MUSIQ(MUSIC INTERFACE FOR QUERIES)

-15CS10008 Ayush Sharma

-15CS10048 Ujjawal Kumar

-15CS10049 Vaibhav Gupta

The web application proposed allows a user to search for information using different search criteria for music. The different search criteria include

- Artist
- Albums
- Top rated playlist
- New songs
- Tracks
- Genre
- User playlist
- Top charts

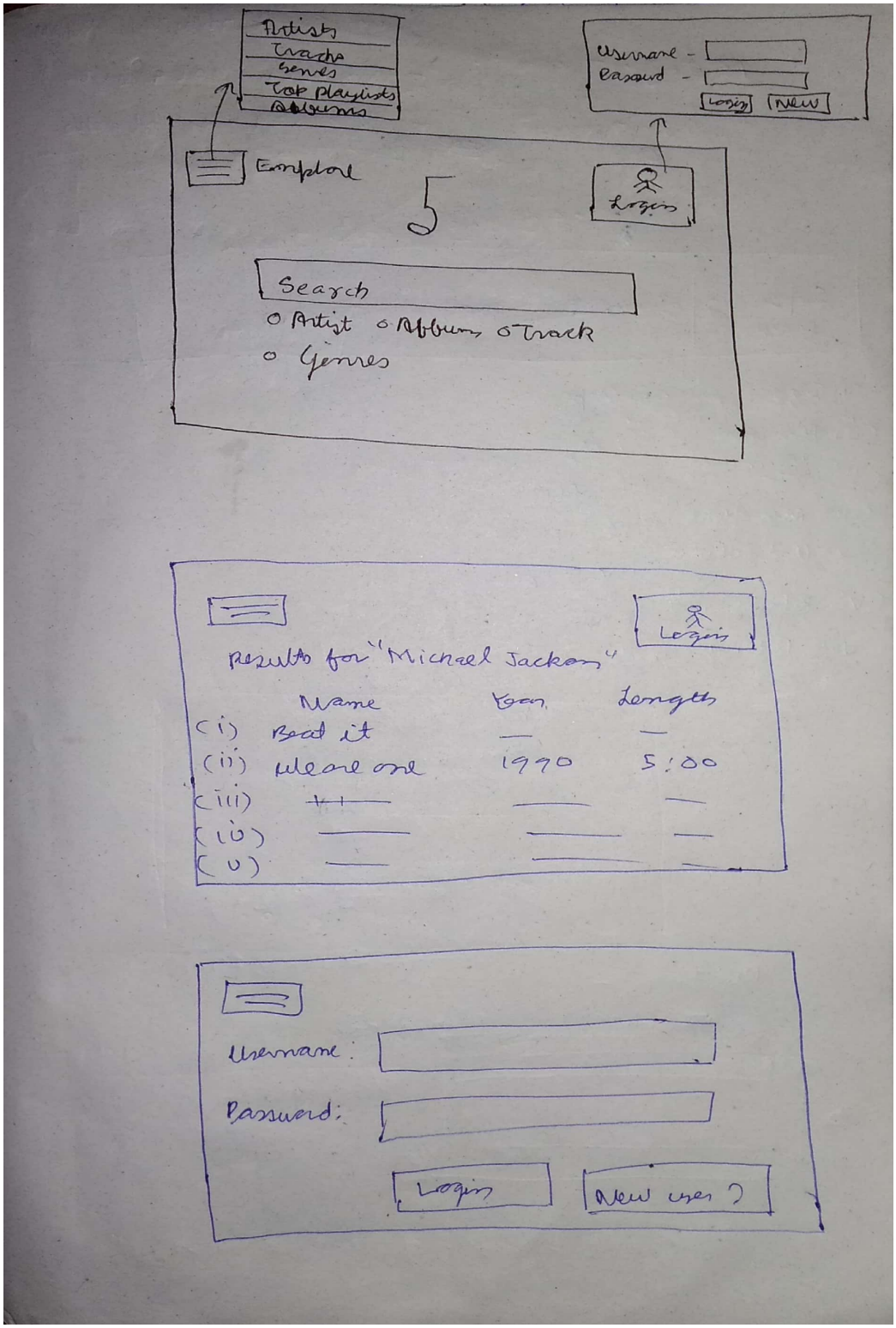
✓ A user has option to create an account after which he can create playlists which can be reviewed by other users.

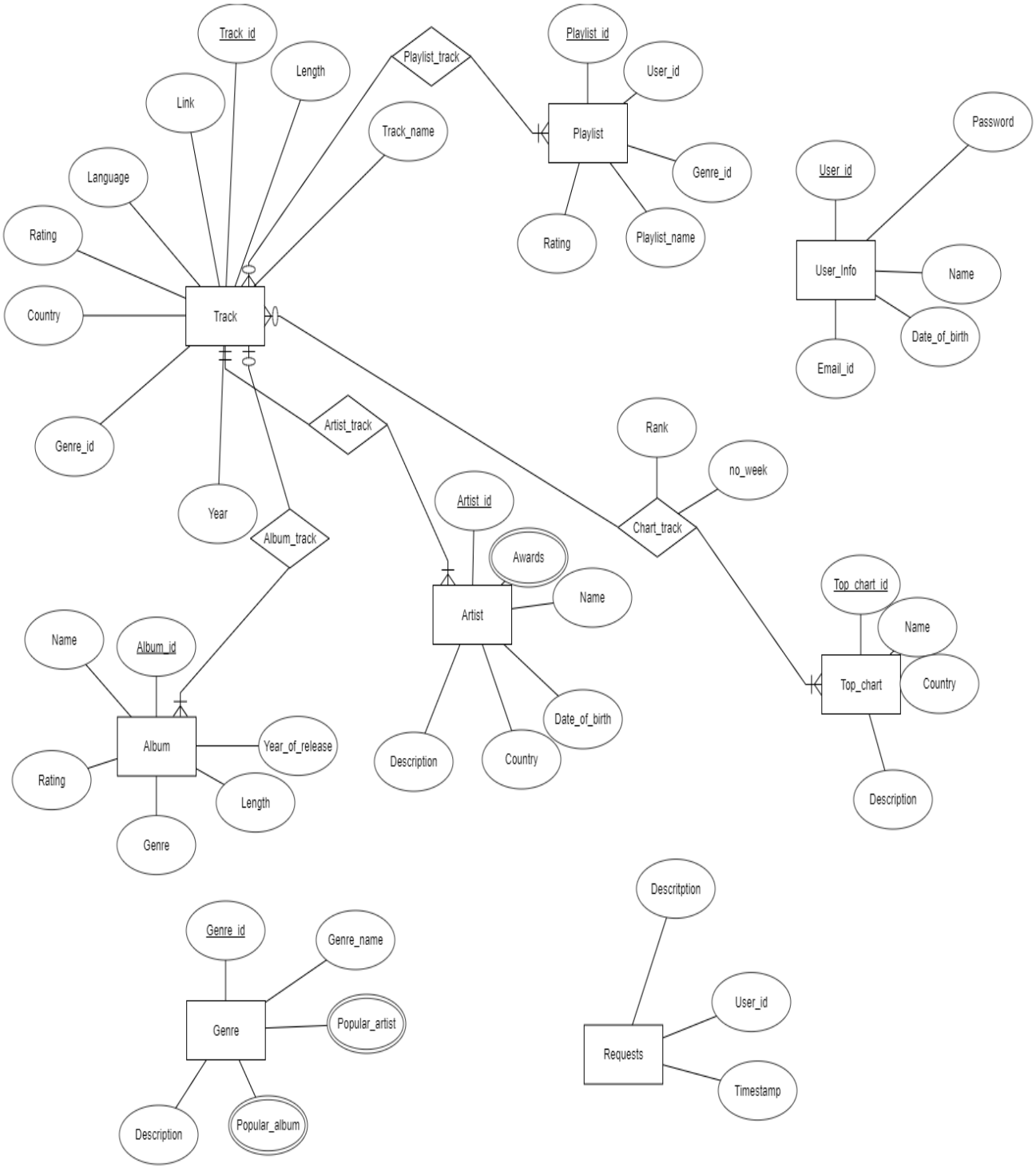
✓ A user has more option to create a request regarding information not available currently.

✓ A user has option to become a contributor for the application.

Implementation:

Our website will have the interfaces as shown below.


Artists
Tracks
Genres
Top Playlists
Albums
Username -
Password -
Login
NEW
Explore
Login
Search
Artist
Album
Track
Genres
Login
Results for "Michael Jackson"
Name
Year
Length
(i) Beat it
(ii) Wanna one
1990
5:00
Login
Username:
Password:
Login
New user?


Track_id
Length
Link
Track_name
Language
Rating
Country
Track
Genre_id
Playlist_track
Playlist_id
User_id
Playlist
Genre_id
Rating
Playlist_name
Password
User_id
User_Info
Name
Date_of_birth
Email_id
Artist_track
Year
Album_track
Artist_id
Awards
Name
Artist
Description
Country
Date_of_birth
Rank
no_week
Chart_track
Top_chart_id
Name
Top_chart
Country
Description
Name
Album_id
Album
Year_of_release
Rating
Length
Genre
Genre_id
Genre_name
Genre
Popular_artist
Description
Popular_album
Descritption
User_id
Requests
Timestamp

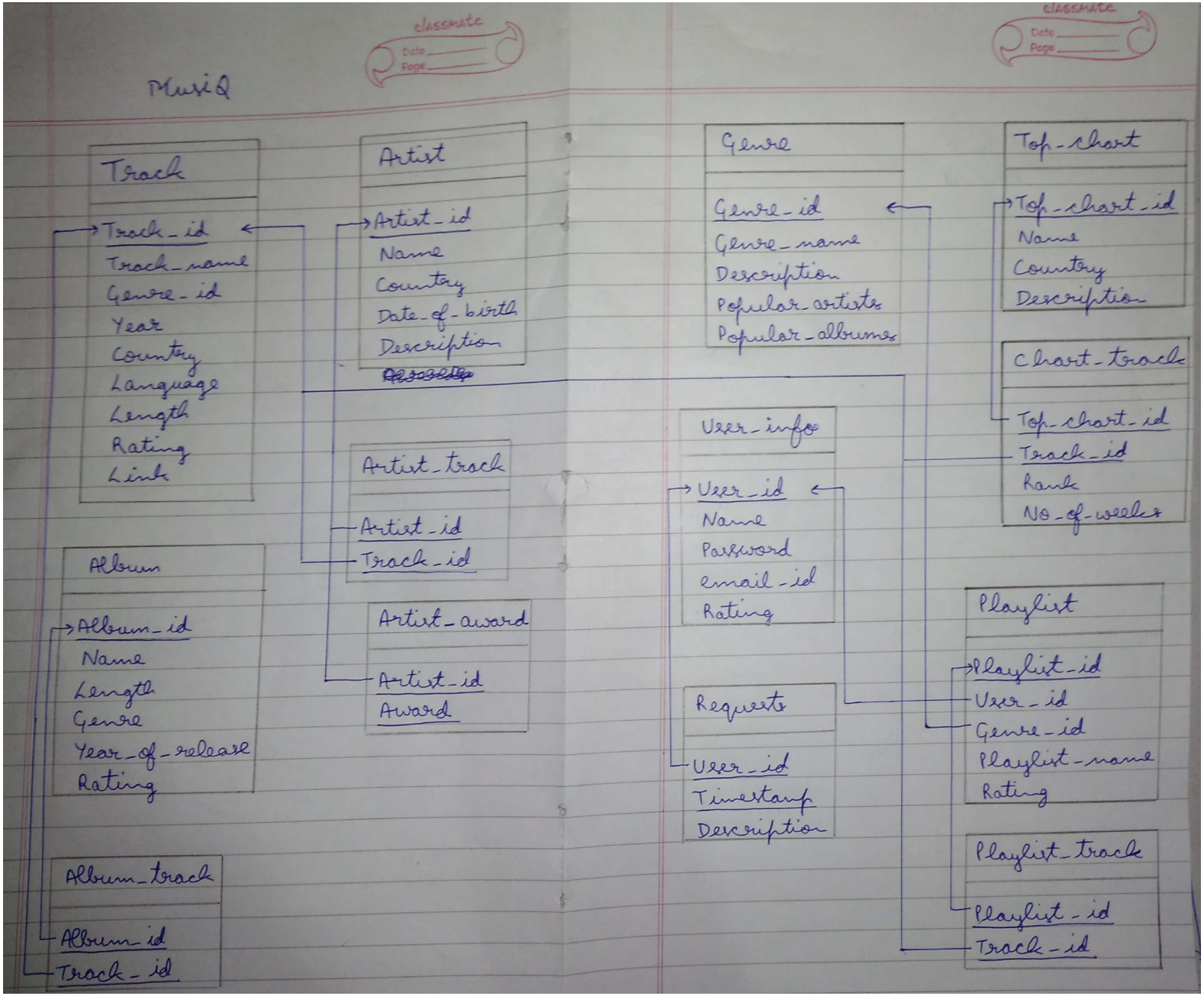
ER DIAGRAM

Database Information:

The considered relational schemas are as follows:

- ✓ Track: Information about the track
- ✓ Artist: Information regarding the artists
- ✓ Artist_track: The track composed by an artist.
- ✓ Album: Information about the albums
- ✓ Album track: details of a track in a particular album.
- ✓ Artist_award: awards received by an artist
- ✓ Genre: Description and popular artist in a genre
- ✓ Top_chart: Description about official top charts in different contries
- ✓ Chart track: Track listing, ranking of a top chart
- ✓ User info: account information regarding a user.
- ✓ Playlist: playlist description created by an user.
- ✓ Playlist track: track listing in a playlist
- ✓ Requests: the requests made by an user for an unavailable track

Schema Diagram



**Details regarding the framework used:-

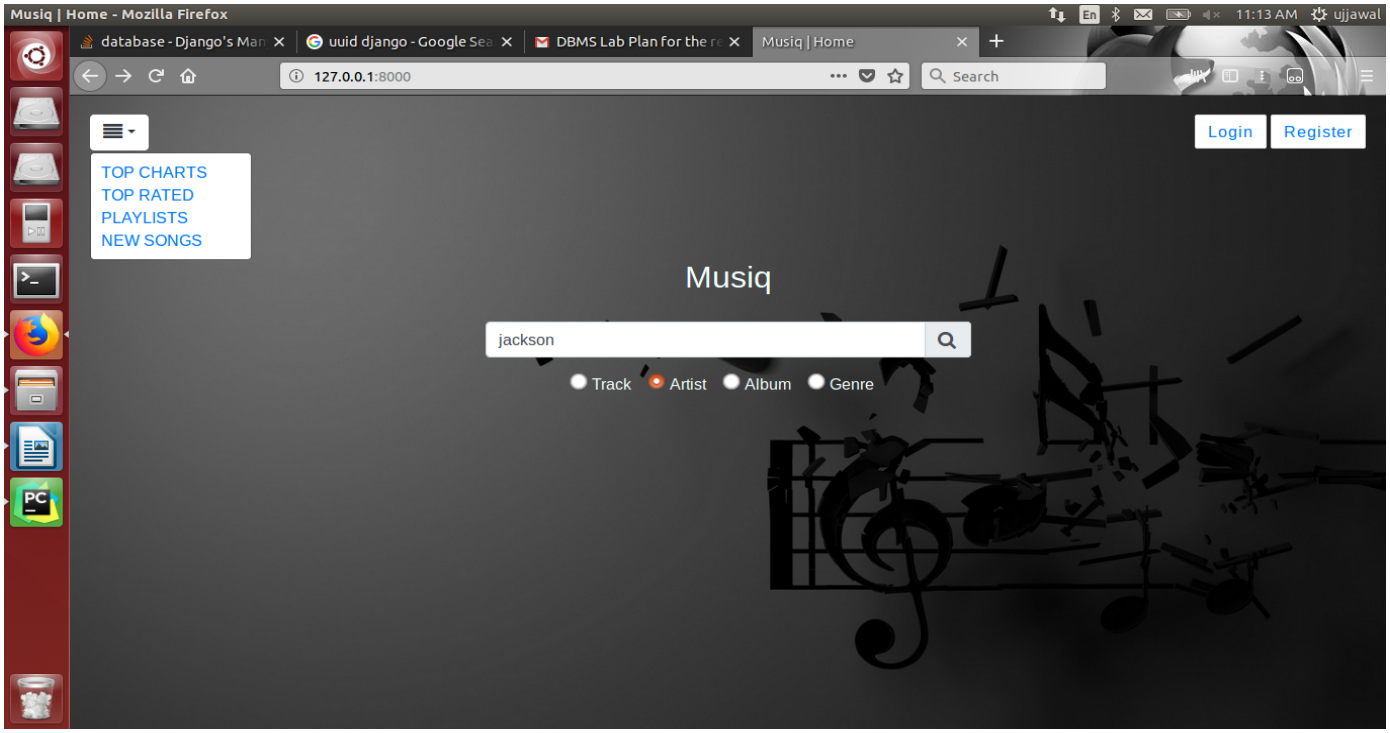
-->Django is used to handle the back end of the application, as it readily provides a unified framework for database handling, a built in admin interface as well as a configured server.

-->To take a modular approach the application is divided into two django apps-

i)Search-handles the search and display part .

ii)Login- handles user login,authentication ,account management.

-->At the home page there is a general search interface



**Accordingly the model based database handling (MVT architecture) modifies tables as follows:

(i) For Search app

Artist

- title - charfield
- country - charfield
- year - IntegerField
- description - TextField (optional)
- artist_logo - FileField (optional)

Track

- title - charfield
- genre - ForeignKey (Genre)
- Year - IntegerField
- Country - charfield
- Language - charfield
- Length - TimeField
- charts - ManyToMany (TopCharts, through = 'Membership')
- Rating - IntegerField
- playlist - ManyToMany (Playlist)
- Link - URLField
- artist - ForeignKey (Artist)

Album

- Title - charfield
- Genre - ForeignKey (Genre)
- length - ~~Integer~~ TimeField
- year - IntegerField
- rating - IntegerField
- Album_logo - FileField (optional)

Top Charts

- Name - charfield()
- Country - charfield()
- Description - TextField()

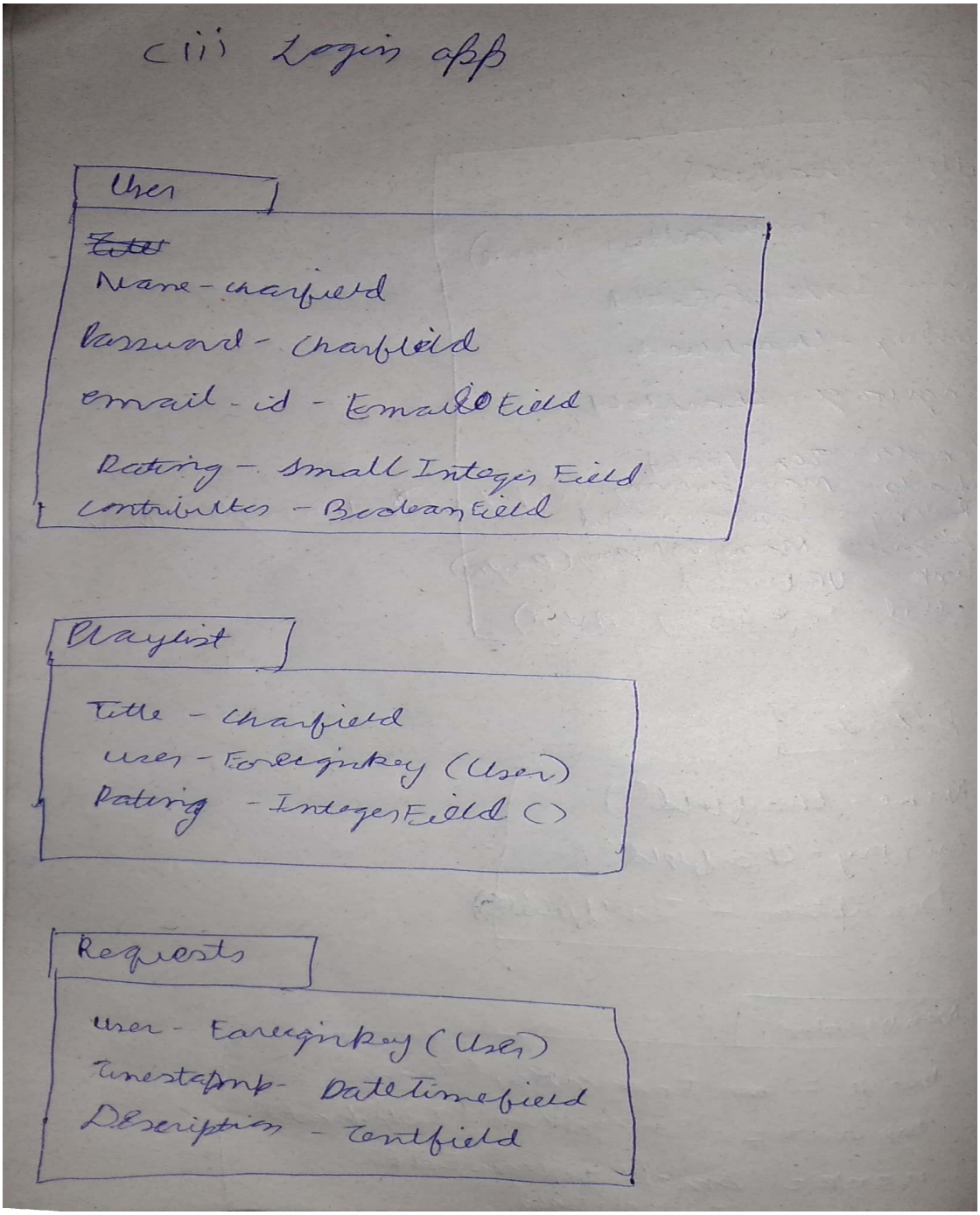
Genre

- Title - charfield
- Description - TextField()
- popular_artist - charfield
- Popular_album - charfield
- Genre_logo - FileField

Membership

- track = ForeignKey (Track)
- top chart = ForeignKey (TopChart)
- rank = SmallIntegerField
- weeks = IntegerField

i)Models for Search app



ii) Models for Login app