

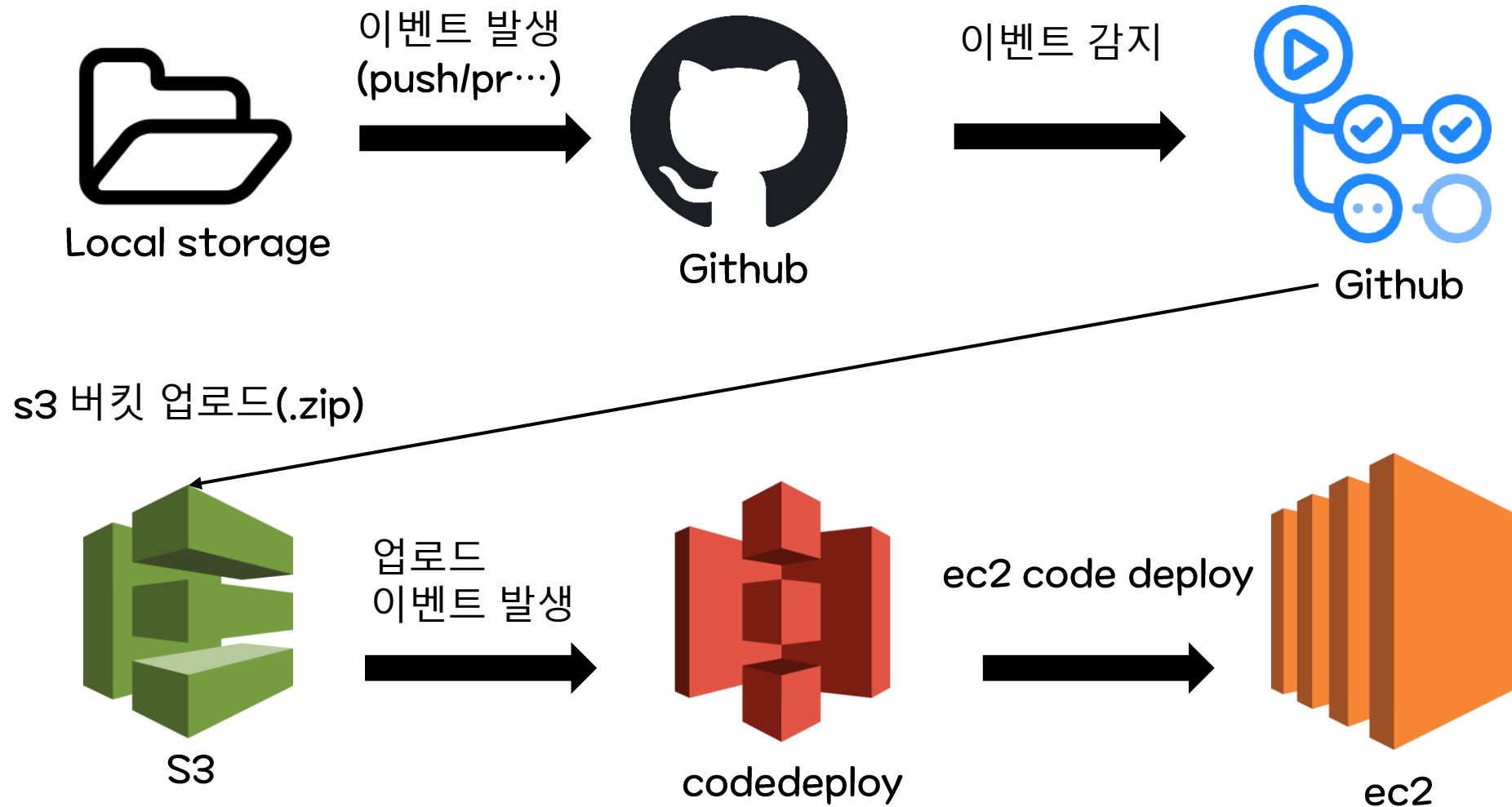
# CI-CD

ssosso.table

# 1. introduction

- github action을 이용한 배포 자동화 구현
- 사용 도구
  - vue.js quasar
    - client app
    - SSR 예제 프로젝트 활용
  - aws
    - ec2: client app 배포 서버
    - s3: client app 저장 버킷
    - codedeploy: s3 -> ec2 배포 자동화
  - github action
    - local -> s3 배포 자동화
- 참조
  - <https://ms3864.tistory.com/383?category=1003779>

# 1. introduction: flowchart



## 2. create repository


### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

---

Owner \*

Repository name \*

 bear-frog ▾

 / 

ci-cd ✓

Great repository names are short and memorable. Need inspiration? How about [congenial-octo-happiness?](#)

**Description** (optional)

---

☒  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**

You choose who can see and commit to this repository.

---

**Initialize this repository with:**

Skip this step if you're importing an existing repository.

☒ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Node ▾

**Choose a license**

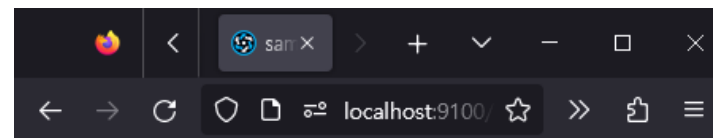
A license tells others what they can and can't do with your code. [Learn more.](#)

License: None ▾

This will set  **main** as the default branch. Change the default name in your [settings](#).

## 2. create repository: upload vue project

1. team-study의 SSR 프로젝트 파일을 가져옵니다
2. 프로젝트 빌드(npm run build)
3. 로컬 환경 프로젝트 정상 동작 확인  
(npm run dev or npm start)

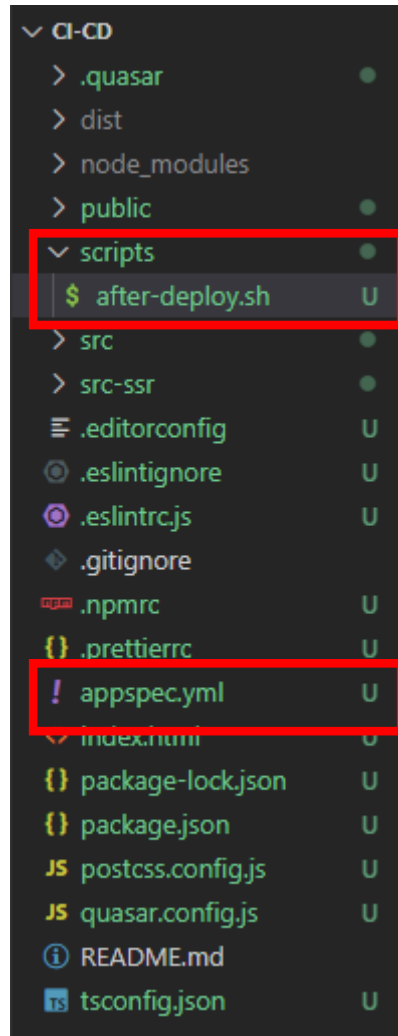


안녕하세요  
김고구마 님!

지금 시간은  
"2023-03-15T08:16:25.916Z" 입니다

### 3. create repository: add deploy script

- appspec.yml
  - scripts/after-deploy.sh
  - 두 파일을 추가합니다
- codedeploy에 의해  
서버에 배포된 이후를 정의합니다



# 3. create repository: appspec.yml

```
version: 0.0
os: linux
files:
    - source: /
      destination: /home/ubuntu/app
      overwrite: yes
permissions:
    - object: /home/ubuntu
      pattern: '**'
      owner: ubuntu
      group: ubuntu
hooks:
    AfterInstall:
        - location: scripts/after-deploy.sh
          timeout: 180
          runas: ubuntu
```

참조

<https://github.com/bear-frog/ci-cd/blob/main/appspec.yml>

# 3. create repository: scripts/after-deploy.sh

```
#!/bin/bash
```

```
# 어플리케이션 디렉터리
```

```
REPOSITORY=/home/ubuntu/app/dist/ssr
```

```
# 배포 디렉터리로 이동
```

```
cd $REPOSITORY
```

```
# 종속성 설치
```

```
sudo npm i
```

```
# MARK: pm2를 사용해 80번 포트에 앱 실행
```

```
# 참조
```

```
# https://unchae.tistory.com/entry/PM2-80-443%ED%8F%AC%ED%8A%B8-%EC%82%AC%EC%9A%A9
```

```
authbind --deep pm2 reload index.js --watch
```

참조

<https://github.com/bear-frog/ci-cd/blob/main/scripts/after-deploy.sh>



# 4. create ec2 instance

동일화를 위해 aws Ubuntu 22.04를 사용하세요

Name and tags [Info](#)

Name

sample

Add additional tags

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Q Search our full catalog including 1000s of application and OS images

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

S

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-00eecd4036573771 (64-bit (x86)) / ami-07ef7933c011a38e1 (64-bit (Arm))

Virtualization: hvm   ENA enabled: true   Root device type: ebs

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-02-08

Architecture

AMI ID

64-bit (x86)

ami-00eecd4036573771

Verified provider

# 4. setup ec2 instance: install node

1. install latest node.js in Ubuntu

참조: <https://github.com/nodesource/distributions>

## Installation instructions

Node.js v19.x:

Using Ubuntu

```
curl -fsSL https://deb.nodesource.com/setup_19.x | sudo -E bash - &&\  
sudo apt-get install -y nodejs
```

```
ubuntu@ip-172-31-10-195:~$ node -v  
v19.7.0
```

# 4. setup ec2 instance: install codedeploy

## 1. 우분투에 codedeploy를 설치합니다

```
$ sudo apt update
$ sudo apt install ruby-full
$ sudo apt install wget
$ cd /home/Ubuntu
```

# 복붙이 아니고!! 사용중인 ec2 region을 확인하고 bucket-name과 region-identifier를 작성

# e.g us-east-2 > wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install

\$ wget https://bucket-name.s3.region-identifier.amazonaws.com/latest/install

\$ chmod +x ./install

\$ sudo ./install auto > /tmp/logfile

# 실행중인지 확인

\$ sudo service codedeploy-agent status

```
● codedeploy-agent.service - LSB: AWS CodeDeploy Host Agent
   loaded: loaded (/etc/init.d/codedeploy-agent; generated)
   Active: active (running) since Wed 2023-03-15 09:15:43 UTC; 8s ago
```

# CodeDeploy 에이전트가 설치되어 실행 중이면 다음과 같은 메시지가 표시되어야 합니다.

# The AWS CodeDeploy agent is running.

# error: No AWS CodeDeploy agent running과 같은 메시지가 표시되면 서비스를 시작하고 다음 두 명령을 한 번에 하나씩 실행

\$ sudo service codedeploy-agent start

\$ sudo service codedeploy-agent status

참조:

[https://docs.aws.amazon.com/ko\\_kr/codedeploy/latest/userguide/codedeploy-agent-operations-install-ubuntu.html](https://docs.aws.amazon.com/ko_kr/codedeploy/latest/userguide/codedeploy-agent-operations-install-ubuntu.html)

## 4. setup ec2 instance: setup app

1. ec2 Ubuntu 의 홈 디렉터리에 app 폴더를 생성하세요
  - 해당 디렉터리에 코드가 배포돼요

```
ubuntu@ip-172-31-0-255:~$ mkdir /home/ubuntu/app
ubuntu@ip-172-31-0-255:~$ cd /home/ubuntu/
ubuntu@ip-172-31-0-255:~$ ls
app
```

2. sudo npm install pm2 -g 명령어를 통해 pm2를 설치하세요
3. 80번 포트에서 node 앱이 실행되기 위한 설정을 하세요

# 참조: <https://unchae.tistory.com/entry/PM2-80-443%E8%AC%ED%8A%B8-%EC%82%AC%EC%9A%A9>

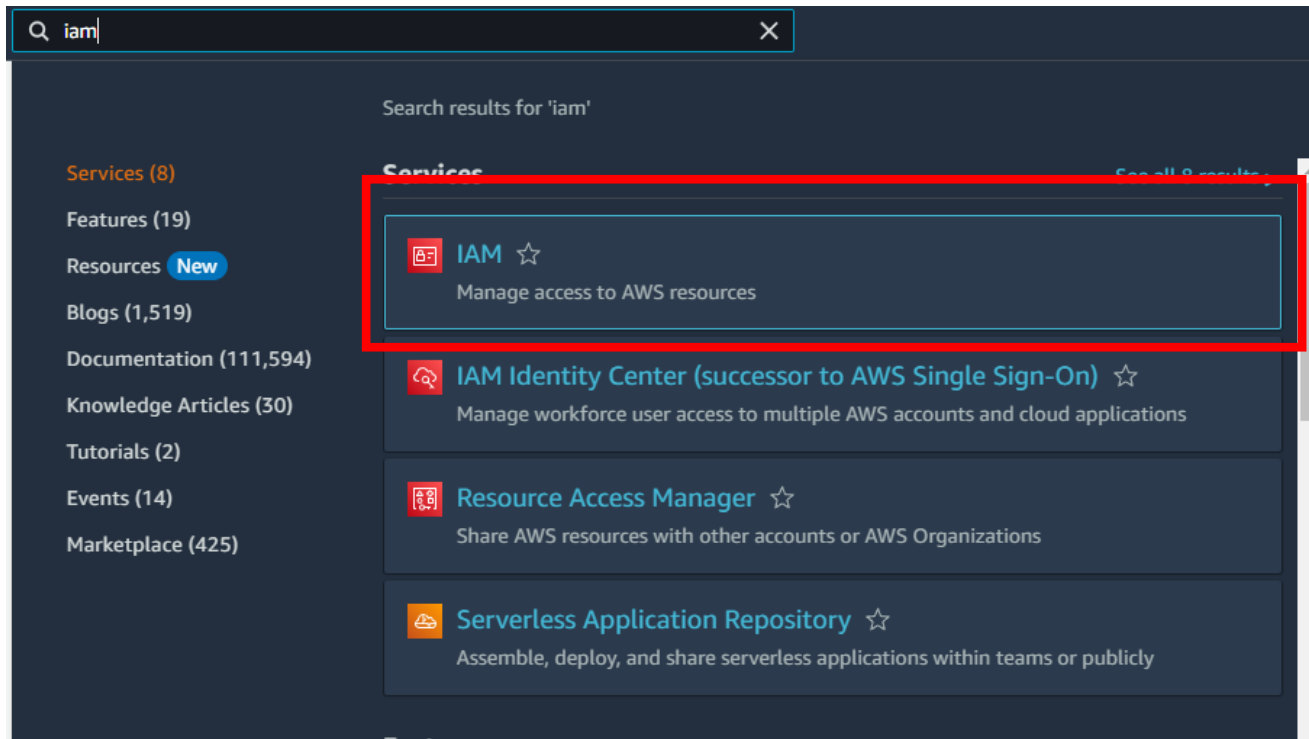
```
$ sudo apt-get install authbind
$ sudo touch /etc/authbind/byport/80
$ sudo chown ubuntu /etc/authbind/byport/80
$ sudo chmod 755 /etc/authbind/byport/80
$ sudo touch /etc/authbind/byport/443
$ sudo chown ubuntu /etc/authbind/byport/443
$ sudo chmod 755 /etc/authbind/byport/443
```

실행:

```
$ authbind --deep pm2 start index.js
```

# 5. create IAM role

ec2가 s3와 codedeploy를 이용할 수 있도록 권한 설정



# 5. create IAM role: create ec2 role

Identity and Access Management (IAM) ×

Search IAM

Dashboard

▼ Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

▼ Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Organization activity

Service control policies (SCPs)

Related consoles

IAM Identity Center ↗ New

AWS Organizations ↗

↻

Delete

Create role

<

1

2

>

⚙

Last activity ▼

Select trusted entity Info

Trusted entity type

☒ AWS service  
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ AWS account  
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ Web identity  
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ SAML 2.0 federation  
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

☐ Custom trust policy  
Create a custom trust policy to enable others to perform actions in this account.

Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Common use cases

☒ EC2  
Allows EC2 instances to call AWS services on your behalf.

☐ Lambda  
Allows Lambda functions to call AWS services on your behalf.

Use cases for other AWS services:

Choose a service to view use case ▼

Cancel

Next

# 5. create IAM role: create ec2 role

1. AWSCodeDeployFullAccess
2. AmazonS3FullAccess

두 개의 권한 추가 / 생성

Add permissions [Info](#)

Permissions policies (Selected 2/831) [Info](#)

Choose one or more policies to attach to your new role.

1 match

[Properties](#) [FullAccess" X](#) [Clear filters](#)

Type	Path	Name	Type	Description
Used as		AmazonS3FullAccess	AWS m...	Provides full access to all buckets via the AWS Management Console.

► Set permissions boundary - optional [Info](#)

Set a permissions boundary to control the maximum permissions this role can have. This is not a common setting, but you can use it to delegate permission management to others.

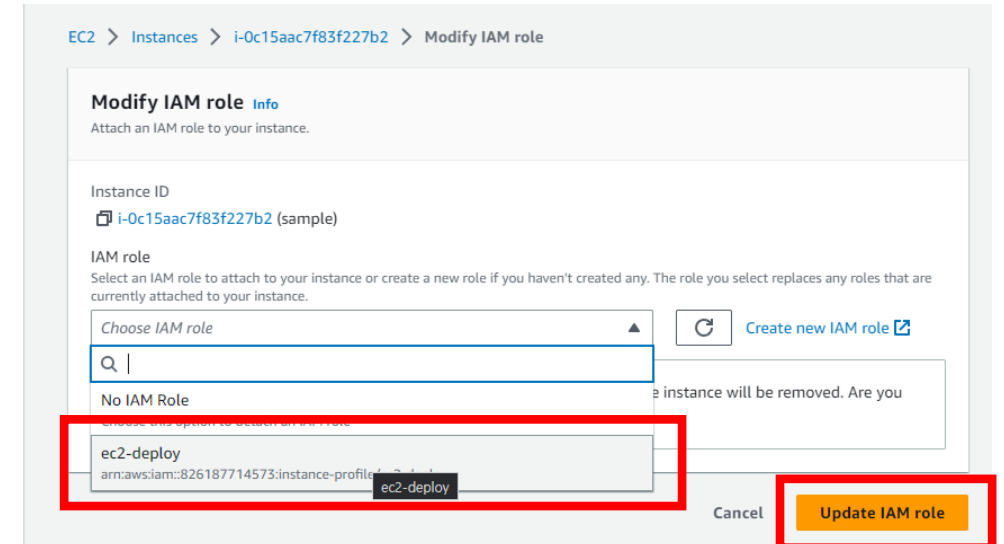
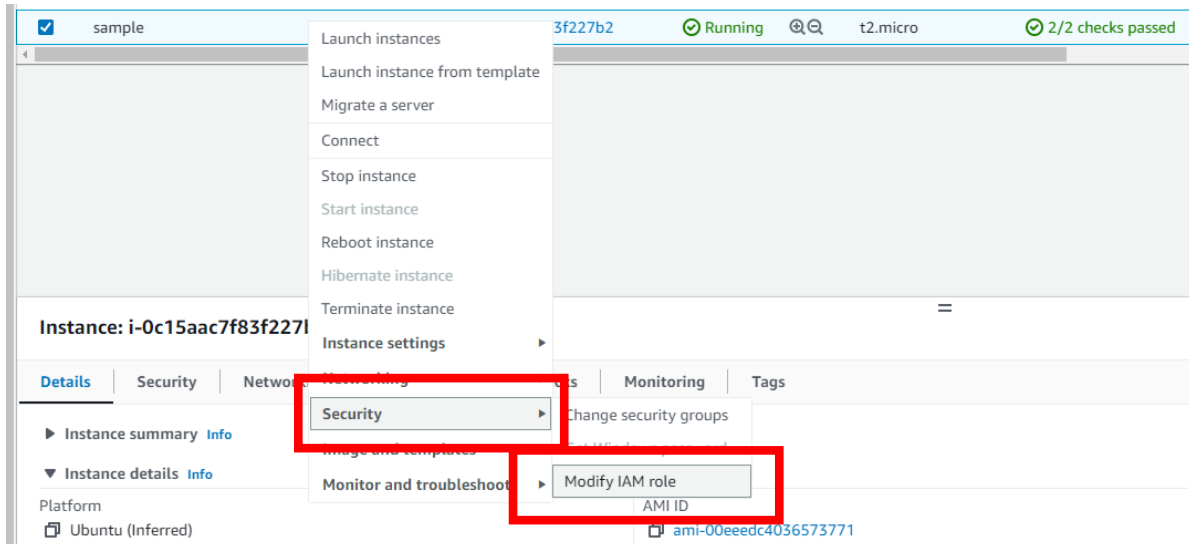
Cancel

Previous

Next

# 5. create IAM role: add ec2 role

생성한 ec2 instance에 IAM role 추가





# 5. create IAM role: create codedeploy role

codedeploy를 위한 권한을 추가합니다

Select trusted entity [Info](#)

## Trusted entity type

### ☒ AWS service

Allow AWS services like EC2, Lambda, or others to perform actions in this account.

### ☐ AWS account

Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

### ☐ Web identity

Allows users federated with an identity provider to perform actions in this account.

### ☐ SAML 2.0 federation

Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.

### ☐ Custom trust policy

Create a custom trust policy to enable others to perform actions in this account.

## Permissions policies (1) [Info](#)

The type of role that you selected requires the following policy.

Policy name ↗	Type	Attached entities
---------------	------	-------------------

AWSCodeDeployRole	AWS m...	0
-------------------	----------	---

권한 추가 / 생성

## Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

### Common use cases

#### ☐ EC2

Allows EC2 instances to call AWS services on your behalf.

#### ☐ Lambda

Allows Lambda functions to call AWS services on your behalf.

### Use cases for other AWS services:

CodeDeploy

#### ☒ CodeDeploy

Allows CodeDeploy to call AWS services such as Auto Scaling on your behalf.

#### ☐ CodeDeploy for Lambda

Allows CodeDeploy to route traffic to a new version of an AWS Lambda function version on your behalf.

#### ☐ CodeDeploy - ECS

Allows CodeDeploy to read S3 objects, invoke Lambda functions, publish to SNS topics, and update ECS services on your behalf.

# 6. create Codedeploy

The screenshot shows the AWS CodeDeploy console interface. On the left, a sidebar lists navigation options under 'Developer Tools': Source (CodeCommit), Artifacts (CodeArtifact), Build (CodeBuild), and Deploy (CodeDeploy). Under 'Deploy • CodeDeploy', the 'Getting started' link is highlighted with a red box. Below this are links for Deployments, Applications, Deployment configurations, and On-premises instances. Further down are Pipeline (CodePipeline) and Settings. At the bottom of the sidebar are search and feedback options.

The main content area has a dark header with 'Developer Tools' and 'AWS CodeDeploy' in large white text, followed by the subtitle 'Automate code deployments to maintain application uptime'. Below this is a paragraph describing AWS CodeDeploy as a fully managed service. To the right of this text is a white box with the heading 'Create AWS CodeDeploy deployment', a brief instruction, and an orange 'Create application' button highlighted with a red border.

Below the main text is a 'How it works' section with a video player showing an 'Introduction to AWS CodeDeploy - Automated Software Deployment' video. To the right of the video player is a 'Pricing (US)' section with a table:

Pricing (US)	
For CodeDeploy on EC2/Lambda	Free
For CodeDeploy On-Premises	\$0.02 per instance update

Below the pricing table is a 'Learn more' link with an external icon.

codedeploy를 생성합니다

# 6. create Codedeploy

Developer Tools > CodeDeploy > Applications > Create application

## Create application

### Application configuration

Application name  
Enter an application name

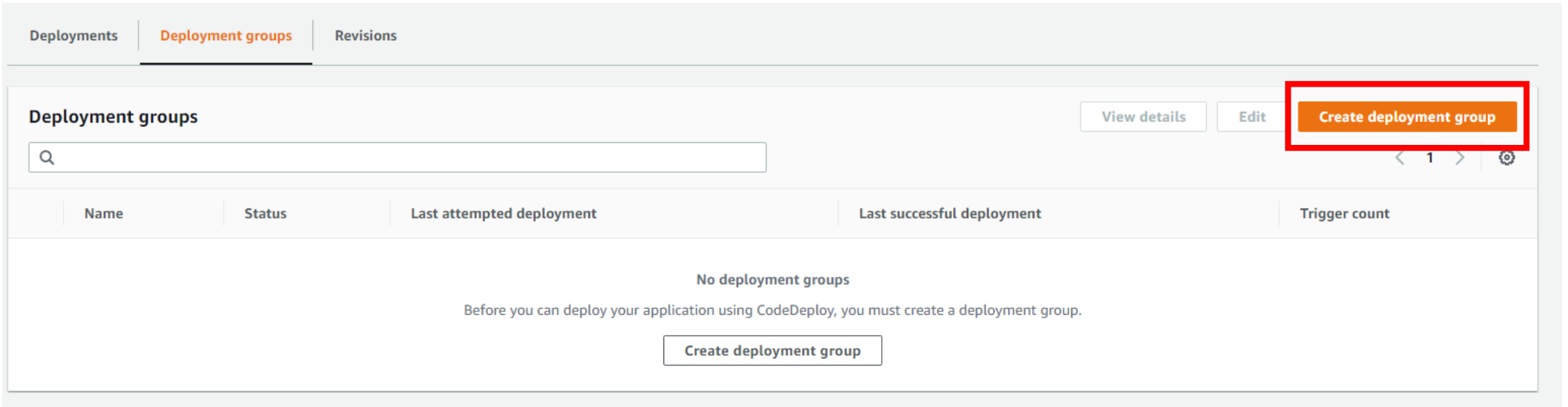
100 character limit

Compute platform  
Choose a compute platform

Cancel Create application

ec2상에 배포하기 때문에 ec2로 플랫폼을 선택하세요

## 6. create Codedeploy



The screenshot shows the AWS CodeDeploy console interface. At the top, there are three tabs: 'Deployments', 'Deployment groups' (which is selected and highlighted in orange), and 'Revisions'. Below the tabs, the 'Deployment groups' section is visible. It includes a search bar with a magnifying glass icon. To the right of the search bar, there are three buttons: 'View details', 'Edit', and 'Create deployment group'. The 'Create deployment group' button is highlighted with a red rectangular box. Below the buttons, there is a table with the following headers: 'Name', 'Status', 'Last attempted deployment', 'Last successful deployment', and 'Trigger count'. The table is currently empty, and a message is displayed in the center: 'No deployment groups. Before you can deploy your application using CodeDeploy, you must create a deployment group.' Below this message is a button labeled 'Create deployment group'.

codedeploy의 배포 그룹을 추가합니다

# 6. create CodeDeploy

### Application

Application  
app  
Compute type  
EC2/On-premises

### Deployment group name

Enter a deployment group name

100 character limit

### Service role

Enter a service role

Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.

### Deployment type

Choose how to deploy your application

☒ In-place  
Updates the instances in the deployment group with the latest application revisions. During a deployment, each instance will be briefly taken offline for its update

☐ Blue/green  
Replaces the instances in the deployment group with new instances and deploys the latest application revision to them. After instances in the replacement environment are registered with a load balancer, instances from the original environment are deregistered and can be terminated.

1. 생성한 codedeploy IAM role을 추가합니다

### Environment configuration

Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances, and on-premises instances to add to this deployment

☐ Amazon EC2 Auto Scaling groups

☒ Amazon EC2 instances  
2 unique matched instances. [Click here for details](#)

You can add up to three groups of tags for EC2 instances to this deployment group.  
**One tag group:** Any instance identified by the tag group will be deployed to.  
**Multiple tag groups:** Only instances identified by all the tag groups will be deployed to.

Tag group 1

Key	Value - optional	
<input type="text" value="Name"/>	<input type="text" value="sample"/>	<input type="button" value="Remove tag"/>
<input type="button" value="Add tag"/>		
<input type="button" value="+ Add tag group"/>		

☐ On-premises instances

**Matching instances**  
2 unique matched instances. [Click here for details](#)

2. 생성한 ec2 instace와 연동합니다

## 6. create Codedeploy

### Load balancer

Select a load balancer to manage incoming traffic during the deployment process. The load balancer blocks traffic from each instance while it's being deployed to and allows traffic to it again after the deployment succeeds.

☐ Enable load balancing

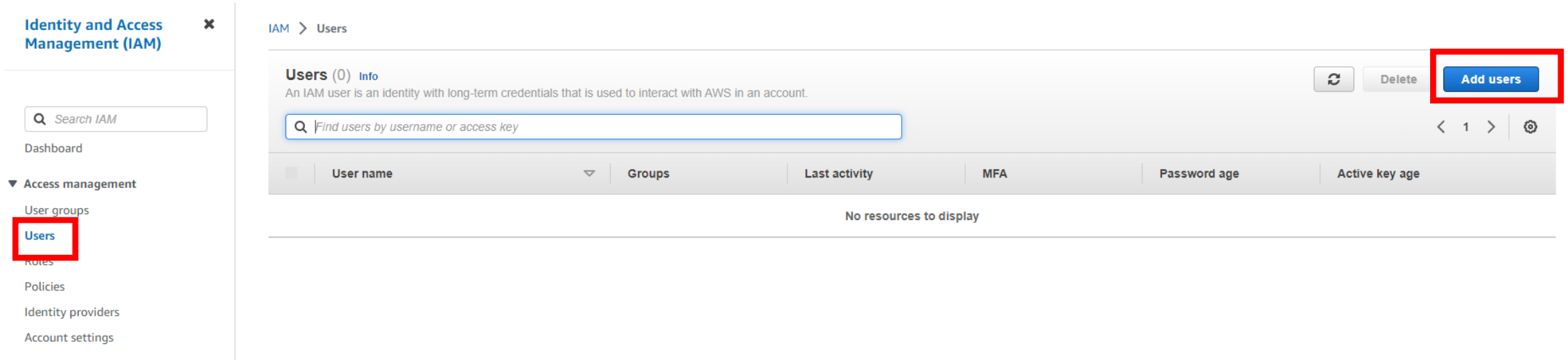
► Advanced - optional

Cancel

Create deployment group

로드밸런싱을 해제합니다

# 7. create IAM user



The screenshot displays the AWS IAM console interface. On the left sidebar, under 'Identity and Access Management (IAM)', the 'Users' link is highlighted with a red box. The main content area shows the 'Users (0)' page. At the top right of this area, the 'Add users' button is also highlighted with a red box. Below the button, there is a search bar and a table with columns: User name, Groups, Last activity, MFA, Password age, and Active key age. The table currently shows 'No resources to display'.

Identity and Access Management (IAM) x

Search IAM

Dashboard

▼ Access management

User groups

**Users**

Notes

Policies

Identity providers

Account settings

IAM > Users

**Users (0)** [Info](#)

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

Refresh Delete **Add users**

Find users by username or access key

< 1 > ⚙

	User name	Groups	Last activity	MFA	Password age	Active key age
No resources to display						

ec2 서버 상에서 사용할 IAM user를 생성합니다

# 7. create IAM user

## Set permissions

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

### Permissions options

☐ Add user to group  
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.

☐ Copy permissions  
Copy all group memberships, attached managed policies, and inline policies from an existing user.

☒ Attach policies directly  
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

### Permissions policies (2/1064)

Choose one or more policies to attach to your user.

1 match

AmazonS3FullAccess

Clear filters

<input checked="" type="checkbox"/>	Policy name	Type	Attached entities
<input checked="" type="checkbox"/>	AmazonS3FullAccess	AWS managed	1

1. AWSCodeDeployFullAccess

2. AmazonS3FullAccess

두 개의 권한 체크



# 7. create IAM user: create access key

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

User groups

**Users**

Policies

Identity providers

Account settings

IAM > Users > deployUser

deployUser

Summary

ARN  
arn:aws:iam::826187714573:user/deployUser

Console access  
Disabled

Created  
March 15, 2023, 20:15 (UTC+09:00)

Last console sign-in  
-

Permissions Groups Tags **Security credentials** Access Advisor

## Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

### Access key

### Secret access key

AKIA4AXFACAGWDMA6JHM

Vu9coMe4v1u2AykXfO982nb2eP5fZ/dlHOirmnZ7 [Hide](#)

창 나가면 다시는 못 보니까  
csv파일 다운 or 적어 두세요

Access keys (0)

Use access keys to send programmatic calls to AWS from the AWS CLI, AWS Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time. [Learn more](#)

Create access key

No access keys

As a best practice, avoid using long-term credentials like access keys. Instead, use tools which provide short term credentials. [Learn more](#)

Create access key

## Command Line Interface (CLI)

You plan to use this access key to enable the AWS CLI to access your AWS account.

# 7. create S3 bucket

## General configuration

Bucket name

Bucket name must be globally unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

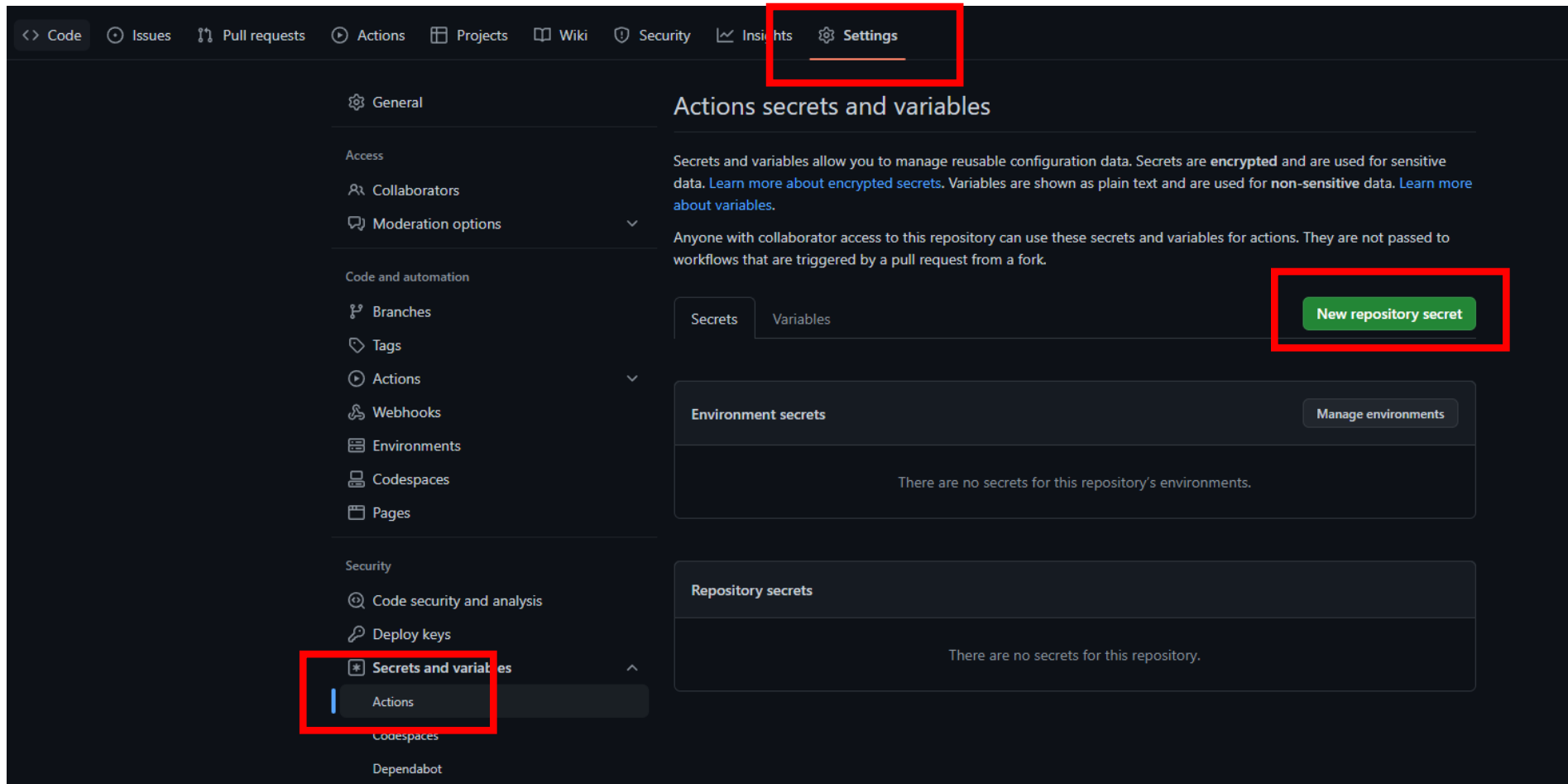
US East (Ohio) us-east-2 ▼

Copy settings from existing bucket - *optional*  
Only the bucket settings in the following configuration are copied.

Choose bucket

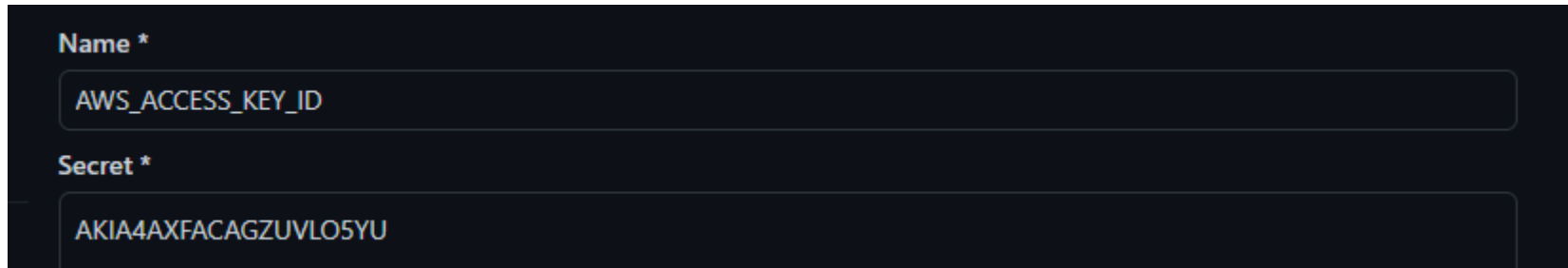
따로 설정할 것 없이 만드세요

## 4. create action workflow: add access key



깃허브 setting > action에서 사용할 비밀 키를 추가합니다

## 4. create action workflow: add access key

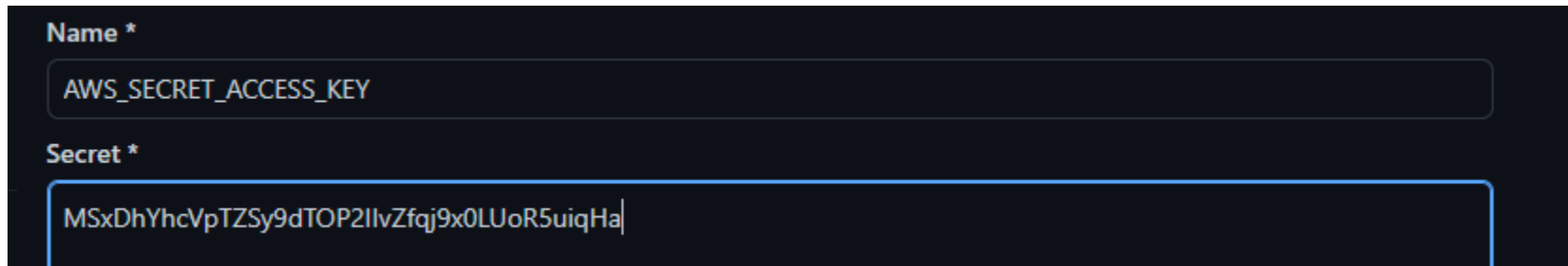


Name \*

AWS\_ACCESS\_KEY\_ID

Secret \*

AKIA4AXFACAGZUVLO5YU



Name \*

AWS\_SECRET\_ACCESS\_KEY

Secret \*

MSxDhYhcVpTZSy9dTOP2llvZfqj9x0LUoR5uiqHa

IAM user에서 생성한  
AWS\_ACCESS\_KEY\_ID  
AWS\_SECRET\_ACCESS\_KEY  
추가합니다  
github action script에서 사용됩니다

# 4. create action workflow: connect aws in ec2

ec2 상에 aws cli를 설치하고 만들어 둔 IAM user와 연동합니다

# 설치

\$ sudo apt update

\$ sudo apt install awscli

# 설치 확인

\$ aws help

# 사용자 설정

\$ aws configure

```
ubuntu@ip-172-31-10-195:~$ aws configure
AWS Access Key ID [None]: AKIA4AXFACAGZUVL05YU
AWS Secret Access Key [None]: MSxDhYhcVpTZSy9dT0P2IIvZfqj9x0LUoR5uiqHa
Default region name [None]: us-east-2
Default output format [None]:
```

AWS Access Key ID [None]: 액세스 키를 입력

AWS Secret Access Key [None]: 시크릿 액세스 키를 입력

Default region name [None]: us-east-2

# 혹시 리전이 다르면 해당 리전 기입

Default output format [None]: 그냥 Enter 입력


## 4. create action workflow: connect aws in ec2

```
# codedeploy 재시작을 통해 IAM role을 갱신합니다  
$ sudo service codedeploy-agent restart
```

```
# 이후 배포 시 에러 발생한다면  
# 아래 명령어로 서버에서 로그를 확인 해 보세요  
$ cat /var/log/aws/codedeploy-agent/codedeploy-agent.log
```

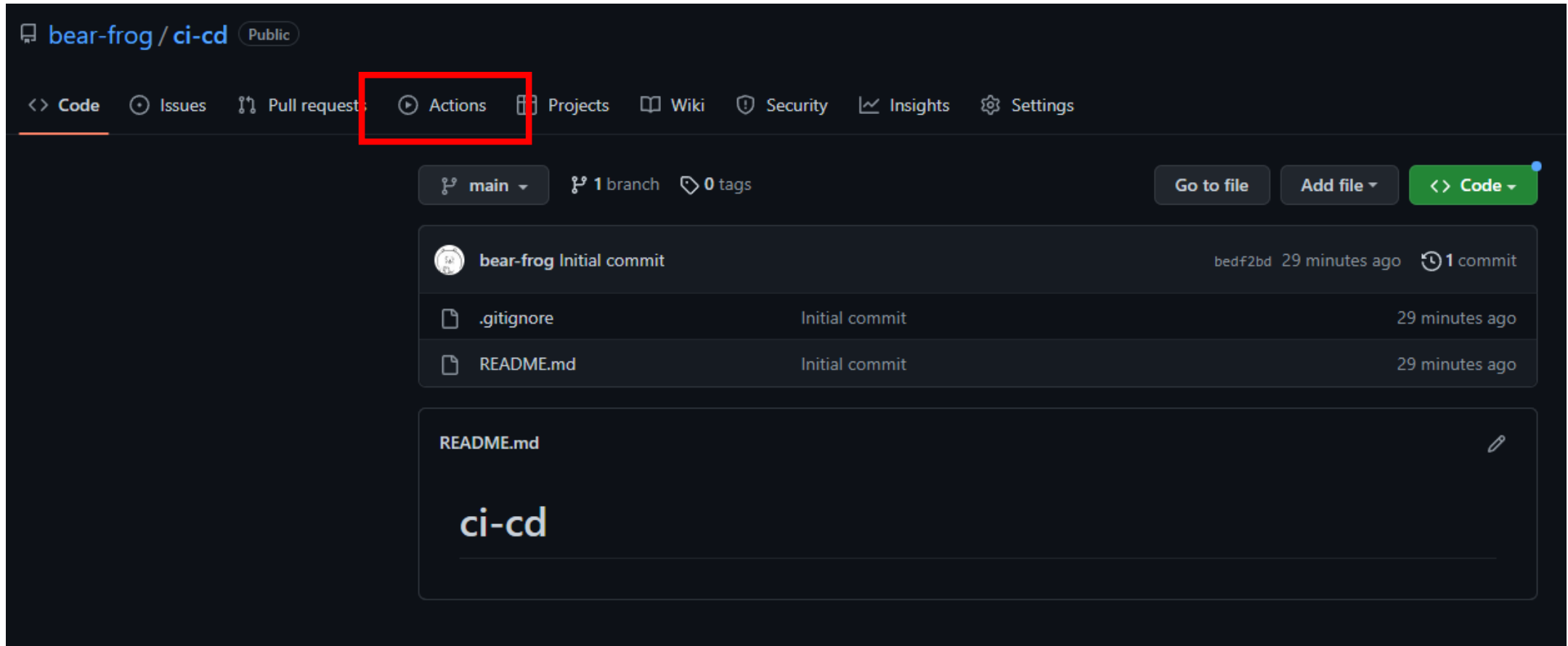
# 4. create action workflow: push file

cicd 스크립트를 작성하기 이전 로컬의 sample SSR app을 레포지토리에 push 해 주세요



bear-frog INIT		e06a86b 2 minutes ago 1 commit
public	INIT	2 minutes ago
scripts	INIT	2 minutes ago
src-ssr	INIT	2 minutes ago
src	INIT	2 minutes ago
.editorconfig	INIT	2 minutes ago
.eslintignore	INIT	2 minutes ago
.eslintrc.js	INIT	2 minutes ago
.gitignore	INIT	2 minutes ago
.npmrc	INIT	2 minutes ago
.prettierrc	INIT	2 minutes ago
appspec.yml	INIT	2 minutes ago
index.html	INIT	2 minutes ago
package-lock.json	INIT	2 minutes ago
package.json	INIT	2 minutes ago
postcss.config.js	INIT	2 minutes ago
quasar.config.js	INIT	2 minutes ago
tsconfig.json	INIT	2 minutes ago

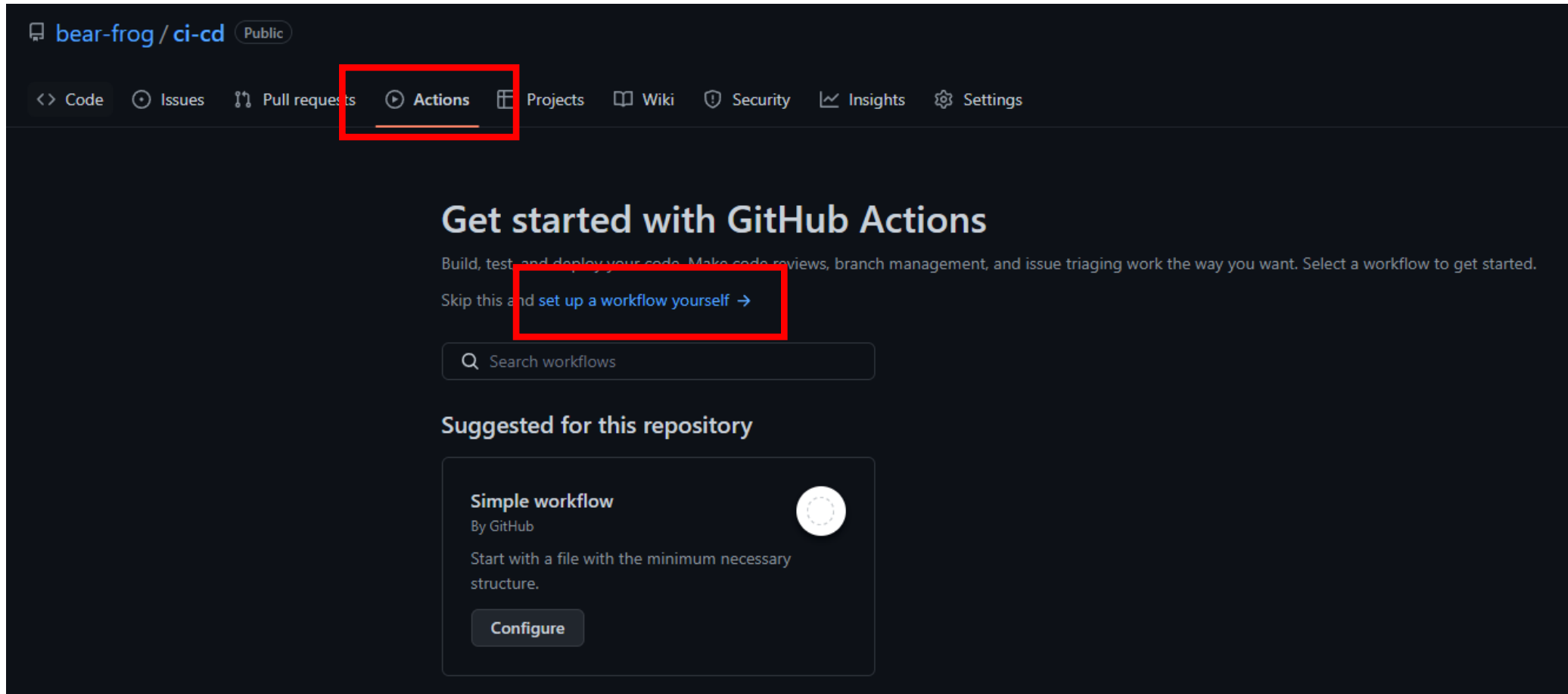
# 4. create action workflow



생성한 레포지토리 -> Actions



# 4. create action workflow



set up a workflow yourself

## 4. create action workflow: add script

- 참조
- <https://github.com/bear-frog/ci-cd/blob/main/.github/workflows/main.yml>
- 스크립트가 길어서 ppt상에 작성하지 않았어요
- 코드 보고 확인하세요

## 4. create action workflow: add script

```
# This workflow will do a clean install of node dependencies, cache/restore them, build the source code and run tests across different versions of node
# For more information see: https://help.github.com/actions/language-and-framework-guides/using-nodejs-with-github-actions

name: main
```

- name: yml file 이름

## 4. create action workflow: add script

```
on:  
  push:  
    branches: [main]  
  pull_request:  
    branches: [main]
```

- on: 이벤트에 반응
  - main 브랜치에 대한 push, pull\_request 이벤트에 반응하여
  - github action 스크립트 실행

## 4. create action workflow: add script

```
jobs:
```

- jobs: github action에서 수행할 작업을 작성

```
build:  
  runs-on: ubuntu-22.04  
  
  strategy:  
    matrix:  
      node-version: [19.x]  
      # See supported Node.js release schedule at https://nodejs.org/en/about/releases/
```

- build: 빌드 환경 작성
  - runs-on: 운영체제 기입
  - strategy: 개발 환경 기입
    - node-version: 사용 노드 버전 기입

## 4. create action workflow: add script

```
steps:
  - name: Checkout source code.
    uses: actions/checkout@v2

  - name: Use Node.js ${{ matrix.node-version }}
    uses: actions/setup-node@v2
    with:
      node-version: ${{ matrix.node-version }}
```

- steps: 빌드 과정 서술
  - names: 명령어 이름 해당 이름을 통해 명령어 실행됨

## 4. create action workflow: add script

```
- name: build files
  working-directory: ./
  run: |
    npm i
    npm run build
```

- app build
  - 현재 디렉터리에서 npm i(종속성 설치)
  - npm run build(앱 빌드)
  - 두 명령어를 통해 빌드된 앱(dist) 폴더 생성

```
- name: zip distributions
  run: zip -r cicd-app.zip ./dist ./appspec.yml ./scripts
```

- S3 버킷 전송을 위한 zip파일 생성
  - 빌드된 폴더(./dist) codedeploy script(./appspec.yml, ./scripts)를 압축(cicd-app.zip)

## 4. create action workflow: add script

```
- name: upload to S3
run: aws s3 cp --region us-east-2 ./cicd-app.zip s3://uds-cicd/public/
```

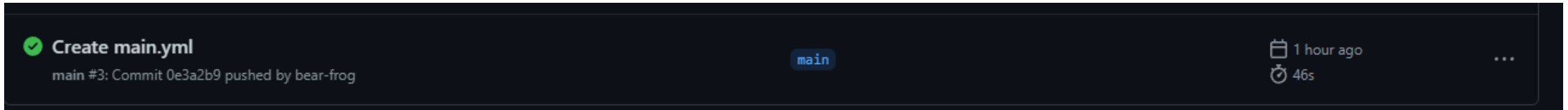
- S3버킷에 zip파일 업로드
  - 압축된 파일(cicd-app.zip)을 S3버킷(uds-cicd)의 public폴더로 이동

```
- name: deploy with AWS codeDeploy
run: aws deploy create-deployment
    --application-name app
    --deployment-config-name CodeDeployDefault.OneAtATime
    --deployment-group-name group
    --s3-location bucket=uds-cicd,bundleType=zip,key=public/cicd-app.zip
```

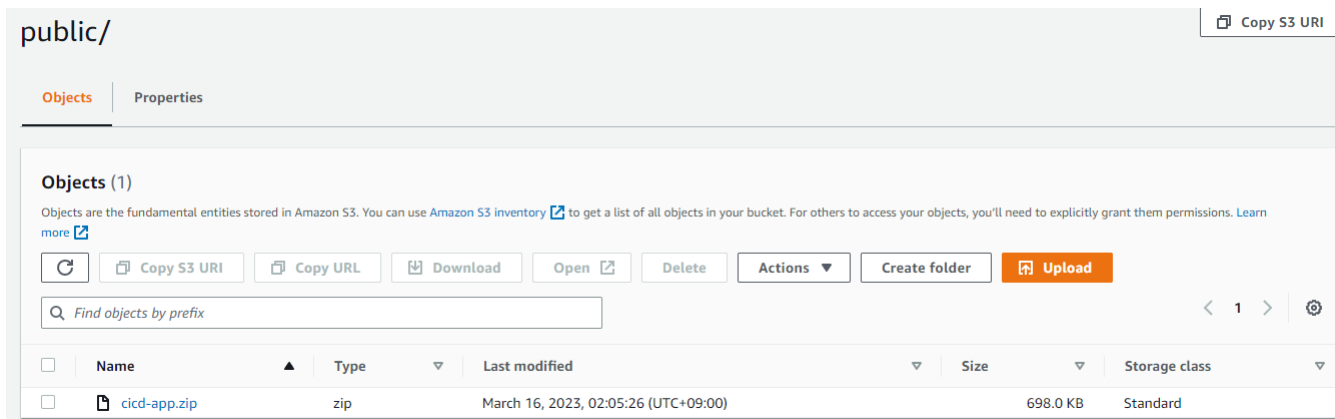
- codedeploy를 통한 서버 배포
  - application name(app)
  - group name(group)을 작성
  - S3 버킷의 배포 파일을 명시(public/cicd-app.zip)



# 4. CI/CD



- 빌드 과정 완료



- S3버킷에 app이 업로드

## 4. CI/CD

○	d- PWYZE840M	✔ Succeeded	In-place	EC2/On- premises	app	group	s3://uds-ci...	User action	Mar 16, 2023 2:05 AM (UTC+9:00)	Mar 16, 2023 2:05 AM (UTC+9:00)
---	-----------------	-------------	----------	---------------------	-----	-------	----------------	-------------	---------------------------------------	---------------------------------------

- 배포 과정 완료

```
ubuntu@ip-172-31-0-255:~/app$ ls
appspec.yml  dist  scripts
```

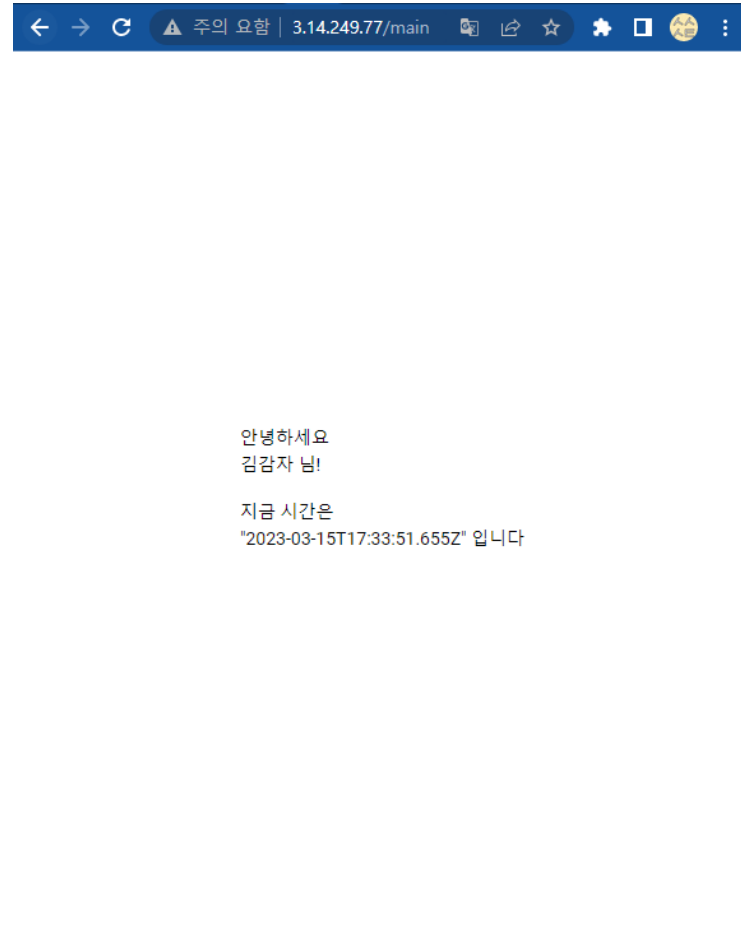
- ec2 서버에 업로드 완료

```
ubuntu@ip-172-31-0-255:~/app/dist/ssr$ pm2 status
```

id	name	namespace	version	mode	pid	uptime	↻	status	cpu	mem	user	watching
0	index	default	0.0.1	fork	7767	72s	0	online	0%	63.0mb	ubuntu	enabled

- after-deploy.sh에 따라 배포 후 앱 실행

## 4. CI/CD





수고  
많았어  
넌 ^^