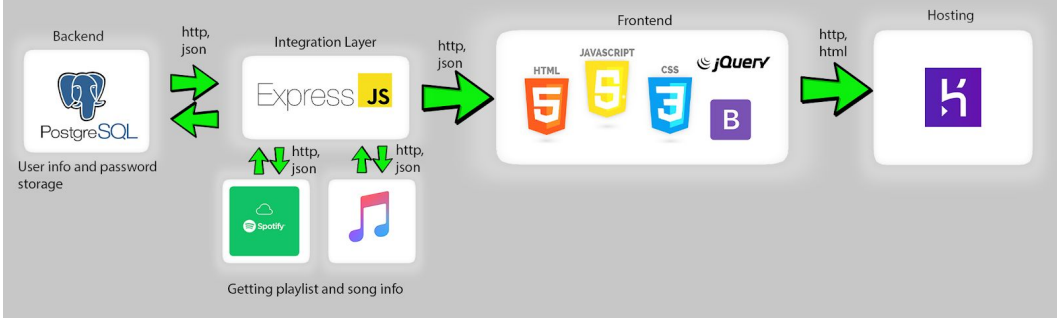| Revised List of Features	● Login Page
	    ○ Prompts user to select which music services they would like to log into
	        ■ WIll support Apple Music and Spotify
	● Home
	    ○ Contains an element (square button) which will contain songs
	    ○ Will have another button for playlists
	    ○ Some additional elements that could be integrated include:
	        ■ Artists
	        ■ Downloaded Songs
	        ■ Stations
	● Search
	    ○ Allows user to search from music/artists on several different platforms and integrate it into elements listed above
	● Playlist Migration
	    ○ Able to put all songs from one playlist on one platform into another

Architecture Diagram	 Or click here for bigger image: https://drive.google.com/file/d/1xZsn4hKHvw1mahVqdH1jzdE6MrmpnN5L/view?usp=sharing
Front End Design	While user is logging in, they can search songs that they wish to listen to, user can have the ability to create playlist and add in their favorite music, they can choose how they want to play music in the playlist such as random or play music in order. They will have the ability to listen to music on the website. All of above function will be done by html and we will change the style of the website by css. There should be multiply html files that links together in order to achieve the goal.

Home

Create username

Create password

Forget password?

Home
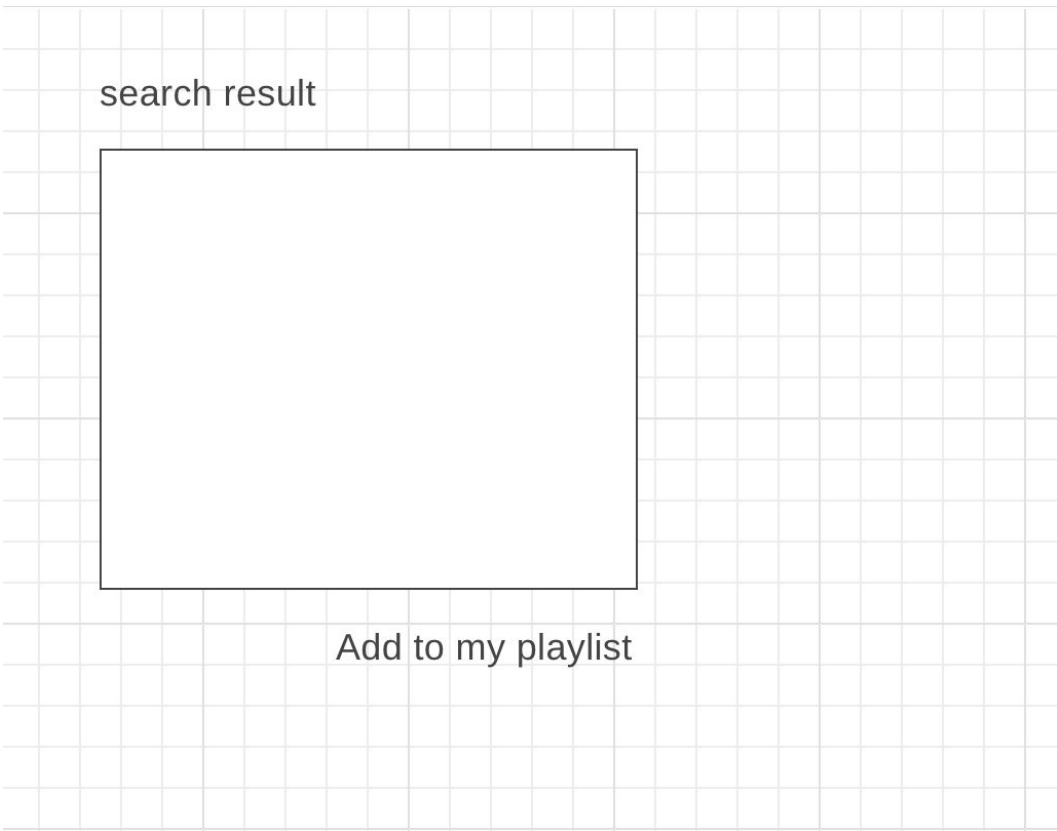
Search

Playlist

Play order

random
in order

login

platforms to choose from

username

apple music

password

spotify

not a user? create one

	search result Add to my playlist
Web Service Design	The initial web services we will be starting with our app is Apple Music Web Services and Spotify Web Services. Both of these are just the web services associated with their music providing service. We may add more, depending on time constraints. Both are Restful and use Http and jsons to communicate. The information we will be getting and receiving on each of these web services is virtually the same. We will be getting music from a playlist in the form of a json, and searching for songs using song title and artist value paris for the other playlist, retrieving that object id in the form of a json, and then adding it to a new playlist on the other service. Of course, this will be after the user enters their username and password for each music service in the web service for authentication. Additionally, if we find a way to get songs from a playlist and play through browser, we will do that if we have time and can figure out the logistics of that. This, however, is a stretch goal.
Back End Design	

User
- 🔑 User ID INT
- ◇ Linked Apps LONGTEXT
- ◇ MusicListID LONGTEXT
- **Indexes** ▶

MusicList
- 🔑 MusiclistID INT
- ◇ UserID INT
- ◇ Apps TEXT(60)
- 🔴 User_User ID INT
- **Indexes** ▶

User_Account
- 🔑 UserID INT
- ◇ Username VARCHAR(45)
- ◇ UserPassword VARCHAR(45)
- ◇ Phone INT
- 🔴 User_User ID INT
- **Indexes** ▶

MusicApps
- 🔑 MusicAppId INT
- ◇ MusicID VARCHAR(45)
- ◇ LinkedUserID VARCHAR(45)
- 🔴 User_User ID INT
- 🔴 Music_MusicID INT
- **Indexes** ▶

Music
- 🔑 MusicID INT
- ◇ MusicName VARCHAR(45)
- ◇ Singer VARCHAR(45)
- ◇ MusicTime TIME
- ◇ MusicApps VARCHAR(45)
- ◇ MusicURL VARCHAR(45)
- 🔴 MusicApps_MusicAppId INT
- 🔴 MusicList_MusiclistID INT
- **Indexes** ▶