Name of Members: Seo Pallichirayil(sgp322)

Rahul Purushottam Gaonkar(rpg283)

Project Part 2

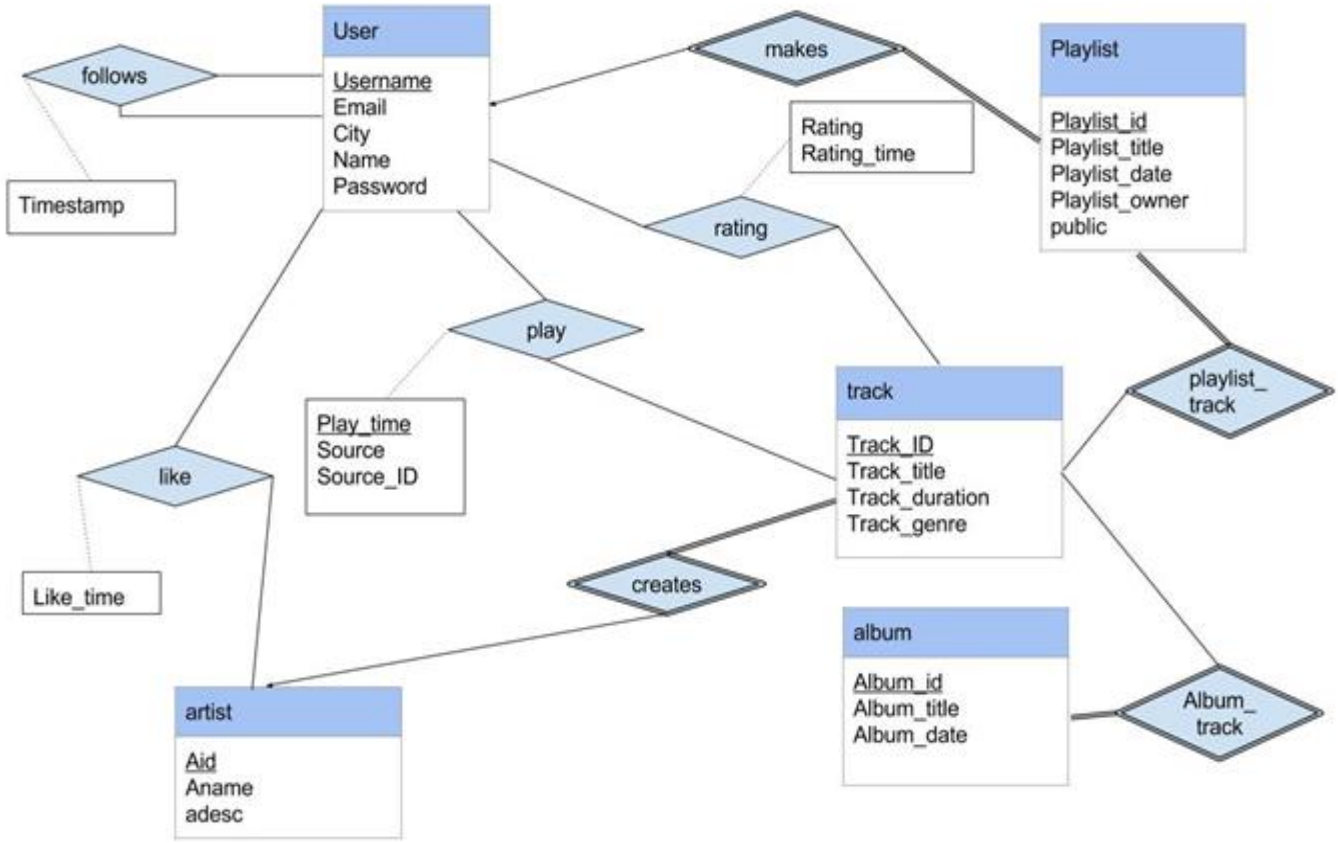
Project Description:

Musify is an online music streaming web application. It allows users to sign up, create a profile, log in, play tracks, make playlists, like an artist, rate songs, follow other users, and browse and search for music. A new user will first have to sign up by providing some basic details like their full name, city, email id, username and password. The username provided by the new user must be unique and cannot be same as an existing username in the database. The password provided by the user is converted into hash value and stored in the database for providing better security. Once a new user has signed up, he can login using the Username and Password into the system which will redirect him to the homepage of the web application. **The homepage of the web application will show recent events relevant to the user in different sections namely Listen Again Section: Showing the last played songs by the user, Recommended section which is divided into two more sections namely tracks: Displaying the latest tracks of the artist liked by the user and Playlist : Displaying the latest playlist by the users they follow, You May also like these Artist : Displays the artist related to the artist liked by the User depending on the number of common fans using Jaccard similarity.** The user can play and rate (any value between 0 and 5) each track displayed on the homepage. The number of times the track is played i.e. views and the average rating of the track calculated from the previous rating given to the track by different users is also displayed along with the tracks on the homepage. The number of times the track is played i.e. views and the average rating of the track can be 0 and 'NOT RATED' if the track is not played or rated even once. The user can play and rate the same song multiple times, the database will store all the play records and only the latest rating record given by the user in the database. When the user is trying to rate the song, a modal box pop up opens which allows the user to enter the new rating of the track and also displays the previous rating of the track. The user can also navigate to artist page of the artist who created a track displayed on the homepage to get an information about the other details of the artist like the artist description and the other tracks created by that artist. The user can also like or unlike the artist using the LIKE or UNLIKE button on this page and can also check the list of users who liked this artist. The user can also navigate to other user's page by clicking on any username in the list to get some more information about the User like name, city, email, the number of public playlist created by the user, the list of other artist liked by the user. The user can follow or unfollow other users by clicking on the follow or unfollow button on this page. The user can also check list of followers and following of the other user on this page. The user is also able to search by tracks, album or playlist on the homepage. Search by track will match the keyword to be searched with the track name or the artist name of the track. Search by Album or Playlist will match the keyword to be searched with Album title or Playlist title respectively. It will show 'NO RESULTS FOUND' in either of the three cases. The user can also navigate to the Album or Playlist page by clicking on the Album title or Playlist title found in the search result. The Album page will give more additional information about the Album like the tracks included in the album and will give an option to play or rate the track. The Playlist page will give more additional information about the Playlist like the creation date of the playlist, playlist owner and the tracks included in that playlist. If the user plays a track on the album or playlist page then the play record inserted into the database will also store that the track was played from an album/playlist thereby also storing the Album id/Playlist id of the Album/Playlist which can help to keep a track of which Album/Playlist does the user visits frequently. If the user rates the track from an Album/Playlist page it will simply store the rate record with the track_id and will not include Album id/Playlist id. On the homepage we have four main tabs on the menu bar namely Home, My Playlist, Profile and logout. The Home tab will navigate you to the homepage and the MyPlaylist tab will navigate you

to a page where the user can check the playlist created by him, can edit them by adding or deleting tracks or changing the Playlist Title, and can create new public/private playlist by searching and adding new tracks in the playlist. The profile tab will navigate you to a page displaying the profile of the user in which all fields of the User details section can be updated except the Username assigned to the User. The profile page will also show all the public/private playlist created by the user, list of artists liked by the user, list of following and followers of the user. The user session created after the User logins to the web application is destroyed by clicking on the logout tab of the menu bar. We have tried to prevent SQL injections by writing all the SQL queries in the stored procedures and then calling the stored procedures in the php script. **You can bookmark artist and User page.**

Data from Project Part 1:

ER diagram:



Relational Schema:

Album (Album_id, Album_title, Album_date)
Album_track (Album_id, Track_id)
Artist (Aid, Aname, Adesc)
follow (Username, Following_id, Timestamp)
like (Username, Aid, Like_time)

play (Username, Track_id, Play_time, Source, Source_ID)
playlist (Playlist_id, Playlist_title, Playlist_date, Playlist_owner, public)
playlist_track (Playlist_id, Track_id)
rating (Username, Track_id, Rating, Rating_time)
track (Track_id, Track_title, Track_duration, Track_genre, Track_aid)
User (Username, Email, City, Name, Password)

Foreign Keys:

Album_track.Album_id references Album.Album_id
Album_track.Track_id references track.Track_id
follow.Username references User.Username
follow.Following_id references User.Username
like.Username references User.Username
like.Aid references Artist.aid
play.Username references User.Username
play.Track_id references track.Track_id
playlist.Playlist_owner references User.Username
playlist_track.Playlist_id references playlist.Playlist_id
playlist_track.Track_id references track.Track_id
rating.Username references User.Username
rating.Track_id references track.Track_id
track.Track_aid references Artist.Aid

Assumptions:

1. Each playlist and Album should contain atleast one track.
2. A user can't follow himself.

Design Decisions:

1. The 'public' attribute of 'playlist' relation will be a Boolean value which will be set depending on if the playlist is public or private.
2. The 'Source' attribute of the 'Play' relation will have three values namely 'Album', 'Playlist' and 'Null' depending on whether the track being played is a part of any Album, Playlist or is an individual track respectively.
3. The 'Source_ID' attribute of 'Play' relation will have values from the 'Album_id' or 'Playlist_id' attribute of Album or Playlist relation respectively if track being played is a part of any Album or Playlist and the value of Source_ID would be null if it is an individual track.
4. The Album relation is normalized and divided into two relations Album and Album_Track to reduce redundancy as a single Album can have multiple tracks and the Album_id would

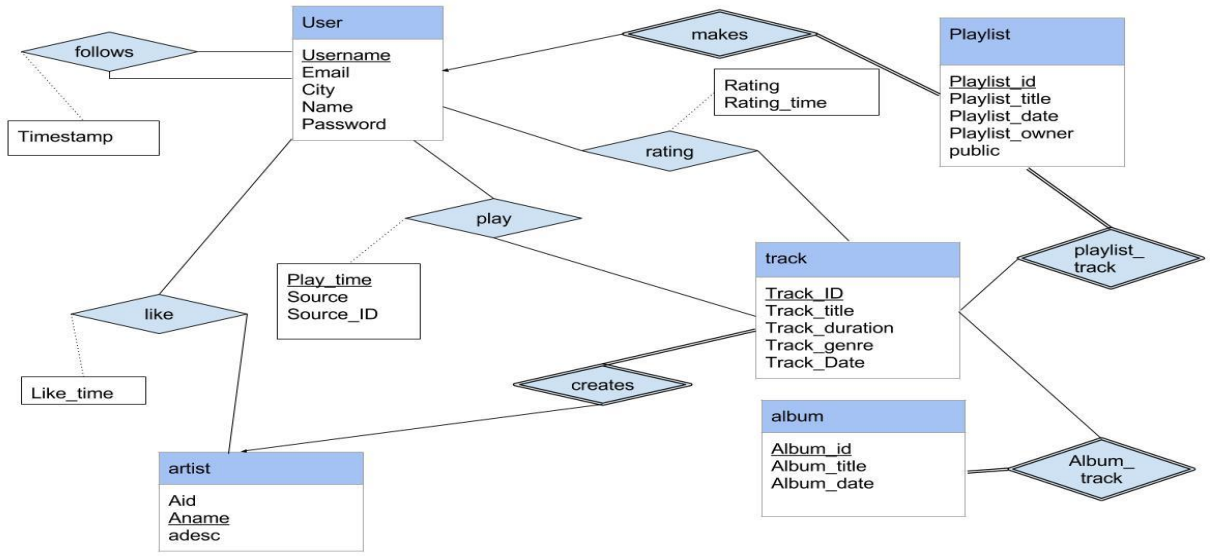
be repeated for each record and it would also violate the primary key(Album_id) constraint of the Album relation if not normalized.

5. The playlist relation is normalized and divided into two relations playlist and playlist_track to reduce redundancy as a single playlist can have multiple tracks and the Playlist_id would be repeated for each record and it would also violate the primary key(Playlist_id) constraint of the playlist relation if not normalized.
6. A User can play same track multiple times, so we define (Username, Track_id, Play_time) as the primary key of the Play table.
7. We only store one record for the rating given by a user to a particular product in the rating relation, if the user changes his rating for that product in future then we update the old record. The rating can be any value between 0 and 5.
8. The record is inserted in the follow and like relation if a user follows another user or likes an artist respectively and a record is deleted in the follow and like relation if a user unfollows another user or unlike an artist respectively.
9. The hash value of the password will be stored in the 'password' attribute of the User relation for every User record. (This will be implemented in the second part of the project)

Modification of Data in Project Part 2: (Based on the spotify dataset given)

Data marked in green is the modified data in Project Part2

Modified ER diagram:



Modified Relational Schema:

Album (Album_id, Album_title, Album_date)
Explanation: The Album table stores Album_id, Name of the Album(Album_title) and the Album creation date (Album_date)

Album_track (Album_id, Track_id)
Explanation: The Album track table stores the Album_id and the Track_id of the track included in it.

Artist (Aid, Aname, Adesc) //Change made based on the spotify dataset provided but making aname a primary key is not a good practice as two artist can have same aname
Explanation: The Artist table stores the Artist id (Aid), Name of the Artist(Aname) and description of the Artist(Adesc)

follow (Username, Following_id, Timestamp)
Explanation: The follow table stores the Username of the user(Username), the Username of the user he is following(following_id) and the time when he started following(timestamp)

like (Username, Aname, Like_time) //Change based on the dataset provided
Explanation: The like table stores the Username of the user(Username), the name of the artist he likes(Aname) and the time when he liked the artist (Like_time)
play (Username, Track_id, Play_time, Source, Source_ID)
Explanation : The Play table stores the username of the user who played the track(Username), the track_id of the track being played (Track_id),the time when the track was played (Play_time),the source from where the track was played(Source) i.e Album, Playlist or ‘ ’(incase directly played from tracks) and the (Source _id) i.e. Album_ID,Playlist_ID or ‘ ’(incase directly played from tracks)

playlist (Playlist_id, Playlist_title, Playlist_date, Playlist_owner, public)
Explanation : The Playlist table stores the Playlist_id, name of the Playlist (Playlist_title),creation date of the Playlist(Playlist_date),owner of the Playlist (Playlist_owner),if the playlist is public or private(public)

playlist_track (Playlist_id, Track_id)
Explanation: The Playlist track table stores the Playlist_id and the Track_id of the track included in it.

rating (Username, Track_id, Rating, Rating_time)
Explanation: The rating table stores the Username of the User giving the rating(Username), Track_id of the track being rated(Track_id), the value of the rating(Rating), the time when the rating is given (Rating time)
track (Track_id, Track_title, Track_duration, Track_genre, Track_aname, Track_Date) //Change based on the dataset provided
Explanation: The track table stores the Track_id of the track(Track_ID), name of the

track(Track_title), the duration of the track (Track_duration),genre of the track (Track_genre),Name of the Artist who created the track (Track_aname),Track creation date (Track_Date)
User (Username, Email, City, Name, Password)
Explanation: The User table stores the Username of the user (Username), Email id of the user (Email),the city the user lives in (city),Name of the user (Name) and the hash value of the input password (Password)

Modified Foreign Keys:

Album_track.Album_id references Album.Album_id
Album_track.Track_id references track.Track_id
follow.Username references User.Username
follow.Following_id references User.Username
like.Username references User.Username
like.Aname references Artist.aname //Change based on the dataset provided
play.Username references User.Username
play.Track_id references track.Track_id
playlist.Playlist_owner references User.Username
playlist_track.Playlist_id references playlist.Playlist_id
playlist_track.Track_id references track.Track_id
rating.Username references User.Username
rating.Track_id references track.Track_id
track.Track_aname references Artist.aname //Change based on the dataset provided

Modified Assumptions & Design Constraints:

Assumptions:

1. Each playlist and Album should contain atleast one track.
2. A user can't follow himself.

Design Decisions:

1. The 'public' attribute of 'playlist' relation will be a Boolean value which will be set depending on if the playlist is public or private.
2. The 'Source' attribute of the 'Play' relation will have three values namely 'Album', 'Playlist' and 'Null' depending on whether the track being played is a part of any Album, Playlist or is an individual track respectively.
3. The 'Source_ID' attribute of 'Play' relation will have values from the 'Album_id' or 'Playlist_id' attribute of Album or Playlist relation respectively if track being played is a part of any Album or Playlist and the value of Source_ID would be null if it is an individual track.

4. The Album relation is normalized and divided into two relations Album and Album_Track to reduce redundancy as a single Album can have multiple tracks and the Album_id would be repeated for each record and it would also violate the primary key(Album_id) constraint of the Album relation if not normalized.

5. The playlist relation is normalized and divided into two relations playlist and playlist_track to reduce redundancy as a single playlist can have multiple tracks and the Playlist_id would be repeated for each record and it would also violate the primary key(Playlist_id) constraint of the playlist relation if not normalized.

6. A User can play same track multiple times, so we define (Username, Track_id, Play_time) as the primary key of the Play table.

7. We only store one record for the rating given by a user to a particular product in the rating relation, if the user changes his rating for that product in future then we update the old record. The rating can be any value between 0 and 5.

8. The record is inserted in the follow and like relation if a user follows another user or likes an artist respectively and a record is deleted in the follow and like relation if a user unfollows another user or unlike an artist respectively.

9. The hash value of the password will be stored in the 'password' attribute of the User relation for every User record using the SHA2(Password input,256) function.

10. The login functionality procedure will also use SHA2(Password input,256) function to convert the password to its hash value to compare it with the hash values stored in the database

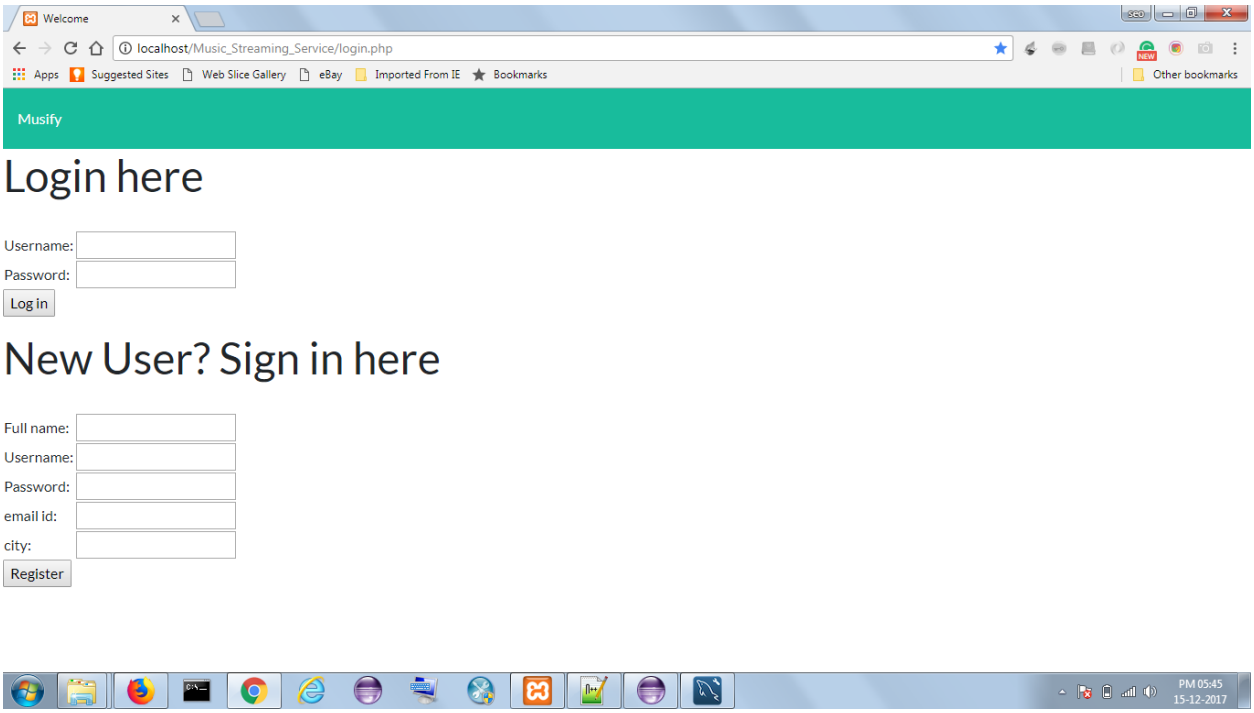
11. Username once assigned to the user during the User sign up process cannot be updated while updating the User Profile Page.

12. If the user clicks on the play button it opens a modal box pop up with the spotify widget in it. Due to issues in integration of the spotify widget with our web application, we insert a record in the play table after the user clicks the Play button on the user interface which opens the modal box with spotify widget that plays the song.

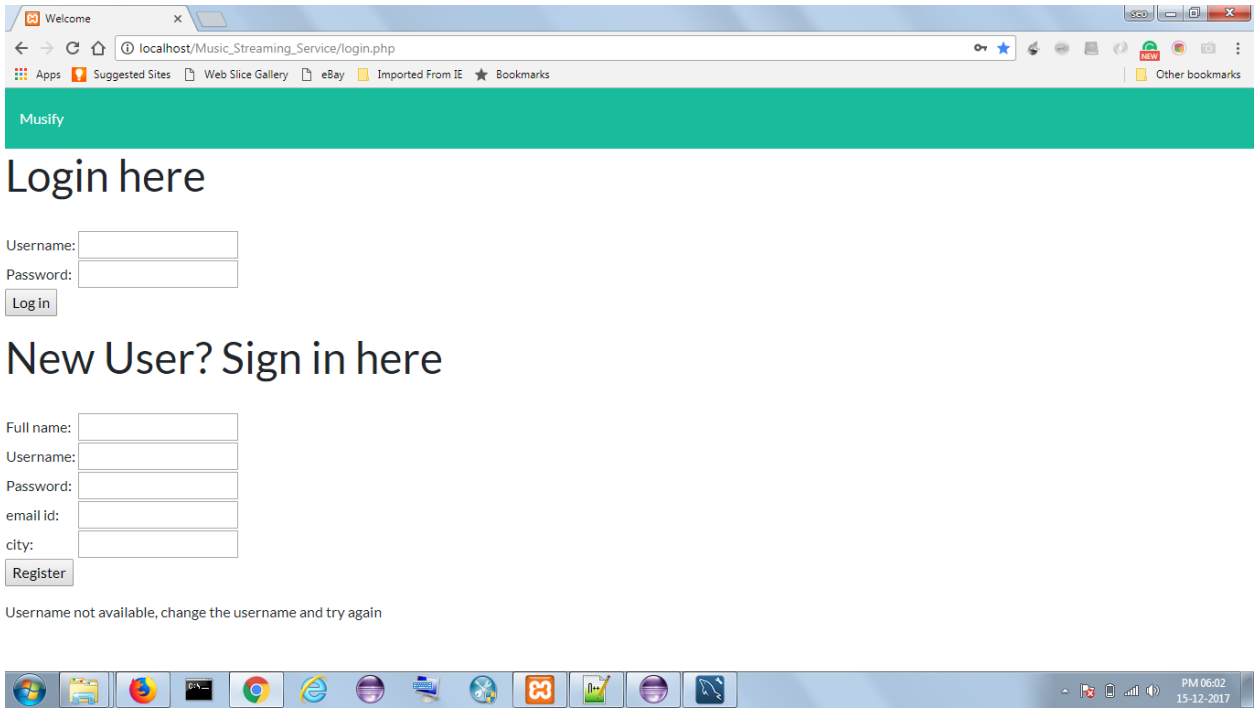
13. The number of times the track is played i.e. views and the average rating of the track can be 0 and 'NOT RATED' if the track is not played or rated even once.

Detailed Description of the Musify:

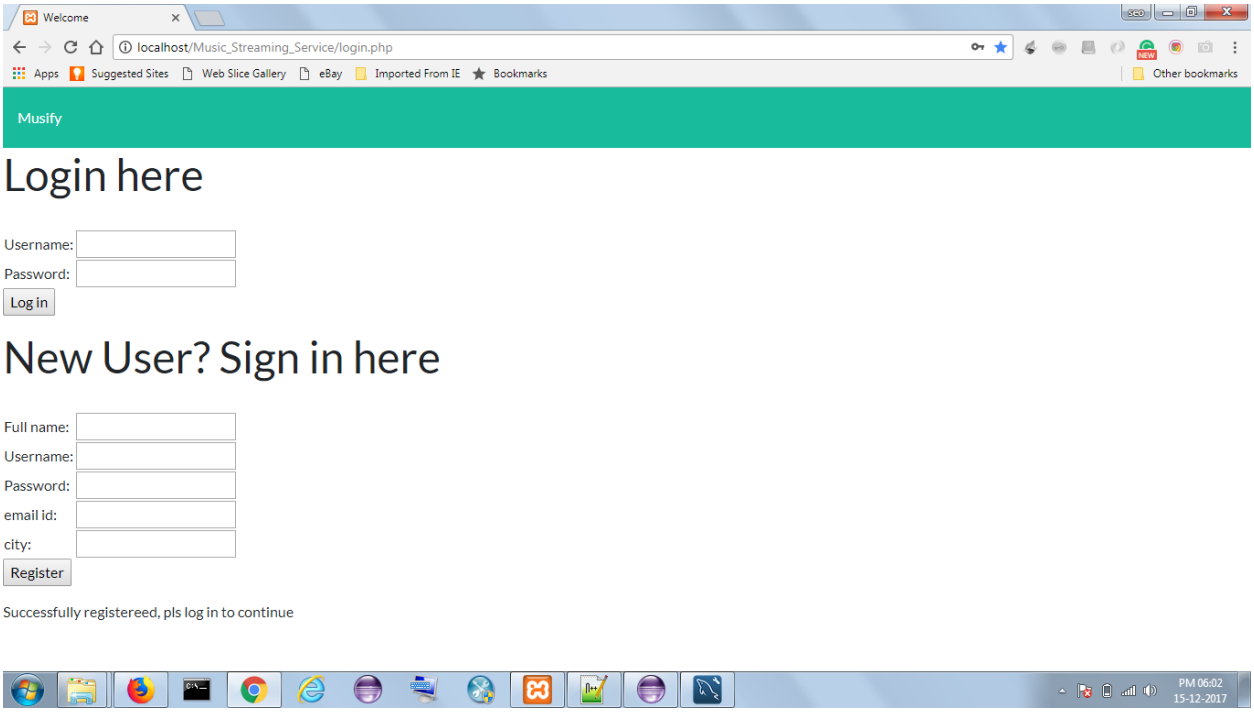
Login Page/Sign in Page:



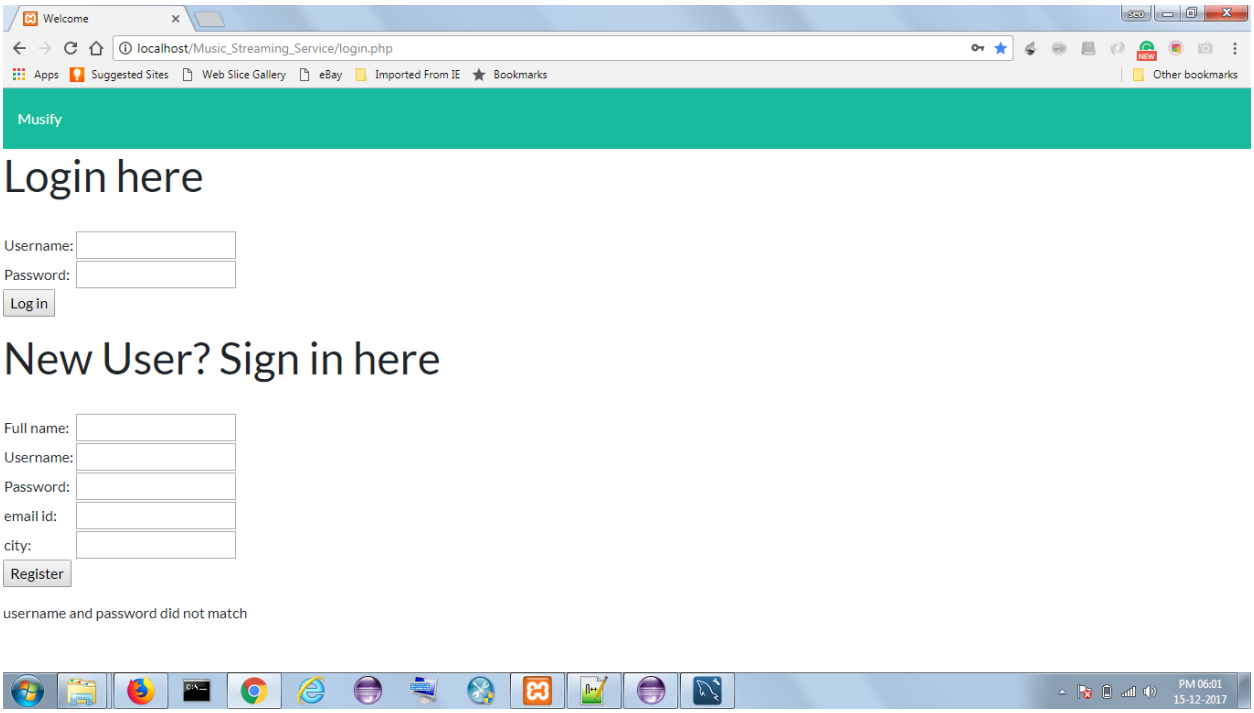
Sign in fail Message:



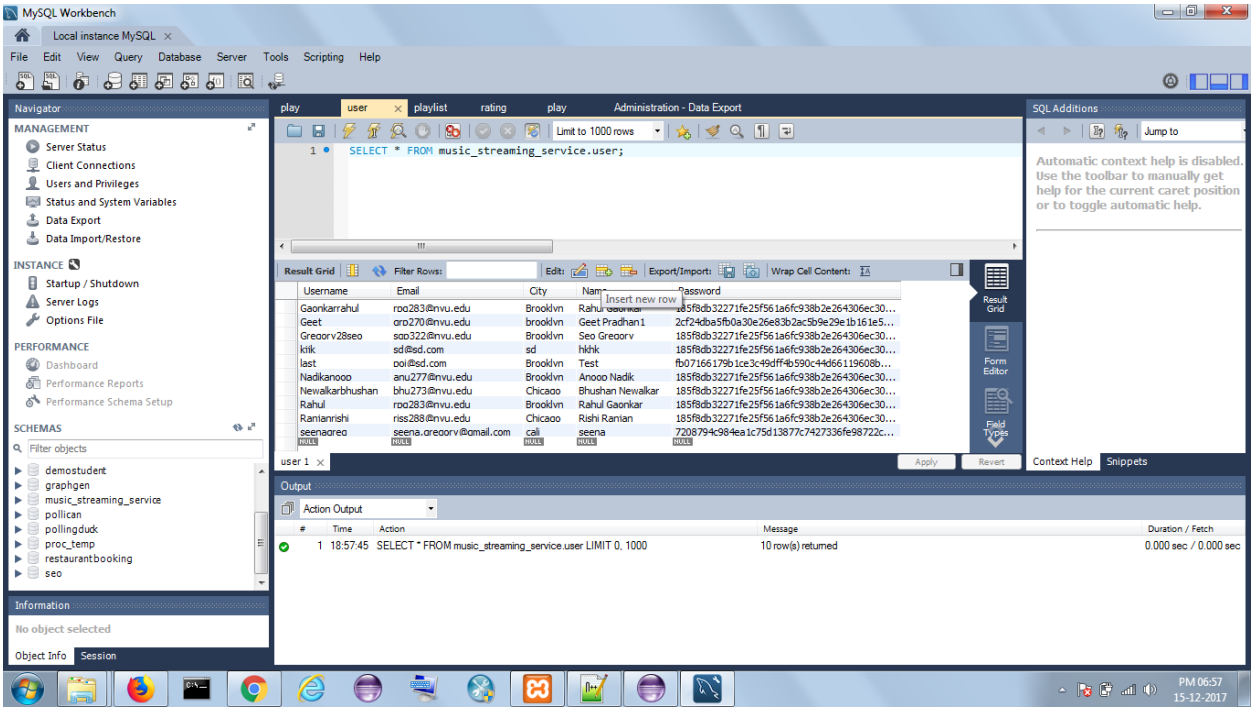
Sign in Successful Message:



Log in Fail Message:



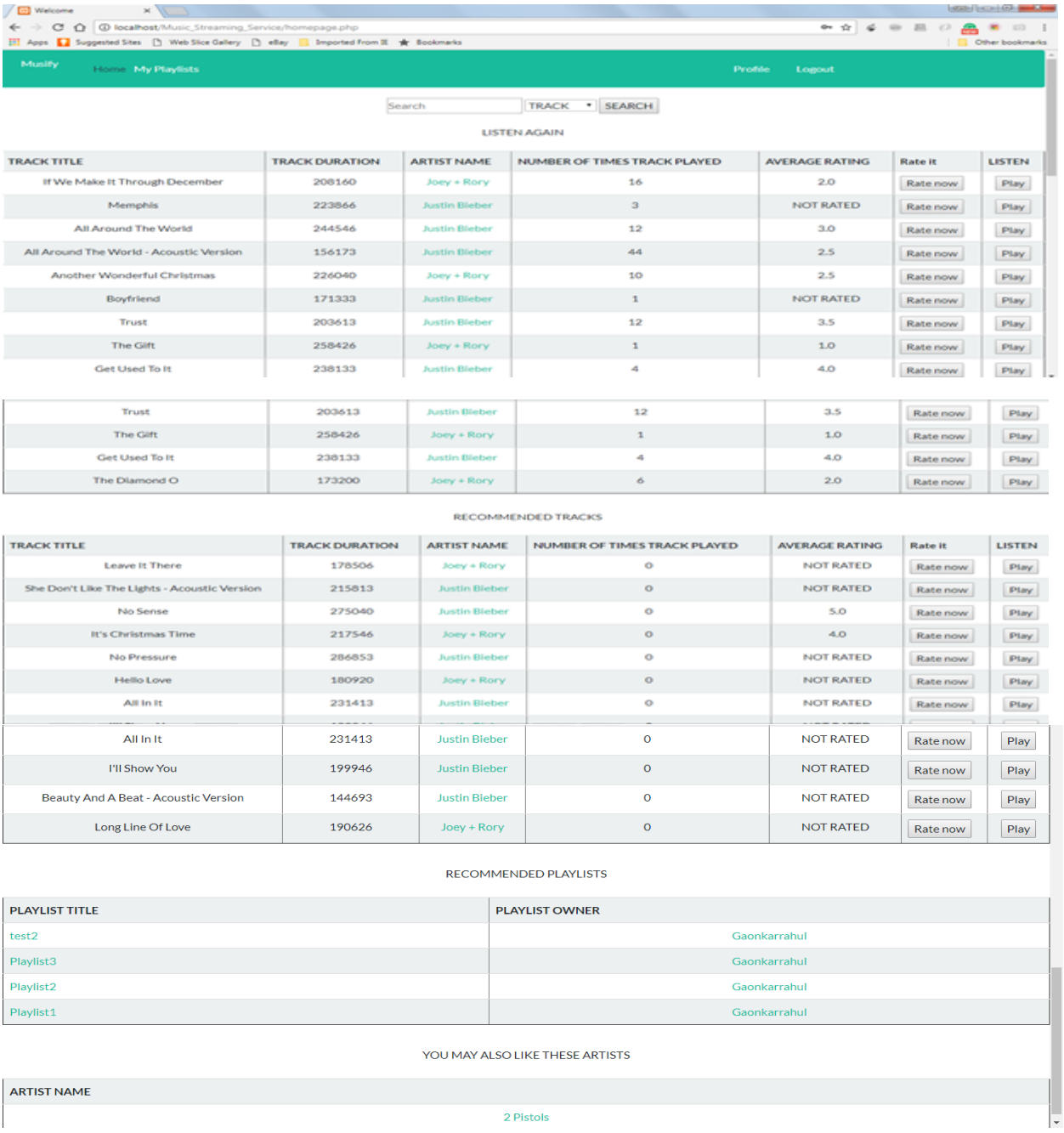
Hashed values of Password stored in Password attribute in the Database Table User:



New user should first sign in by filling out some basic personal details like full name, city, email id, username and password. All the fields are mandatory fields and proper validations are present to ensure all the details entered are accurate. Here, the input is passed to SQL procedure that creates the User record in the database thereby preventing SQL injection. If user with same username is already present, then system will not allow the new user to register successfully. The hash value of the input password is stored in the database while creating a user record during sign in process thereby providing security. Once the user has signed in, he can login into the system using the Username and password. The login functionality is handled by a stored procedure which converts the input password into hash value and the (Username, hash value of the Password) pair is compared with the records in the database system and if a match is found the user is logged in to the system and redirected to the homepage and a user session is created.

Homepage and Different Tabs on the Homepage:

Home Tab:



Musify Home My Playlists Profile Logout

Search TRACK SEARCH

LISTEN AGAIN

TRACK TITLE	TRACK DURATION	ARTIST NAME	NUMBER OF TIMES TRACK PLAYED	AVERAGE RATING	Rate it	LISTEN
If We Make It Through December	208160	Joey + Rory	16	2.0	Rate now	Play
Memphis	223866	Justin Bieber	3	NOT RATED	Rate now	Play
All Around The World	244546	Justin Bieber	12	3.0	Rate now	Play
All Around The World - Acoustic Version	156173	Justin Bieber	44	2.5	Rate now	Play
Another Wonderful Christmas	226040	Joey + Rory	10	2.5	Rate now	Play
Boyfriend	171333	Justin Bieber	1	NOT RATED	Rate now	Play
Trust	203613	Justin Bieber	12	3.5	Rate now	Play
The Gift	258426	Joey + Rory	1	1.0	Rate now	Play
Get Used To It	238133	Justin Bieber	4	4.0	Rate now	Play
Trust	203613	Justin Bieber	12	3.5	Rate now	Play
The Gift	258426	Joey + Rory	1	1.0	Rate now	Play
Get Used To It	238133	Justin Bieber	4	4.0	Rate now	Play
The Diamond O	173200	Joey + Rory	6	2.0	Rate now	Play

RECOMMENDED TRACKS

TRACK TITLE	TRACK DURATION	ARTIST NAME	NUMBER OF TIMES TRACK PLAYED	AVERAGE RATING	Rate it	LISTEN
Leave It There	178506	Joey + Rory	0	NOT RATED	Rate now	Play
She Don't Like The Lights - Acoustic Version	215813	Justin Bieber	0	NOT RATED	Rate now	Play
No Sense	275040	Justin Bieber	0	5.0	Rate now	Play
It's Christmas Time	217546	Joey + Rory	0	4.0	Rate now	Play
No Pressure	286853	Justin Bieber	0	NOT RATED	Rate now	Play
Hello Love	180920	Joey + Rory	0	NOT RATED	Rate now	Play
All In It	231413	Justin Bieber	0	NOT RATED	Rate now	Play
All In It	231413	Justin Bieber	0	NOT RATED	Rate now	Play
I'll Show You	199946	Justin Bieber	0	NOT RATED	Rate now	Play
Beauty And A Beat - Acoustic Version	144693	Justin Bieber	0	NOT RATED	Rate now	Play
Long Line Of Love	190626	Joey + Rory	0	NOT RATED	Rate now	Play

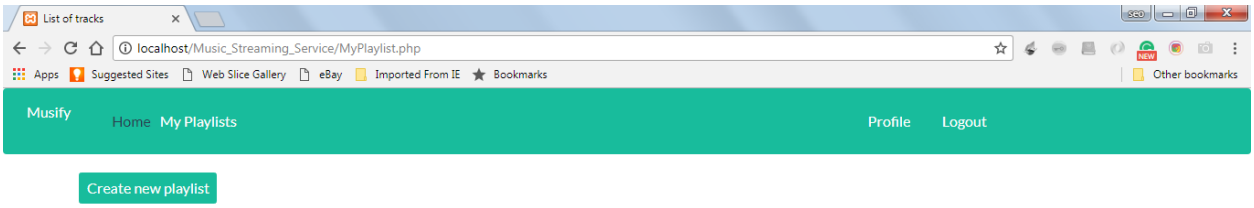
RECOMMENDED PLAYLISTS

PLAYLIST TITLE	PLAYLIST OWNER
test2	Gaonkarrahul
Playlist3	Gaonkarrahul
Playlist2	Gaonkarrahul
Playlist1	Gaonkarrahul

YOU MAY ALSO LIKE THESE ARTISTS

ARTIST NAME
2 Pistols

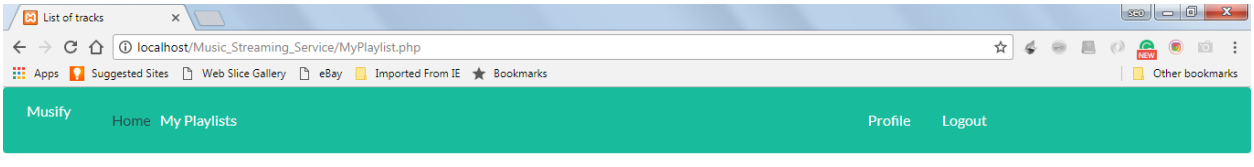
My Playlists Tab (General View):



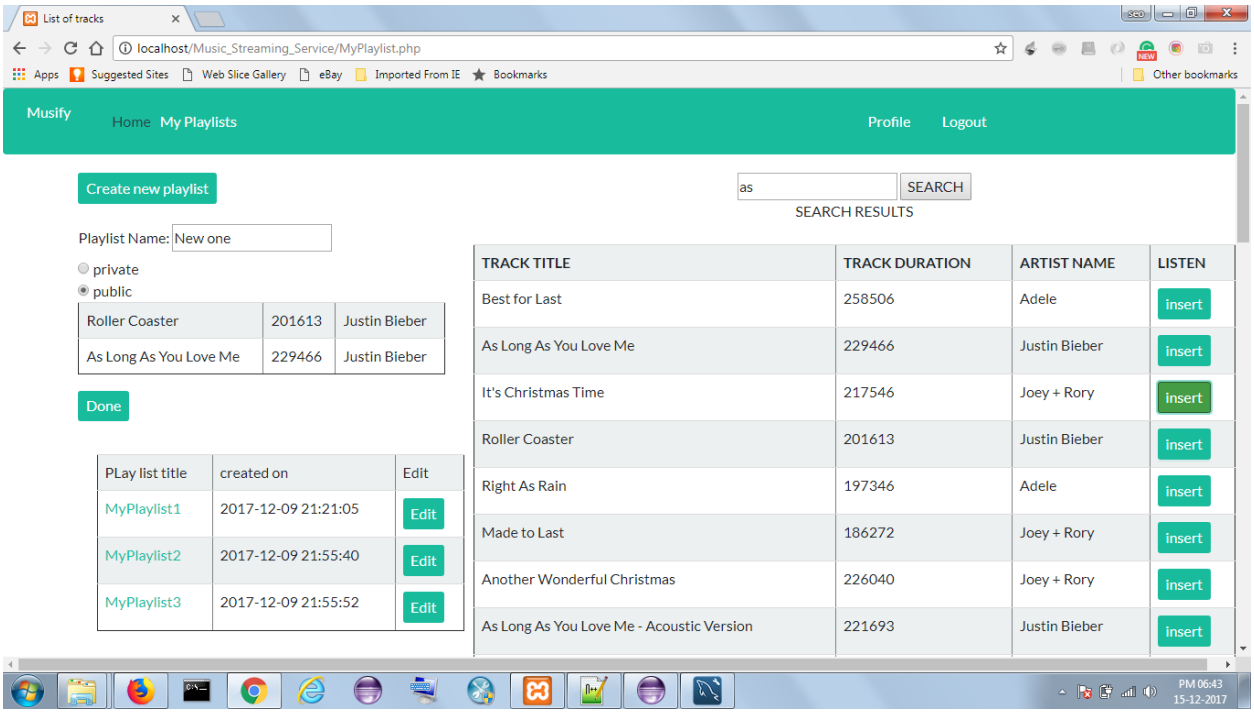
PLay list title	created on	Edit
MyPlaylist1	2017-12-09 21:21:05	Edit
MyPlaylist2	2017-12-09 21:55:40	Edit
MyPlaylist3	2017-12-09 21:55:52	Edit

MyPlaylist1, MyPlaylist2, MyPlaylist3 are the playlist created by the user.

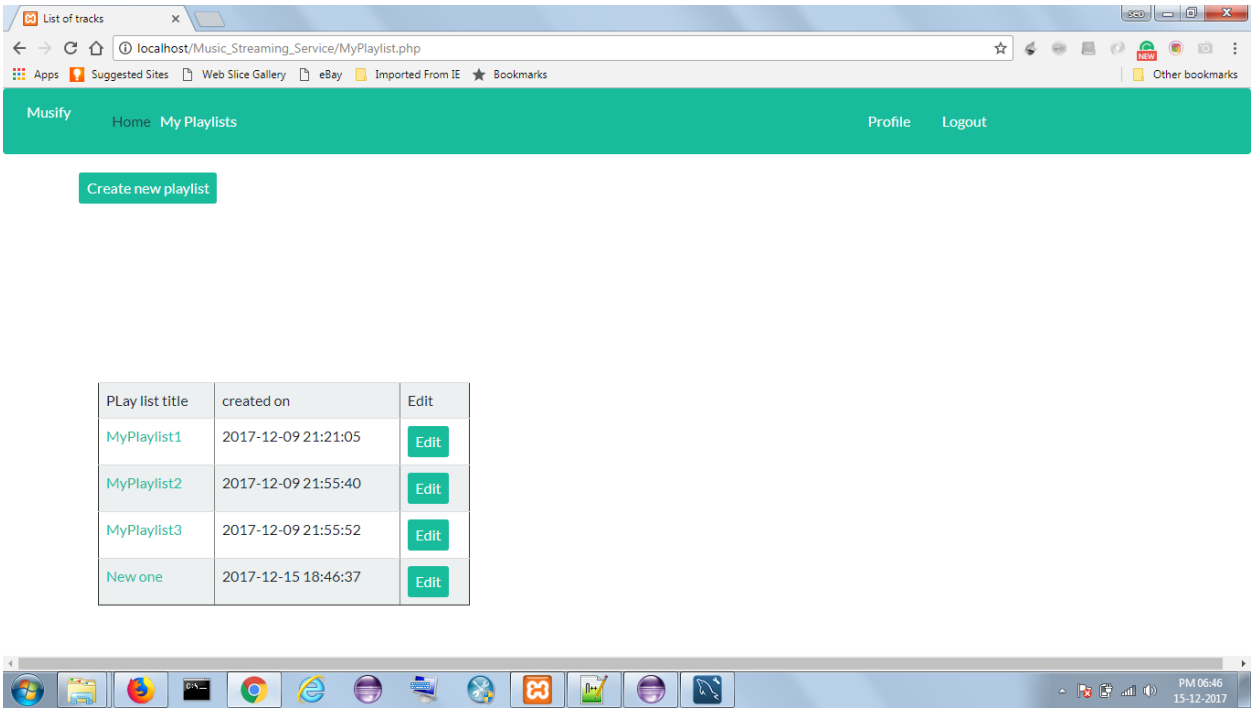
My Playlists Tab (After clicking Create New Playlist):



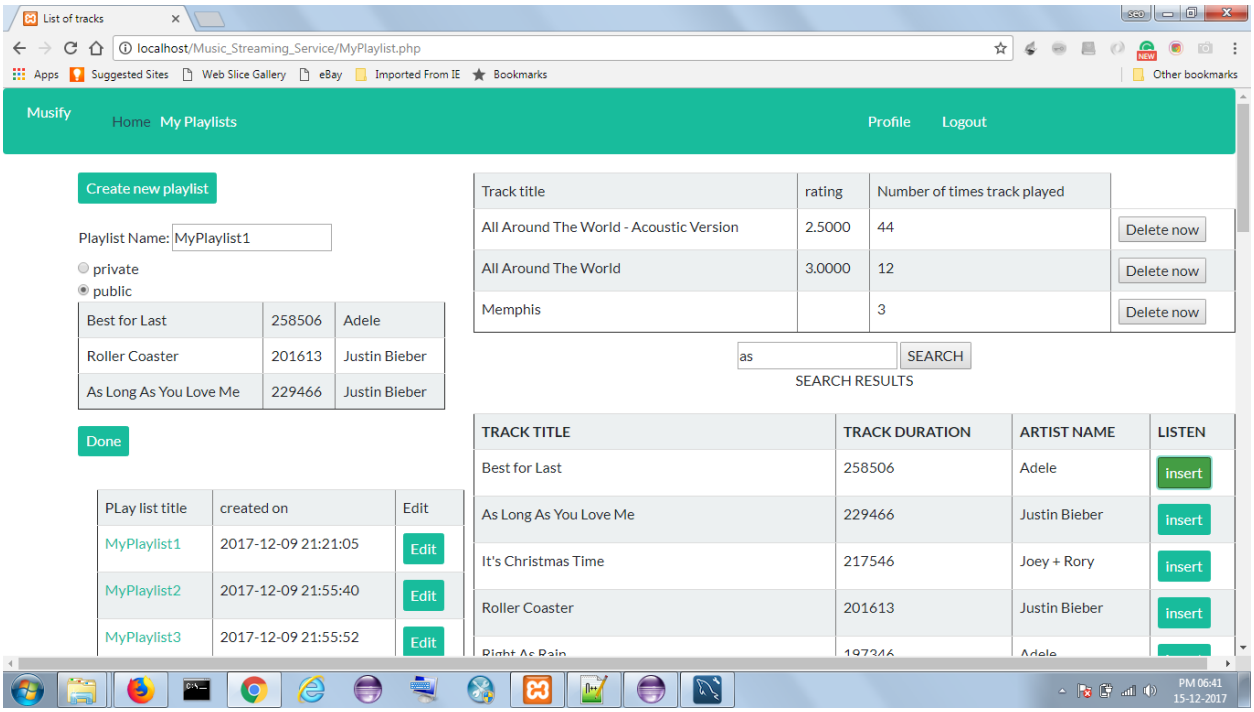
PLay list title	created on	Edit
MyPlaylist1	2017-12-09 21:21:05	Edit
MyPlaylist2	2017-12-09 21:55:40	Edit
MyPlaylist3	2017-12-09 21:55:52	Edit



A Playlist named ‘New one’ is created as a public playlist and the tracks are searched and added to it i.e. Track ‘As Long As You Love Me’ is searched and added to the playlist. After the user is done with adding the tracks he can click on done to save the playlist.

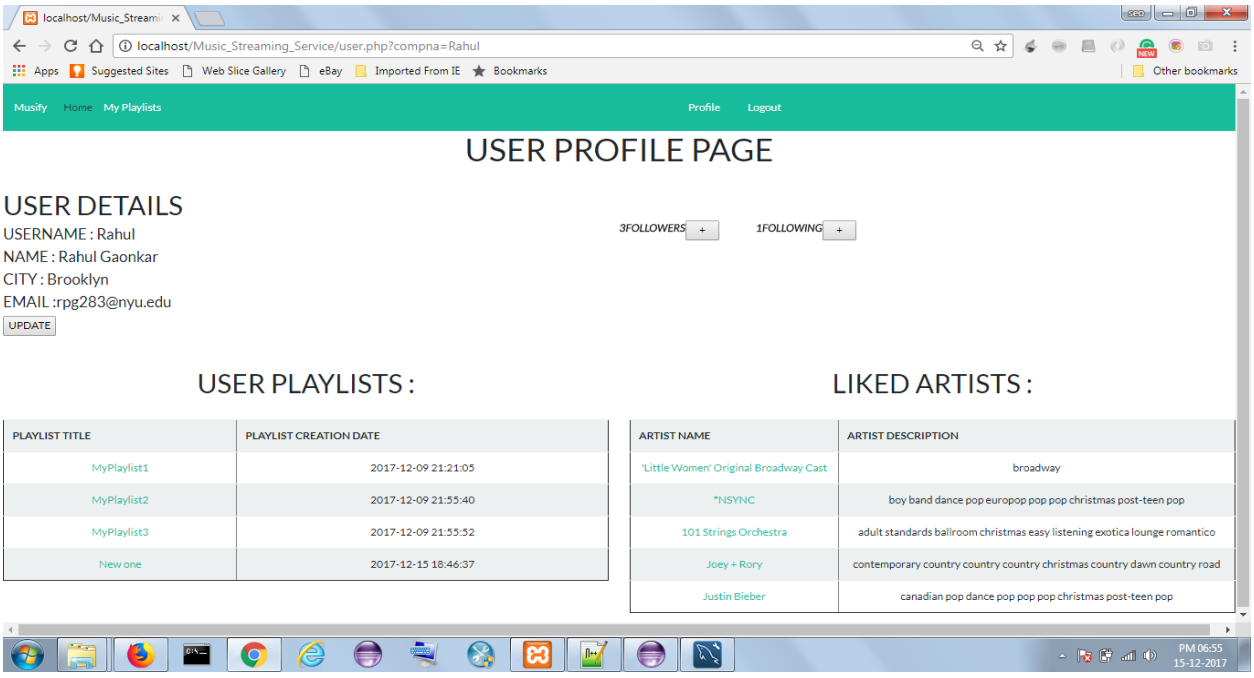


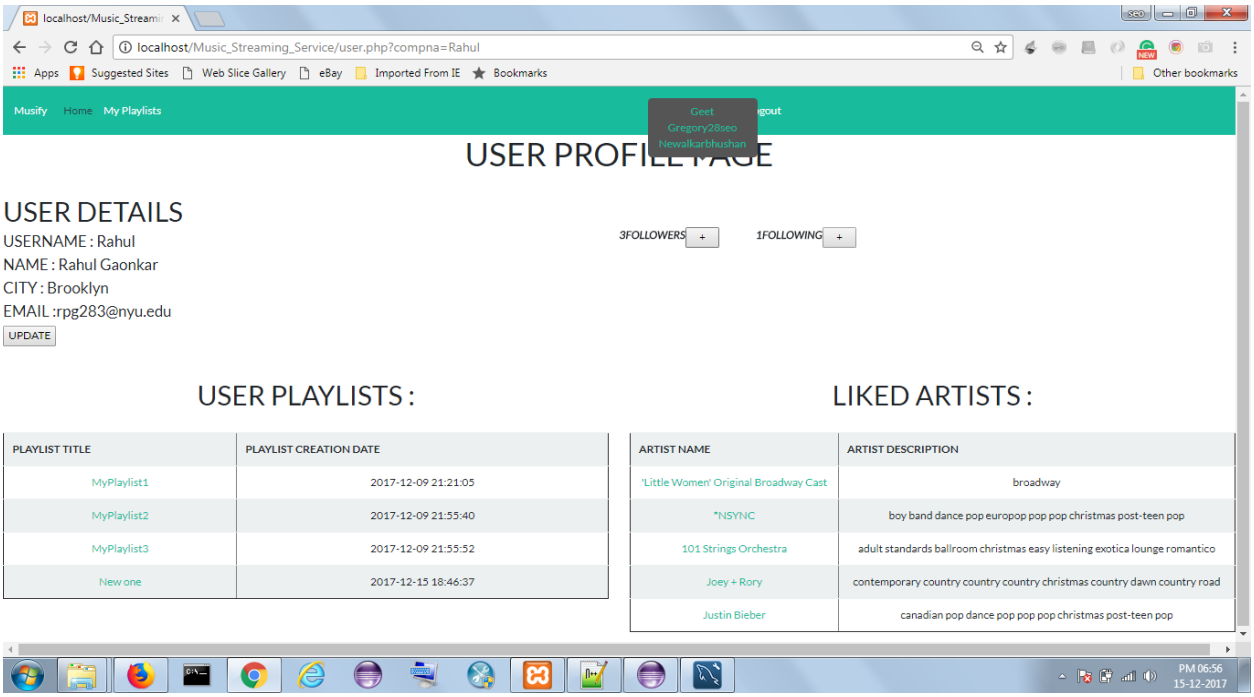
My Playlists Tab (After clicking on Edit to Edit a Playlist):

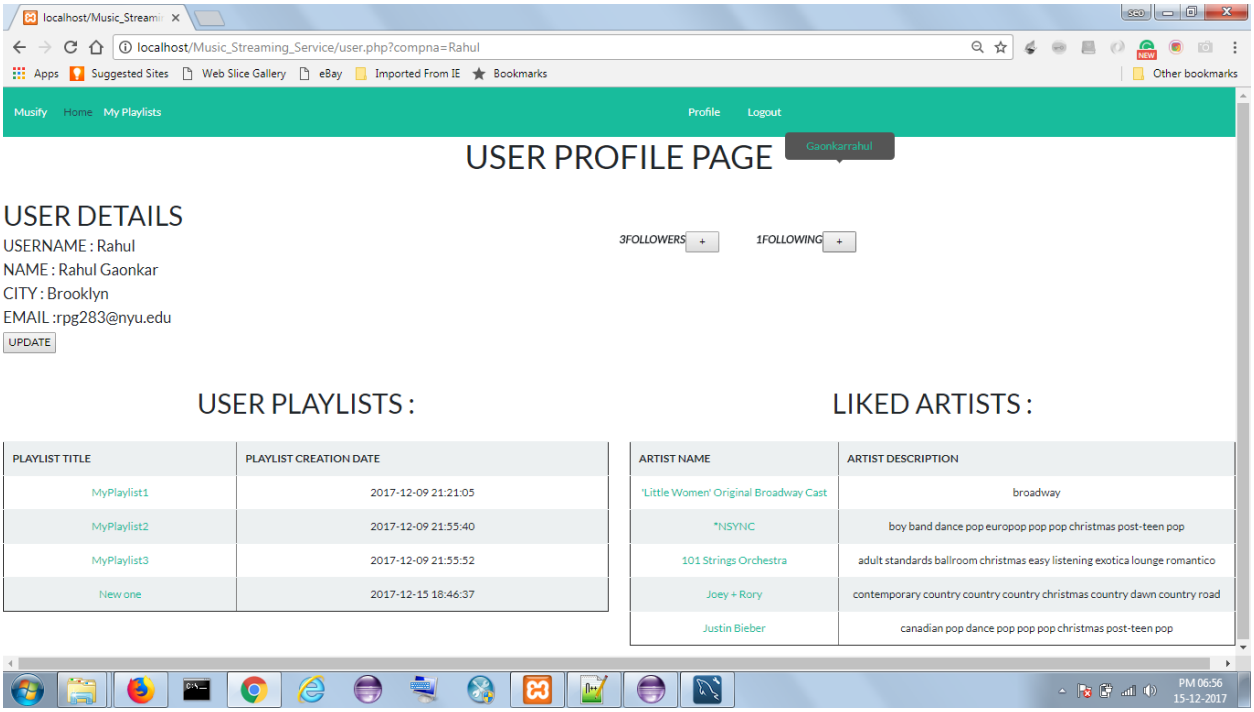


After clicking on edit button to Edit Playlist 'MyPLaylist1', you can change the Playlist name, Playlist type i.e. public or private, add new tracks by searching or delete the existing tracks .Once the user is done with editing the Playlist he can click on Done button to save the Playlist.

Profile Tab (General View):

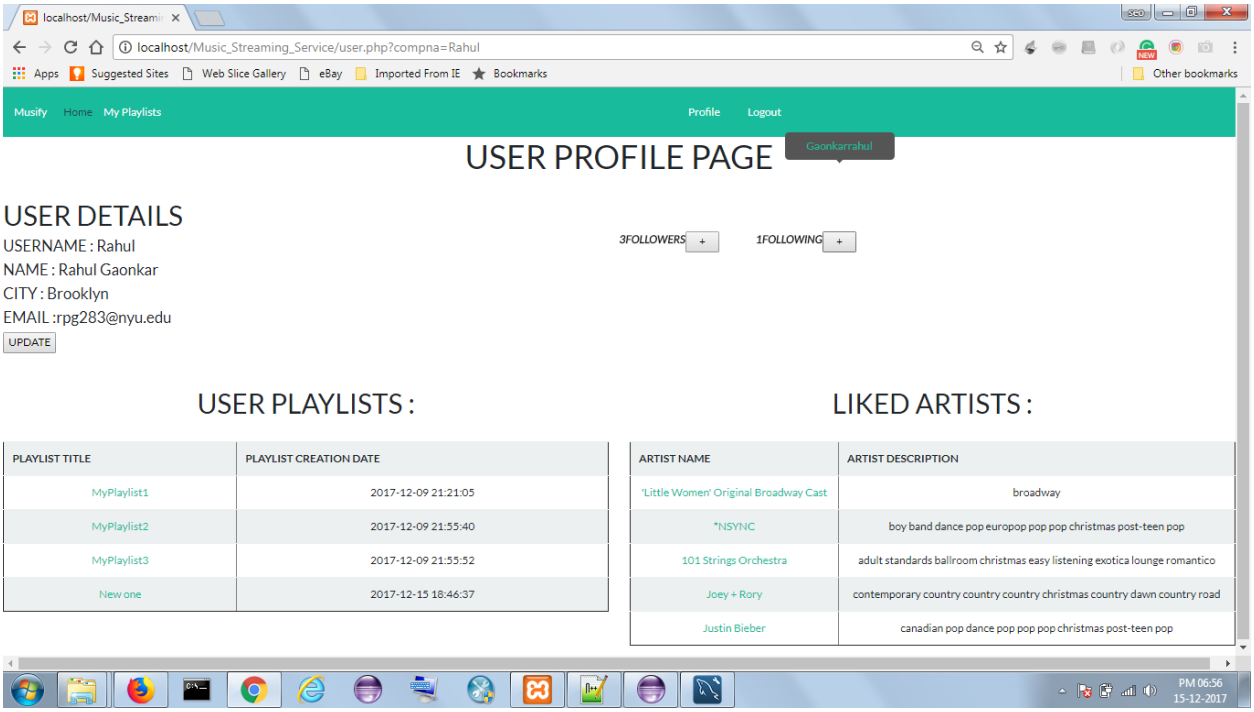


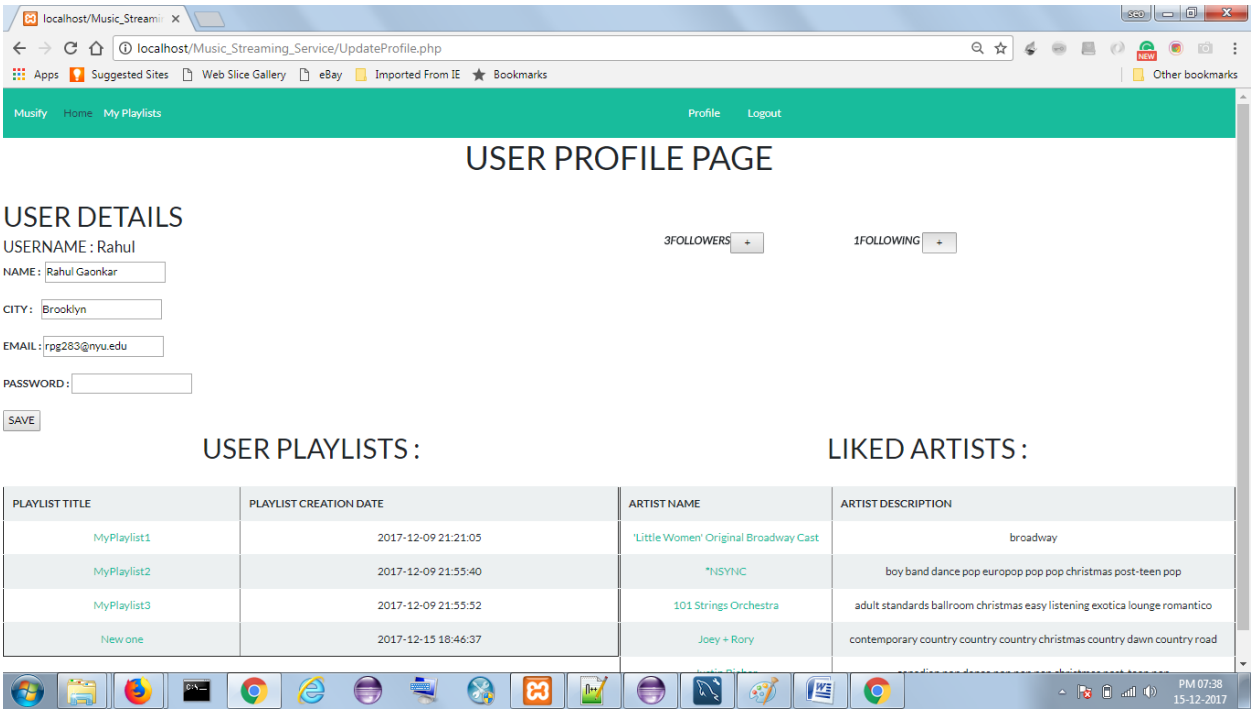




The profile Page of the Logged in User ‘Rahul’ will show all his User details, the play list created by him under User Playlist section, the list of artist like by him under the Liked Artist section. The number of followers and the number of users he is following are also displayed on his User profile page i.e. 3 and 1 respectively. The users in followers and following list are displayed by clicking on their respective ‘+’ button. The User can navigate to the Playlist page or the Artist page by clicking on the Playlist title or Artist Name respectively.

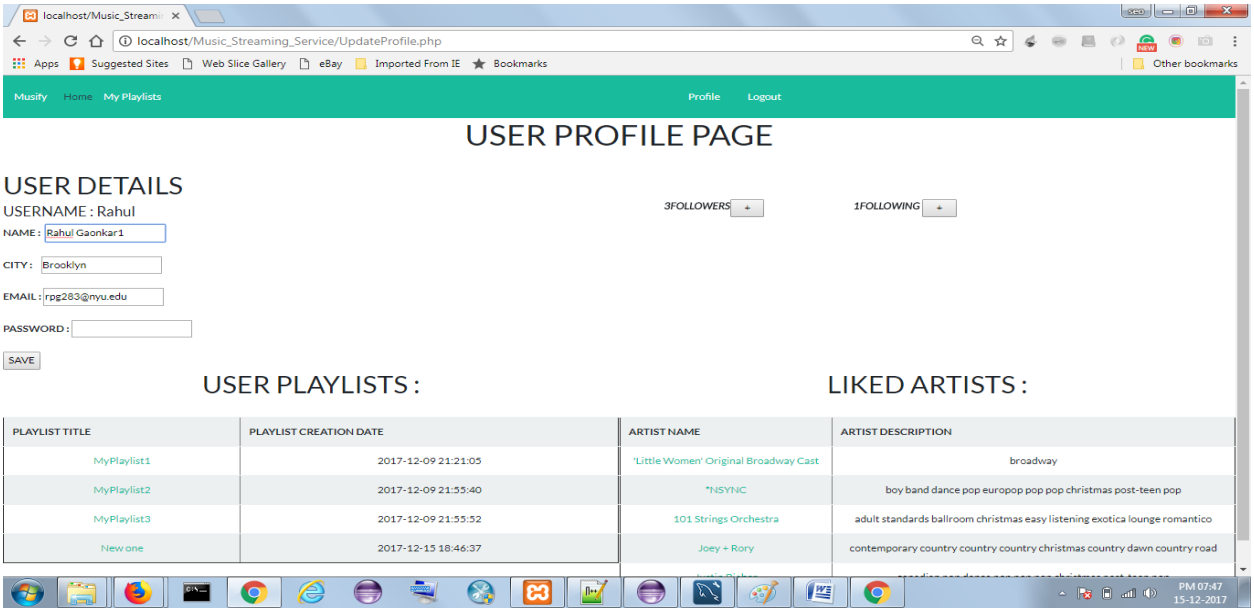
Profile Tab (Updating the Profile by clicking on the update button):

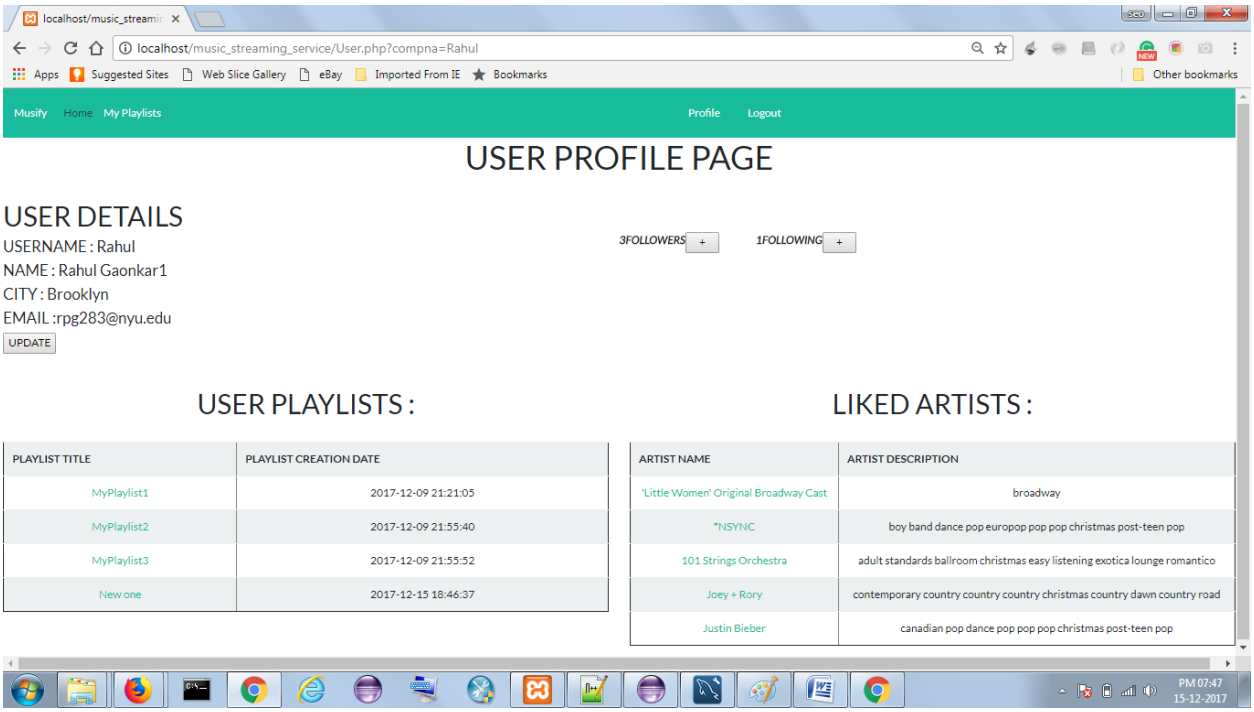




The user can update all the user information except for the Username after clicking on the update button. The previously entered data by the user is retrieved and displayed on the form for all the editable fields except password. Once he is done updating, he can click on save to save the User details. All the fields except the password field is made a required field so that the user doesn't submit empty data here. If the password field is left empty, the existing password is not changed

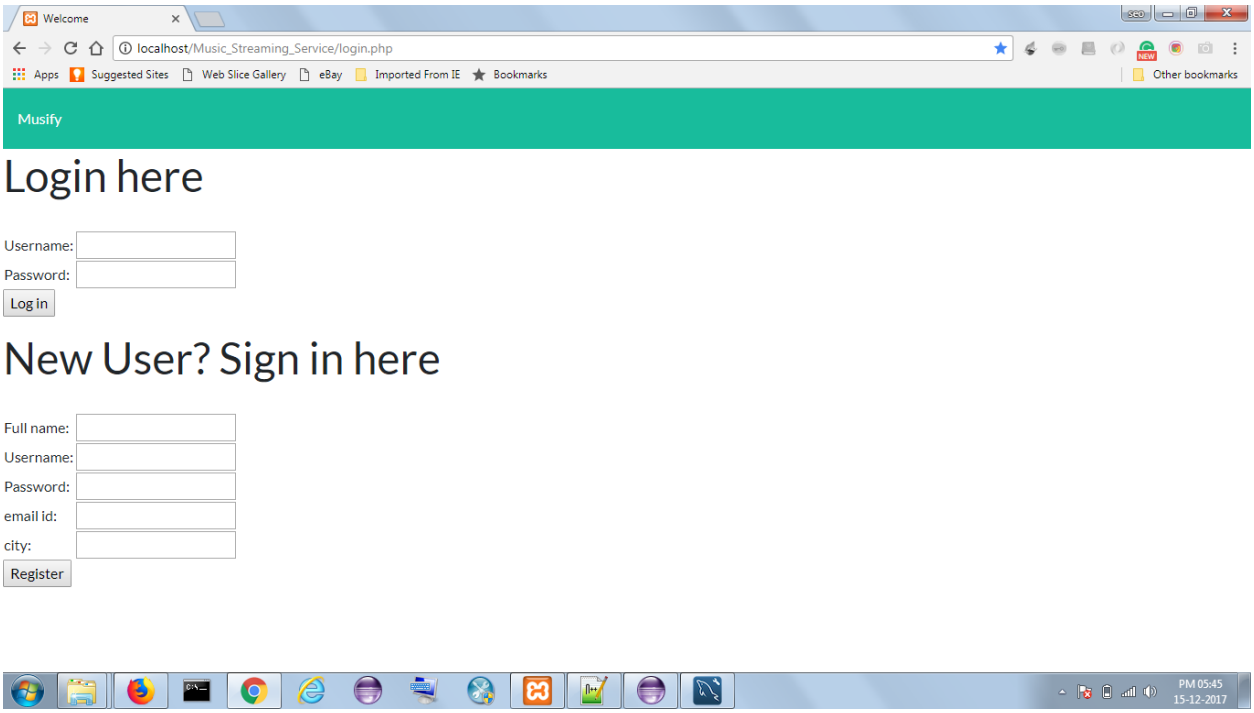
in the database and if the user enters something in the password field of the form, then the existing password is updated with this new value of password.





Here the name of the User with Username ‘Rahul’ is changed from ‘Rahul Gaonkar’ to ‘Rahul Gaonkar1’

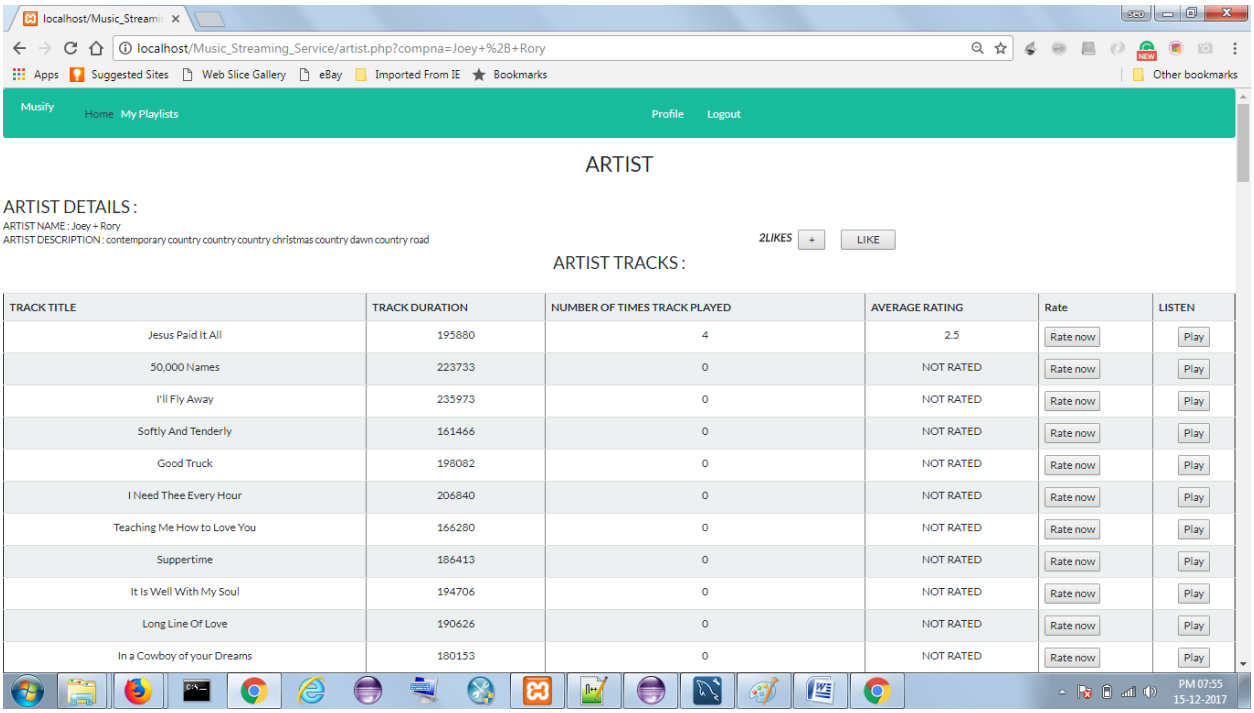
Logout Tab (General View):

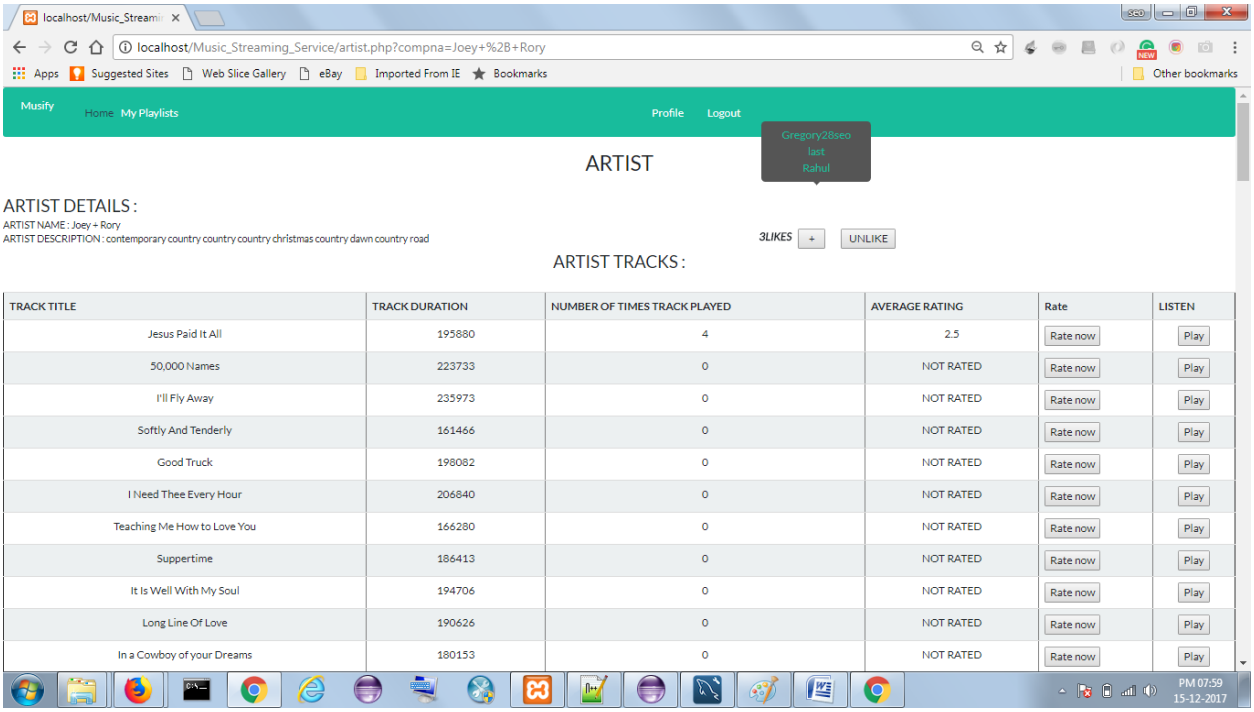


When you click on the logout tab the user session is destroyed and the user is logged out and redirected to the Login/Sign in page so that the user can log in again to the system.

The Homepage will have a menu bar with four tabs namely Home, My Playlists, Profile, Logout. The Home tab navigates the User to the homepage of the web application. The My Playlists tab navigates the User to the page containing all the Playlist created by the user. The User can also create or edit a Playlist on this page. The Profile tab will take you to the user profile page of the logged in User where the User can update all his profile information except the Username, can view the private/public playlist created by him, can view the list of artists liked by him and can view the list of his followers and list of the users followed by him. The homepage of the web application will show recent events relevant to the user in different sections namely Listen Again Section: Showing the last played songs by the user, Recommended section which is divided into two more sections namely tracks: Displaying the latest tracks of the artist liked by the user and Playlist : Displaying the latest playlist by the users they follow, You May also like these Artist : Displays the artist related to the artist liked by the User depending on the number of common fans using Jaccard similarity. The user can play and rate (any value between 0 and 5) each track displayed on the homepage. The number of times the track is played i.e. views and the average rating of the track calculated from the previous rating given to the track by different users is also displayed along with the tracks on the homepage. The number of times the track is played i.e. views and the average rating of the track can be 0 and 'NOT RATED' if the track is not played or rated even once. We can navigate to the Playlist page, User Page or the Artist Page by clicking on the Playlist title, Playlist owner or the Artist name respectively on the Homepage. The user is also able to search by tracks, album or playlist on the homepage. Search by track will match the keyword to be searched with the track name or the artist name of the track. Search by Album or Playlist will match the keyword to be searched with Album title or Playlist title respectively. It will show 'NO RESULTS FOUND' in either of the three cases. The user can also navigate to the Album or Playlist page by clicking on the Album title or Playlist title found in the search result.

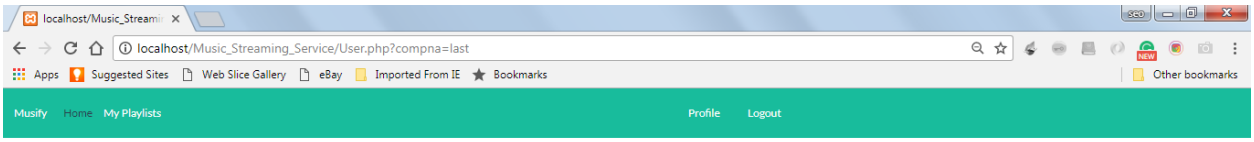
Artist Page:





The artist page will display all the details of the artist like artist name, artist description, tracks created by the artist, number of likes the artist has received and the list of users who liked the artist. The logged in user can also like/unlike the artist by clicking on the like/unlike button. The user can also play or rate the artist's tracks on this page.

User Page:



USER PROFILE PAGE

USER DETAILS

USERNAME : last
NAME : Test
CITY : Brooklyn
EMAIL :poi@sd.com
UPDATE

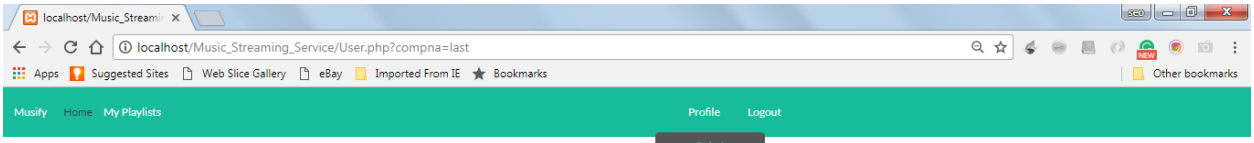
0FOLLOWERS + 1FOLLOWING + FOLLOW

USER PLAYLISTS :

PLAYLIST TITLE	PLAYLIST CREATION DATE
last play 1	2017-12-15 16:25:47

LIKED ARTISTS :

ARTIST NAME	ARTIST DESCRIPTION
Adele	dance pop pop post-teen pop
Joey + Rory	contemporary country country country christmas country dawn country road



USER PROFILE PAGE

USER DETAILS

USERNAME : last
NAME : Test
CITY : Brooklyn
EMAIL :poi@sd.com
UPDATE

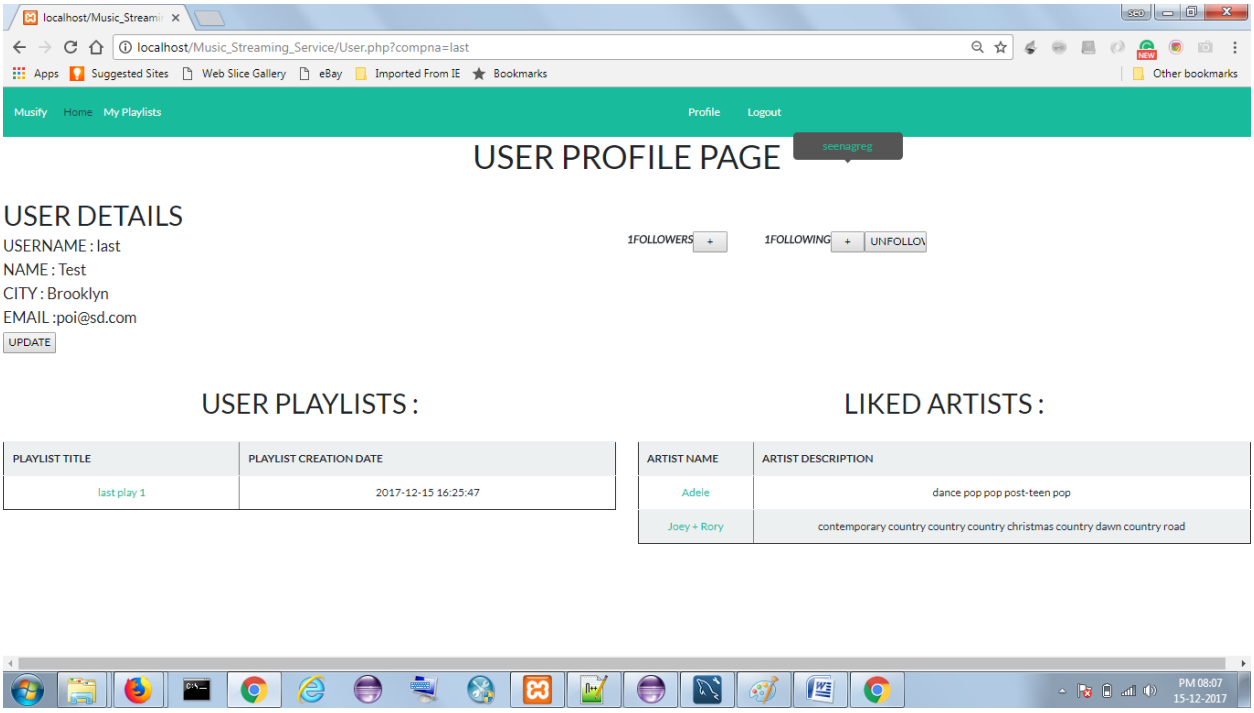
1FOLLOWERS + 1FOLLOWING + UNFOLLOW

USER PLAYLISTS :

PLAYLIST TITLE	PLAYLIST CREATION DATE
last play 1	2017-12-15 16:25:47

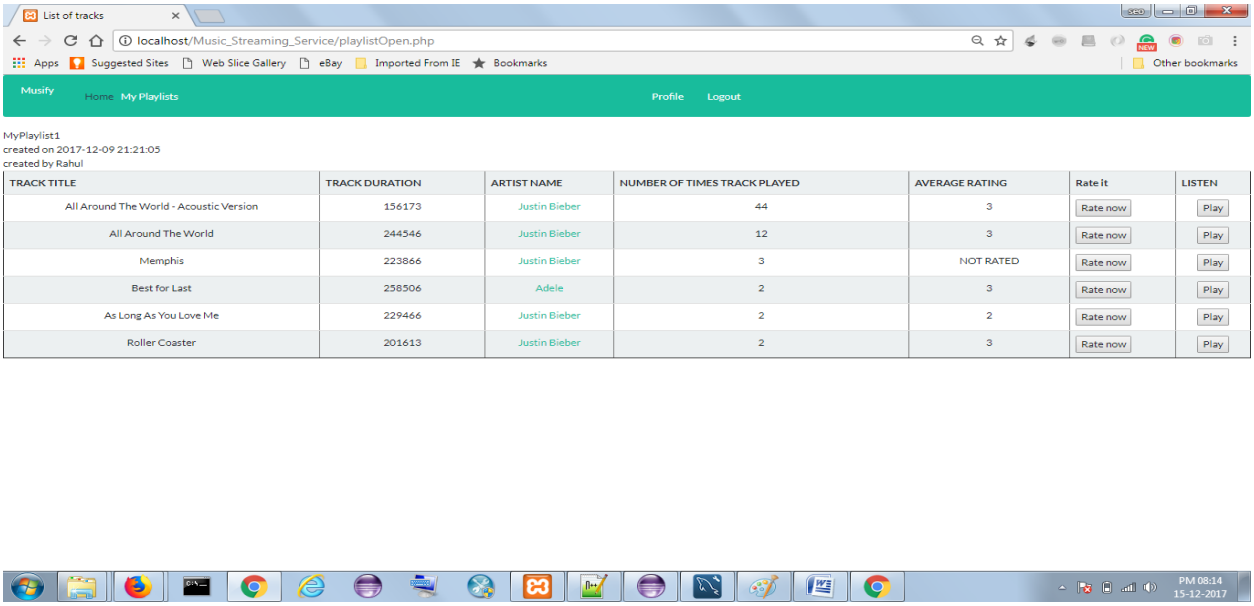
LIKED ARTISTS :

ARTIST NAME	ARTIST DESCRIPTION
Adele	dance pop pop post-teen pop
Joey + Rory	contemporary country country country christmas country dawn country road



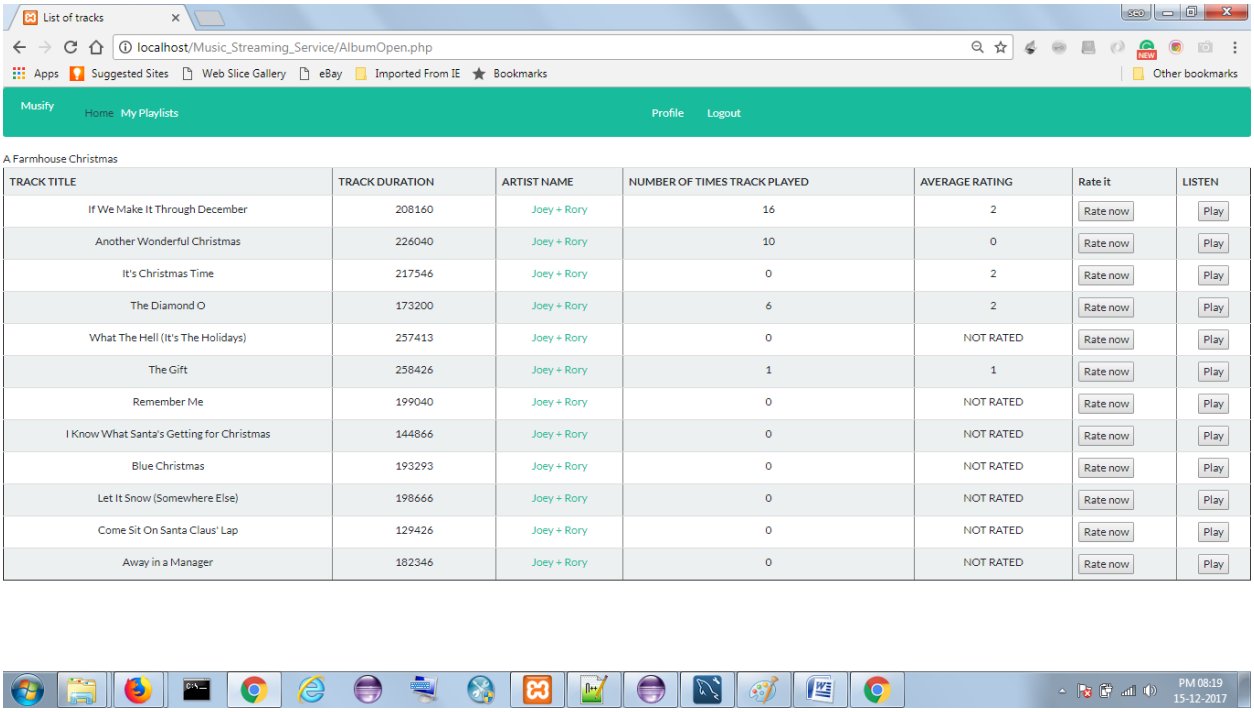
The User page will display all the details of the User like Username, name, city, email, list of playlists created by the User, list of artists liked by the user, number of followers and following of the user, list of followers and following of the user. The logged in user can also follow/unfollow the User by clicking on the follow/unfollow button. The user can also navigate to the Playlist page or the Artist page by clicking on the Playlist Title or the Artist name respectively.

Playlist Page:

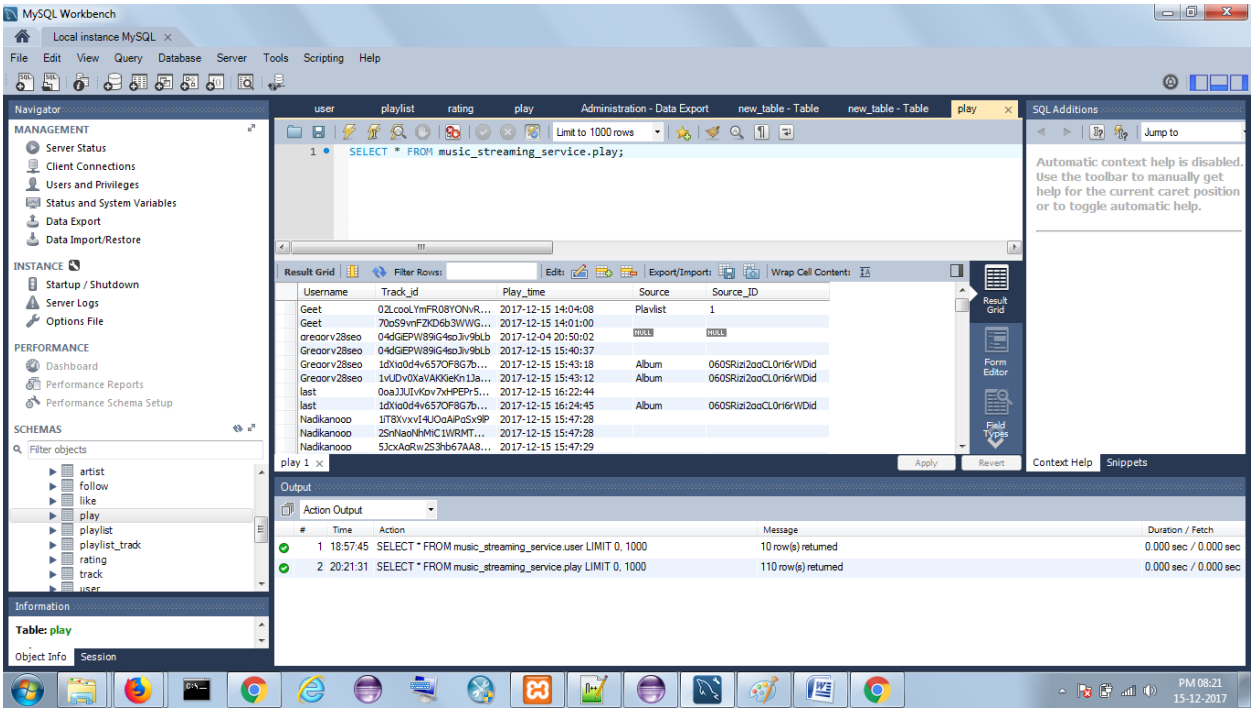


The playlist page displays all the information related to the Playlist namely Playlist name, creation date, Playlist owner name, list of tracks in the Playlist. The user can navigate on the artist page by clicking on Artist name on this page. The user can even play or rate the track on this page. If the user plays a track on this page the play record stored in the database will also include the source of the track i.e. Playlist and its corresponding Playlist ID.

Album Page:



The album page displays all the information related to the album namely Album name, list of tracks in the Album. The user can navigate on the artist page by clicking on Artist name on this page. The user can even play or rate the track on this page. If the user plays a track on this page the play record stored in the database will also include the source of the track i.e. Album and its corresponding Album ID.

PFB the screenshot of the Play table in the database to see how the records are stored in the database. If there is no value for Source and Source_ID attribute of the play table that indicates that the track is played directly and not form any album or Playlist.

Musify, an online music streaming service