웹 페이지, '씬 스토리 맵'의 UI 기본 기능 구현에 필요한 언어와 코드의 구조 정리.

_ by 정동진.

<1>_ 지도 상 **피드 확인 및 검색용** 사이드 창의 1~2 기능을 구현하기 위해 필요한 언어들과 코드의 구조들.

위 기능을 구현하기 위해서는 **백엔드와 프론트엔드**가 협력하여 작동해야 합니다.

1. **백엔드 (Flask + Beautiful Soup)**
   - 검색 키워드를 받아 웹 크롤링(Beautiful Soup)으로 장소 정보를 가져옴.
   - Google Maps API와 연동하여 장소 좌표(위도, 경도) 정보를 조회.
   - 데이터베이스에 피드 및 장소 정보 저장 및 제공.
2. **프론트엔드 (HTML, CSS, JavaScript + Google Maps API)**
   - 검색 창에서 사용자가 키워드를 입력하면 Flask 서버로 요청.
   - 서버에서 받은 장소 정보를 Google Maps API를 사용해 지도에 표시.
   - 검색한 장소가 지도에서 자동으로 이동되도록 설정.
   - 해당 장소에 피드가 있으면 마커를 클릭하면 피드 정보를 표시.

구현 코드

1. 백엔드 (Flask + Beautiful Soup + Google Maps API)

(1) `app.py` - Flask 서버 구현

from flask import Flask, request, jsonif.py

2. 프론트엔드 (HTML, CSS, JavaScript + Google Maps API)

(1) `templates/index.html`

1.html

설명

1. Flask 백엔드 (`app.py`)

- `get_place_info(query)` : Beautiful Soup을 이용해 네이버 플레이스에서 장소 검색 결과를 가져옴.
- `get_place_coordinates(place_name)` : Google Maps API를 이용해 장소의 위도/경도를 검색.
- `/search` : 검색 요청을 받아서 장소 정보를 JSON 형태로 반환.
- `/` : 메인 페이지 (`index.html`)를 렌더링.

2. 프론트엔드 (`index.html`)

- `initMap()` : Google Maps API를 이용해 기본 지도 설정.
- `searchPlace()` : 사용자가 검색하면 Flask 서버(`/search`)에 요청을 보내고, 받은 데이터를 지도에 표시.
- `google.maps.Marker` : 검색된 장소마다 지도에 마커 추가.
- `google.maps.InfoWindow` : 마커 클릭 시 장소 정보를 표시.
- `map.setCenter()` : 검색된 장소로 지도 자동 이동.

결과

1. 사용자가 **검색창에 키워드를 입력**하고 검색 버튼 클릭.
2. Flask 서버가 **Beautiful Soup을 이용해 네이버 플레이스 크롤링**, 장소 목록 가져옴.

3. Flask가 **Google Maps API를 이용해 각 장소의 위도/경도 가져옴**.
4. 검색 결과를 프론트엔드에 **JSON 형태로 반환**.
5. 프론트엔드에서 **Google Maps API를 이용해 지도에 마커 표시**.
6. 마커를 클릭하면 해당 장소의 정보를 표시.

다음 단계

- 검색 결과를 지도 아래 리스트로 표시.
- 데이터베이스를 활용해 피드(게시물) 저장.
- 피드(게시물)가 존재하는 경우 마커 색상 변경.

이제 이 기능을 실행하면 **네이버 맵처럼 장소 검색이 가능하고, 지도에서 자동 이동 & 마커 표시 기능**이 동작하게 됩니다.

<1> 피드 확인 및 피드 검색용 창의 3 ~ 6 까지의 기능을 구현하기 위해 필요한 언어와 코드들의 구조.

이제 마커 클릭 시 해당 장소의 **피드 목록을 표시**하고, **장소 정보 버튼을 추가**하며, **사진 및 피드 UI를 구성**하는 기능을 구현해야 합니다.

---

1. 전체적인 구조

기능	사용 기술
장소 검색 & 지도 이동	Google Maps API (JavaScript), Flask (백엔드)
웹 크롤링 (장소 정보)	Beautiful Soup (Python)
DB에 피드 저장 및 제공	Flask (백엔드), SQLite 또는 MySQL
마커 클릭 시 피드 목록 표시	JavaScript (프론트), Flask (백엔드)
좋아요 & 댓글 기능	JavaScript (AJAX), Flask (API)
이미지 슬라이더	JavaScript (Swiper.js)
음악 삽입 및 재생	HTML5 Audio, JavaScript

---

2. 상세 기능 구현

1. 사용자가 마커를 클릭하면 해당 장소의 피드를 가져와 **화면 왼쪽에 표시**.

2. **장소 정보 버튼을 추가**하고, 클릭하면 크롤링한 메뉴, 가격, 전화번호, 운영시간을 표시.
3. 피드는 **좋아요 수 & 댓글 수 기준으로 정렬**.
4. **사진 & GIF를 가로 슬라이드(좌우 스크롤)로 볼 수 있도록 구현**.
5. **피드 하단에 좋아요, 댓글, URL 공유 버튼 추가**.
6. **음악 삽입 및 재생 기능 구현**.

---

3. 데이터베이스 설계 (Flask + SQLite)

from flask_sqlalchemy import SQLAlchemy.py

---

4. Flask 백엔드 (`app.py`)

from flask import Flask, request, jsonif.py

---

5. 프론트엔드 (`index.html`)

1.html

결과

✅ 마커 클릭 시 **해당 장소의 피드 목록 & 장소 정보 버튼 표시**

✅ **사진 슬라이드 기능 & 음악 재생**

✅ **좋아요 & 댓글 기능 포함**

<2> _ **디폴트 메인 화면**과 툴바를 구현하기 위해 필요한 언어와 코드의 구조.

이제 **툴바를 추가하여 메인 화면을 전환**하고, **기본적으로 지도와 검색창이 열려 있도록 설정**하는 기능을 구현해야 합니다.

1. 기능별 기술 선택

기능	사용 기술
툴바 버튼 (피드, 지도, 프로필 창 전환)	HTML, CSS, JavaScript
버튼 클릭 시 화면 전환 (탭 변경 효과)	JavaScript (DOM 조작)
기본 화면 (지도 + 검색창 활성화)	Google Maps API (JavaScript)
각 화면 데이터 제공	Flask (백엔드, API)
프로필 정보 (팔로워, 팔로잉, 게시물 수)	Flask + 데이터베이스 (SQLite/MySQL)

2. 메인 화면 구조

- **좌측에 툴바**를 배치하고 3개의 버튼을 배치
  (1)
  피드 작성 (새 게시물 작성)
  (2)
  지도 검색 (디폴트 화면)
  (3)
  프로필 관리 (사용자 정보)
- 기본적으로 **지도 검색창과 피드가 있는 창이 보이도록 설정**
- 버튼 클릭 시 **각 창으로 전환**되도록 구현

3. HTML/CSS 구조 (`index.html`)

1.html

4. Flask 백엔드 코드 (`app.py`)

```
from flask import Flask, request, jsonif.txt
```

---

5. 기능 구현 요약

✅ **툴바 추가 (피드, 지도, 프로필 버튼 3개 배치)**

✅ **기본적으로 지도와 검색창이 보이는 화면 활성화**

✅ **버튼 클릭 시 화면 전환 (피드 작성 / 지도 검색 / 프로필 관리)**

✅ **Flask API를 통해 프로필 정보 불러오기**

이제 전체적인 화면 전환 구조가 갖춰졌어요!

<3>_ 자신만의 **피드 작성 및 게시를 위한 사이드 창**을 구현하기 위해 필요한 언어와 코드의 구조.

사용 기술

✅ **백엔드:** Flask (Python) - 데이터 저장 및 API 처리

✅ **웹 크롤링:** Beautiful Soup - 장소 정보 자동 수집 (예: 네이버, 구글 등)

✅ **지도 API:** Google Maps API - 지도 검색 및 위치 표시

✅ **프론트엔드:** HTML/CSS/JavaScript - UI 및 동적 기능 구현

✅ **데이터베이스:** SQLite (또는 MySQL, PostgreSQL) - 피드 데이터 저장

---

1. 기능별 코드 설계 및 구조

🔷 1) 기본 화면 구성

- 사용자가 접속하면 기본 화면에 지도와 검색창이 보인다.
- 툴바에서 "피드 작성" 버튼을 클릭하면 피드 작성 창으로 전환된다.

구현 방식:

- `index.html`에 `display: none;`을 활용해 필요한 UI만 보이도록 설정
- JavaScript로 페이지 전환 기능 추가

🔷 2) 피드 작성 기능

- **사진 업로드:** 로컬 파일 선택 또는 URL 입력
- **코멘트 입력:** 사용자가 텍스트로 후기 작성 가능
- **음악 설정:** 사용자가 BGM으로 추가할 음악 URL 입력 가능
- **게시하기 버튼:** 작성한 데이터를 Flask 서버로 전송하고 DB에 저장

구현 방식:

- JavaScript에서 `FormData`를 활용하여 이미지 파일 및 텍스트 데이터를 백엔드로 전송
- Flask에서 API 엔드포인트를 만들어 데이터 저장

🔷 3) 게시 후 처리

- 피드가 성공적으로 저장되면 자동으로 지도 화면으로 이동

구현 방식:

- JavaScript `window.location.href`를 활용해 페이지 전환

---

2. HTML & JavaScript (프론트엔드)

1.html

---

3. Flask 백엔드 (`app.py`)

from flask import Flask, request, jsonif.py

---

4. BeautifulSoup을 활용한 장소 정보 크롤링.

import requests.py

최종 요약

✅ **Flask (백엔드) + SQLAlchemy (DB) → 피드 데이터 저장**

✅ **HTML/CSS/JavaScript → UI 및 피드 작성 기능**

✅ **Google Maps API → 지도 검색 및 마커 표시**

✅ **Beautiful Soup → 네이버/구글에서 장소 정보 크롤링**

<4> '씬 스토리맵'의 **계정 프로필 화면**을 구현하기 위해 필요한 기술과 구조.

---

1. 사용 기술

- **백엔드 (Flask)**
  - 사용자 인증 및 세션 관리 (`Flask-Login`, `Flask-SQLAlchemy`)
  - 데이터베이스에서 사용자 정보 및 게시물 가져오기
  - 프로필 수정 및 저장 API
- **프론트엔드 (HTML/CSS/JavaScript)**
  - `HTML/CSS`: 프로필 페이지 UI 디자인
  - `JavaScript (AJAX, Fetch API)`: 비동기 요청으로 데이터 로드 및 수정
- **데이터베이스 (SQLAlchemy - SQLite, PostgreSQL, MySQL)**
  - 사용자 테이블 (이름, 이메일, 프로필 이미지, 소개글 등)
  - 게시물 테이블 (작성한 피드 목록)
  - 팔로우 테이블 (팔로워 및 팔로잉 관리)
- **BeautifulSoup (웹 크롤링 - 선택적)**
  - 다른 플랫폼(예: 인스타그램, 네이버 블로그)에서 관련 게시물 크롤링 (법적 이슈 고려)
- **Google Maps API**

- 장소별 게시물 관리 (위치 기반으로 필터링)

---

2. 코드 구조

1) Flask 백엔드 - 계정 정보 조회 API

---

2) HTML + JavaScript (프로필 페이지)

---

3. 기능 요약

✅ **로그인 사용자만 프로필 페이지 접근 가능**

✅ **사용자의 기본 정보 (이름, 이메일, 프로필 사진, 소개글) 표시**

✅ **JavaScript로 백엔드에서 프로필 데이터 가져오기**

✅ **프로필 수정 기능 추가 가능 (AJAX 활용)**

이제 원하는 기능을 추가로 발전시키면서, **게시물 보기/팔로우 기능**도 구현할 수 있습니다.