간단한 Webpage (login site) 구현

10304 김지환

Index

Index

Index

Index

- 제작이유

- 제작이유
  - 백엔드 기초를 잘 배울 수 있는 주제가 없을까?

    로그인 페이지를 만들어보자

- 제작이유
  - 백엔드 기초를 잘 배울 수 있는 주제가 없을까?

    로그인 페이지를 만들어보자

  - 어떠한 방식으로 구현 할껀가?

    flask를 백엔드를 사용하며 SQL로 DB 구현
    html로 프론트 구현

- 구현전 기본 지식

- 구현전 기본 지식
  - SQL
  - HTML 및 js
  - Flask

- SQL

  DB 관리 언어
  테이블로 이루어져 있으며 열과 행으로
  이루어져 저장됨

- SQL

  DB 관리 언어
  테이블로 이루어져 있으며 열과 행으로
  이루어져 저장됨

- HTML 및 js
  - HTML
    웹페이지의 뼈대
    요소를 <p>, <h1>등 테그로 감싸서 표현함.

- SQL

  DB 관리 언어
  테이블로 이루어져 있으며 열과 행으로
  이루어져 저장됨

- HTML 및 js
  - HTML
    웹페이지의 뼈대
    요소를 <p>, <h1>등 테그로 감싸서 표현함.
  - js
    웹페이지의 중심임
    클라이언트와 백엔드를 연결함

- Flask

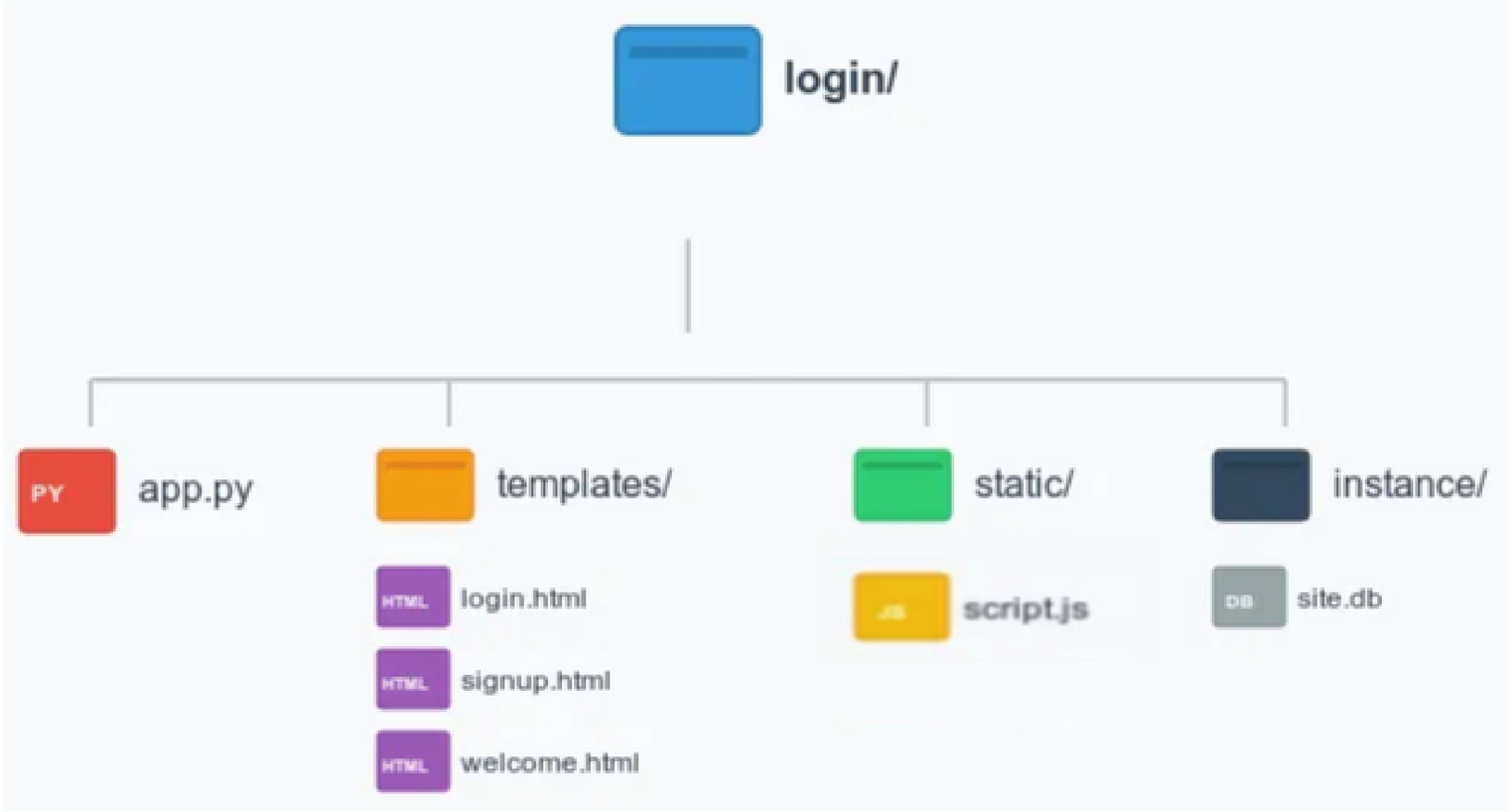
  웹 애플리케이션을 개발하기 위한 마이크로 웹 프레임워크

  라우팅:@app.route를 통해 해당 url로 이동

  서버 시작 및 포트지정:

```
if __name__ == '__main__':
    app.run(debug=True,host='0.0.0.0', port=3000)
```

- 프로젝트 구조


login/
PY
app.py
templates/
HTML
login.html
HTML
signup.html
HTML
welcome.html
static/
JS
script.js
instance/
DB
site.db

- 중요 로직 설명(서버열기(app.py))

```python
from flask import Flask, render_template, request, jsonify, redirect, url_for, session
from flask_sqlalchemy import SQLAlchemy
from werkzeug.security import generate_password_hash, check_password_hash
from functools import wraps
```

```python
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(80), unique=True, nullable=False)
    password_hash = db.Column(db.String(128), nullable=False)

    def set_password(self, password):
        self.password_hash = generate_password_hash(password)

    def check_password(self, password):
        return check_password_hash(self.password_hash, password)

    def __repr__(self):
        return f'<User {self.username}>'
```

```python
@app.route('/')
def index():
    return render_template('login.html')
```

```python
if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=3000)
```

- 중요 로직 설명(회원가입 페이지(signup.html))

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>회원가입</title>
</head>
<body>

    <form id="signupForm">
      <h2>회원가입</h2>
      <input type="text" id="signupUsername" placeholder="아이디" required>
      <input type="password" id="signupPassword" placeholder="비밀번호" required>
      <input type="password" id="signupPasswordConfirm" placeholder="비밀번호 확인" required>
      <button type="submit">가입하기</button>
      <a href="/">로그인으로 돌아가기</a>
    </form>

    <script src="{{ url_for('static', filename='script.js') }}"></script>

</body>
</html>
```

- 중요 로직 설명(회원가입 처리(script.js))

```
const signupForm = document.getElementById('signupForm');
if (signupForm) {
    signupForm.addEventListener('submit', async (event) => {
        event.preventDefault();

        const username = document.getElementById('signupUsername').value;
        const password = document.getElementById('signupPassword').value;
        const passwordConfirm = document.getElementById('signupPasswordConfirm').value;

        if (password !== passwordConfirm) {
            alert('비밀번호가 일치하지 않습니다.');
            return;
        }
```

- 중요 로직 설명(회원가입 처리(app.py))

```python
@app.route('/signup', methods=['GET', 'POST'])
def signup():
    if request.method == 'POST':
        if request.is_json:
            data = request.get_json()
            username = data.get('username')
            password = data.get('password')

            existing_user = User.query.filter_by(username=username).first()
            if existing_user:
                return jsonify({'success': False, 'message': '이미 존재하는 아이디입니다.'}), 409

            new_user = User(username=username)
            new_user.set_password(password)

            db.session.add(new_user)
            db.session.commit()

            return jsonify({'success': True, 'message': f'{username}님, 회원가입이 완료되었습니다!'}), 201
        else:
            return jsonify({'success': False, 'message': '잘못된 요청 형식입니다.'}), 400
    else:
        return render_template('signup.html')
```

- 중요 로직 설명(회원가입 처리(js))

```
            const data = { username, password };

            try {
                const response = await fetch('/signup', {
                    method: 'POST',
                    headers: { 'Content-Type': 'application/json' },
                    body: JSON.stringify(data)
                });

                const result = await response.json();

                if (response.ok) {
                    alert(result.message);
                    window.location.href = '/';
                } else {
                    alert('회원가입 실패: ' + (result.message || '알 수 없는 오류'));
                }

            } catch (error) {
                console.error('회원가입 Fetch 에러:', error);
                alert('회원가입 중 서버와 통신 오류가 발생했습니다.');
            }
        });
    }
});
```

- 중요 로직 설명(로그인 페이지(login.html))

```html
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>로그인</title>
</head>
<body>

    <form id="loginForm">
      <h2>로그인</h2>
      <input type="text" id="loginUsername" placeholder="아이디" required>
      <input type="password" id="loginPassword" placeholder="비밀번호" required>
      <button type="submit">로그인</button>
      <a href="signup">회원가입 하러가기</a>
    </form>

    <script src="{{ url_for('static', filename='script.js') }}"></script>

</body>
</html>
```

- 중요 로직 설명(로그인 처리(script.js))

```
const loginForm = document.getElementById('loginForm');
if (loginForm) {
    loginForm.addEventListener('submit', async (event) => {
        event.preventDefault();

        const username = document.getElementById('loginUsername').value;
        const password = document.getElementById('loginPassword').value;
```

- 중요 로직 설명(로그인 처리(app.py))

```python
@app.route('/login', methods=['POST'])
def login():
    if request.is_json:
        data = request.get_json()
        username = data.get('username')
        password = data.get('password')

        user = User.query.filter_by(username=username).first()

        if user and user.check_password(password):
            session['logged_in'] = True
            session['username'] = username
            return jsonify({'success': True, 'message': '로그인 성공!'}), 200
        else:
            return jsonify({'success': False, 'message': '아이디 또는 비밀번호가 올바르지 않습니다.'}), 401
    else:
        return jsonify({'success': False, 'message': '잘못된 요청 형식입니다.'}), 400
```

- 중요 로직 설명(로그인 처리(script.js))

```
const data = { username, password };

try {
    const response = await fetch('/login', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(data)
    });

    const result = await response.json();

    if (response.ok) {
        alert(result.message);
        window.location.href = '/welcome';
    } else {
        alert('로그인 실패: ' + (result.message || '알 수 없는 오류'));
    }

} catch (error) {
    console.error('로그인 Fetch 에러:', error);
    alert('로그인 중 서버와 통신 오류가 발생했습니다.');
}
```

- 중요 로직 설명(환영 페이지(welcome.html))

```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>환영합니다!</title>
</head>
<body>
    <div class="welcome-container">
        <h1>환영합니다!</h1>
        <p>성공적으로 로그인하셨습니다.</p>
        <button id="logoutButton">로그아웃</button>
        <p></p>

    <script src="{{ url_for('static', filename='script.js') }}"></script> </body>
</html>
```

- 중요 로직 설명(로그아웃 처리(app.py))

```python
@app.route('/logout')
def logout():
    session.pop('logged_in', None)
    session.pop('username', None)
    print("로그아웃 요청 수신됨.")
    return redirect(url_for('index'))
```

- 중요 로직 설명(로그아웃 처리(script.js))

```
document.addEventListener('DOMContentLoaded', () => {
    const logoutButton = document.getElementById('logoutButton');
    if (logoutButton) {
        logoutButton.addEventListener('click', () => {
            window.location.href = '/logout';
        });
    }
});
```

- 시연

감사합니다

Thank you