

## A507 : SOLVER

삼성SW청년아카데미 서울캠퍼스 5기  
공통프로젝트 (6주; 2021/07/12 ~ 2021/08/20)

## 포팅 매뉴얼

담당 컨설턴트 : 박찬국  
당현아(팀장), 나승호, 박나영, 조현민

---

## <<목차>>

1. 기술 스택	1
2. 빌드 상세내용	2
3. 배포 특이사항	4
4. DB 계정	5
5. 프로퍼티 정의	6
6. 외부 서비스	8

---

당신의 질문/답변은 특별하다. 사소하지만 어려운 문제를 해결하지 못해 답답했던 경험은 공부했던 모든 이들이 접했던 문제일 것입니다. 코로나19 시대에 도래하며, 대면하며 질의응답하는 것이 어려워진 만큼 학습에 대한 해소가 시원하지 않은 경우가 많죠.

이런 어려움을 해결하면서도, 답변자에게 활동을 기반으로 다양한 통계를 제공합니다. 여러 절차를 거치지 않고도 답변자는 본인을 어필할 수 있습니다. 질문자와 답변자 간의 예약도 도와드리며, 포인트 제도를 통해 더욱 효과적인 홍보도 가능합니다. 누구든 해결사를 찾고, 해결사가 되어보세요.

### 1. 프로젝트 기술 스택

가. 이슈관리 : Jira

나. 형상관리 : Gitlab

다. 커뮤니케이션 : Mattermost, Notion

라. 개발 환경

1) OS : Windows 10

2) IDE

가) Spring Tool Suite 3.9.14

나) Visual Studio Code 1.58

다) UI/UX : Adobe Photoshop, Figma

3) Database : MySQL Workbench 8.0.22

4) Server : AWS EC2 (MobaXterm)

가) Ubuntu 20.04.2 LTS

나) Docker 20.10.7

마. 상세 사용

1) Backend

가) Java (OpenJDK 8.33.0.1)

나) Spring Boot Gradle 7.2

다) Lombok 1.18.20, Kurento 2.0, swagger2 3.0.0, Querydsl-Jpa

2) Frontend

가) HTML5, CSS3, JavaScript(ES6)

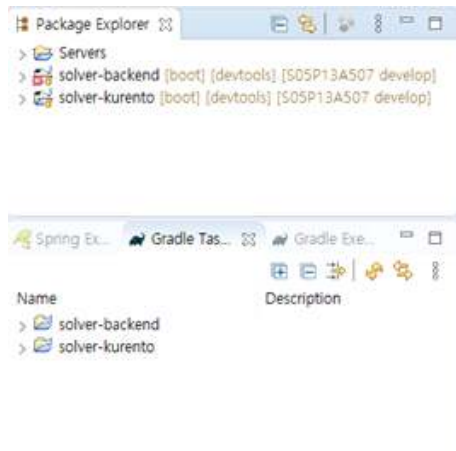
나) Vue 2.6.11, Vuex 3.4.0

다) Node.js 14.17.3

라) ckeditor5 1.0.5, chart.js 3.5.1, kurento 6.16.0, js-modal 2.0.1, randomcolor 1.0.4

3) AWS S3 1.12.39

## 2. 빌드 상세내용

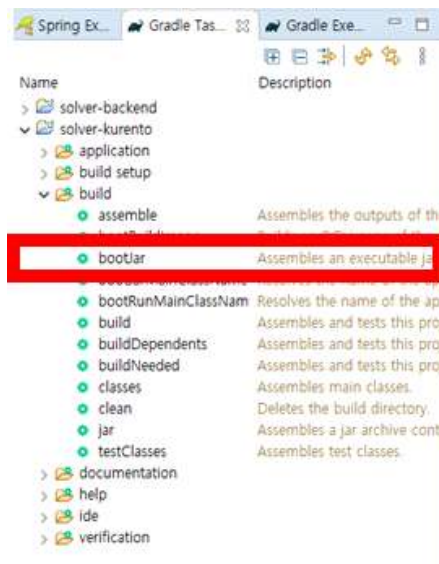


STS에서 solver-backend와 solver-kurento를 gradle로 import합니다. 첫 import 후에는 위 사진과 같이 보여지는 것(backend error)이 정상입니다. 지금부터 정상적인 빌드 과정을 위해 상세하게 진행해보도록 하겠습니다.

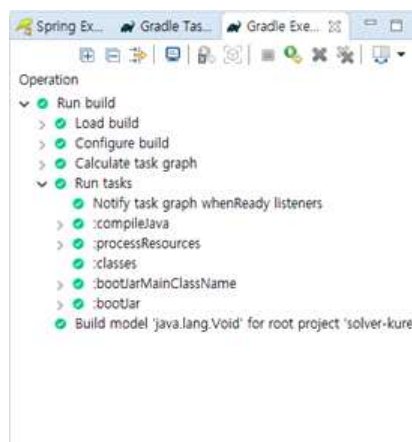
상단엔 Package Explorer, 하단엔 Gradle Tasks입니다. Gradle Tasks를 사용하면 명령어와 동일한 구동을 진행할 수 있으며, 단계별 성공 상태를 확인할 수 있습니다.

### 가. Solver Kurento 빌드

- 1) Gradle Tasks > solver-kurento > build 순으로 이동하여 BootJar를 더블클릭합니다.



- 2) 순차적으로 진행하며, 다음과 같은 성공 Operation이 뜨면 Jar가 정상적으로 만들어집니다.



#### 나. Solver Backend 빌드

: solver는 querydsl-jpa를 사용하여 src/main/generated를 만들며, Q-Entity가 만들어져야 정상적인 구동을 할 수 있습니다.

```
def querydslDir = 'src/main/generated'

querydsl {
    jpa = true
    querydslSourcesDir = querydslDir
}

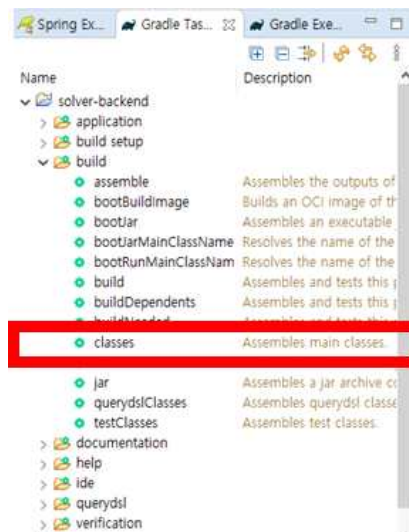
sourceSets {
    main.java.srcDir querydslDir
}

configurations {
    querydsl.extendsFrom compileClasspath
}

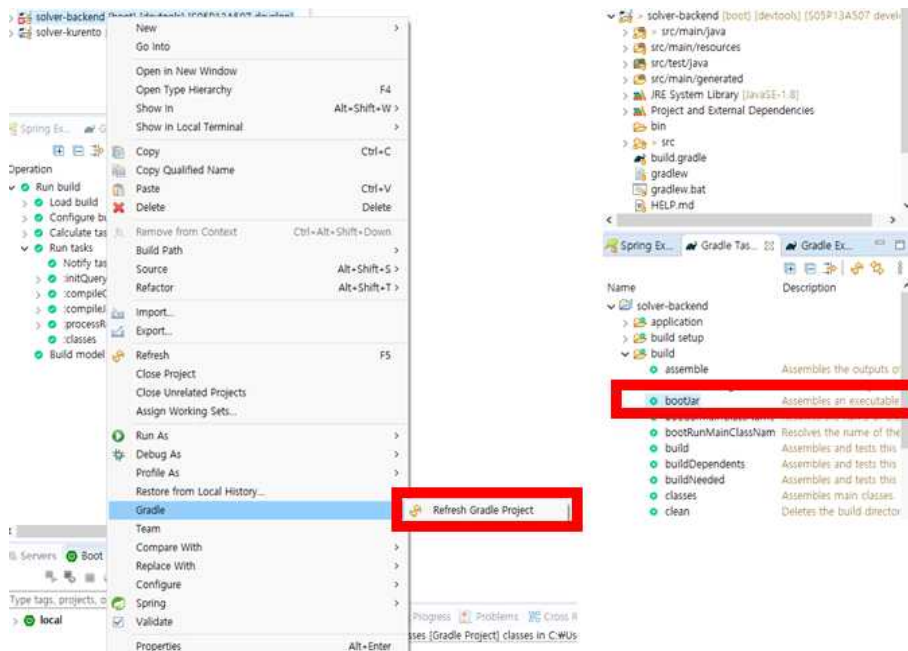
compileQuerydsl {
    options.annotationProcessorPath = configurations.querydsl
}
```

(생성을 위한 세팅 값; build.gradle 중 일부)

1) Gradle Tasks > solver-backend > build 순으로 이동하여 classes를 더블클릭합니다.



2) Project 우클릭 > Gradle > Refresh Gradle Project를 실행합니다. 우측 하단에 Gradle Update Bar가 동작하며, 모두 끝나면 사라지게 됩니다. 또한 project의 오류도 사라집니다. 이후 kurento와 동일하게 bootJar로 backend jar로 만들어줍니다.



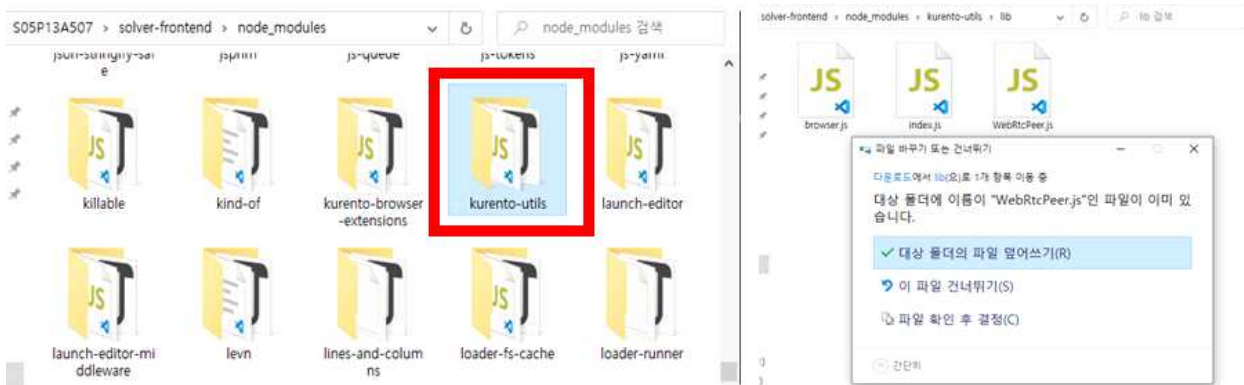
#### 다. Solver Frontend 빌드

##### 1) node\_modules를 위한 기본 install

```
npm install
```

##### 2) kurento setting 수정

: kurento의 값을 프로젝트 서비스에 필요한 기능 위해 코드추가한 WebRtcPeer.js를 추가합니다.  
위치는 node\_modules > kurento-utils > lib 내부입니다.



##### 3) 현재 상태로 빌드하기

```
npm run build
```

### 3. 배포 특이사항

가. AWS EC2에 만들어진 jar 파일 2개와 dist 폴더를 업로드 후 (수동)배포를 진행합니다.

나. 다음과 같은 명령어로 확인합니다.

##### 1) 현재 구동 중인 jar 파일 확인

```
ps -ef | grep .jar
```

명령어를 기재하면 현재 구동 중인 jar 프로세스와 PID가 보입니다.

##### 2) 구동 중인 프로세스 종료

```
kill -9 <PID>
```

해당으로 프로세스를 종료하여, 배포 서버에서 정상적으로 배포가 중단되었는지 확인합니다.

##### 3) 새로운 jar 무중단 배포 진행

```
nohup java -jar <jar-file-name>.jar &
```

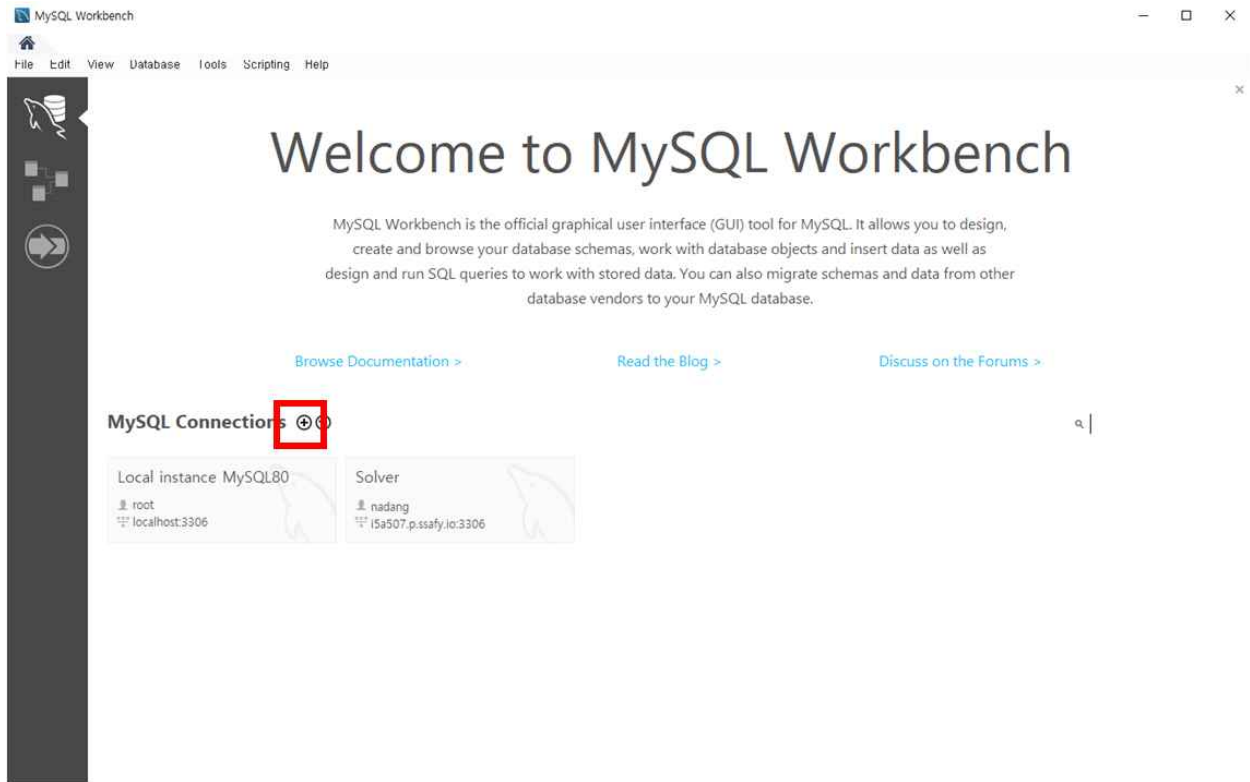
##### 4) nginx 재실행

```
sudo service nginx restart
```

다. 배포가 정상적으로 진행된 후 서버에서 확인하면서 마무리합니다.

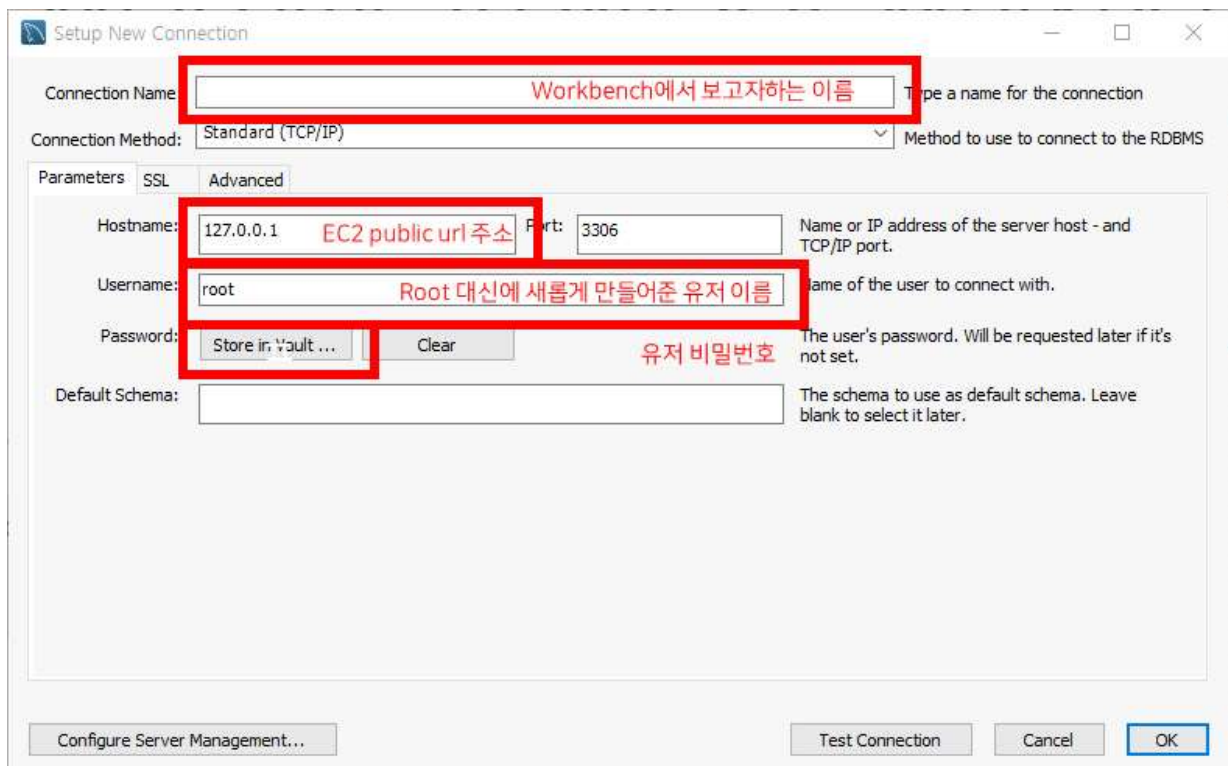
## 4. DB 계정

### 가. MySQL WorkBench 추가하기



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

### 나. EC2 계정 정보 넣기



- username : nadang, password : nadang0820

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

## 5. 프로퍼티 정의

가. nginx 세팅

1) ec2에서 세팅 파일로 접근

```
sudo apt-get update
```

2) 세팅값 다음과 같이 변경하기

```
server{
    root /home/ubuntu/dist;          # front root
    index index.html;

    location / {
        try_files $uri $uri/ /index.html;
        proxy_set_header X-Forwarded-Proto $scheme;
    }

    location /api/ {
        proxy_pass https://localhost:8080/api/v1/;
    }

    location /groupall/ {
        proxy_pass https://localhost:8433/groupcall/;
    }

    listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/i5a507.p.ssafy.io/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/i5a507.p.ssafy.io/privkey.pem;
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

server {
    if ($host = i5a507.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80 default_server;
    listen [::]:80 default_server;
    server_name i5a507.p.ssafy.io;
    return 404; # managed by Certbot
}
```

## 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 service 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
```

## 다. AWS EC2 Docker 세팅

### 1) apt 업데이트

```
sudo apt-get update  
sudo apt-get upgrade
```

### 2) npm 설치

```
sudo apt install npm
```

### 3) docker 설치

```
curl https://get.docker.com | sudo sh
```



## 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

### 애플리케이션 추가하기

앱 아이콘

이미지 업로드

파일 선택

JPG, GIF, PNG  
권장 사이즈 128px, 최대 250KB

앱 이름

SolverLoginApi

사업자명

solver

• 입력된 정보는 사용자가 카카오 로그인할 때 표시됩니다.

• 정보가 정확하지 않은 경우 서비스 이용이 제한될 수 있습니다.

취소

저장

#### 2) 도메인 등록

### Web

사이트 도메인

http://localhost

https://localhost

http://15a507.p.ssafy.io

https://15a507.p.ssafy.io

삭제

수정

#### 3) redirect URI 설정

### Redirect URI

Redirect URI

https://localhost:8080/api/v1/auth/login

http://localhost:8080/api/v1/auth/login

https://15a507.p.ssafy.io:8080/api/v1/auth/login

http://15a507.p.ssafy.io:8080/api/v1/auth/login

삭제

수정

#### 4) 로그인 활성화

카카오 로그인

ON

### 활성화 설정

상태

ON

## 5) 카카오 인가 코드 수신

### 앱 키

플랫폼	앱 키	복사	재발급
네이티브 앱 키	0325881be0a6e18222648a58463d81d7	복사	재발급
REST API 키	4d0b843e88238ebf6614549fce8bff85	복사	재발급
JavaScript 키	2648454fe4f717600d3afe43cb82b85d	복사	재발급
Admin 키	f005362bfd8b0ffec0ac3ab6a9fb74	복사	재발급

- 네이티브 앱 키: Android, iOS SDK에서 API를 호출할 때 사용합니다.
- JavaScript 키: JavaScript SDK에서 API를 호출할 때 사용합니다.
- REST API 키: REST API를 호출할 때 사용합니다.
- Admin 키: 모든 권한을 갖고 있는 키입니다. 노출이 되지 않도록 주의가 필요합니다.

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 액세스가 필요합니다.

### 1) 버킷 만들기



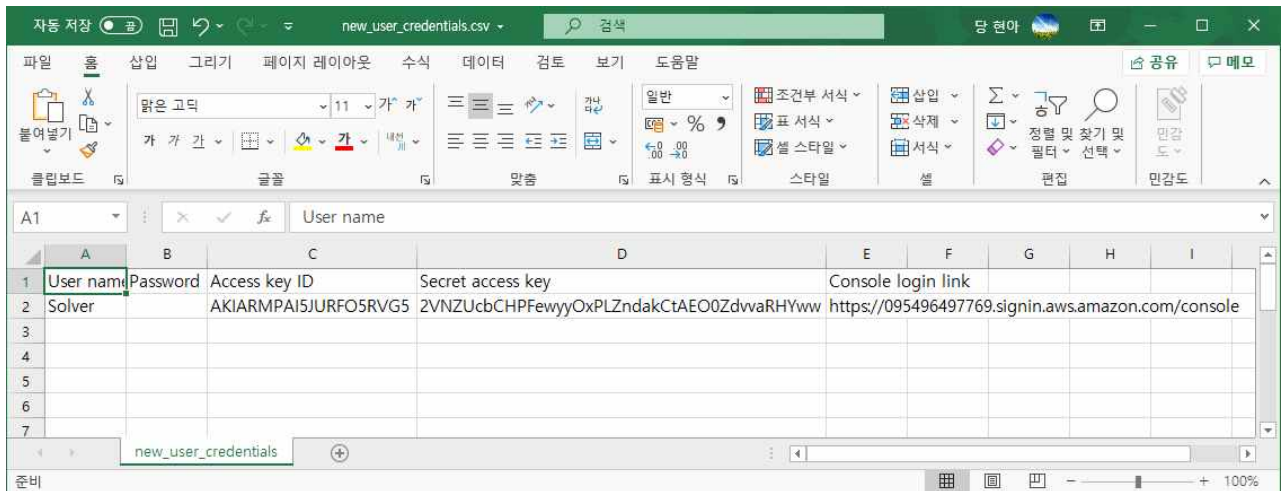
### 2) 기본 버킷 설정

The screenshot shows the 'General configuration' section of the AWS S3 'Create bucket' wizard. It includes a 'Bucket name' field with the value 'solver-service', a note that the name must be unique and not contain spaces or uppercase letters, an 'AWS Region' dropdown menu set to 'Asia Pacific (Seoul) ap-northeast-2', and a section for 'Copy settings from existing bucket - optional' with a 'Choose bucket' button.

### 3) 버킷 액세스 조건

The screenshot shows the 'Block Public Access settings for this bucket' section. It contains a paragraph explaining public access and a list of five checkboxes to block different types of public access. The first checkbox, 'Block all public access', is selected. Below the list is a warning box with a red triangle icon stating that turning off block all public access might result in the bucket and its objects becoming public. At the bottom, there is an acknowledgment checkbox: 'I acknowledge that the current settings might result in this bucket and the objects within becoming public.'

#### 4) 버킷 액세스 키 발급



User name	Password	Access key ID	Secret access key	Console login link
Solver		AKIARMPAI5JURFO5RVG5	2VNZUcbCHPFewyyOxPLZndakCtAE00ZdwvaRHYww	https://095496497769.signin.aws.amazon.com/console

버킷 생성이 정상적으로 종료되고 난 후, 해당 버킷 자료 엑셀을 받을 수 있습니다. 해당 액세스 키 값을 코드 적용해야만, 현재 개설된 버킷에 접근 가능합니다.

#### 5) 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/*"
    }
  ]
}
```

## 다. AWS EC2 – Kurento 세팅

### 1) docker kms 실행

```
sudo docker run --network host -p 8888:8888 -e TZ="Asia/Seoul" -e  
KMS_STUN_IP="3.35.48.39" -e KMS_STUN_PORT="3478" -e  
KMS_STUN_URL="myuser:mypassword@3.35.48.39?transport=udp" -d  
kurento/kurento-media-server:latest
```

### 2) kurento 설정

#### 가) docker 컨테이너 접근

```
docker exec -it e6 /bin/bash
```

#### 나) kurento https 키 설정

```
sudo vim /etc/kurento/kurento.conf.json
```