



포팅메뉴얼

1. 프로젝트 기술 스택
 - A. Back-end
 - B. Front-end
 - C. AI
2. 빌드 방법
 - A. 백엔드 빌드 방법
 - B. 프론트엔드 빌드 방법
 - C. AI 서버 실행 방법
 - D. 배포 명령어 정리
3. DB 계정
 - A. MySQL WorkBench 추가하기
 - B. EC2 계정정보 넣기
4. 프로퍼트 정의
 - A. NginX Default 값 세팅
 - 1) EC2 에서 세팅 파일로 접근
 - 2) 세팅값 다음과 같이 변경하기
 - 3) default 파일
 - B. Git ignore 파일
 - 1) app.yaml 파일
5. EC2 설정
 - A. AWS EC2 DB 세팅
 - B. 에러 해결
6. Jenkins 설정
 - A. Jenkins 구성
 - B. GitLab 설정
 - C. Jenkins 빌드, 배포 명령어
7. 외부 서비스
 - A. AWS S3

1. 프로젝트 기술 스택

A. Back-end

기술 스택 (버전) : Spring boot 2.7.3, MySQL 8.0.30, Nginx 1.18.0, Jenkins 2.361.1, AWS EC2, AWS S3

사용 툴 : IntelliJ 2021.2.4, MobaXterm 22.0, MySQL Workbench 8.0.20, JDK 11.0.15.1

B. Front-end

기술 스택 (버전) : React 18.2.0, stomp 6.1.2 (node-sass 7.0.3, mui 5.10.5)

사용 툴 : VSCode 1.71.2, Chrome

C. AI

기술 스택 (버전) : python 3.10.4, YOLOv7 v0.1, openCV 4.6.0.66, FastAPI 0.85.0, labelling v1.8.1

사용 툴 : pycharm 2022.2, JupyterHub SSIFY GPU Suver, Anaconda

2. 빌드 방법

A. 백엔드 빌드 방법

1. Command Shell을 통해 프로젝트 폴더 안의 BE\findit 폴더 안으로 이동한다.

2. `./gradlew clean build` 명령어를 통해 빌드한다.
3. 프로젝트 폴더 안의 `BE\findit\build\libs` 안에 build 파일이 생성된다.

```

관리자: Windows PowerShell

PS C:\pjt\특화PJT\SO7P22A203\BE> cd findit
PS C:\pjt\특화PJT\SO7P22A203\BE\findit> ./gradlew clean build

> Task :compileJava
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

> Task :test
2022-10-02 19:55:05.152 INFO 6716 --- [ionShutdownHook] o.s.m.s.b.SimpleBrokerMessageHandler : Stopping...
2022-10-02 19:55:05.152 INFO 6716 --- [ionShutdownHook] o.s.m.s.b.SimpleBrokerMessageHandler : BrokerAvailabilityEvent[available=false, SimpleBrokerMessageHandler [org.springframework.messaging.simp.broker.DefaultSubscriptionRegistry@4d19ddeb]]
2022-10-02 19:55:05.153 INFO 6716 --- [ionShutdownHook] o.s.m.s.b.SimpleBrokerMessageHandler : Stopped.
2022-10-02 19:55:05.180 INFO 6716 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence unit 'default'
2022-10-02 19:55:05.183 INFO 6716 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown initiated...
2022-10-02 19:55:05.240 INFO 6716 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Shutdown completed.

BUILD SUCCESSFUL in 26s
8 actionable tasks: 8 executed
PS C:\pjt\특화PJT\SO7P22A203\BE\findit> cd build/libs
PS C:\pjt\특화PJT\SO7P22A203\BE\findit\build\libs> ls

디렉터리: C:\pjt\특화PJT\SO7P22A203\BE\findit\build\libs

Mode                LastWriteTime         Length Name
----                -
-a----          2022-10-02 오후 7:54          112724 findit-0.0.1-SNAPSHOT-plain.jar
-a----          2022-10-02 오후 7:54          67460807 findit-0.0.1-SNAPSHOT.jar

PS C:\pjt\특화PJT\SO7P22A203\BE\findit\build\libs>

```

주의! 빌드 하기 전에, `aws.yml` 파일이 프로젝트 폴더 안의 `BE\findit\src\main\resources` 폴더 안에 존재해야 한다.

B. 프론트엔드 빌드 방법

1. Node.js 환경에서 `FE\findit` 디렉토리로 이동
2. `npm i --force` 를 통해 `package-lock.json`에 정의된 패키지를 다운로드
3. 해당 폴더에서 아래의 명령어를 입력하여 배포 버전 파일 생성

```
npm run build
```

4. `findit` 디렉토리에 `build` 폴더가 생성됨
5. 생성된 `build` 폴더를 서버에 배포하여 사용

C. AI 서버 실행 방법

1. 서버 내에 `AI/fast` 폴더 자체를 배포하여 사용
2. `fast` 폴더 안으로 이동.
3. `pip install requirements.txt`로 필요 라이브러리를 설치한다.
4. 아래의 명령어를 입력하여 서버 실행

```
uvicorn main:app --reload
```

D. 배포 명령어 정리

1. 현재 실행 중인 서버 pid 확인

```
ps -ef | grep java
```

현재 실행 중인 서버의 pid를 확인한다.

2. 실행 중인 서버 종료

```
sudo kill -9 <pid>
```

만약 실행 중인 서버가 존재한다면, kill 명령어를 통해 종료한다.

3. 새로운 서버 백그라운드에서 실행

```
nohup java -jar findit-0.0.1-SNAPSHOT.jar
```

BE 빌드 과정에서 생성된 빌드 파일의 경로로 이동해서 서버를 실행시킨다.

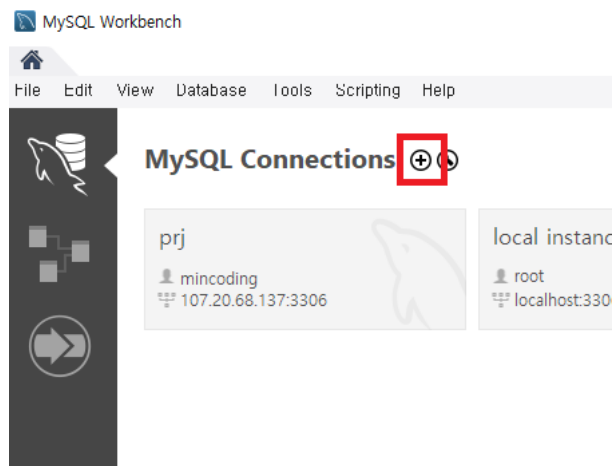
4. Nginx 재시작

```
sudo systemctl restart nginx
```

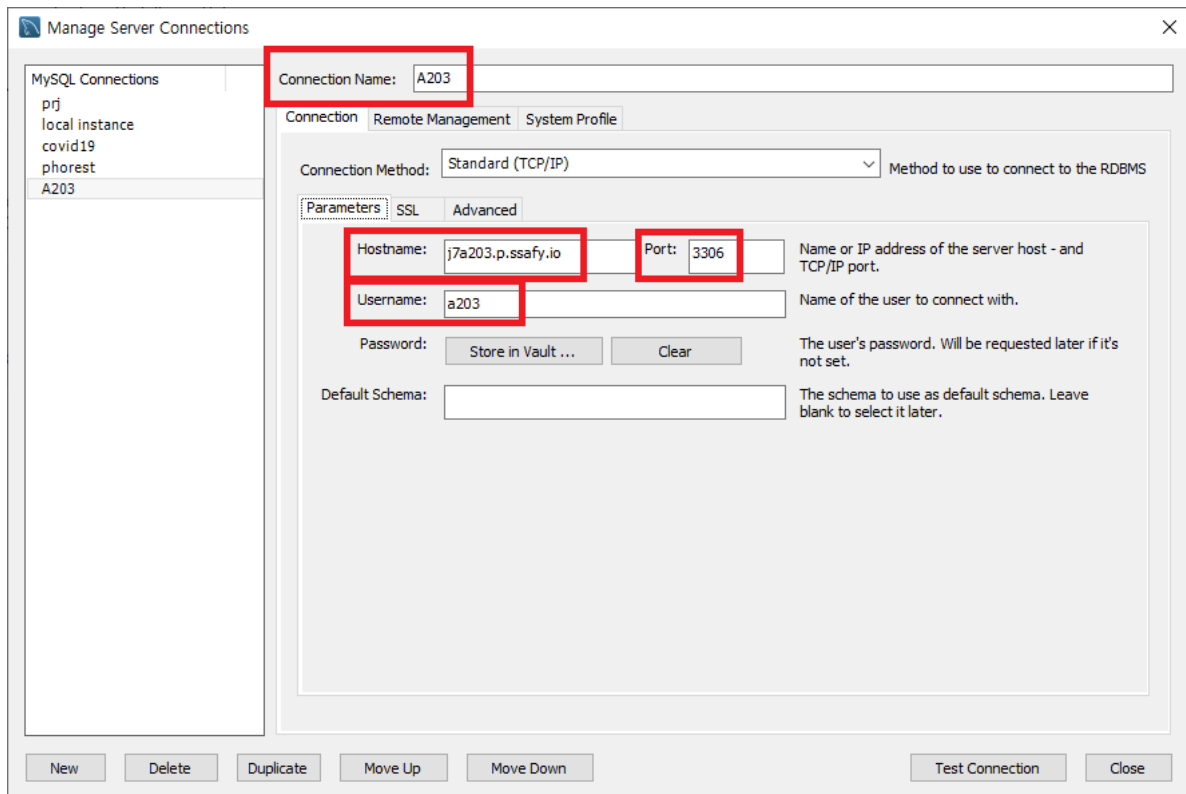
Nginx를 재시작한다.

3. DB 계정

A. MySQL WorkBench 추가하기



B. EC2 계정정보 넣기



- username : a203 , password : ?p27IQiwc1941Ykol
기존 root 계정이 아닌 별도의 a203 계정을 만들어서 진행했습니다.

4. 프로퍼트 정의

A. NginX Default 값 세팅

1) EC2 에서 세팅 파일로 접근

```
sudo apt get update
```

```
sudo vim /etc/nginx/sites-available/default
```

2) 세팅값 다음과 같이 변경하기

```
server {
    root /var/lib/jenkins/workspace/build;

    index index.html;

    server_name j7a203.p.ssafy.io findit.life;

    location / {
        try_files $uri $uri/ /index.html;
    }

    location /api/v1 {

        proxy_set_header Host $host;

        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header X-NginX-Proxy true;

        proxy_redirect off;
        charset utf-8;
    }
}
```

```

    proxy_pass http://localhost:8399/api/v1;

}

location /api/v1/ws {

    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
    proxy_set_header Host $host;

    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_set_header X-NginX-Proxy true;

    proxy_pass http://localhost:8399/api/v1/ws;

}

location /fast/ {
    proxy_pass http://localhost:8000/;
}

listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/findit.life/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/findit.life/privkey.pem; # managed by Certbot
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

server {
    if ($host = findit.life) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = j7a203.p.ssafy.io) {
        return 301 https://$host$request_uri;
    }
    return 404; # managed by Certbot

    server_name findit.life j7a203.p.ssafy.io;

    listen 80 ;
    listen [::]:80 ;

}

```

3) default 파일

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/a0c293c4-c1bb-4f95-b037-80dc28b6052d/default.txt>

B. Git ignore 파일

1) app.yaml 파일

```

spring:
  jpa:
    generate-ddl: true
    properties:
      hibernate:
        show_sql: true
        ddl-auto : none
        format_sql: true
    hibernate:
      ddl-auto: update
    database-platform: org.hibernate.dialect.MySQL5InnoDBDialect

devtools:
  liveload:
    enabled: true
  restart:
    enabled: false

freemarker:
  cache: false

```

```

datasource:
  password: ?p27IqIwci941Ykol
  driver-class-name: com.mysql.cj.jdbc.Driver
  username: a203
  url: jdbc:mysql://j7a203.p.ssafy.io:3306/FindIt

server:
  servlet:
    contextPath: /api/v1
  port: 8399

password:
  reset:
    expire-time: 300000

jwt:
  secret-key: sasdKAnzkaAlfFimmAzsQkgYodiuqlxw
  access-token-expire-time: 1800000
  refresh-token-expire-time: 1209600000

frontEnd: http://localhost:3000

id_min_length: 4

##Aws S3
cloud:
  aws:
    credentials:
      access-key: AKIA3TJ4SYU6L4TIPYPB
      secret-key: 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
    s3:
      bucket: a203findit
      region:
        static: ap-northeast-2
      stack:
        auto: false

```

- aws.yaml은 프로젝트 폴더의 BE\findit\src\main\resources 폴더 안에 위치해야 한다.

5. EC2 설정

A. AWS EC2 DB 세팅

1. 세팅을 위한 최신 상태 업데이트

```
sudo apt-get update
```

2. MySQL 설치

```
sudo apt-get install mysql-server
```

3. 추가 세팅을 위한 이동 후 편집

```
cd /etc/mysql/mysql.conf.d
```

```
sudo vi mysqld.cnf
```

4. 바뀔 내용

```
bind-address = 0.0.0.0
```

5. 세팅 값 적용을 위한 재시작

```
sudo service mysql restart
```

6. root 계정 외에 사용할 계정 생성

```
sudo mysql -u root -p
```

```
CREATE USER 'admin'@'%' IDENTIFIED BY 'new password';
```

```
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;
```

```
FLUSH PRIVILEGES;
```

7. 확인

```
sudo mysql -u admin -p
```

B. 에러 해결

1. 에러 : `/usr/bin/xauth: file /root/.Xauthority does not exist`
해결 : <https://positivemh.tistory.com/534>
2. `sudo apt-get update`


6. Jenkins 설정


Jenkins를 이용해 CICD 환경을 구축, 개발 과정 중 약 300번의 빌드와 배포를 진행하였습니다.


A. Jenkins 구성


Dashboard > gitlab >


Configuration General


Enabled 


 General

 소스 코드 관리

 빌드 유발

 빌드 환경

 Build Steps

 빌드 후 조치

설명

[Plain text] [미리보기](#)

☒ GitHub project

Project url ?

☐ 사용자 빌드 경로 사용 ?

GitLab Connection

☐ Use alternative credential

- Gitlab의 프로젝트를 사용하므로, Github project를 누르고, Project url에 현재 개발하는 git lab repository 주소를 입력합니다.

Configuration

General

소스 코드 관리

빌드 유발

빌드 환경

Build Steps

빌드 후 조치

소스 코드 관리

☐ None☒ Git ?

Repositories ?

Repository URL ?

`https://lab.ssafy.com/s07-ai-image-sub2/S07P22A203.git`

Credentials ?

`asdfmelody@naver.com/*****`

+ Add

고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

`*/develop`

Add Branch

- 소스 코드 관리에서, Git을 선택하고 Repository에는 현재 개발하는 프로젝트의 Repository 주소를 입력합니다. Credentials에는 add를 통해 Gitlab에서 사용하는 아이디 비밀번호를 입력 한 후, 선택해줍니다. Branch Specifier에는 변화를 감지할 branch를 선택 하는 곳입니다. 저희는 develop branch를 선택했습니다.

Configuration

빌드 유발

General

소스 코드 관리

빌드 유발

빌드 환경

Build Steps

빌드 후 조치

☐ 빌드를 원격으로 유발 (예: 스크립트 사용) ?☐ Build after other projects are built ?☐ Build periodically ?☒ Build when a change is pushed to GitLab. GitLab webhook URL: <http://j7a203.p.ssafy.io:9090/project/gitlab> ?

Enabled GitLab triggers

☒ Push Events☐ Push Events in case of branch delete☒ Opened Merge Request Events☐ Build only if new commits were pushed to Merge Request ?☐ Accepted Merge Request Events☐ Closed Merge Request Events

Rebuild open Merge Requests

Never

☒ Approved Merge Requests (EE-only)☒ Comments

Comment (regex) for triggering a build ?

Jenkins please retry a build

- 빌드 유발에서, webhook을 통해 빌드를 유발하기 위해 Build when a change is pushed to GitLab 부분을 체크해주었고, Push Events와 Merge Request가 발생했을 때 빌드를 유발하였습니다.

Secret token ?

e9594c60d387ec9bd025a6346c17403e

Generate

- 빌드 유발의 고급 탭을 눌러서 나오는 Secret token을 Generate 한 후, 이후 GitLab webhook 설정에 사용하였습니다.

Build Steps

≡ Execute shell ?

Command

See [the list of available environment variables](#)

```
cd /var/lib/jenkins/workspace/gitlab/FE/findit
npm cache clean --force
npm i --legacy-peer-deps
npm install styled-components --force
npm install react-camera-pro --force
CI=false npm run build
rm -r /var/lib/jenkins/workspace/build
cp -rpf /var/lib/jenkins/workspace/gitlab/FE/findit/build /var/lib/jenkins/worksp

cd /var/lib/jenkins/workspace/gitlab/BE/findit
chmod +x gradlew
cp /var/lib/jenkins/workspace/.keys/app.yaml src/main/resources/app.yaml
./gradlew clean build
cd build/libs
cp -pf /var/lib/jenkins/workspace/gitlab/BE/findit/build/libs/findit-0.0.1-SNAPSHOT.jar /var/lib/jenkins/workspace/build/libs
cd /var/lib/jenkins/workspace

# cp -rpf /var/lib/jenkins/workspace/gitlab/AI/fastAPI /var/lib/jenkins/workspace/build
# cd /var/lib/jenkins/workspace/fastAPI
# python3 -m uvicorn main:app --reload

rm -rf /var/lib/jenkins/workspace/gitlab
```

고급...

- Build 탭에서 Execute shell을 선택하고, 직접 리눅스 명령어를 실행시켜 빌드와 배포를 수행하였습니다.

B. GitLab 설정

Search page

Webhooks

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

URL

URL must be percent-encoded if it contains one or more special characters.

Secret token

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

Trigger

☒ Push events

Push to the repository.

Project Hooks (1)

Push Events SSL Verification: enabled

- GitLab repository의 설정의 Webhook 탭에서 URL과 jenkins에서 얻은 webhook을 위한 Secret token을 입력하고, Push event가 발생했을 때 web hook이 되도록 설정하였습니다.

C. Jenkins 빌드, 배포 명령어

```
cd /var/lib/jenkins/workspace/gitlab/FE/findit
npm cache clean --force
npm i --legacy-peer-deps
npm install styled-components --force
npm install react-camera-pro --force
CI=false npm run build
rm -r /var/lib/jenkins/workspace/build
cp -rpf /var/lib/jenkins/workspace/gitlab/FE/findit/build /var/lib/jenkins/workspace/build

cd /var/lib/jenkins/workspace/gitlab/BE/findit
chmod +x gradlew
cp /var/lib/jenkins/workspace/.keys/app.yaml src/main/resources/app.yaml
./gradlew clean build
cd build/libs
cp -pf /var/lib/jenkins/workspace/gitlab/BE/findit/build/libs/findit-0.0.1-SNAPSHOT.jar /var/lib/jenkins/workspace/findit-0.0.1-SNAPSHOT
cd /var/lib/jenkins/workspace

cp -rpf /var/lib/jenkins/workspace/gitlab/AI/fastAPI /var/lib/jenkins/workspace/fastAPI
cd /var/lib/jenkins/workspace/fastAPI
python3 -m uvicorn main:app --reload

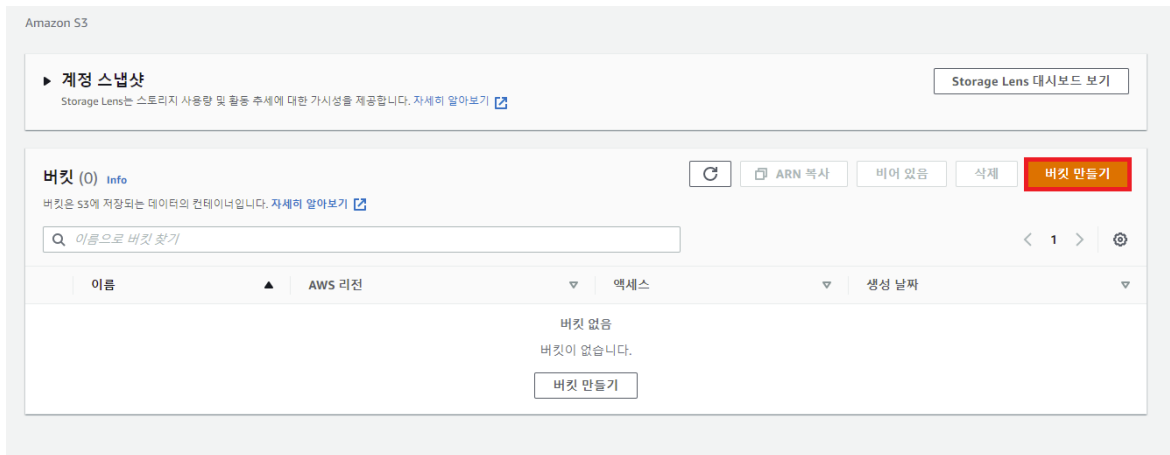
rm -rf /var/lib/jenkins/workspace/gitlab

echo "> 현재 구동중인 pid 확인"
CURRENT_PID=$(ps -ef | grep java | grep jenkins | grep gitlab | awk '{print $2}')
echo "$CURRENT_PID"
if [ -z $CURRENT_PID ]; then
    echo "> 종료할 pid가 없습니다."
else
    echo "> kill -9 $CURRENT_PID"
    kill -9 $CURRENT_PID
fi
BUILD_ID=dontkillME nohup java -jar findit-0.0.1-SNAPSHOT.jar & echo $! > program.pid
```

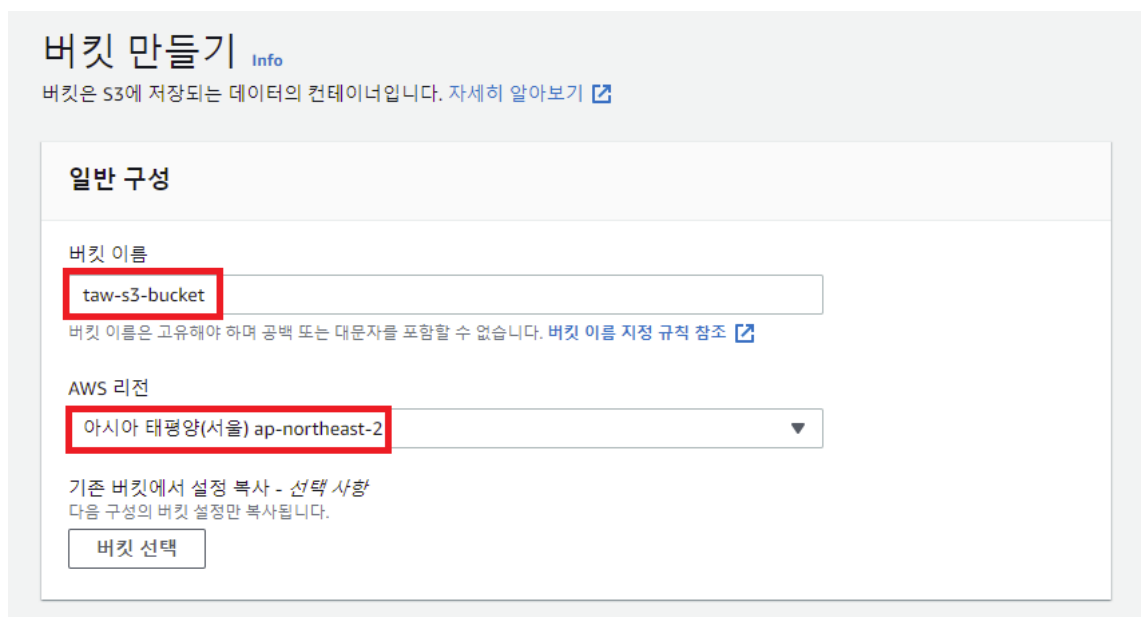
7. 외부 서비스

A. AWS S3

1. 버킷 만들기 클릭



2. 버킷 이름, 리전 입력



3. 퍼블릭 액세스 설정 : 체크 모두 해제

퍼블릭 액세스 차단 편집(버킷 설정) Info

퍼블릭 액세스 차단(버킷 설정)

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 모든 S3 버킷 및 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 [모든 퍼블릭 액세스 차단]을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 [모든 퍼블릭 액세스 차단]을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 버킷 또는 내부 객체에 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

☐ 모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

- ☐ **ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단**
S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하며, 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.
- ☐ **임의의 ACL(액세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단**
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.
- ☐ **새 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단**
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지점 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.
- ☐ **임의의 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 교차 계정 액세스 차단**
S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지점에 대한 퍼블릭 및 교차 계정 액세스를 무시합니다.

취소

변경 사항 저장

4. 버킷 생성 확인

버킷 (1) <small>Info</small>					ARN 복사	비어 있음	삭제	버킷 만들기
버킷은 S3에 저장되는 데이터의 컨테이너입니다. 자세히 알아보기								
<input type="text" value="이름으로 버킷 찾기"/>				<div>< 1 > ⌂</div>				
이름	AWS 리전	액세스	생성 날짜					
taw-s3-bucket	아시아 태평양(서울) ap-northeast-2	객체를 퍼블릭으로 설정할 수 있음	2021. 9. 7. pm 3:02:11 PM KST					

5. 버킷 정책 편집

버킷 정책 편집

Info

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

정책 예제

정책 생성기

2

버킷 ARN

arn:aws:s3:::js-test1-bucket

정책

1

1

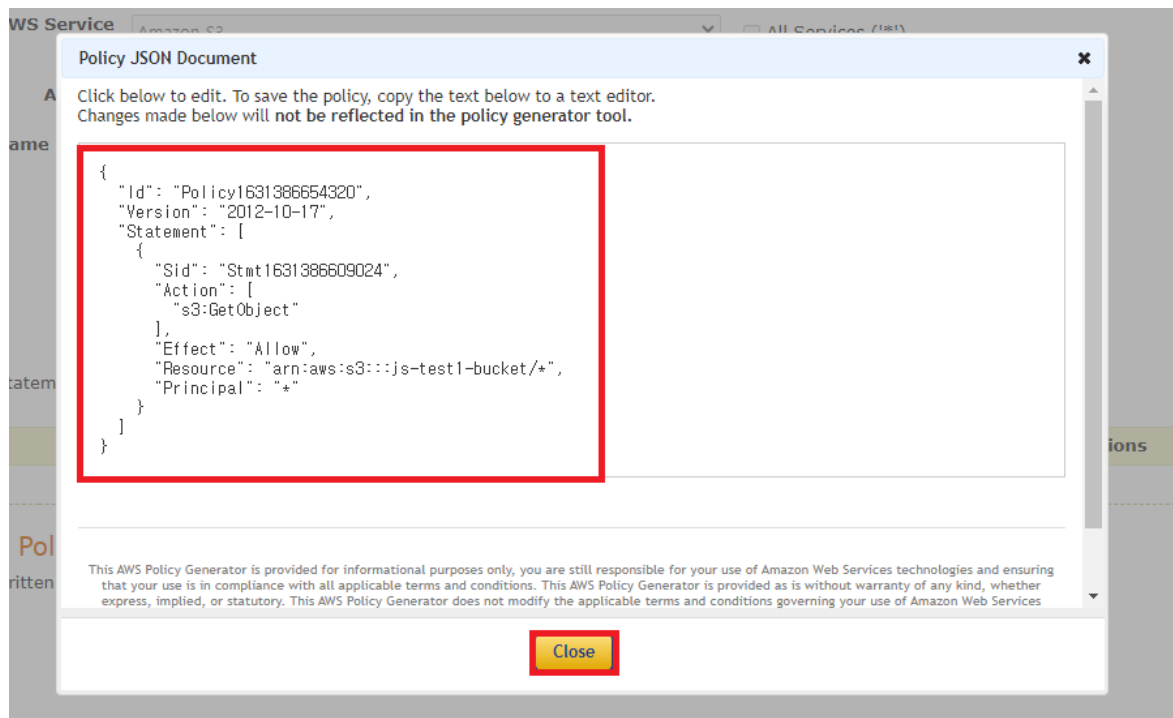
외부 액세스 미리 보기

6. 버킷 정책 생성

포팅메뉴얼

14

9. 버킷 정책 생성(4)



10. 버킷 정책 편집 적용

버킷 ARN
arn:aws:s3::js-test1-bucket

정책

```

1 {
2   "Id": "Policy1631386654320",
3   "Version": "2012-10-17",
4   "Statement": [
5     {
6       "Sid": "Stmt1631386609024",
7       "Action": [
8         "s3:GetObject"
9       ],
10      "Effect": "Allow",
11      "Resource": "arn:aws:s3::js-test1-bucket/*",
12      "Principal": "*"
13    }
14  ]
15 }

```

외부 액세스 미리 보기

리소스에 대한 외부 액세스를 스캔한 Access Analyzer 결과를 미리 보고 검증합니다. 자세히 알아보기

Access Analyzer를 사용하여 버킷에 대한 외부 액세스 미리 보기

Access Analyzer는 버킷에 대한 외부 액세스를 스캔한 결과를 미리 볼 수 있도록 기존 버킷 권한과 함께 버킷 정책을 분석합니다. 이를 통해 정책을 저장하기 전에 버킷에 대한 퍼블릭 및 교차 계정 액세스를 검증할 수 있습니다. 시작하려면 분석기를 선택하고 미리 보기를 선택합니다.

[Access Analyzer에 대해 자세히 알아보기](#)

분석기 없음

버킷에 대한 외부 액세스를 미리 보려면 버킷의 리전에 분석기를 생성합니다.

[Access Analyzer로 이동](#)

취소 **변경 사항 저장**

- ERROR : "The bucket does not allow ACLs"

1. assume you have created the s3 bucket, in the list page

Amazon S3 > Buckets > data-store.raindrop.link

data-store.raindrop.link

Objects | Properties | **Permissions** | Metrics | Management | Access Points

Permissions overview

Access
Objects can be public

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Edit ← step1. click this button (other wise you can not edit the ACL)

Block all public access
Off
▶ Individual Block Public Access settings for this bucket

Bucket policy
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. [Learn more](#)

Edit **Delete**

2. don't toggle the "block" options

Edit Block public access (bucket settings) [Info](#)

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ Block *all* public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ Block public access to buckets and objects granted through *new* access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ Block public access to buckets and objects granted through *any* access control lists (ACLs)

S3 will ignore all ACLs that grant public access to buckets and objects.

☐ Block public access to buckets and objects granted through *new* public bucket or access point policies

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

☐ Block public and cross-account access to buckets and objects through *any* public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Cancel

Save changes

3. find the ownership, then click edit.

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Object Ownership

Bucket owner enforced

ACLs are disabled. All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

Edit

4. edit the object owner ship (ACLs enabled)

Amazon S3 > Buckets > data-store > Edit Object Ownership

Edit Object Ownership Info

Object Ownership
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ **ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

☒ **ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Enabling ACLs turns off the bucket owner enforced setting for Object Ownership
Once the bucket owner enforced setting is turned off, access control lists (ACLs) and their associated permissions are restored. Access to objects that you do not own will be based on ACLs and not the bucket policy.

☒ I acknowledge that ACLs will be restored.

Object Ownership

☒ **Bucket owner preferred**
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.

☐ **Object writer**
The object writer remains the object owner.

Info If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#)

Cancel **Save changes**

5. now the edit button for ACL is clickable.

Access control list (ACL)
Grant basic read/write permissions to other AWS accounts. [Learn more](#)

Info The console displays combined access grants for duplicate grantees. To see the full list of ACLs, use the Amazon S3 REST API, AWS CLI, or AWS SDKs.

now it's editable [Edit](#)

Grantee	Objects	Bucket ACL
Bucket owner (your AWS account) Canonical ID: 1d706b4429ecc39917e4cf53d6e737460a7e10d0ff0c57b12138b79dcf8985b	List, Write	Read, Write
Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers	-	-
Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers	-	-
S3 log delivery group Group: http://acs.amazonaws.com/groups/s3/LogDelivery	-	-

6. toggle the permissions you want and save changes.

Edit access control list (ACL) [Info](#)

Access control list (ACL)

Grant basic read/write permissions to other AWS accounts. [Learn more](#)

Grantee	Objects	Bucket ACL
<div>Bucket owner (your AWS account)</div> <div>Canonical ID: 1d706b4429ecc39917e4cf53d6e737460a7e10d0ff0c57b12138b79dcf8985b</div>	<div><input checked="" type="checkbox"/> List</div> <div><input checked="" type="checkbox"/> Write</div>	<div><input checked="" type="checkbox"/> Read</div> <div><input checked="" type="checkbox"/> Write</div>
<div>Everyone (public access)</div> <div>Group: http://acs.amazonaws.com/groups/global/AllUsers</div>	<div><input type="checkbox"/> List</div> <div><input type="checkbox"/> Write</div>	<div><input checked="" type="checkbox"/> <div>⚠ Read</div></div> <div><input type="checkbox"/> Write</div>
<div>Authenticated users group (anyone with an AWS account)</div> <div>Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers</div>	<div><input type="checkbox"/> List</div> <div><input type="checkbox"/> Write</div>	<div><input type="checkbox"/> Read</div> <div><input type="checkbox"/> Write</div>
<div>S3 log delivery group</div> <div>Group: http://acs.amazonaws.com/groups/s3/LogDelivery</div>	<div><input type="checkbox"/> List</div> <div><input type="checkbox"/> Write</div>	<div><input type="checkbox"/> Read</div> <div><input type="checkbox"/> Write</div>

⚠ When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access the objects in this bucket.

[Learn more](#)

☐ I understand the effects of these changes on my objects and buckets.

Access for other AWS accounts

No other AWS accounts associated with the resource.

Add grantee

Save changes