

홈트윗미



스마트 미러 홈트윗미와 함께하는
스마트한 홈트레이닝 서비스

- 포팅 매뉴얼 -

자율 3반 A306

팀장: 김보경 팀원: 김도현

팀원: 김준수 팀원: 유현우

팀원: 윤희욱 팀원: 최희선



HTWM 포팅 매뉴얼

1. 프로젝트 기술 스택

- A. FE - web
- B. FE - app
- C. BE
- D. IOT

2. 빌드 방법

- A. 백엔드 빌드 방법
- B. 앱 빌드 방법
- C. 일렉트론 어플리케이션 빌드 방법
- D. 배포 명령어 정리

3. DB 계정

- A. MySQL WorkBench 추가하기
- B. EC2 계정정보 넣기

4. 프로퍼티 정의

- A. Nginx Default 값 세팅
 - 1. ec2에서 세팅 파일로 접근
 - 2. 세팅값 다음과 같이 변경하기
- B. Git Ignore 파일
 - 1. credentials.yaml 파일

5. EC2 세팅

- A. AWS EC2 DB 세팅
 - 1. 세팅을 위한 최신 상태 업데이트
 - 2. MySQL 설치
 - 3. 추가 세팅을 위한 이동 후 편집
 - 4. 바뀐 내용
 - 5. 세팅 값 적용을 위한 재시작
 - 6. root 계정 외에 사용할 계정 생성
 - 7. 확인

6. Docker 세팅

- A. Docker 설치 명령어
- B. Dockerfile 내용

7. Jenkins 세팅

- A. Jenkins 구성
- B. GitLab 설정
- C. Jenkins 빌드, 배포 명령어

8. 외부서비스

- A. AWS S3
 - 1. 버킷 생성
 - 2. 버킷 설정
 - 3. 버킷 정책 설정

4. AWS IAM 설정

9. Jetson Nano 포팅 매뉴얼

A. 실행 방법

B. 실행 파일 별 설명

C. cmake 빌드시 주의사항

1. 프로젝트 기술 스택

A. FE - web

기술 스택(버전): node 16.14.0, Electron 21.1.1, webpack 5.74.0, @babel/core 7.19.3, React 18.2.0, reduxjs/toolkit 1.8.6, axios 1.1.3, sockjs-client 1.6.1, @stomp/stompjs 6.1.2

사용 툴: Visual Studio Code 1.70.1

B. FE - app

기술 스택(버전): expo 46.0.16, react 18.0.0, react-native 0.69.6, @babel/core 7.12.9, eslint 8.25.0, @types/react-native 0.69.1, typescript 4.3.5, axios 1.1.3, reduxjs/toolkit 1.8.6, @react-navigation/native 6.0.13

사용 툴: Visual Studio Code 1.73.1

C. BE

기술 스택(버전): Spring boot 2.7.1, MySQL 8.0.30, Nginx 1.18.0, Jenkins 2.346.2, OAuth2, AWS EC2, AWS S3, SockJS, STOMP, Docker 20.10.20

사용 툴: IntelliJ 2022.2, Mobaxterm, MySql workbench 8.0.20, JDK 11.0.15.1

D. IOT

기술 스택(버전): Python3 3.6, Raspberry Pi 4 Rasbian OS 34bit, Jetson Nano JetPack 4.5, OpenCV 4.5.0, websocket, openpose 1.7.0, cmake 3.25.0, Cuda 10.2, Cudnn 8.0, Nodejs 16.16.0, Electron 21.21.1

사용 툴: VNC Viewer, Mobaxterm, VScode

2. 빌드 방법

A. 백엔드 빌드 방법

1. Command Shell을 통해 프로젝트 폴더 안의 back\htwm 폴더 안으로 이동한다.
2. `./gradlew clean build` 명령어를 통해 빌드한다.
3. 프로젝트 폴더 안의 back\htwm\build\libs 안에 build 파일이 생성된다.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 https://aka.ms/pscore6

PS C:\Users\xofkd> cd "C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest"
PS C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest> ./gradlew clean build
Starting a Gradle Daemon (subsequent builds will be faster)

> Task :compileJava
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:54: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<MyPage> mypages = new ArrayList<>();
                                ^
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:57: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<Comment> comments = new ArrayList<>();
                                ^
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:60: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<Like> likes = new ArrayList<>();
                                ^
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:63: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<Bookmark> bookmarks = new ArrayList<>();
                                ^
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:66: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<Follow> followings = new ArrayList<>();
                                ^
C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\domain\User.java:69: warning: @Builder
r will ignore the initializing expression entirely. If you want the initializing expression to serve as default,
add @Builder.Default. If it is not supposed to be settable during building, make the field final.
    private List<Follow> followers = new ArrayList<>();
                                ^
Note: C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest\src\main\java\101\phorest\config\SecurityConfig.java uses
or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
6 warnings

> Task :test
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by retrofit2.Platform (file:/C:/Users/xofkd/.gradle/caches/modules-2/files-2.1
/net.nurigo/sdk/4.2.3/404cb5380b30d91e054610e4efe18a3333335d5b/sdk-4.2.3.jar) to constructor java.lang.invoke.Met
hodHandles$Lookup(java.lang.Class,int)
WARNING: Please consider reporting this to the maintainers of retrofit2.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2022-08-18 10:01:19.043 INFO 26828 --- [ionShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA
EntityManagerFactory for persistence unit 'default'
2022-08-18 10:01:19.047 INFO 26828 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1
- Shutdown initiated...
2022-08-18 10:01:19.107 INFO 26828 --- [ionShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1
- Shutdown completed.

BUILD SUCCESSFUL in 49s
8 actionable tasks: 8 executed
PS C:\Users\xofkd\Desktop\git\S07P12A101\BE\phorest>
```



주의! 빌드 하기 전에, credentials.yaml 파일이 프로젝트 폴더 안의
back\htwm\src\main\resources 폴더 안에 존재해야 한다.

B. 앱 빌드 방법

1. front_app 밑의 HTM 디렉토리로 이동
2. `npm i` 로 node_modules 패키지 설치
3. `expo build:android` 로 빌드
4. apk 로 빌드 선택
5. 콘솔창에 뜨는 다운로드 링크 클릭하여 apk 파일 다운로드
6. 해당 파일 기기에 설치

C. 일렉트론 어플리케이션 빌드 방법

1. 해당 디렉토리로 이동
2. (yarn이 없을 때) `$ npm install -g yarn` 설치 후 `$ yarn` 명령어로 패키지 설치
3. 해당 디렉토리에서 아래 명령어를 입력하여 빌드

(리눅스 앱의 경우 리눅스 환경에서 빌드 진행해야 함)

```
$ yarn build:linux
```

(윈도우)

```
$ yarn build
```

5. 생성된 dist 폴더에서 front_web 실행파일 사용

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 htwm-0.0.1-SNAPSHOT.jar &
```

BE 빌드 과정에서 생성된 빌드 파일의 경로로 이동해서 서버를 실행시킨다.

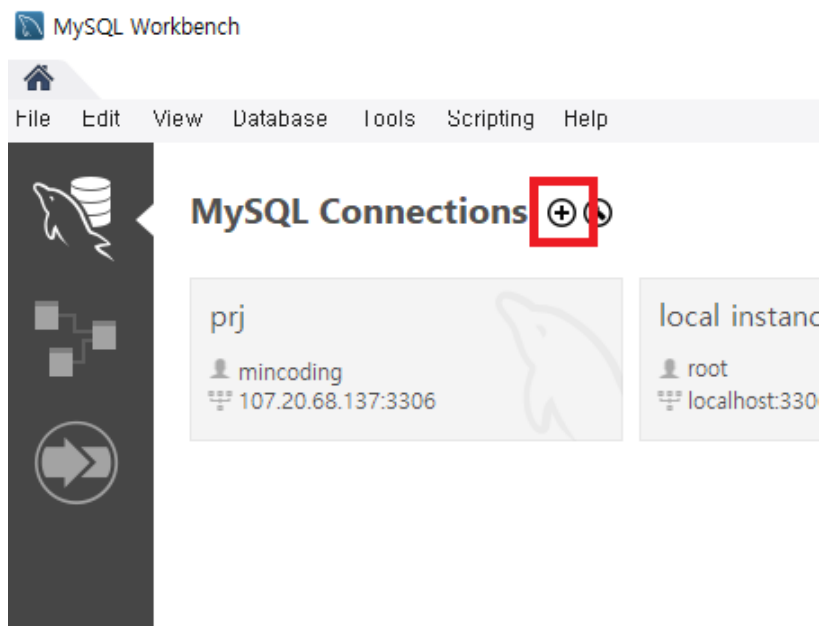
4. Nginx 재시작

```
sudo systemctl restart nginx
```

Nginx를 재시작한다.

3. DB 계정

A. MySQL WorkBench 추가하기



B. EC2 계정정보 넣기

Connection Name:

Connection Remote Management System Profile

Connection Method: Method to use to connect to the RDBMS

Parameters SSL Advanced

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

- username : `ssafy` , password : `/J}_2a=EKsHk6E*!8`

기존 root 계정이 아닌 별도의 ssafy 계정을 만들어서 진행했습니다.

4. 프로퍼티 정의

A. Nginx Default 값 세팅

1. ec2에서 세팅 파일로 접근

```
sudo apt get update
```

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

2. 세팅값 다음과 같이 변경하기

```
server {

    # SSL configuration
    #
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    #
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    #
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    #
    # include snippets/snakeoil.conf;

    root /var/www/html;
```



```

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html;
server_name k7a306.p.ssafy.io; # managed by Certbot

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

location /api{
    proxy_pass http://localhost:8399;
    proxy_redirect off;
    charset utf-8;

    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;
}

location /api/socket {

    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/socket;
}

# pass PHP scripts to FastCGI server
#
#location ~ /\.php$ {
#    include snippets/fastcgi-php.conf;
#
#    ## With php-fpm (or other unix sockets):
#    fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
#    ## With php-cgi (or other tcp sockets):
#    fastcgi_pass 127.0.0.1:9000;
#}

# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#
#location ~ /\.ht {
#    deny all;
#}

listen [::]:443 ssl ipv6only=on; # managed by Certbot
listen 443 ssl; # managed by Certbot
ssl_certificate /etc/letsencrypt/live/k7a306.p.ssafy.io/fullchain.pem; # managed by Certbot
ssl_certificate_key /etc/letsencrypt/live/k7a306.p.ssafy.io/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 ($host = k7a306.p.ssafy.io) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    listen 80 ;
    listen [::]:80 ;
    server_name k7a306.p.ssafy.io;
    return 404; # managed by Certbot

}

```

<https://s3-us-west-2.amazonaws.com/secure.notion-static.com/b504db0b-a3ea-449e-a5de-0c97946d15f2/default.txt>

B. Git Ignore 파일

1. credentials.yaml 파일

```

spring:
  datasource:
    url: jdbc:mysql://k7a306.p.ssafy.io:3306/htwm
    username: ssafy
    password: /J}_2a=EKsHk6E*!8
    driver-class-name: com.mysql.cj.jdbc.Driver

cloud:
  aws:
    stack:
      auto: false
    region:
      static: ap-northeast-2
    credentials:
      access-key: AKIA4JCUY2V0TRFFICKZ
      secret-key: WcCB8jxkg0CXsA50X0FJUZGmsRXyFABELT4V0zST
  s3:
    bucket: htm-ssafy

```

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

5. EC2 세팅

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. 바꿀 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
```

6. Docker 세팅

Docker를 이용해 컨테이너를 띄워 서버를 ec2에 배포하였습니다.

A. Docker 설치 명령어

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

ubuntu 환경에서 다음과 같이 입력하여 도커 설치

```
sudo usermod -aG docker $USER
```

sudo권한 없이 docker를 사용하기 위해 user에게 docker 권한 부여
이후 재부팅 시켜야합니다.

B. Dockerfile 내용

```
FROM openjdk:11-jdk

ARG JAR_FILE=./build/libs/htwm-0.0.1-SNAPSHOT.jar

COPY ${JAR_FILE} app.jar

RUN mkdir /imagefiles

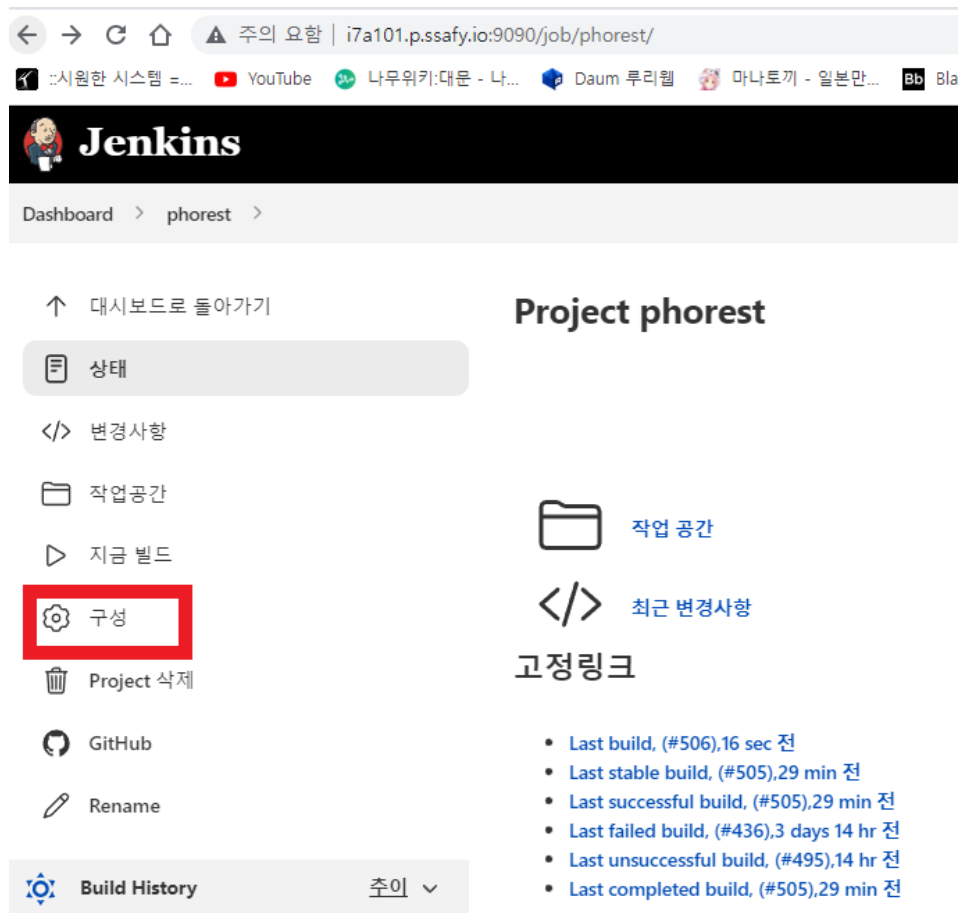
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Dockerfile은 back/htwm 폴더 안에 위치해야 합니다.

7. Jenkins 세팅

Jenkins를 이용해 CICD 환경을 구축, 개발 과정 중 약 280번의 빌드와 배포를 진행하였습니다.

A. Jenkins 구성



- EC2 인스턴스에 젠킨스를 설치한 후, 프로젝트를 만든 후 구성으로 들어갑니다.

Dashboard > phorest >

General소스 코드 관리빌드 유발빌드 환경Build빌드 후 조치

설명

[Plain text] [마리보기](#)

☒ GitHub project

Project url ?

https://lab.ssafy.com/s07-webmobile3-sub2/S07P12A101/

고급...

☐ 사용자 빌드 경로 사용 ?

GitLab Connection

☐ Use alternative credential

☐ Throttle builds ?

☐ 오래된 빌드 삭제 ?

저장

Apply

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

General **소스 코드 관리** 빌드 유발 빌드 환경 Build 빌드 후 조치

소스 코드 관리

☐ None

☒ Git ?

Repositories ?

Repository URL ? ✕

Credentials ?

 ▼

+ Add

고급...

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ? ✕

- 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 선택하는 곳입니다. 저희는 dev branch를 선택했습니다.

General 소스 코드 관리 **빌드 유발** 빌드 환경 Build 빌드 후 조치

빌드 유발

☐ 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://i7a101.p.ssafy.io:9090/project/phorest> ?

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

저장 Apply

- 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 ?

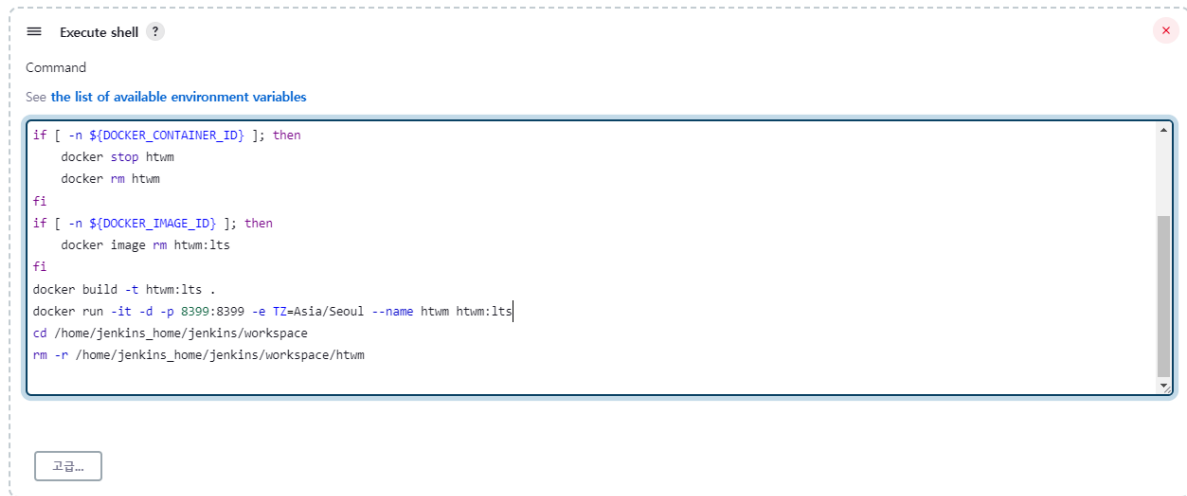
5ea7d4fe47f3aec75207399e3c842064

Generate

Clear

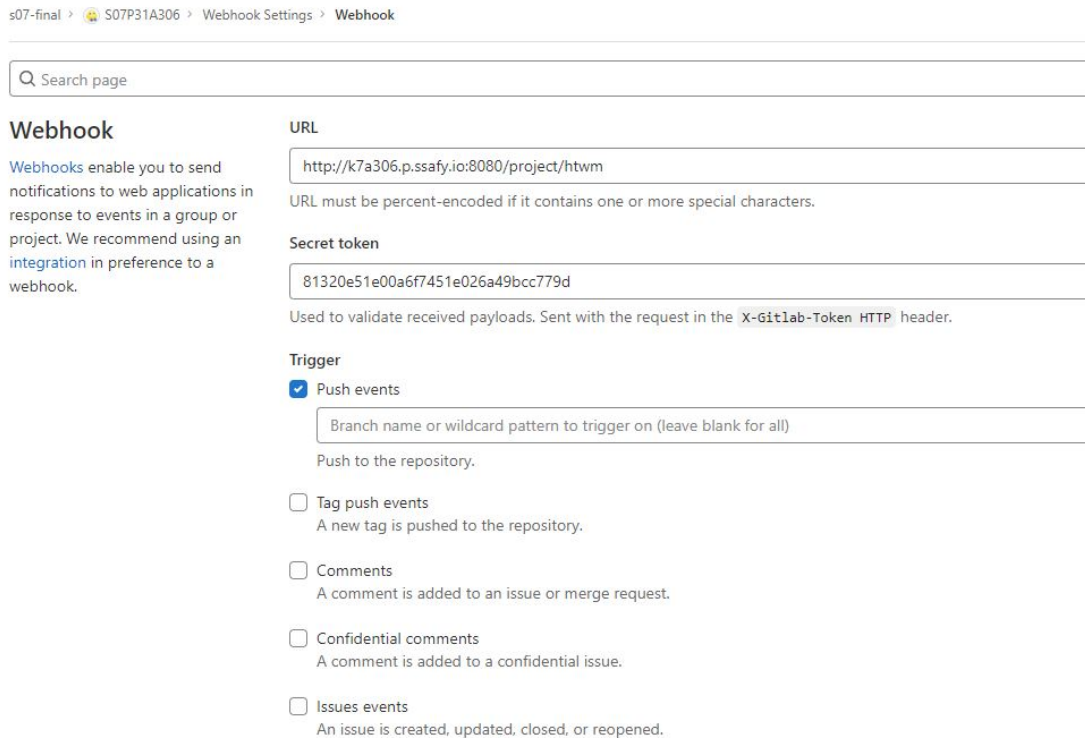
- 빌드 유발의 고급 탭을 눌러서 나오는 Secret token을 Generate 한 후, 이후 GitLab webhook 설정에 사용하였습니다.

Build Steps



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

B. GitLab 설정



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



참고 : <https://lemontia.tistory.com/882>

C. Jenkins 빌드, 배포 명령어

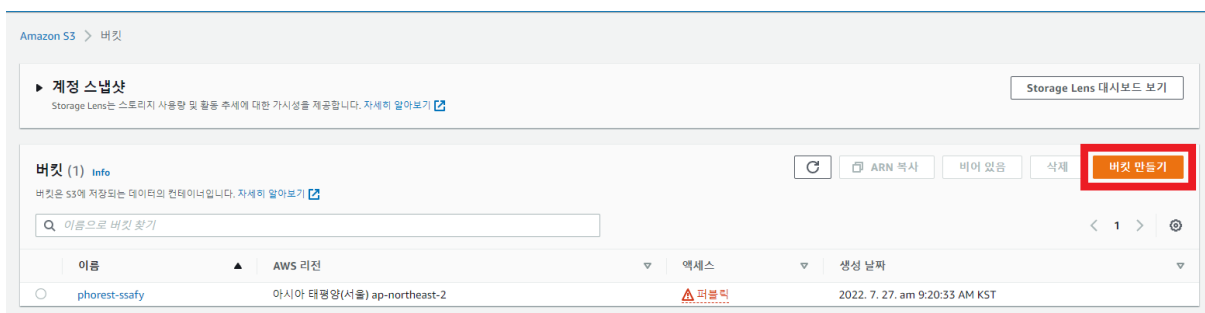
```
cd /home/jenkins_home/jenkins/workspace/htwm/back/htwm
chmod +x gradlew
cp /home/jenkins_home/jenkins/workspace/.keys/credentials.yaml src/main/resources/credentials.yaml
./gradlew clean build
DOCKER_IMAGE_ID=$(docker image ls -aq htwm:lts)
DOCKER_CONTAINER_ID=$(docker ps -aq -f name=htwm)
if [ -n ${DOCKER_CONTAINER_ID} ]; then
    docker stop htwm
    docker rm htwm
fi
if [ -n ${DOCKER_IMAGE_ID} ]; then
    docker image rm htwm:lts
fi
docker build -t htwm:lts .
docker run -it -d -p 8399:8399 -e TZ=Asia/Seoul --name htwm htwm:lts
cd /home/jenkins_home/jenkins/workspace
rm -r /home/jenkins_home/jenkins/workspace/htwm
```

8. 외부서비스

A. AWS S3

: AWS에서 제공하는 Cloud Simple Storage Service입니다. 서비스에서 발생하는 이미지, 동영상 파일들을 저장하기 위해 사용했습니다.

1. 버킷 생성



- 버킷 만들기 버튼을 눌러 새로운 버킷을 생성합니다.

2. 버킷 설정

버킷 만들기 Info

버킷은 S3에 저장되는 데이터의 컨테이너입니다. 자세히 알아보기 [\[링크\]](#)

일반 구성

버킷 이름

myawsbucket

버킷 이름은 전역에서 고유해야 하며 공백 또는 대문자를 포함할 수 없습니다. 버킷 이름 지정 규칙 참조 [\[링크\]](#)

AWS 리전

아시아 태평양(서울) ap-northeast-2 ▼

기존 버킷에서 설정 복사 - 선택 사항
다음 구성의 버킷 설정만 복사됩니다.

버킷 선택

객체 소유권 Info

다른 AWS 계정에서 이 버킷에 작성한 객체의 소유권 및 액세스 제어 목록(ACL)의 사용을 제어합니다. 객체 소유권은 객체에 대한 액세스를 지정할 수 있는 사용자를 결정합니다.

☐ ACL 비활성화됨(권장)

이 버킷의 모든 객체는 이 계정이 소유합니다. 이 버킷과 그 객체에 대한 액세스는 정책을 통해서만 지정됩니다.

☒ ACL 활성화됨

이 버킷의 객체는 다른 AWS 계정에서 소유할 수 있습니다. 이 버킷 및 객체에 대한 액세스는 ACL을 사용하여 지정할 수 있습니다.

객체 소유권

☐ 버킷 소유자 선호

이 버킷에 작성된 새 객체가 bucket-owner-full-control 삼입 ACL을 지정하는 경우 새 객체는 버킷 소유자가 소유합니다. 그렇지 않은 경우 객체 라이터가 소유합니다.

☒ 객체 라이터

객체 라이터는 객체 소유자로 유지됩니다.

- 버킷의 이름을 설정하고, 객체 소유권을 ACL 활성화됨으로 두고, 객체 라이터를 선택합니다.

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 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.

- ☐ **새 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.



모든 퍼블릭 액세스 차단을 비활성화하면 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있습니다.

정적 웹 사이트 호스팅과 같은 구체적으로 확인된 사용 사례에서 퍼블릭 액세스가 필요한 경우가 아니면 모든 퍼블릭 액세스 차단을 활성화하는 것이 좋습니다.



현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알고 있습니다.

- 모든 퍼블릭 액세스 차단을 체크 해제하고, 현재 설정으로 인해 이 버킷과 그 안에 포함된 객체가 퍼블릭 상태가 될 수 있음을 알 수 있습니다. 를 체크합니다.

3. 버킷 정책 설정

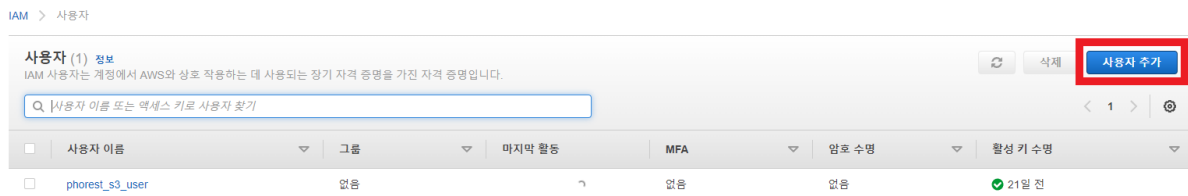
```
{
  "Version": "2012-10-17",
  "Id": "Policy1666336718066",
  "Statement": [
    {
      "Sid": "Stmt1666336709537",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::htm-ssafy"
    }
  ]
}
```

- 위와 같은 버킷 정책을 사용하였습니다.

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "ExposeHeaders": [
      "x-amz-server-side-encryption",
      "x-amz-request-id",
      "x-amz-id-2"
    ],
    "MaxAgeSeconds": 3000
  }
]
```

- 위와 같은 CORS 구성을 사용하였습니다.

4. AWS IAM 설정



- AWS IAM → 사용자 탭에서 사용자 추가를 누릅니다.

사용자 추가

1 2 3 4 5

사용자 세부 정보 설정

동일한 액세스 유형 및 권한을 사용하여 한 번에 여러 사용자를 추가할 수 있습니다. [자세히 알아보기](#)

사용자 이름*

[+ 다른 사용자 추가](#)

AWS 액세스 유형 선택

이러한 사용자가 주로 AWS에 액세스하는 방법을 선택합니다. 프로그래밍 방식의 액세스만 선택하면 사용자가 위임된 역할을 사용하여 콘솔에 액세스하는 것을 방지할 수 없습니다. 액세스 키와 자동 생성된 암호가 마지막 단계에서 제공됩니다. [자세히 알아보기](#)

AWS 자격 증명 유형 선택* ☒ 액세스 키 – 프로그래밍 방식 액세스

AWS API, CLI, SDK 및 기타 개발 도구에 대해 액세스 키 ID 및 비밀 액세스 키 을(를) 활성화합니다.

☐ 암호 – AWS 관리 콘솔 액세스


사용자가 AWS Management Console에 로그인할 수 있도록 허용하는 비밀번호 을(를) 활성화합니다.


- 임의의 사용자 이름을 넣고, AWS 자격 증명 유형은 액세스 키를 선택합니다.


사용자 추가

1 2 3 4 5

▼ 권한 설정

 그룹에 사용자 추가

 기존 사용자에서 권한 복사

 기존 정책 직접 연결

정책 생성



| 정책 필터 | | Q S3 | | 9 결과 표시 | |
|-------------------------------------|-----------------------------------------------------------|---------|------------------------|---------|--|
| | 정책 이름 | 유형 | 사용 용도 | | |
| <input type="checkbox"/> | ▶ AmazonDMSRedshiftS3Role | AWS 관리형 | 없음 | | |
| <input checked="" type="checkbox"/> | ▶ AmazonS3FullAccess | AWS 관리형 | Permissions policy (1) | | |
| <input type="checkbox"/> | ▶ AmazonS3ObjectLambdaExecutionRolePolicy | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ AmazonS3OutpostsFullAccess | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ AmazonS3OutpostsReadOnlyAccess | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ AmazonS3ReadOnlyAccess | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ AWSBackupServiceRolePolicyForS3Backup | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ AWSBackupServiceRolePolicyForS3Restore | AWS 관리형 | 없음 | | |
| <input type="checkbox"/> | ▶ QuickSightAccessForS3StorageManagementAnalyticsReadOnly | AWS 관리형 | 없음 | | |

▶ 권한 경계 설정

취소

이전

다음: 태그

- 권한 설정에서 기존 정책 직접 연결을 선택한 후, S3를 검색해서 AmazonS3FullAccess를 선택합니다.

| User name | Password | Access key ID | Secret access key | Console login link |
|-----------|----------|----------------------|------------------------------------------|----------------------------------------------------|
| ABCDEFGH | | AKIA3YQMVWHRBYT4ZQ5B | 4jcQrFFq3aHtiPi0vjhfzSNmBRoYJLbqOooXNSHb | https://808554181090.signin.aws.amazon.com/console |

- 이후 .csv파일의 키를 받아서, aws.yaml 파일의 cloud: credentials의 access-key와 secret-key에 입력해줍니다.



주의! 만약 Access Key와 Secret key 안에 /나 %가 포함되어 있다면 인식하지 못합니다. 만약 포함되어 있다면 포함되지 않을 때까지 IAM 계정을 새로 만들어야 합니다.

참고 : https://velog.io/@daydream_03/SignatureDoesNotMatch-오류-해결한-썰

9. Jetson Nano 포팅 매뉴얼

A. 실행 방법

```
$ cd openpose/myOpenPose
$ python3 TurnOn.py
```

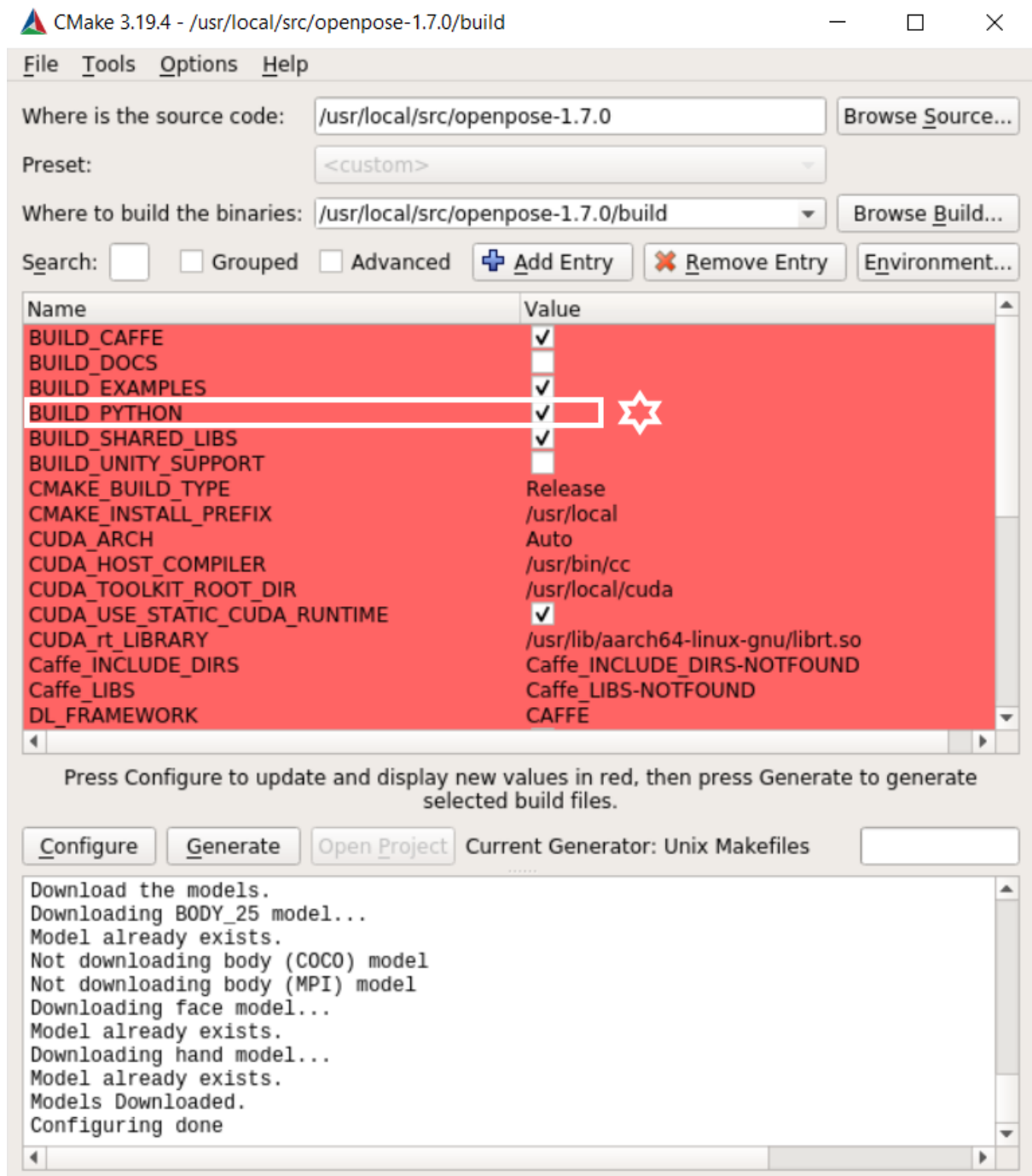
B. 실행 파일 별 설명

- TurnOn.py
 - 카메라 전환 및 주요 소켓 수신받는 로직
- Point.py
 - 하드코딩 관련한 변수들

- ExerciseDef.py
 - 운동 인식 (상단 카메라 : 5개, 하단 카메라 : 5개)
- pose.py
 - 소켓으로 수신받아서 운동 전환 및 운동 했을 시 소켓 송신 관련 py
- isExcute.sh
 - 프로세스 켜져 있는지 확인하는 sh파일
- PoseKill.sh
 - 프로세스에게 2의 신호를 보내주는 sh파일

C. cmake 빌드시 주의사항

- BUILD_PYTHON 클릭



- pyopenpose import error

```
$ cp -r (openpose 설치 디렉토리)/build/python/openpose/ /usr/lib/python-3.6/dist-packages
```

- GPU 사용 : BODY_25_MODEL 사용 (7~8 fps)
- CPU ONLY : MPI_MODEL 사용 (1~2 fps)