



애저 서버리스에서 파이썬/장고 웹서비스 쉽게 배포/운영하기 장고 애플리케이션 생성 및 애저 펌션 연동

여러분의 파이썬/장고/리액트 페이스메이커가 되겠습니다.

세션 목차

- EP01. 애저펍션과 파이썬 구동 환경 소개
- EP02. 애저펍션 개발환경 구축 및 기본 프로젝트 생성
- EP03. 장고 애플리케이션 생성 및 애저펍션 연동
- EP04. Azure SQL Database와 장고 연동
- EP05. Azure Blob Storages와 장고 연동
- EP06. 애저펍션 인프라로 배포하기
- EP07. 리액트 웹 애플리케이션 배포와 애저 장고 API 연동
- EP08. 마무리

Agenda

- 장고 프로젝트 생성
- 애저펄션에서 받는 요청을 장고 WSGI로 전달하기
- 보다 편리한 환경변수 처리를 위한 django-environ 라이브러리
- 사진 포스팅 REST API 개발 및 API 테스트

장고 프로젝트 생성

가상환경에 장고 라이브러리 설치

```
python -m pip install "django ≈ 3.0.0"
```

Azure Functions 프로젝트 루트 경로에서 새로운 장고 프로젝트 생성

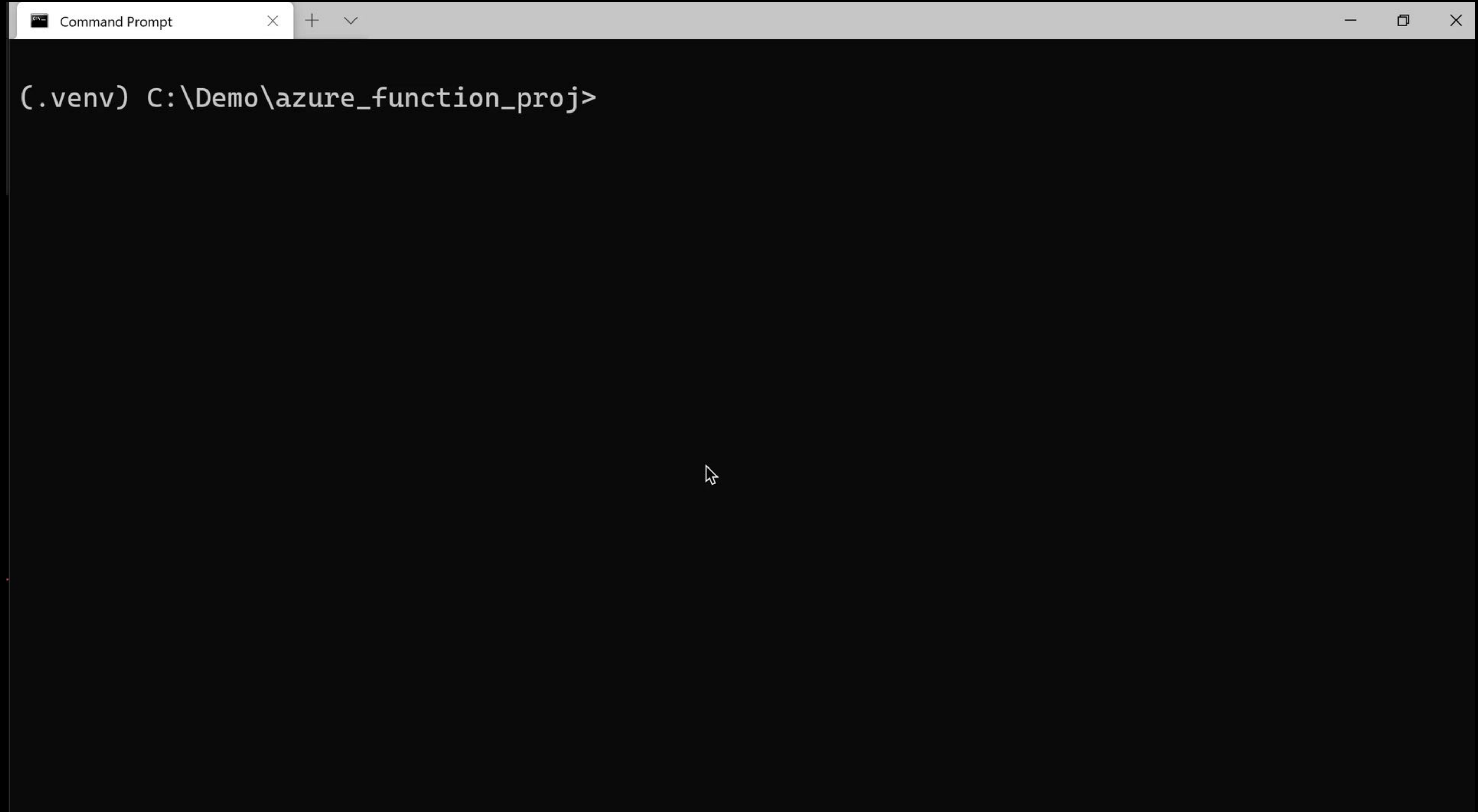
```
python -m django startproject dj_proj
```

프로젝트 디렉토리 이동 및 장고 개발서버 구동 (Azure Functions에서는 사용 X) ➡ 구동 확인 후에, 서버 중단해주세요.

```
cd dj_proj
```

```
python manage.py runserver
```

장고 프로젝트 생성 (Demo)



```
(> .venv) C:\Demo\azure_function_proj>
```

아저평션에서 받는 요청을 장고 WSGI로 전달하기

/host.json

```
{
  "version": "2.0",
  "extensions": {
    "http": {
      "routePrefix": ""
    }
  },
  "logging": {
    "applicationInsights": {
      "samplingSettings": {
        "isEnabled": true,
        "excludedTypes": "Request"
      }
    }
  },
  "extensionBundle": {
    "id": "Microsoft.Azure.Functions.ExtensionBundle",
    "version": "[1.*, 2.0.0)"
  }
}
```

디폴트 url prefix인 /api/ 를 제거합니다.

/dj_http_trigger/function.json

```
{
  "scriptFile": "__init__.py",
  "bindings": [
    {
      "authLevel": "anonymous",
      "type": "httpTrigger",
      "direction": "in",
      "name": "req",
      "methods": [
        "get",
        "post"
      ],
      "route": "{*route}"
    },
    {
      "type": "http",
      "direction": "out",
      "name": "$return"
    }
  ]
}
```

이하 URL에 대한 라우팅을 모두 위임받습니다.

/dj_http_trigger/__init__.py

```
import sys
sys.path.insert(0, 'dj_proj')

import azure.functions as func
from dj_proj.wsgi import application

wsgi = func.WsgiMiddleware(app=application)

def main(req: func.HttpRequest, context: func.Context) → func.HttpResponse:
    return wsgi.main(req, context)
```

/requirements.txt

azure-functions

django \simeq 3.0.0

django-environ

/admin/ 주소는 애저펍션에서 선점

- `/admin/functions/...` 경로를 통해 **Http Trigger가 아닌 함수**를 실행할 수 있습니다.
- 장고의 admin을 사용하려면, 다른 주소로 변경해줘야만 합니다.

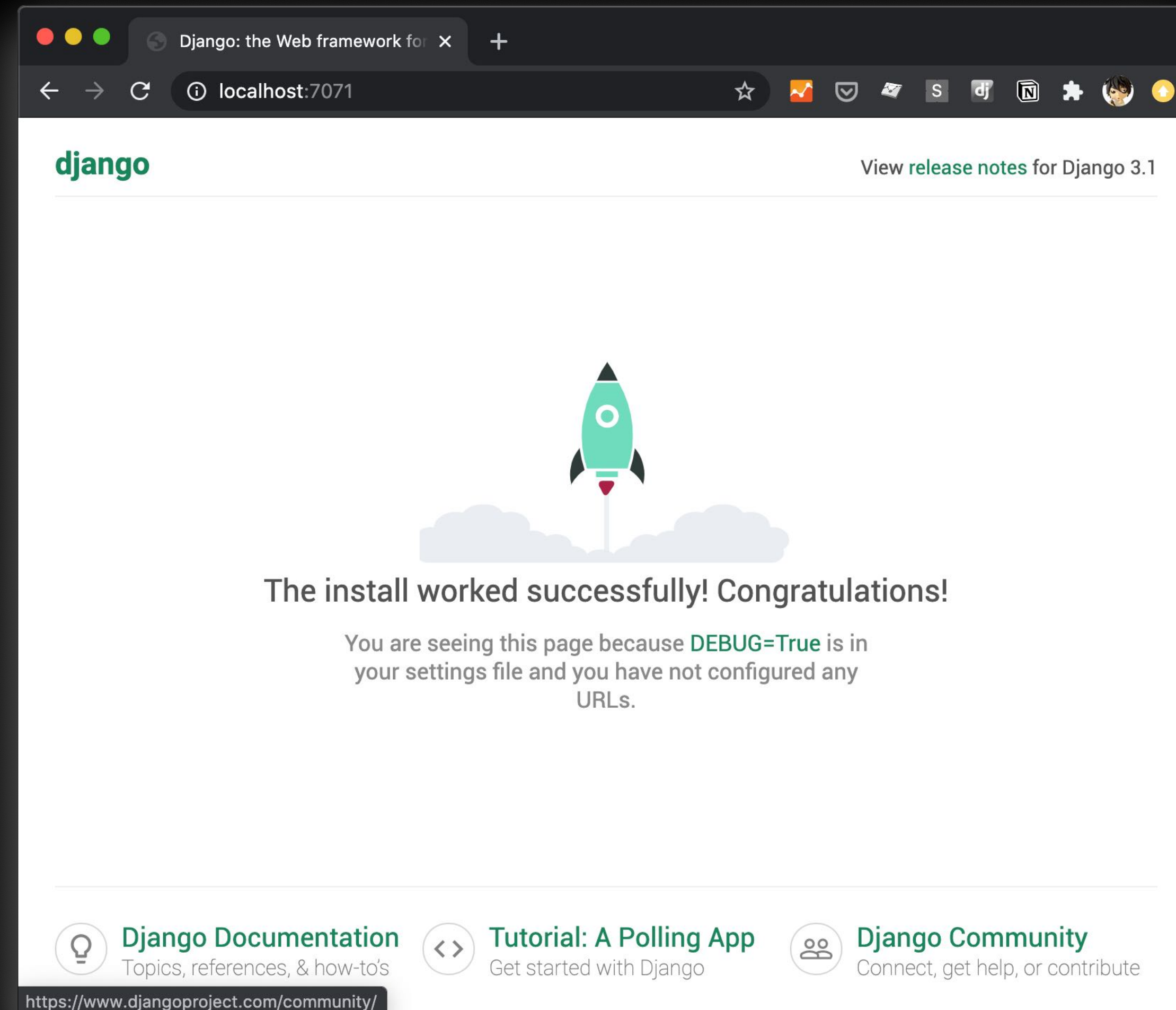
`https://myfunctions demos.azurewebsites.net/admin/functions/QueueTrigger`

HOST NAME	FOLDER PATH	FUNCTION NAME
<code>https://myfunctions demos.azurewebsites.net</code>	<code>/admin/functions/</code>	<code>QueueTrigger</code>

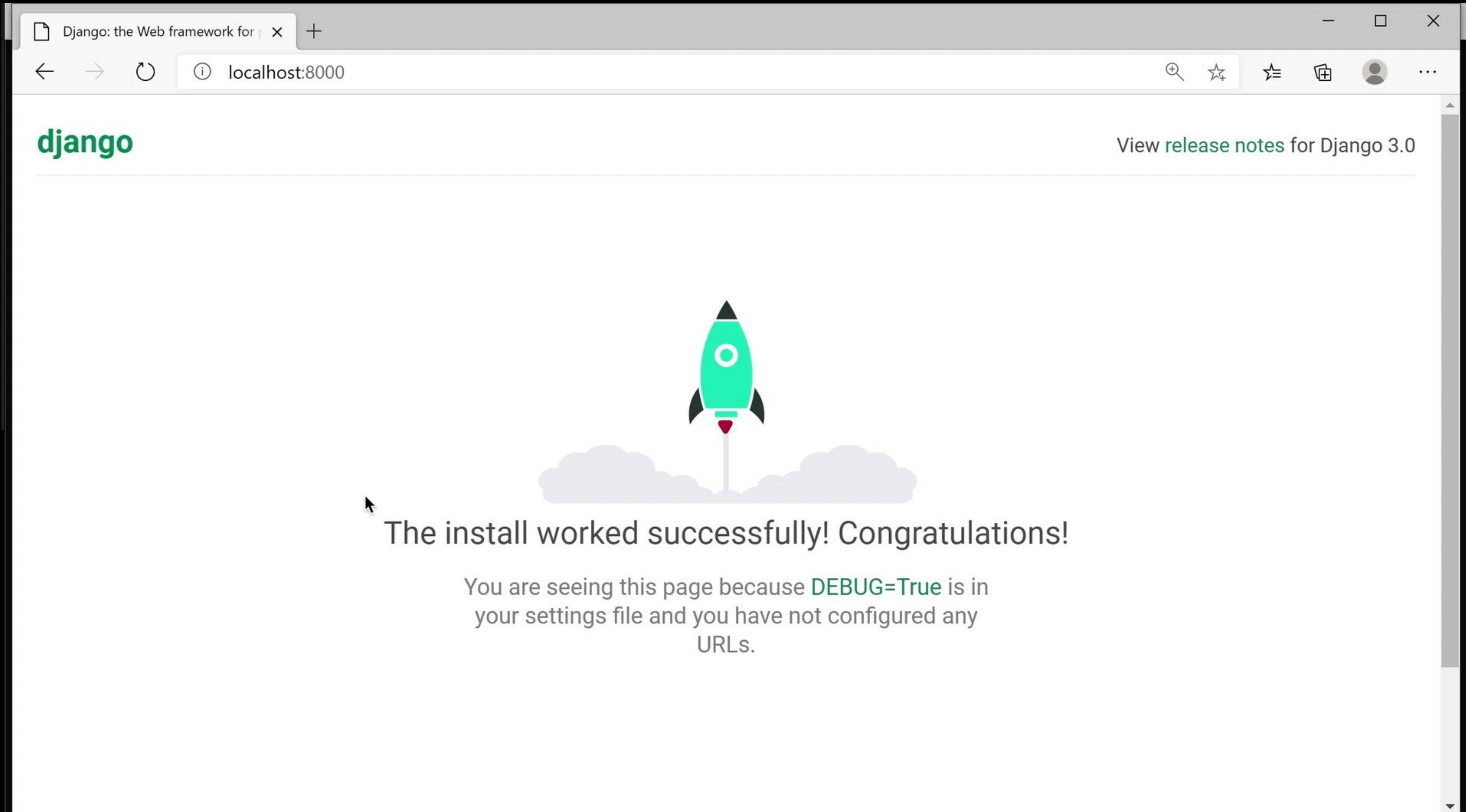
(공식문서) HTTP 이외 트리거 함수를 수동으로 실행

Azure Functions 개발서버 통해 장고 프로젝트 접근

`func start` # Azure Functions 개발서버 구동



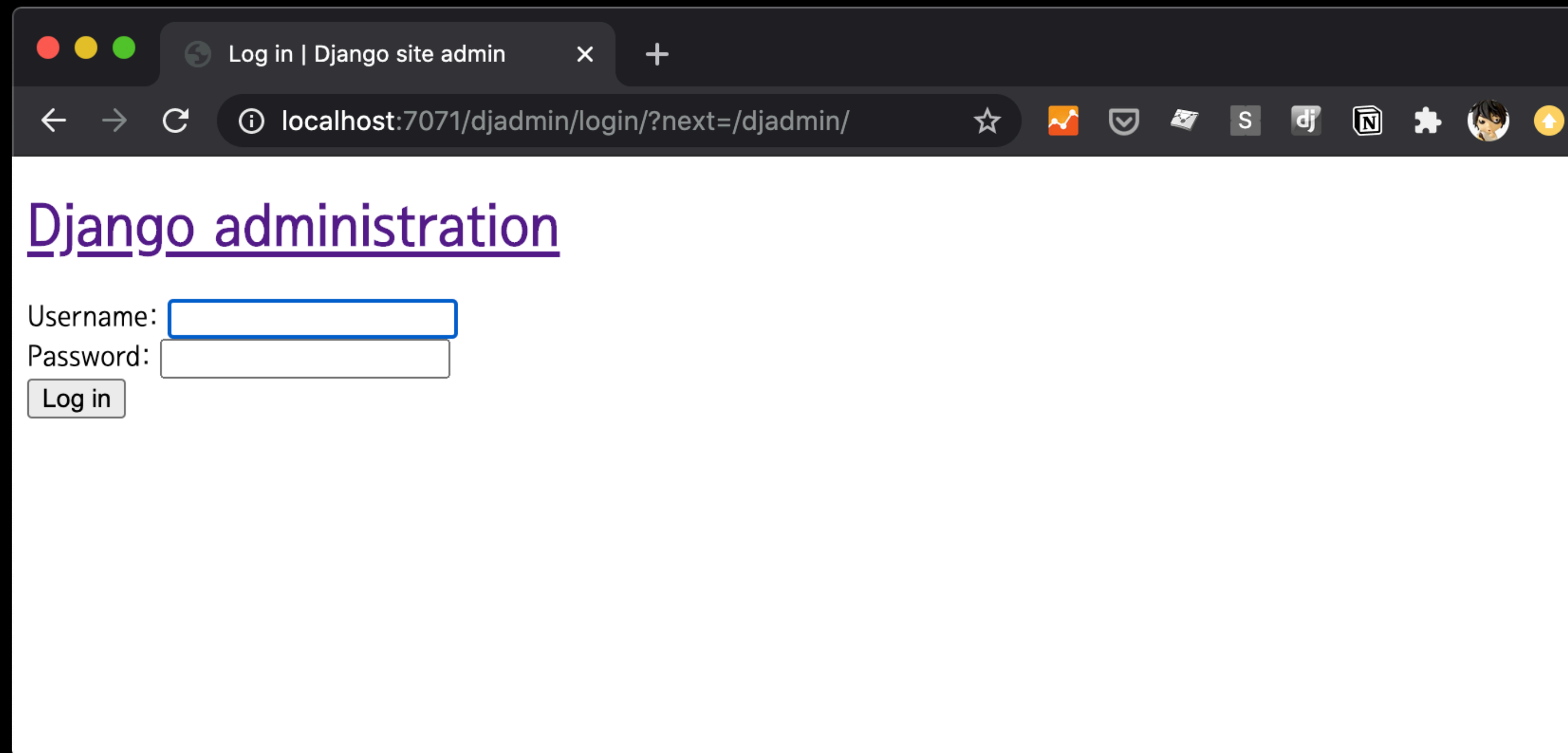
아저 평션 프로젝트에 장고 프로젝트 맞추기 (Demo)



참고) 개발모드에서의 static 서빙

장고개발서버/디버그모드에서만 static 파일 서빙을 한정적으로 지원

- 방법1) Azure Functions은 배포 시에만 활용하고, 장고 개발에서는 장고 개발서버를 활용
- 방법2) whitenoise 라이브러리를 통해, 장고가 직접 static 서빙



보다 편리한 환경변수 처리를 위한 django-environ 라이브러리

Azure Functions 로컬서버 환경변수

```
{ } local.settings.json > ...
```

```
1 {
2   "IsEncrypted": false,
3   "Values": {
4     "FUNCTIONS_WORKER_RUNTIME": "python",
5     "AzureWebJobsStorage": "",
6     "DATABASE_URL": "mssql://myadmin:!![REDACTED]r@se[REDACTED].database.
7     windows.net:1433/s[REDACTED]?driver=ODBC Driver 17 for SQL Server&
8     timeout=5&conn_max_age=30",
9     "AZURE_ACCOUNT_NAME": "s[REDACTED]",
10    "AZURE_ACCOUNT_KEY": "i[REDACTED]Q=="
  }
```

Azure Functions 프로세스 내에서
참조하는 환경변수 목록

장고 개발서버에서 local.settings.json 로딩

```
import json
from environ import Env
from pathlib import Path

env = Env()

BASE_DIR = Path(__file__).resolve().parent.parent

# local.settings.json 내 Values 값들을 읽어들이고, 현재 프로세스 환경변수 내역을 업데이트합니다.
try:
    with BASE_DIR.parent.joinpath("local.settings.json").open() as f:
        local_settings = json.load(f)
        for key, value in local_settings['Values'].items():
            env.ENVIRON[key] = value
except (IOError, KeyError):
    pass
```

장고 개발서버에서 local.settings.json 로딩 (Demo)

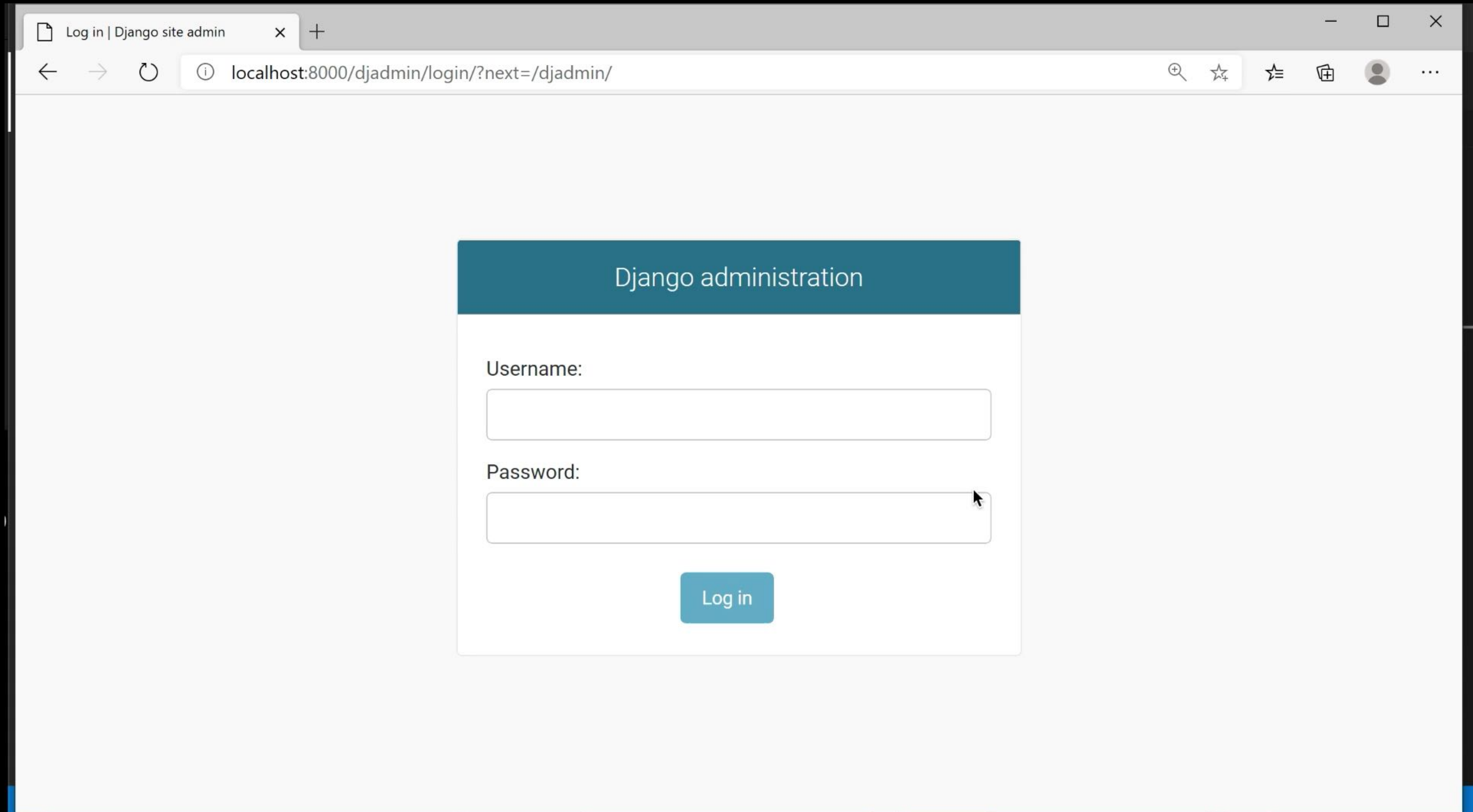


사진 포스팅 REST API 개발 및 API 테스트

추가 라이브러리

- 추가 라이브러리
 - djangorestframework : REST API 도움 라이브러리
 - pillow : 이미징 라이브러리 (ImageField에서 내부적으로 사용)
- 진행 순서
 - rest_framework 앱 등록
 - blog 앱 생성 및 등록
 - DRF를 활용하여 포스팅 REST API 구현

주요 장고 코드

```
# blog/models.py
```

```
from django.db import models
```

```
class Post(models.Model):  
    message = models.TextField()  
    photo = models.ImageField(blank=True)  
    created_at = models.DateTimeField(auto_now_add=True)  
    updated_at = models.DateTimeField(auto_now=True)
```

```
# blog/serializers.py
```

```
from rest_framework import serializers  
from .models import Post
```

```
class PostSerializer(serializers.ModelSerializer):  
    class Meta:  
        model = Post  
        fields = '__all__'
```

```
# blog/views.py
```

```
from django.shortcuts import render  
from rest_framework import viewsets
```

```
from .models import Post  
from .serializers import PostSerializer
```

```
class PostViewSet(viewsets.ModelViewSet):  
    queryset = Post.objects.all()  
    serializer_class = PostSerializer
```

```
# blog/urls.py
```

```
from rest_framework.routers import DefaultRouter  
from . import views
```

```
router = DefaultRouter()  
router.register('posts', views.PostViewSet)
```

```
urlpatterns = router.urls
```

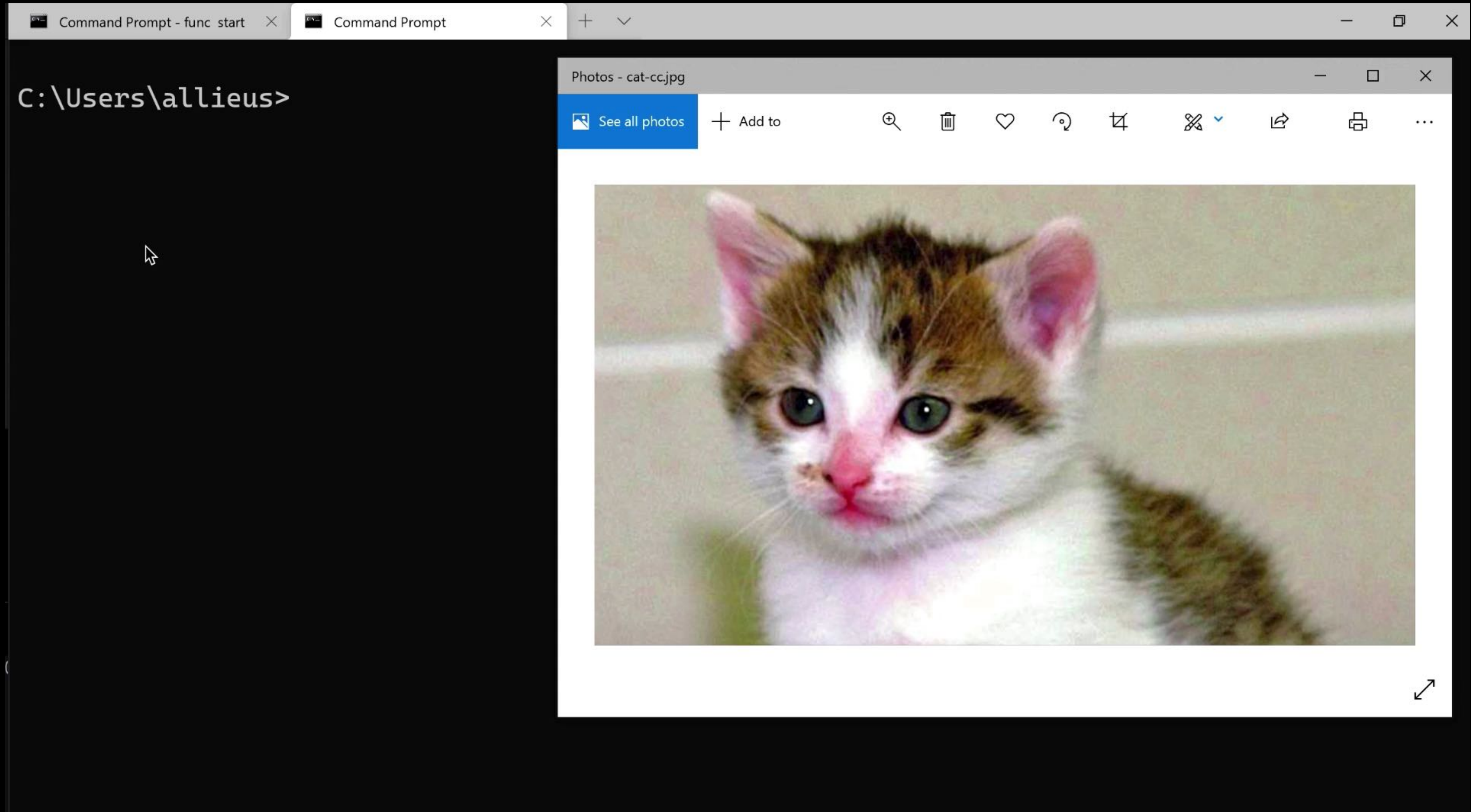
포스팅 CRUD API 개발 (Demo)

```
Command Prompt - python ma X + v
(.venv) C:\Demo\azure_function_proj\dj_proj> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 17 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
September 22, 2020 - 12:26:46
Django version 3.0.10, using settings 'dj_proj.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```


HTTPIe를 활용한 API 테스트 (Demo)



다음 에피소드에서는 ...

EP04. Azure SQL Database와 장고 연동

- Azure SQL 소개
- Azure SQL Database 생성 및 방화벽 설정
- ODBC 17 Driver
- Azure SQL Database 와 장고 프로젝트 연동

Life is short.
You need Python and Django.

I will be your pacemaker.

