

외부 서비스

소셜 로그인

공통 작업

카카오 로그인

구글 로그인

네이버 로그인

AWS S3

소셜 로그인

공통 작업

1. 애플리케이션 등록
2. 도메인 등록
3. Redirect URI 설정
4. yml 파일에 각 플랫폼 별 클라이언트 ID, 클라이언트 Secret, Redirect URI 설정

JWT 설정

```
token:
  # 일단 하루
  expiration_time: 86400000
  secret: ${secretcode} # openssl rand -hex 64 (HS512 알고리즘 맞는 키 발급)
  refresh-cookie-key: ${refreshkey} # 이걸 원하는대로
```

로그인 인증 성공 후 클라이언트에게 리다이렉트 uri 넘기는 부분

```
oauth2:
```

```
authorizedRedirectUri: ${client-host}/oauth2/redirect
```

카카오 로그인

- 공통 작업
- yml 파일

```
security:
  oauth2:
    client:
      registration:
        kakao:
          client-id: ${KAKAO.CLIENT_ID}
          client-secret: ${KAKAO.CLIENT_SECRET}
          redirect-uri: ${BASE.URL}${KAKAO.REDIRECT_URI}
          authorization-grant-type: authorization_code
          client-authentication-method: POST
          client-name: Kakao
          scope:
            - profile_nickname
      provider:
        kakao:
          authorization-uri: https://kauth.kakao.com/oauth/authorize
          token-uri: https://kauth.kakao.com/oauth/token
          user-info-uri: https://kapi.kakao.com/v2/user/me
          user-name-attribute: id
```

구글 로그인

- 구글 클라우드 콘솔 어플리케이션 등록후
- yml 파일

```
security:
  oauth2:
    client:
      registration:
        google:
          client-id: ${GOOGLE.CLIENT_ID}
          client-secret: ${GOOGLE.CLIENT_SECRET}
          redirect-uri: ${BASE.URL}${GOOGLE.REDIRECT_URI}
          scope:
            - profile
```

네이버 로그인

- 네이버 어플리케이션 등록후 네이버 로그인 API 설정
- yml 파일

```
security:
  oauth2:
    client:
      registration:
        naver:
          client-id: ${NAVER.CLIENT_ID}
          client-secret: ${NAVER.CLIENT_SECRET}
          redirect-uri: ${BASE.URL}${NAVER.REDIRECT_URI}
          authorization-grant-type: authorization_code
          scope:
            - name

      provider:
        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
```

AWS S3

- 상품 이미지 및 작가 프로필, 배경 이미지 등록 시 이미지를 저장하는 객체 스토리지
- <https://aws.amazon.com/ko/s3/getting-started>
- 버킷 권한 설정

권한 개요

액세스

 파블릭

파블릭 액세스 차단(버킷 설정)

파블릭 액세스는 ACL(액세스 제어 목록), 버킷 정책, 액세스 지정 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 모든 S3 버킷 및 객체에 대한 파블릭 액세스가 차단되었는지 확인하려면 [모든 파블릭 액세스 차단]을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지정에만 적용됩니다. AWS에서는 [모든 파블릭 액세스 차단]을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 파블릭 액세스가 있어도 애플리케이션이 올바르게 작동하는지 확인합니다. 버킷 또는 내부 객체에 어느 정도 수준의 파블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. 자세히 알아보기 [\[링크\]](#)

편집

모든 파블릭 액세스 차단

 비밀성

▶ 이 버킷의 개별 파블릭 액세스 차단 설정

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. 자세히 알아보기 [\[링크\]](#)

편집

삭제

```
{
  "Version": "2012-10-17",
  "Id": "Policy",
  "Statement": [
    {
      "Sid": "Stmt1659422899118",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::beedly-img/*"
    }
  ]
}
```

 복사

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 [\[링크\]](#)

```
{
  {
    "AllowedHeaders": [
      "*"
    ],
    "AllowedMethods": [
      "HEAD",
      "GET",
      "PUT",
      "POST",
      "DELETE"
    ],
    "AllowedOrigins": [
      "*"
    ],
    "ExposeHeaders": [
      "ETag",
      "x-amz-meta-custom-header"
    ]
  }
}
```

- IAM 설정
 - 사용자 추가 후 권한 설정

권한

그룹

태그

보안 자격 증명


액세스 관리자

▼ Permissions policies (1 정책이 적용됨)

권한 추가

정책 이름 ▼

직접 연결

▶  AmazonS3FullAccess