



HEROKU

헤로쿠

멋쟁이 사자처럼 at 성공회대학교 7기
운영진 김명석

**수현이는
지금까지 배우고 만들었던
내용들을 보여주고
자랑하고 싶어졌다...**

**근데 그러려면
노트북 키고
VSCODE 키고
Runserver를 해야만
자랑 가능..**



(헬로비너스 나라임 노잼이면 ㅈㅈ)

**이게 나라냐..
이게 진짜 자랑이냐..**

자 그럼 이제 !

**진짜 자랑할 수 있게
“헤로쿠”라는 걸 한 번 써보자!**

**앞으로 배울 과정을
'배포'
라고 합니다.**

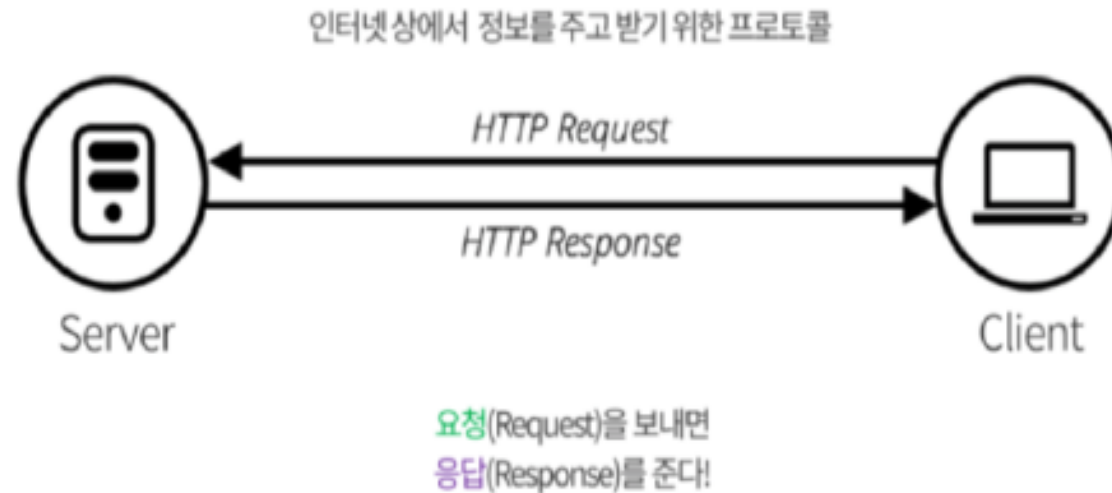


**원래 배포를 하려면
'서버'가 필요합니다.**

**<= 당신들이 생각하는 서버의 이미지
근데 진짜 이게 서버 맞음 ㅎ**

서버(Server)란?

클라이언트에게 서비스를 제공하는 측



서버(Server)란?

**서버는 개인의 컴퓨터가 될 수도 있고,
인터넷 자체로 제공하는 서버가 될 수도 있다.**

**우리가 사용하는
Python manage.py runserver
이 명령어도 지금 사용하는 이 컴퓨터를 서버
로 사용한다는 명령어**

**그래서 우리가 웹 호스팅을 직접 해주는 서버를
이용하려면 원래는 돈이 든다!**

**서버를 제공해주는 호스팅 업체의 서버를
사용하니까!**

**(회사의 서버 용량을
24시간 우리가 사용하는 만큼을
늘 차지해야하니까)**

서버를 제공해주는 호스팅 업체 종류

cafe24TM



Amazon
EC2



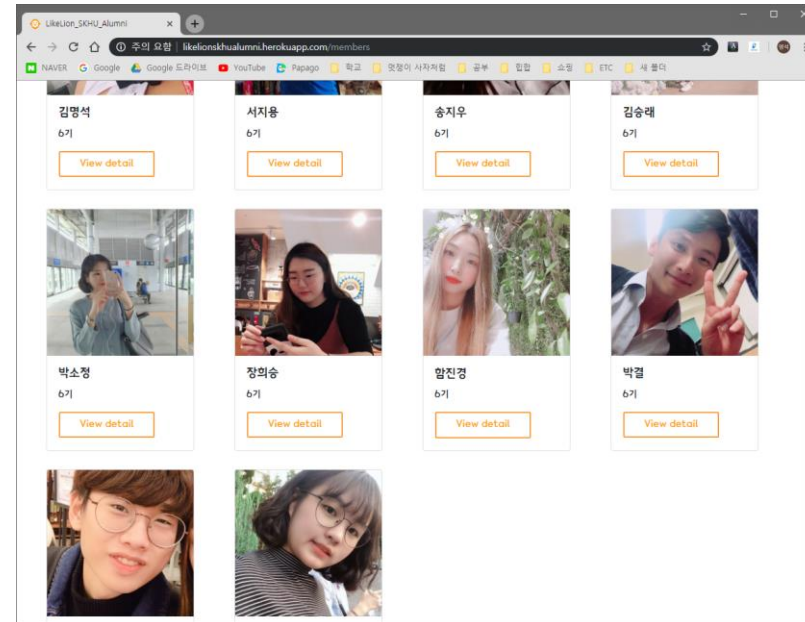
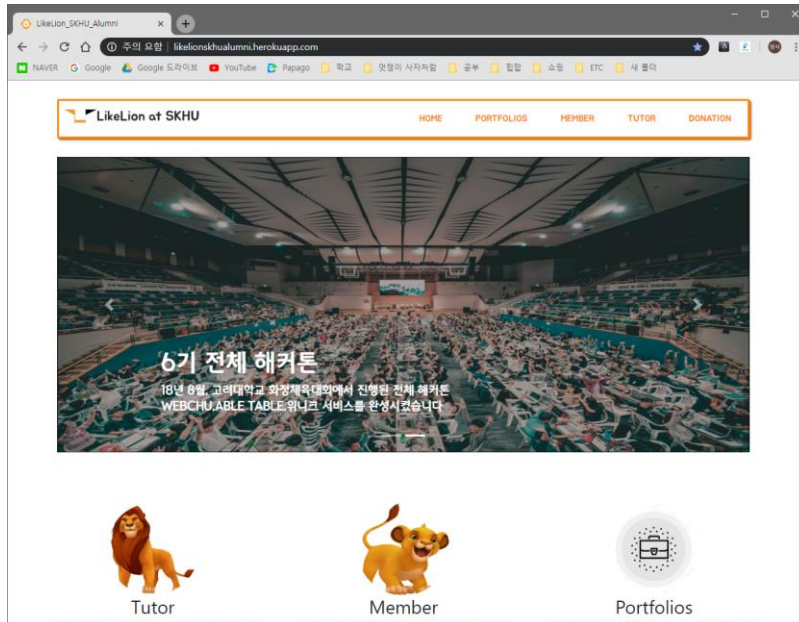
우리가 헤로쿠를 선택한 이유?



**공짜니까!
(5개까지 무료)**

실제 헤로쿠를 사용하여 배포한 사이트

<http://likelionskhualumni.herokuapp.com/>



(홍보 맞음 〇〇)

자 그럼 이제 진짜로 한번

“ 배 — 포 ”

라는 걸 해봅시다.

역시나 먼저 *git clone*을 떠봅시다.

Repositories 6

People 27

Teams 0

Projects 0

Settings

Pinned repositories

Customize pinned repositories

Week3_Python_Assignment

SKHU 멧쟁이 사자처럼 7기 3주차 파이썬 과제 모음입니다.

Python

Find a repository...

Type: All

Language: All

New

Heroku_Practice

헤로쿠 실습 완료

Python Updated 23 minutes ago

heroku_layout

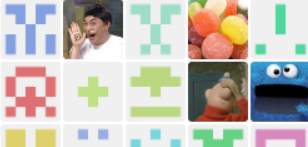
Python Updated an hour ago

Top languages

Python

People

27 >



Clone 다 뜨면

가상환경 실행

source myvenv/Scripts/activate

서버 한번 켜보기

Python manage.py runserver

지금 배포하려는 이 프로젝트의 설명 :

가상환경명 : myvenv

프로젝트 폴더 : crud

앱 폴더 : blog

Model : Blog

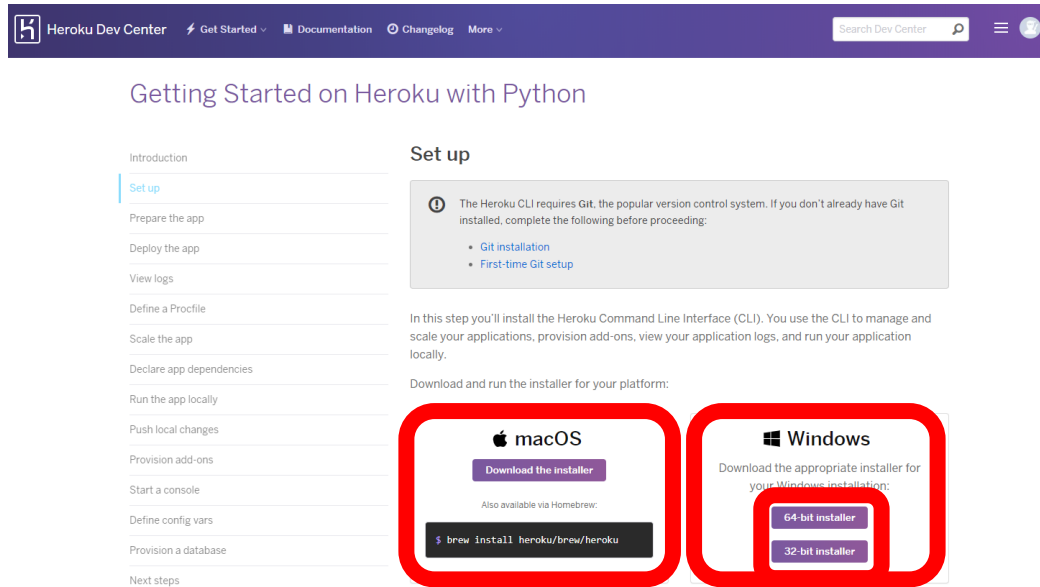
Admin : admin / admin123

부트스트랩 테마 사용 중,

CRUD 구현완료

마이그레이트까지 다 되어있는 상황

자 그럼 이제 진짜 본격적 시작을 위해 준비를 해봅시다.



**헤로쿠로 접속해서
헤로쿠 CLI를 설치
(Command Line Interface)**

<https://devcenter.heroku.com/articles/getting-started-with-python#set-up>

헤로쿠 회원가입하기!!

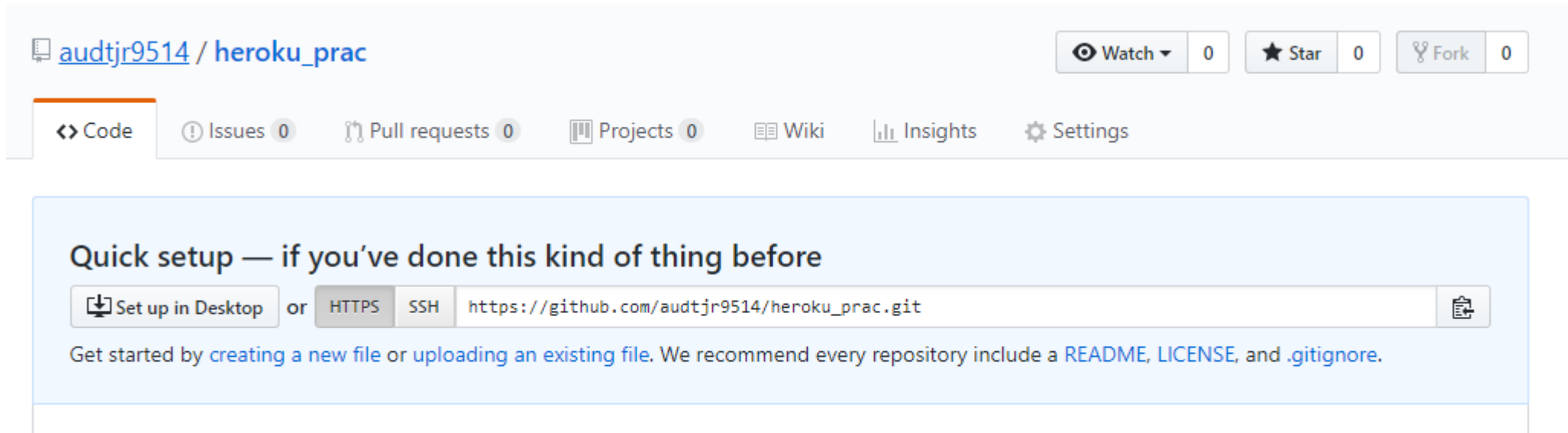
아이디는 본인 gmail로 가입하시고
(멋사계정 씹가능 - 추천함)

까먹지 않게 **제발 다른데 적어두시길**
제발이라고 두번 말합니다.
제.발.

**헤로쿠는 깃을 기반으로
서버를 돌리기 때문에
서버로 돌리기 위한
레포지터리가 필요합니다.**

각자 깃 레포지터리를 생성합니다.

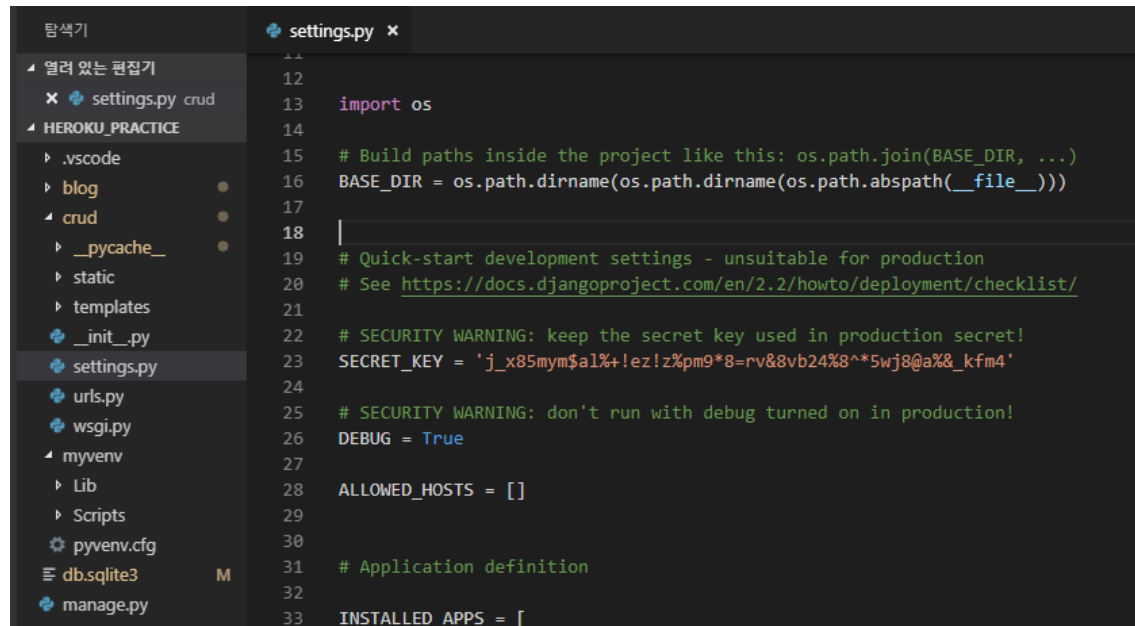
전 heroku_prac으로 생성했습니다.



**님들은 heroku_practice로 ㄱㄱ
나는 이미 있어서 저렇게 함 ㅌㅌ**

이제 실습 시작!

settings.py로 이동



```
12
13 import os
14
15 # Build paths inside the project like this: os.path.join(BASE_DIR, ...)
16 BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
17
18
19 # Quick-start development settings - unsuitable for production
20 # See https://docs.djangoproject.com/en/2.2/howto/deployment/checklist/
21
22 # SECURITY WARNING: keep the secret key used in production secret!
23 SECRET_KEY = 'j_x85mym$a1%+!ez!z%pm9*8=rv&8vb24%8^*5wj8@a%&_kfm4'
24
25 # SECURITY WARNING: don't run with debug turned on in production!
26 DEBUG = True
27
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
```

수정!

```

23 # SECURITY WARNING: keep the secret key used in production secret!
24 # SECRET_KEY = 'j_x85mym$aL%+!ez!z%pm9*8=rv&8vb24%8^*5wj8@a%&_kfm4'
25 SECRET_KEY = os.environ.get('DJANGO_SECRET_KEY', 'j_x85mym$aL%+!ez!z%pm9*8=rv&8vb24%8^*5wj8@a%&_kfm4')
26
27 # SECURITY WARNING: don't run with debug turned on in production!
28 # DEBUG = True
29 DEBUG = bool(os.environ.get('DJANGO_DEBUG', True))
30
31 ALLOWED_HOSTS = ['*']

```

기존의 자기 시크릿 키를 여기다 넣어주세요.

`SECRET_KEY = os.environ.get('DJANGO_SECRET_KEY', '시크릿키')`

`DEBUG = bool(os.environ.get('DJANGO_DEBUG', True))`

`ALLOWED_HOSTS = ['*']`

기존의 코드와 달라진 점 :

Secret key :

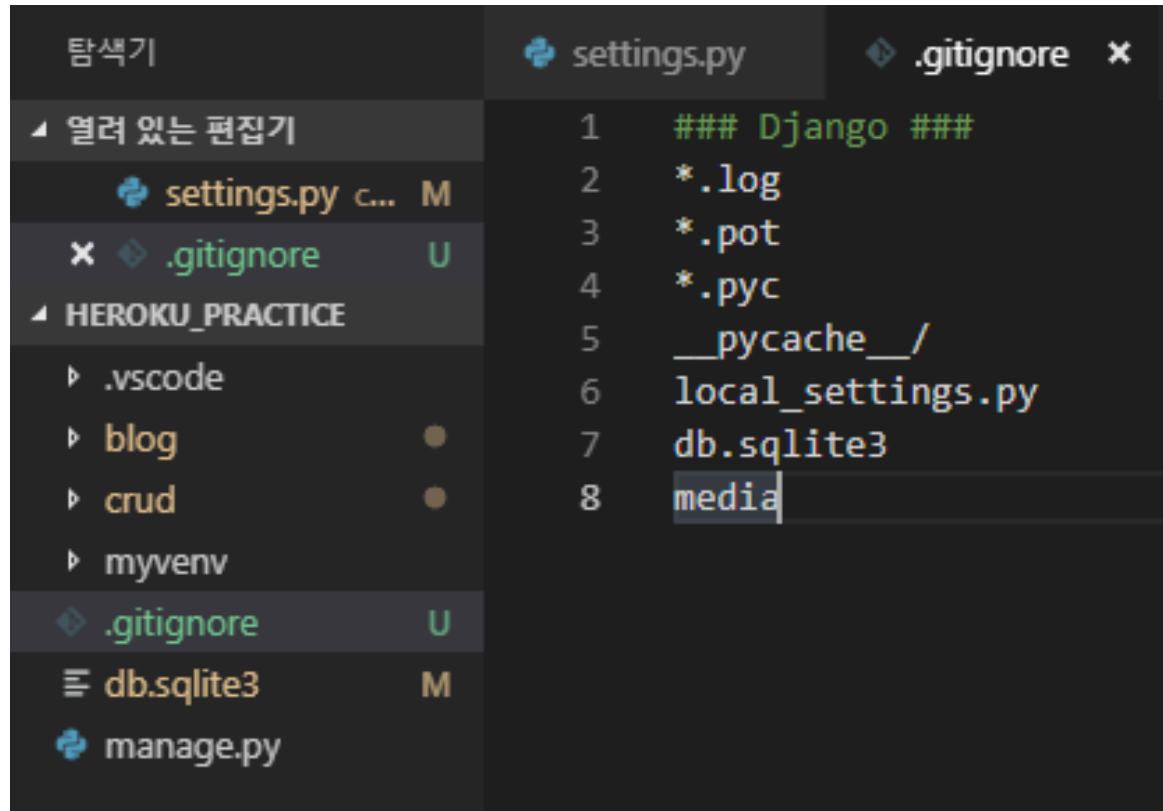
**CRSF 보안 등을 위해 사용되는 랜덤값
이걸 다른 사람이 알면 그냥 뚫린다.**

DEBUG :

**개발시 페이지가 없다던가, url이 잘못되었다는 등의 설명을 해주는 기능
- 이 기능이 켜진 상태로 배포되면 다른 개발고인물이 와서 수정을 할지도 몰라**

**그래서 우리는 이 부분들을 관리해주어야 한다.
앞으로 쓸 방법은 위의 값들을 환경 변수로써 사용하는 것**

.gitignore 생성하기



The screenshot shows the VS Code editor with the `.gitignore` file open. The file content is as follows:

```

1  ### Django ###
2  *.log
3  *.pot
4  *.pyc
5  __pycache__ /
6  local_settings.py
7  db.sqlite3
8  media
  
```

The Explorer sidebar on the left shows the project structure for `HEROKU_PRACTICE`, including files like `settings.py`, `.gitignore`, `db.sqlite3`, and `manage.py`.

.gitignore ?

**깃에 올릴 프로젝트 중
제외할 파일을
설정해두는 것!**
**모든 언어, 프레임워크, IDE
해당 사항이 있다.**

gitignore.io
**모든 gitignore를
모아놓은 사이트 - 개꿀**

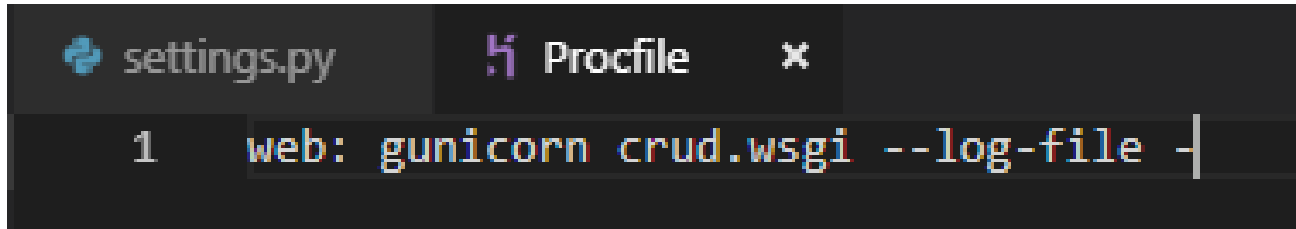
Procfile 생성하기

Procfile?

애플리케이션을 헤로쿠를 통해 배포해서 실행할때, 헤로쿠가 맨 처음 실행하는 명령어를 지정하는 파일

wsgi : 프레임워크 웹서버 (미들웨어)
Web Server Gateway Interface
웹서버와 파이썬을 사용한 웹 어플리케이션
개발환경 간의 인터페이스에 관한 규칙

Gunicorn : 유닉스에서 사용하기 위한 wsgi



```

settings.py  Procfile  x
1 web: gunicorn crud.wsgi --log-file -
  
```

Manage.py가 있는 폴더에 Procfile 생성 (확장자 X)

web: gunicorn 프로젝트명.wsgi --log-file -

pip install gunicorn

gunicorn 설치하기

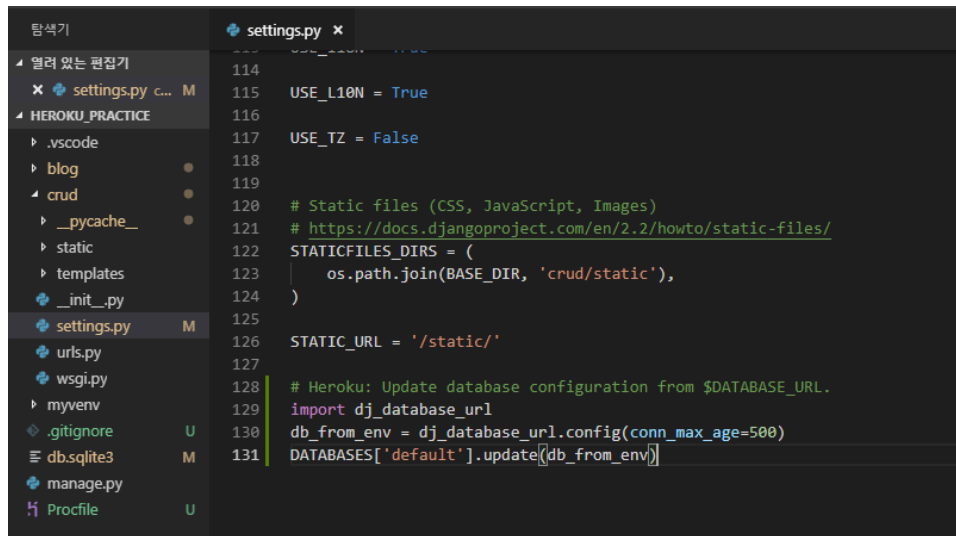
DataBase 설정하기

**헤로쿠에서는 원래 내장 DB를 사용할수 없다.
헤로쿠가 PostgreSQL을 기본옵션으로 설정하기 때문
그래서 이를 위해 새로운 설정이 필요**

```
pip install dj-database-url  
pip install psycopg2-binary
```

***dj-database-url : url에 지정된 장고 데이터베이스의 데이터를 딕셔너리 형식으로 반환
psycopg2-binary : 파이썬 프로그래밍용 SQL 데이터베이스 어댑터***

DataBase 설정하기



The screenshot shows a VS Code editor with the `settings.py` file open. The left sidebar shows the project structure with files like `settings.py`, `urls.py`, `wsgi.py`, `myenv`, `.gitignore`, `db.sqlite3`, `manage.py`, and `Procfile`. The main editor area shows the following code:

```

114 USE_L10N = True
115
116
117 USE_TZ = False
118
119
120 # Static files (CSS, JavaScript, Images)
121 # https://docs.djangoproject.com/en/2.2/howto/static-files/
122 STATICFILES_DIRS = (
123     os.path.join(BASE_DIR, 'crud/static'),
124 )
125
126 STATIC_URL = '/static/'
127
128 # Heroku: Update database configuration from $DATABASE_URL.
129 import dj_database_url
130 db_from_env = dj_database_url.config(conn_max_age=500)
131 DATABASES['default'].update(db_from_env)

```

settings.py 수정

장고데이터베이스의 설정 중 `conn_max_age`를 500으로 하여 기본 설정을 업데이트한다.

`conn_max_age` :
초 단위의 데이터 베이스 연결 수명

```

128 # Heroku: Update database configuration from $DATABASE_URL.
129 import dj_database_url
130 db_from_env = dj_database_url.config(conn_max_age=500)
131 DATABASES['default'].update(db_from_env)

```

Static File 설정하기

pip install whitenoise

Settings.py MIDDLEWARE 부분 수정하기

```
MIDDLEWARE = [  
    'whitenoise.middleware.WhiteNoiseMiddleware',  
    'django.middleware.security.SecurityMiddleware',  
    'django.contrib.sessions.middleware.SessionMiddleware',  
    'django.middleware.common.CommonMiddleware',  
    'django.middleware.csrf.CsrfViewMiddleware',  
    'django.contrib.auth.middleware.AuthenticationMiddleware',  
    'django.contrib.messages.middleware.MessageMiddleware',  
    'django.middleware.clickjacking.XFrameOptionsMiddleware',  
]
```

whitenoise?

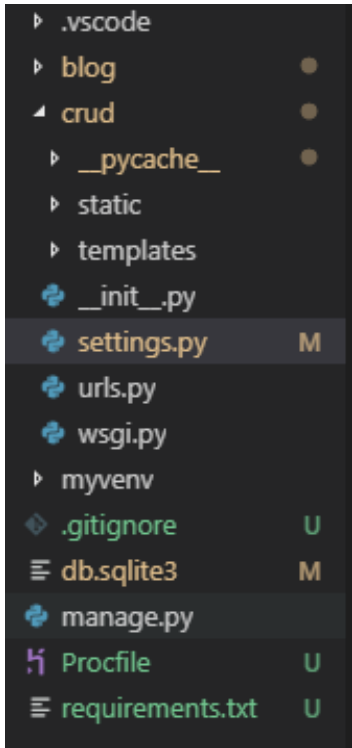
원래 개발과 배포는 다르다
개발시에는 개발자가 정적파일을
마음껏 건드릴수 있지만
배포시에는 안된다(디버그 기능 꺼짐)

그래서 헤로쿠에서 정적파일을
볼 수 있게 해주는 역할을 하는게
whitenoise

'whitenoise.middleware.WhiteNoiseMiddleware', 작성 ㄱ

파이썬 라이브러리 설치

pip freeze > requirements.txt



```

settings.py  requirements.txt x
1  autopep8==1.4.3
2  dj-database-url==0.5.0
3  Django==2.1.5
4  gunicorn==19.9.0
5  psycopg2-binary==2.7.7
6  pycodestyle==2.5.0
7  pytz==2018.9
8  whitenoise==4.1.2
9

```

**지금까지 pip로 우리는
많은 라이브러리를 깔았다.**

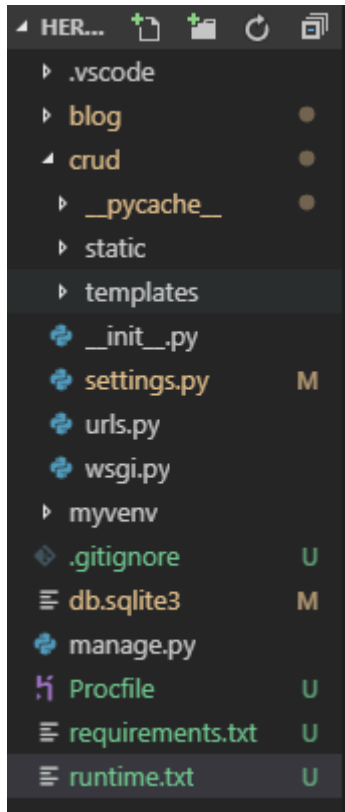
**지금까지 설치했던 라이브러리들을
서버에도 설치해주어야 한다.**

**위 명령어를 통해 관련 목록을 리스트업
해주는 것!**

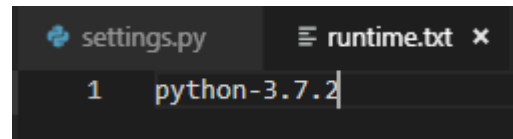
**명령어 실행시 프로젝트 폴더에 생성이
requirements.txt가 실행되어야 한다.**

헤로쿠에 파이썬 버전 알려주기

python --version
***runtime.txt* 생성하기**

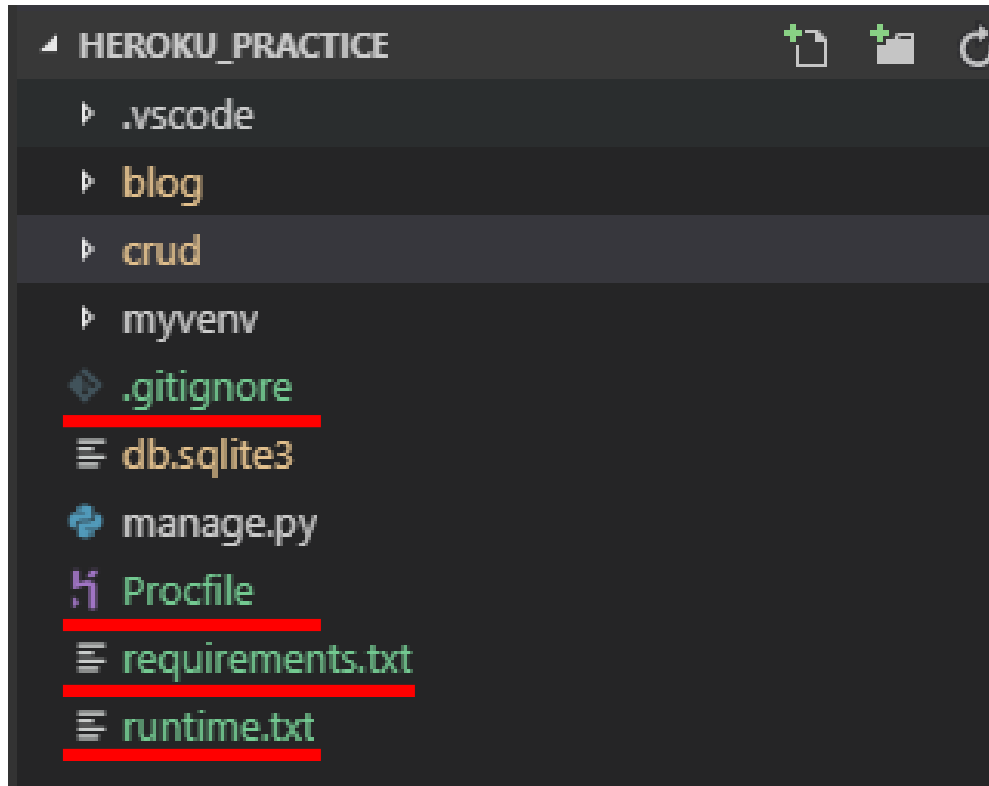


```
myoun@myoun MINGW64 ~/Desktop/Heroku_Practice (master)
$ python --version
Python 3.7.2
(myvenv)
```



A screenshot of a code editor window showing the content of the 'runtime.txt' file. The file is named 'runtime.txt' and contains the text 'python-3.7.2' on the first line.

확인 한번 조지자



**지금까지 생성된 파일들은
다 `manage.py`가
있는 폴더에 있어야한다!**

**.gitignore
Procfile
Requirements.txt
runtime.txt**

**이제 진짜
헤로쿠 렛-츠 기---릿!**

하기 전에 ㅎ

**깃에 올립니다.
그전에 폴더에 있던 .git 폴더 삭제 ㄱ**

manage.py가 있는 폴더에서
git init (.git을 삭제했으니까 다시 만들어야겠죠)
git remote add origin 레포지터리 주소(좀전에만든거)
git add .
git commit -m “커밋메세지”
git push origin master

heroku login

```
myoun@myoun MINGW64 ~/Desktop/Heroku_Practice (master)
$ heroku login
» Warning: heroku update available from 7.22.4 to 7.24.1.
heroku: Press any key to open up the browser to login or q to exit:
Opening browser to https://cli-auth.heroku.com/auth/browser/d51cf28c-2902-46d2-9383-664ae05358cb...
Logging in... done
Logged in as audtjr9514@gmail.com
(myvenv)
```

**중간에 뭐치라하면 걔 키보드 아무거나 ㄱ
q 말고 무난히 z 누르시다**

그러면 헤로쿠 사이트가 뜰텐데 로그인 ㄱ

heroku create

```
$ heroku create
» Warning: heroku update available from 7.22.4 to 7.24.1.
Creating app... done, ● shrouded-thicket-50466
https://shrouded-thicket-50466.herokuapp.com/ | https://git.heroku.com/shrouded-thicket-50466.git
(myvenv)
```

헤로쿠를 만듭시다.

오... 뭔가 생성이 된다 이거야

git push heroku master

**현재 git 레포지터리 master 가지에
있는 내용을 heroku 가지로 올린다.**

아마 오류가 날겁니다.

?!

```
remote: !      Error while running '$ python manage.py collectstatic --noinput'.
remote:      See traceback above for details.
remote:
remote:      You may need to update application code to resolve this error.
remote:      Or, you can disable collectstatic for this application:
remote:
remote:      $ heroku config:set DISABLE_COLLECTSTATIC=1
remote:
remote:      https://devcenter.heroku.com/articles/django-assets
remote: !      Push rejected, failed to compile Python app.
remote:
remote: !      Push failed
remote: Verifying deploy...
remote:
remote: !      Push rejected to cryptic-ocean-17174.
remote:
To https://git.heroku.com/cryptic-ocean-17174.git
! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'https://git.heroku.com/cryptic-ocean-17174.git'
(myvenv)
```

**원래 배포는 오류가 많아야 제맛.
헤로쿠는 특히나 예민함
근데 이 새끼 의외로 굉장히 친절ㄱ**

```
myoun@myoun MINGW64 ~/Desktop/Heroku_Practice (master)
$ heroku config:set DEBUG_COLLECTSTATIC=1
» Warning: heroku update available from 7.22.4 to 7.24.1.
Setting DEBUG_COLLECTSTATIC and restarting ● cryptic-ocean-17174... done, v3
DEBUG_COLLECTSTATIC: 1
(myvenv)
```

**heroku config:set DEBUG_COLLECTSTATIC=1
복불됨**

치라니까 쳐봅시다.

근데 아마 또 안될걸 ㅋㅋㅋㅋㅋ

```
remote: !    Push rejected, failed to compile Python app.
remote:
remote: !    Push failed
remote: Verifying deploy...
remote:
remote: !    Push rejected to cryptic-ocean-17174.
remote:
To https://git.heroku.com/cryptic-ocean-17174.git
! [remote rejected] master -> master (pre-receive hook declined)
error: failed to push some refs to 'https://git.heroku.com/cryptic-ocean-17174.git'
(myvenv)
myoun@myoun MINGW64 ~/Desktop/Heroku Practice (master)
```

```
$ heroku config:set DISABLE_COLLECTSTATIC=1
» Warning: heroku update available from 7.22.4 to 7.24.1.
Setting DISABLE_COLLECTSTATIC and restarting ● cryptic-ocean-17174... done, v4
DISABLE_COLLECTSTATIC: 1
(myvenv)
myoun@myoun MINGW64 ~/Desktop/Heroku Practice (master)
```

heroku config:set DISABLE_COLLECTSTATIC=1

다시 git push heroku master

**내가 이 오류를
어떻게 해결했을까?**

정답 : 구글링

**이 오류를 구글링하니까
헤로쿠 공식문서가 나왔음.**

<https://devcenter.heroku.com/articles/django-assets>

**나라고 모든 오류를 해결할 수 있는 능력이 있는 것도 아니고
나도 모르는건 하나하나 쳐보면서 개발함**

이제 헤로쿠의 모델을 마이그레이트 해봅시다.

heroku run python manage.py migrate

```
myoun@myoun MINGW64 ~/Desktop/Heroku_Practice (master)
$ heroku run python manage.py migrate
» Warning: heroku update available from 7.22.4 to 7.24.1.
Running python manage.py migrate on ● cryptic-ocean-17174... up, run.7427 (Free)

Operations to perform:
  Apply all migrations: admin, auth, blog, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying blog.0001_initial... OK
  Applying sessions.0001_initial... OK
(myvenv)
myoun@myoun MINGW64 ~/Desktop/Heroku_Practice (master)
```

**makemigrations는 왜 안하냐면
헤로쿠가 자동으로 해주기 때문이라고~**

슈퍼유저도 create해야겠쥬?

heroku run python manage.py createsuperuser

난 간단하게

admin // 아이디

admin@naver.com // 이메일

Admin123 // 비밀번호

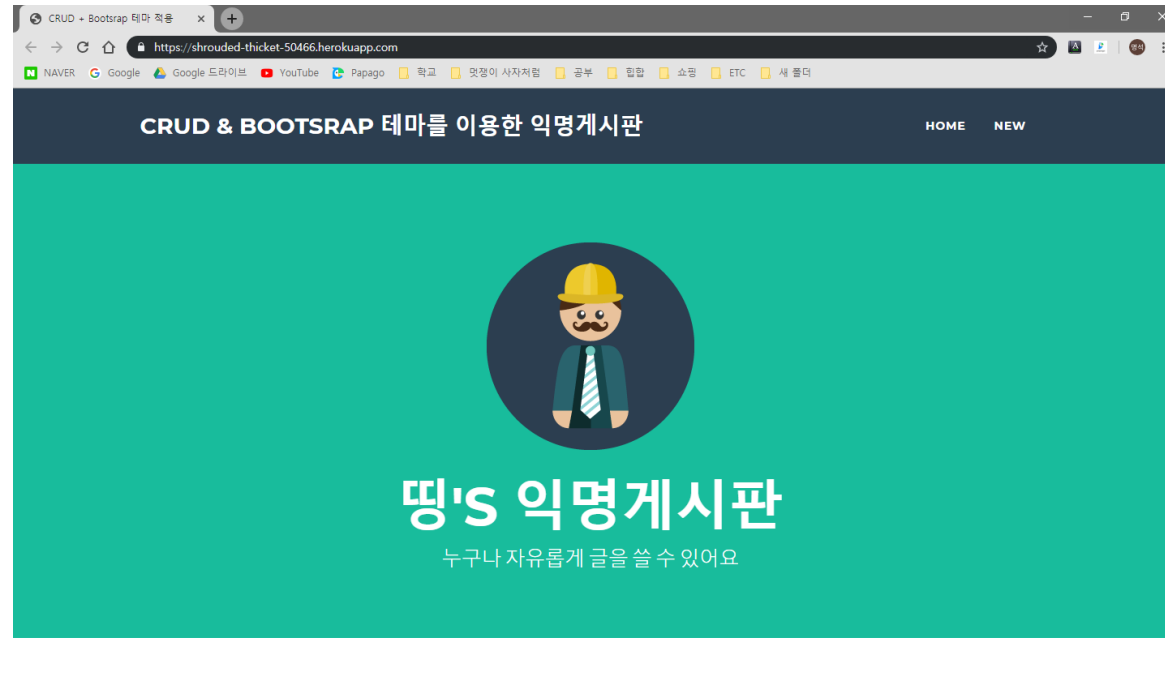
이렇게 만들었음

자기 하고 싶은대로 해도 됩니다.

열어 봅시다.

heroku open

ㅋㅋㅋㅋㅋㅋㅋㅋ
군.



근데 이름이 너무 조깅은걸? ππ
이거 어케 외웁 ㅌㅌ.



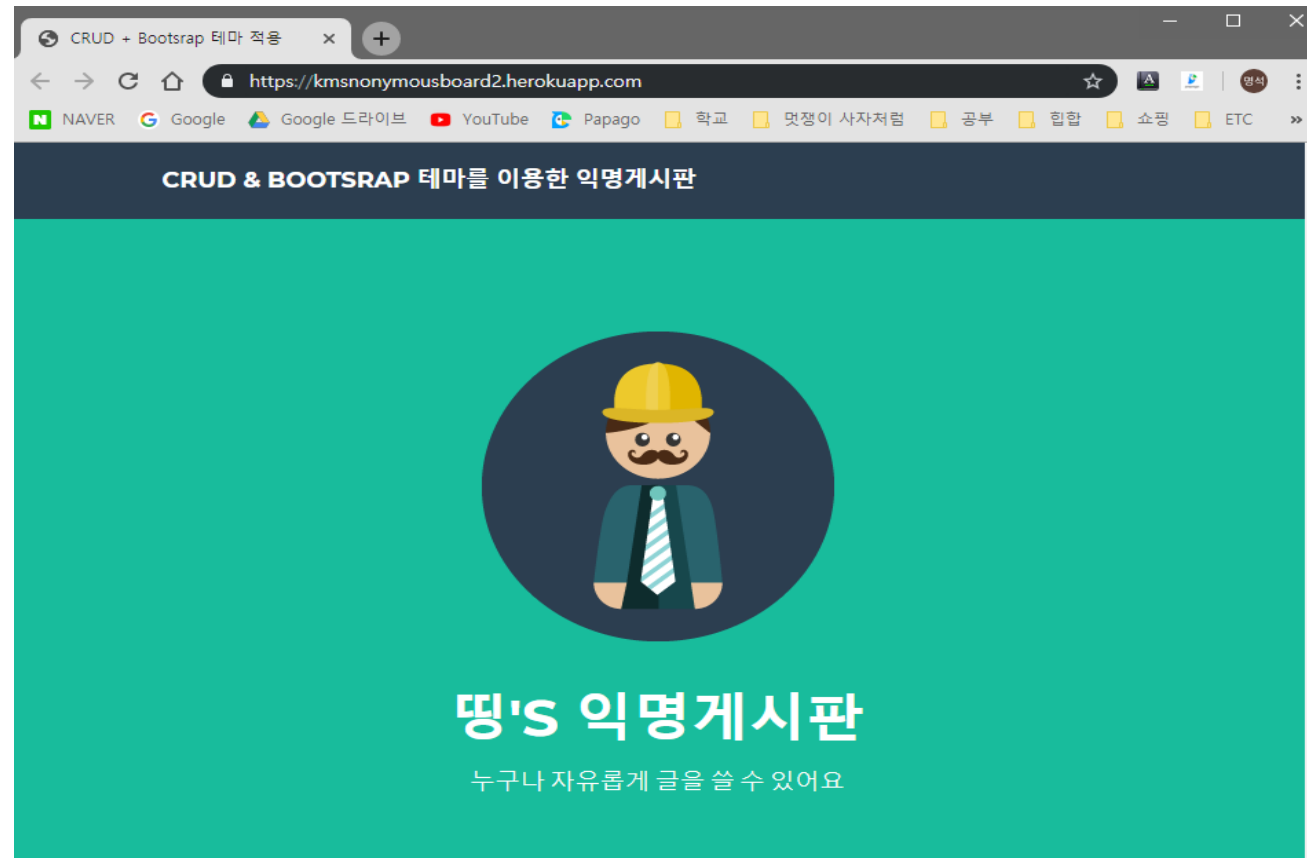
<https://shrouded-thicket-50466.herokuapp.com>

바꿔봅시다 url

***heroku apps:rename* 바꾸고 싶은 이름**

```
$ heroku apps:rename kmsnonymousboard2
» Warning: heroku update available from 7.22.4 to 7.24.1.
Renaming shrouded-thicket-50466 to kmsnonymousboard2... done
https://kmsnonymousboard2.herokuapp.com/ | https://git.heroku.com/kmsnonymousboard2.git
Git remote heroku updated
! Don't forget to update git remotes for all other local checkouts of the app.
(myvenv)
myoun@myoun-MINGW64 ~ /Desktop/Heroku-Practice (master)
```


히히



TE!

헤로쿠 하면서 볼수 있는 오류들 해결 방법

- 1. `error=H10` / `Error=H14`
`Procfile` 등 오타 확인할 것**
- 2. `error=H10`
`heroku ps:scale web=1`**
- 3. 이외에도 존나 다양함 진짜 개많음**

오류 수정 후 해야할 것

말했듯이 헤로쿠는 깃에 올라가 있는 내용을 기반으로 작동됩니다.
즉슨, 오타가 있다거나 내용을 수정한 경우 깃에 푸시하고
한번 더 다시 헤로쿠에 푸시해야 합니다

로컬에 있는 프로젝트



git



Heroku

로컬에 있는 프로젝트 수정

git add .
git commit -m "커밋메세지"
git push origin master

Git push heroku master