표시되면 프라이빗 키를 사용자 컴퓨터의 안전하고 액세스 가능한 위치에 저장합니다. 나중에 인스턴스에 연결할 때 필요합니다. 자세히 알아보기

취소

키 페어 생성

## 네트워크 세팅

보안 그룹은 방화벽에 관한 내용으로 추후에 따로 생성해 세팅할 예정이다. 만약 기존에 보안 그룹을 설정해두었다면 기존 보안 그룹을 사용해도 된다.

- SSH 트래픽 허용 부분 체크하기
  - EC2 인스턴스를 생성하고 생성한 인스턴스에 로컬 머신(노트북, 데스크탑)으로 접속하기 위한 부분으로써 이때 SSH 통신을 사용한다.
  - SSH 통신을 할 때 ip 제한을 어떻게 할것인지에 관한 내용으로
    - 고정된 ip로만 접근할 경우에는 내 ip 를 선택해주면 된다.
    - 이동하면서 사용할 경우 위치무관(0.0.0.0/0)으로 설정해주면 된다.

해당 설정은 추후에 보안 그룹 탭에서 변경 가능하므로 넘어가도 좋다.

▼ 네트워크 설정

정보

편집

네트워크

정보

vpc-039ab70453684022c

서브넷

정보

기본 설정 없음(가용 영역의 기본 서브넷)

퍼블릭 IP 자동 할당

정보

활성화

Additional charges apply when outside of free tier allowance

방화벽(보안 그룹)

정보

보안 그룹은 인스턴스에 대한 트래픽을 제어하는 방화벽 규칙 세트입니다. 특정 트래픽이 인스턴스에 도달하도록 허용하는 규칙을 추가합니다.

☒ 보안 그룹 생성

☐ 기존 보안 그룹 선택

다음 규칙을 사용하여 'launch-wizard-3'(이)라는 새 보안 그룹을 생성합니다.

☒ 다음에서 SSH 트래픽 허용  
인스턴스 연결에 도움

위치 무관

0.0.0.0/0

☐ 인터넷에서 HTTPS 트래픽 허용  
예를 들어 웹 서버를 생성할 때 엔드포인트를 설정하려면

☐ 인터넷에서 HTTP 트래픽 허용  
예를 들어 웹 서버를 생성할 때 엔드포인트를 설정하려면

⚠

소스가 0.0.0.0/0인 규칙은 모든 IP 주소에서 인스턴스에 액세스하도록 허용합니다. 알려진 IP 주소의 액세스만 허용하도록 보안 그룹을 설정하는 것이 좋습니다.

×

## 스토리지 세팅

- 프리티어의 경우 최대 30 GiB 까지 지원해주므로 30으로 설정

▼ 스토리지 구성

정보

고급

1x

30

GiB

gp2

루트 볼륨 (암호화되지 않음)

ⓘ

프리 티어를 사용할 수 있는 고객은 최대 30GB의 EBS 범용(SSD)또는 마그네틱 스토리지를 사용할 수 있습니다.

×

새 볼륨 추가

선택한 AMI에 인스턴스가 허용하는 것보다 많은 인스턴스 스토어 볼륨이 포함되어 있습니다. AMI에서 처음 0개의 인스턴스 스토어 볼륨에만 액세스할 수 있습니다.

🕒

백업 정보를 보려면 새로 고침 클릭

🔄

할당한 태그에 따라 Data Lifecycle Manager 정책으로 인스턴스를 백업할지 여부가 결정됩니다.

0 x 파일 시스템

편집

## 인스턴스 최종 생성

- 우측의 인스턴스 시작 을 눌러서 최종 생성을 완료하자

▼ 요약

인스턴스 개수 | 정보

1

소프트웨어 이미지(AMI)

Canonical, Ubuntu, 22.04 LTS, ...[더 보기](#)

ami-09a7535106fbd42d5

가상 서버 유형(인스턴스 유형)

t2.micro

방화벽(보안 그룹)

새 보안 그룹

스토리지(볼륨)

1개의 볼륨 – 30GiB

❗ 프리 티어: In your first year includes

750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

×

취소

인스턴스 시작

[명령 검토](#)

인스턴스 생성이 완료된 모습

- 다음과 같은 창이 나오고 EC2 대시보드-인스턴스 로 이동하면 인스턴스가 추가된걸 확인할 수 있다.

EC2 > 인스턴스 > Launch an instance

🟢 성공

인스턴스를 시작했습니다. (i-03bc49a62d61dc8cd)

[로그 시작](#)

다음 단계

결제 및 프리 티어 사용 알림 생성

비용을 관리하고 높은 금액의 청구서를 방지하려면 결제 및 프리 티어 사용 임계값에 대한 이메일 알림을 설정합니다.

[결제 알림 생성](#)

인스턴스에 연결

인스턴스가 실행되면 로컬 컴퓨터에서 인스턴스에 로그인합니다.

[인스턴스에 연결](#)

[자세히 알아보기](#)

RDS 데이터베이스 연결

EC2 인스턴스와 데이터베이스 간의 트래픽 흐름을 허용하도록 연결을 구성합니다.

[RDS 데이터베이스 연결](#)

[새 RDS 데이터베이스 생성](#)

[자세히 알아보기](#)

[모든 인스턴스 보기](#)

2. 보안그룹 설정하기

생성된 인스턴스를 눌러 인스턴스 요약으로 들어가보자

인스턴스 (3) 정보

🔄

연결

인스턴스 상태 ▼

작업 ▼

인스턴스 시작 ▼

🔍 인스턴스를 속성 또는 (case-sensitive) 태그로 찾기

모든 상태 ▼

<input type="checkbox"/>	Name ↗	인스턴스 ID	인스턴스 상태 ▼	인스턴스 유형 ▼	상태 검사	경보 상태	가용 영역 ▼	퍼블릭 IPv4 DNS ▼	퍼블릭 IPv4 ... ▼	탄력적 IP	IPv6 IP ▼	모니터링
<input type="checkbox"/>	wuzuzu	i-03bc49a62d61dc8cd	🟢 실행 중 🔍 🔍	t2.micro	🕒 초기화	<a href="#">경보 보기</a> +	ap-northeast-2c	ec2-43-201-113-100.ap...	43.201.113.100	-	-	disa

보안 그룹이란?

AWS 에서 제공하는 방화벽으로 인바운드 규칙, 아웃바운드 규칙이 존재

- 인바운드 규칙(inbound) : 외부에서 EC2나 RDS 등의 내부로 접근할때 사용되는 방화벽 규칙
- 아웃바운드 규칙(outbound) : EC2나 RDS 등의 내부에서 외부로 접근할때 사용되는 방화벽 규칙

인스턴스 요약의 보안 을 눌러 보안 그룹 확인

세부 정보 | 상태 및 경보 신규 | 모니터링 | 보안 | 네트워킹 | 스토리지 | 태그

▼ 보안 세부 정보

IAM 역할  
-

보안 그룹  
sg-09fbc369e3f8ee286 (launch-wizard-3)

소유자 ID  
381025169114

시작 시간  
Mon Apr 01 2024 11:37:21 GMT+0900 (한국 표준시)

▼ 인바운드 규칙

Q 필터 규칙

이름	보안 그룹 규칙 ID	포트 범위	프로토콜	원본	보안 그룹	설명
-	sgr-0a7753436ebe57129	22	TCP	0.0.0.0/0	<a href="#">launch-wizard-3</a>	-

▼ 아웃바운드 규칙

Q 필터 규칙

이름	보안 그룹 규칙 ID	포트 범위	프로토콜	대상	보안 그룹	설명
-	sgr-0fbdac7d36a1e6235	전체	전체	0.0.0.0/0	<a href="#">launch-wizard-3</a>	-

보안 그룹으로 이동

- 좌측의 네트워킹 및 보안 - 보안 그룹 혹은 인스턴스 요약 - 보안 - 보안 그룹 을 눌러서 이동

- ▼ 네트워킹 및 보안
- 보안 그룹

탄력적 IP

배치 그룹

키 페어

네트워크 인터페이스

보안그룹 생성하기(이미 존재한다면 생략 가능)

- 보안 그룹 생성을 눌러 새로운 보안 그룹을 생성하기

보안 그룹 (4) 정보

🔄 작업 ▼ 보안 그룹을 CSV로 내보내기 ▼ **보안 그룹 생성**

Q Find resources by attribute or tag

< 1 > ⓘ

<input type="checkbox"/>	Name ▼	보안 그룹 ID ▼	보안 그룹 이름 ▼	VPC ID ▼	설명 ▼	소유자 ▼	인바운드 규칙
<input type="checkbox"/>	-	<a href="#">sg-0adc7be3f250fe108</a>	default	<a href="#">vpc-039ab70453684022c</a>	default VPC security group	381025169114	1 권한 항목

그룹 이름과 설명 작성

기본 세부 정보

보안 그룹 이름 정보

MyWebServerGroup

생성 후에는 이름을 편집할 수 없습니다.

설명 정보

개발자에게 SSH 액세스 허용

VPC 정보

vpc-039ab70453684022c ▼

인바운드 규칙 설정

- 인바운드 규칙 - 규칙 추가를 눌러 다음과 같이 인바운드 규칙을 설정하자
- 이때 0.0.0.0/0 은 Anywhere-IPv4 이다.

인바운드 규칙 정보						
보안 그룹 규칙 ID	유형 정보	프로토콜 정보	포트 범위 정보	소스 정보	설명 - 선택 사항 정보	
sgr-00acf7e28b99aee97	사용자 지정 TCP ▼	TCP	8080	사용자 지정 ▼	Q 0.0.0.0/0 ✕	삭제
sgr-0c0db4959bed73823	SSH ▼	TCP	22	사용자 지정 ▼	Q 0.0.0.0/0 ✕	삭제
sgr-0feb37e510be847d5	HTTPS ▼	TCP	443	사용자 지정 ▼	Q 0.0.0.0/0 ✕	삭제
sgr-01983aed43ea62c8b	HTTP ▼	TCP	80	사용자 지정 ▼	Q 0.0.0.0/0 ✕	삭제

위에서부터 순서대로

- 사용자 지정 TCP** : 스프링 부트 기반 서버를 열어줄것이기 때문에 사용자 지정으로 8080 포트를 설정해준 뒤 url을 아는 누구나 접속할수있도록 Anywhere-IPv4 로 설정
- SSH** : 원격 EC2 인스턴스에 접속할때 사용되는 ssh 관련 방화벽으로 자택의 고정 ip가 아닌 Anywhere-IPv4 로 설정
  - ssh는 기본 포트 연결로 22번 포트를 사용
- HTTP** : HTTP 연결시 사용
- HTTPS** : HTTPS 연결시 사용 (HTTPS 를 사용하려면 별도의 과정이 필요한데 그 과정은 여기서는 생략함)아웃바운드 규칙은 따로 세팅해주지 않고 기본세팅으로 설정

보안 그룹 설정하기

- 다시 생성한 인스턴스의 인스턴스 요약 으로 돌아와서 우측 상단의 작업 - 보안 - 보안 그룹 변경 클릭

인스턴스 상태 ▼		작업 ▲
3-1	연결	
	인스턴스 상태 관리	
	인스턴스 설정	▶
	네트워킹	▶
	보안 그룹 변경	▶
	Windows 암호 가져오기	▶
IAM 역할 수정		▶

- 다음과 같이 생성한 보안 그룹을 선택해주고 저장해주면 된다.



보안 그룹 변경

정보

Amazon EC2는 선택한 보안 그룹의 모든 규칙을 평가하여 인스턴스에서 송수신되는 인바운드 및 아웃바운드 트래픽을 제어합니다. 이 창을 사용하여 보안 그룹을 추가 및 제거할 수 있습니다.

인스턴스 세부 정보

인스턴스 ID

i-03bc49a62d61dc8cd (wuzuzu)

네트워크 인터페이스 ID

eni-0e15ae58933001919

연결된 보안 그룹

네트워크 인터페이스에 하나 이상의 보안 그룹을 추가합니다. 보안 그룹을 제거할 수도 있습니다.

sg-09554942f0f749a43

✕

보안 그룹 추가

네트워크 인터페이스와 연결된 보안 그룹(eni-0e15ae58933001919)

보안 그룹 이름	보안 그룹 ID	
launch-wizard-2	sg-09554942f0f749a43	제거

취소

저장

- 다음과 같이 인바운드 규칙이 추가된걸 확인할 수 있다.

세부 정보 | 상태 및 경보 신규 | 모니터링 | **보안** | 네트워킹 | 스토리지 | 태그

▼ 보안 세부 정보

IAM 역할

-

보안 그룹

sg-0a1cb7fcca2361f5b (launch-wizard-1)

소유자 ID

381025169114

시작 시간

Mon Apr 01 2024 11:37:21 GMT+0900 (한국 표준시)

▼ 인바운드 규칙

🔍 필터 규칙

< 1 >

이름	보안 그룹 규칙 ID	포트 범위	프로토콜	원본	보안 그룹	설명
-	sgr-00acf7e28b99aee97	8080	TCP	0.0.0.0/0	<a href="#">launch-wizard-1</a>	-
-	sgr-0c0db4959bed73823	22	TCP	0.0.0.0/0	<a href="#">launch-wizard-1</a>	-
-	sgr-0feb37e510be847d5	443	TCP	0.0.0.0/0	<a href="#">launch-wizard-1</a>	-
-	sgr-01983aed43ea62c8b	80	TCP	0.0.0.0/0	<a href="#">launch-wizard-1</a>	-

### 3. EC2 우분투 콘솔에 접속하기

이제 EC2 생성을 완료했으니 Ubuntu 에 접속을 해야하는데 접속하는 방법으로는 2가지 방법이 있다.

1. EC2 를 생성할때 다운받은 ssh 키를 이용해 원격 접속하기 (window 의 경우 별도 작업 필요)
2. EC2 대시보드에서 접근(여기서는 이 방법을 사용할 예정)

#### 인스턴스 요약에서 연결 클릭

i-03bc49a62d61dc8cd (wuzuzu)에 대한 인스턴스 요약 정보

2 minutes 전에 업데이트됨

🔄

연결

인스턴스 상태 ▼

작업 ▼

- 다음과 같은 창에서 연결을 눌러 Ubuntu와 연결하기

## 인스턴스에 연결 정보

다음 옵션 중 하나를 사용하여 인스턴스 i-03bc49a62d61dc8cd (wuzuzu)에 연결

EC2 인스턴스 연결	Session Manager	SSH 클라이언트	EC2 직렬 콘솔
-------------	-----------------	-----------	-----------

인스턴스 ID

i-03bc49a62d61dc8cd (wuzuzu)

연결 유형

☒ EC2 Instance Connect을 사용하여 연결  
퍼블릭 IPv4 주소가 있는 EC2 인스턴스 연결 브라우저 기반 클라이언트를 사용하여 연결합니다.

☐ EC2 인스턴스 연결 엔드포인트를 사용하여 연결  
프라이빗 IPv4 주소 및 VPC 엔드포인트가 있는 EC2 인스턴스 연결 브라우저 기반 클라이언트를 사용하여 연결합니다.

퍼블릭 IP 주소

43.201.113.100

사용자 이름

인스턴스를 시작하는 데 사용되는 AMI에 정의된 사용자 이름을 입력합니다. 사용자 지정 사용자 이름을 정의하지 않은 경우 기본 사용자 이름인 ubuntu을(를) 사용합니다.

**참고:** 대부분의 경우 기본 사용자 이름 ubuntu은(는) 정확합니다. 하지만 AMI 사용 지침을 읽고 AMI 소유자가 기본 AMI 사용자 이름을 변경했는지 확인하세요.

취소

연결

- Ubuntu 가 연결된 모습

```
aws | 서비스 | 🔍 검색 | [알트+S]
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 6.5.0-1014-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/pro

System information as of Mon Apr  1 03:04:17 UTC 2024

System load:  0.0               Processes:           96
Usage of /:   5.4% of 28.89GB   Users logged in:    0
Memory usage: 21%              IPv4 address for eth0: 172.31.47.251
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-47-251:~$
```

## 4. 우분투 콘솔에서 git ssh 연동후 git clone 하기

정적 파일 배포는 방식이 두 가지가 있는데 여기서는 1번 방식 사용

1. EC2에서 프로젝트 git clone 후 실행하기
2. 로컬 머신에서 jar 파일하여 EC2에 복사 후 실행

Ubuntu에서의 복사/붙여넣기는 `ctrl+insert` / `shift+insert` 를 이용하면 된다.

## Git 설치하기

```
sudo apt-get install git
```

```
ubuntu@ip-172-31-47-251:~$ sudo apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.10).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-47-251:~$
```

## Git 설치 확인

```
git --version
```

```
ubuntu@ip-172-31-47-251:~$ git --version
git version 2.34.1
```

## Github에서 SSH KEY 생성하기

아래 명령어를 통해 `.ssh` 디렉토리에서 키페어를 생성하게 되고 `id_res.pub` 파일이 생성됩니다.

```
cd ~/.ssh
ssh-keygen -t rsa -C github계정 메일(example@github.com)
```

```
ubuntu@ip-172-31-47-251:~/.ssh$ ssh-keygen -t rsa -C chwogus0303@naver.com
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:2gNCL3ZdXlCDYEhFW5XRgXFRhM0kPw2Yi8fDr4zDZm8 chwogus0303@naver.com
The key's randomart image is:
+---[RSA 3072]-----+
|      ..+=.oBX&B|
|      .. o+o+==|
|      .  o+ ...o|
|      . . . . .*. .|
|      + + S  ..o  |
|      . + +      . |
|      . o. o .    |
|      . * E       |
|      o +.        |
+-----[SHA256]-----+
ubuntu@ip-172-31-47-251:~/.ssh$ ls
authorized_keys  id_rsa  id_rsa.pub
```

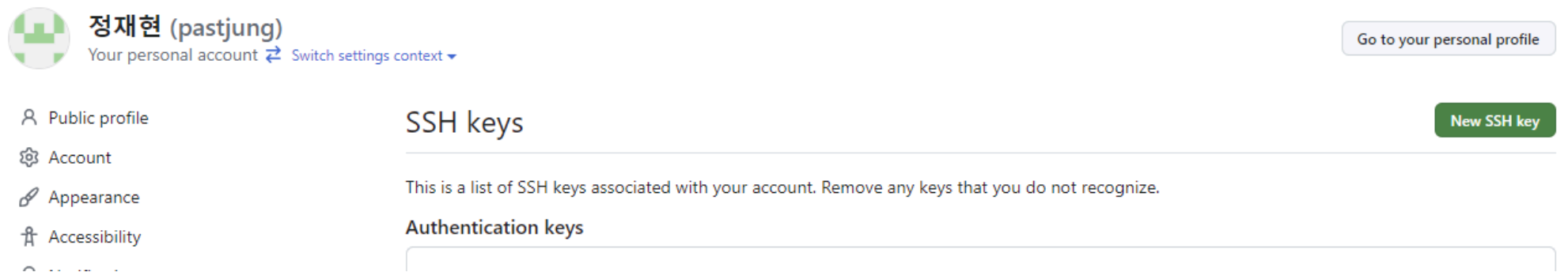
## id\_rsa.pub 을 통해서 ssh 키페어 확인

```
cat id_rsa.pub
```

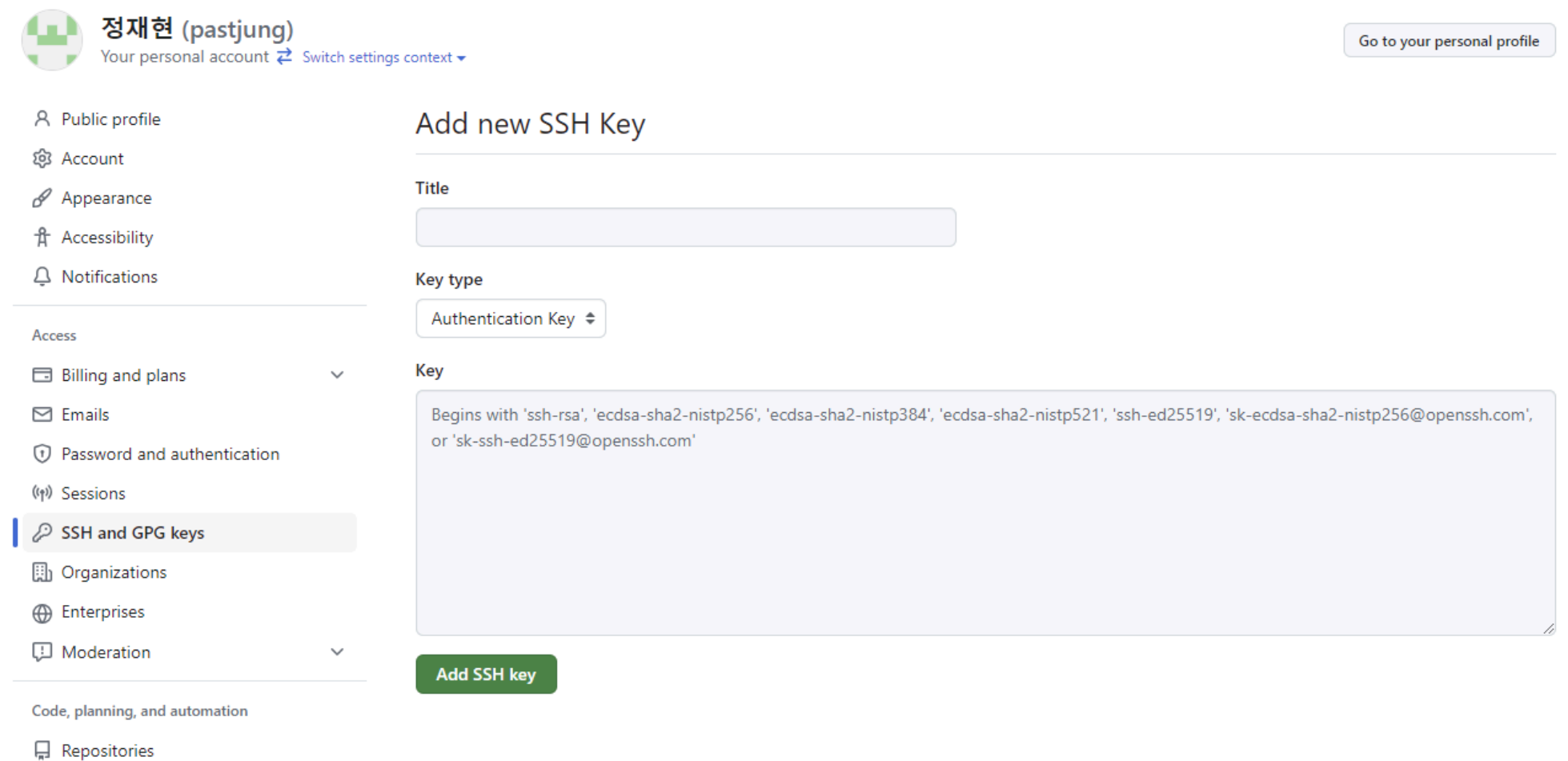
## Github 에 ssh 키페어 추가

- `cat` 명령어로 `id_rsa.pub` 파일을 출력후 이것을 깃헙 ssh에 저장을 해야 함

- github -> setting -> SSH and GPG keys 탭으로 이동후 new SSH key 버튼을 클릭

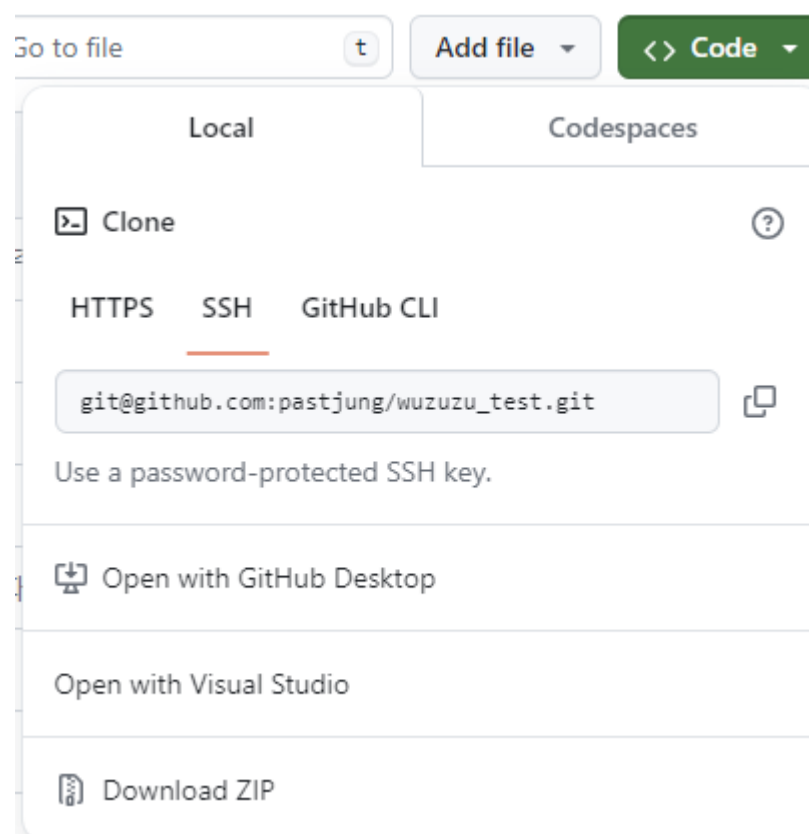


- 타이틀에는 임의로 정해주시면 되고 key 부분에 복사한 위에서 출력한 id\_rsa.pub 값을 넣어주면 된다.



## git clone 하기

- Github 레포지토리로 이동후 code 버튼 -> ssh 탭을 누르고 나온 주소를 복사



- EC2에 접속한 터미널에서 git clone할 디렉토리로 이동후 복사한 값을 아래 명령어에 포함시켜 입력

git clone [ 복사한 github ssh 주소 ]

```
ubuntu@ip-172-31-47-251:~/.ssh$ git clone git@github.com:pastjung/wuzuzu_test.git
Cloning into 'wuzuzu_test'...
The authenticity of host 'github.com (20.200.245.247)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 584, done.
remote: Counting objects: 100% (584/584), done.
remote: Compressing objects: 100% (233/233), done.
remote: Total 584 (delta 154), reused 584 (delta 154), pack-reused 0
Receiving objects: 100% (584/584), 100.55 KiB | 277.00 KiB/s, done.
Resolving deltas: 100% (154/154), done.
```

- 프로젝트 파일이 생성된 모습

```
Resolving deltas: 100% (154/154), done.
ubuntu@ip-172-31-47-251:~/.ssh$ ls
authorized keys  id_rsa  id_rsa.pub  known hosts  known hosts.old  wuzuzu_test
```

## 5. 내려받은 파일로 빌드하고 jar 파일 실행시켜 스프링 부트 서버 실행시키기

### 시스템 패키지 업데이트하기

- 자바 17이 없을수도 있기 때문에 미리 최신화

```
sudo apt update
```

### 자바(17 버전) 설치하기

```
sudo apt install openjdk-17-jdk
```

### build 를 위한 gradlew 의 권한 추가

- 먼저 생성한 파일로 이동

```
ubuntu@ip-172-31-47-251:~/.ssh$ cd wuzuzu_test
ubuntu@ip-172-31-47-251:~/.ssh/wuzuzu_test$ ls
build.gradle  docs  gradle  gradlew  gradlew.bat  settings.gradle  src
```

- 파일 권한 추가

```
chmod +x ./gradlew
```

### gradlew 파일 실행

```
./gradlew build
```

### 빌드가 성공적으로 끝났다면 build -> libs 디렉토리가 새롭게 생기고

```
// jar 파일이 생성되어 있음
cd build/libs
```

```
ex.
cd build/libs
```

## 서버 실행파일을 이용해 백그라운드로 서버 실행

- 서버 실행(nuhub 참고 자료)

```
// 2개의 파일이 존재하는데 오른쪽 파일을 실행하면 된다.  
nohup java -jar [생성한 파일]-0.0.1-SNAPSHOT.jar &
```

```
ex.  
nohup java -jar wuzuzu-0.0.1-SNAPSHOT.jar &
```

## 서버가 백그라운드에서 잘 돌아가고 있는지 확인

```
ps
```

## 서버가 돌아가고 있는 내용을 실시간으로 확인하기

```
tail -f nohup.out // 여기서 오류 확인도 가능
```

## 배포가 되었는지 확인하기 위해 CMD 에서 telnet 에 접근해보기

```
telnet [public 주소] 8080
```

## 백그라운드로 돌아가고 있는 파일 확인

```
jobs
```

## 백그라운드로 돌아가고 있는 파일 중지

```
fg %[jobs로 확인한 번호]  
ctrl + c
```