프로젝트 실행 전 작업 순서

1. 현재 만들어진 Ssavi\_db DROP
2. 동봉된 SQL문에서 Ssavi\_db 스키마 생성
   1. Use Ssavi\_db; 까지 실행하고 수행
3. users\_app/models.py에 다음 key 생성
4. class User(AbstractUser):
5. # 이메일은 기본적으로 묻는 관계로 초기 생성 시 필요 없음
6. # 이름, 선호장르 필드
7. user\_name = models.CharField(max\_length=100)
8. user\_genre = models.CharField(max\_length=200)

터미널 venv 환경 activate

Venv 가상환경에서 python manage.py makemigrations

오류 없이 이뤄지면 migrate 해주기

4. 동봉된 SQL문 남은 명령어 실행시키기.

table 잘 만들어졌는지 확인

venv 가상환경에서 python manage.py inspectdb

DB 정보 불러오면 Ssavi\_app/models.py에 우리가 사용할 DB class 연결.

기초 작업 완료.

데이터 크롤링

크롤링할 데이터 목록

1. 현재 등록된 장르는 총 12개. 장르 하나 당 100 앨범씩 가져와 총 1200개의 앨범 DB에 등록.
   1. 현재 search(q=”genre:POP”, type=’artist’) 검색하면 pop이란 장르의 노래, 앨범들이 검색된다.

앨범 내의 track id 통해서 track에 정보 추가.

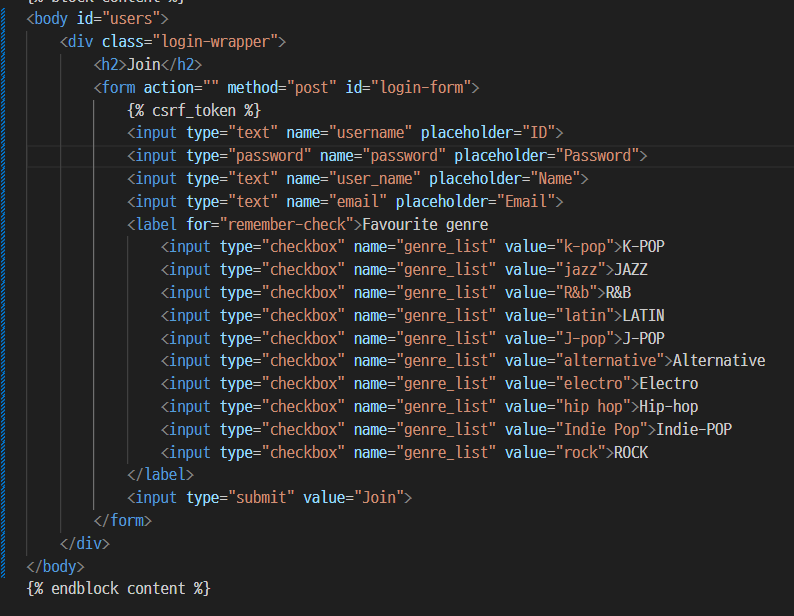
Audio\_feature 분석에 필요한 12개의 장르별 track\_id 이용하여 값 저장

로그인, 회원가입 기능 구현

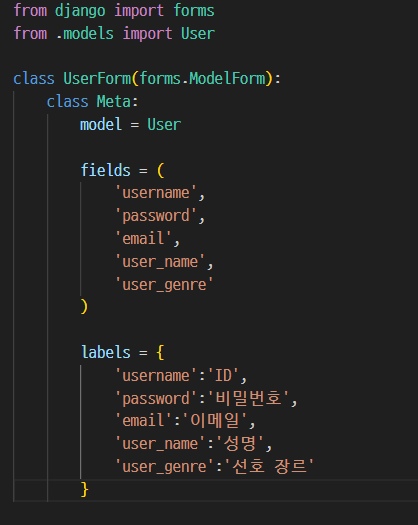
로그인 기능은 예제의 폼을 사용하면 될 것 같은데, 회원가입에서 checkbox로 선택한 정보를 DB로 어떻게 보내야 할까?

우선 블로그에서 찾은 방법은 아래와 같다.

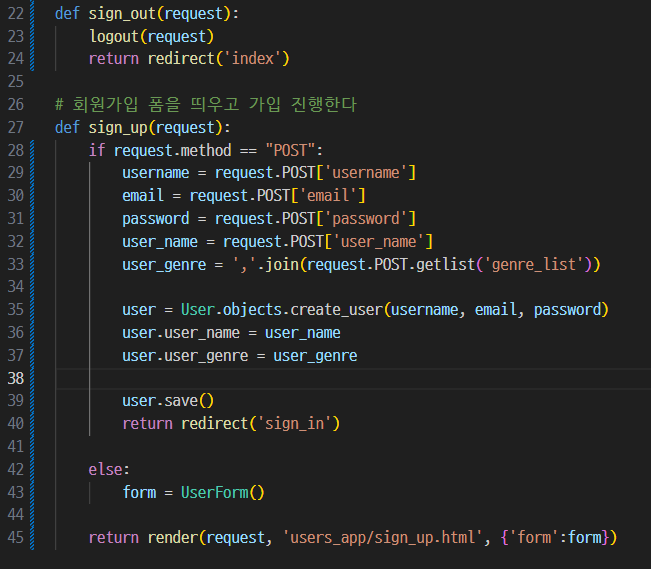
현재 회원가입 페이지



Forms.py



views.py



join 써서 getlist 함수 쓰면 받아온다.

**논의 내용 : 선호음악 페이지가 하는 일**

선호음악은 사용자 로그인 후에 나타나는 곳으로, 사용자 맞춤 음악(track)을 띄워주는 곳이다. Table은 Liked\_Track이며 구조는 Play\_list와 동일하나 저장되는 곡의 종류가 다르다.

Play\_list는 단순히 유저가 리스트에 곡을 올리는 것이지만, Liked\_Track은 유저가 별도로 선호하는 곡의 track\_id만 기록해놓는 공간이다.

로그인 한 유저가 선호음악 페이지를 누르면 spotipy API의 recommendations()에 audio\_feature 값과 seed\_genre 값을 넣어서 돌아온 결과를 DB에 한번 저장했다가 띄워준다.

users\_app\_user를 이용해 맞춤 음악을 제공하려면 user\_genre 값을 가져와 seed\_genre에 넣고, 음악의 audio\_feature 값도 넣어야 검색할 수 있을 것이다.

recommendations(seed\_genre=[’jazz’])에 넣을 수 있는 장르는 하나 뿐이다. 검색을 돌렸을 때 돌아오는 애들은 장르값이 따로 없는 track이다. 해당 track의 id를 audio\_feature(‘track\_id’)에 넣고, if 조건문을 통해 선별된 track들만 선호음악 페이지에 띄운다.

해당 음악의 기본 정보(곡명, 아티스트명, 앨범명, 앨범이미지) 뿐만 아니라, 플레이리스트 추가 버튼, 따봉 버튼도 있어야 한다.

상세페이지는 앨범에 대한 정보를 보여준다.