

포팅 메뉴얼

1. 개발 환경

형상 관리

- GITLAB

이슈 관리

- JIRA
 - 매주 목표량을 설정하여 스프린트 진행
 - In-Progress 및 Done으로 각자 지금 하고있는 업무와 완료된 업무를 공유

소통 관리

- Mattermost
 - 프로젝트 자료 공유
 - 의견 작성
 - 미완료 업무 공유
- Notion

UI/UX

- Figma

IDE

- IntelliJ IDEA 2022.3.2
- Visual studio code 1.75

DATABASE

- MYSQL 8.0.31
- MYSQL workbench 8.0 CE

SERVER

- AWS EC2
 - UBUNTU 20.04 LTS
 - MobaXterm_Personal_22.3.exe
 - DOCKER 20.10.23

- NGINX 1.18.0
- S3

협업툴

- SWAGGER 2.9.2
- POSTMAN for Windows Version 10.9.4

BACK-END

- Java Open-JDK azul 11
- SpringBoot Gradle 2.7.7
 - Spring Data JPA
 - Lombok
 - Swagger 2.9.2

FRONT-END

- NPM 9.4.2
- React-Redux 8.0.5
- Redux-toolkit 1.9.1
- toast UI 3.0
- react-easy-swipe 0.0.22

외부 서비스 문서

🔒 소셜 로그인

Google

- OAuth 기반 소셜 로그인 API 제공
 - redirect URI : <https://i8d210.p.ssafy.io/oauth2/callback/google>

웹 앱에 Google 로그인 통합 | Authentication | Google Developers

<https://developers.google.com/identity/sign-in/web/sign-in?hl=ko>

Google Developers

Kakao

- OAuth 기반 소셜 로그인 API 제공
 - redirect URI : <https://i8d210.p.ssafy.io/oauth2/callback/kakao>

Kakao Developers

이 문서는 REST API를 사용한 로그인 구현 방법을 안내합니다. 이 문서에 포함된 기능 일부는 [도구] > [REST API 테스트]를 통해 사용해 볼 수 있습니다. 카카오 로그인 구현에 필요한 로그인 버튼 이미지는 [도구] > [리소스 다운로드]에서 제공합니다. 해당 로그인 버튼은 디자인 가이드를 참고하여 서비스 UI에 적합한 크기로 수정하여 사용할 수 있습니다.

<https://developers.kakao.com/docs/latest/ko/kakaologin/rest-api>

kakao developers

*** Swagger에서 테스트할 때, Bearer {토큰} ***

2. 개발 설정

EC2 내부 Nginx 설정

/etc/nginx/conf.d/youtil.conf

```
server {
    #listen [::]:443 ssl ipv6only=on; # managed by Certbot
    listen 443 ssl; # managed by Certbot

    server_name i8d210.p.ssafy.io; # managed by Certbot

    client_max_body_size 50M;

    ## SSL 인증서 적용
    ssl_certificate /etc/letsencrypt/live/i8d210.p.ssafy.io/fullchain.pem; # managed by Certbot
    ssl_certificate_key /etc/letsencrypt/live/i8d210.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

    location / {
        proxy_pass http://localhost:3000;
    }

    location /oauth2/redirect {
        proxy_pass http://localhost:3000;
    }

    location /api { # location 이후 특정 url을 처리하는 방법을 정의
        proxy_pass http://localhost:8081/api; # Request에 대해 어디로 리다이렉트하는지
        #proxy_pass http://i8d210.p.ssafy.io:8081;
        proxy_redirect off;
        charset utf-8;

        proxy_http_version 1.1;
        proxy_set_header Connection "upgrade";
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Host $http_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;
    }

    location /oauth2/callback{
        proxy_pass http://localhost:8081;/
    }
}
```

사용 명령어

```
sudo vim /etc/nginx/conf.d/godhsl.conf

sudo ln -sf /etc/nginx/conf.d/godhsl.conf /etc/nginx/sites-enabled

sudo nginx -t //문법 오류 확인

sudo service nginx restart
sudo systemctl restart nginx

docker pull nylee98/nginx-react:0.4
docker run --name nginx-react -d -p 3000:80 nylee98/nginx-react:0.4
```

```
sudo systemctl restart nginx
```

*****주의사항: << sudo systemctl restart nginx >> 이 명령어 꼭꼭 실행하기*****

관련 명령어

```
docker stop nginx-react
docker rm nginx-react

docker images

docker rmi {이미지id}
```

프론트 설정(VSCode)

src/constants/index.js 설정

```
////////////////////////////////////
//// 로컬 ///////////////////////////////////
////////////////////////////////////
export const API_BASE_URL = 'http://i8d210.p.ssafy.io:8081/api';
export const OAUTH_API_BASE_URL = 'http://i8d210.p.ssafy.io:8081/api';

export const ACCESS_TOKEN = 'accessToken';

export const TOKEN = () => {
  const token = localStorage.getItem("accessToken");
  return `Bearer ${token}`;
};

export const OAUTH2_REDIRECT_URI = "http://localhost:3000/oauth2/redirect";

export const GOOGLE_AUTH_URL =
  OAUTH_API_BASE_URL + "/oauth2/authorize/google?redirect_uri=" + OAUTH2_REDIRECT_URI;
export const KAKAO_AUTH_URL =
  OAUTH_API_BASE_URL + "/oauth2/authorize/kakao?redirect_uri=" + OAUTH2_REDIRECT_URI;
```

```
////////////////////////////////////
//// 배포 ///////////////////////////////////
////////////////////////////////////
export const API_BASE_URL = 'https://i8d210.p.ssafy.io/api'; //배포용
export const OAUTH_API_BASE_URL = 'https://i8d210.p.ssafy.io/api'; //배포용

export const ACCESS_TOKEN = 'accessToken';

export const TOKEN = () => {
  const token = localStorage.getItem("accessToken");
  return `Bearer ${token}`;
};

export const OAUTH2_REDIRECT_URI = "https://i8d210.p.ssafy.io/oauth2/redirect"; //배포용

export const GOOGLE_AUTH_URL =
  OAUTH_API_BASE_URL + "/oauth2/authorize/google?redirect_uri=" + OAUTH2_REDIRECT_URI;
export const KAKAO_AUTH_URL =
  OAUTH_API_BASE_URL + "/oauth2/authorize/kakao?redirect_uri=" + OAUTH2_REDIRECT_URI;
```

도커에 올리기 위한 설정

- **Dockerfile**

```
# nginx 이미지를 사용합니다. 뒤에 tag가 없으면 latest 를 사용.
FROM nginx

# root 에 app 폴더를 생성
RUN mkdir /app

# work dir 고정
WORKDIR /app

# work dir 에 build 폴더 생성 /app/build
RUN mkdir ./build
```

```
# host pc의 현재경로의 build 폴더를 workdir 의 build 폴더로 복사
ADD ./build ./build

# nginx 의 default.conf 를 삭제
RUN rm /etc/nginx/conf.d/default.conf

# host pc 의 nginx.conf 를 아래 경로에 복사
COPY ./nginx.conf /etc/nginx/conf.d

# 80 포트 오픈
EXPOSE 80

# container 실행 시 자동으로 실행할 command. nginx 시작함
CMD ["nginx", "-g", "daemon off;"]
```

- **nginx.conf**

```
server {
    listen 80;
    location / {
        root    /app/build;
        index   index.html;
        try_files $uri $uri/ /index.html;
    }
}
```

- **도커 이미지 생성 명령어**

```
npm run build
docker build -t nylee98/nginx-react:0.8 .
docker push nylee98/nginx-react:0.8
```

- **docker image 생성 및 push**

```
npm run build
docker pull nylee98/nginx-react:0.4
docker run --name nginx-react -d -p 3000:80 nylee98/nginx-react:0.4
```

젠킨스 파이프라인

서버 자동 배포를 위한 파이프라인 설정

```
pipeline {
    agent any

    stages{
        stage('Prepare') {
            steps {
                echo 'Cloning Repository'
                git url: 'https://lab.ssfy.com/s08-webmobile2-sub2/S08P12D210',
                    branch: 'server',
                    credentialsId: 'gms94249424'
            }
            post {
                success {
                    echo 'Successfully Cloned Repository'
                }
                failure {
                    error 'This pipeline stops here...'
                }
            }
        }
    }
    stage('build'){
        steps{
            dir('server'){
                sh '''
                    chmod +x gradlew
                    echo 'start bootJar'
                    ./gradlew clean bootJar
                '''
            }
        }
    }
}
```

```

    }

    stage('dockerizing'){
        steps{
            // sh 'docker build . -t nylee98/test'
            dir('server'){

                ///////////////젠킨스 실패하면 애네를 주석처리해보자/////////////////
                //docker rmmrrmm
                sh 'docker stop test2'
                sh 'docker rm test2'
                sh 'docker rmi -f $(docker images -f "dangling=true" -q)'
                //docker rmmrrmm
                ///////////////

                sh 'docker build -t nylee98/youtil-server:0.0.1 .'

            }
        }
    }

    stage('Deploy') {
        steps {
            sh 'docker run -d -p 8081:8081 --name test2 nylee98/youtil-server:0.0.1 .'
        }

        post {
            success {
                echo 'success'
            }

            failure {
                echo 'failed'
            }
        }
    }
}
}
}

```

서버 실행 확인

```
docker logs {컨테이너이름}
```

인텔리제이 - application.yml 설정

SpringBoot 2.x.x 사용 경우, file 크기 제한 설정 시 spring.servlet.multipart 사용

```

spring:
  mvc:
    pathmatch:
      matching-strategy: ANT_PATH_MATCHER
  profiles.active: local
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    url: jdbc:mysql://i8d210.p.ssafy.io:3306/youtil?useSSL=false&serverTimezone=UTC&useLegacyDatetimeCode=false&allowPublicKeyRetrieval=true
    username: ssafy
    password: ssafy
    hikari:
      pool-name: jpa-hikari-pool
      maximum-pool-size: 5
      jdbc-url: ${spring.datasource.url}
      username: ${spring.datasource.username}
      password: ${spring.datasource.password}
      driver-class-name: ${spring.datasource.driver-class-name}
      data-source-properties:
        rewriteBatchedStatements: true
  # JPA ??
  jpa:
    generate-ddl: true
    hibernate:
      ddl-auto: update
  #
  properties:
    hibernate:
      jdbc:
        time_zone: Asia/Seoul
      dialect: org.hibernate.dialect.MySQL8Dialect
      hbm2ddl.import_files_sql_extractor: org.hibernate.tool.hbm2ddl.MultipleLinesSqlCommandExtractor
      current_session_context_class: org.springframework.orm.hibernate5.SpringSessionContext

```

```

    default_batch_fetch_size: ${chunkSize:100}
    jdbc.batch_size: 20
    order_inserts: true
    order_updates: true
    format_sql: true
servlet:
  multipart:
    max-file-size: 15MB
    max-request-size: 15MB
    enabled: true

# Security OAuth
security:
  oauth2.client:
    registration:
      google:
        clientId: '913035013476-bebq70bsneh30vbjjd0ktsk0con4ofmr.apps.googleusercontent.com'
        clientSecret: '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'
        redirect-uri: 'https://i8d210.p.ssafy.io/oauth2/callback/google' #배포 프론트
        scope:
          - email
          - profile
      kakao:
        clientId: '5ccc0d6303d193e2fc171cde4c9a8cf9'
        clientSecret: 'PAZgffRuTr5fh89fBMJ0Gxd6DInXqYVt'
        clientAuthenticationMethod: post
        authorizationGrantType: authorization_code
        redirectUri: 'http://i8d210.p.ssafy.io:8081/oauth2/callback/kakao' #로컬 프론트
#      redirect-uri: 'https://i8d210.p.ssafy.io/oauth2/callback/kakao' #배포 프론트
        scope:
          - profile_nickname
          - profile_image
          - account_email
        clientName: Kakao
      naver:
        client-id: ZAnoJt_6zzAuxdYrpGHU
        client-secret: VoSJrzsmw5
        redirect-uri: 'https://i8d210.p.ssafy.io/oauth2/callback/naver' #배포 프론트
        authorization-grant-type: authorization_code
        scope: name, email, profile_image
        client-name: Naver

  provider:
    kakao:
      authorizationUri: https://kauth.kakao.com/oauth/authorize
      tokenUri: https://kauth.kakao.com/oauth/token
      userInfoUri: https://kapi.kakao.com/v2/user/me
      userNameAttribute: id
    naver:
      authorization_uri: https://nid.naver.com/oauth2.0/authorize
      token_uri: https://nid.naver.com/oauth2.0/token
      user-info-uri: https://openapi.naver.com/v1/nid/me
      user_name_attribute: response

# cors
cors:
  allowed-origins: 'http://localhost:'
  allowed-methods: GET, POST, PUT, DELETE, OPTIONS
  allowed-headers: '*'
  max-age: 3600

# jwt secret key
jwt:
  header: Authorization
  secret: 23kljgljewlfjldjsfklj3lkjkllegjkdjkqljslfjdkfjdkajgk3ejhkgdxaj3lj1jelkdjl3hleflkajdfkljhadfasdfadsfasg4ehegfdddjkljel1ldjfld
  token-validity-in-seconds: 86400
  access:
    expire-length: 864000
  refresh:
    expire-length: 864000
  token:
    secret-key: 23kljgljewlfjldjsfklj3lkjkllegjkdjkqljslfjdkfjdkajgk3ejhkgdxaj3lj1jelkdjl3hleflkajdfkljhadfasdfadsfasg4ehegfdddjkljel1ldjfld

app:
  auth:
    tokenSecret: 926D96C90030DD58429D2751AC1BDBBC23kljgljewlfjldjsfklj3lkjkllegjkdjkqljslfjdkfjdkajgk3ejhkgdxaj3lj1jelkdjl3hleflkajdfk
    tokenExpirationMsec: 864000000
  oauth2:
    authorizedRedirectUris:
      - http://i8d210.p.ssafy.io:8081/oauth/token
      - https://i8d210.p.ssafy.io/oauth/token
      - http://localhost:3000/oauth/redirect
      - https://i8d210.p.ssafy.io/oauth2/redirect
      - http://localhost:3000/oauth2/redirect
      - http://i8d210.p.ssafy.io:3000/oauth2/redirect
      - https://i8d210.p.ssafy.io
      - myandroidapp://oauth2/redirect

```

```

- myiosapp://oauth2/redirect
cors:
  allowedOrigins: http://localhost:3000, https://i8d210.p.ssafy.io, http://localhost:8080, http://i8d210.p.ssafy.io:8081

cloud:
  aws:
    credentials:
      access-key: AKIA5N4A3A664JIM6GP
      secret-key: N+DFJ3lztIy/SdUTAL334H3VWfcDov80/frnHZuz
    s3:
      bucket: utilbucket
      region:
        static: ap-northeast-2
      stack:
        auto: false
  server:
    port: 8081
    tomcat:
      remoteip:
        protocol-header: x-forwarded-proto
    servlet:
      session:
        cookie:
          same-site: none
          secure: true

```

인텔리제이 - Dockerfile 설정

프로젝트 루트에 Dockerfile 생성

```

FROM openjdk:11
VOLUME /tmp
EXPOSE 8081
ARG JAR_FILE=build/libs/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
ENV TZ=Asia/Seoul
RUN apt-get install -y tzdata

```

인텔리제이 - gradle.build 설정

```

buildscript {
    dependencies {
        classpath("gradle.plugin.com.ewerk.gradle.plugins:querydsl-plugin:1.0.10")
    }
}

plugins {
    id 'java'
    id 'org.springframework.boot' version '2.7.7'
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'
    id 'com.ewerk.gradle.plugins.querydsl' version "1.0.10"
}

group = 'com.youtil'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '11'

apply plugin: "com.ewerk.gradle.plugins.querydsl"

configurations {
    compileOnly {
        extendsFrom annotationProcessor
    }
}

repositories {
    mavenCentral()
}

dependencies {
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'
    compileOnly 'org.projectlombok:lombok'
    developmentOnly 'org.springframework.boot:spring-boot-devtools'
    runtimeOnly 'com.h2database:h2'
    annotationProcessor 'org.projectlombok:lombok'
}

```



```

testImplementation 'org.springframework.boot:spring-boot-starter-test'

implementation 'io.springfox:springfox-swagger2:2.9.2'
implementation 'io.springfox:springfox-swagger-ui:2.9.2'

implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
developmentOnly 'org.springframework.boot:spring-boot-devtools'

implementation 'org.springframework.boot:spring-boot-starter-validation'

runtimeOnly 'mysql:mysql-connector-java'

// QueryDSL
implementation 'com.querydsl:querydsl-jpa'
implementation 'com.querydsl:querydsl-apt'

implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'
implementation 'org.springframework.boot:spring-boot-starter-web'
implementation 'org.springframework.boot:spring-boot-starter-security'
implementation 'io.jsonwebtoken:jjwt-api:0.11.2'
implementation 'jakarta.xml.bind:jakarta.xml.bind-api:2.3.2'
runtimeOnly 'io.jsonwebtoken:jjwt-impl:0.11.2'
runtimeOnly 'io.jsonwebtoken:jjwt-jackson:0.11.2'
testImplementation 'org.springframework.security:spring-security-test'

implementation 'com.google.code.gson:gson:2.8.6'
implementation 'org.springframework.cloud:spring-cloud-starter-aws:2.2.6.RELEASE'

annotationProcessor "org.springframework.boot:spring-boot-configuration-processor"

implementation 'org.jsoup:jsoup:1.11.3'
implementation 'com.vladmihalcea:hibernate-types-55:2.20.0'
}

tasks.named('test') {
    useJUnitPlatform()
}
def querydslDir = "$buildDir/generated/querydsl"

querydsl {
    library = "com.querydsl:querydsl-apt"
    jpa = true
    querydslSourcesDir = querydslDir
}

sourceSets {
    main {
        java {
            srcDirs = ['src/main/java', querydslDir]
        }
    }
}

compileQuerydsl {
    options.annotationProcessorPath = configurations.querydsl
}

configurations {
    querydsl.extendsFrom compileClasspath
}

```

s3 설정

IAM 사용자 생성

사용자 세부 정보 지정

사용자 세부 정보

사용자 이름

youitlll

사용자 이름은 최대 64자까지 가능합니다. 유효한 문자: A-Z, a-z, 0-9 및 +, -, @, _ (하이픈)

☐ 콘솔 액세스 활성화 - 선택 사항

사용자가 AWS 관리 콘솔에 로그인할 수 있도록 허용하는 암호를 활성화합니다.

프로그래밍 방식의 액세스의 경우 사용자를 생성한 후 액세스 키를 생성할 수 있습니다. 자세히 알아보기

취소

다음

• 권한 옵션 : 직접 정책 연결

IAM > 사용자 > 사용자 생성

1단계
사용자 세부 정보 지정

2단계
권한 설정

3단계
검토 및 생성

권한 설정

기존 그룹에 사용자를 추가하거나 새 그룹을 생성합니다. 직무별로 사용자의 권한을 관리하려면 그룹을 사용하는 것이 좋습니다. 자세히 알아보기

권한 옵션

☐ 그룹에 사용자 추가
기존 그룹에 사용자를 추가하거나 새 그룹을 생성합니다. 그룹을 사용하여 직무별로 사용자 권한을 관리하는 것이 좋습니다.

☐ 권한 복사
기존 사용자의 모든 그룹 멤버십, 연결된 관리형 정책 및 인라인 정책을 복사합니다.

☒ 직접 정책 연결
관리형 정책을 사용자에게 직접 연결합니다. 사용자에게 연결하는 대신, 정책을 그룹에 연결한 후 사용자를 적절한 그룹에 추가하는 것이 좋습니다.

권한 정책 (1038)
새 사용자에게 연결할 정책을 하나 이상 선택합니다.

텍스트 또는 값을 기준으로 배포 필터링

< 1 2 3 4 5 6 7 ... 52 > ⚙

<input type="checkbox"/>	정책 이름	유형	연결된 엔터티
<input type="checkbox"/>	AccessAnalyzerServiceRolePolicy	AWS 관리형	0

• 권한 정책 :

- AmazonS3FullAccess 선택
- AWSLambda_fullAccess 선택

권한 정책 (1/1038)

↺

정책 생성

새 사용자에게 연결할 정책을 하나 이상 선택합니다.

🔍

텍스트 또는 값을 기준으로 배포 필터링

11 개 일치

<

1

>

s3

✕

필터 지우기

	정책 이름	유형	연결된 엔터티
<input type="checkbox"/>	AmazonDMSRedshiftS3Role	AWS 관리형	0
<input checked="" type="checkbox"/>	AmazonS3FullAccess	AWS 관리형	2
<input type="checkbox"/>	AmazonS3ObjectLambdaExecution...	AWS 관리형	0
<input type="checkbox"/>	AmazonS3OutpostsFullAccess	AWS 관리형	0
<input type="checkbox"/>	AmazonS3OutpostsReadOnlyAccess	AWS 관리형	0
<input type="checkbox"/>	AmazonS3ReadOnlyAccess	AWS 관리형	0
<input type="checkbox"/>	AWSBackupServiceRolePolicyForS3...	AWS 관리형	0
<input type="checkbox"/>	AWSBackupServiceRolePolicyForS3...	AWS 관리형	0
<input type="checkbox"/>	IVSRecordToS3	AWS 관리형	0
<input type="checkbox"/>	QuickSightAccessForS3StorageMan...	AWS 관리형	0

권한 정책 (2/1038)

↺

정책 생성

새 사용자에게 연결할 정책을 하나 이상 선택합니다.

🔍

텍스트 또는 값을 기준으로 배포 필터링

13 개 일치

<

1

>

⚙️

awslambda

✕

필터 지우기

	정책 이름	유형	연결된 엔터티
<input checked="" type="checkbox"/>	AWSLambda_FullAccess	AWS 관리형	1
<input type="checkbox"/>	AWSLambda_ReadOnlyAccess	AWS 관리형	0
<input type="checkbox"/>	AWSLambdaBasicExecutionRole	AWS 관리형	0

액세스 키 생성

- 사용자 선택 후 액세스 키 생성(엑세스키 만들기)

액세스 키 (0)

엑세스 키 만들기

엑세스 키를 사용하여 AWS CLI, AWS Tools for PowerShell, AWS SDK 또는 직접 AWS API 호출을 통해 AWS에 프로그래밍 방식 호출을 전송합니다. 한 번에 최대 두 개의 액세스 키(활성 또는 비활성)를 가질 수 있습니다. [Learn more](#)

엑세스 키 없음

엑세스 키와 같은 장기 자격 증명을 사용하지 않는 것이 모범 사례입니다. 대신 단기 자격 증명을 제공하는 도구를 사용하세요. [Learn more](#)

엑세스 키 만들기

포팅 메뉴얼


11

1단계
액세스 키 모범 사례 및 대안2단계 - 선택 사항
설명 태그 설정3단계
액세스 키 검색

액세스 키 모범 사례 및 대안

보안 개선을 위해 액세스 키와 같은 장기 자격 증명을 사용하지 마세요. 다음과 같은 사용 사례와 대안을 고려하세요.

- ☐ Command Line Interface(CLI)
AWS CLI를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☐ 로컬 코드
로컬 개발 환경의 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☒ AWS 컴퓨팅 서비스에서 실행되는 애플리케이션
Amazon EC2, Amazon ECS 또는 AWS Lambda와 같은 AWS 컴퓨팅 서비스에서 실행되는 애플리케이션 코드를 사용하여 AWS 계정에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☐ 서버 파티 서비스
AWS 리소스를 모니터링 또는 관리하는 서버 파티 애플리케이션 또는 서비스에 액세스할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☐ AWS 외부에서 실행되는 애플리케이션
애플리케이션을 온프레미스 호스트에서 실행하거나 로컬 AWS 클라이언트 또는 서버 파티 AWS 플러그인을 사용할 수 있도록 이 액세스 키를 사용할 것입니다.
- ☐ 기타
귀하의 사용 사례가 여기에 나열되어 있지 않습니다.

 **권장되는 대안**
EC2 인스턴스 또는 Lambda 함수와 같은 컴퓨팅 리소스에 IAM 역할을 할당하여 액세스를 위한 임시 보안 인증을 자동으로 제공합니다. [Learn more](#)

☐ 위의 권장 사항을 이해했으며 액세스 키 생성을 계속하려고 합니다.

취소 다음

3. 버킷 생성

- 버킷 이름은 소문자만 가능
- 지역은 서울로 설정

버킷 만들기 [Info](#)

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

일반 구성

버킷 이름

UTILBUCKET

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.



ACL 활성화 관련 향후 권한 변경 사항

2023년 4월부터는 S3 콘솔을 사용하여 버킷을 생성할 때 ACL을 활성화하기 위해 s3:PutBucketOwnershipControls 권한이 있어야 합니다. [자세히 알아보기](#)

• 퍼블릭으로 설정

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지정 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 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.

- 정책 설정

버킷 정책 편집
info

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. 자세히 알아보기

버킷 ARN

arn:aws:s3:::myyoutilbucket

정책

```

1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "AddPerm",
6       "Effect": "Allow",
7       "Principal": "*",
8       "Action": "s3:*",
9       "Resource": [
10        "arn:aws:s3:::myyoutilbucket/*",
11        "arn:aws:s3:::myyoutilbucket"
12      ]
13    }
14  ]
15 }

```

문 편집
AddPerm
제거

1. 작업 추가
서비스 선택

포함됨
s3

사용 가능
s3 Object Lambda
s3 Outposts

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AddPerm",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": [
        "arn:aws:s3:::myyoutilbucket/*",
        "arn:aws:s3:::myyoutilbucket"
      ]
    }
  ]
}

```

- cors 설정

CORS(cross-origin 리소스 공유) 편집 Info

CORS(Cross-origin 리소스 공유)

JSON으로 작성된 CORS 구성은 한 도메인에 로드되어 다른 도메인의 리소스와 상호 작용하는 클라이언트 웹 애플리케이션에 대한 방법을 정의합니다. [자세히 알아보기](#)

```

1  [
2    {
3      "AllowedHeaders": [
4        "*"
5      ],
6      "AllowedMethods": [
7        "GET",
8        "PUT",
9        "POST",
10       "HEAD"
11     ],
12     "AllowedOrigins": [
13       "*"
14     ],
15     "ExposeHeaders": [
16       "x-amz-server-side-encryption",
17       "x-amz-request-id",
18       "x-amz-id-2"
19     ],
20     "MaxAgeSeconds": 3000
21   }
22 ]

```

취소

변경 사항 저장

```

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

```