Session 14

AWS S3 버킷 생성 및 DJANGO 연동

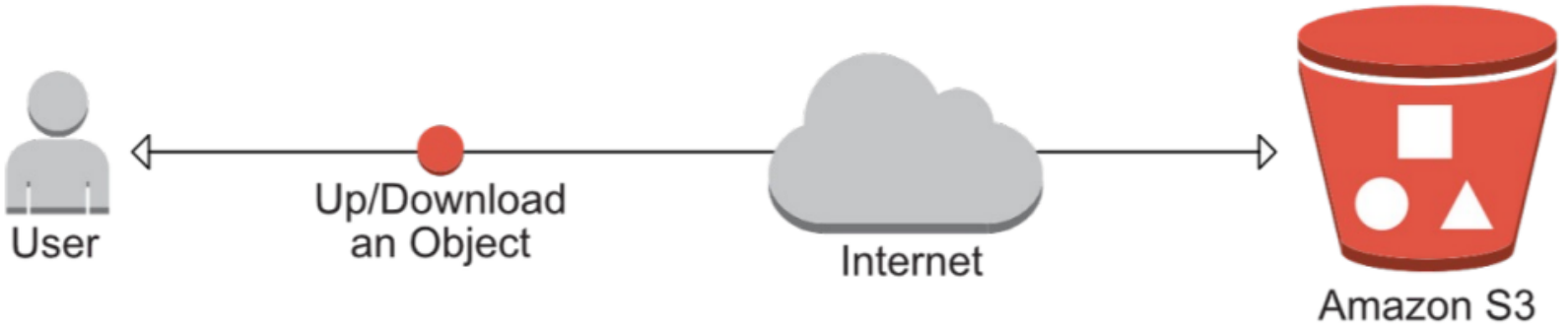
NEXT X LIKELION 이혁준

목차

S3?

S3 = Simple Storage Service

- 유저가 인터넷 상에 파일을 쉽게 업로드하고, 빠르게 접근할 수 있도록 하는 서비스



S3?

Amazon Simple Storage Service (Amazon S3)는 업계 최고의 확장성과 데이터 가용성 및 보안과 성능을 제공하는 **객체 스토리지 서비스**입니다. 즉, 어떤 규모 어떤 산업의 고객이든 이 서비스를 사용하여 웹 사이트, 모 바일 애플리케이션, 백업 및 복원, 아카이브, 엔터프라이즈 애플리케이션, IoT 디바이스, 빅 데이터 분석 등과 같 은 다양한 사용 사례에서 원하는 만큼의 데이터를 저장하고 보호할 수 있습니다. Amazon S3는 사용하기 쉬운 관리 기능을 제공하므로 특정 비즈니스, 조직 및 규정 준수 요구 사항에 따라 데이터를 조직화하고 **세부적인 액세스 제어를 구성**할 수 있습니다. Amazon S3는 **99.999999999%의 내구성**을 제공하도록 설계되었으며, 전 세계 기업의 수백만 애플리케이션을 위한 데이터를 저장합니다.

S3 사용 이유

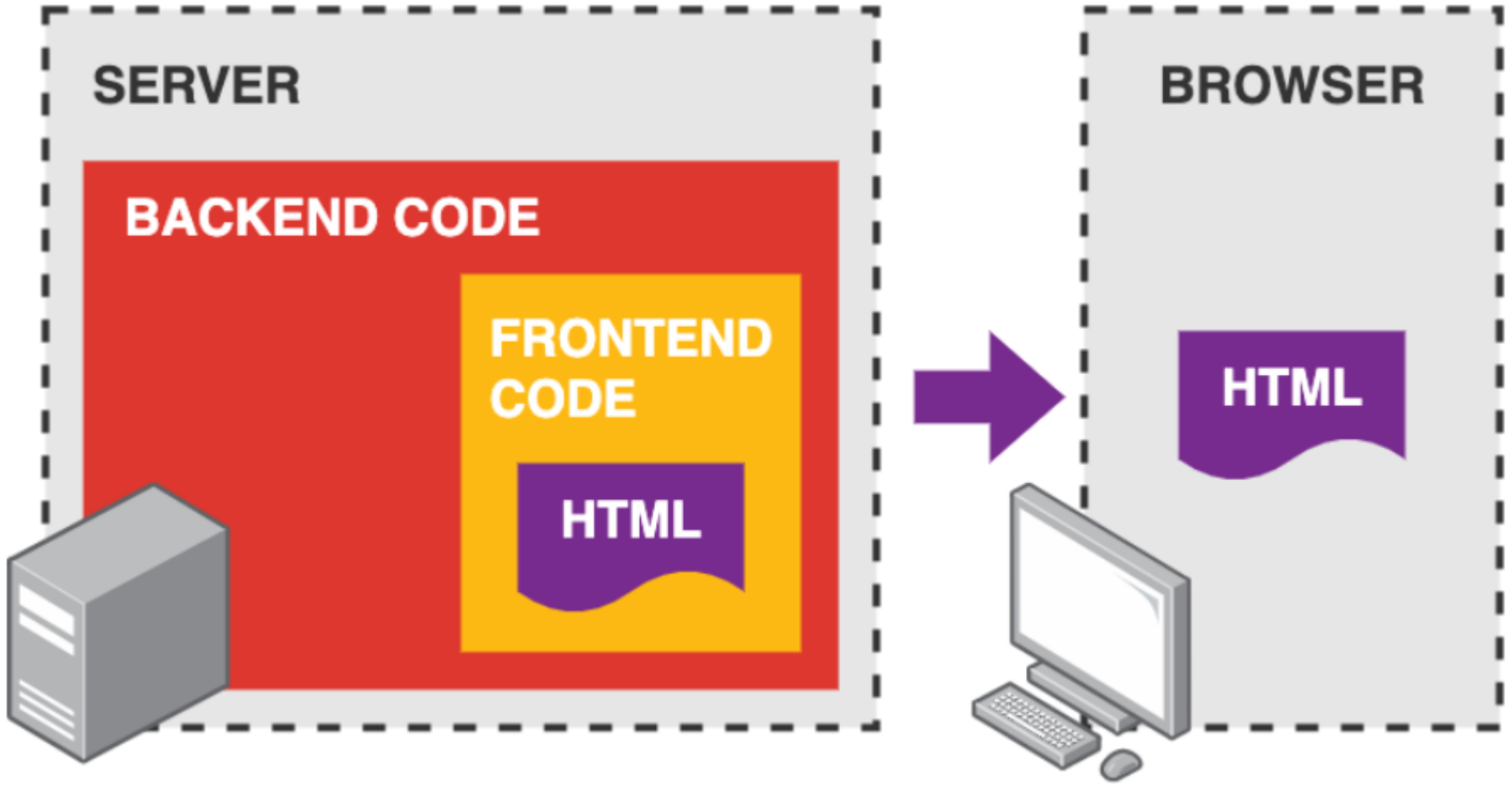
- S3는 저장 용량이 무한대이고, 파일 입출력에 최적화 되어 있다
- 별도의 클라이언트나 중간 매개체 없이 REST API로도 파일 업로드, 다운로드가 가능하다.

- S3 또한 하나의 서버이기 때문에 S3 자체로 정적인 웹서비스를 제공할 수 있다. 이제 여러분들이 하게 될 REACT는 S3에 배포된다.

- 보통 정적 웹사이트를 S3에, WAS를 EC2에 배포해서 서로 소통하도록 구축한다.

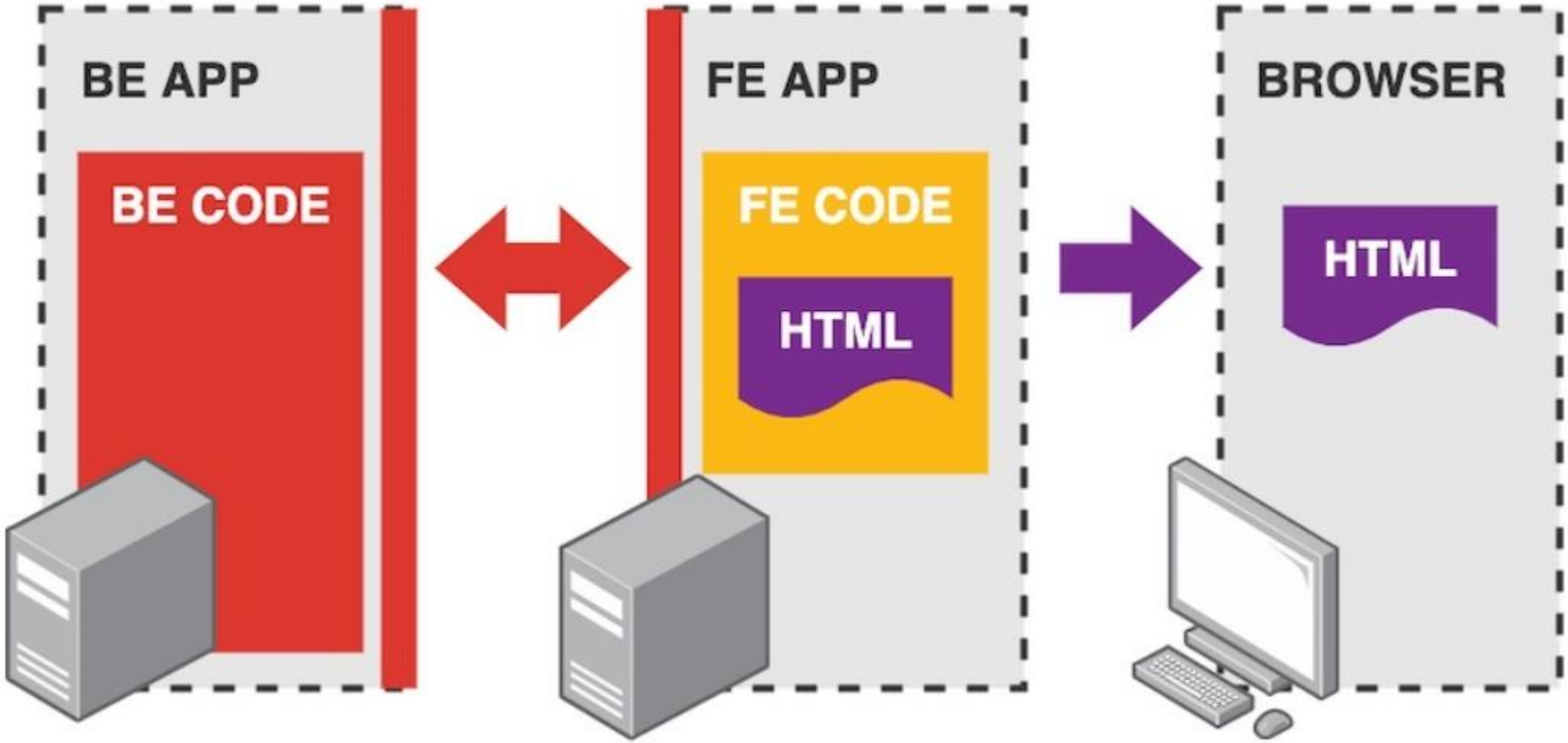
정적 웹사이트, REST..?

기존의 Monolithic Architecture



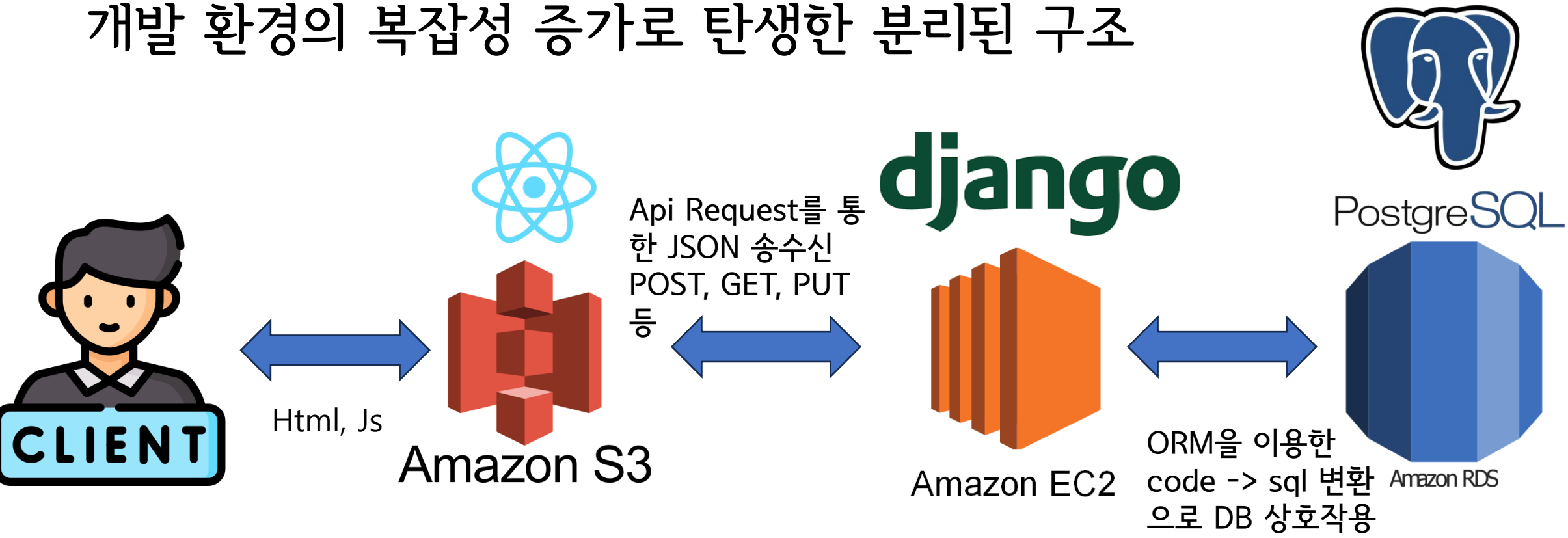
정적 웹사이트, REST..?

개발 환경의 복잡성 증가로 탄생한 분리된 구조



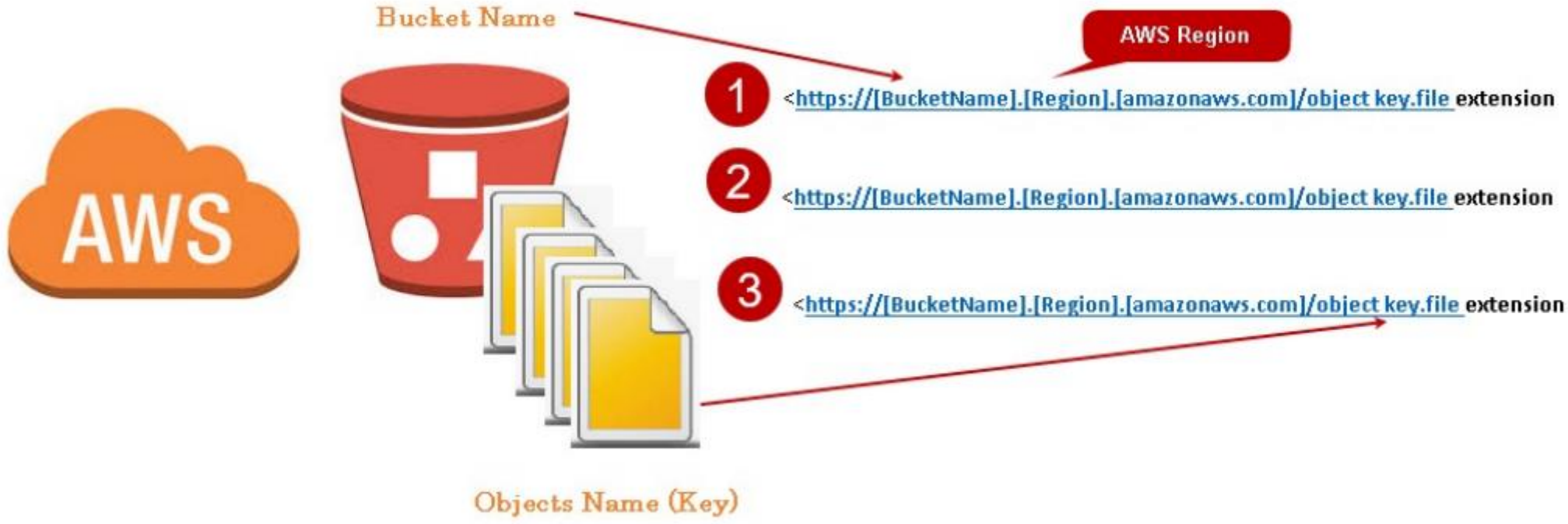
정적 웹사이트, REST..?

개발 환경의 복잡성 증가로 탄생한 분리된 구조



S3 구성 : 버킷과 객체

- 버킷(Bucket): 저장된 객체들의 묶음 (하나의 디비로 생각하면 편해요)
- 객체(Object): 데이터를 구성하는 저장 단위

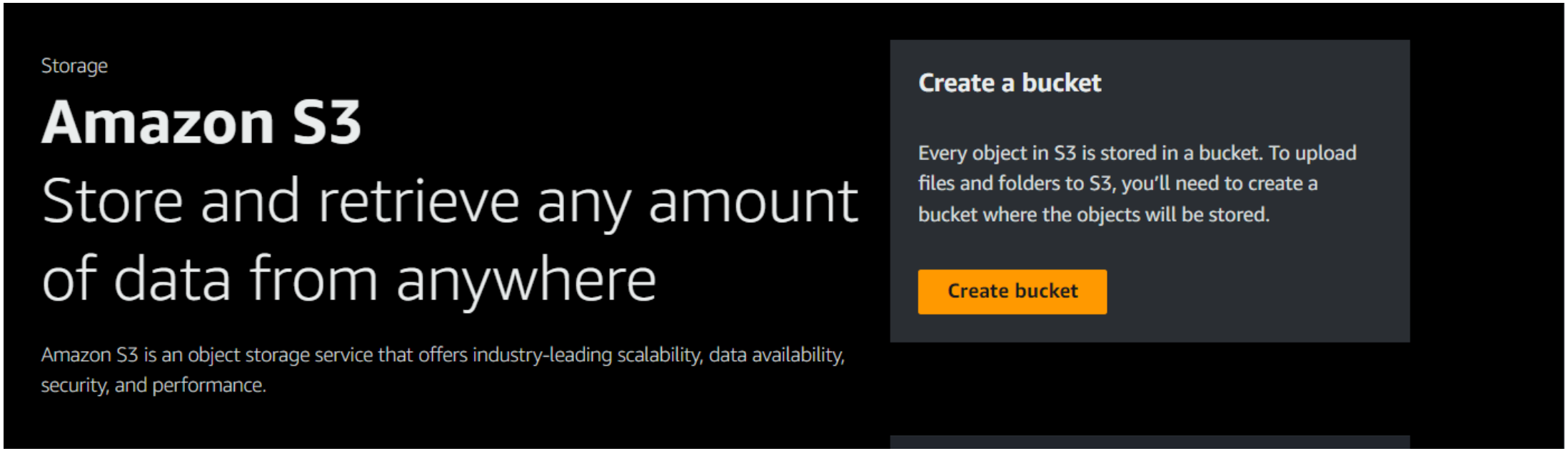


S3: 실습 개괄

- 오늘 할 일
  - S3 버킷 생성
  - S3 접속 권한을 위한 IAM 생성
  - S3 버킷 권한 정책 설정
  - 미리 만들어둔 깃헙 레포 포크(Fork)
  - Django에 버킷 연결
  - 실제로 이미지 업로드
  - 행복한 마무리…

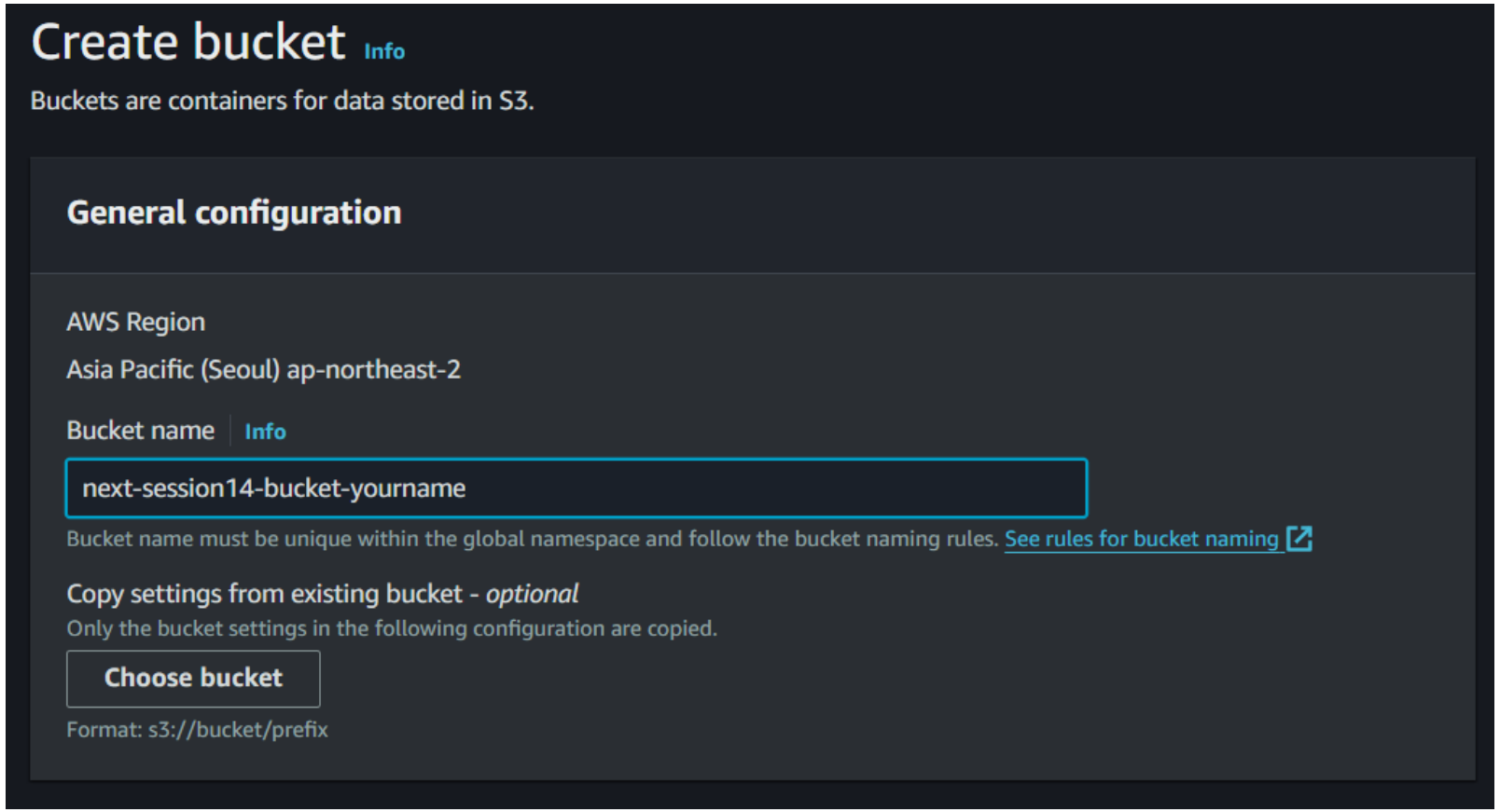
S3: 버킷 생성

https://ap-northeast-2.console.aws.amazon.com/s3/get-started?region=ap-northeast-2



S3: 버킷 생성: 버킷 이름 정하기(Unique)

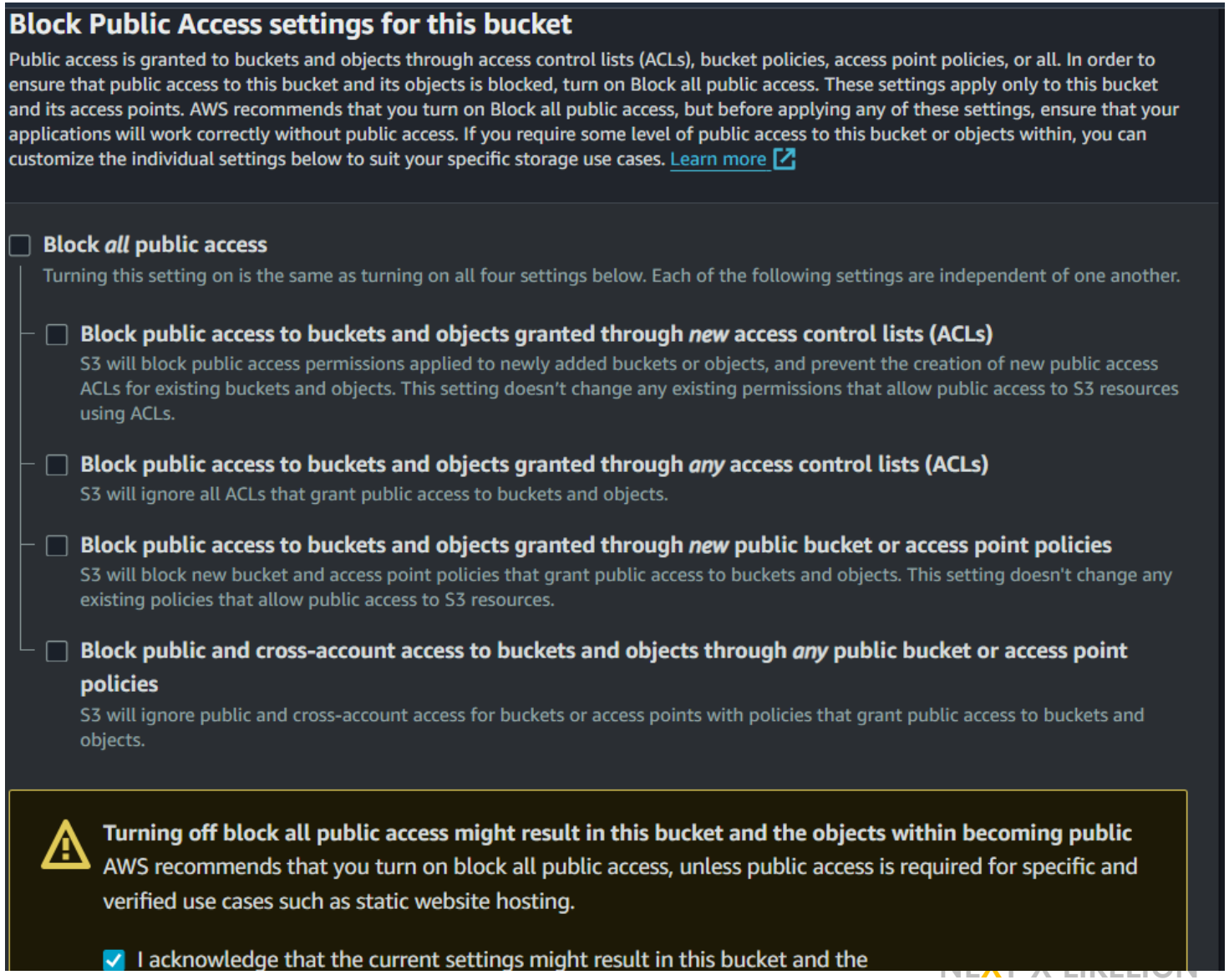
https://ap-northeast-2.console.aws.amazon.com/s3/get-started?region=ap-northeast-2



S3: 버킷 생성

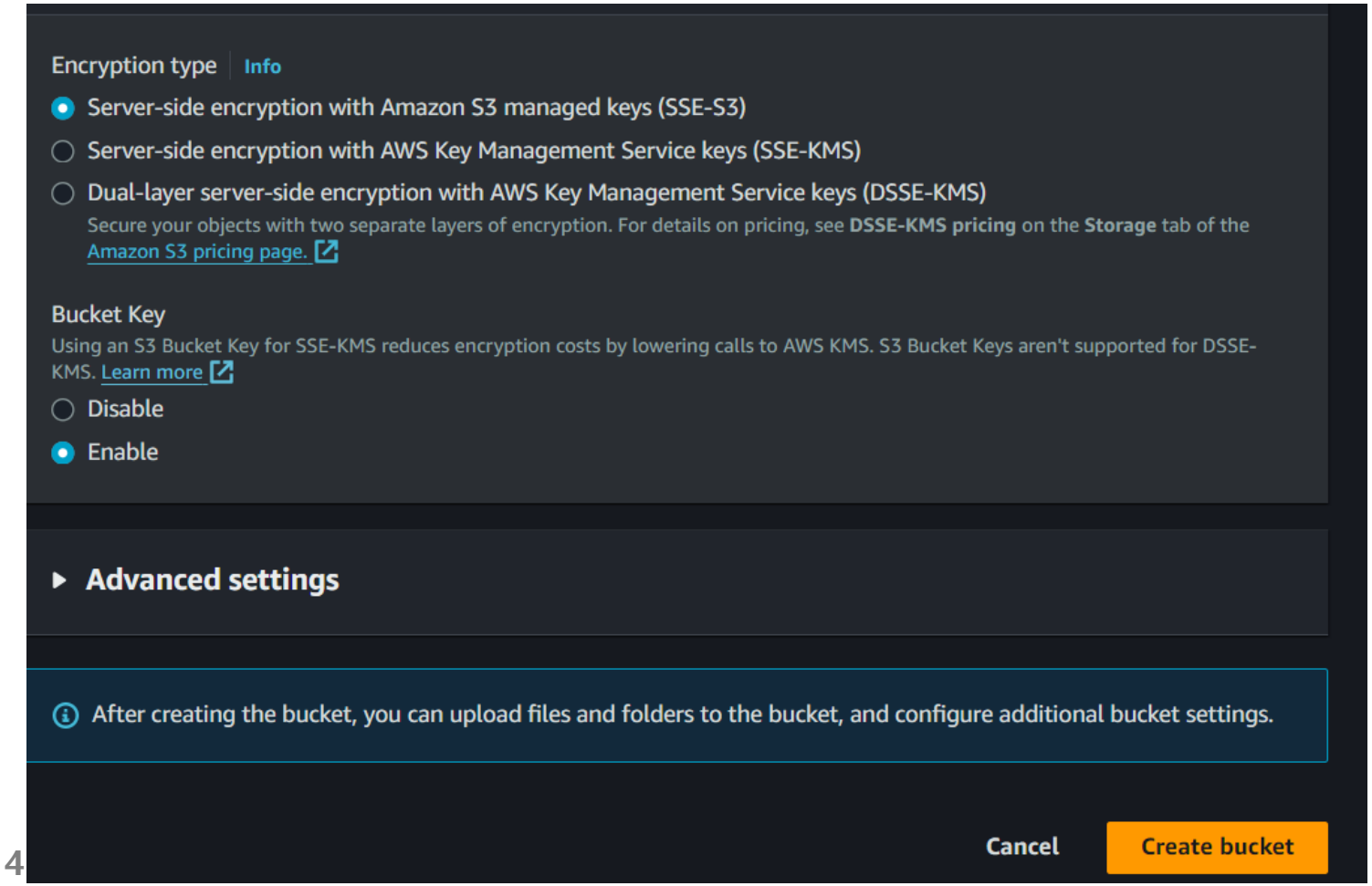
지금은 이렇게 다 허용하지만, 버킷 생성 이후에 권한 설정 다시 할 것입니다.

모든 퍼블릭 엑세스를 허용해주세요



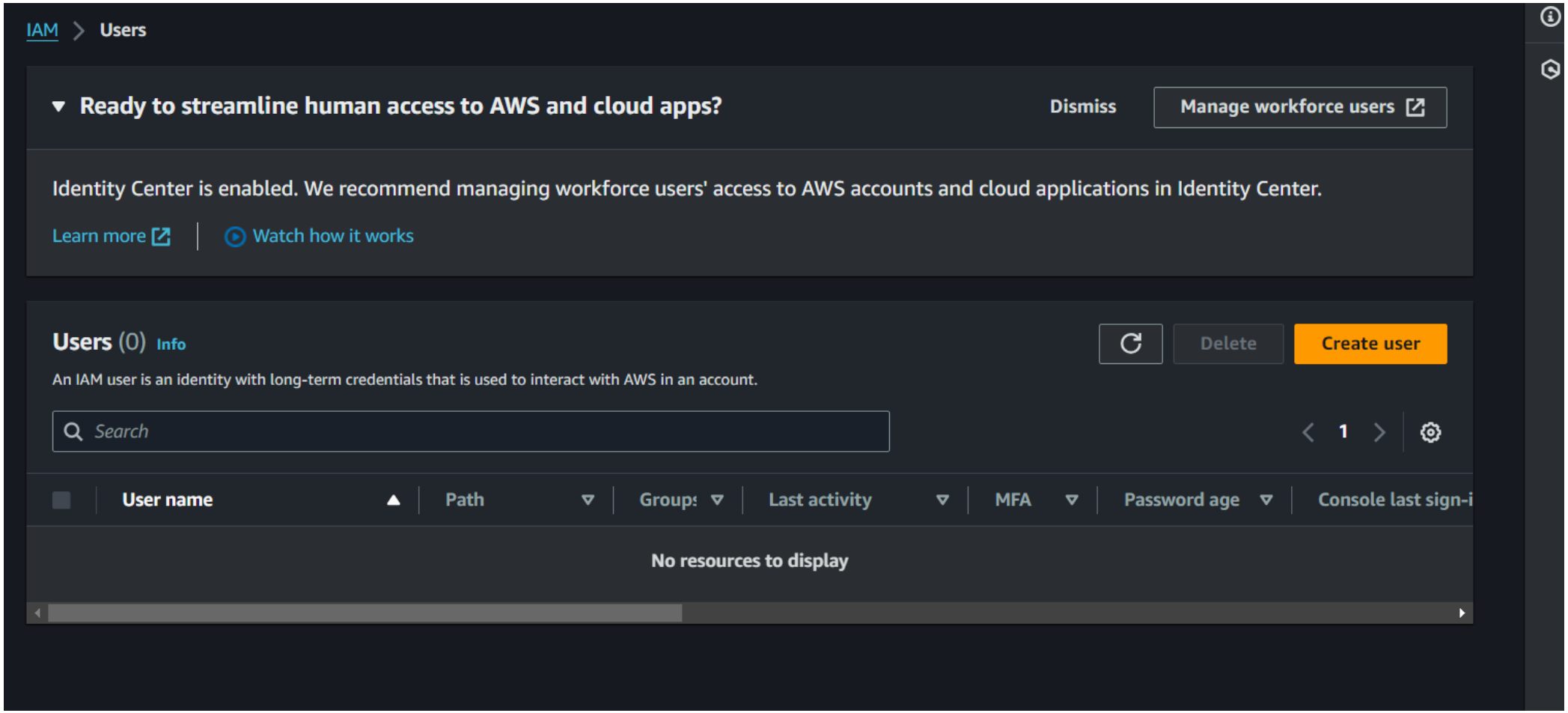
S3: 버킷 생성

다 했으면 생성!

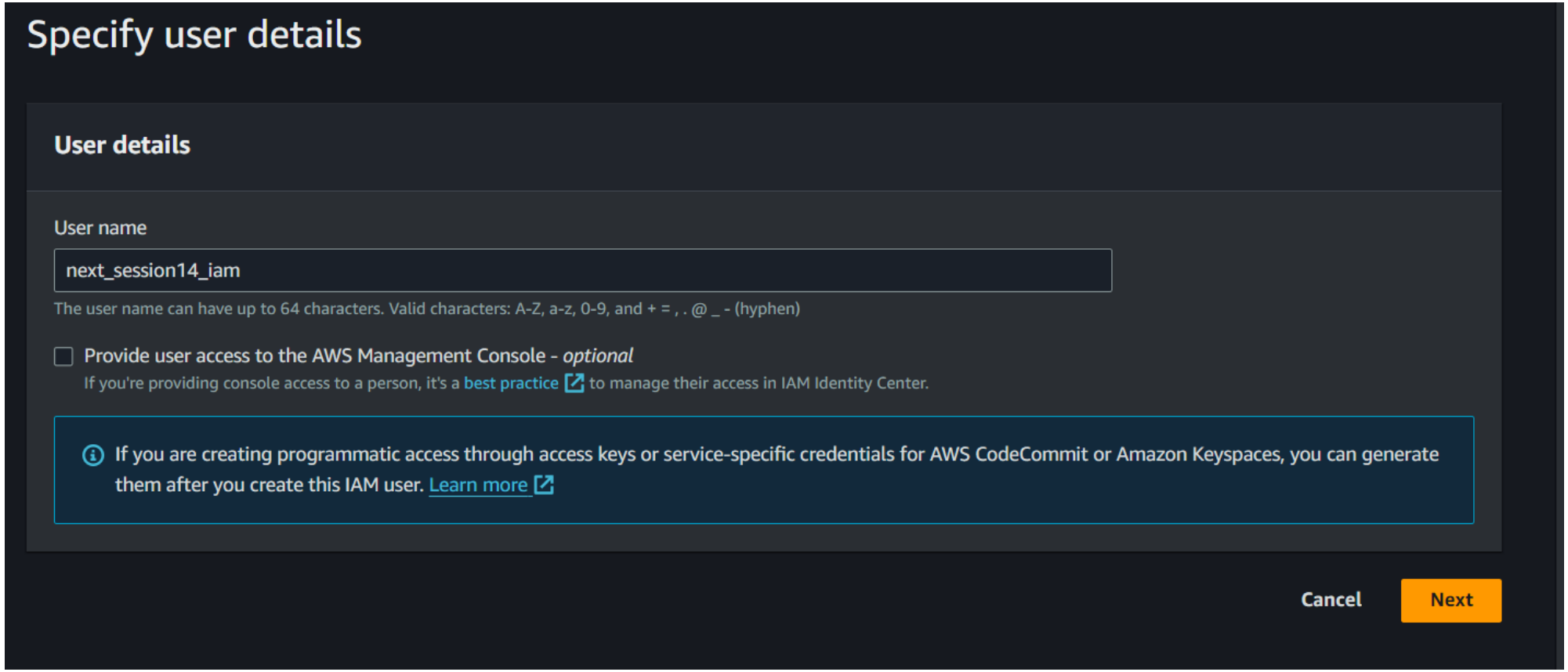


S3: IAM 생성

https://us-east-1.console.aws.amazon.com/iam/home?region=ap-northeast-2#/users

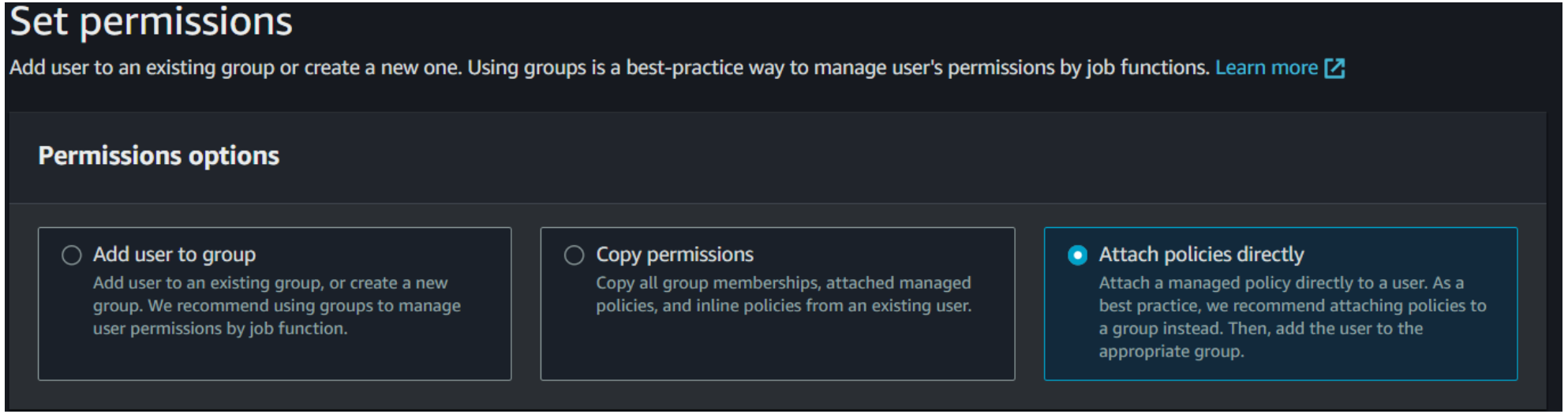


S3: IAM 생성


Specify user details
User details
User name
next_session14_iam
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next

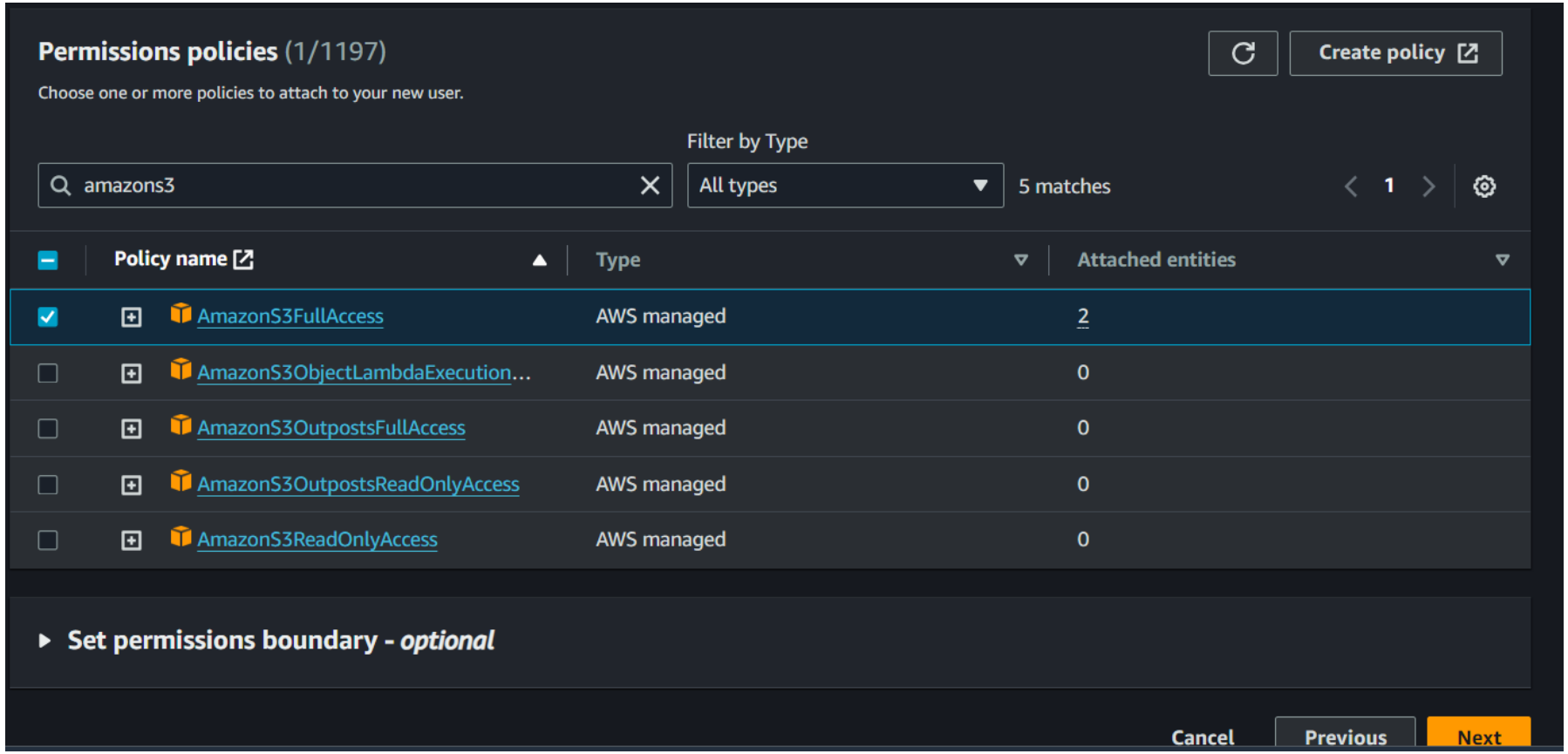
S3: IAM 생성

Django에서 접속할 때 사용하기 위해 관련 정책을 직접 추가해 줄 것입니다.



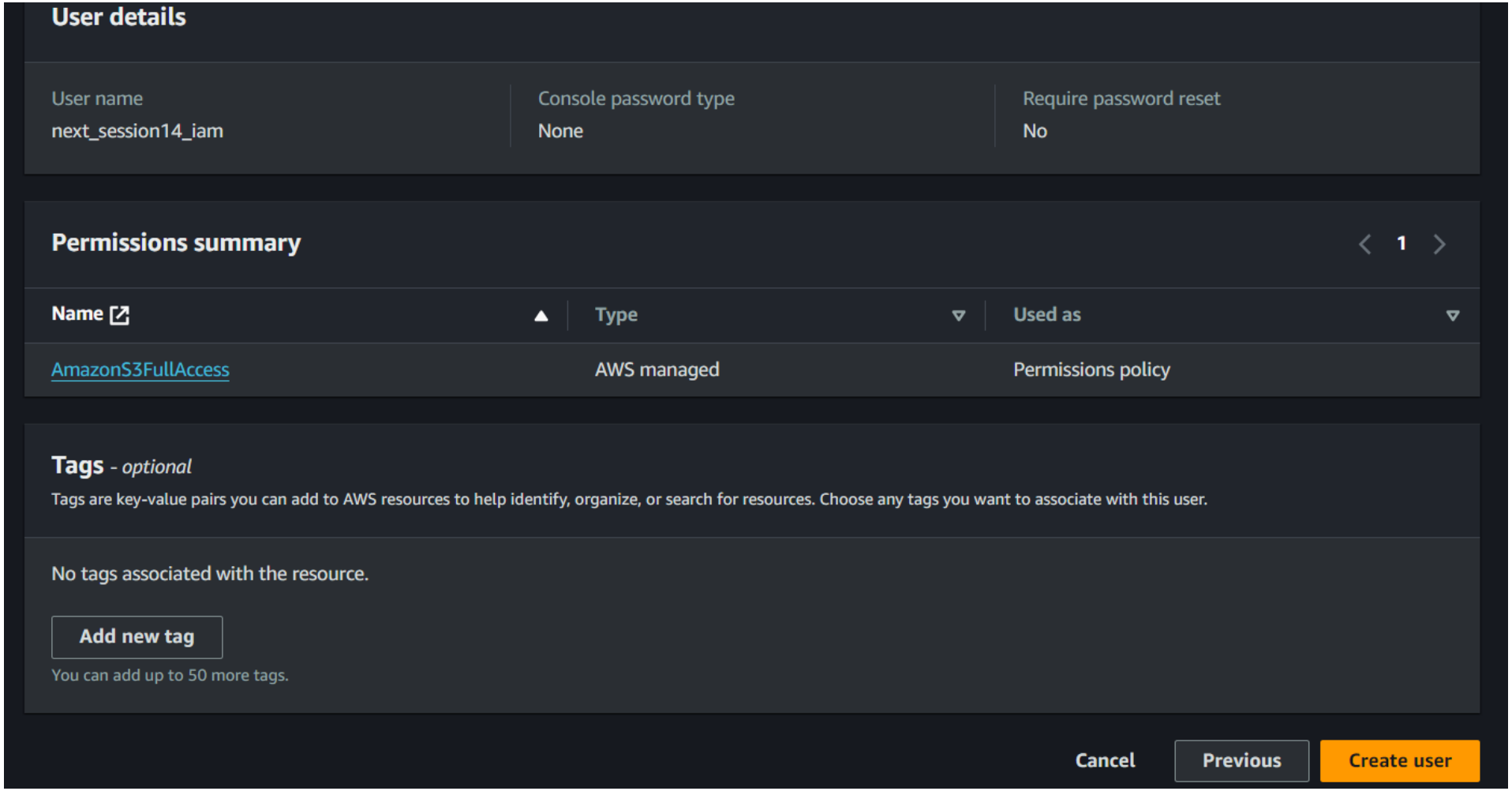
S3: IAM 생성

Django에서 접속할 때 사용하기 위해 관련 정책을 직접 추가해 줄 것입니다.



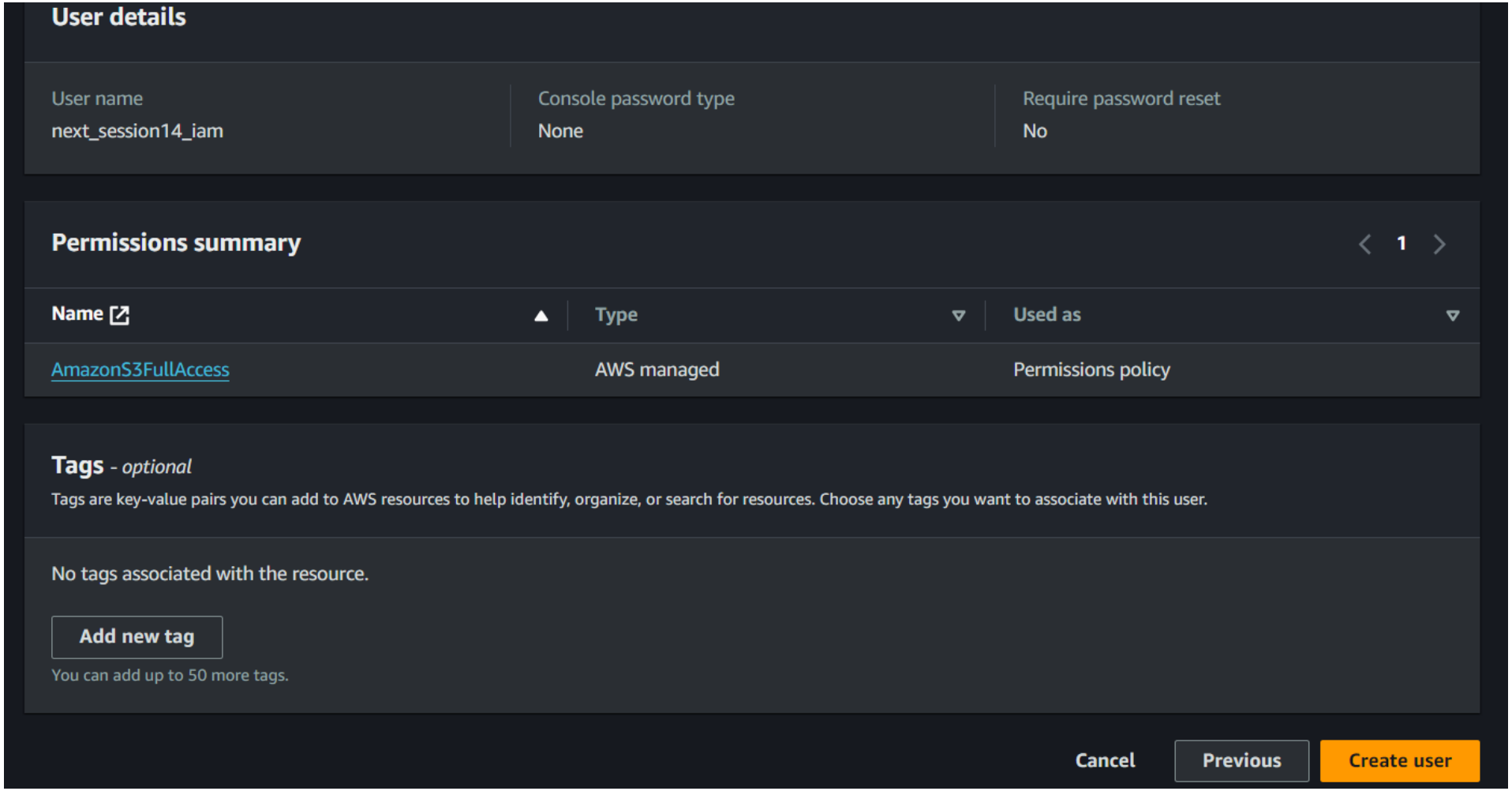
S3: IAM 생성

유저 생성!



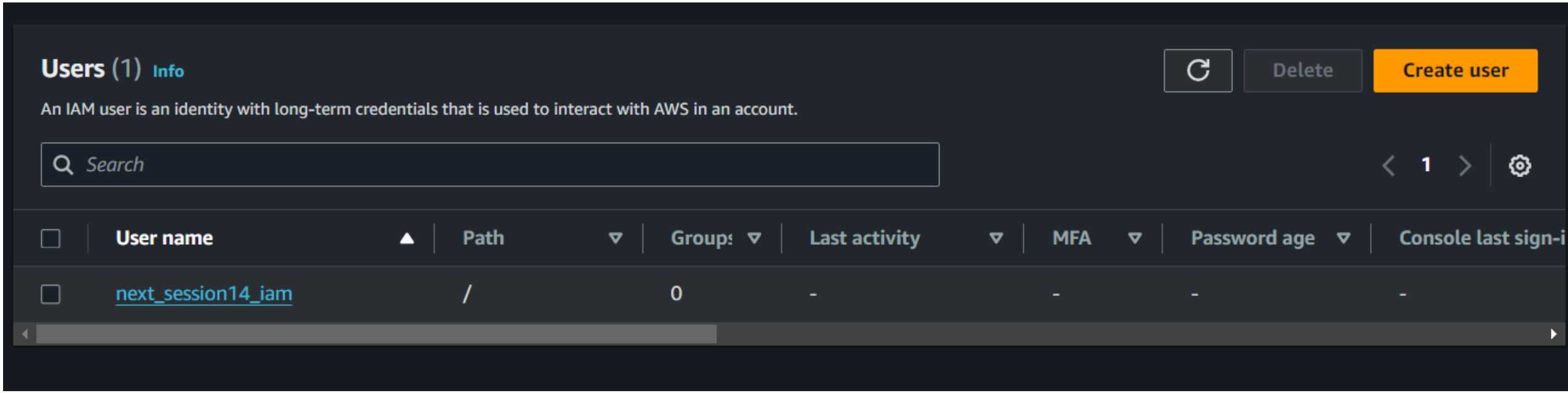
S3: IAM 생성

유저 생성!



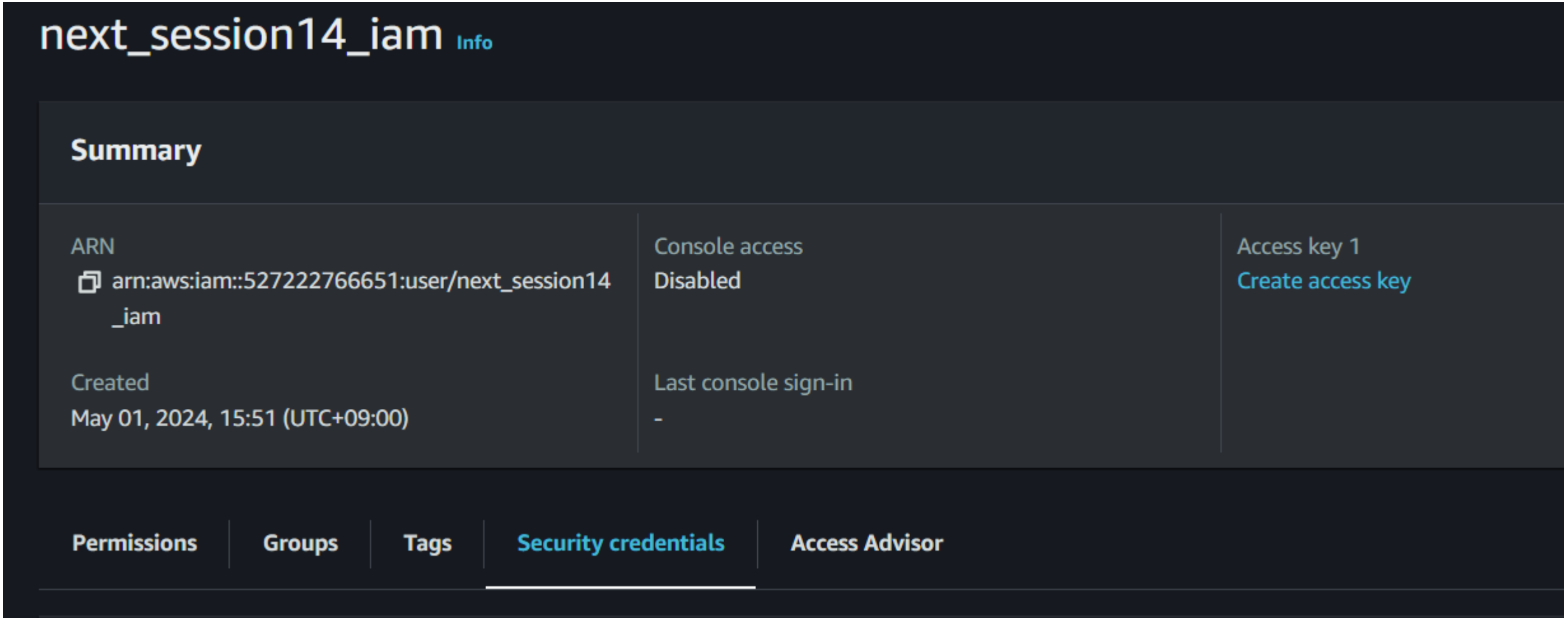
S3: IAM 생성

이제 django 서버에서 접속할 수 있도록 Programmatic Call을 허용하러 가즙시다



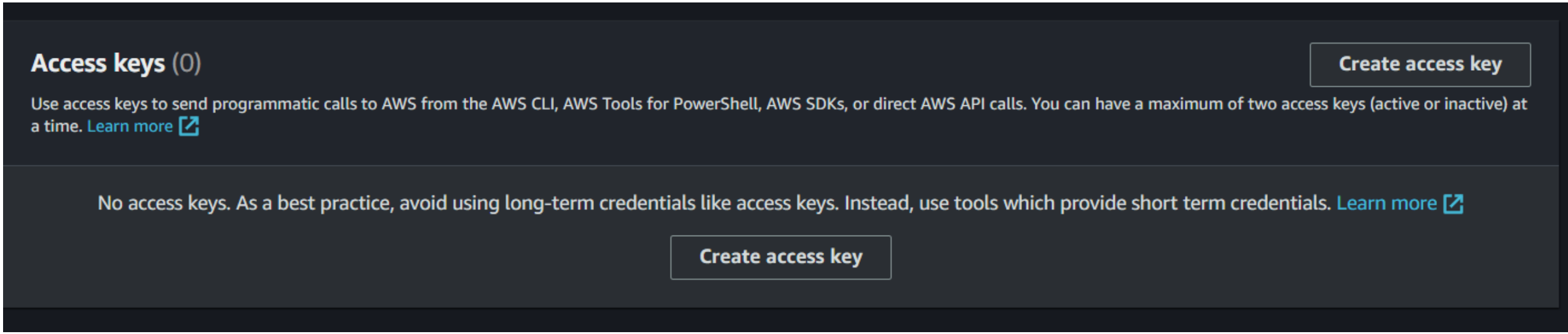
S3: IAM 생성

이제 django 서버에서 접속할 수 있도록 Programmatic Call을 허용하러 가줍시다



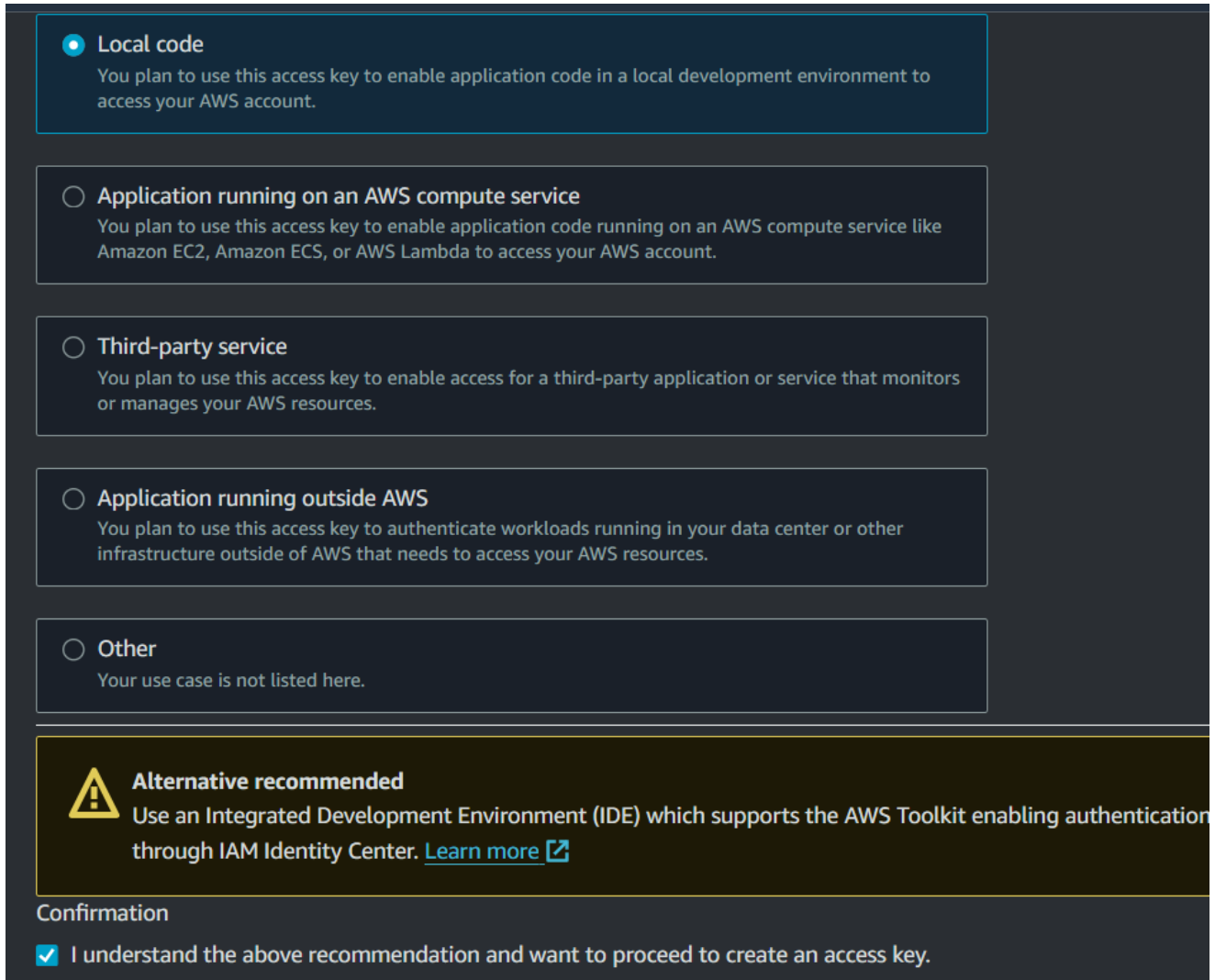
S3: IAM Access Key 생성

이제 django 서버에서 접속할 수 있도록 Programmatic Call을 허용하러 가즙시다

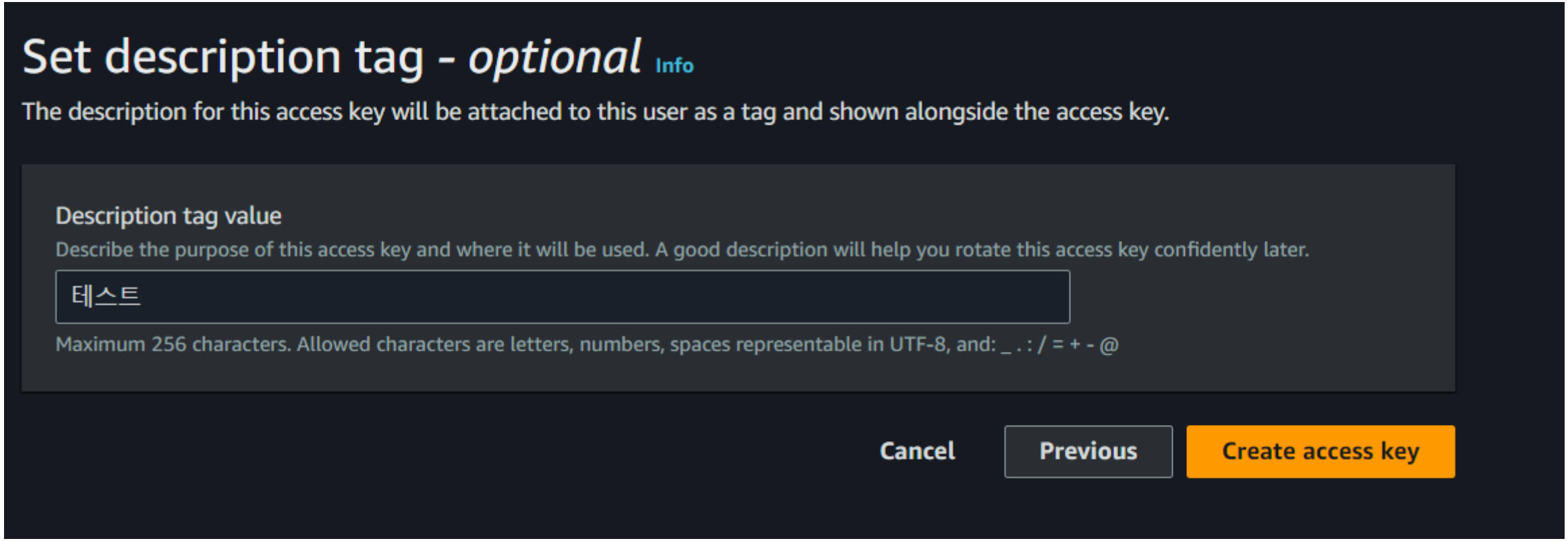


S3: IAM Access Key 생성

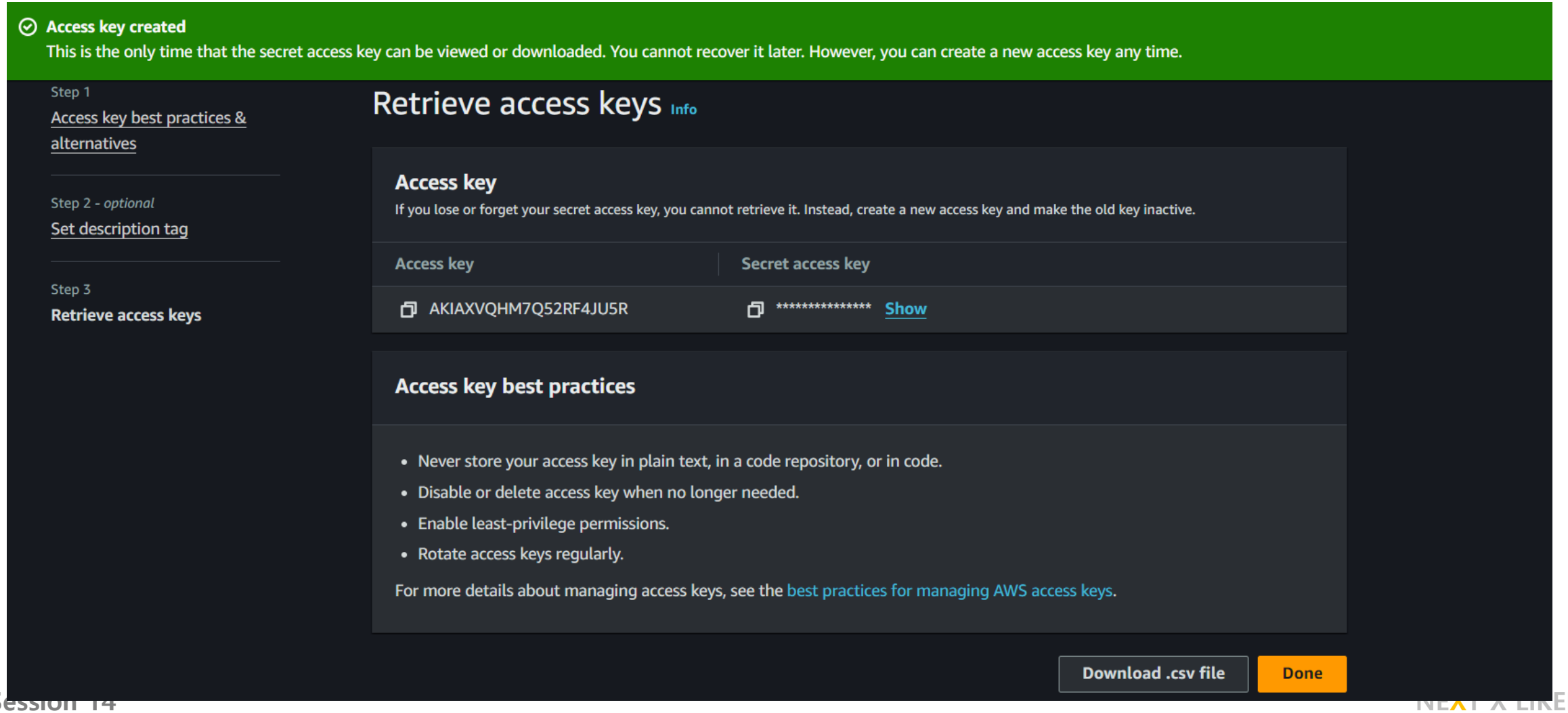
원래는 Application running on an AWS compute service를 하고, 사실 User가 아닌, role을 생성하고 이걸 EC2에 엮어서 사용하는 것이 베스트인데, 지금은 API 사용을 배우는 상황이므로 Credential을 생성합시다.



S3: IAM Access Key 생성



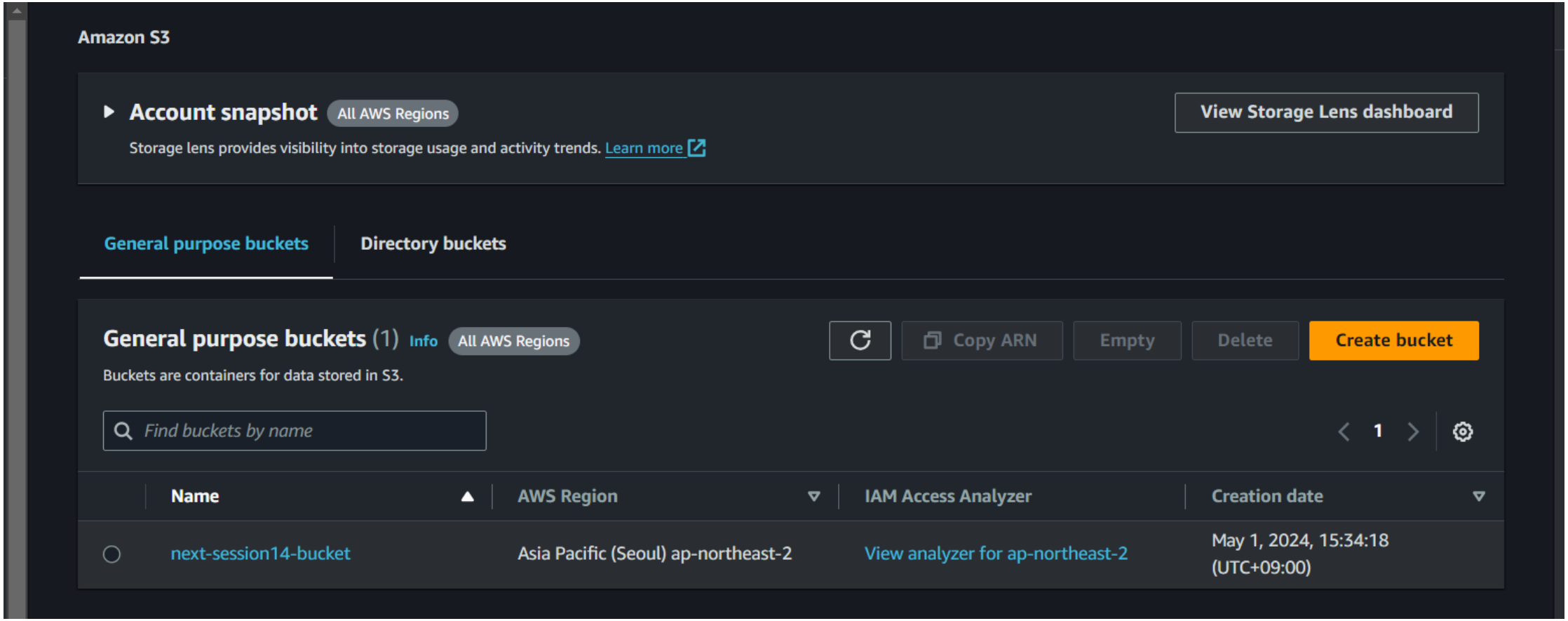
S3: IAM Access Key 생성



S3: 버킷 권한 설정

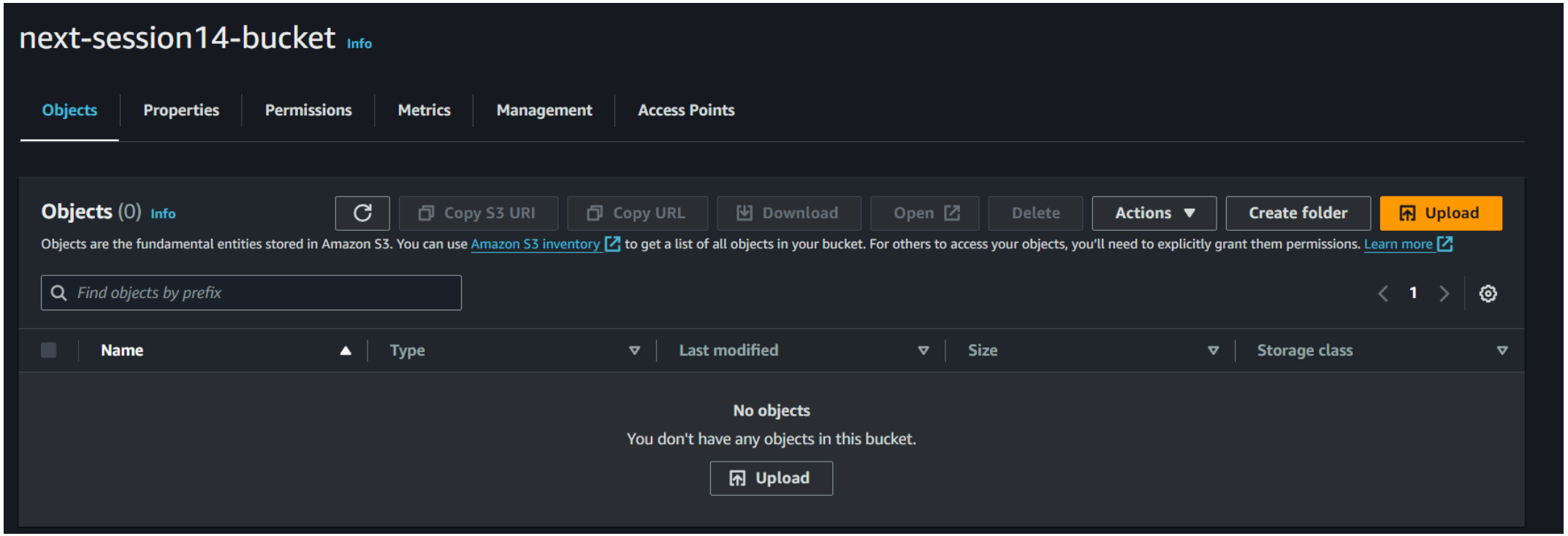
S3로 이동해서 버킷 설정 창으로 들어가기

https://ap-northeast-2.console.aws.amazon.com/s3/home?region=ap-northeast-2



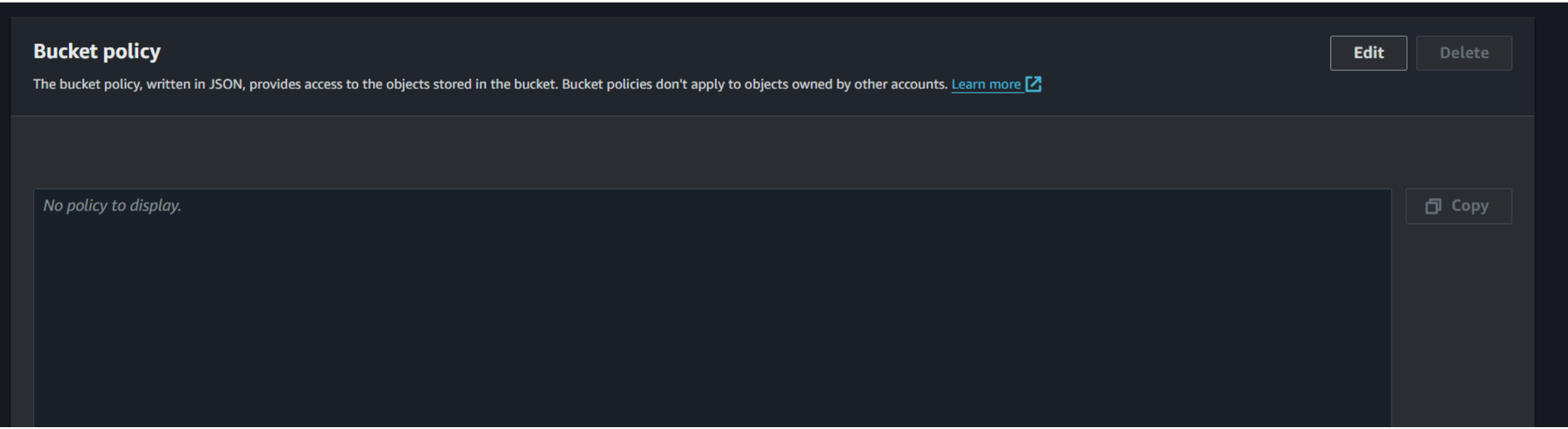
S3: 버킷 권한 설정

Permissions 눌러주세요!



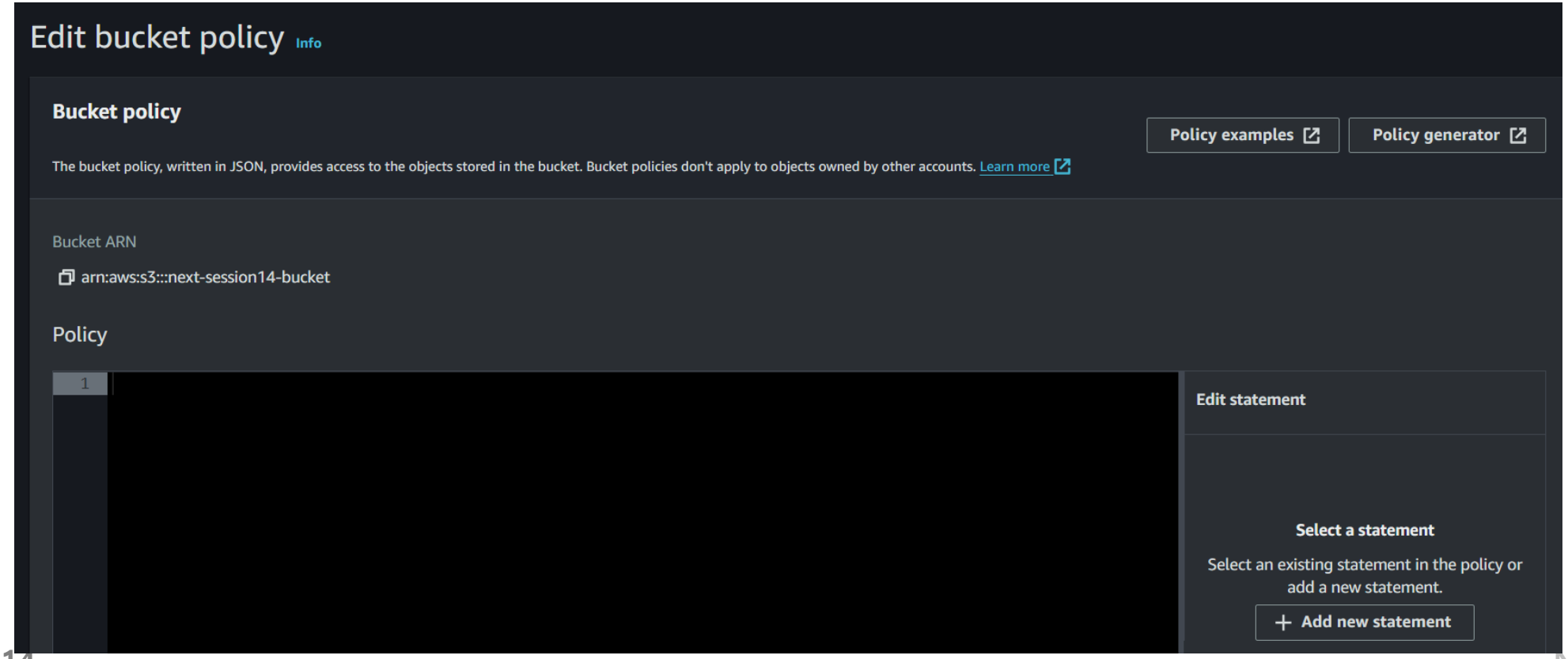
S3: 버킷 권한 설정

버킷 정책에서 수정을 눌러서 보기 권한은 익명의 모두에게 제공하지만, 업로드 업데이트 같은 권한은 서버에서만 가능하도록 정할 것입니다.



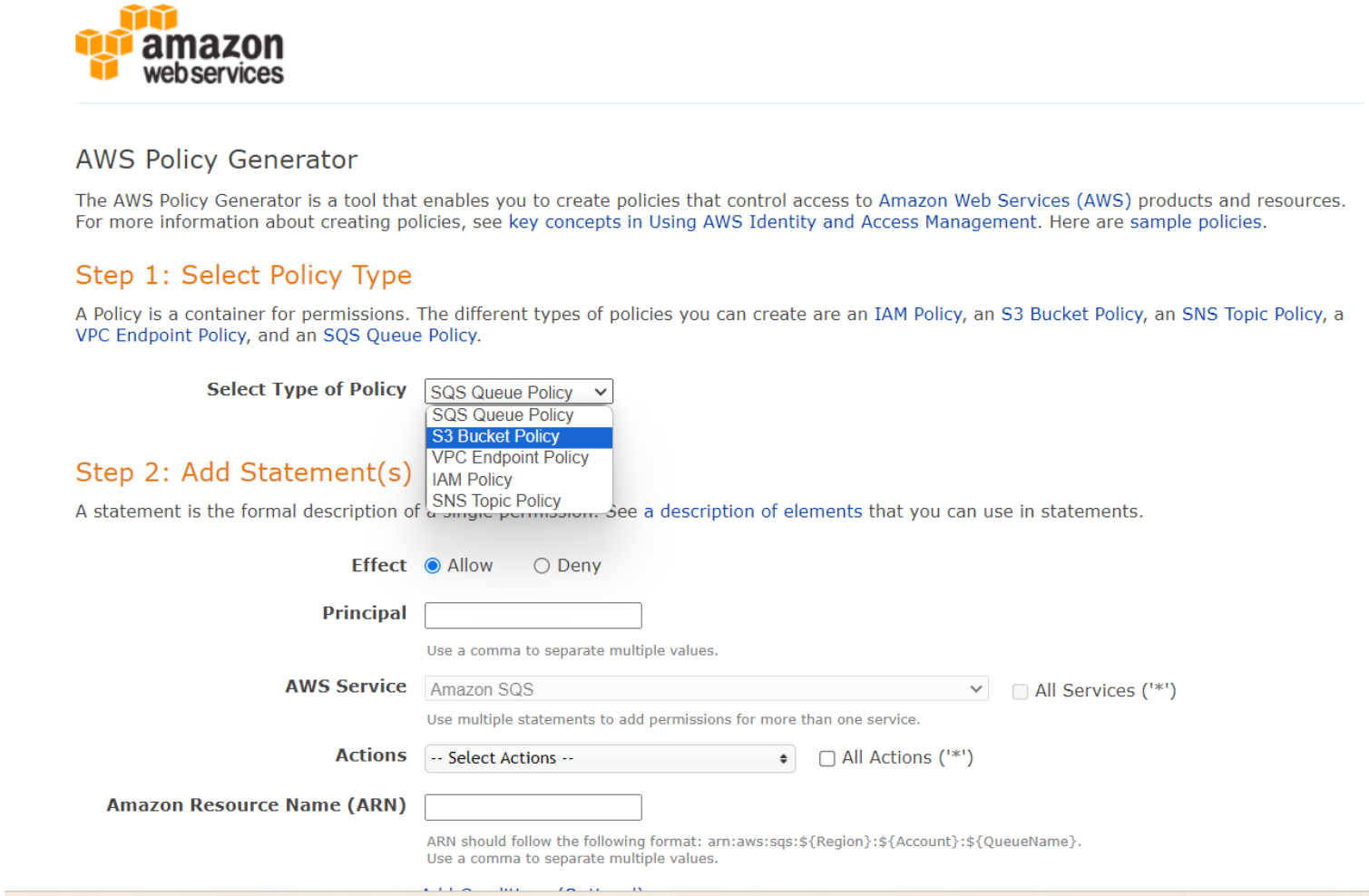
S3: 버킷 권한 설정

정책 생성기를 눌러주세요!



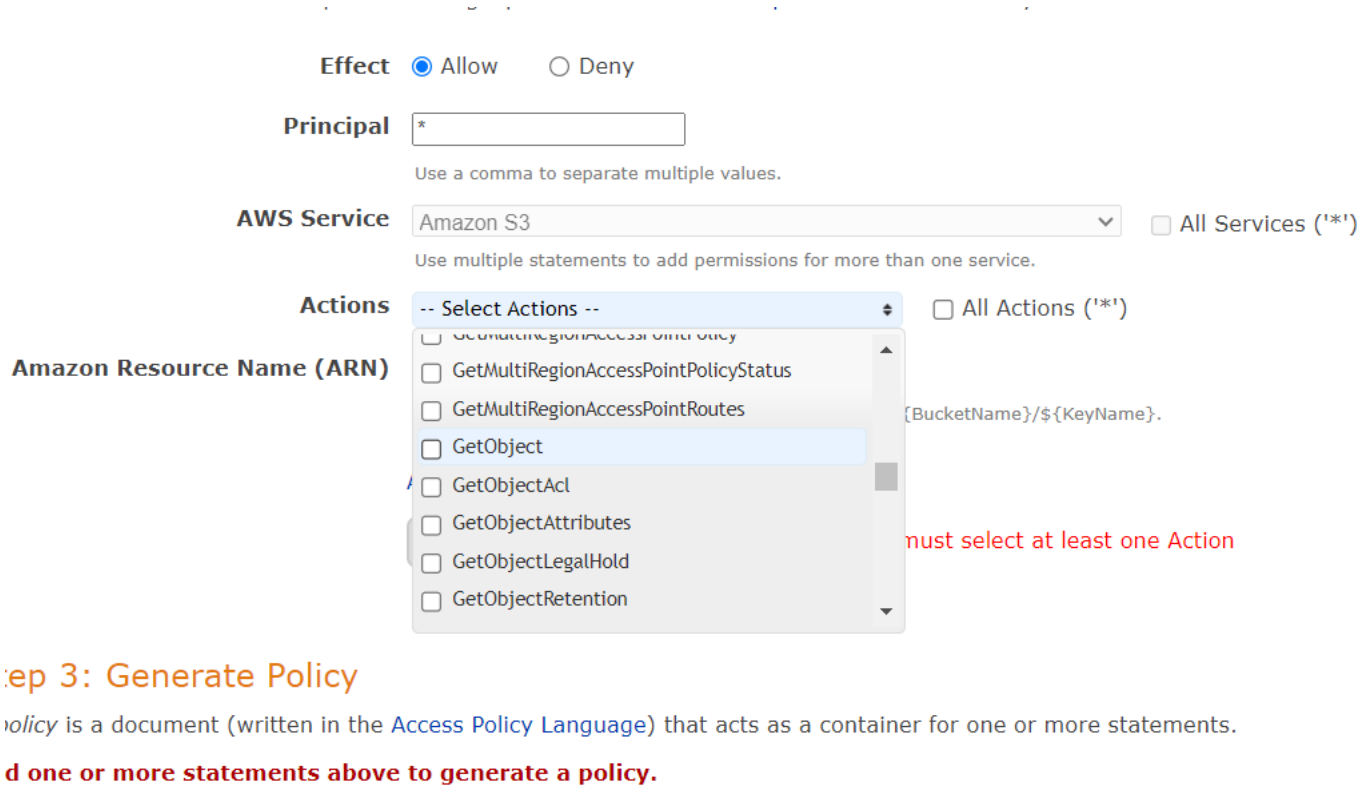
S3: 버킷 권한 설정

S3 버킷에 대한 정책 설정임을 명시하고



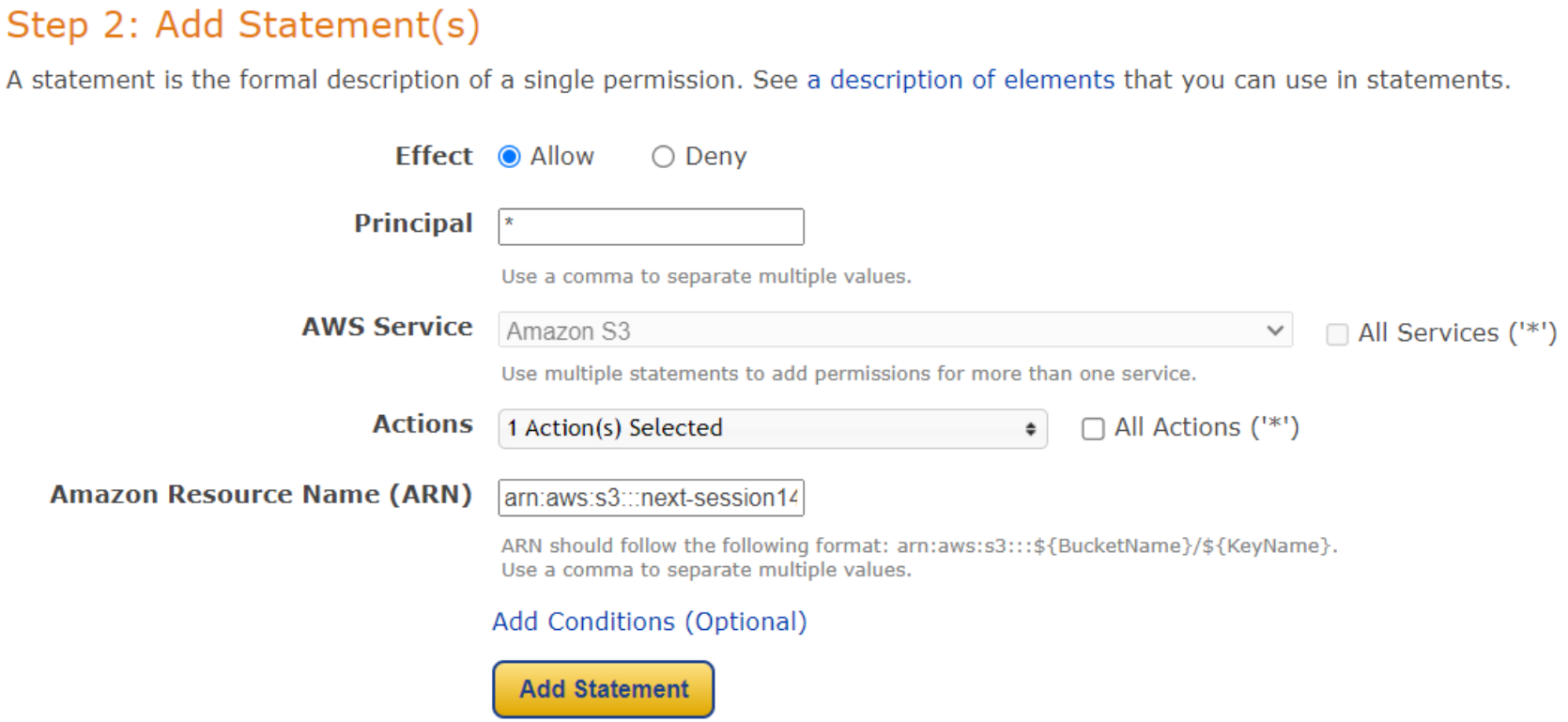
S3: 버킷 권한 설정

S3 내의 파일에 대한 읽기 권한을 모두에게 제공하는 것으로 설정합니다.
파일 = Object



S3: 버킷 권한 설정

해당 권한이 사용될 리소스를 설정합니다. ARN에 아래 형식에 맞게 작성하고 Add Statement를 써주세요



S3: 버킷 권한 설정

이제 해당 정책에 맞는 설정을 생성해주세요

You added the following statements. Click the button below to Generate a policy.

Principal(s)	Effect	Action	Resource	Conditions
• *	Allow	• s3:GetBucketPolicy	arn:aws:s3:::next-session14-bucket/*	*None*

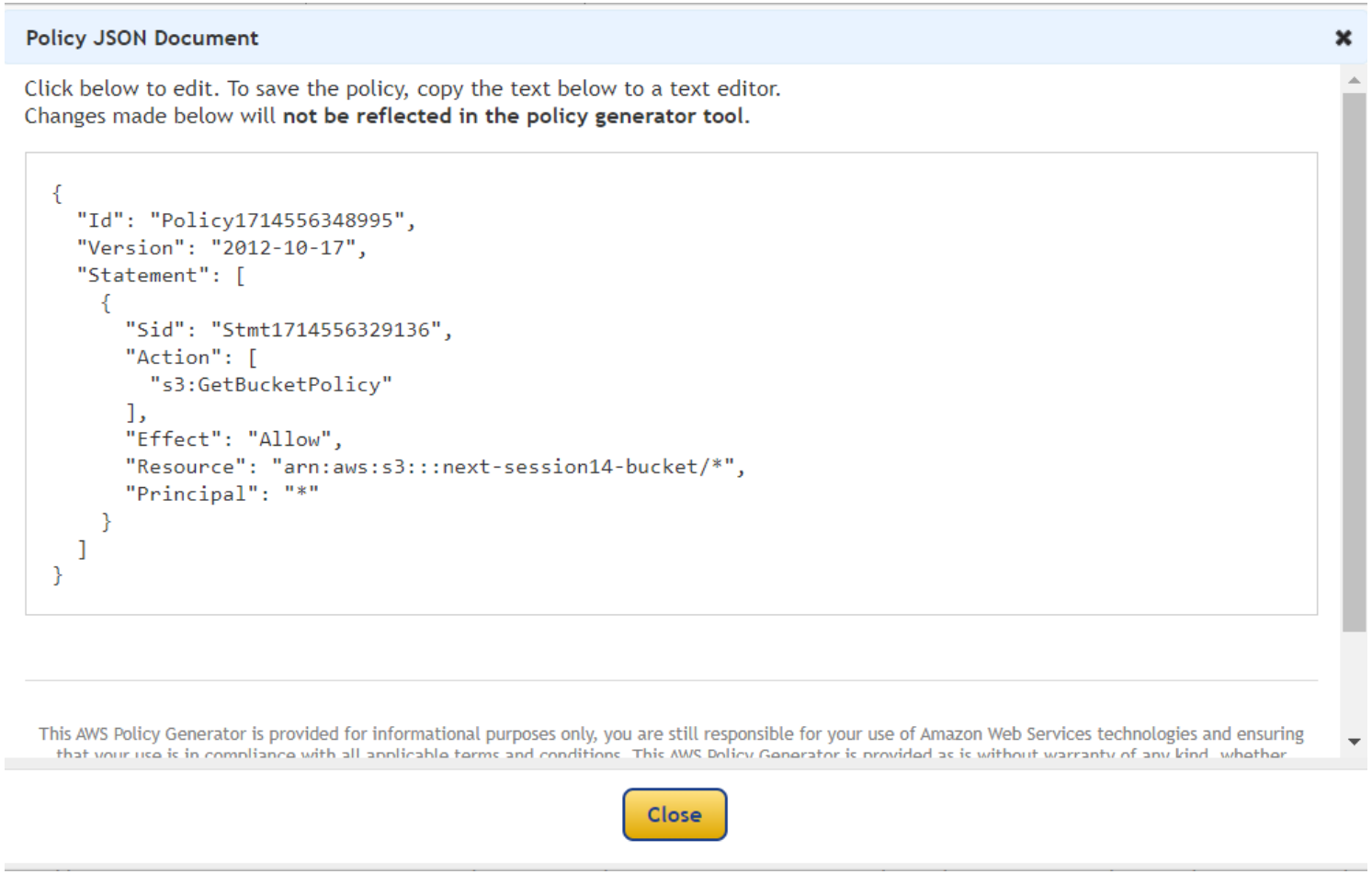
Step 3: Generate Policy

A *policy* is a document (written in the Access Policy Language) that acts as a container for one or more statements.

Generate Policy Start Over

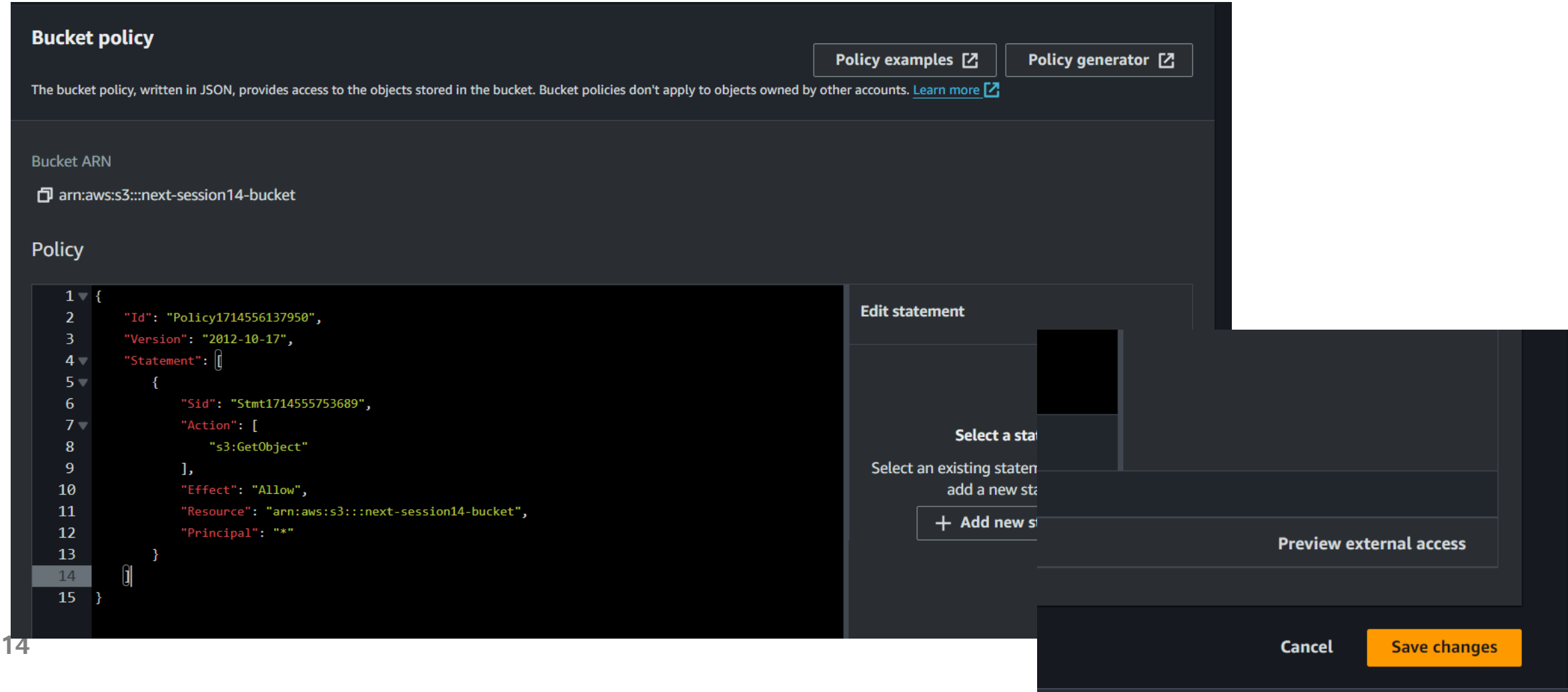
S3: 버킷 권한 설정

정책 JSON을 복사하고,



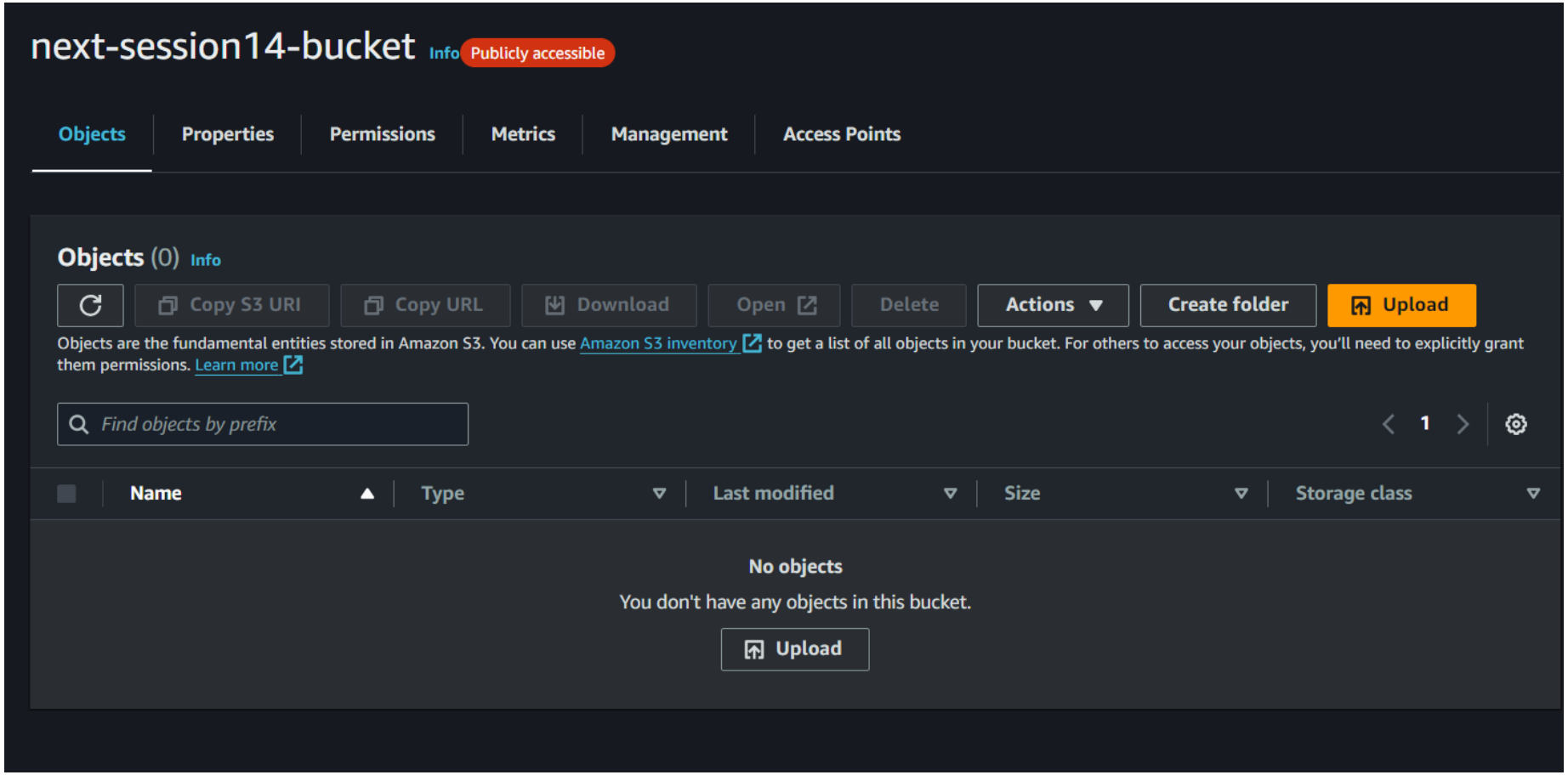
S3: 버킷 권한 설정

S3 버킷 정책에 붙여넣어주세요. 다했으면 아래에 save!



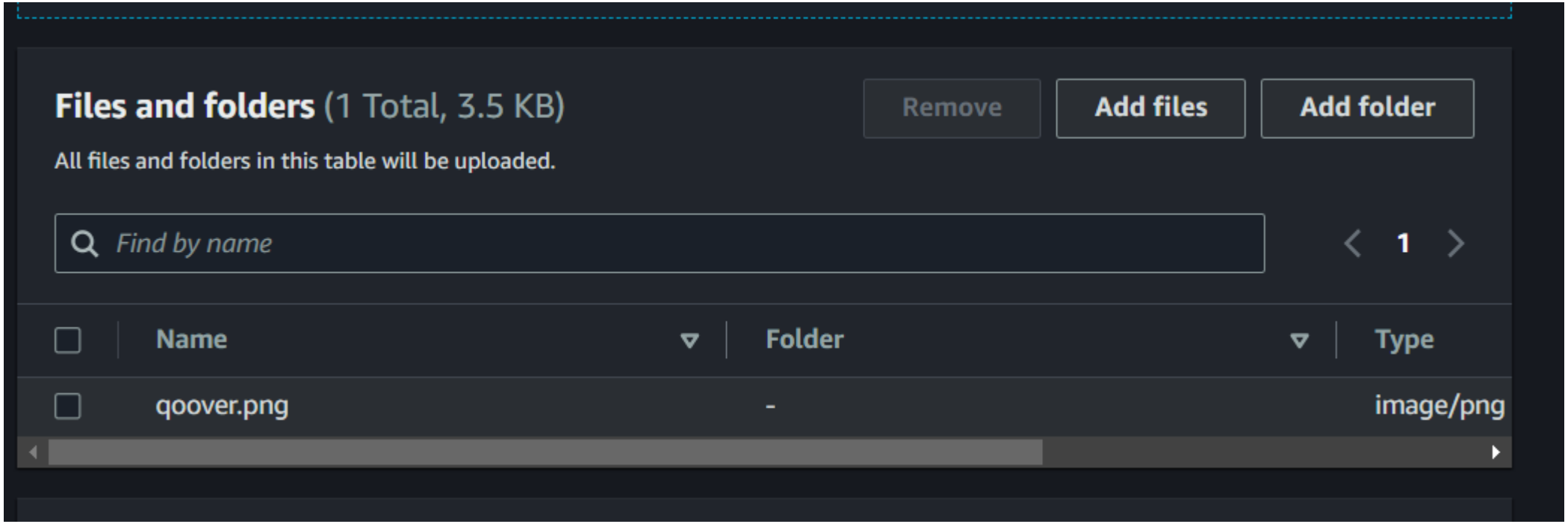
S3: 버킷 테스트

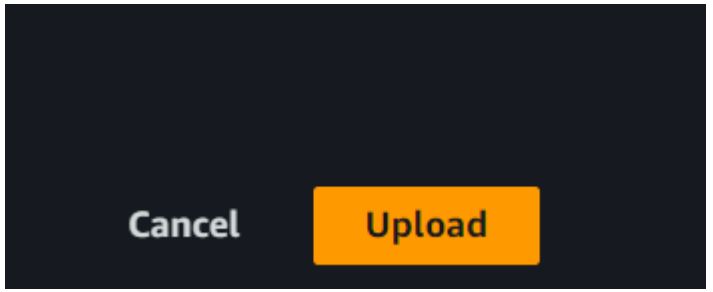
이제 다시 버킷 메인으로 돌아와서, 테스트용 이미지를 하나 올려보죠



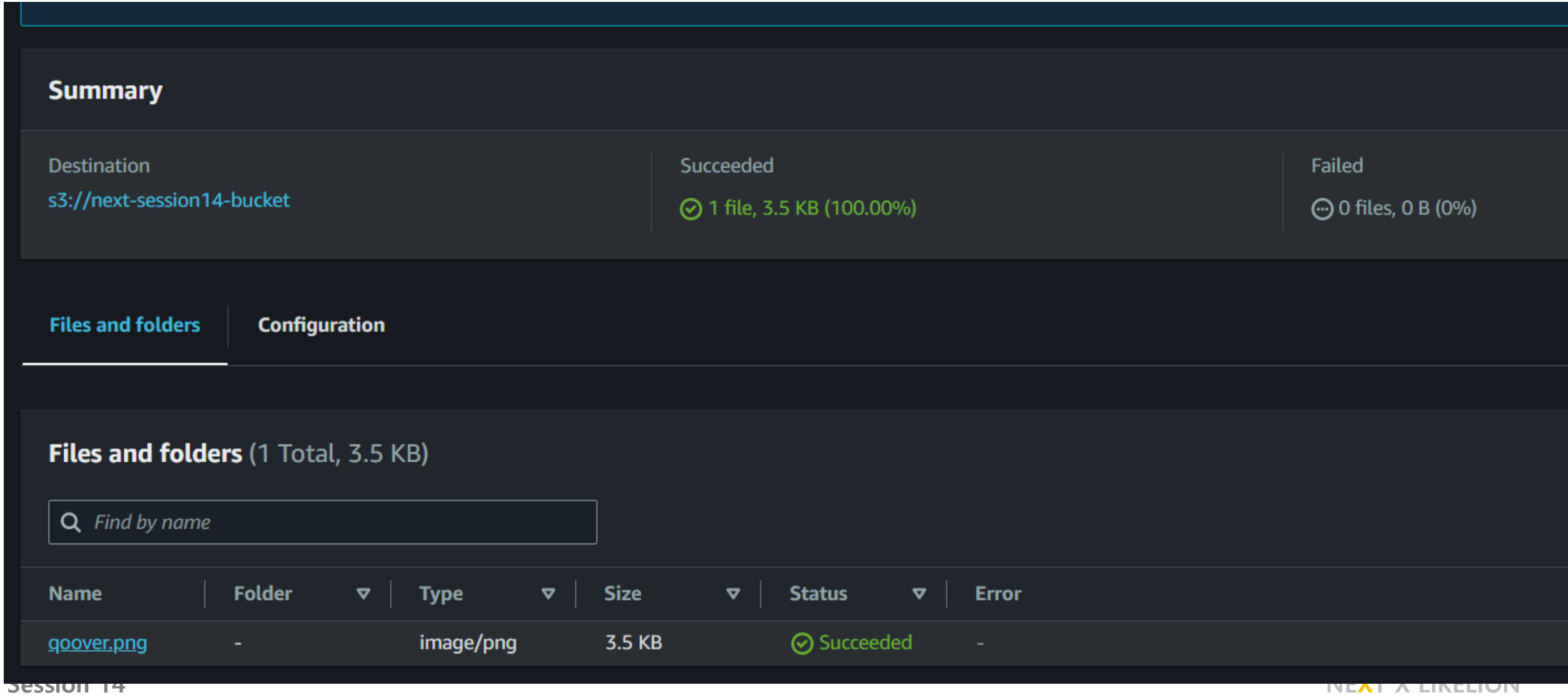
S3: 버킷 테스트

이제 다시 버킷 메인으로 돌아와서, 테스트용 이미지를 하나 올려보죠

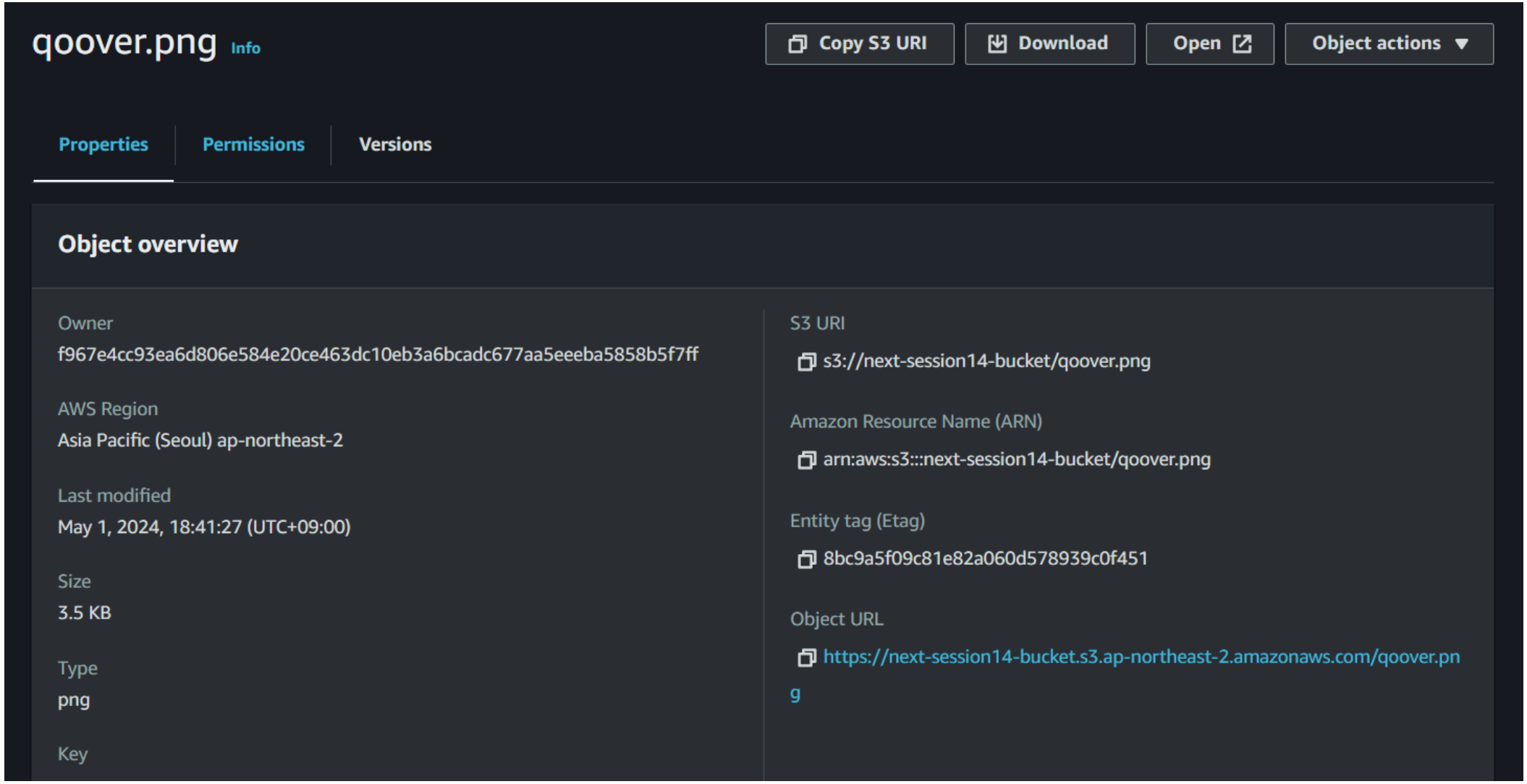




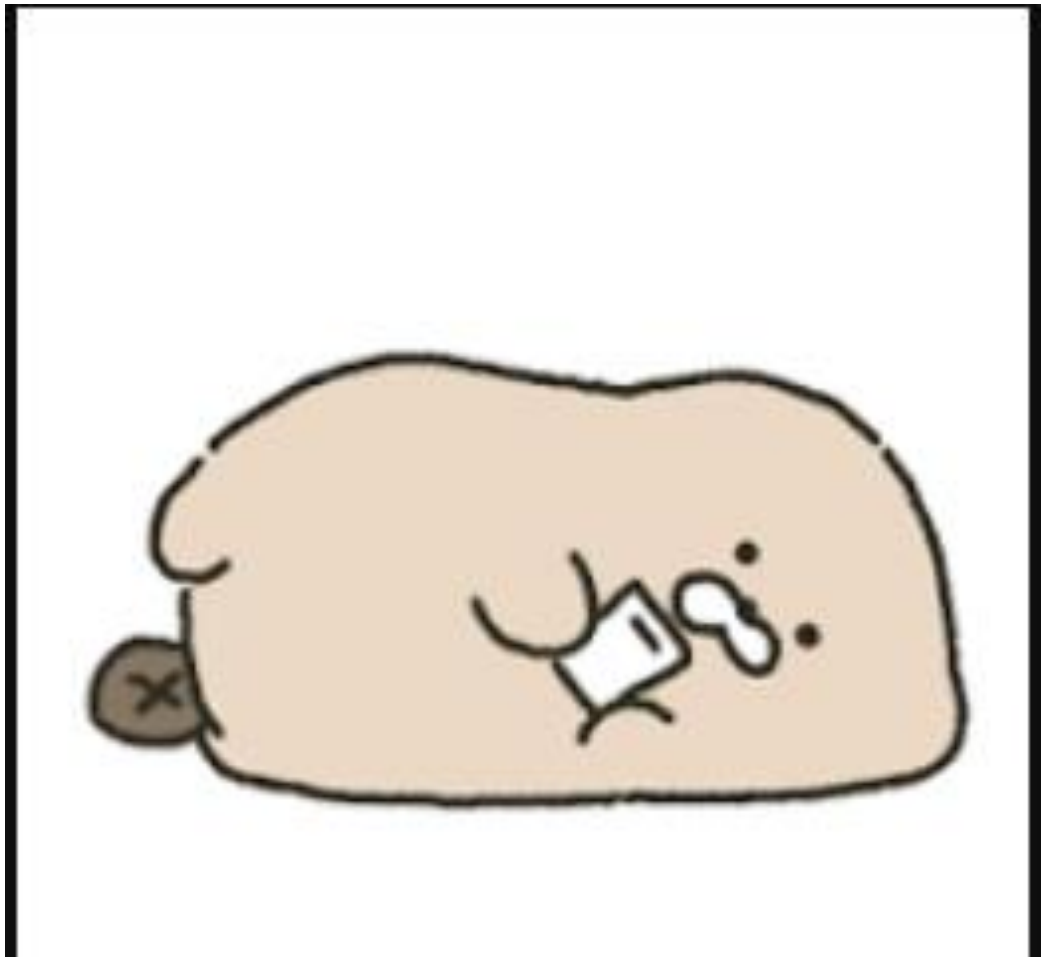
S3: 버킷 테스트


Summary
Destination
s3://next-session14-bucket
Succeeded
1 file, 3.5 KB (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 Total, 3.5 KB)
Find by name
Name
Folder
Type
Size
Status
Error
qoover.png
-
image/png
3.5 KB
Succeeded
-

S3: 버킷 테스트


qoover.png Info
Copy S3 URI
Download
Open
Object actions
Properties
Permissions
Versions
Object overview
Owner
f967e4cc93ea6d806e584e20ce463dc10eb3a6bcadc677aa5eeeba5858b5f7ff
AWS Region
Asia Pacific (Seoul) ap-northeast-2
Last modified
May 1, 2024, 18:41:27 (UTC+09:00)
Size
3.5 KB
Type
png
Key
S3 URI
s3://next-session14-bucket/qoover.png
Amazon Resource Name (ARN)
arn:aws:s3:::next-session14-bucket/qoover.png
Entity tag (Etag)
8bc9a5f09c81e82a060d578939c0f451
Object URL
https://next-session14-bucket.s3.ap-northeast-2.amazonaws.com/qoover.png

S3: 버킷 테스트



S3: Django 연동

전에 제가 올려놓은 Django 프로젝트의 조금 수정 버전을 그대로 차용하겠습니다.

다만, 시간을 절약하기 위해서 이번 세션에서 중요하다고 판단되지 않은 내용은 제가 임의로 작성을 완료했고, 여러분들은 필수적인 내용만 함께 실습하면 됩니다.

https://github.com/newxxson/next-session-11-hw.git

오늘은 배포까지 하기에는 시간이 부족하므로 로컬에서 진행합니다.

오늘 할 일
- Static storage S3로 이동 및 관련 파일 전체 업로드
- File storage S3로 이동
- 블로그 글 쓸 때 사진도 업로드 가능하도록 수정하고, S3에 업로드하고 Url 제공하기

S3: Static Storage 로컬에서 S3로 변경

- 현재 Static은 다 여러분 각자 app 아래의 static 아래에 저장되어 있고, django에서 알아서 html에 주입해줍니다.
- 이걸 배포할 때 nginx에서 사용할 수 있도록 한 곳에 모으게 된다면 (python manage.py collectstatic) 그때부터는 django.contrib.staticfiles.storage.StaticFilesStorage에서 관리하게 됩니다.

- 문제는 이러면 아직 WAS 서버에서 해당 파일들을 제공하는 것이므로, 비효율적입니다.
- S3로 분리해보겠습니다.

- 이거 좀 해주세요
- $pip install django, pytz, boto3 django-storages django-environ

S3: 보안 정보를 담는 곳 .env

접속 키, 아이디, 주소 등의 정보는 서버 코드에 하드 코딩하면 당연히 github에 올렸을 때 심각한 보안 문제를 야기할 수 있습니다. 그래서 보통 이런 중요 정보들은 .env라는 파일을 별도로 만들어 함께 보관하고, 실제 코드에서는 .env 파일에서 변수로 불러와 사용하는 식으로 보안을 강화합니다.

오늘은 그래서 S3를 접속하기 위한 정보들을 .env에 저장할 것입니다!

Base_auth/settings.py에서 S3를 연동하기 위한 보안 설정과 추가 설정을 진행합니다.
전에 다운 받은 Access Key를 찾아주세요!

찾은 다음에 Vscode에서 루트 디렉토리에 .env라는 파일을 작성한 뒤에 다음 코드를 작성해주세요

```
AWS_S3_ACCESS_KEY_ID=
AWS_S3_SECRET_ACCESS_KEY=
AWS_STORAGE_BUCKET_NAME=<여러분 버킷 이름 next-session14-~>
AWS_S3_REGION_NAME=ap-northeast-2
AWS_S3_SIGNATURE_VERSION=s3v4
```

S3: 보안 정보를 담는 곳 .env

이제 이걸 import하는 코드를 작성합니다. Base_auth/settings.py에 가주세요

```
from pathlib import Path
import os
import environ


# Build paths inside the project like this: BASE_DIR / 'subdir'
BASE_DIR = Path(__file__).resolve().parent.parent

env = environ.Env()
environ.Env.read_env(os.path.join(BASE_DIR, ".env"))

AWS_S3_ACCESS_KEY_ID = env("AWS_S3_ACCESS_KEY_ID")
AWS_S3_SECRET_ACCESS_KEY = env("AWS_S3_SECRET_ACCESS_KEY")
AWS_STORAGE_BUCKET_NAME = env("AWS_STORAGE_BUCKET_NAME")
AWS_S3_REGION_NAME = env("AWS_S3_REGION_NAME")
AWS_S3_SIGNATURE_VERSION = env("AWS_S3_SIGNATURE_VERSION")
```

S3: 보안 정보를 담는 곳 .env

이제 실제 설정을 하는 코드를 작성합니다. Base_auth/settings.py 더 아래를 가세요

```
AWS_S3_CUSTOM_DOMAIN = "%s.s3.%s.amazonaws.com" % (
    AWS_STORAGE_BUCKET_NAME,
    AWS_S3_REGION_NAME,
)

# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/5.0/howto/static-files/

STATIC_URL = "https://%s/static/" % AWS_S3_CUSTOM_DOMAIN
STATICFILES_STORAGE = "base_auth.storage.StaticStorage"
```

S3: 보안 정보를 담는 곳 .env

Boto default인 S3Boto3Storage를 대체하기 위한 커스텀 StaticStorage를 제작합니다.
base_auth/storage.py를 만들어주세요

```python
from storages.backends.s3boto3 import S3Boto3Storage


class StaticStorage(S3Boto3Storage):
    location = "static"
```

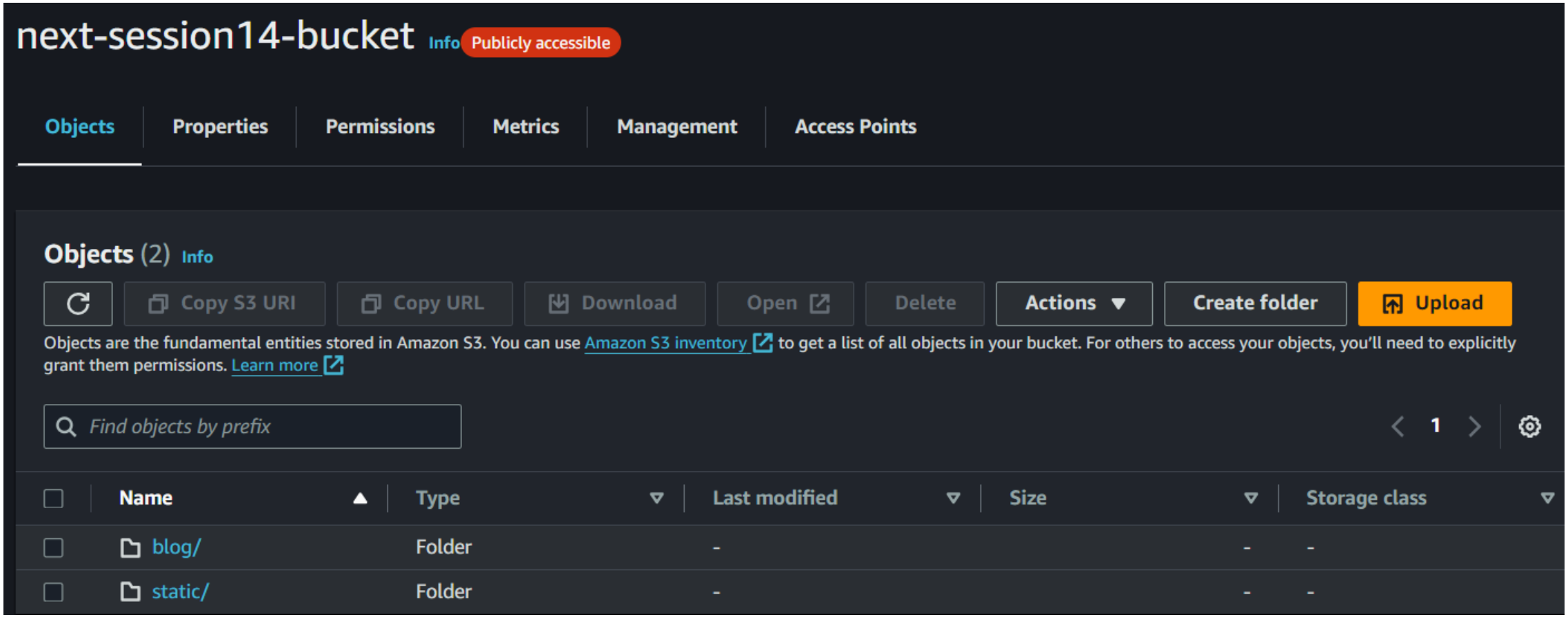
S3: 보안 정보를 담는 곳 .env

이제 static 파일을 한 곳에 모아 업로드하는 명령을 수행합니다!

```
(venv) PS C:\Users\duddn\PycharmProjects\testAWS1> python .\manage.py collectstatic

You have requested to collect static files at the destination
location as specified in your settings.

This will overwrite existing files!
Are you sure you want to do this?

Type 'yes' to continue, or 'no' to cancel: yes

130 static files copied.
```

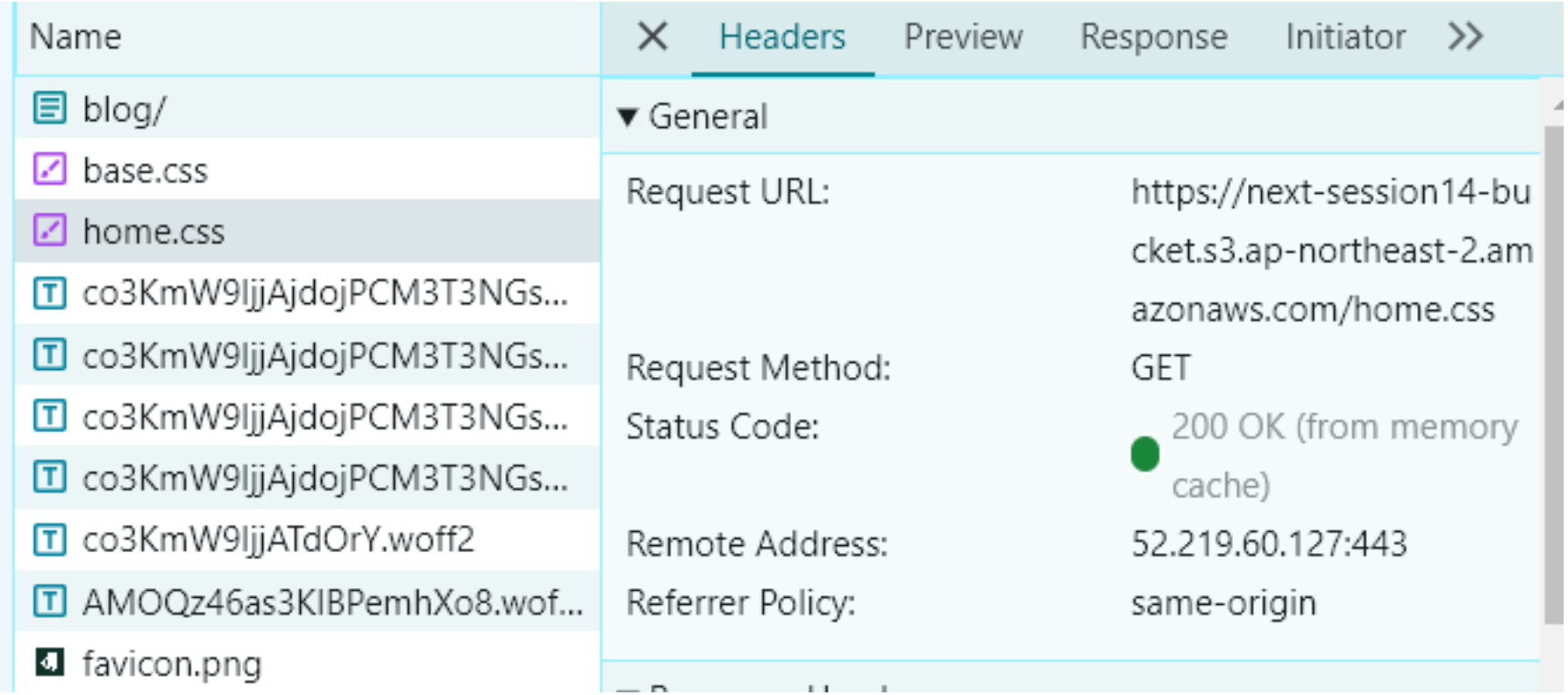
S3: 보안 정보를 담는 곳 .env

잘 올라가는 것을 확인할 수 있습니다!(blog는 안 보이는 것이 정상이에요)



S3: css 확인

Python runserver를 하고 css 주소를 확인해도 url이 제대로 설정되어 있는 것을 확인할 수 있습니다.



S3:django에서 사진 업로드 하기

이제 블로그 글에 사진을 추가할 수 있도록 파일 입출력을 s3에 연동하겠습니다.

이미 되어있는 것
- Html에서 관련 로직 및 구조
- Views에서 사용할 처리 관련 로직 (utils.py)
- Models.py에서 저장할 attribute

해야하는 것
- settings.py에서 s3 연동하기
- views.py에서 로직 구현하기

S3:django에서 사진 업로드 하기

```
You, 2 hours ago • Uncommitted changes
DEFAULT_FILE_STORAGE = "storages.backends.s3boto3.S3Boto3Storage"
```

```python
@login_required
def new(request):
    if request.method == "POST":
        title = request.POST["title"]
        content = request.POST["content"]

        new_post = Post(title=title, content=content, creator=request.user)

        image = request.FILES["image"]
        if image:
            image_name = get_new_file_name(image, request.user, title)

            saved_path = default_storage.save(image_name, image)
            image_url = default_storage.url(saved_path)

            new_post.image_url = image_url
        new_post.save()

        return redirect("detail", new_post.pk)

    return render(request, "new.html")
```

S3:django에서 사진 업로드 하기

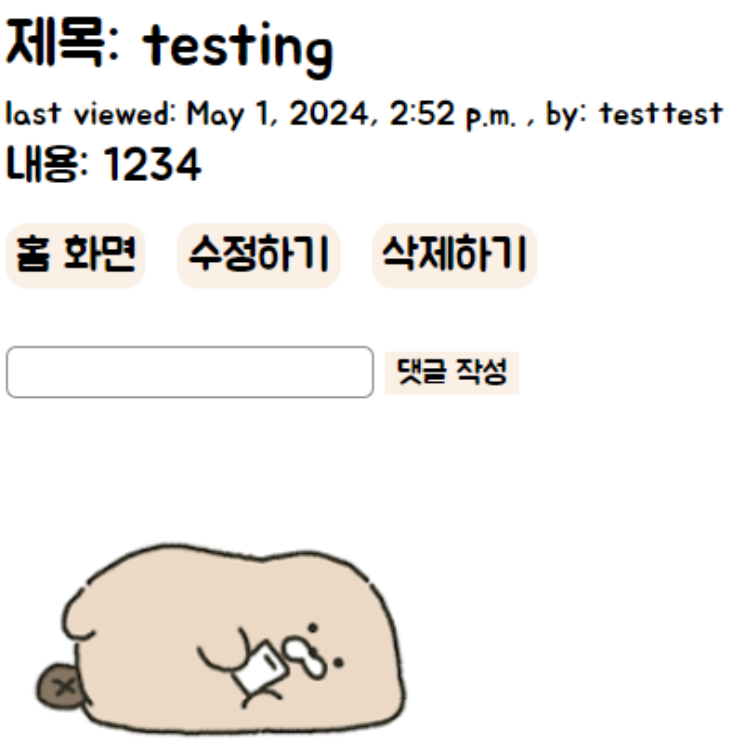
```python
@login_required
@check_is_creator_or_admin(Post, "post_pk")
def edit(request, post_pk):
    post = Post.objects.get(pk=post_pk)

    if request.method == "POST":
        title = request.POST["title"]
        content = request.POST["content"]
        Post.objects.filter(pk=post_pk).update(title=title, content=content)

        # Image handling
        image = request.FILES["image"]
        if image:
            image_name = get_new_file_name(image, request.user, title)

            saved_path = default_storage.save(image_name, image)
            image_url = default_storage.url(saved_path)

            post.image_url = image_url
            post.save()

        return redirect("detail", post_pk)

    return render(request, "edit.html", {"post": post})
```

S3:django에서 사진 업로드 하기

Makemigration, migrate 진행하고 서버를 구동해서 새로운 글 쓰기로 가면

S3:django에서 사진 업로드 하기

Makemigration, migrate 진행하고 서버를 구동해서 새로운 글을 쓰면



오늘의 과제

1. 오늘 만든 내용 서버에 배포하기 -> 직접 새로 레포 파서 만든 다음 올리고 clone ...

2. 댓글에 사진 첨부할 수 있도록 만들기

3. (선택)사진 삭제할 수 있도록 하기
   - 현재 저희 서버는 글을 작성한 후에 삭제하거나, 다른 사진으로 수정을 해도 서버에 올라간 이미지는 삭제되지 않습니다.
   - 해당 글을 삭제하거나, 사진을 새로운 사진으로 변경할 때 기존 사진을 삭제하는 로직을 구현해 주세요!

   - 현재 저희는 FileField를 사용하지 않고 UrlField를 사용하고 있기 때문에 on delete로직을 새로 구현해야 합니다.
   - https://docs.djangoproject.com/en/dev/topics/db/models/#overriding-predefined-model-methods
   - Delete 메서드를 수정해야겠죠? 이외에도, edit에서 온 파일과 기존 디비에서의 파일이 다른지를 검증하는 로직도 필요합니다.