텍스트, 스크린샷, 폰트, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

**필독사항**

1. **sqlolchemy 같은 SQL 문법을 작성하지 않고 Python 코드로만 데이터베이스를 관리하는 모듈 사용 금지**
2. **ORM 모듈 사용 금지**

orm이란(Object Relational Manager)은 객체-관계 mapping을 의미합니다. 객체 지향 프로그래밍 언어를 사용해 호환되지 않는 type 시스템 간 데이터를 변환하기 위한 프로그래밍 기술입니다.

즉 프로그램을 만들기 위해 데이터가 필요하고 데이터는 DB에서 관리하고 있습니다. DB에서 쿼리로 원하는 데이터를 가져오려면 원래 SQL Tool 사용이 필요했지만 현재 프로그램 안에서 sql의 동작을 대신해주는 ORM 기능입니다.

ORM을 통해 쿼리를 실행할 필요 없이 프로그래밍 방식으로 안전하고 편리하게 프로그램 안에서 DB에 연결할 수 있습니다.

sqlolchemy란 python에서 사용할 수 있는 ORM중 하나입니다. 다른 ORM 라이브러리와 다른 점 중 하나로는 자체적으로 스키마를 생성하지 않는다는 것입니다.

1. **pymysql말고도 sql 문법을 수동으로 작성하여 사용할 수 있으면 됩니다.**

즉 pymysql을 사용하거나 sql 문법을 수동으로 사용할 수 있으면 된다는 의미로 받아드렸습니다.

**구현 및 기록 과제**

**3) 디자인도 신경 쓰는 것을 추천 합니다.[UI]**

Bootstrap 기능을 찾아보았습니다.

Bootstrap 홈페이지 -> Download -> CDN vis jsDelivr 부분

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

해당 부분에서 가장 첫 줄을 head에서 맨 밑쪽에 넣어줍니다.

두 번째 줄을 body의 맨 마지막 줄에 넣어줍니다

세,네번째 줄을 두 번째 줄 위에 넣어줍니다.

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이후 btn을 넣어 간단하게 테스트해보았습니다.

텍스트, 스크린샷, 멀티미디어, 폰트이(가) 표시된 사진

자동 생성된 설명

정삭적으로 작동하는 것을 확인했습니다.

bootstrap 기능을 이용해 디자인을 진행하였습니다.

1. **학습노트**

과제를 진행하며 구글링과 학습을 진행한 순서대로 노트를 작성했습니다.

우선 pymysql을 사용하기 위해 pip install pymysql로 설치를 해준 후 import pymysql로 불러와줍니다.

이후 pymysql.connect 함수를 이용해 내 컴퓨터의 루트 계정으로 비밀번호를 설정한 후 불러올 때 utf8기반으로 한다고 설정을 해 줍니다.

이후 정상적으로 돌아가는지 테스트 진행 도중

텍스트, 스크린샷, 소프트웨어, 폰트이(가) 표시된 사진

자동 생성된 설명

구글링해보니 cryptography라는 모듈이 설치되어 있어야 한다고 해서 pip install cryptography 를 해주었습니다.

처음에 공부했던 내용(이론적인 부분)과 직접 데이터베이스를 만들어서 연결하는 과정은 조금 달라서 공부를 다시 했습니다. 데이터베이스를 만드는 과정도 오타가 있었지만 발견하지 못해 MySQL을 잘 못 설치한줄 알고 삭제했다 다시 깔기도 했었습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

MYSQL을 켜서 free\_board라는 이름의 데이터베이스를 생성했습니다. 이 데이터베이스는 utf8문자를 사용하도록 세팅하고 utf8\_general\_ci로 대소문자 구분 없이 사용할 수 있도록 세팅했습니다. ci는 case-insensitive의 줄인말입니다.

텍스트, 스크린샷, 폰트, 블랙이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 블랙이(가) 표시된 사진

자동 생성된 설명

임의의 값 넣어두고

pymysql.connect()메소드를 사용해 MySQL에 연결합니다. 이때 주요 파라미터를 검색해 보았을 때

host : 접속할 mysql server 주소

port : 접속할 mysql server의 포트 번호

user : mysql ID

passwd : mysql ID의 암호

db : 접속할 데이터베이스

charset=’utf8’ : mysql에서 select하여 데이터를 가져올 때 한글이 깨질 수 있어서 연결 설정해둠

이후 Connection 객체로부터 cursor() 메서드 호출해 Cursor 객체를 가져왔습니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이후 print문을 이용해 잘 연결되었는지 확인 했습니다

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

render\_template는 HTML 템플릿 파일을 렌더링해 HTML 페이지를 생성합니다. 받아온 값을 인자로 넣어 보내주고

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

index.html 파일에서 받은 인자를 for문을 이용해 출력했습니다.

수정 버튼과 삭제 버튼도 우선 만들어 두었습니다. data[0]에 오는건 db에서 설정한 num입니다. 번호를 이용해 구분하고 경로를 앞서 설명한 render\_template를 통해 랜더링을 했으므로 Flask에서 {{}}는 변수나 표현식을 표현하는 구문입니다.

첫 테스트시 에러가 나왔는데 검색해보니 flask파일이 같은 ‘/server.py’에 있을 때 /templates/index.html 위치에 있어야 파일을 찾아 랜더링 할 수 있는 것을 알게되었습니다.



그 밑에 글쓰기 버튼을 만들었습니다.

글쓰기 버튼을 눌렀을 때 해야 할 일은 GET방식과 POST 방식으로 구분해야 합니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

우선 글쓰기를 눌렀을 때 사용자가 보여주는 페이지는 render\_template(‘write.html’)로 write.html 페이지를 보여줍니다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

write.html의 구성입니다. <form>태그를 이용해 데이터를 전송합니다. 이때 url\_for을 이용해 전달될 경로를 가리킵니다. write로 경로를 설정해주면 server.py에서 write함수에서 if문 POST 부분에서 데이터를 받아올 수 있게 됩니다.

class=”form-group” 그룹화를 해주고 button을 눌렀을 때 데이터를 보냅니다.

텍스트, 스크린샷, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

다시 write함수로 돌아와 request.method 함수를 이용해 POST방식으로 데이터를 받아왔을 때 request.form을 이용해 원하는 데이터를 추출해냅니다. write.html을 보면 제목을 입력하는 곳의 name=title로 설정해두고 글쓴이는 writer, 짧은 코멘트는 context로 설정해두었습니다.

쓴 글을 db에 저장해 두어야 하는데 쿼리문을 execute를 이용해 쿼리문을 입력합니다.

db.commit을 이용해 db 변경사항을 영구적으로 저장을 한 후 redirect해서 메인 페이지를 다시 보여주게 됩니다.

다음은 수정 기능입니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

수정 또한 GET방식과 POST 방식으로 구분을 합니다. 그리고 edit함수는 인자로 num을 받게 됩니다. 게시글은 각각 고유의 num이 지정되어 있습니다. 따라서 db에서 일치하는 num을 찾아 해당 데이터를 가져옵니다. 이후 edit.html에 인자로 보내게됩니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

edit.html에서 게시글의 경로는 고유번호로 /edit/num 으로 설정됩니다. 인자로 받은 데이터를 value값에 넣어 보여주면 됩니다. 다시 edit함수로 돌아옵니다.

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

edit함수의 POST방식인 경우 write와 마찬가지로 request.form을 이용해 값을 받아오고 cursor.execute로 쿼리문을 실행하고 commit을 이용해 저장합니다. 이후 redirect시켜 메인 페이지를 다시 보여줍니다.

다음은 delete 기능입니다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

처음에 delete 기능을 구현할 때 POST 방식만 있으면 된다고 생각했습니다. 인자로 고유 번호인 num을 받고 해당 num을 db에서 삭제하고 redirect만 시켜주면 되겠구나 생각을 했고 이건 요청을 받은걸 처리하는거니 POST라 생각했습니다. 계속해서 오류가 나서 DELETE 쿼리를 잘 못 알고 있었나 싶어 구글링을 진행했고 delete 구현을 이렇게 하는게 아닌가 싶어 조금 더 찾아보았었습니다. 이후 에러 메시지 

405를 검색해 보았습니다. GET<->POST를 바꾸면 대부분 해결되는 문제라고 해서 GET으로 바꿨더니 바로 해결이 되었었습니다. 조금 생각을 해보니 요청한 것을 보여줘야 하니 GET으로 하는게 맞다고 생각이 들었습니다.

마지막으로 검색 기능입니다. 가장 많이 고민했던 것 같습니다. 아래는 최종 index 함수입니다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

아래는 처음 index.html을 만들었을 때와 검색 기능을 추가하고 난 뒤의 코드 차이입니다.

텍스트, 스크린샷, 소프트웨어, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

처음에 검색 페이지를 따로 만들까 고민했습니다. 메인 페이지에 만드는게 깔끔할 것 같아 메인 페이지를 어떻게 수정할지 고민했었습니다.

1. 우선 처음 들어갔을 때 메인 페이지는 전부 다 보여야 한다
2. 검색을 했을 때 기능이 총 세 가지가 있어야 한다

를 생각하고 코드를 구성했습니다.

우선 index.html에 조건 검색을 버튼을 눌러 검색내용을 입력하고 제목 기준 검색 버튼, 내용 기준 검색 버튼, 전체 기준 검색 버튼을 만들어 했었습니다. 너무 이상해서 구글링을 스크롤을 이용한 조건 검색 이런식으로 조금 하다 보니 생각한 것과 일치한 기능이 나왔습니다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이후 다시 index함수로 돌아와

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

처음에 입력한 값들을 제대로 받아오는지 테스트를 진행했었습니다.

텍스트, 폰트, 스크린샷, 번호이(가) 표시된 사진

자동 생성된 설명

잘 받아오는 것을 확인하고 코드를 짰습니다.

search\_option을 if문 세 개로 전체 기준 검색, 제목 기준 검색, 내용 기준 검색으로 나눴습니다. 이후 쿼리문을 짤 때 조금 헤맸습니다. LIKE %찾으려는문장%로 앞뒤로 %를 붙여야 하는데 %%s% 로 했더니 오류가 나서 구글링을 했더니 %s 인자로 전달하는 부분에 %+search\_name+%를 붙여서 보낼 수 있다는걸 알게 되었습니다.

제목 기준 검색과 내용 기준 검색은 WHERE 부분을 title과 context로 나눠주면 되고 전체 기준은 작성자 까지 검색이 되도록 구현했습니다. for문을 돌려 title LIKE %s를 OR로 이어붙여 쿼리를 만들었습니다.

텍스트, 스크린샷, 번호, 라인이(가) 표시된 사진

자동 생성된 설명

메인 페이지

텍스트, 번호, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

제목 기준 title 검색

텍스트, 라인, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

내용 기준 검색

텍스트, 라인, 번호, 스크린샷이(가) 표시된 사진

자동 생성된 설명

전체 기준 검색

다 만들고 나서 과제를 git에 올릴 때 db를 어떻게 올리지 생각을 했습니다. 구글링을 해보니 docker 파일로 만들어서 배포, aws 등등 여러가지가 있었는데 생각해보니 실행할 때 여기서 db를 만들면 되지 않나 생각이 들었습니다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

그래서 프로그램 젤 처음에 free\_board라는 데이터베이스가 존재하지 않는다면 CREATE하도록 하고 board라는 테이블이 존재하지 않는다면 생성하도록 만들었습니다.

**코드 첨부**

server.py

import pymysql

from flask import Flask, render\_template, url\_for, request, redirect

def create\_database():

    db\_connection = pymysql.connect(host="localhost", user="root", passwd="1234", charset="utf8")

    cursor = db\_connection.cursor()

    cursor.execute("CREATE DATABASE IF NOT EXISTS free\_board")

    cursor.execute("USE free\_board")

    cursor.execute("""CREATE TABLE IF NOT EXISTS board (

                      num INT NOT NULL,

                      title VARCHAR(50) NOT NULL,

                      writer VARCHAR(50) NOT NULL,

                      context VARCHAR(200) NOT NULL)""")

    db\_connection.close()

create\_database()

db = pymysql.connect(host="localhost", user="root", passwd="1234", db='free\_board', charset="utf8")

cursor = db.cursor()

check\_num = 0

app = Flask(\_\_name\_\_)

@app.route('/', methods=['GET', 'POST'])

def index():

    global check\_num

    if request.method == 'POST':

        search\_option = request.form['option']

        search\_name = request.form['search']

        if search\_option == "전체 기준 검색":

            query = "SELECT \* FROM board WHERE "

            columns = ["title", "context", "writer"]

            conditions = []

            for column in columns:

                conditions.append(f"{column} LIKE %s")

            query += " OR ".join(conditions)

            cursor.execute(query, tuple(['%' + search\_name + '%'] \* len(columns)))

            data\_list = cursor.fetchall()

        elif search\_option == "제목 기준 검색":

            cursor.execute("SELECT \* FROM board WHERE title LIKE %s", ('%' + search\_name + '%'))

            data\_list = cursor.fetchall()

        elif search\_option == "내용 기준 검색":

            cursor.execute("SELECT \* FROM board WHERE context LIKE %s", ('%' + search\_name + '%'))

            data\_list = cursor.fetchall()

    else:

        sql = "SELECT \* FROM board"

        cursor.execute(sql)

        data\_list = cursor.fetchall()

        for i in data\_list:

            check\_num = i[0]

        check\_num = check\_num + 1

    return render\_template('index.html', data\_list=data\_list)

@app.route('/write', methods=['GET','POST'])

def write():

    global check\_num

    if request.method == 'POST':

        title = request.form['title']

        writer = request.form['writer']

        context = request.form['context']

        cursor.execute("INSERT INTO board (num, title, writer, context) VALUES (%s,%s,%s,%s)", (check\_num, title, writer, context))

        db.commit()

        check\_num = check\_num + 1

        return redirect(url\_for('index'))

    else:

        return render\_template('write.html')

@app.route('/edit/<num>', methods=['GET','POST'])

def edit(num):

    if request.method == 'GET':

        cursor.execute('SELECT \* FROM board WHERE num = %s', (num))

        data = cursor.fetchall()

        return render\_template('edit.html', data = data[0])

    else:

        title = request.form['title']

        writer = request.form['writer']

        context = request.form['context']

        cursor.execute("UPDATE board SET title = %s, writer = %s, context = %s WHERE num = %s",(title,writer,context,num))

        db.commit()

        return redirect(url\_for('index'))

@app.route('/delete/<num>', methods=['GET'])

def delete(num):

    if request.method=='GET':

        cursor.execute('DELETE FROM board WHERE num = %s', (num))

        db.commit()

    return redirect(url\_for('index'))

if \_\_name\_\_ == "\_\_main\_\_":

    app.run(debug=True)

index.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>게시판</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous"></link>

</head>

<body>

    <div class="container">

        <br>

        <a href="/"><h1 class="text-center">P4C 게시판 assignment</h1></a>

        <br>

        <br>

        <div class="container">

            <form action="{{url\_for('index')}}" method="POST">

                <label for="lang">검색조건</label>

                <select  name="option" id="options">

                    <option value="제목 기준 검색">제목 기준 검색</option>

                    <option value="내용 기준 검색">내용 기준 검색</option>

                    <option value="전체 기준 검색">전체 기준 검색</option>

                </select>

                <input type="text" placeholder="검색할 내용" name="search">

                <button class="btn btn-primary btn-block">검색</button>

            </form>

        </div>

        <table class="table table-hover table-striped text-center" style="border: 1px solid;">

            <thead>

                <tr>

                    <th>번호</th>

                    <th>제목</th>

                    <th>글쓴이</th>

                    <th>짧은코멘트</th>

                </tr>

            </thead>

            <tbody>

                {% for data in data\_list %}

                <tr>

                    <td>{{data[0]}}</td>

                    <td>{{data[1]}}</td>

                    <td>{{data[2]}}</td>

                    <td>{{data[3]}}</td>

                    <td width="130">

                        <a href="/edit/{{data[0]}}" class="btn btn-secondary btn-sm">수정</a>

                        <a href="/delete/{{data[0]}}" class="btn btn-danger btn-delete btn-sm">삭제</a>

                    </td>

                </tr>

                {% endfor %}

            </tbody>

        </table>

        <hr>

        <a href="/write"class="btn btn-outline-info float-right">글쓰기</a>

        <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js" integrity="sha384-I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+hVqc2pM8ODewa9r" crossorigin="anonymous"></script>

        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.min.js" integrity="sha384-BBtl+eGJRgqQAUMxJ7pMwbEyER4l1g+O15P+16Ep7Q9Q+zqX6gSbd85u4mG4QzX+" crossorigin="anonymous"></script>

        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

    </div>

</body>

</html>

edit.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>수정</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous"></link>

</head>

<body>

    <form action="/edit/{{data[0]}}" method="POST">

        <div class="form-group">

            <input type="text" name="writer" value="{{data[1]}}" class="form-control">

        </div>

        <div class="form-group">

            <input type="text" name="title" value="{{data[2]}}" class="form-control">

        </div>

        <div class="form-group">

            <input type="text" name="context" value="{{data[3]}}" class="form-control">

        </div>

        <div class="form-group">

            <button type="submit" class="btn btn-primary btn-block">

                업데이트하기

            </button>

        </div>

    </form>

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js" integrity="sha384-I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+hVqc2pM8ODewa9r" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.min.js" integrity="sha384-BBtl+eGJRgqQAUMxJ7pMwbEyER4l1g+O15P+16Ep7Q9Q+zqX6gSbd85u4mG4QzX+" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

</body>

</html>

write.html

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>쓰기</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN" crossorigin="anonymous"></link>

</head>

<body>

    <div class="container">

        <form action="{{url\_for('write')}}" method="POST">

            <div class="form-group">

                <input type="text" class="form-control" name="title" placeholder="제목">

            </div>

            <div class="form-group">

                <input type="text" class="form-control" name="writer" placeholder="글쓴이">

            </div>

            <div class="form-group">

                <input type="text" class="form-control" name="context" placeholder="짧은 코멘트">

            </div>

            <button class="btn btn-primary btn-block">저장</button>

        </form>

    </div>

    <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js" integrity="sha384-I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+hVqc2pM8ODewa9r" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.min.js" integrity="sha384-BBtl+eGJRgqQAUMxJ7pMwbEyER4l1g+O15P+16Ep7Q9Q+zqX6gSbd85u4mG4QzX+" crossorigin="anonymous"></script>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46cDfL" crossorigin="anonymous"></script>

</body>

</html>

참고자료

<https://thalals.tistory.com/163>

<https://journeytosth.tistory.com/31>

<https://www.youtube.com/watch?v=3N5XHl8EE78&list=PLuHgQVnccGMClNOIuT3b3M4YZjxmult2y&index=12&ab_channel=%EC%83%9D%ED%99%9C%EC%BD%94%EB%94%A9>

<https://www.youtube.com/watch?v=B956r7XzJIA&t=398s&ab_channel=%EC%BD%94%EB%94%A9%EC%99%95%EA%B5%AD>

<https://www.fun-coding.org/post/mysql_basic6.html#gsc.tab=0>

<https://www.freecodecamp.org/korean/news/html-select-taegeu-deurobdaun-menyu-ddoneun-kombo-riseuteureul-mandeuneun-bangbeob/>

https://code-hoon.tistory.com/145

https://velog.io/@hjun0917/%EC%9B%B9%EA%B8%B0%EB%B3%B8-HTMLCSS-%EA%B2%80%EC%83%89-%EA%B8%B0%EB%8A%A5-%EA%B5%AC%ED%98%84%ED%95%98%EA%B8%B0