

## A101: *PhoRest*

삼성SW청년아카데미 서울캠퍼스 7기  
공통프로젝트 (6주, 2022/07/11 - 2022/08/19)

### 포팅 매뉴얼

담당 컨설턴트: 이상현  
최희선(팀장), 김준수, 김도현, 김보경, 윤희욱, 유현우



# PhoRest 포팅 매뉴얼

## 1. 프로젝트 기술 스택

- A. FE
- B. BE
- C. IOT

## 2. 빌드 방법

- A. 백엔드 빌드 방법
- B. 프론트엔드 빌드 방법
- C. 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. aws.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. 바뀔 내용
  - 5. 세팅 값 적용을 위한 재시작
  - 6. root 계정 외에 사용할 계정 생성
  - 7. 확인

## 6. Jenkins 세팅

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

## 7. 외부서비스

- A. 카카오
  - 1. 애플리케이션 추가
  - 2. 도메인 등록
  - 3. Redirect URI 설정
  - 4. 로그인 활성화
  - 5. 카카오 로그인 동의항목
  - 6. 카카오 인가코드 수신

## B. AWS S3

- 1. 버킷 생성
- 2. 버킷 설정
- 3. 버킷 정책 설정
- 4. AWS IAM 설정

## 8. IoT 포팅 매뉴얼

- A. 설치 라이브러리 (Qt & Raspberry Pi)
- B. PyQT 명령어 및 Raspberry Pi 설정 과정 및 명령어, 필수 디렉토리
- C. 화면 별 기능 상세 설명

# 1. 프로젝트 기술 스택

## A. FE

기술 스택(버전): React 18.2.0, react-dom 18.2.0, react-router-dom 6.3.0, reduxjs/toolkit 1.8.3, node package manager 8.11.0, axios 0.27.2, react-filerobot-image-editor 4.3.1 bootstrap 5.2.0, styled-components 5.3.5

사용 툴: Visual Studio Code 1.70.1

## B. 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

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

## C. IOT

기술 스택(버전): PyQt5 5.15.4, Python3 3.9.2, Raspberry Pi 4 Rasbian OS 64bit, OpenCV window : 4.6.0.66, RPI : 4.5.5, PIL window : 9.2.0, RPI : 9.2.0, moviepy window : 1.0.3, RPI : 1.0.3, requests window : 2.28.1 RPI : 2.25.1

사용 툴: PyCharm 2018ver, VNC Viewer, MobaXterm, Qt Designer 4.9.1

# 2. 빌드 방법

## A. 백엔드 빌드 방법

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

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

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

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/4.2.3/404cb5380b30d91e054610e4defe18a3333365db/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
2022-09-18 10:01:19.043 INFO 26828 --- [onShutdownHook] J.LocalContainerEntityManagerFactoryBean : Closing JPA
EntityManagerFactory for persistence unit 'default'
2022-09-18 10:01:19.047 INFO 26828 --- [onShutdownHook] com.zaxxer.hikari.HikariDataSource : HikariPool-1
- Shutdown initiated...
2022-09-18 10:01:19.107 INFO 26828 --- [onShutdownHook] 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, aws.yml 파일이 프로젝트 폴더 안의 BE\phorest\src\main\resources 폴더 안에 존재해야 한다.

## B. 프론트엔드 빌드 방법

1. Node.js 환경에서 FE\101 디렉토리로 이동
2. npm i 를 통해 package-lock.json에 정의된 패키지를 다운로드
3. 해당 폴더에서 아래의 명령어를 입력하여 배포 버전 파일 생성

```
npm run build
```

4. 101 디렉토리에 build 폴더가 생성됨
5. 생성된 build 폴더를 서버에 배포하여 사용

## C. 배포 명령어 정리

1. 현재 실행 중인 서버 pid 확인

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

현재 실행중인 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 phorest-0.0.1-SNAPSHOT.jar &
```

BE 빌드 과정에서 생성된 빌드 파일의 경로로 이동해서 서버를 실행시킨다.

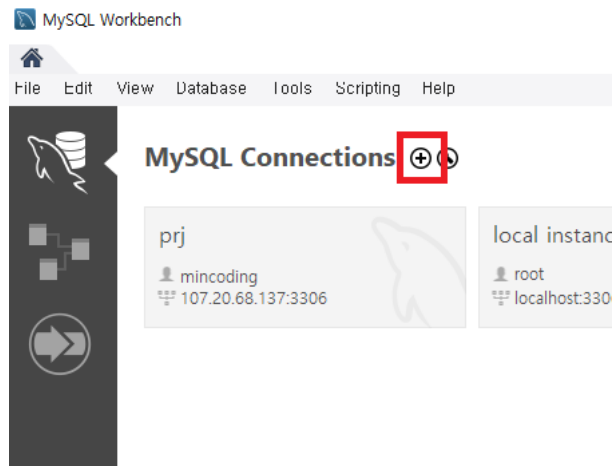
4. Nginx 재시작

```
sudo systemctl restart nginx
```

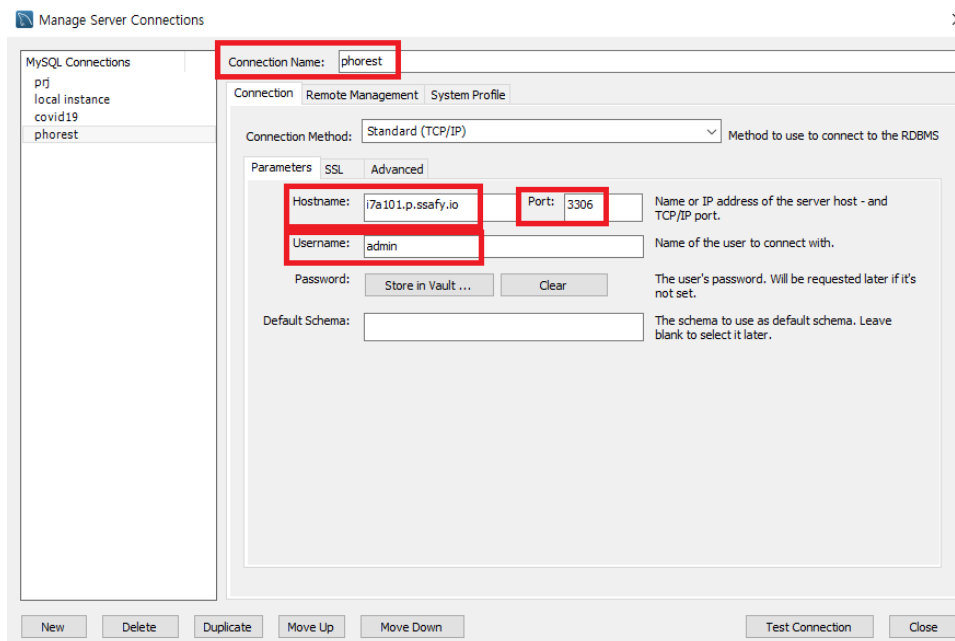
Nginx를 재시작한다.

### 3. DB 계정

#### A. MySQL WorkBench 추가하기



#### B. EC2 계정정보 넣기



- username : admin , password : Y).+]emWJXSMK[h()-9=[r0<Xw

기존 root 계정이 아닌 별도의 admin 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 {

    root /var/lib/jenkins/workspace/build;

    index index.html;

    server_name i7a101.p.ssafy.io phorest.site www.phorest.site;

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

    location /api {

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

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

    if ($host = www.phorest.site) {
        return 301 https://$host$request_uri;
    } # managed by Certbot

    if ($host = 3.36.53.251) {
        return 301 https://$host$request_uri;
    } # managed by Certbot
    server_name i7a101.p.ssafy.io phorest.site www.phorest.site # managed by Certbot
    return 404; # managed by Certbot

    listen 80;
    listen [::]:80;
}
```

## B. Git Ignore 파일

### 1. aws.yaml 파일

```
cloud:
  aws:
    stack:
      auto: false
    region:
      static: ap-northeast-2
    credentials:
      access-key: AKIA3YQMVWHRPATP4D5Y
      secret-key: h+KmncfvaMwwWfy2eS15UCdkEW54IKj+e9jIgoU
    s3:
      bucket: phorest-ssafy

jwt:
  header: Authorization
  secret: fdfb2373a3eaae03e0e453f83ac43c552d6f89bac3084b17310bd310236f98086c4c31e242383630913238c8330b93aa012f780756240aa8333a6ba369bc
```

```

token-validity-in-seconds: 86400

kakao:
  password: D08B1CAFC28DDCC04283A01536DF84562DFAFE0508A6A27CAEAB6DDFB1287015

sns:
  api-key: NCS4DZKWQJQ2ZJDQ
  api-secret-key: GDPMUFDYHGZ0VQOZWCBFJOI2TOHNPCUA
  domain: https://api.coolsms.co.kr

spring:
  datasource:
    url: jdbc:mysql://i7a101.p.ssafy.io:3306/admin
    username: admin
    password: Y}.+]emWJXSMK[h()-9=[r0<Xw
    driver-class-name: com.mysql.cj.jdbc.Driver

```

- aws.yaml은 프로젝트 폴더의 BE\phorest\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. 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;
```

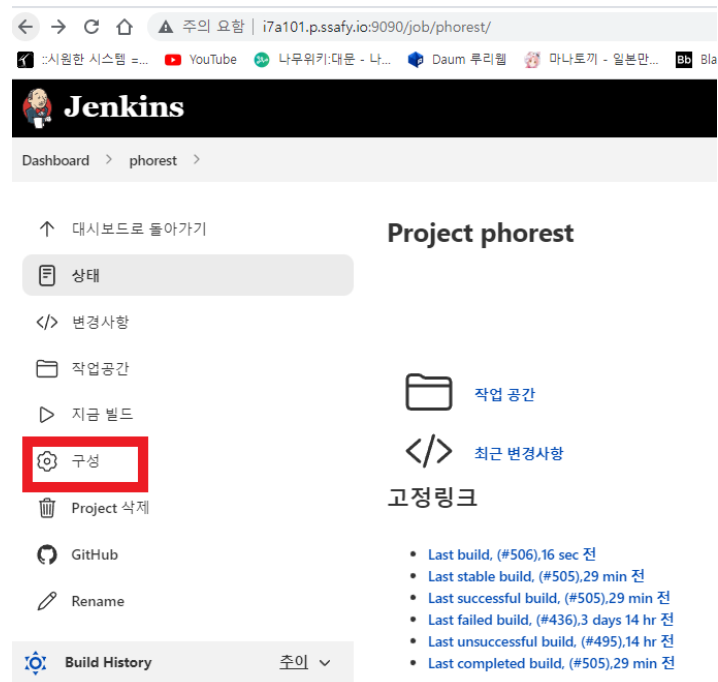
#### 7. 확인

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

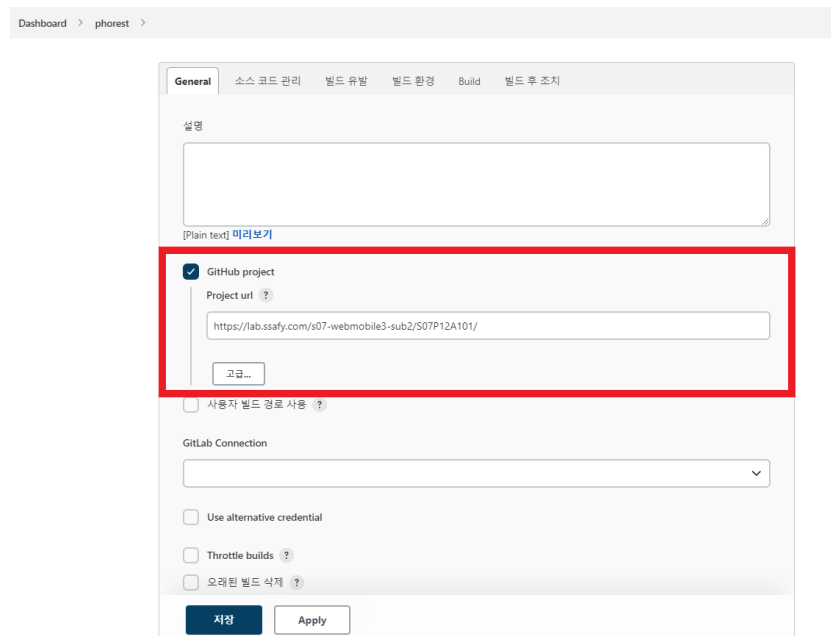
## 6. Jenkins 세팅

Jenkins를 이용해 CICD 환경을 구축, 개발 과정 중 약 500번의 빌드와 배포를 진행하였습니다.

### A. Jenkins 구성



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



- 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 선택 하는 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.

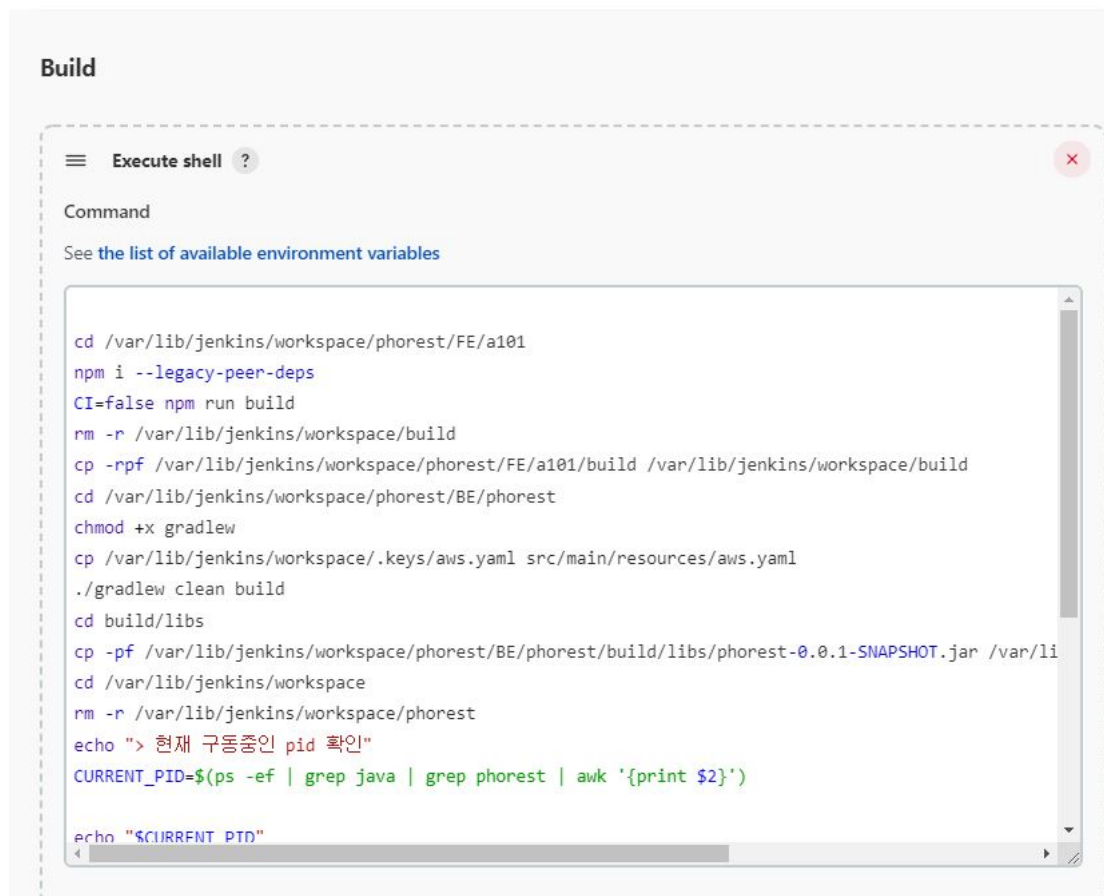
Secret token ?

5ea7d4fe47f3aec75207399e3c842064

Generate

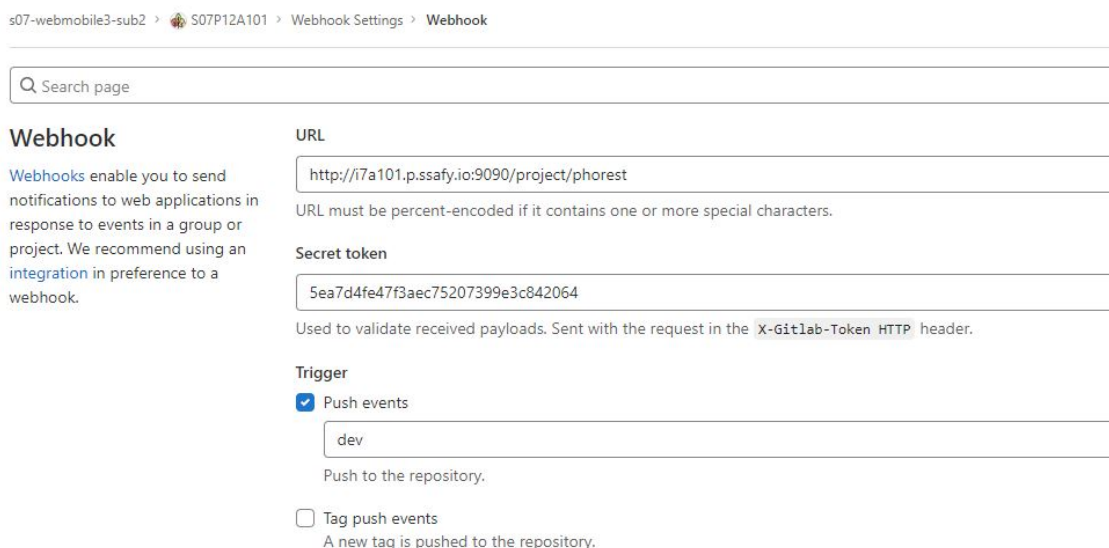
Clear

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



- Build 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 /var/lib/jenkins/workspace/phorest/FE/a101
npm i --legacy-peer-deps
CI=false npm run build
rm -r /var/lib/jenkins/workspace/build
cp -rpf /var/lib/jenkins/workspace/phorest/FE/a101/build /var/lib/jenkins/workspace/build
cd /var/lib/jenkins/workspace/phorest/BE/phorest
chmod +x gradlew
cp /var/lib/jenkins/workspace/.keys/aws.yml src/main/resources/aws.yml
./gradlew clean build
cd build/libs
cp -pf /var/lib/jenkins/workspace/phorest/BE/phorest/build/libs/phorest-0.0.1-SNAPSHOT.jar /var/lib/jenkins/workspace/phorest-0.0.1-SNAPSHOT.jar
cd /var/lib/jenkins/workspace
rm -r /var/lib/jenkins/workspace/phorest
echo "> 현재 구동중인 pid 확인"
CURRENT_PID=$(ps -ef | grep java | grep phorest | 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 phorest-0.0.1-SNAPSHOT.jar & echo $! > program.pid
```

## 7. 외부서비스

### A. 카카오

: 서비스의 회원가입/로그인, 메시지 보내기 기능을 카카오로 진행하였습니다. 서비스 기능에 집중할 수 있으며, 회원가입/로그인의 다양한 절차를 생략할 수 있어서 이용자 편의성을 제공합니다.

#### 1. 애플리케이션 추가

##### 기본 정보

앱 아이콘	
앱 이름	PhoRest
사업자명	a101

- 앱 이름 : PhoRest
- 사업자명: a101

#### 2. 도메인 등록

## Web

사이트 도메인	<a href="https://phorest.site">https://phorest.site</a> <a href="https://getpostman.com">https://getpostman.com</a> <a href="http://localhost:8399">http://localhost:8399</a> <a href="http://localhost:3000">http://localhost:3000</a>
---------	--

- 카카오 로그인 사용 시 Redirect URI를 등록해야 합니다. [등록하러 가기](#)

- 사이트 도메인 등록시 <https://getpostman.com> , <http://localhost:8399> , <http://localhost:3000> 은 로컬 환경에서 테스트 하기 위해 등록하였습니다.

### 3. Redirect URI 설정

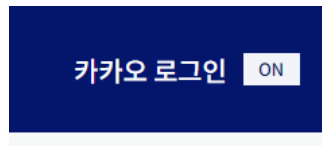
#### Redirect URI

Redirect URI	<a href="https://phorest.site/kakao/oauth">https://phorest.site/kakao/oauth</a> <a href="https://getpostman.com/oauth2/callback">https://getpostman.com/oauth2/callback</a> <a href="http://localhost:8399/api/user/kakao">http://localhost:8399/api/user/kakao</a> <a href="https://i7a101.p.ssafy.io/api/user/kakao">https://i7a101.p.ssafy.io/api/user/kakao</a> <a href="https://phorest.site/api/user/kakao">https://phorest.site/api/user/kakao</a> <a href="http://localhost:3000">http://localhost:3000</a> <a href="http://localhost:3000/kakao">http://localhost:3000/kakao</a> <a href="https://phorest.site">https://phorest.site</a> <a href="https://phorest.site/kakao">https://phorest.site/kakao</a>
--------------	---

- 카카오 로그인에서 사용할 OAuth Redirect URI를 설정합니다. (최대 10개)
- REST API로 개발하는 경우 필수로 설정해야 합니다.

- 나머지 Redirect URI는 테스트 하기 위해 등록하였습니다.

### 4. 로그인 활성화



#### 활성화 설정

상태 ☒ ON

### 5. 카카오 로그인 동의항목

#### 개인정보

항목 이름	ID	상태
닉네임	profile_nickname	● 필수 동의
프로필 사진	profile_image	● 필수 동의

## 접근권한

항목 이름	ID	상태
카카오스토리 글 목록	story_read	● 사용 안함
카카오스토리 글 작성	story_publish	● 사용 안함
카카오톡 메시지 전송	talk_message	● 선택 동의

- 닉네임, 프로필 사진, 카카오톡 메시지 전송 등의 항목을 다음과 같이 활성화 해줍니다.

## 6. 카카오 인가코드 수신

### 앱 키

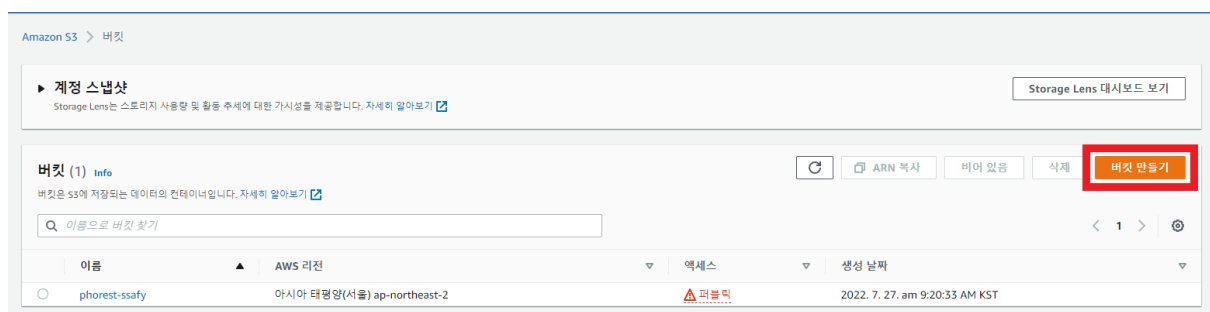
네이티브 앱 키	23f4359158c8215093bc5a5a95042170
REST API 키	4656da19556d6f608f3a297dd7c7b994
JavaScript 키	2ac98c5f57a9e7652f777219dbe795be
Admin 키	842bdacdc06e01822ae8aa449815a0ab

- KakaoService.java 내 getAccessToken 함수와 getToken 함수 내 client\_id 값에 해당 REST API 키를 설정해주어야 합니다.

## B. 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](#)

### 일반 구성

#### 버킷 이름

myawsbucket

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

#### 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 삽입 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": "Policy165888153451",
  "Statement": [
    {
      "Sid": "Stmt1658881534004",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::phorest-ssafy"
    }
  ]
}
```

- 위와 같은 버킷 정책을 사용하였습니다.

```
[
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "GET",
      "HEAD"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "ExposeHeaders": [
```



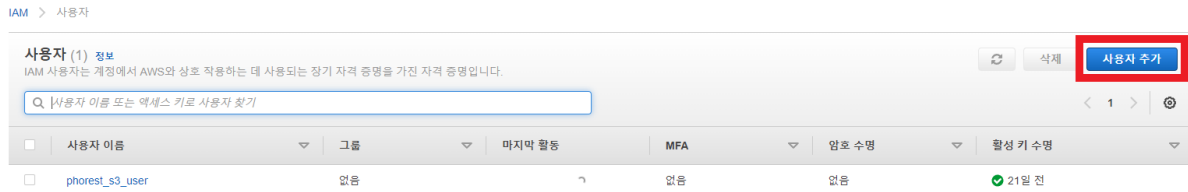
```

        "x-amz-server-side-encryption",
        "x-amz-request-id",
        "x-amz-id-2"
    ],
    "MaxAgeSeconds": 3000
}
1

```

- 위와 같은 CORS 구성을 사용하였습니다.

## 4. AWS IAM 설정



- AWS IAM → 사용자 탭에서 사용자 추가를 누릅니다.

### 사용자 추가



#### 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.

new\_user\_credentials.csv - Excel

	A	B	C	D	E	F	G	H	I	J
1	User name	Password	Access key ID	Secret access key	Console login link					
2	ABCDEF		AKIA3YQMVWHRBYT4ZQ5B	4jcQrFq3aHtiPi0vjhfzSNmBRoYjLbqOooXNSht	https://808554181090.signin.aws.amazon.com/console					
3										
4										

- 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-씩](https://velog.io/@daydream_03/SignatureDoesNotMatch-오류-해결한-씩)

## 8. IoT 포팅 매뉴얼

### A. 설치 라이브러리 (Qt & Raspberry Pi)

Pillow	9.2.0	9.2.0
PyQt5	5.15.4	▲ 5.15.7
PyQt5-Qt5	5.15.2	5.15.2
PyQt5-sip	12.11.0	12.11.0
PyQt5Designer	5.14.1	5.14.1
certifi	2022.6.15	2022.6.15
charset-normalizer	2.1.0	2.1.0
click	8.1.3	8.1.3
colorama	0.4.5	0.4.5
decorator	4.4.2	▲ 5.1.1
idna	3.3	3.3
imageio	2.21.0	▲ 2.21.1
imageio-ffmpeg	0.4.7	0.4.7
importlib-metadata	4.12.0	4.12.0
moviepy	1.0.3	1.0.3
numpy	1.21.6	▲ 1.23.2
opencv-python	4.6.0.66	4.6.0.66
pip	20.1.1	▲ 22.2.2
proglog	0.1.10	0.1.10
pyqt5-plugins	5.15.4.2.2	5.15.4.2.2
pyqt5-tools	5.15.4.3.2	5.15.4.3.2
python-dotenv	0.20.0	0.20.0
qrcode	7.3.1	7.3.1
qt5-applications	5.15.2.2.2	5.15.2.2.2
qt5-tools	5.15.2.1.2	5.15.2.1.2
requests	2.28.1	2.28.1
setuptools	47.1.0	▲ 65.0.2
tqdm	4.64.0	4.64.0
typing-extensions	4.3.0	4.3.0
urllib3	1.26.11	1.26.11
zipp	3.8.1	3.8.1

### B. PyQT 명령어 및 Raspberry Pi 설정 과정 및 명령어 , 필수 디렉토리

#### ▼ Ui to Py

```
$ pyuic5 -x Main_Ui.ui -o Main_Ui.py
```

#### ▼ 듀얼 모니터 설정 과정

```
$xinput --list
```

명령어로 연결된 디스플레이의 ID값 확인

```
$xinput --map-to-output {id} HDMI-{num}
```

디스플레이 ID 값과 라즈베리파이에서 연결된 HDMI 포트 번호를 입력하여 터치 영역을 조정

#### ▼ RPI 프린터 연결

1. CUPS를 설치하기 전, 라즈비안 패키지 목록을 최신화하고 업그레이드를 진행

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

## 2. CUPS를 설치

```
$sudo apt-get install cups
```

## 3. CUPS를 사용할 수 있는 권한을 사용자에게 부여함 (예시 사용자명 : pi)

```
$sudo usermod -a -G lpadmin pi
```

## 4. 프린터를 라즈베리파이에 연결

## 5. 네트워크 내 다른 컴퓨터가 CUPS의 컨트롤 패너에 접속할 수 있도록 설정

```
$sudo cupscctl --remote-any  
$sudo /etc/init.d/cups restart
```

## 6. 라즈베리파이와 같은 네트워크에 접속된 PC로 <http://라즈베리파이IP주소:631> 로 접속

## 7. 상단 메뉴 Administration → Add Printer → 사용자 이름 + 비밀번호 입력 → 본인의 프린터 선택 → continue → 프린터 제작사와 프린터 모델을 선택 → Add Printer

\*리눅스 프린터 드라이버는 매우 한정적!

\*가지고 있는 프린터가 없는 경우가 많음

## 8. 프린터 확인하기

```
$lpstat -p
```

본인이 설정한 프린터 이름으로 등록 됨을 확인. 프린터가 idle상태일 때 사용 가능

## 9. 프린트 하기

```
$lp -d 프린터이름 -n 출력수 이미지파일이름  
ex) $lp -d epson -n 3 Img.jpg
```

다양한 옵션을 확인하기 위해서는 다음 명령어 사용

```
$lp --help
```

## ▼ PhoRest 실행 Bash Shell 작성

라즈베리파이 부팅 시 자동 실행을 위해 설정

```
$sudo vi /etc/xdg/lxsession/LXDE-pi/autostart
```

마지막 줄에 실행할 스크립트를 작성

```
@sh /home/pi/스크립트이름.sh
```

\*절대 경로로 입력해야 실행됨

## 스크립트 작성

```
#!/bin/bash
cd /home/pi/Desktop/mainprogram/
python3 main.py
read reply
```

실행하고자 하는 파이썬 파일이 있는 경로로 이동해 파이썬 파일 실행

#### ▼ 필수 디렉토리 및 코드 설정

1. Run\_Camera 함수내 `self.cap = cv2.VideoCapture({연결한 카메라 포트 번호})`
2. 비디오가 저장될 `./video`
3. 사진이 저장될 `./photoDir`
4. 소리 출력을 위한 `./camera_sound.wav` 파일
5. Frame들이 저장될 `./Frame`
6. 폰트 적용에 필요한 `.ttf` 파일들

## C. 화면 별 기능 상세 설명

#### ▼ 초기 화면 (상부 모니터 / 하부 모니터)



**터치하여 PhoRest를 시작해 보세요**

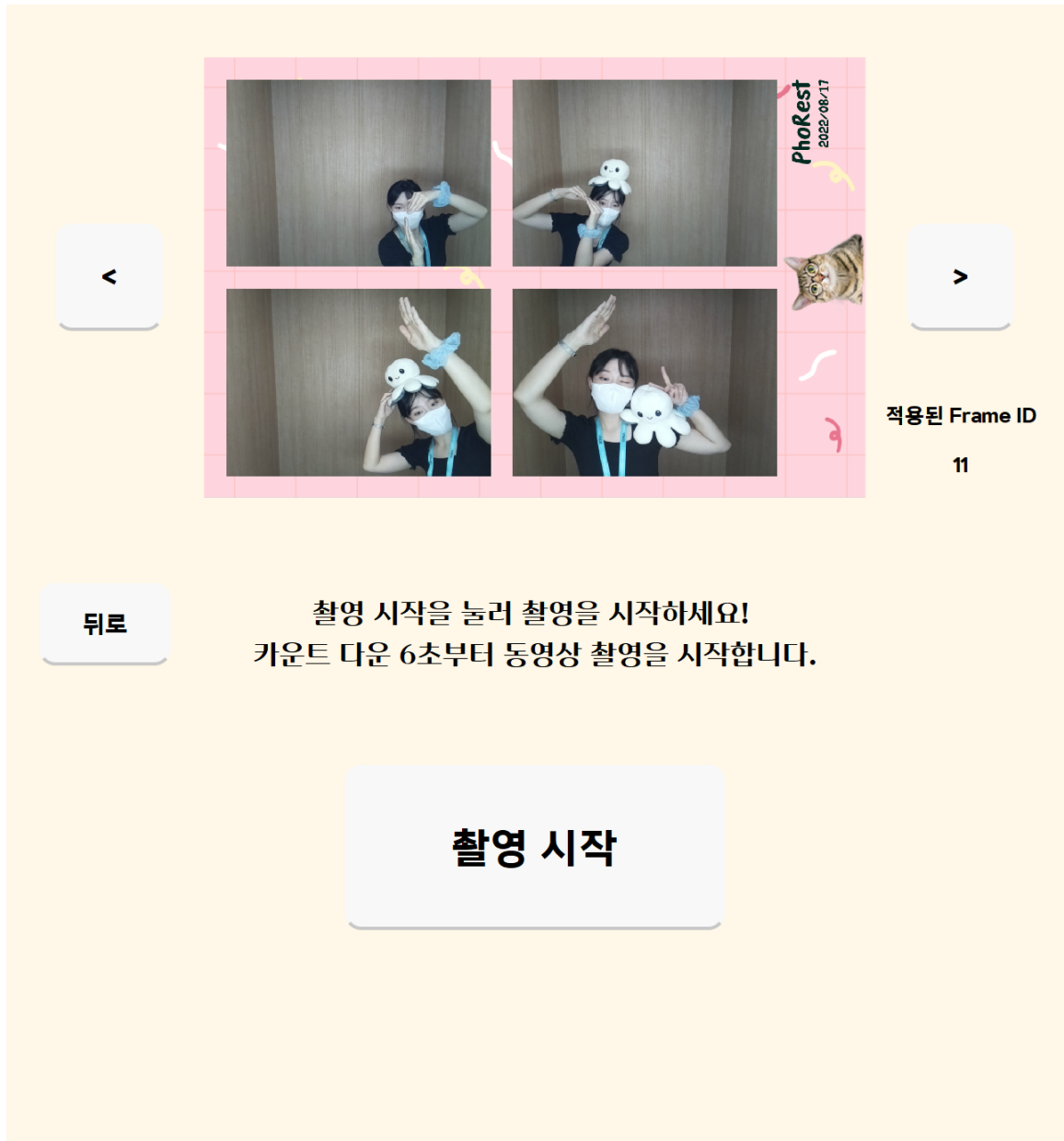
- 화면 어느 곳이든 터치하여 인원수 선택 화면으로 전환한다.

▼ 인원 수 선택 화면



- 1 ~ 6명 사이의 인원수를 선택하면 해당 인원수의 추천 포즈를 상단 화면을 통해 보여준다.
- 상단 화면에서 좌 / 우 버튼을 클릭하여 현재 DB에 올라와 있는 전체의 포즈를 탐색할 수 있다.
- 보여지는 추천 포즈의 프레임 번호를 상단 화면 우측 하단에 같이 보여줌으로써, 편리함을 높였다.
- 포즈는 홈페이지에서 좋아요를 많이 누른 순서대로 보여준다.
- 다음 버튼을 클릭하여 촬영 준비 화면으로 넘어갈 수 있다.  
이 때, 인원수를 선택하지 않으면 다음 버튼을 숨겨 인원수를 선택 해야지만 화면 전환을 할 수 있도록 만들어 주었다.

#### ▼ 촬영 준비 화면



- 이전에 선택했던 인원 수의 추천 포즈를 마지막으로 점검할 수 있다.
- 뒤로 버튼을 누르면 인원 선택 칸으로 돌아간다.
- 촬영 시작 버튼을 촬영 페이지로 넘어가게 된다.

#### ▼ 촬영 화면

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/823ace42-80dc-4294-b5e0-f455f2851323/MainWindow\\_2022-08-18\\_10-34-45.mp4](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/823ace42-80dc-4294-b5e0-f455f2851323/MainWindow_2022-08-18_10-34-45.mp4)

- 10초간의 준비시간 후에 사진이 촬영된다.  
이 때, 촬영이 됨과 동시에 연결한 스피커에서 “찰칵” 효과음을 내주며 사용자에게 촬영이 됐음을 알린다.



- 6초부터 동영상 촬영을 동시에 진행한다.
- 모든 촬영이 끝나면 사진 확대 페이지로 넘어가게 된다.

#### ▼ 촬영 사진 확대 화면

[https://s3-us-west-2.amazonaws.com/secure.notion-static.com/52fba1de-513a-4ea1-a152-7666e725caf8/MainWindow\\_2022-08-18\\_10-39-20.mp4](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/52fba1de-513a-4ea1-a152-7666e725caf8/MainWindow_2022-08-18_10-39-20.mp4)

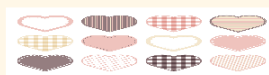
- 각 화면을 클릭하여 사진을 확대할 수 있다.
- 다음 버튼을 누르면 프레임 선택 화면으로 넘어가게 된다.

#### ▼ 프레임 선택 화면



사용할 프레임을 선택해주세요

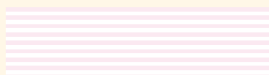
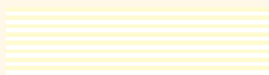
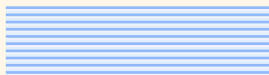
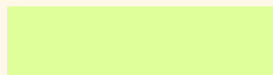
프린트



더 많은 프레임 적용하기



프레임 번호를 입력 후 적용 클릭




적용

지우기

- 8가지의 기본 프레임이 나타나고, 해당 프레임들을 선택하면 프레임이 적용된 사진의 모습이 상단 화면에 나타나게 된다.
- 기본 프레임 외의 사람들이 제작한 다른 프레임들을 보고 싶다면 QR코드를 통해 홈페이지로 들어가서 다양한 프레임들을 찾아볼 수 있다.

- 홈페이지에서 마음에 드는 프레임 번호를 우측 하단 입력창에 입력하고 숫자키패드("Enter") 혹은 적용 버튼을 클릭하면 해당 프레임 번호를 적용 시킨 이미지가 상단 화면에 나타난다.
- 프레임에서는 PhoRest와 현재 날짜가 같이 표시되는데, 이는 배경색의 보색을 찾아서 어떤 배경에서든 눈에 잘 띄게 보여준다.
- 다음 버튼을 클릭하여 프린트를 진행할 수 있다.

#### ▼ 최종 화면



- 화면이 전환됨과 동시에 프린트가 진행된다.
- 출력된 4컷 사진에는 QR코드가 삽입되어 출력되는데, 해당 QR코드로 들어가면 [phorest.site](http://phorest.site) 홈페이지로 들어가서 로그인을 진행하면 자동으로 커뮤니티에 업로드할 수 있으며 해당 사진의 원본 이미지 파일, 동영상 파일을 다운 받을 수 있다.
- 화면의 아무 곳이나 클릭하면 초기 화면으로 돌아간다.