

**알쏭달쏭**

**(ALSONG DALSONG)**

“감정 기반 음악 추천 일기 서비스”

**SSAFY 특화 프로젝트**

**포팅 매뉴얼**

- D204 -

**팀장 박주현**

**팀원 김효선 박소정 신혜연 조경수 홍석현**

# 1. 목 차

I. 프로젝트 기술 스택.....	3
II. 백엔드 빌드 방법.....	5
III. 프론트엔드 빌드 방법.....	6
IV. CI/CD 설정 및 명령어 정리.....	7
V. DB 설정.....	15
VI. Google Console 설정.....	17
VI. Kakao Console 설정.....	19
VII. AWS S3 사용 방법.....	21
VIII. 환경 변수 설정.....	29
IX. gitignore .....	30

## I. 프로젝트 기술 스택

### ● Backend

- ✓ python 3.7
- ✓ DjangoRESTframework
- ✓ Django 3.2.12
- ✓ DjangoRESTframework-simplejwt
- ✓ Oauth2
- ✓ Amazone S3

### ● FrontEnd

- ✓ React 18.2.0
- ✓ moment.js 2.29.4
- ✓ react-bootstrap 2.5.0
- ✓ react-router-dom 6.4.0
- ✓ react-dom 18.2.0
- ✓ styled-components 5.3.5

- DATA

- ✓ Jupyter notebook
- ✓ Python 3.7
- ✓ Django 3.2.12

- 배포

- ✓ AWS EC2
- ✓ Nginx
- ✓ Docker
- ✓ Jenkins

## II. 백엔드 빌드 방법

### 1. Django 프로젝트 폴더 이동

### 2. 가상환경 설정

`python -m venv <가상환경이름>`

-Windows : `source <가상환경이름>/Scripts/activate`

-Mac OS : `source <가상환경이름>/bin/activate`

### 3. 패키지 설치

`pip install requirements.txt`

### 4. Migration

`python manage.py makemigrations` (Migration 생성)

`python manage.py migrate` (Migration 적용)

### 5. 서버 구동

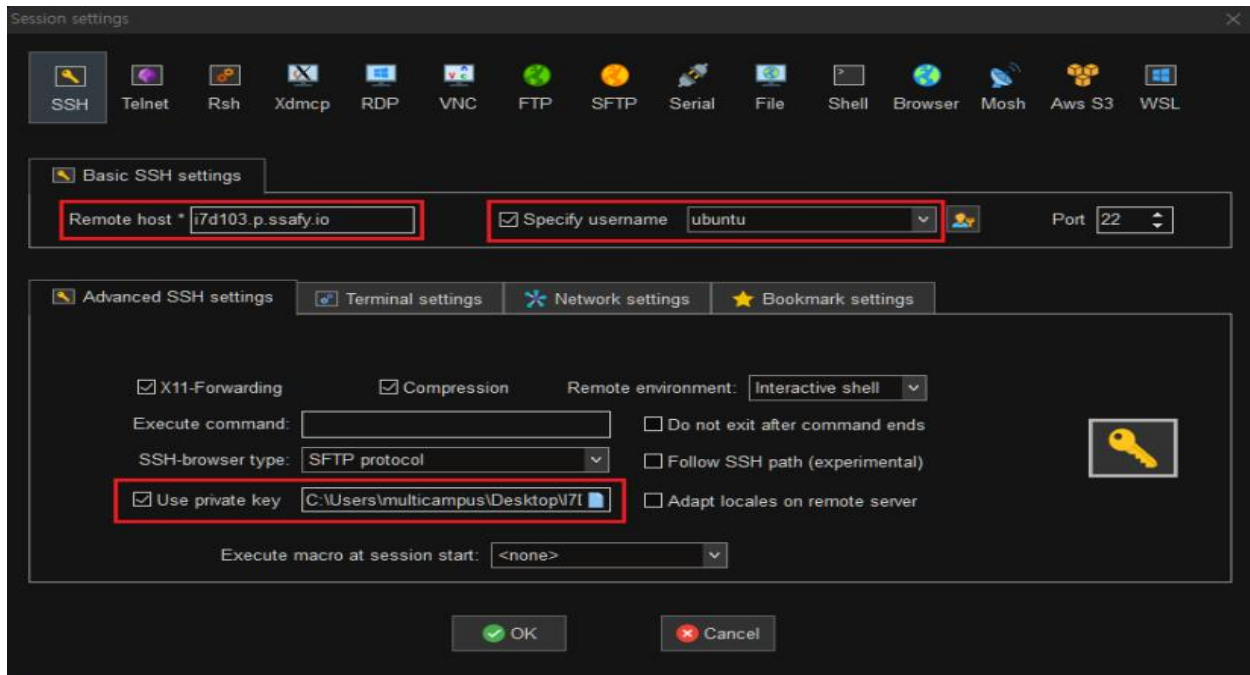
`python manage.py runserver`

### Ⅲ. 프론트엔드 빌드 방법

- npm install
- npm run build

## IV. CI/CD 설정 및 명령어 정리

### 1. EC2 접속을 위한 SSH 설정



- Remote host : AWS EC2 ip 주소 또는 연결된 도메인
- username : 사용자명
- Use private key 체크, 발급받은 key 파일 설정

### 2. Nginx 설치

```
- sudo apt-get update
- sudo apt-get upgrade
- sudo apt-get install nginx
```

### 3. docker 설치

```
# 필수 패키지 설치
- sudo apt-get install apt-transport-https ca-certificates
- curl gnupg-agent software-properties-common
# GPG Key 인증
- curl -fsSL https://download.docker.com/linux/ubuntu/gpg |
- sudo apt-key add -
# docker repository 등록
- sudo add-apt-repository "deb [arch=amd64]
- https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"
# 도커 설치
sudo apt-get update && sudo apt-get install docker-ce
docker-ce-cli containerd.io
# 도커 확인
sudo service docker status
```

### 4. Jenkins 컨테이너 실행

```
sudo docker run -d --name jenkins -u root --privileged \
-p '9090:8080' \
-v '/home/ubuntu/docker-volume/jenkins:/var/jenkins_home' \
-v '/var/run/docker.sock:/var/run/docker.sock' \
-v '/usr/bin/docker:/usr/bin/docker' \
jenkins/jenkins
```



## 5. Jenkins 실행 후 빌드 파라미터 등록

### Global properties

☐ Disable deferred wipeout on this node ?

☒ Environment variables

키-값 목록 ?

이름

env

값

REACT\_APP\_GOOGLE\_CLIENT\_ID="421385414738-hlk6fqfkbur8k03nuh1ffftukoo8umd.apps.googleusercontent.com" REACT\_APP\_REST\_API\_KEY=

## 6. Webhook 등록

### A. Jenkins

#### Build Triggers

☐ Build after other projects are built ?

☐ Build periodically ?

☐ Build when a change is pushed to BitBucket

☒ Build when a change is pushed to GitLab. GitLab webhook URL: `http://i7d103.p.ssafy.io:9090/project/test` ?

Enabled GitLab triggers

☒ Push Events

☐ Push Events in case of branch delete

### B. GitLab

s07-bigdata-recom-sub2 > S07P22D204 > Webhook Settings

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

`http://j7d204.p.ssafy.io:9090/project/ASDS/`

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

### Secret token

Used to validate received payloads. Sent with the request.

## 7. Jenkins pipeline script

```
pipeline {
    agent none
    stages {
        stage('Create .env') {
            agent any
            steps {
                sh 'echo "${env}" > .env'
                sh 'cat .env'
                sh 'ls -al'
                sh 'cp .env frontend/music-diary'

                sh 'rm -rf backend/.config_secrets'
                sh 'mkdir backend/.config_secrets'

                // sh 'rm secrets.json'
                sh 'echo "${secrets_json}" > secrets.json'
                sh 'cat secrets.json'
                sh 'ls -al'
                sh 'mv secrets.json backend/.config_secrets'

                // sh 'rm settings_common.json'
                sh 'echo "${settings_common}" > settings_common.json'
                sh 'cat settings_common.json'
                sh 'ls -al'
                sh 'mv settings_common.json backend/.config_secrets'
            }
        }

        stage('Docker build') {
            agent any
            steps {
                sh 'docker build -t backing ./backend'
                sh 'docker build -t frontimg ./frontend/music-diary'
```

```

        sh 'echo hello2'
    }
}
stage('Docker run') {
    agent any
    steps {
        sh 'docker ps -f name=front -q \
            | xargs --no-run-if-empty docker
container stop'

        sh 'docker ps -f name=back -q \
            | xargs --no-run-if-empty docker
container stop'

        sh 'docker container ls -a -f name=front -q \
            | xargs -r docker container rm'

        sh 'docker container ls -a -f name=back -q \
            | xargs -r docker container rm'

        sh 'docker run -d --name front -p 80:80
fronting'

        sh 'docker run -d --name back -p 8080:8080
backing'

        sh 'echo hello3'
    }
}
}
}
}

```

## 1) stage ("Create .env")

- A. Jenkins 에 등록된 환경변수를 사용해 .env 파일과 secrets.json 파일
- B. 각 파일을 frontend 디렉토리로 복사

## 2) stage (Docker build")

- A. backend 폴더에 있는 Dockerfile 을 backing 라는 이름으로 이미지화

B. frontend 폴더에 있는 Dockerfile frontimg 라는 이름으로 이미지화

### 3) stage("Docker run")

- A. front 라는 이름의 컨테이너가 있으면 stop
- B. back 이라는 이름의 컨테이너가 있으면 stop
- C. front 라는 이름의 컨테이너가 있으면 remove
- D. back 이라는 이름의 컨테이너가 있으면 remove
- E. 빌드된 frontimg 컨테이너를 80 포트에서 실행
- F. 빌드된 backimg 컨테이너를 8080 포트에서 실행

## 8. Dockerfile 설정

### 1. Backend

```
FROM python:3.7.0
COPY requirements.txt ./

RUN pip install --upgrade pip
RUN pip install -r requirements.txt
COPY . .
CMD ["python", "manage.py", "runserver", "0.0.0.0:8080"]
# sddd
```

- 패키지 설치
- Django 서버 8080 포트에서 실행

## 2. Frontend

```
PowerShell ▾  
  
FROM node:lts-alpine as build-stage  
WORKDIR /app  
COPY package*.json ./  
  
RUN npm install --force  
COPY . .  
RUN npm run build  
  
FROM nginx:stable-alpine as production-stage  
RUN rm /etc/nginx/conf.d/default.conf  
COPY ./nginx/deploy.conf /etc/nginx/conf.d/deploy.conf  
  
RUN rm -rf /usr/share/nginx/html/*  
COPY --from=build-stage /app/build /usr/share/nginx/html  
  
EXPOSE 80  
CMD ["nginx", "-g", "daemon off;"]
```

- /usr/share/nginx/html/\* 삭제
- build 폴더를 /usr/share/nginx/html 로 복사

## 3. nginx 세팅

- /etc/nginx/conf.d/default.conf 삭제
- ./nginx/deploy.conf 파일을 /etc/nginx/conf.d/deploy.conf 복사
- **deploy.conf**

```

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

    index index.html;
    proxy_hide_header Access-Control-Allow-Origin;

    add_header 'Access-Control-Allow-Origin' '*';

    server_name _;

    location / {
        root /usr/share/nginx/html;
        try_files $uri $uri/ /index.html;
    }

    location /rest {
        proxy_pass http://j7d204.p.ssafy.io:8080;
    }
}

```

- 80 포트를 기본 포트로 설정: 포트 입력 없으면 80 포트로 연결
- 기본 URI 에서 user/share/nginx/html 에 있는 index.html 을 전달
- CORS Header 추가
- 프록시 설정을 통해 /rest 로 시작하는 uri 를 8080 포트로 연결

## V. DB 설정

### 8. AWS RDS에서 엔진으로 MySQL 선택

asds

요약

DB 식별자 asds	CPU 2.46%	상태 🟢 사용 가능	클래스 db.t3.micro
역할 인스턴스	현재 활동 1 연결	엔진 MySQL Community	리전 및 AZ ap-northeast-2d

연결 & 보안 | 모니터링 | 로그 및 이벤트 | 구성 | 유지 관리 및 백업 | 태그

연결 & 보안

엔드포인트 및 포트 엔드포인트 asds.cspd92r3jqje.ap-northeast-2.rds.amazonaws.com 포트 3306	네트워킹 가용 영역 ap-northeast-2d VPC vpc-0e0826d83277b51c5	보안 VPC 보안 그룹 default (sg-0ca16011ef96c8daa) 🟢 활성화 퍼블릭 액세스 가능 예
---	--	---

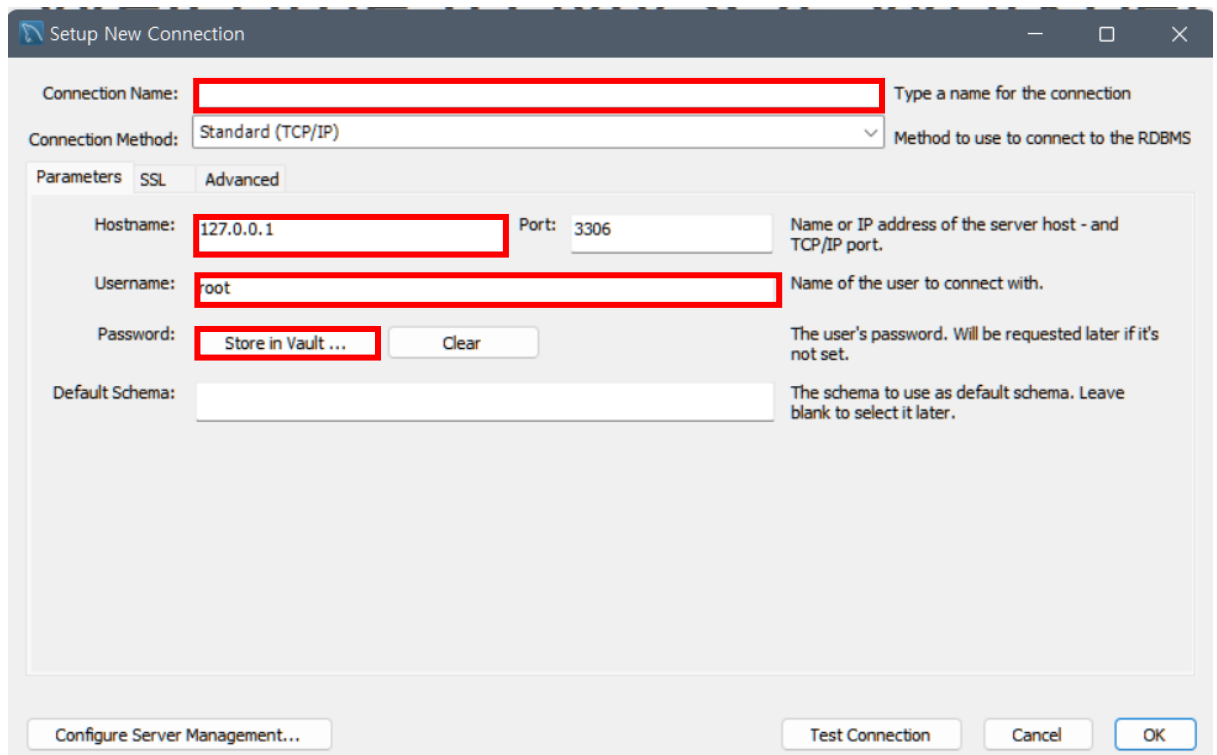
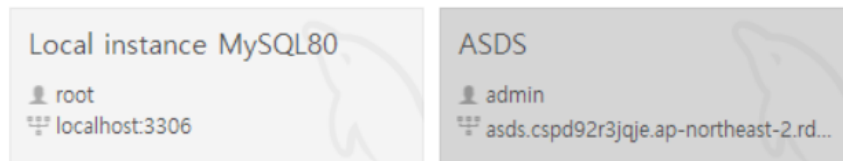
### 9. Django Projcet와 연동

- (장고에서 mysql 연결한 사진)

### 10.MySQL Workbench 설정

- AWS RDS에 연결

## MySQL Connections



The 'Setup New Connection' dialog box is shown. It has a title bar with standard window controls. The 'Connection Name' field is empty and highlighted with a red box. The 'Connection Method' is set to 'Standard (TCP/IP)'. The 'Parameters' tab is selected, showing fields for 'Hostname' (127.0.0.1), 'Port' (3306), 'Username' (root), 'Password' (Store in Vault ...), and 'Default Schema'. The 'Password' field is also highlighted with a red box. The 'Test Connection' button is highlighted with a red box. The 'OK' button is highlighted with a red box.

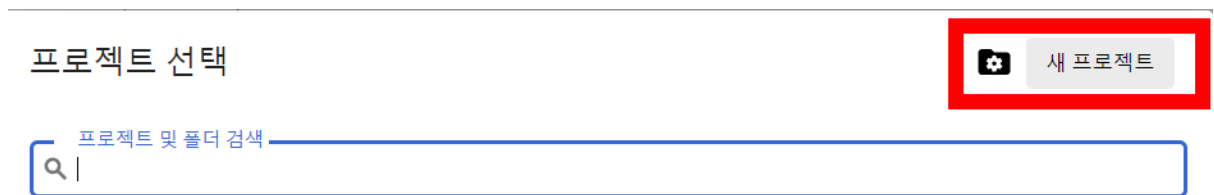
- Connection Name: ASDS
- Hostname: AWS RDS URL
- Username: 사용자 계정
- Password: 사용자 비밀번호



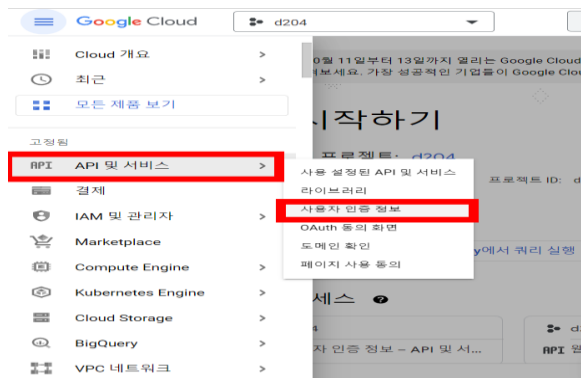
## VI. Google console 설정

1. <https://console.cloud.google.com/> 접속 및 로그인

2. 프로젝트 생성



3. 사용자 인증 정보 만들기



4. OAuth 클라이언트 ID 생성



## 5. 동의 화면 구성

## 6. OAuth 클라이언트 ID 설정

**RPI** API 및 서비스

[←](#) 웹 애플리케이션의 클라이언트 ID [JSON 다운로드](#) [↺](#) [보](#)

이름 \*

알쏭달쏭

OAuth 2.0 클라이언트의 이름입니다. 이 이름은 콘솔에서 클라이언트를 식별하는 용도로만 사용되며 최종 사용자에게 표시되지 않습니다.

아래에 추가한 URI의 도메인이 [승인된 도메인](#)으로 OAuth 동의 화면에 자동으로 추가됩니다.

승인된 자바스크립트 원본

브라우저 요청에 사용

URI 1 \*

<http://j7d204.p.ssafy.io>

승인된 리디렉션 URI

웹 서버의 요청에 사용

URI 1 \*

<http://j7d204.p.ssafy.io/google/login/callback/>

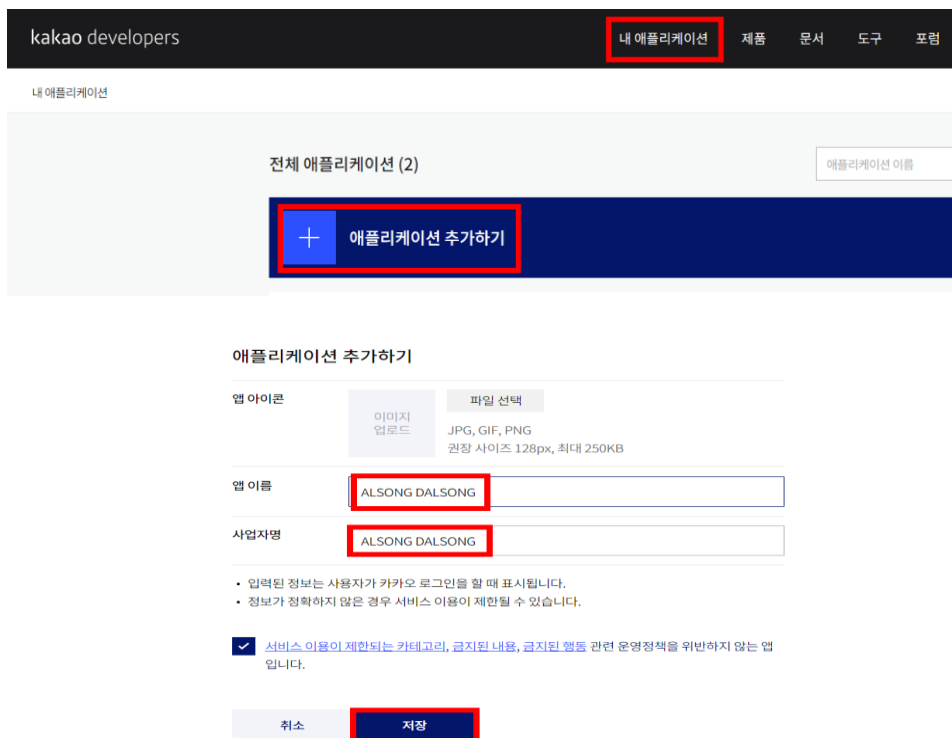
## 7. 클라이언트 ID & Secret key 발급

클라이언트 ID	401285414720- b1k6fgflchur9lk02pub1f4ftukce9ued.apps.googleusercontent.com
클라이언트 보안 비밀	C288PX1W1_5k_B1kkj731Mj_fSp8P...
생성일	2022년 9월 8일 AM 12시 23분 47초 GMT+9

## VI. Kakao console 설정

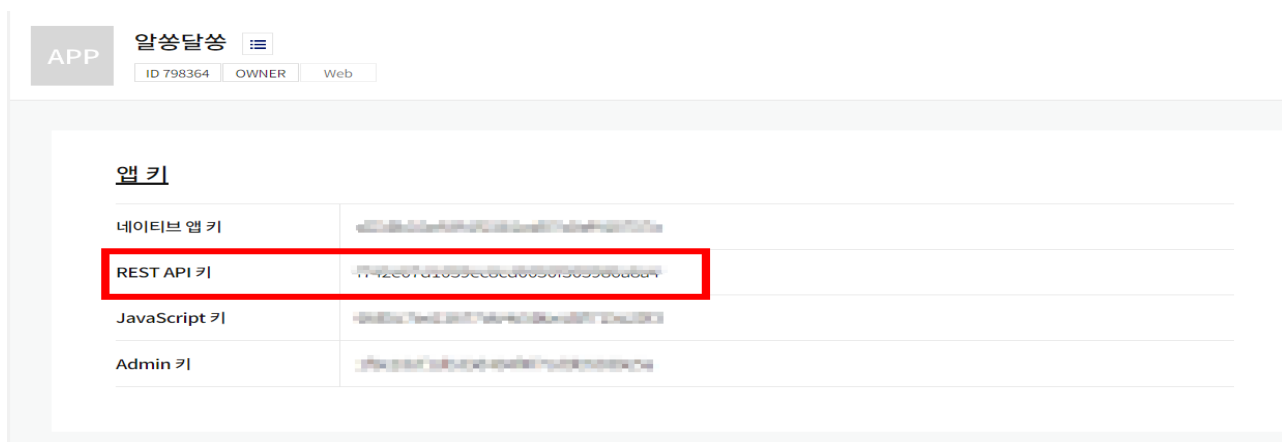
1. <https://developers.kakao.com/> 접속 및 로그인

2. 내 애플리케이션 > 애플리케이션 추가하기



The screenshot shows the Kakao Developers Console interface. At the top, the navigation bar includes 'kakao developers' and a menu with '내 애플리케이션' (My Applications), '제품' (Products), '문서' (Docs), '도구' (Tools), and '포럼' (Forum). The '내 애플리케이션' menu item is highlighted with a red box. Below the navigation bar, the page title is '내 애플리케이션' (My Applications). The main content area shows '전체 애플리케이션 (2)' (Total Applications (2)) and a button labeled '+ 애플리케이션 추가하기' (Add Application) with a plus icon, which is also highlighted with a red box. Below this, the '애플리케이션 추가하기' (Add Application) form is displayed. It includes fields for '앱 아이콘' (App Icon) with an '이미지 업로드' (Image Upload) button and '파일 선택' (File Select) button, and '앱 이름' (App Name) and '사업자명' (Business Name) fields, both containing 'ALSONG DALSONG' and highlighted with red boxes. There are also checkboxes for '서비스 이용이 제한되는 카테고리, 금지된 내용, 금지된 행동' (Categories, content, and actions restricted for service use) and a '저장' (Save) button highlighted with a red box.

3. REST API 키 발급



The screenshot shows the '앱 키' (App Key) section of the Kakao Developers Console. The page title is '앱 키' (App Key). The main content area shows a table with the following keys: '네이티브 앱 키' (Native App Key), 'REST API 키' (REST API Key), 'JavaScript 키' (JavaScript Key), and 'Admin 키' (Admin Key). The 'REST API 키' is highlighted with a red box. The 'REST API 키' value is a long alphanumeric string: '7742e0701053cc0c3d0501305900a0a7'.

## 4. 앱 설정 > 플랫폼 > Web > 도메인 등록

Web

삭제 수정

사이트 도메인	<a href="http://j7d204.p.ssafy.io:8080">http://j7d204.p.ssafy.io:8080</a> <a href="http://j7d204.p.ssafy.io">http://j7d204.p.ssafy.io</a> <a href="#">http://j7d204.p.ssafy.io:8080</a> <a href="#">http://j7d204.p.ssafy.io:8080</a> <a href="#">http://j7d204.p.ssafy.io:8080</a> <a href="#">http://j7d204.p.ssafy.io:8080</a>
---------	--

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

## 5. 제품 설정 > 카카오 로그인 활성화 & Redirect URI 설정

kakao developers

내 애플리케이션 제품

내 애플리케이션 > 제품 설정 > 카카오 로그인

앱 설정

요약 정보

일반

비즈니스

앱 키

플랫폼

팀 관리

제품 설정

카카오 로그인

동의 항목

간편가입

카카오톡 채널

개인정보 국외이전

연결 끊기

사용자 프로퍼티

보안

고급

메시지

카카오톡 채널

카카오 로그인 ON

동의 화면 미리보기

활성화 설정

상태 ON

카카오 로그인 API를 활용하면 사용자가 번거로운 회원가입 절차 대신, 카카오톡으로 서비스를 시작할 수 있습니다. 상태가 OFF일 때도 카카오 로그인 설정 항목을 변경하고 서버에 저장할 수 있습니다. 상태가 ON일 때만 실제 서비스에서 카카오 로그인 화면이 연결됩니다.

OpenID Connect 활성화 설정

상태 ON

카카오 로그인의 확장 기능인 OpenID Connect를 활성화합니다. 이 설정을 활성화하면 카카오 로그인 시 사용자 인증 정보가 담긴 ID 토큰을 액세스 토큰과 함께 발급받을 수 있습니다.

Redirect URI

삭제 수정

Redirect URI

<http://j7d204.p.ssafy.io/kakao/login/callback>

## 6. 카카오 로그인 동의 항목 설정

APP

알송달송

ID 798364 OWNER Web

카카오 로그인 ON

동의 화면 미리보기

동의 항목

카카오 로그인으로 서비스를 시작할 때 동의 받는 항목을 설정합니다. 미리 보기를 통해 사용자에게 보여질 화면을 확인할 수 있습니다. 사업자 정보를 등록하여 비즈 앱으로 전환하고 비즈니스 채널을 연결하면 권한이 없는 동의 항목에 대한 검증 신청을 할 수 있습니다.

비즈니스 설정 바로가기

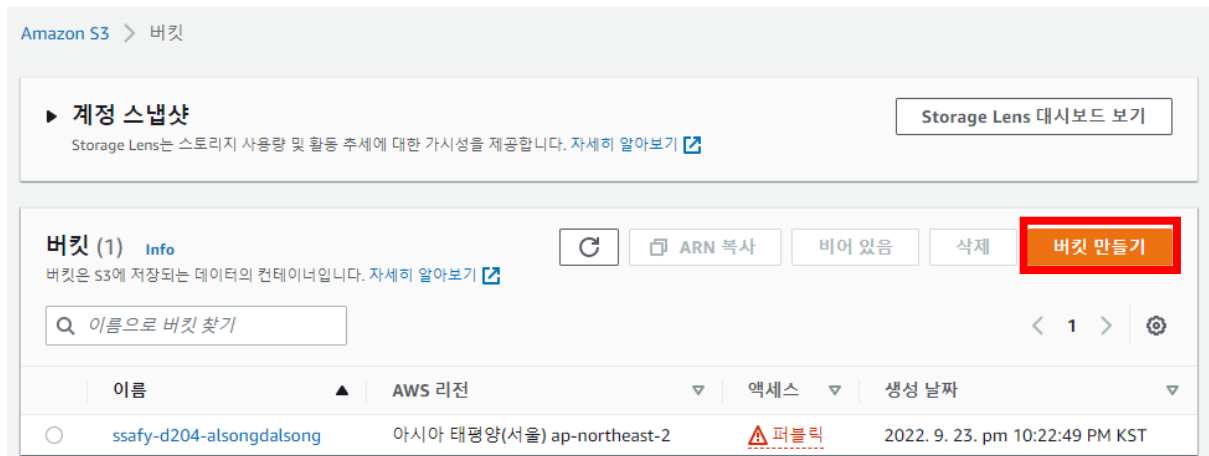
개인정보

항목 이름	ID	상태	
닉네임	profile_nickname	필수 동의	설정
프로필 사진	profile_image	필수 동의	설정
카카오계정(이메일)	account_email	선택 동의	설정

20

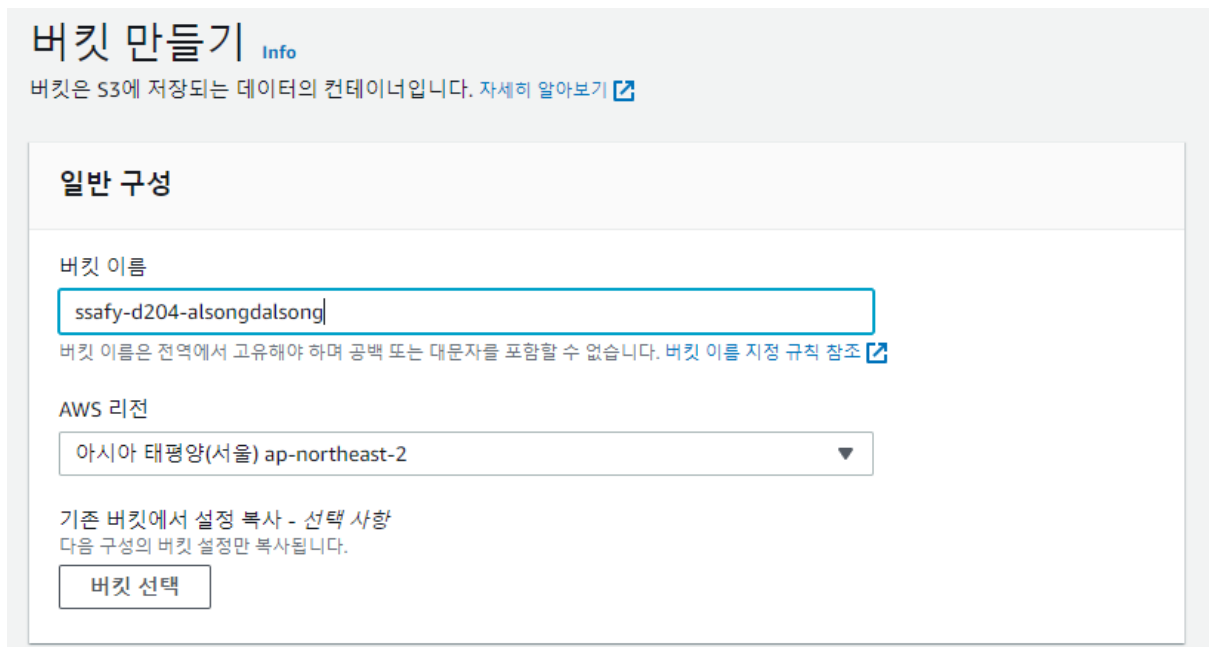
## VII. AWS S3 사용 방법

1. AWS 로그인 후 <https://s3.console.aws.amazon.com/s3/> 접속



2. 버킷 만들기

- A. AWS region을 서울로 지정한다.



3. 퍼블릭 액세스 차단 설정

## 이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지정 정책을 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 지정에만 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 지정 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지정 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.

#### ☒ 임의의 퍼블릭 버킷 또는 액세스 지정 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 지정에 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.

## 4. 권한 설정을 위해 <https://us-east-1.console.aws.amazon.com/iamv2> 접속

IAM > 사용자

사용자 (1) 정보

IAM 사용자는 계정에서 AWS와 상호 작용하는 데 사용되는 장기 자격 증명을 가진 자격 증명입니다.

삭제

사용자 추가

Q 사용자 이름 또는 액세스 키로 사용자 찾기

< 1 >

⚙

<input type="checkbox"/>	사용자 이름	그룹	마지막 활동	MFA	암호 수명	활성 키 수명
<input type="checkbox"/>	KHyoseon	alsongdalsong-group	3시간 전	없음	없음	12일 전

## 5. 사용자 추가

### 사용자 세부 정보 설정

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

사용자 이름\* KHyoseon2

[다른 사용자 추가](#)

### AWS 액세스 유형 선택

이러한 사용자가 주로 AWS에 액세스하는 방법을 선택합니다. 프로그래밍 방식의 액세스만 선택하면 사용자가 위임된 역할을 사용하여 콘솔에 액세스할 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.

그룹 생성

그룹을 만들고 그룹에 연결할 정책을 선택하십시오. 그룹을 사용하여 직무, AWS 서비스 액세스 또는 사용자 지정 권한별로 사용자의 권한을 관리하는 것이 좋습니다. [자세히 알아보기](#)

그룹 이름

alsongdalsong-group2

정책 생성

새로 고침

정책 필터

Q s3

9 결과 표시

	정책 이름	유형	사용 용도	설명
<input type="checkbox"/>	AmazonDMSRedshi...	AWS 관리형	없음	Provides access to manage S3 settings f...
<input type="checkbox"/>	AmazonS3FullAccess	AWS 관리형	Permissions policy (1)	Provides full access to all buckets via th...
<input type="checkbox"/>	AmazonS3ObjectLa...	AWS 관리형	없음	Provides AWS Lambda functions permis...
<input type="checkbox"/>	AmazonS3Outposts...	AWS 관리형	없음	Provides full access to Amazon S3 on O...
<input type="checkbox"/>	AmazonS3Outposts...	AWS 관리형	없음	Provides read only access to Amazon S...
<input type="checkbox"/>	AmazonS3ReadOnl...	AWS 관리형	없음	Provides read only access to all buckets ...
<input type="checkbox"/>	AWSBackupService...	AWS 관리형	없음	Policy containing permissions necessary...
<input type="checkbox"/>	AWSBackupService...	AWS 관리형	없음	Policy containing permissions necessary...
<input type="checkbox"/>	QuickSightAccessF...	AWS 관리형	없음	Policy used by QuickSight team to acces...

취소

그룹 생성

## 그룹에 사용자 추가

그룹 생성

새로 고침

Q 검색

1 결과 표시

그룹	연결된 정책
<input checked="" type="checkbox"/> alsongdalsong-group	AmazonS3FullAccess

## ▶ 권한 경계 설정

취소

이전

다음: 태그

## 태그 추가(선택 사항)

IAM 태그는 사용자 사용자에게 추가할 수 있는 키-값 페어입니다. 태그는 이메일 주소와 같은 사용자 정보를 포함하거나 정책과 같은 내용일 수 있습니다. 태그를 사용하여 이 사용자에게 대한 액세스를 구성, 추적 또는 제어할 수 있습니다. [자세히 알아보기](#)

키	값(선택 사항)	제거
<input type="text" value="새 키 추가"/>	<input type="text"/>	

50 태그를 더 추가할 수 있습니다.

[취소](#)[이전](#)[다음: 검토](#)

## 검토

선택 항목을 검토합니다. 사용자를 생성한 후 자동으로 생성된 비밀번호와 액세스 키를 보고 다운로드할 수 있습니다.

### 사용자 세부 정보

사용자 이름	KHyoseon2
AWS 액세스 유형	프로그래밍 방식 액세스 - 액세스 키 사용
권한 경계	권한 경계가 설정되지 않았습니다

### 권한 요약

위에 표시된 사용자를 다음 그룹에 추가합니다.

유형	이름
그룹	alsongdalsong-group

### 태그

태그가 추가되지 않았습니다.

[취소](#)[이전](#)[사용자 만들기](#)

## 사용자 추가

[1](#)[2](#)[3](#)[4](#)[5](#)

### 성공

아래에 표시된 사용자를 생성했습니다. 사용자 보안 자격 증명을 보고 다운로드할 수 있습니다. AWS Management Console 로그인에 필요한 사용자 지침을 이메일로 보낼 수도 있습니다. 지금이 이 자격 증명을 다운로드할 수 있는 마지막 기회입니다. 하지만 언제든지 새 자격 증명을 생성할 수 있습니다.

AWS Management Console 액세스 권한이 있는 사용자가 <https://987941189838.signin.aws.amazon.com/console>에 로그인할 수 있습니다.

[.csv 다운로드](#)

	사용자	액세스 키 ID	비밀 액세스 키
▶	✓ KHyoseon2	AKIA6MBPCHDHABXAF7EJ	***** <a href="#">표시</a>

위 과정에서 얻은 액세스 키와 비밀 액세스 키는 반드시 csv로 다운받거나 따로 보관을 해 두어야 한다.



## 6. 버킷 정책 설정

```
{
  "Version": "2012-10-17",
  "Id": "Policy1663949340416",
  "Statement": [
    {
      "Sid": "Stmt1663949279832",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::[redacted]:user/KHyoseon"
      },
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::ssafy-d204-alsongdalsong"
    },
    {
      "Sid": "Stmt1663949334249",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject"
      ],
      "Resource": "arn:aws:s3:::ssafy-d204-alsongdalsong/*"
    }
  ]
}
```

(첫번째 statement: KHyoseon이라는 사용자에게 모든 권한을 허용, 두번째 statement: 모든 사용자에게 객체 삭제와 조회 권한 허용)

버킷과 사용자 arn 주소를 맞게 입력한다.

## 7. 프로젝트 가상환경에 boto3 설치

Pip install boto3

## 8. Secrets.json 생성

```
{
  "aws": {
    "access_key_id": "[redacted]",
    "secret_access_key": "[redacted]",
    "storage_bucket_name": "ssafy-d204-alsongdalsong",
    "region": "ap-northeast-2"
  }
}
```

## 9. Settings.py에 필수 변수 지정

```
BASE_DIR = Path(__file__).resolve().parent
ROOT_DIR = os.path.dirname(BASE_DIR)
CONFIG_SECRET_DIR = os.path.join(ROOT_DIR, '.config_secrets')

CONFIG_SETTINGS_COMMON_FILE = os.path.join(CONFIG_SECRET_DIR, 'settings_common.json')
config_secret = json.loads(open(CONFIG_SETTINGS_COMMON_FILE).read())

# AES key
SECRET_KEY = config_secret['aes']['secret_key']

# AWS Access
AWS_ACCESS_KEY_ID = config_secret['aws']['access_key_id']
AWS_SECRET_ACCESS_KEY = config_secret['aws']['secret_access_key']
AWS_STORAGE_BUCKET_NAME = config_secret['aws']['storage_bucket_name']
AWS_REGION = config_secret['aws']['region']

# S3 Storages
AWS_S3_CUSTOM_DOMAIN = '%s.s3.%s.amazonaws.com' % (AWS_STORAGE_BUCKET_NAME, AWS_REGION)
AWS_S3_OBJECT_PARAMETERS = {
    'CacheControl': 'max-age=86400',
}
DEFAULT_FILE_STORAGE = 'storages.backends.s3boto3.S3Boto3Storage'
MEDIA_ROOT = os.path.join(BASE_DIR, 'path/to/store/my/files/')
```

## 10. S3 사용하는 앱에 storages.py 파일 추가

```
import boto3
import uuid
from django.conf import settings

class FileUpload:
    def __init__(self, client):
        self.client = client

    def upload(self, file):
        return self.client.upload(file)

    def delete(self, image_id):
        return self.client.delete(image_id)

class MyS3Client:
    def __init__(self, access_key, secret_key, bucket_name):
        boto3_s3 = boto3.client(
            's3',
```

```

        aws_access_key_id = access_key,
        aws_secret_access_key = secret_key
    )
    self.s3_client = boto3_s3
    self.bucket_name = bucket_name

def upload(self, file):
    try:
        file_id = str(uuid.uuid4())
        extra_args = { 'ContentType' : file.content_type }

        self.s3_client.upload_fileobj(
            file,
            self.bucket_name,
            file_id,
            ExtraArgs = extra_args
        )
        return settings.AWS_S3_CUSTOM_DOMAIN + "/" + file_id
    except:
        return None

def delete(self, image_id):
    try:
        self.s3_client.delete_object(Bucket=self.bucket_name, Key=image_id)
        return "SUCCESS"
    except:
        return "FAIL"

# MyS3Client instance
s3_client = MyS3Client(settings.AWS_ACCESS_KEY_ID,
settings.AWS_SECRET_ACCESS_KEY, settings.AWS_STORAGE_BUCKET_NAME)

```

## 11. Views.py 사용

```

from .storages import FileUpload, s3_client

class ImageDetail(GenericAPIView):
    serializer_class = ImageSerializer
    renderer_classes = (renderers.JSONRenderer,)

    def post(self, request, format=None):
        file = request.FILES['image']
        profile_image_url = FileUpload(s3_client).upload(file)

```

```

        if profile_image_url != None:
            return Response(profile_image_url, status=status.HTTP_200_OK)
        return Response(status=status.HTTP_500_INTERNAL_SERVER_ERROR)

@swagger_auto_schema(request_body=ImageSerializer)
def delete(self, request, format=None):
    image_url = request.data['image_url']
    image_id = image_url.split('com/')[1]
    ret = FileUpload(s3_client).delete(image_id)
    if ret=="SUCCESS":
        return Response({'result': ret}, status=status.HTTP_204_NO_CONTENT)
    return Response({'result': ret},
status=status.HTTP_500_INTERNAL_SERVER_ERROR)

```

## VIII. 환경 변수 설정

- Backend

```
# backend/.config_secrets/
```

- **secrets.json**

```
{ } secrets.json > {} secrets.json > ...  
1 {  
2   "KAKAO_REST_API_KEY": "  
3  
4   "SECRET_KEY": "  
5   "SOCIAL_AUTH_GOOGLE_CLIENT_ID": "  
6   "SOCIAL_AUTH_GOOGLE_SECRET": "  
7   "STATE": "  
8   "DATABASES": {  
9     "default": {  
10       "ENGINE": "django.db.backends.mysql",  
11       "NAME": "testDB",  
12       "USER": "admin",  
13       "PASSWORD": "  
14       "HOST": "asds.cspd92r3jaje.ap-northeast-2.rds.amazonaws.com",  
15       "PORT": "3306"  
16     }  
17   }  
18 }  
19
```

- **settings\_common.json**

```
{ settings_common.json X  
└─ .config_secrets > {} settings_common.json > ...  
1 {  
2   "aes": {  
3     "secret_key": "[REDACTED]",  
4   },  
5   "aws": {  
6     "access_key_id": "[REDACTED]",  
7     "secret_access_key": "[REDACTED]",  
8     "storage_bucket_name": "ssafy-d204-alsongdalsong",  
9     "region": "ap-northeast-2"  
10  }  
11 }  
12
```

## ● FrontEnd

```
# music-diary/.env
```

```
.env
1 REACT_APP_GOOGLE_CLIENT_ID='...'
2 REACT_APP_REST_API_KEY='...'
```

# IX. gitignore

## ● BackEnd

```
.gitignore
1 # Created by https://www.toptal.com/developers/gitignore/api/django,python,windows,macos,visualstudiocode
2 # Edit at https://www.toptal.com/developers/gitignore/templates-django,python,windows,macos,visualstudiocode
3
4 ### Django ###
5 *.log
6 *.pot
7 *.pyc
8 __pycache__/
9 local_settings.py
10 db.sqlite3
11 db.sqlite3-journal
12 media
13 .config_secrets
14
15 # If your build process includes running collectstatic, then you probably don't need or want to include staticfiles/
16 # in your Git repository. Update and uncomment the following line accordingly.
17 # <django-project-name>/staticfiles/
18
19 ### Django.Python Stack ###
20 # Byte-compiled / optimized / DLL files
21 *.py[co]
22 *$py.class
23
24 # C extensions
25 *.so
26
27 # Distribution / packaging
28 .Python
29 build/
30 develop-eggs/
31 dist/
32 downloads/
33 eggs/
34 .eggs/
35 lib/
36 lib64/
37 parts/
38 sdist/
39 var/
40 wheels/
41 share/python-wheels/
42 *.egg-info/
43 .installed.cfg
44 *.egg
45 MANIFEST
46
47 # PyInstaller
48 # Usually these files are written by a python script from a template
49 # before PyInstaller builds the exe, so as to inject date/other infos into it.
50 *.manifest
51 *.spec
52
53 # Installer logs
54 pip-log.txt
55 pip-delete-this-directory.txt
56
57 # Unit test / coverage reports
58 htmlcov/
59 # Jupyter Notebook
60 .ipynb_checkpoints
61
62 # IPython
63 profile.default/
64 ipython_config.py
65
66 # pyenv
67 # For a library or package, you might want to ignore these files since the code is
68 # intended to run in multiple environments; otherwise, check them in:
69 # .python-version
70
71 # pipenv
72 # According to pyenv/pipenv#598, it is recommended to include Pipfile.lock in version control.
73 # However, in case of collaboration, if having platform-specific dependencies or dependencies
74 # having no cross-platform support, pipenv may install dependencies that don't work, or not
75 # install all needed dependencies.
76 #Pipfile.lock
77
78 # poetry
79 # Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
80 # This is especially recommended for binary packages to ensure reproducibility, and is more
81 # commonly ignored for libraries.
82 # https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
83 #poetry.lock
84
85 # pdm
86 # Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
87 #pdm.lock
88 # pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
89 # in version control.
90 # https://pdm-project.org/en/latest/#pdm-toml
91 .pdm.toml
92
93 # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
94 __pypackages__/
95
96 # Celery stuff
97 celerybeat-schedule
98 celerybeat-pid
99
100 # SageMath parsed files
101 *.sage.py
102
103 # Environments
104 .env
105 .venv
106 env/
107 env/
108 ENV/
109 env.bak/
110 venv.bak/
111
112 # Spyder project settings
113 .spyderproject
114 .spyproject
115
116 # Rope project settings
117 .ropeproject
118
119 # mkdocs documentation
120 /site
121
122 # mypy
123 .mypy_cache/
124 .dmypy.json
125 dmypy.json
126
127 # Pyre type checker
128 .pyre/
129
130 # pytype static type analyzer
131 .pytype/
132
133
```

```

165 # Cython debug symbols
166 cython_debug/
167
168 # PyCharm
169 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
170 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
171 # and can be added to the global gitignore or merged into this file. For a more nuclear
172 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
173 #.idea/
174
175 ### macOS ###
176 # General
177 .DS_Store
178 .AppleDouble
179 .LSOverride
180
181 # Icon must end with two \n
182 Icon
183
184 # Yarn
185 # Yarn.lock
186
187 # Installer logs
188
189 # Unit test / coverage reports
190
191 # Translations
192
193 # Django stuff:
194
195 # Flask stuff:
196
197 # scrapy stuff:
198
199 # Sphinx documentation
200
201 # PyBuilder
202
203 # Jupyter Notebook
204
205 # IPython
206
207 # pyenv
208 # For a library or package, you might want to ignore these files since the code is
209 # intended to run in multiple environments; otherwise, check them in:
210 # .python-version
211
212 # pipenv
213 # According to pya/pipenv#598, it is recommended to include Pipfile.lock in version control.
214 # However, in case of collaboration, if having platform-specific dependencies or dependencies
215 # having no cross-platform support, pipenv may install dependencies that don't work, or not
216 # install all needed dependencies.
217
218 # poetry
219 # Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
220 # This is especially recommended for binary packages to ensure reproducibility, and is more
221 # commonly ignored for libraries.
222 # https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
223
224 # pdm
225 # Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
226 # pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
227 # in version control.
228 # https://pdm.fming.dev/#use-with-ide
229
230 # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
231
232 # Celery stuff
233
234 # SageMath parsed files
235
236 # Environments
237
238 # Spyder project settings
239
240 # Rope project settings
241
242 # mkdocs documentation
243
244 # mypy
245
246 # Pyre type checker
247
248 # Cython debug symbols
249
250 # PyCharm
251 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
252 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
253 # and can be added to the global gitignore or merged into this file. For a more nuclear
254 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
255 #.idea/
256
257 ### VisualStudioCode ###
258 # .vscode/
259 !.vscode/settings.json
260 !.vscode/tasks.json
261 !.vscode/launch.json
262 !.vscode/extensions.json
263 !.vscode/*.code-snippets
264
265 # Local History for Visual Studio Code
266 .history/
267
268 # Built Visual Studio Code Extensions
269 *.vsix
270
271 ### VisualStudioCode Patch ###
272 # Ignore all local history of files
273 .history
274 .ionide
275
276 # Support for Project snippet scope
277 .vscode/*.code-snippets
278
279 # Ignore Code-workspaces
280 *.code-workspace
281
282 ### Windows ###
283 # Windows thumbnail cache files
284 Thumbs.db
285 Thumbs.db:encryptable
286 ehthumbs.db
287 ehthumbs_vista.db
288
289 # Dump file
290 *.stackdump
291
292 # Folder config file
293 [Dd]esktop.ini
294
295 # Recycle Bin used on file shares
296 $RECYCLE.BIN/
297
298 # Windows Installer files
299 *.cab
300 *.msi
301 *.msix
302 *.msm
303 *.msp
304
305 # Windows shortcuts
306 *.lnk
307
308 # End of https://www.toptal.com/developers/gitignore/api/django,python,windows,macos,visualstudiocode

```

## ● FrontEnd

```
📁 .gitignore
1  # See https://help.github.com/articles/ignoring-files/ for more about ignoring files.
2
3  # dependencies
4  /node_modules
5  /.pnp
6  .pnp.js
7
8  # testing
9  /coverage
10
11 # production
12 /build
13
14 # misc
15 .DS_Store
16 .env.local
17 .env.development.local
18 .env.test.local
19 .env.production.local
20
21 npm-debug.log*
22 yarn-debug.log*
23 yarn-error.log*
24
```