A403 : Safers

삼성SW청년아카데미 서울캠퍼스 5기

자율프로젝트 (6주; 2021/10/12 ~ 2021/11/19)

포팅 매뉴얼

담당 컨설턴트 : 한기철

팀명 : 떡잎방범대

당현아(팀장), 안상훈, 이상현, 이영주, 정원석

<<목차>>

1. 프로젝트 기술 스택

가. 이슈관리 : Jira

나. 형상관리 : Gitlab

다. 커뮤니케이션 : Mattetmost, Notion

라. 디자인 : Adobe Photoshop, Figma

마. 개발 환경

1) OS : Windwos 10

2) DB : MySQL 8.0.22

3) Server : AWS EC2, AWS S3, Ubuntu 20.04, Jenkins 2.303.2, nginx 1.18.0

바. Frontend

1) HTML5, CSS3, JavaScript(ES6)

2) Vue 2.6.11, Vuex 3.4.0, Node.js 10.19.0

3) Lib : Bootstrap 5.1.3, vue modal 2.0.1, simple alert 1.1.1

사. Backend

1) Java open JDK zulu 8.33.0.1

2) SpringBoot Gradle 6.7

3) Lombok 1.18.20, JPA, Swagger, https, Security

아. Unity

1) Unity Community 2019.4.32f1

2) WebGL 2.0, Photon 2.38

자. Machine Learning

1) Tensorflow 3.11.0

2) Tensorflow Mofilenet models 2.1.0

3) Tensorflow Teachable Machine Image 0.8.5

4) Tensorflow core 3.11.0, Tensorflow backend wasm 3.11.0

차. IDE&Tool

1) intelliJ 21.1.2

2) MySQL Workbench

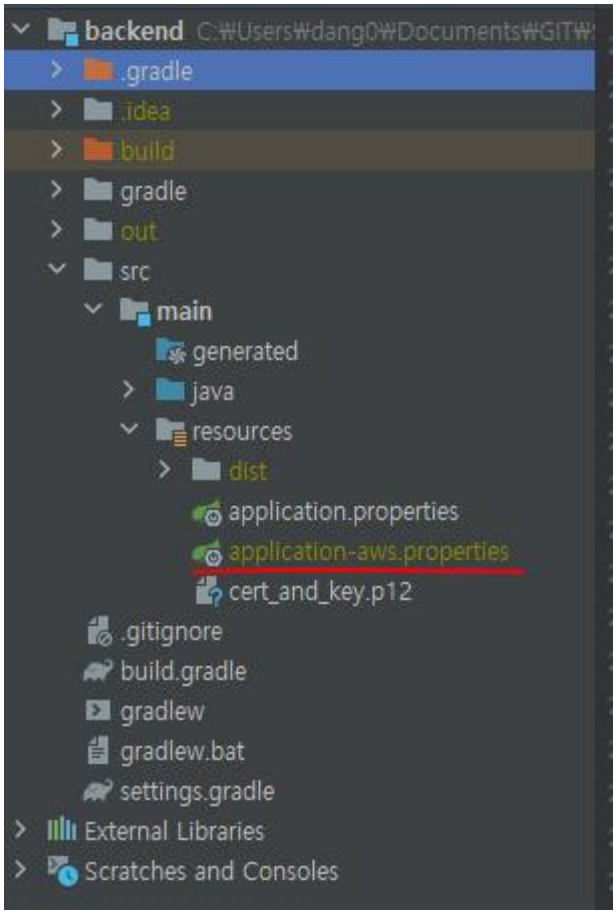
3) Visual Studio Code 1.58

4) MobaXterm

2. 빌드 상세내용

가. Safers Backend 빌드

: safers는 보안을 위해 aws 등의 id/pw를 별도로 운영합니다. 따라서 application-aws.properties는 gitignore 필터링으로 repo에 없습니다. 따라서 빌드를 위해서는 application-aws.properties를 다음과 같은 위치에 첨부해야하는 과정이 필요합니다.



application-aws.properties 전문

```
# AWS
cloud.aws.credentials.access-key=AKIARMPAI5JUT7G62I6N
cloud.aws.credentials.secret-key=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
cloud.aws.stack.auto=false

# AWS S3
cloud.aws.s3.bucket=safers
cloud.aws.region.static=ap-northeast-2
```

나. Safers Frontend 빌드

1) node_modules를 위한 기본 install

```
npm install
```

2) 현재 상태로 빌드하기

```
npm run build
```

3. 배포 특이사항

가. AWS EC2에 Jenkins를 설치해 자동 배포를 진행하고 있습니다.

나. 다음과 같은 세팅을 설정했습니다.

1) Jenkins Setting Key

profile Personal Access
name: safers_access_token
sHJi5n-ssPkoY9QixAbb

Project Access Token
name: safers_repository
FxKxW_8PiyTkc1Vcg9VV

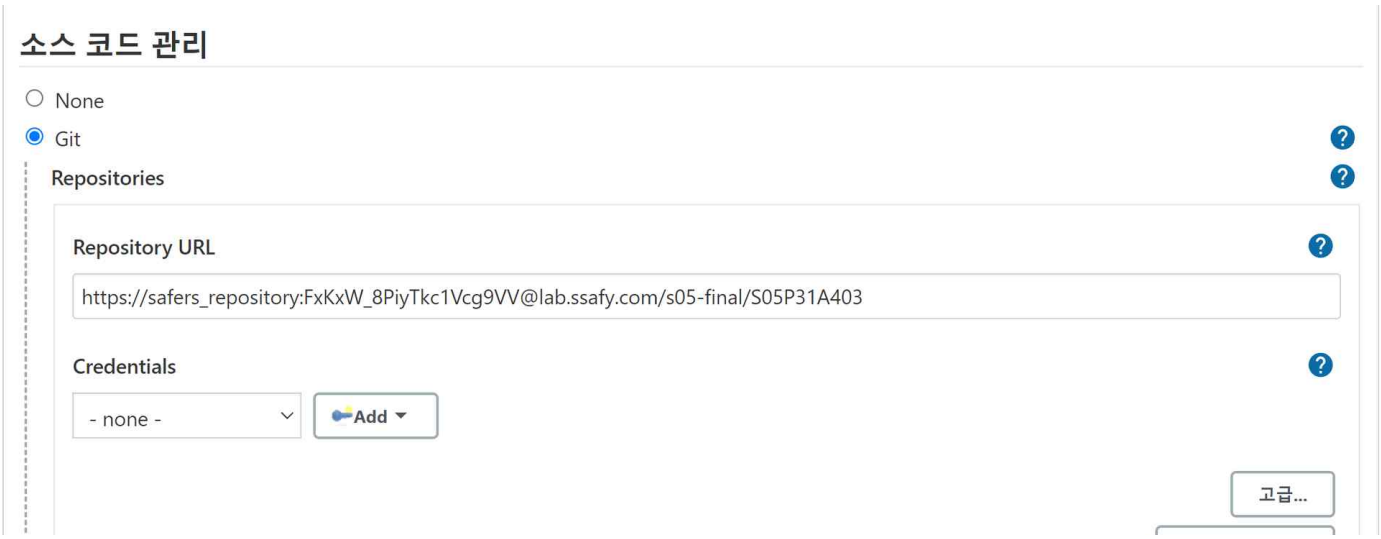
2) 배포 EC2에 다음과 같이 세팅합니다.

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key |
  sudo apt-key add -
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > ₩
/etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt-get install jenkins
sudo apt-get install openjdk-8-jdk
```

3) 3-나-2) 과정에서 `E: Sub-process /usr/bin/dpkg returned an error code (1)` 오류가 발생할 시 다음과 같은 명령어로 해결합니다.

```
sudo rm /var/lib/dpkg/info/*
sudo dpkg --configure -a
sudo apt update -y
```

4) 이후 다음과 같이 세팅했습니다.



Build

Execute shell

Command

```
echo 'jenkins build started...'

### frontend build ###
cd frontend
npm install
npm run build

### backend build ###
cd ..
sudo chmod -R 777 backend
cd backend
./gradlew clean build

pid=$(ps -eaf | grep backend-0.0.1-SNAPSHOT.jar | grep -v "grep" | grep -v $0 | awk '{print $2}')
if [[ $pid == "" ]]
then
 echo backend-0.0.1-SNAPSHOT.jar is not running
else
 sudo kill -9 $pid
 echo backend-0.0.1-SNAPSHOT.jar process killed forcefully, process id $pid.
fi
```

See the list of available environment variables

빌드 후 조치

Post build task

Tasks

Log text

BUILD SUCCESS

Operation

-- AND --

추가

Script

```
# sudo 를 붙이지 않으면 실행되지 않았음.(nohup 이 붙어서 그런걸로 판단)
sudo nohup java -jar backend/build/*.jar &
```

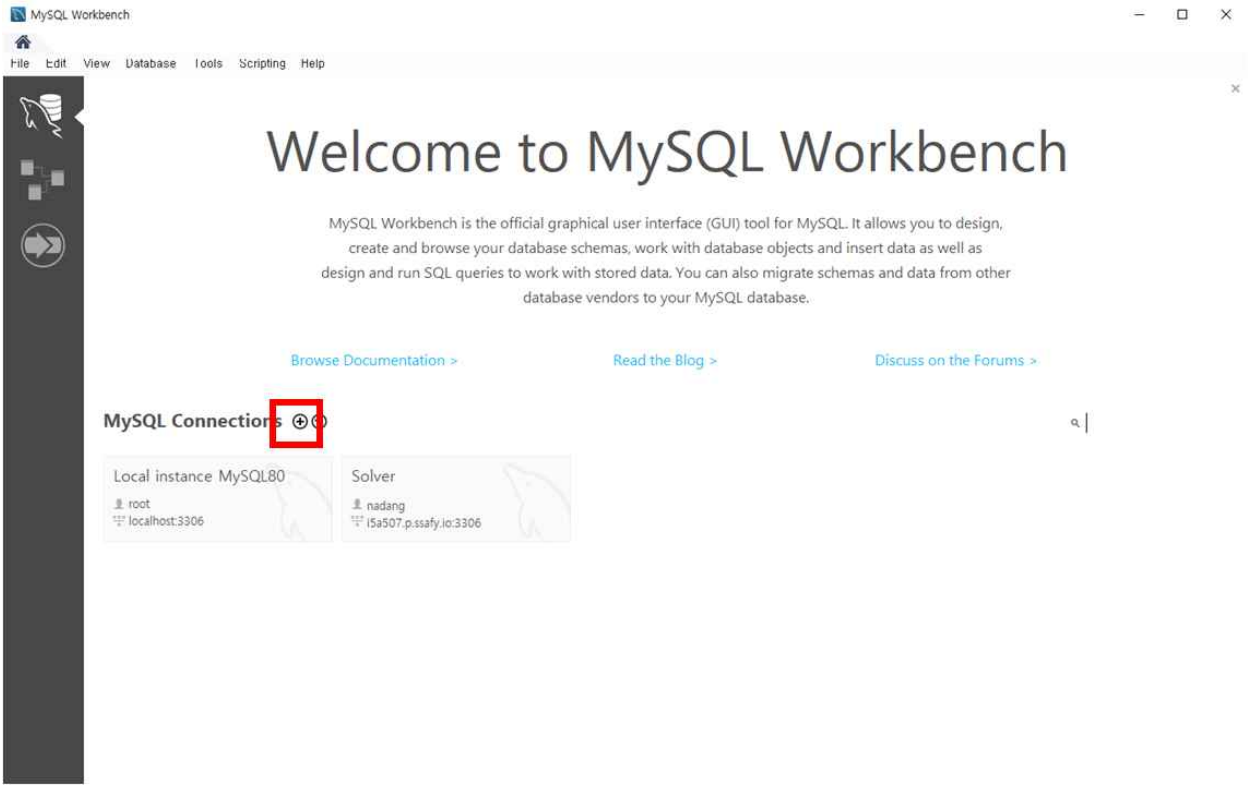
Run script only if all previous steps were successful

Escalate script execution status to job status

다. 배포가 정상적으로 진행된 후 서버에서 확인하면서 마무리합니다.

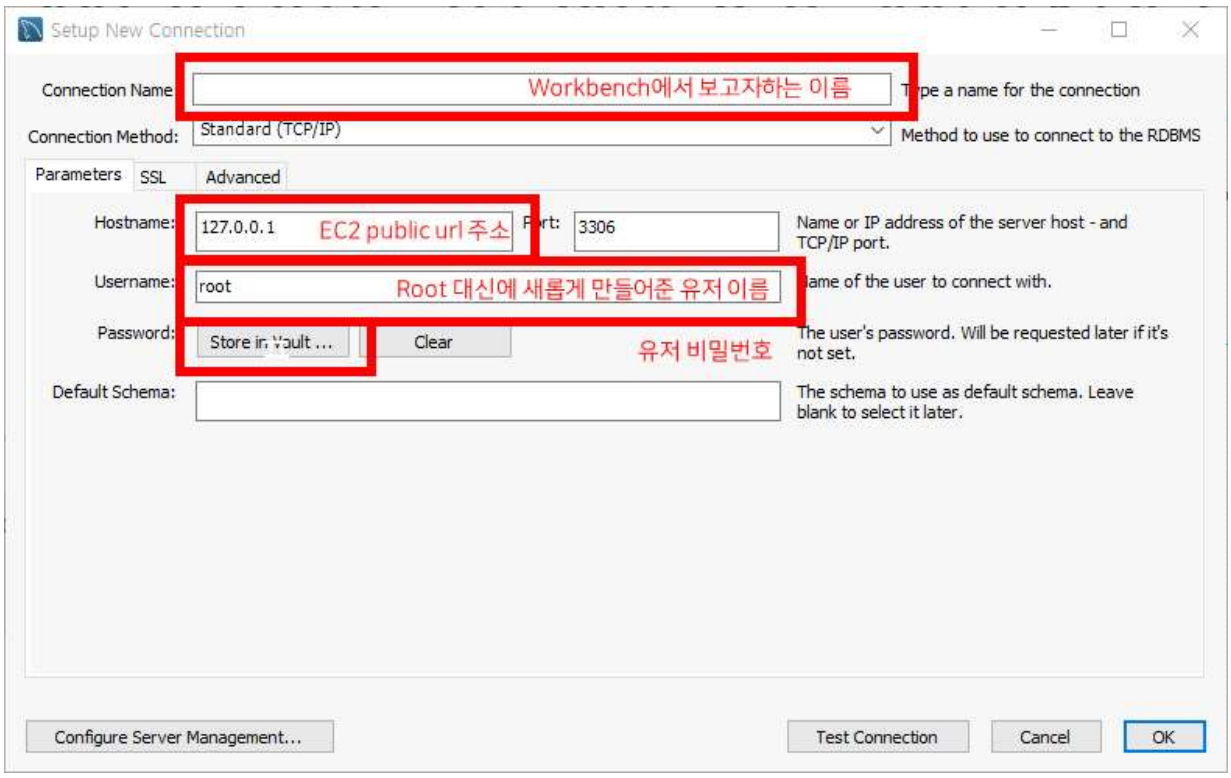
4. DB 계정

가. MySQL WorkBench 추가하기



MySQL WorkBench를 열어서 새로운 내용을 추가하기 위해 '+' 버튼을 눌러줍니다.

나. EC2 계정 정보 넣기



- username : safers, password : safers1119!

: 기본 root계정이 아닌 별도의 safers(팀명) 계정을 만들어서 진행했습니다.

5. 프로퍼티 정의

가. ngnix 세팅

1) ec2에서 세팅 파일로 접근

```
sudo vi /etc/nginx/sites-enabled/defulat
```

2) 세팅값 다음과 같이 변경하기

```
server{
        listen 80;
        listen [::]:80;

        server_name k5a403.p.ssafy.io;
        return 301 https://k5a403.p.ssafy.io$requeset-uri;
}

server {
        client_max_body_size 10M;

         listen 443 default_server ssl;

        server_name k5a403.p.ssafy.io;

        ssl_certificate /etc/letsencrypt/live/k5a403.p.ssafy.io/fullchain.pem;
        ssl_certificate_key /etc/letsencrypt/live/k5a403.p.ssafy.io/privkey.pem;

         location / {
                #root    /var/lib/jenkins/workspace/Safers/frontend/dist;
                #index   index.html index.htm index.nginx-debian.html;
                #try_files $uri $uri/ /index.html;

                add_header 'Access-Control-Expose-Headers' 'ETag';

                proxy_set_header X-Forwarded-For    $proxy_add_x_forwarded_for;
                proxy_set_header X-Forwarded-Proto https;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header HOST $http_host;
                proxy_set_header X-NginX-Proxy true;

                proxy_pass https://127.0.0.1:8080;
                proxy_redirect off;
        }
}
```

나. 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 servie mysql restart
```

6) root 계정 외의 사용할 계정 생성

```
sudo mysql -u root –p
CREATE USER 'new name'@'%' IDENTIFIED BY 'new password';
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;
```

7) 확인

```
sudo mysql -u <new-name> -p
```

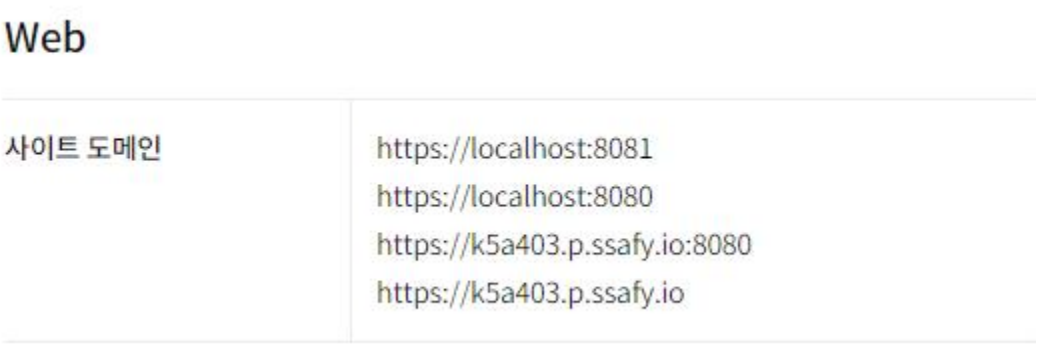
6. 외부 서비스

가. 카카오

: 서비스의 회원가입/로그인을 카카오로 진행하였습니다. 서비스 기능에 집중할 수 있으며, 회원가입/로그인의 다양한 절차를 생략할 수 있어서 이용자의 편의성을 제공합니다.

1) 애플리케이션 추가

2) 도메인 등록



3) redirect URI 설정

4) 로그인 활성화

5) 카카오 인가 코드 수신

GET

HTTP/1.1

Host: kauth.kakao.com

/oauth/authorize?client_id={REST_API_KEY}&redirect_uri={REDIRECT_URI}&response_type=code

6) kakao accessToken 수신

POST

HTTP/1.1

Host: kauth.kakao.com

/oauth/token

Content-type: application/x-www-form-urlencoded;charset=utf-8

7) accessToken으로 정보 요청

GET/POST

HTTP/1.1

Host: kapi.kakao.com

/v2/user/me

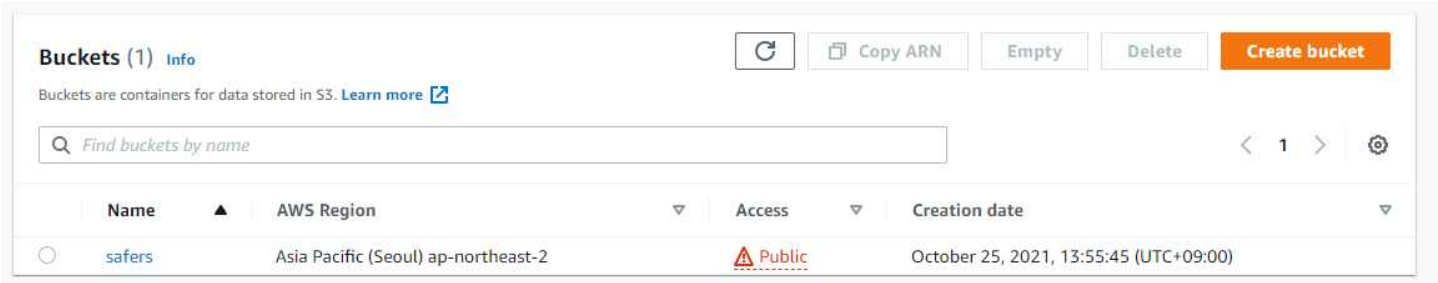
Authorization: Bearer {ACCESS_TOKEN}

Content-type: application/x-www-form-urlencoded;charset=utf-8

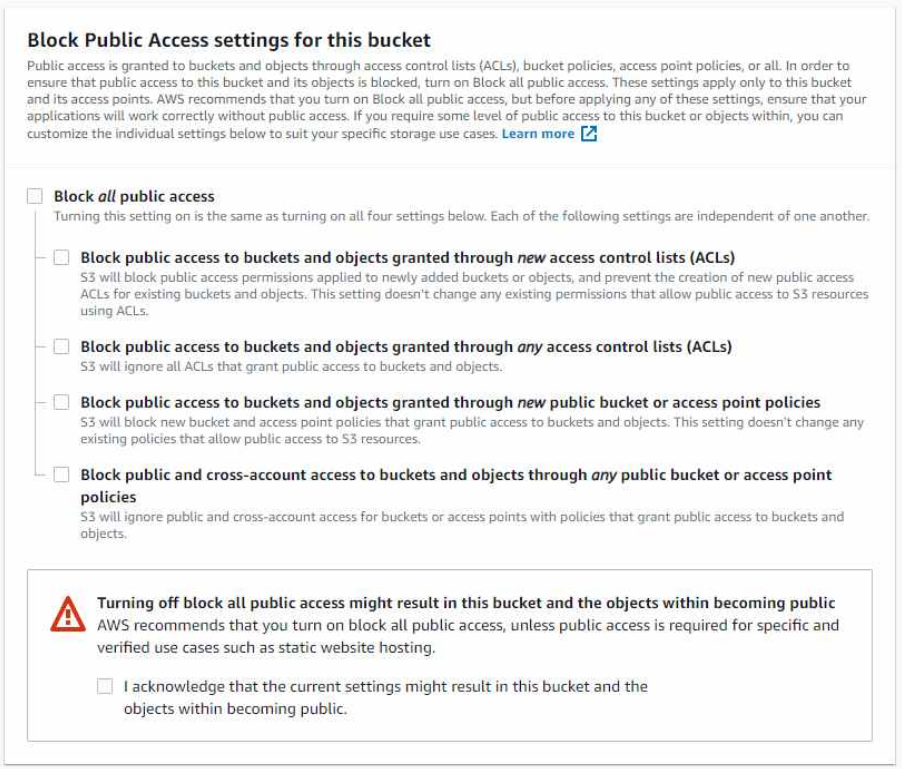
나. AWS S3

: 서비스 내의 영상, 사진 등을 저장하여, 관련 url를 사용할 수 있는 클라우드입니다. Bucket에 대한 기본적인 관리나 엑세스가 필요합니다.

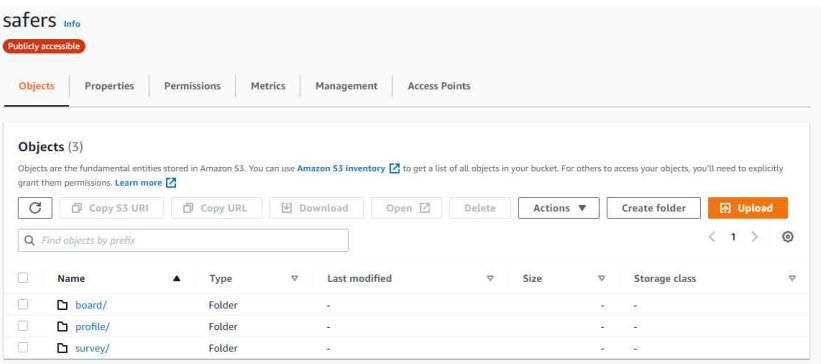
1) 버킷 만들기



2) 버킷 엑세스 조건



3) 버킷 구조 세팅



필요한 구조를 세팅해서 각 담당자에게 S3 url을 알려줍니다.

4) 버킷 정책 편집

```
{
    "Version": "2012-10-17",
    "Id": "Policy1629180304732",
    "Statement": [
        {
            "Sid": "Stmt1629180303259",
            "Effect": "Allow",
            "Principal": "*",
            "Action": [
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:GetObjectAcl",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": "arn:aws:s3:::solver-bucket/*"
        }
    ]
}
```

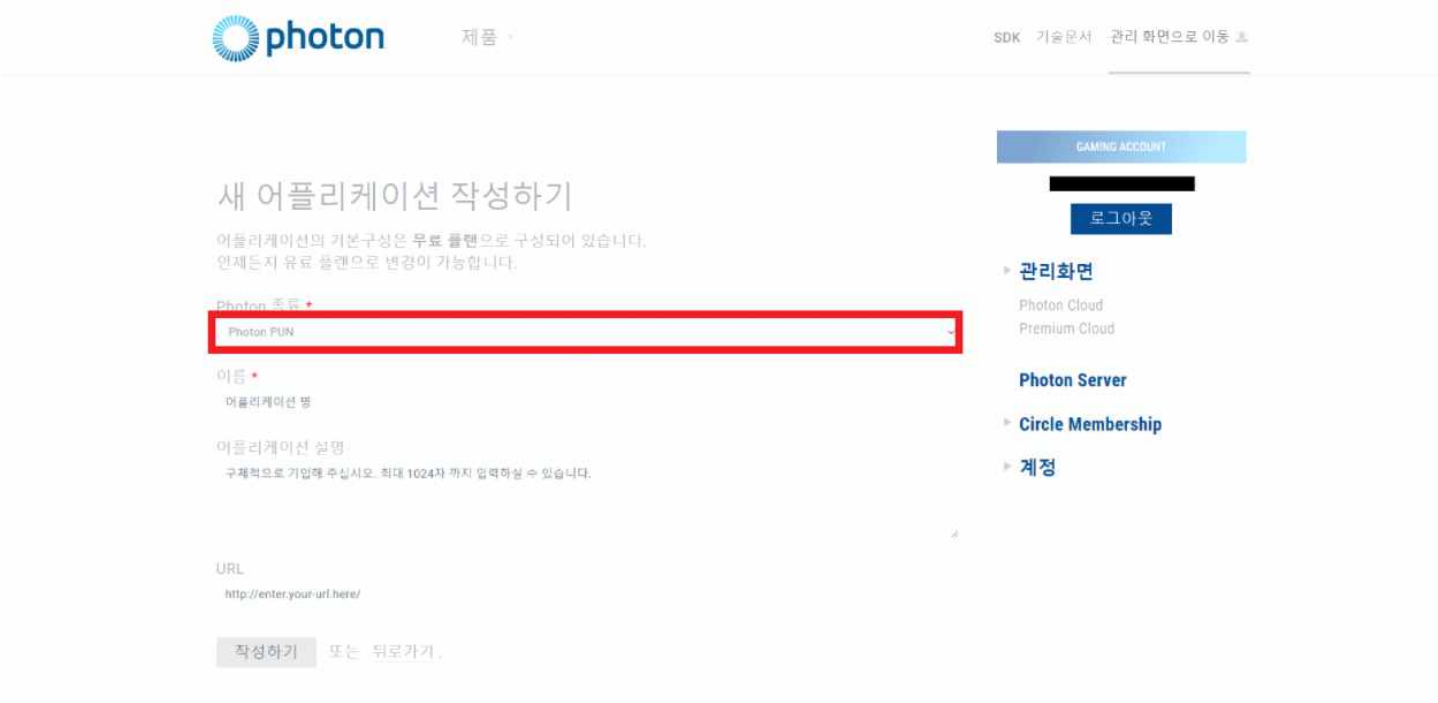
5) 버킷 엑세스 키

```
User
Name : Safers
Access : AKIARMPAI5JUT7G62I6N
Secret access key : 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
```

다. Photon 클라우드 세팅

1) 포톤 엔진 사이트

가) 우선 포톤 엔진 사이트에서 회원가입 후, 새 어플리케이션을 생성합니다. 포톤 종류는 Photon PUN으로 선택하고 해당 어플리케이션 이름, 설명, URL을 입력합니다.



나) 생성한 어플리케이션 ID를 확인합니다.

2) 유니티 포톤 설정

: 유니티 에셋스토어에서 Photon PUN2 무료버전을 구매합니다. 해당 에셋을 유니티 에디터에서 Import한 후에 PUN Wizard에 확인한 어플리케이션 ID를 입력합니다.