Amazon S3

스무번째 세션

NEXT X LIKELION 이소영

S3

Amazon S3

Amazon Simple **S**torage **S**ervice

시작 전에 (1/3)

프로젝트 생성

- 프로젝트 폴더로 이동
- pipenv shell
- pipenv install django
- django-admin startproject file_upload
- cd file_upload
- django-admin startapp uploader
- settings.py INSTALLED_APPS에 uploader 추가

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'uploader',
]
```

시작 전에 (2/3)

- django에는 내장된 Storage Class가 있다.
- 파일을 다루는 데 필요한 기본적인 메서드(open, save, delete 등)이 정의되어 있다.
- 기본적으로 장고는 FileSystemStorageClass가 Storage Class를 상속받아 로컬 저장소 (하드드라이브)에 저장하는 기능을 가지고 있다.
- 하지만 우리는 로컬에다 저장하는 것이 아닌, 외부 저장소인 s3에 저장할 것이다!

시작 전에 (3/3)

패키지 다운로드

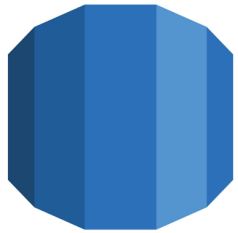
- pipenv install django-storages boto3
- settings.py INSTALLED_APPS에 storages 추가

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'uploader',
    'storages',
]
```

AWS란?

뭘까요

Amazon EC2



Amazon **RDS**

- 클라우드 컴퓨팅을 통해 유동적으로 컴퓨팅 리소스(서버컴퓨터, 데이터베이스 등)을 제공하는 서비스

씨익

S3

란 뭘까

클라우드 스토리지를 제공하는 서비스
간단히 말해 <외부 저장소/>

Amazon Simple **S**torage **S**ervice

저장소를 따로 두는 이유

왜 사용하는지

이미지나 문서를 비롯한 파일은 db에 직접 저장하기에는 크기가 너무 크다.

+

username, age 등의 정보와는 달리 담긴 정보를 조작할 일이 거의 없다.

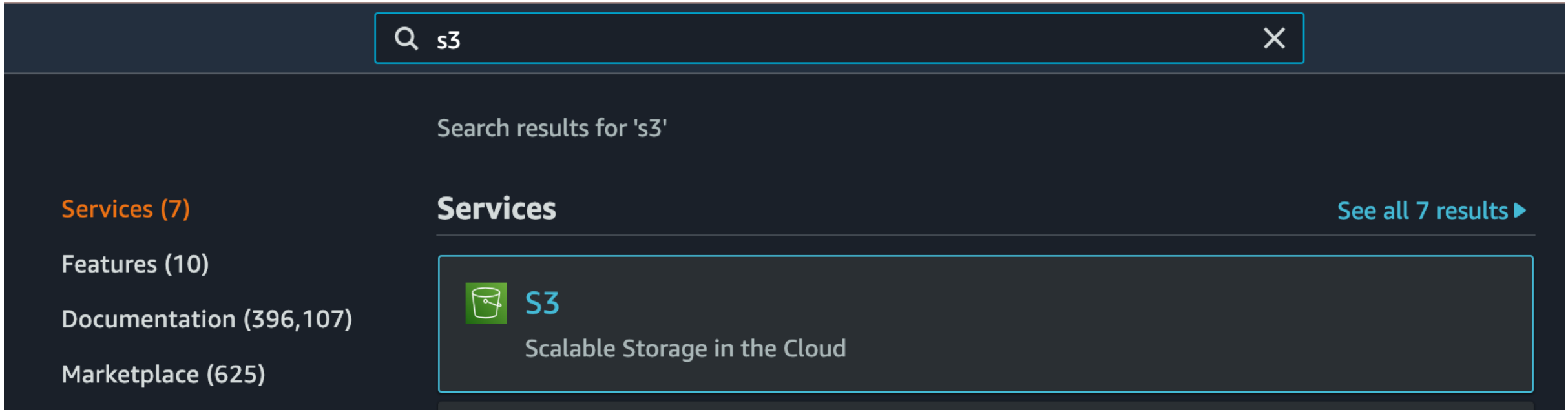
그래서 외부 저장소에 저장한다.

파일 업로드

어떻게 돌아가는가

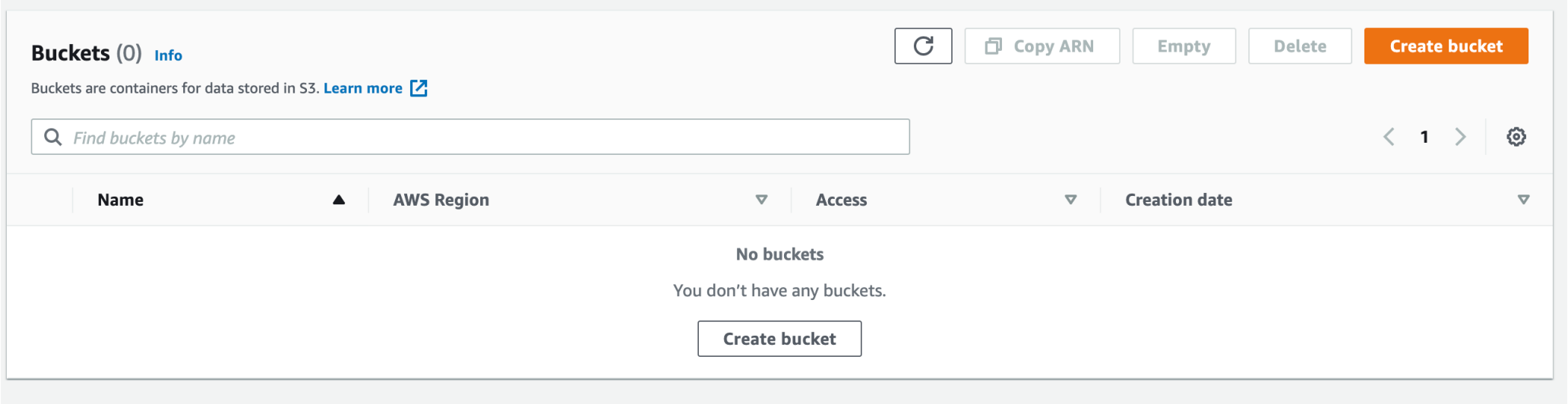
id	이미지 url	사진 이름
1	https://aws.s3/noschoolplz.jpg	학교 싫어
2	https://aws.s3/lovevacation.jpg	방학 좋아

S3 버킷 생성하기


s3
Search results for 's3'
Services (7)
Features (10)
Documentation (396,107)
Marketplace (625)
Services
See all 7 results
S3
Scalable Storage in the Cloud

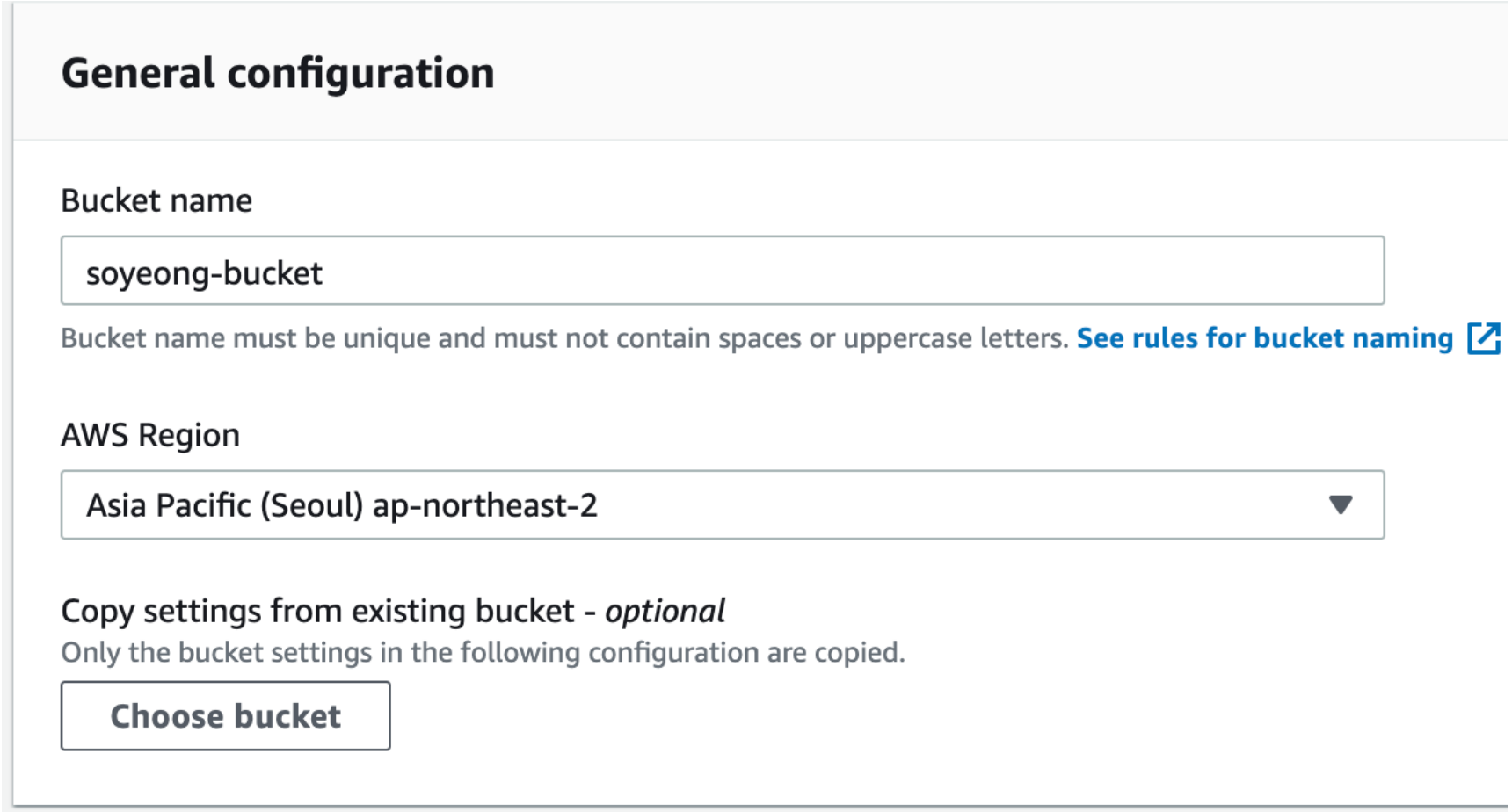
S3 버킷 생성하기

버킷만들기
create bucket



S3 버킷 생성하기

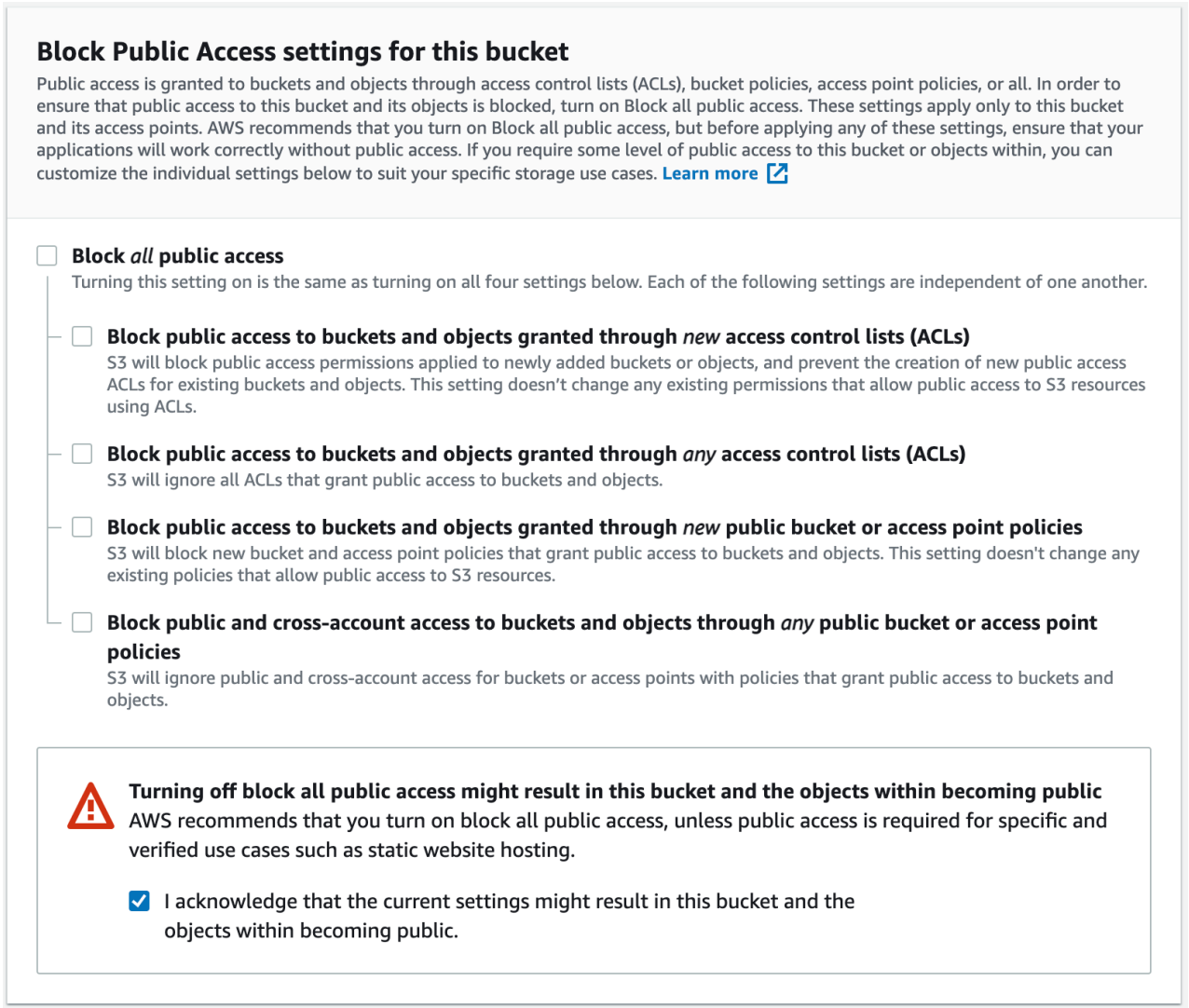
본인이 원하는 버킷 이름



S3 버킷 생성하기

체크 해제

체크



S3 버킷 생성하기

https://docs.aws.amazon.com/AmazonS3/latest/dev/example-bucket-policies.html

Granting read-only permission to an anonymous user

The following example policy grants the `s3:GetObject` permission to any public anonymous users. (For a list of permissions and the operations that they allow, see Amazon S3 actions.) This permission allows anyone to read the object data, which is useful for when you configure your bucket as a website and want everyone to be able to read objects in the bucket. Before you use a bucket policy to grant read-only permission to an anonymous user, you must disable block public access settings for your bucket. For more information, see Setting permissions for website access.

Warning

Use caution when granting anonymous access to your Amazon S3 bucket or disabling block public access settings. When you grant anonymous access, anyone in the world can access your bucket. We recommend that you never grant anonymous access to your Amazon S3 bucket unless you specifically need to, such as with static website hosting.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Sid":"PublicRead",
      "Effect":"Allow",
      "Principal": "*",
      "Action":["s3:GetObject","s3:GetObjectVersion"],
      "Resource":["arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"]
    }
  ]
}
```

클릭

S3 버킷 생성하기

버킷 정책 편집기

Edit bucket policy Info

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects other accounts. Learn more

Policy examples | Policy generator

Bucket ARN

arn:aws:s3:::soyeong-bucket

Policy

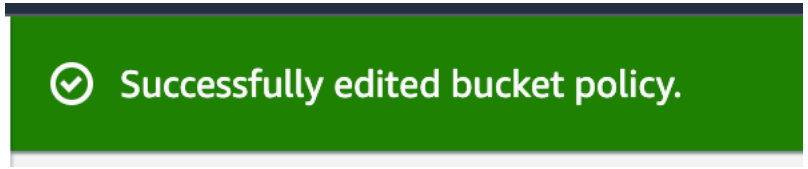
```
1  {
2    "Version":"2012-10-17",
3    "Statement":[
4      {
5        "Sid":"PublicRead",
6        "Effect":"Allow",
7        "Principal": "*",
8        "Action":["s3:GetObject","s3:GetObjectVersion"],
9        "Resource":["arn:aws:s3:::soyeong-bucket/*"]
10     }
11   ]
12 }
```

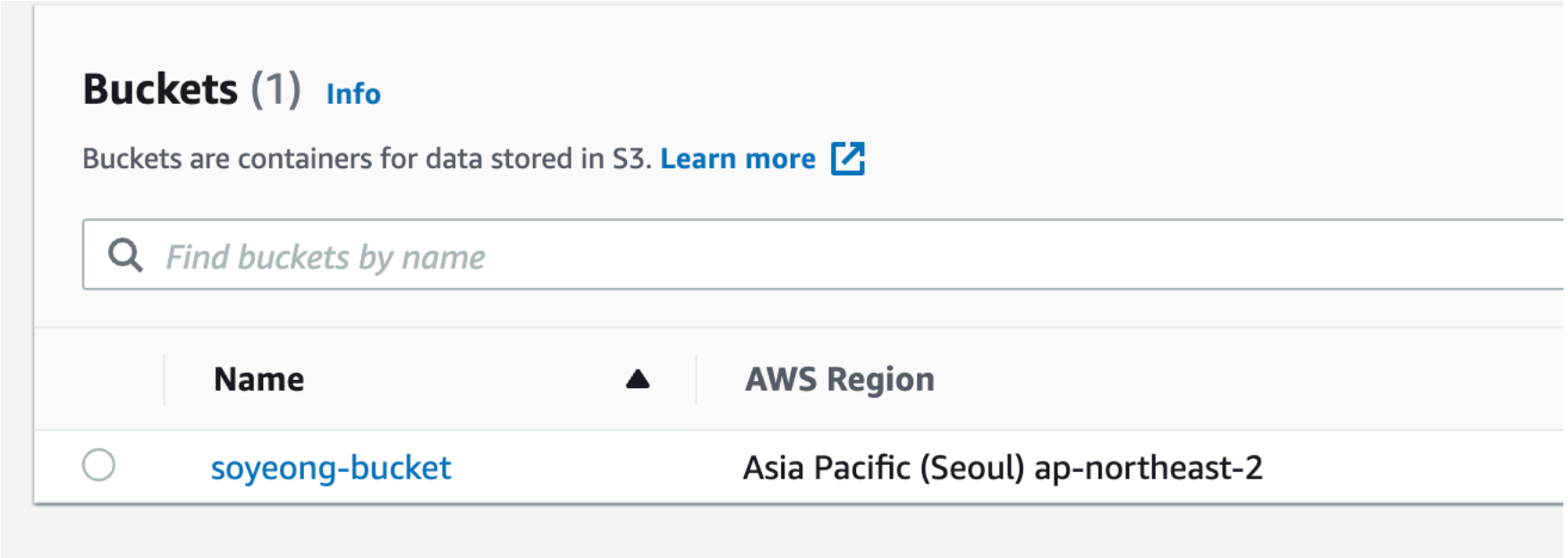
Save changes

클릭

S3 버킷 생성하기

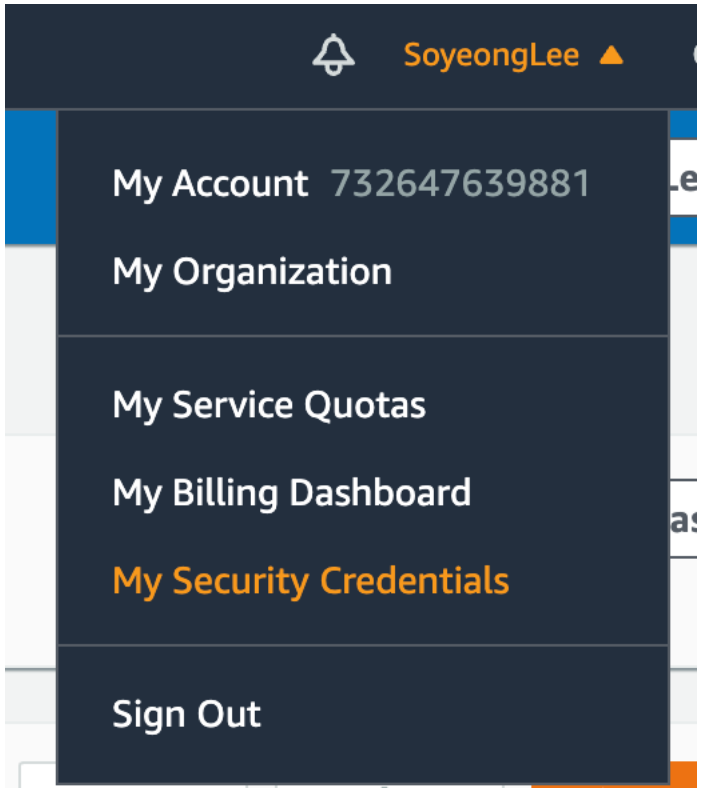
생성 완료



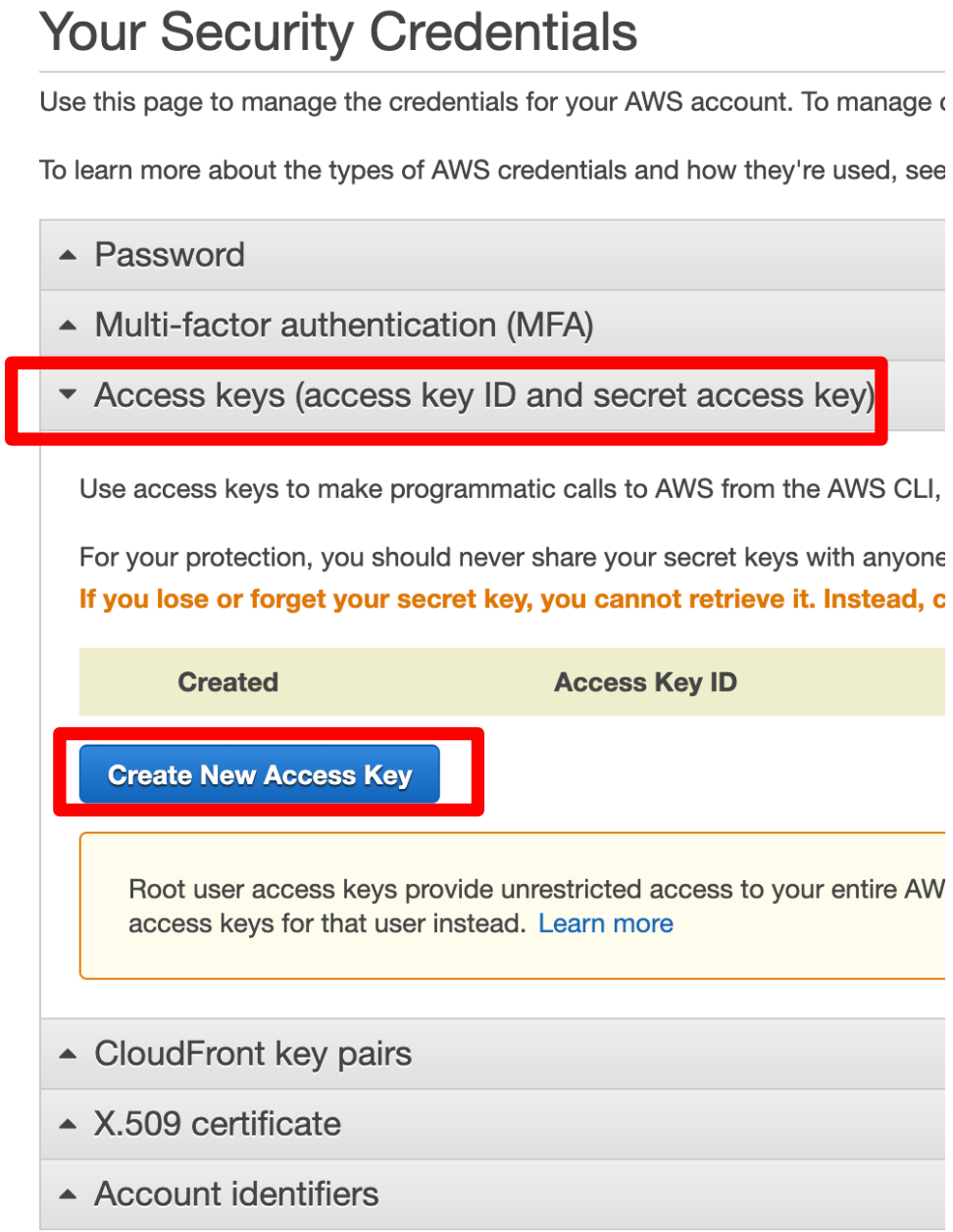


AWS 엑세스키 생성

내 보안 자격 증명



AWS 엑세스키 생성

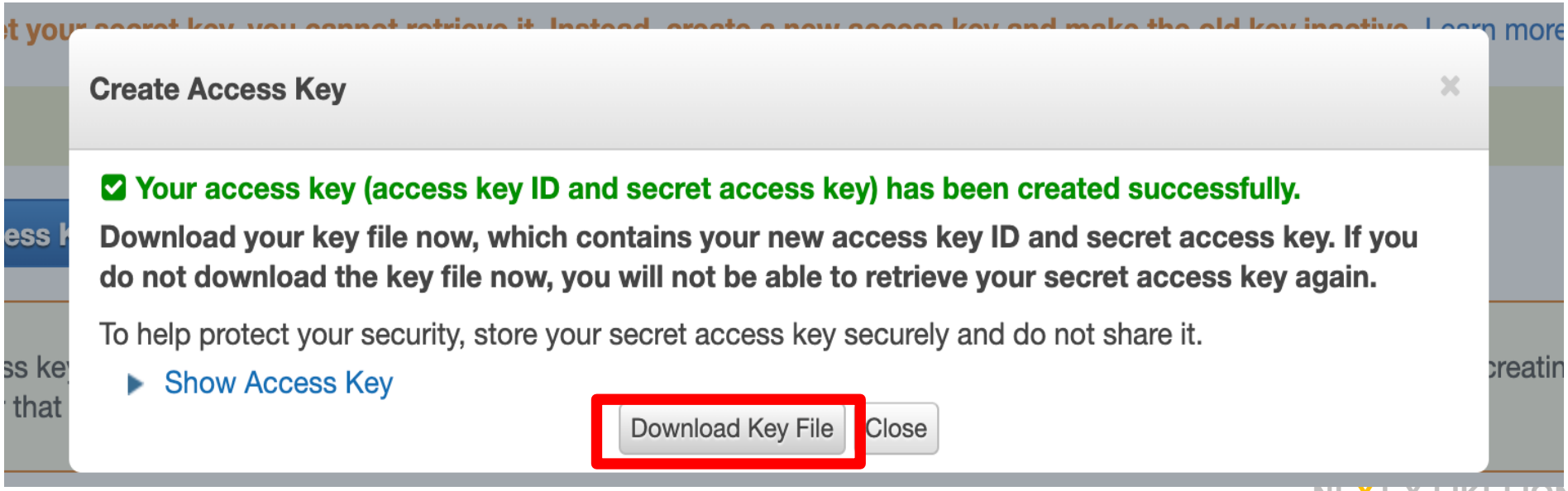

Your Security Credentials
Use this page to manage the credentials for your AWS account. To manage
To learn more about the types of AWS credentials and how they're used, see
Password
Multi-factor authentication (MFA)
Access keys (access key ID and secret access key)
Use access keys to make programmatic calls to AWS from the AWS CLI,
For your protection, you should never share your secret keys with anyone
If you lose or forget your secret key, you cannot retrieve it. Instead, c
Created
Access Key ID
Create New Access Key
Root user access keys provide unrestricted access to your entire AW
access keys for that user instead. Learn more
CloudFront key pairs
X.509 certificate
Account identifiers

AWS 엑세스키 생성

- 키파일 다운로드

- 절 대 로 외부에 노출되면 안됨!!
(깃헙, 친구, 부모 등)

- 한 번 다운받으면 다시 다운 불가능하니 잘 보관하세요



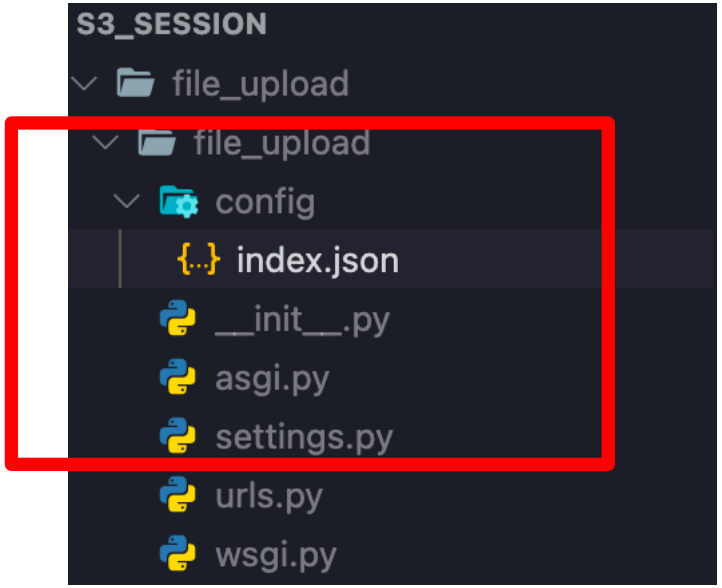
django 세팅

- settings.py에 aws 키를 올리면 안됩니다.
- 깃헙에 올릴 때 settings.py는 같이 올라가기 때문.

- 그렇다면???

config 폴더 생성

- 지금까지는 gitignore만 써봤지 키값을 숨겨보지는 않았습니다.
- config 폴더를 생성해서 환경설정 파일을 따로 관리해봅시다.
- file_upload/config/index.json



config/index.json

- 환경설정 파일을 따로 관리
- file_upload/config/index.json

settings.py

- import json

```python
from pathlib import Path
import os, json

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

with open(os.path.join(BASE_DIR, 'file_upload/config/index.json')) as f:
    secrets = json.loads(f.read())
```

settings.py

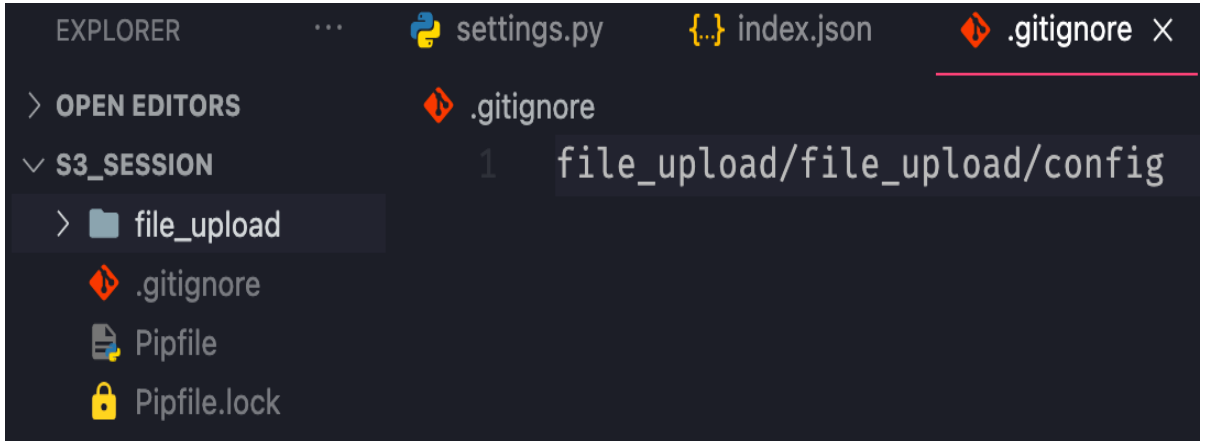
aws 설정 코드 추가

```
132  #S3 storage
133  DEFAULT_FILE_STORAGE = 'storages.backends.s3boto3.S3Boto3Storage'
134
135  # AWS Access
136  AWS_ACCESS_KEY_ID = secrets['AWS']['AWS_ACCESS_KEY']
137  AWS_SECRET_ACCESS_KEY = secrets['AWS']['SECRET_ACCESS_KEY']
138  AWS_STORAGE_BUCKET_NAME = secrets['AWS']['STORAGE_BUCKET_NAME']
139  AWS_S3_REGION_NAME = 'ap-northeast-2'
140  AWS_S3_FILE_OVERWRITE = False
```

gitignore

중요!

- gitignore: git이 기록하지 않는 파일/디렉토리 목록. 이곳에 기록된 파일이나 디렉토리는 vscode 상에서 짙은 회색으로 보인다.
- .gitignore은 .git 파일보다 아래에 위치해야한다.



색이 바뀌지 않았다!

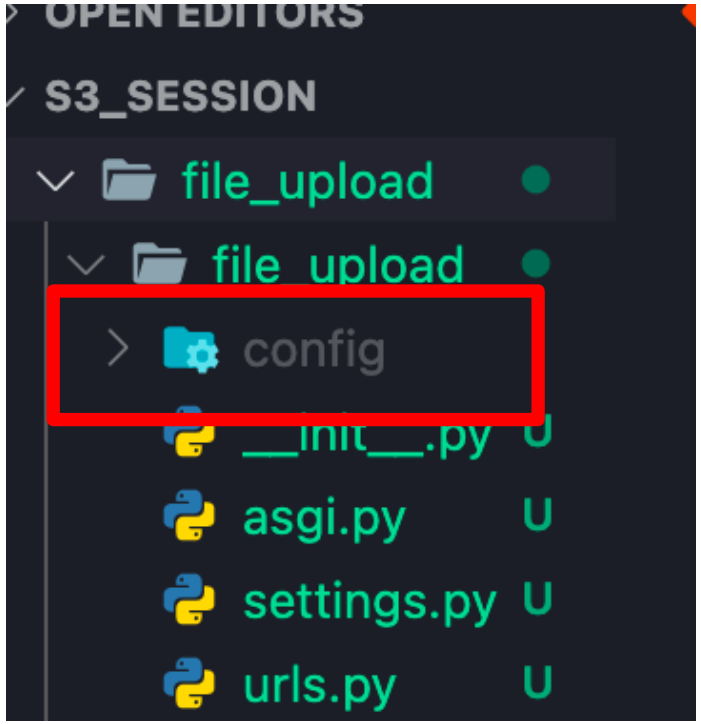
git init

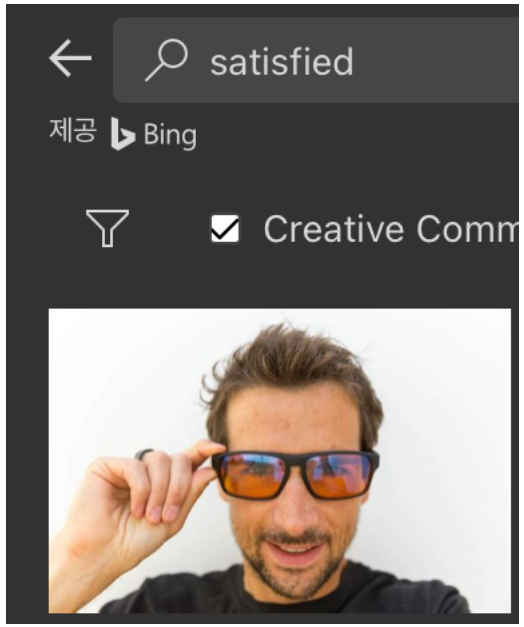
- 프로젝트 폴더(Pipfile, pipfile.lock이 있는 폴더)에서 git init

```
 isoyeong  ~/Desktop/s3_session  ls
Pipfile      Pipfile.lock file_upload
 isoyeong  ~/Desktop/s3_session  git init
힌트: Using 'master' as the name for the initial branch. This default branch
힌트: is subject to change. To configure the initial branch name to use in al
힌트: of your new repositories, which will suppress this warning, call:
힌트:
힌트:   git config --global init.defaultBranch <name>
힌트:
힌트: Names commonly chosen instead of 'master' are 'main', 'trunk' and
힌트: 'development'. The just-created branch can be renamed via this command:
힌트:
힌트:   git branch -m <name>
/Users/isoyeong/Desktop/s3_session/.git/ 안의 빈 깃 저장소를 다시 초기화했습니
```

gitignore이 잘 되었는지 확인

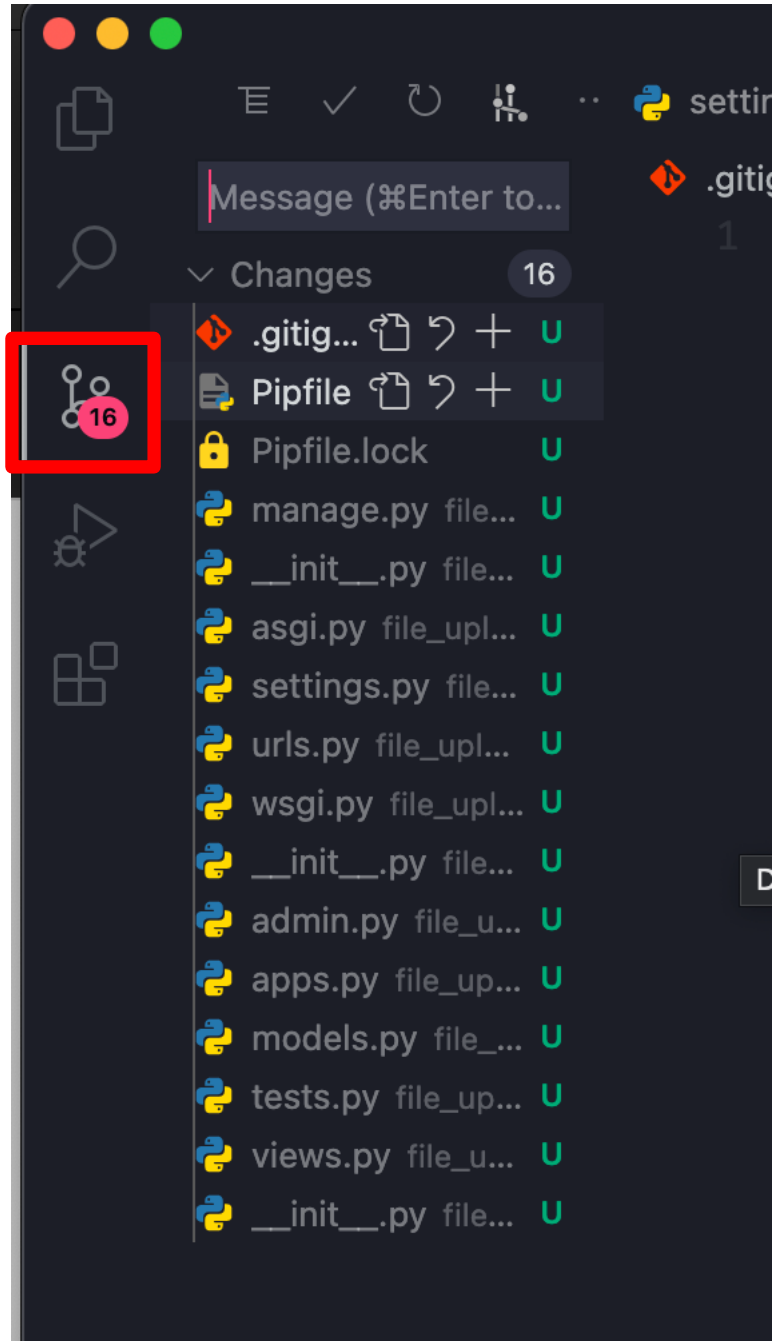
- config 폴더만 회색이다.





gitignore이 잘 되었는지 확인2

- index.json이 없는 것을 확인



중간점검

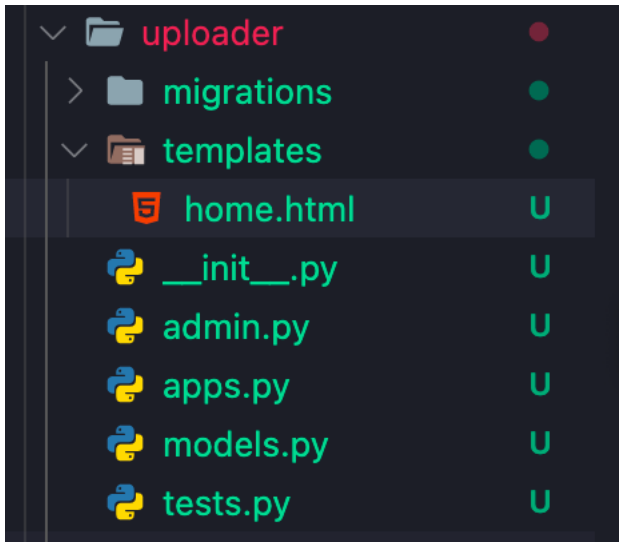
지금까지 한 것

- 장고 프로젝트 생성
- S3와 연결을 위한 boto3 다운로드
- AWS S3 버킷 생성
- 버킷 설정
- 장고 프로젝트에 s3 config 연결 및 보안 설정 완료

파일 업로드하기(model)

```python
# Create your models here.
class Post(models.Model):
    title = models.CharField(max_length=50)
    img = models.TextField()
```

파일 업로드하기(template)




```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scal
    <title>Document</title>
</head>
<body>
    <form method="POST" enctype="multipart/form-data">
    {% csrf_token %}
    <input type="text" name="title">
    <input type="file" name="img">
    <button type="submit">업로드</button>
    </form>

    {% for post in posts %}
    <div>
        <h3>{{ post.title }}</h3>
        <img src="{{ post.img }}" alt="{{ post.title }}">
    </div>
    {% endfor %}
</body>
</html>
```

파일 업로드하기(views)

views.py

file_upload > uploader > views.py > ...

```python
from django.shortcuts import render, redirect
from .models import Post
from file_upload.settings import AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY,
AWS_STORAGE_BUCKET_NAME, AWS_S3_REGION_NAME
import boto3
from boto3.session import Session
from datetime import datetime
```

파일 업로드하기(views) (1/3)

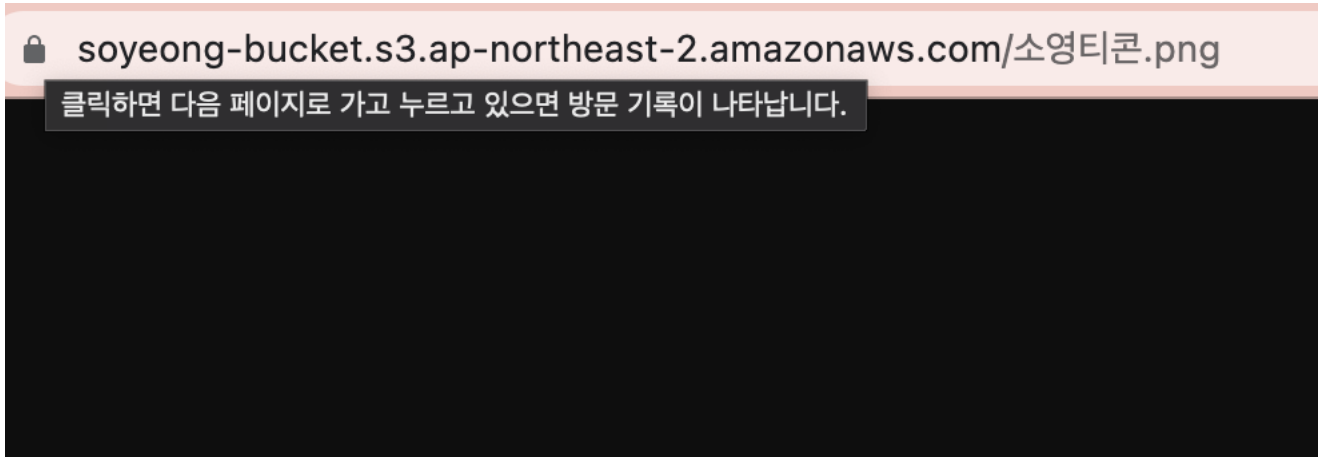
```python
# Create your views here.
def home(request):
    if request.method == 'POST':
        file_to_upload = request.FILES.get('img')
        session = Session(
            aws_access_key_id=AWS_ACCESS_KEY_ID,
            aws_secret_access_key=AWS_SECRET_ACCESS_KEY,
            region_name=AWS_S3_REGION_NAME
        )
        s3= session.resource('s3')
        now = datetime.now().strftime("%Y%H%M%S")

        img_object = s3.Bucket(AWS_STORAGE_BUCKET_NAME).put_object(
            Key = now + file_to_upload.name,
            Body = file_to_upload
        )
        s3_url = 'https://soyeong-bucket.s3.ap-northeast-2.amazonaws.com/'
```

```
2021-07-21 09:35:37.091766
```

```
("%Y%m%d%H%M%S")
```

파일 업로드하기(views) (2/3)

s3 url?



127.0.0.1:8000/detail/1 ⇒ pk=1인 글 불러오기

soyeong-bucket.s3/소영티콘.png ⇒ key=소영티콘.png인 사진 불러오기

우리가 장고실습때 detail 페이지에서 pk로 포스트를 불러왔듯,
s3에서는 개체의 고유한 식별자인 key를 통해 사진을 불러온다.

파일 업로드하기(views) (3/3)

```
26          post = Post.objects.create(
27              title = request.POST['title'],
28              img = s3_url + now + file_to_upload.name,
29          )
30          return redirect('home')
31
32      posts = Post.objects.all()
33      return render(request, 'home.html', {'posts': posts})
```

2021133919 스크린샷 2021-07-19 오후 12.28.06.png

2021133926 스크린샷 2021-07-19 오후 12.28.06.png

파일 업로드하기(urls)

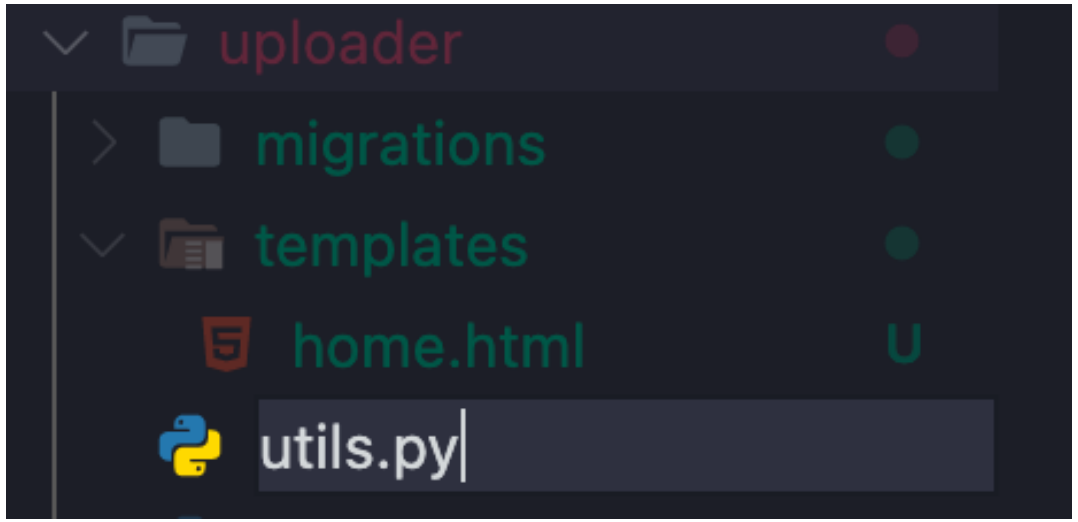
```python
from django.contrib import admin
from django.urls import path
from uploader import views

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.home, name='home'),
]
```

잘 올라가나..?

파일 업로드하기(utils)

- views 코드 너무 길어..
- 앱(uploader)에 새로운 파일을 만들자
- utils: 간단히 말해서, 기능성 함수

- 로직이 동작하여 데이터를 가져오고 결과를 템플릿에 전달하는 역할을 하는 views를 깔끔하게 유지하기 위해 자잘한 기능을 가진 함수는 유틸로 빼주기



파일 업로드하기(views)

views.py

file_upload > uploader > views.py > ...

```python
from django.shortcuts import render, redirect
from .models import Post
from file_upload.settings import AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY,
AWS_STORAGE_BUCKET_NAME, AWS_S3_REGION_NAME
import boto3
from boto3.session import Session
from datetime import datetime
```

파일 업로드하기(utils)

utils.py

file_upload > uploader > utils.py > upload_and_save

```python
from .models import Post
from file_upload.settings import AWS_ACCESS_KEY_ID, AWS_SECRET_ACCESS_KEY, AWS_S
import boto3
from boto3.session import Session
from datetime import datetime
```

파일 업로드하기(utils)

```python
def upload_and_save(title, file_to_upload):
    # client 만들기
    session = Session(
        aws_access_key_id=AWS_ACCESS_KEY_ID,
        aws_secret_access_key=AWS_SECRET_ACCESS_KEY,
        region_name=AWS_S3_REGION_NAME
    )
    s3= session.resource('s3')
    now = datetime.now().strftime("%Y%m%d%H%M%S")

    # s3에 object 업로드
    img_object = s3.Bucket(AWS_STORAGE_BUCKET_NAME).put_object(
        Key = now + file_to_upload.name,
        Body = file_to_upload
    )

    # Post 테이블에 저장
    s3_url = 'https://soyeong-bucket.s3.ap-northeast-2.amazonaws.com/'
    post = Post.objects.create(
        title = title,
        img = s3_url + now + file_to_upload.name,
    )
```

파일 업로드하기(views)

utils.py views.py

file_upload > uploader > views.py > ...

```python
from django.shortcuts import render, redirect
from .models import Post
from .utils import upload_and_save


# Create your views here.
def home(request):
    if request.method == 'POST':
        file_to_upload = request.FILES.get('img')
        title = request.POST['title']
        upload_and_save(title, file_to_upload)

        return redirect('home')

    posts = Post.objects.all()
    return render(request, 'home.html', {'posts': posts})

```

과제

- user별로 다른 폴더에 이미지 업로드 하기 연습
hint: pk, ‘/’

- (필요하다면) 개인 프로젝트에 s3 연결하기