PROJECT REPORT

Project : Music Streaming App

Author:

Name: Debadrita Ghosh

Roll no : 21f1004258

Student email : 21f1004258@ds.study.iitm.ac.in

MSc Physics 1st year student

Description :

A music streaming app to enable users to stream music, create playlists, give song review, register as creator. Creators can add or delete songs and albums. Admins can block or unblock users in addition to the functions performed by creators.

Technologies Used:

blinker==1.6.2
click==8.1.6
colorama==0.4.6
contourpy==1.2.0
cycler==0.12.1
Flask==2.3.2
Flask-Login==0.6.3
Flask-SQLAlchemy==3.1.1
fonttools==4.44.0
greenlet==3.0.1
itsdangerous==2.1.2
Jinja2==3.1.2
kiwisolver==1.4.5
MarkupSafe==2.1.3
matplotlib==3.8.1
numpy==1.26.1
packaging==23.2
Pillow==10.1.0
pyparsing==3.1.1
python-dateutil==2.8.2
six==1.16.0
SQLAlchemy==2.0.23
typing_extensions==4.8.0
Werkzeug==2.3.6

DB Schema Design:

```
class User(db.Model):
    user_id = db.Column(db.Integer, primary_key=True)
    user_name=db.Column(db.String(100),unique=True , nullable = False)
    user_password = db.Column(db.String(100))
    user_status = db.Column(db.String)
    revs = db.relationship("Review", backref = "user")
    user_playlists = db.relationship("Playlist", backref = "user")
```

```
class Song(db.Model):
    song_id = db.Column(db.Integer, primary_key=True)
    song_title= db.Column(db.String , nullable = False)
    song_artist = db.Column(db.String,db.ForeignKey("album.song_artist"))
    song_avg_review = db.Column(db.Float)
    s_revs= db.relationship("Review", backref = "song")
    song_path=db.Column(db.Text)
    song_del_path=db.Column(db.Text)
    song_datetime=db.Column(db.Text)
    song_lyrics=db.Column(db.String)
    total_point=db.Column(db.Integer)
    total_revs=db.Column(db.Integer)

    def __repr__(self):
        return f'<Song "{self.song_title}">'

association = db.Table('association',
                db.Column('playlist_id', db.Integer, db.ForeignKey('playlist.playlist_id'),primary_key=True),
                db.Column('song_id', db.Integer, db.ForeignKey('song.song_id'),primary_key=True)
                )

class Playlist(db.Model):
    playlist_id = db.Column(db.Integer, primary_key=True)
    playlist_name = db.Column(db.String)
    user_id = db.Column(db.Integer,db.ForeignKey("user.user_id"))
    members = db.relationship("Song", secondary = "association",backref = "playlist")

    def __repr__(self):
        return f'<Playlist "{self.playlist_name}">'

class Review(db.Model):
    review_id = db.Column(db.Integer,primary_key=True)
    song_id = db.Column(db.Integer,db.ForeignKey("song.song_id"))
    user_id = db.Column(db.Integer,db.ForeignKey("user.user_id"))
    review = db.Column(db.Integer)

class Album(db.Model):
    album_id = db.Column(db.Integer, primary_key=True)
    album_genere= db.Column(db.String)
    song_artist = db.Column(db.String)
    works = db.relationship("Song",backref = "album")
```

Architecture and Features:

The ‘Project’ folder contains website folder containing the codes of the app ,’requirements.txt’, and ‘main.py’ ,which is run from the terminal to run the app. Inside the website folder there is the ‘__pycache ’ folder containing the caches, the ‘static’ folder containing the image and styling ‘.css’ files .The song files are also uploaded to the static folder. The ‘__init__’.py is the initiator file , ‘auth.py’ and ‘views.py’ files contains basic functions and routes for actions like login , signup, etc. ‘Models.py ’ contains the DB models.

Video:

https://drive.google.com/file/d/1Zh1ihR13cD40l6hYCSscnl6yz8mzHleG/view?usp=sharing