Online Music Store

Christopher Slattery **B00092939**
Joseph Tierney **B00092923**
Derek McCarthy **B00007439**

Supervisor: Michael O'Donnell

Declaration

I herby certify that this material, which I now submit for assessment on the programme of study leading to the award of Degree of **Honours B.Sc. in Computer Science** in the Institute of Technology Blanchardstown, is entirely my own work except where otherwise stated, and has not been submitted for assessment for an academic purpose at this or any other academic institution other than in partial fulfilment of the requirements of that stated above.

Signed: _____ Dated: ____/_____/_____

Christopher Slattery

Signed: _____ Dated: ____/_____/_____

Joseph Tierney

Signed: _____ Dated: ____/_____/_____

Derek McCarthy

Abstract

This paper describes the technologies and approach used to create and maintain a music website that is used to provide users with the details of a lot of the music that is currently out today. The paper also describes the research and analysis involved in undertaking the creation of such a website. The design and implementation will also be portrayed in this paper. It will also describe the learning outcomes that are gained from creating a website like this.

Acknowledgments

We would like to thank our supervisor Michael O’Donnell for his continuous support and guidance.

Table of Contents

Chapter 1: Introduction

We decided to create a music website as there are not so many music websites out there.
There are a few music websites, but not nearly enough for the sheer amount of people that listen to and enjoy music. For many years human culture has listened to and enjoyed music. Humans have been enjoying music since. Music has been around since the year 1000 BC in some form. Music has been slowly evolving all through the history of humankind. In 1948 Columbia Records showed the world the first vinyl record. Even today this remains the standard for a vinyl album. Vinyl's were then advanced to hold around 3 minutes of music on each side. Since then music and how people acquire music has changed drastically. As the internet came to fruition, how people listened to and purchased music changed. Many people didn't expect that in a few short years, it would change how people listen and buy music forever. It would also change how artists release music. In 1994, a band named Aerosmith became the first band to have a song downloaded from the internet. They did this in collaboration with their record label, Geffen. The song took 60-90 minutes to download and it was carried out as a tech experiment at first. Fast forward fifteen years and things like MP3 players and Walkman's were quickly becoming obsolete.

1.1 Music Website

The emergence of websites and software such as Napster, iTunes, Spotify and even YouTube has attributed to devices such as Walkman's becoming obsolete. Napster were the first to make available the possibility to share and download songs in MP3 format. iTunes made it possible for music to be listened to and purchased on the internet and Spotify also did this. There are also many websites that are illegal that allow people to download music for free. This has become a big problem for music artists and record companies as they are losing out on huge amounts of money because of this. It has even led to some artists like Jay-Z creating his own website for his and his artists music to be solely available on his website.

1.2 Project Outline

When discussing the website before we started, we thought it would be a good idea to incorporate some of the main features that are needed and thought it would be a good idea to investigate how other big music websites went about doing this.
While discussing what our website should entail, we decided on some things that should be included. This included:

- A login system for admins and users.
- Security for the website
- A search function for users to search the website.
- A database to store all our user data.

- Pages for albums, songs, and artists.

We would also like the website to have many more features like bringing users the latest songs and albums from today's music charts. We also want the website to have users(Registered) to download songs and albums for future listening. If a user is not a registered user, they can browse songs and albums only and hopefully this will help persuade new users to register with the website. We want the website to have two types of members registered, a regular user and an admin. The regular user once logged in will have a profile which consists of a profile image, name, and the option to reset password. The admin once logged in will have the same features as a regular user but with an added menu bar button which will allow them to interact with the database such as adding, editing, and deleting both users and songs. As for security, we want the website to use Secure Sockets Layer (SSL) certificates to establish encrypted links between the server and the website. Also, as user's passwords need to be stored in the database we will look to store the hashed value of the password which will be generated by the PHP hashing algorithm.

1.3 Prototype

When discussing the project before we started, we talked about how we wanted the website to look and feel. We implemented a prototype using wireframe which can be seen below in chapter 3. This helped us visualize what we wanted the website to look like or to not look like when it was finished. When we finished the prototype, we discussed what else we needed to achieve in order to turn that prototype into a complete website. At the start, we discussed how the layout and main theme of our website should be implemented. We then discussed the finer details of the website, like how people would create their own accounts and how we would implement the 3 of us as administrators on the website.

1.4 Skills/Knowledge Required

Developing a website that is up to industry standard and ready for commercial use requires many skills and a lot of knowledge about different aspects of computing. It also requires skills outside of computing like interpersonal skills for example. One of the main skills we needed for developing this website was knowledge in the programming language, PHP. we used PHP to develop this website. We decided that it would suit us best to use that language as we were implementing a cart which users could purchase songs from. We also used some HTML (Hypertext Markup Language), JavaScript and SQL. As we are computing students, we have built up many years of computing skills and have been doing modules that involve theses skills for many years now. Modules like Relational Databases helped us massively with implementing a database for our website to store users as well as songs. Developing this website as a team was a big help as we all get the same enjoyment out of music and of course computing techniques.

1.5 Deliverables

When developing a website like this, we knew we needed to stick to deadlines with regards to what we needed to deliver and when we needed to deliver it. We set deadlines for when each person needed to have a certain part of the website complete and it helped us immensely. This included prototypes, documentation, evaluation of the website and evaluation of the work complete at different phases. We also discussed how we would test the website. We will test the website with functional testing and unit testing. We will do this when we have the website complete. We will use PHP unit for unit testing. This will help us test every function and it will also allow us to test the URL roots for our website. For functional testing we will use a tool called JMeter to send a lot of users to the website and test the stability of it. We also asked fellow students to use our website and create accounts to make sure tested as much of the website as we could.

1.6 Report Outline

While reading the rest of this report, the reader will grasp everything about the project. The next section will show the reader an analysis of the system and walk them through how the system works behind the scenes. It shows class diagrams to help the reader understand the programming behind the website. This includes the classes and functions that went into creating the website. It will also have a Functional Decomposition Diagram to show the reader the structure of the website and how it works. After that it will show an in-depth explanation of how everything in the website interacts and co-exists with each other. We will also show an ERD (Entity Relationship Diagram). The system design chapter will give the user a deep understanding of how we designed the website. Before we started developing and while we were developing. This will be illustrated with wireframe diagrams which are prototypes of how we wanted the website to look. Chapter 4 will explain to the reader how we implemented the website and what technologies went into creating it. The testing and evaluation chapter will guide the reader through how we tested the website and what the end product is. The paper will then conclude with a summary of the website and the work put into it.

Chapter 2: System Analysis

2.1 Functional Requirements

2.1.1 Class Diagram

Overall class diagram of how each PHP class interacts with one another. To see the classes in more detail including the variables and methods used in that class see Appendix A.

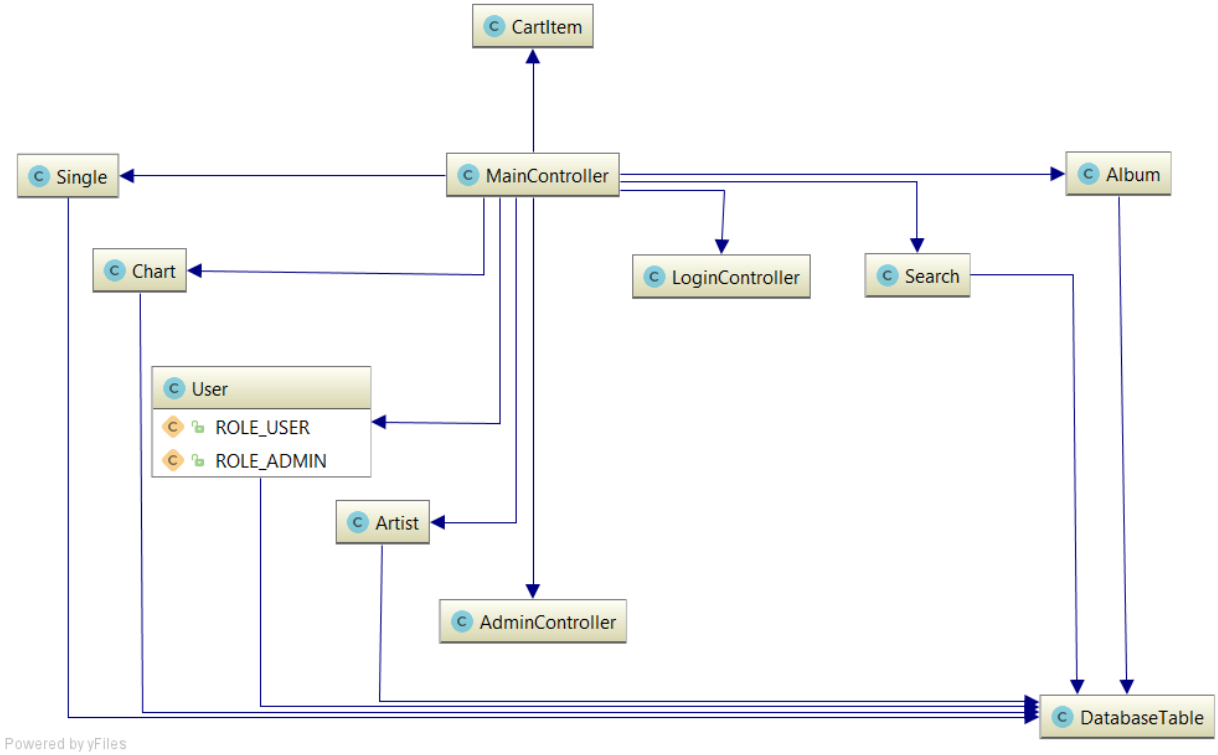


Figure 1 - Class Diagram

2.1.2 Functional Decomposition Diagram

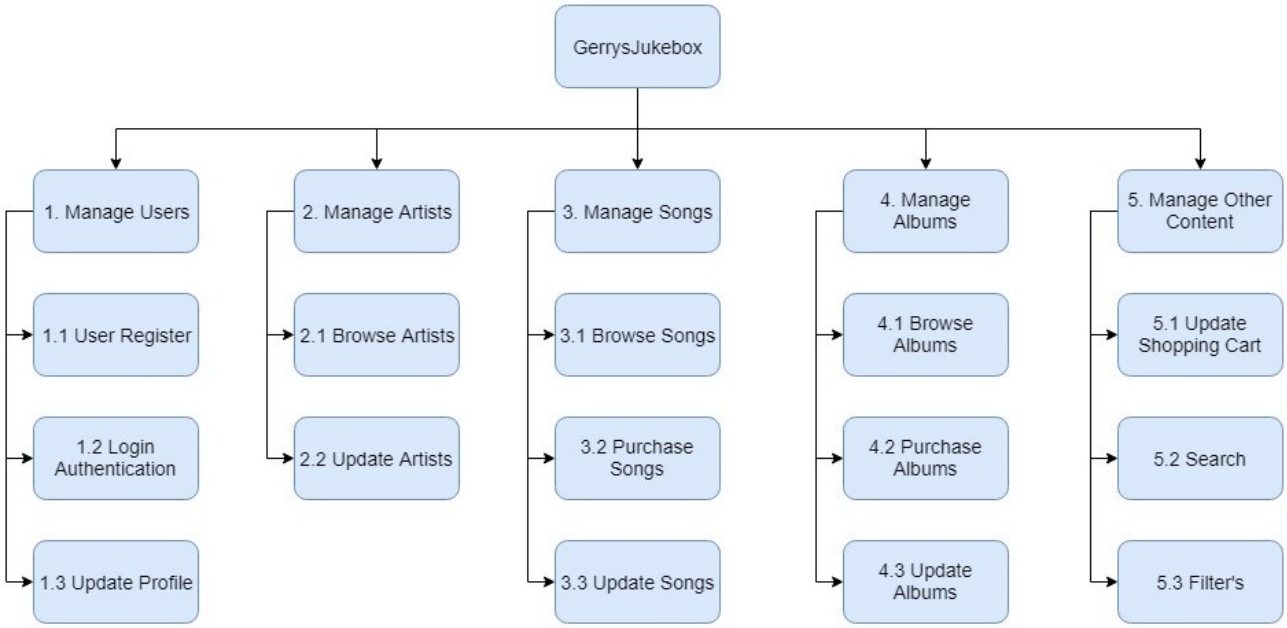


Figure 2 - Functional Decomposition Diagram

This diagram comprises of different pieces of functionality that the website should provide. Each of these functions is broken down further into smaller processes.

2.1.3 Functional Decomposition Diagram Description

Below are the descriptions of the major functions that our website will incorporate from the diagram in fig 2.

Manage Users

User Registration
Only registered members can access certain aspects of the website such as purchasing music etc. To register the user needs to fill in a registration form with their personal information, this includes username, password, and email address.

Login Authentication
This is one of the most important functions of the website as it safeguards the website from malicious users. Only users with administrator rights have the ability to make changes to the websites content. Also, in order to purchase music or access certain aspects such as the shopping cart or user profiles the user needs to be a registered member and logged into the website.

Update Profile
This is a function available to registered users who are logged into the website. With this function users have the ability to customise their profile, by updating their profile image, changing their password and customising the look and feel of the website in general.

Manage Artists
Solely used for all aspects relating to the artists and their details.

Browse Artists
This function allows the user to browse all the artists listed on the website. From this function there is a sub-function allowing the user to view an individual artist and their details.

Update Artists
This is an administrator function where an artist and their details can be added, deleted or updated on the website. This is an essential function for adding future artists to the website.

Manage Songs
Solely used for all aspects relating to the songs on the website and their details.

Browse Songs
This function allows the user to browse all the songs listed on the website. From this function there is a sub-function allowing the user to view an individual song and its details.

Purchase Songs
This feature allows the user to purchase/download song(s) by first adding them to their shopping cart, and then checking out once their credentials have been verified.

Update Songs
This is an administrator function where a song and their details can be added, deleted or updated on the website.

Manage Albums
Solely used for all aspects relating to albums and their details.

Browse Albums
This function allows the user to browse all the albums currently listed on the website. From this function there is a sub-function allowing the user to view an individual album and its details.

Purchase Albums
This feature allows the user to purchase/download album(s) by first adding them to their shopping cart, and then checking out once their credentials have been verified.

Update Albums
This is an administrator function where an album and its details can be added, deleted, or updated on the website.

Manage Other Content
Solely used for all other aspects of the website that don't necessarily belong to a particular group.

Update Shopping Cart
This feature is used to manage the shopping cart and its contents. It allows albums or songs to be added to it or removed. It is also responsible for managing the total cost of all items in the shopping cart.

Search
This feature allows the user to search the website for a particular artist, song, or album. If the users search matches one of the items on the website this item is then displayed. If there is no match the user will be informed.

Filters
This function is present on both the browse albums and songs page. It enables the user to filter by genre. Which displays all the items matching the chosen genre.

2.2 Use Case Diagram

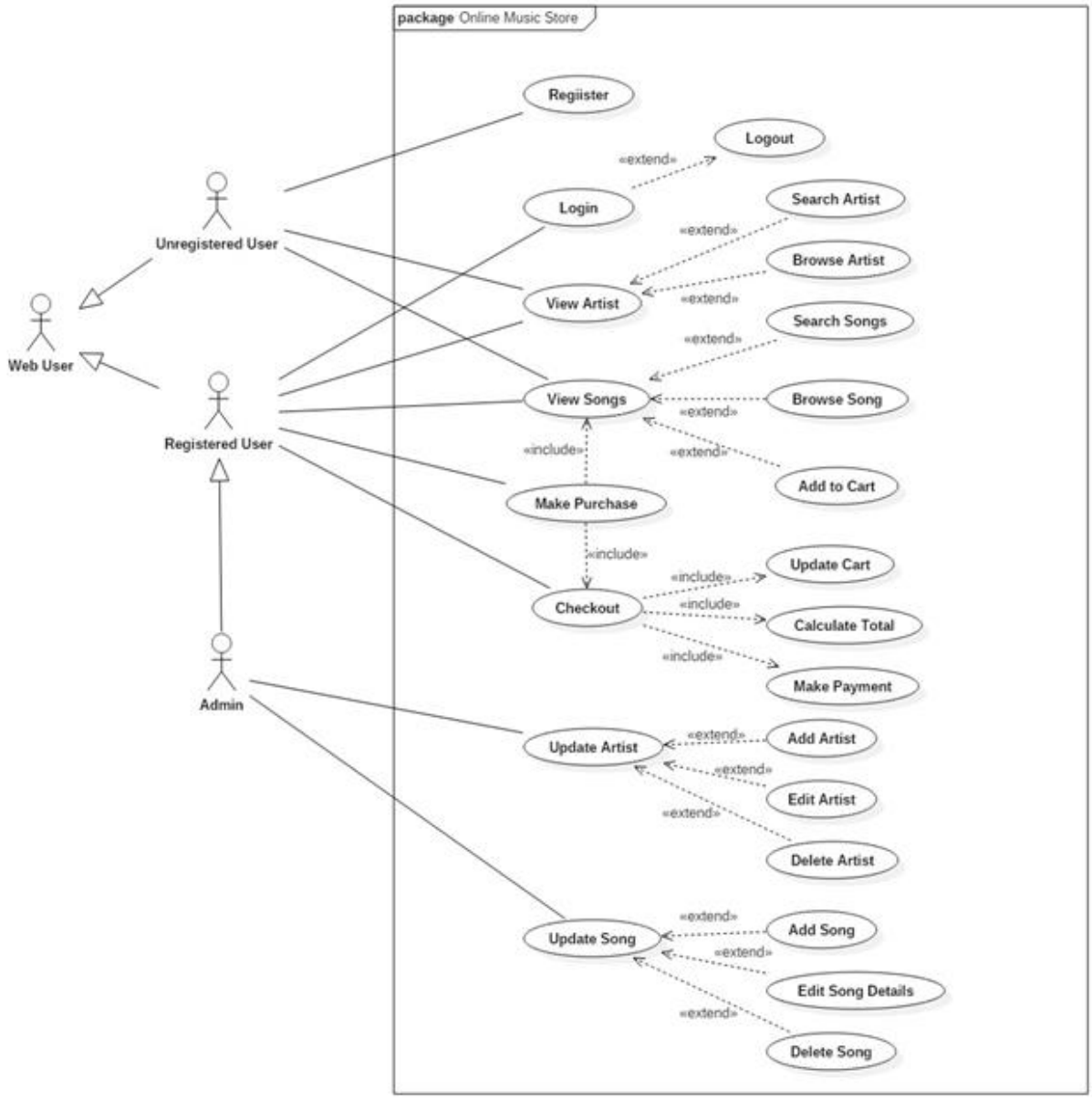


Figure 3 - Use Case Diagram

2.3 Data Requirements

The website uses many different database entities to store its data such as:
Users - stores the required details about the user.
Albums - consists of an artist and a number of songs.
Artists - stores the artists details.
Songs - consists of both an album, artist, and the songs details.
Charts - stores the top 15 chart which consists of both an album and an artist and the number of weeks on the chart.

2.3.1 Entities and their Attributes

Entity Relationship Diagram (ERD)

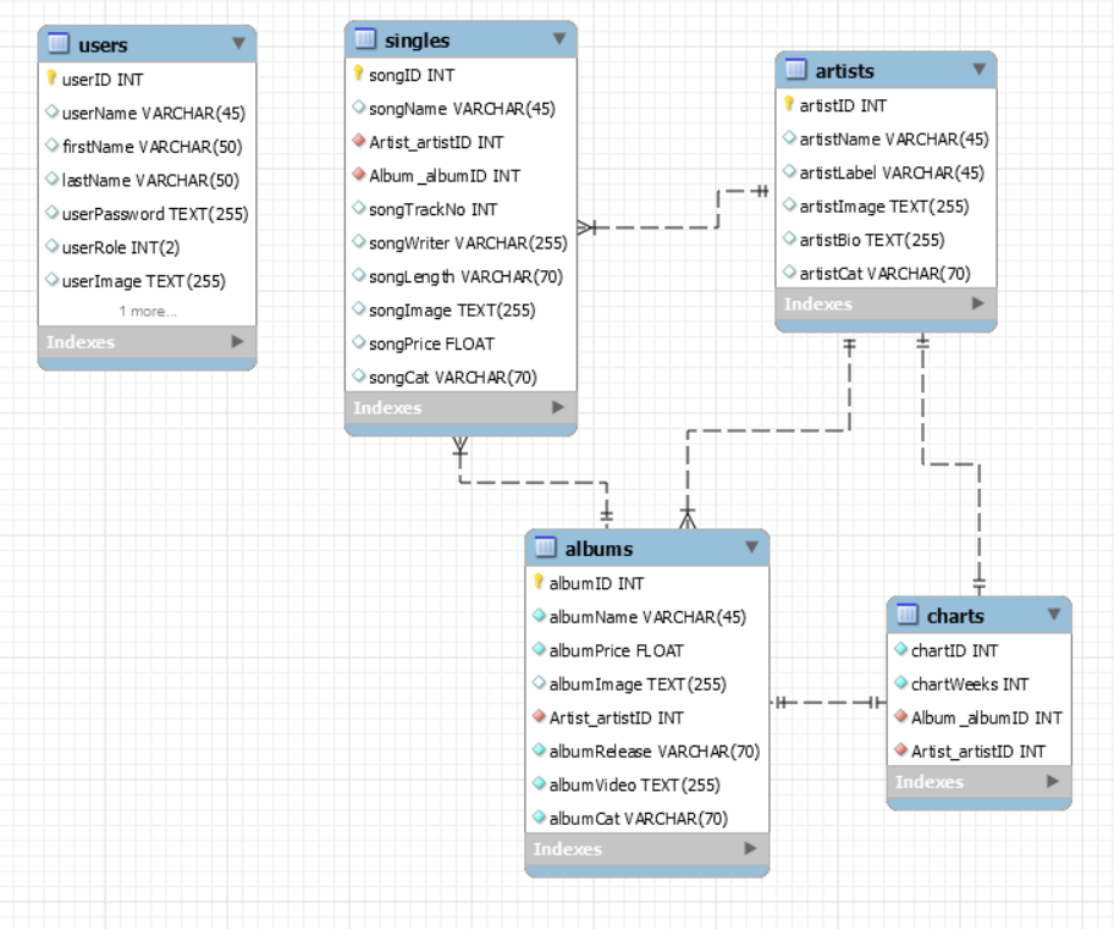


Figure 4 - Database ERD

This is a list of all the attributes for each entity in the ERD in fig 4 and the function it performs in the website.

Users Entity

Attributes:
userID - the user's id which uniquely identifies them from other users.
firstName - the first name of the user.
lastName - the surname of the user.
userName - the name which will be displayed on the user's profile and also used for login authentication of each user.
userPassword - stores the hashed value of the user password for security reasons and is used for login authentication along with the username.
userRole - used to distinguish between regular users and admin users. Admin users will have an integer value of 2 while a regular user will have a value of 1.
userImage - stores the user's profile image and its location directory as a string value.
userEmail - the email address of the registered user.

Albums Entity

Attributes:
albumID - the albums id which uniquely identifies it from other albums.
albumName - the title of the album.
albumPrice - how much the album costs to purchase.
albumImage - stores the album cover image and its location directory as a string value.
albumCat - the musical category of the album e.g. Rock, Pop, Jazz etc.
albumRelease - the date in which the album was released.
albumVideo - stores an embedded link to a video from the album.
artistID - foreign key reference to the artist table which binds both tables together through the artistID.

Singles Entity

Attributes:
songID - the songs id which uniquely identifies it from other songs on the website.
songName - the name of the song.
songPrice - the price of the song to purchase.
songImage - stores the songs cover image and its location directory as a string value.
songCat - the musical category of the song e.g. Rock, Pop, Jazz etc.
songTrackNo - the track number of the song on its album.
songLength - runtime of the song**.**
songWriter - writers of the song.
albumID - foreign key reference to the albums table using the albumID of both tables.

artistID - foreign key reference to the artists table using the artistID of both tables.

Artists Entity

Attributes:
artistID - the artist id which uniquely identifies it from other artists on the website.
artistName - the name of the artist.
artistLabel - record label the artist belongs to.
artistCat - the musical category of the artist e.g. Rock, Pop, Jazz etc.
artistBio - the artist biography.
artistImage - stores the artist image and its location directory as a string value.

Charts Entity

Attributes:
chartID - the chart id which uniquely identifies it.
albumID - foreign key reference to the albums table using the albumID of both tables.
artistID - foreign key reference to the artists table using the artistID of both tables.
chartyWeeks - lists the number of weeks a song has been on the charts.

Chapter 3: System Design

3.1 User Interface/Screen Designs

On the homepage, we will be implementing a menu bar with not more than 5 hyperlinks to different pages on it. Some hyperlinks included here will be the home page, a contact us page, a songs page and an artist page. As we are still in development, some of this could change. For our artist page, we will have many more page hyperlinks in a menu bar leading to the artists we decide to include in the website. When one of those hyperlinks are clicked it will bring the user to the page dedicated solely to that artist. On this page, there will be various details about the artist in question. We plan to implement some of these features in the album page also along with details about certain songs. The design and layout of the website is seen in the screenshots further down in the document. We also plan to implement the artist, the song, the rank and how many weeks the song or album has been in the charts. So far, we have implemented a homepage, multiple artist pages, multiple album pages and a music charts page. The music chart page list the current top 15 ranked music albums. As this is just a prototype, it is possible that there could be many changes to the way the website currently looks.

3.2 Wireframes

Index Page

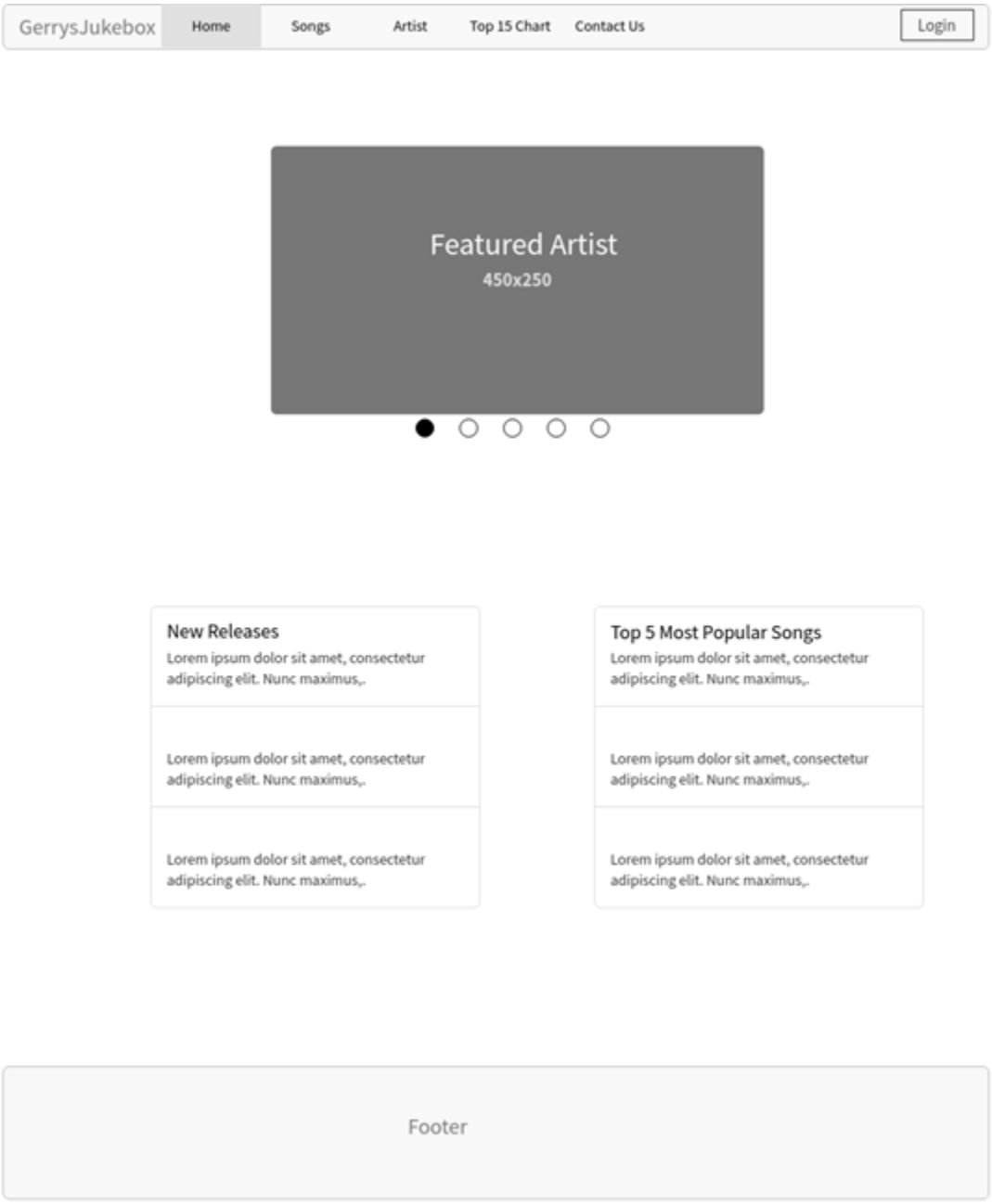


Figure 5 - Index Page Wireframe

Top 15 Chart

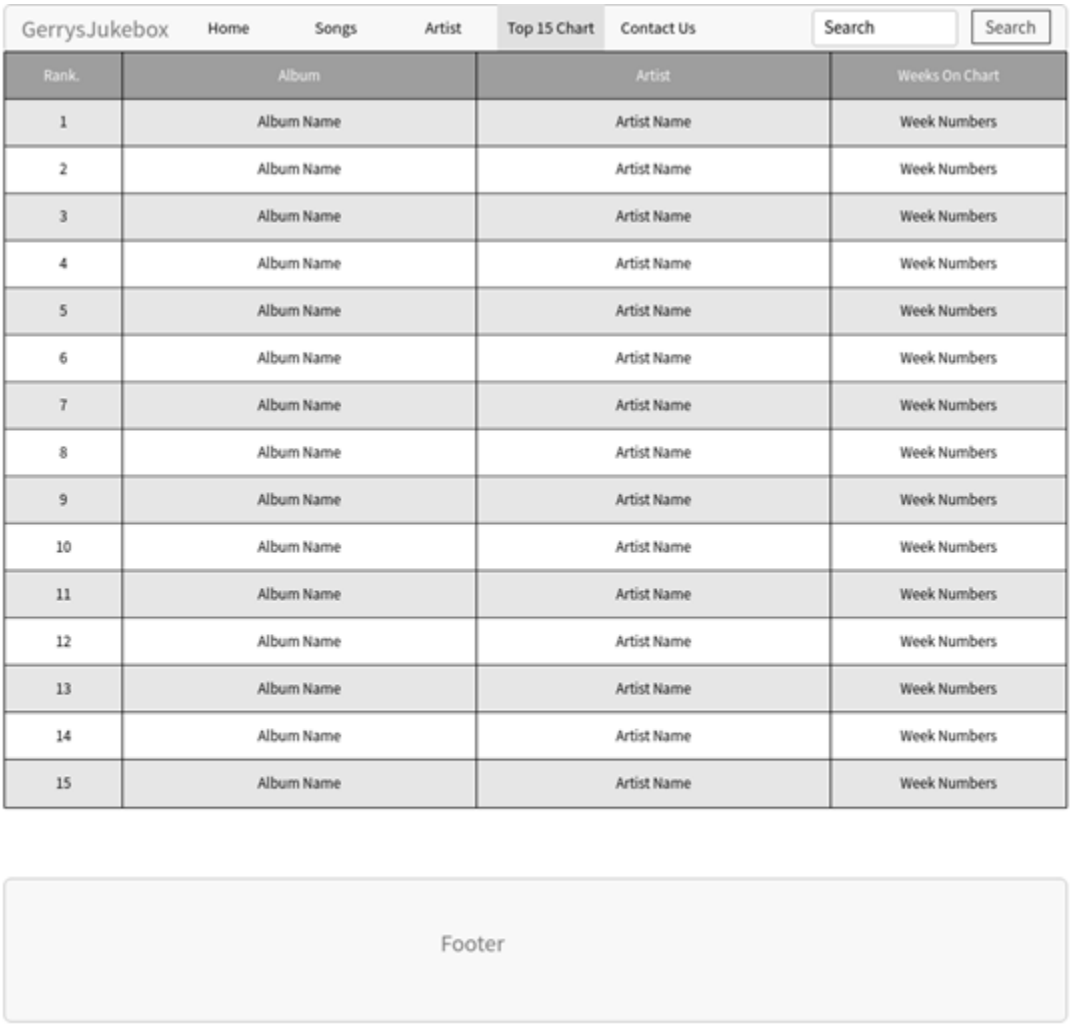


Figure 6 - Album Chart Page Wireframe

Songs Page

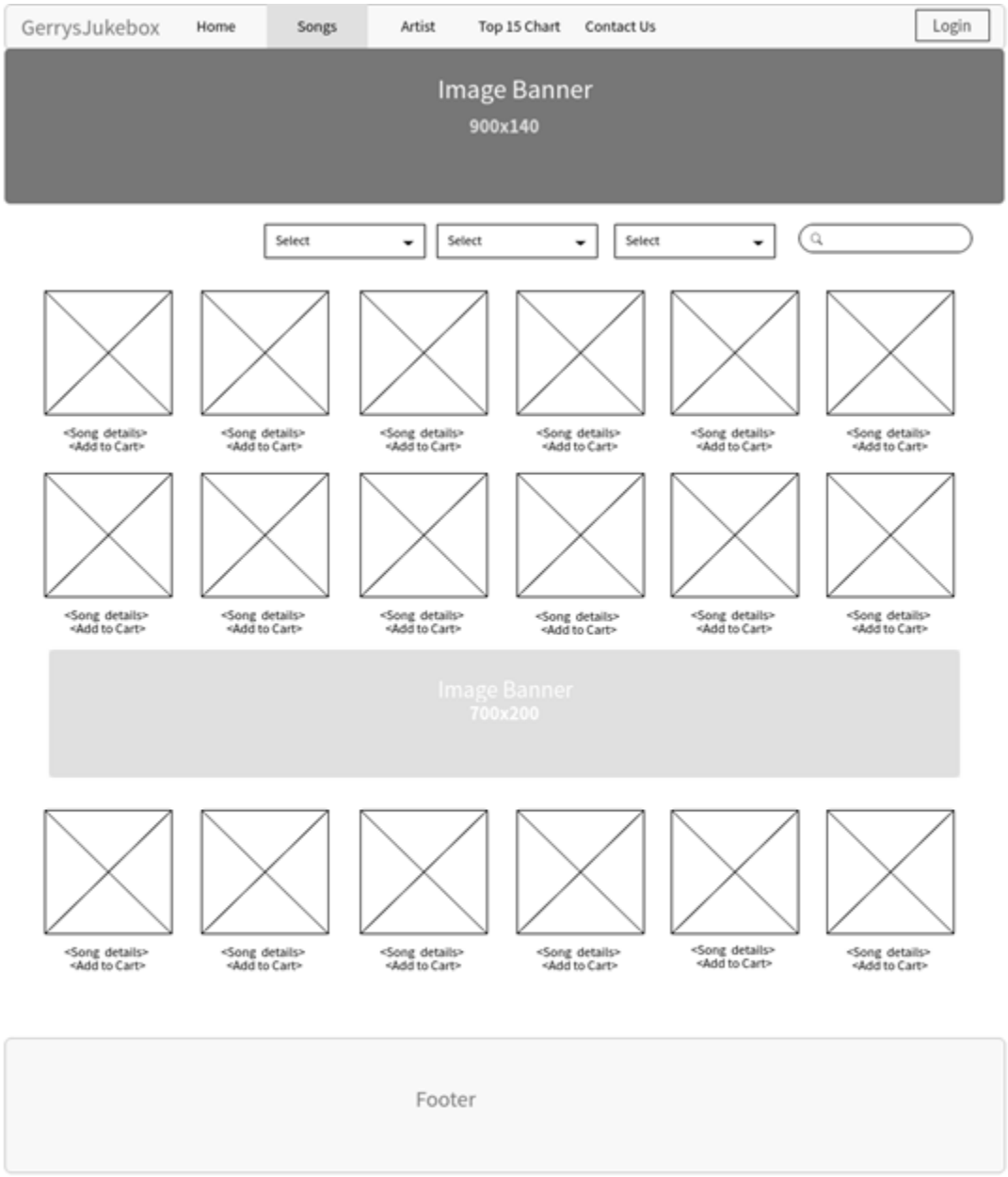


Figure 7 - Songs Page Wireframe

Contact Page

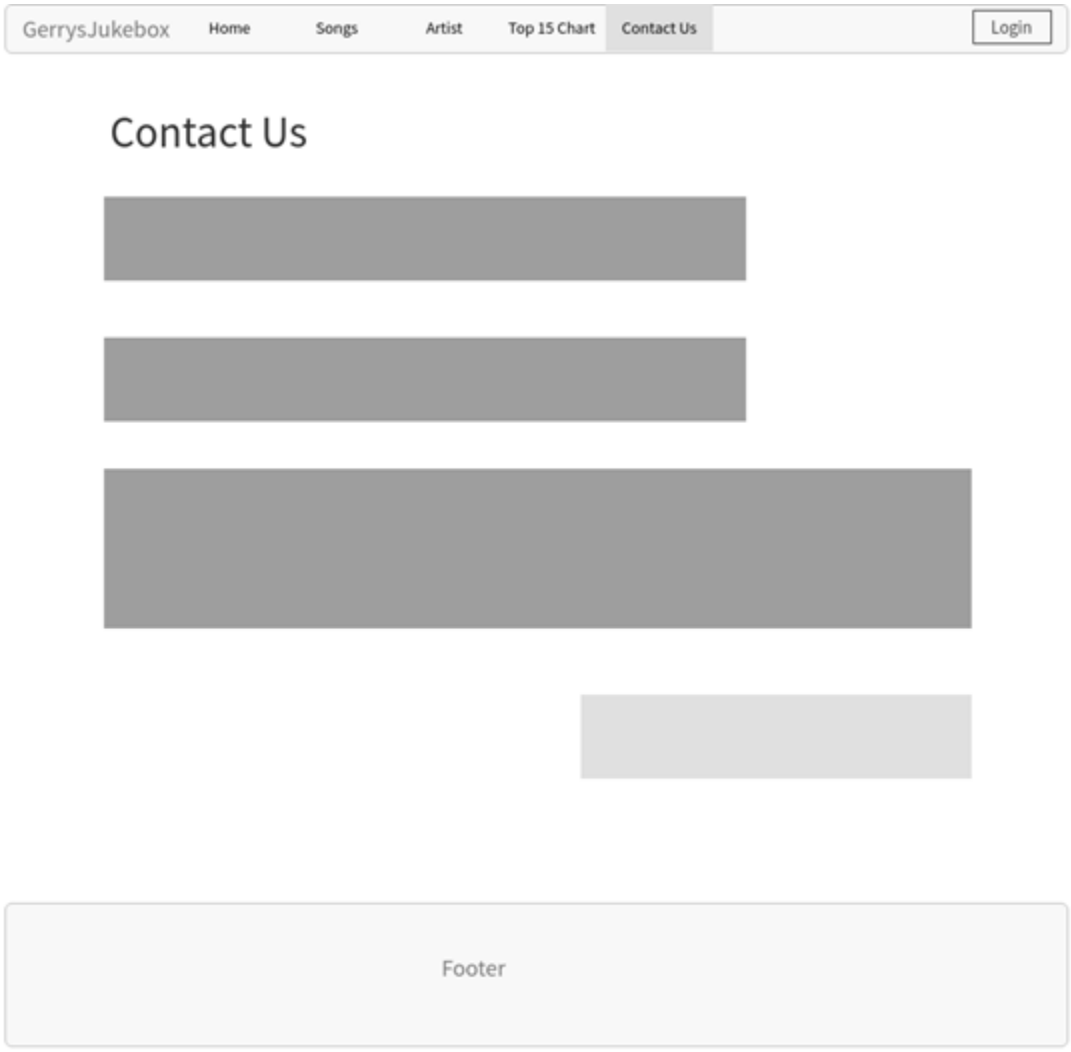


Figure 8 - Contact Page Wireframe

Artists Page

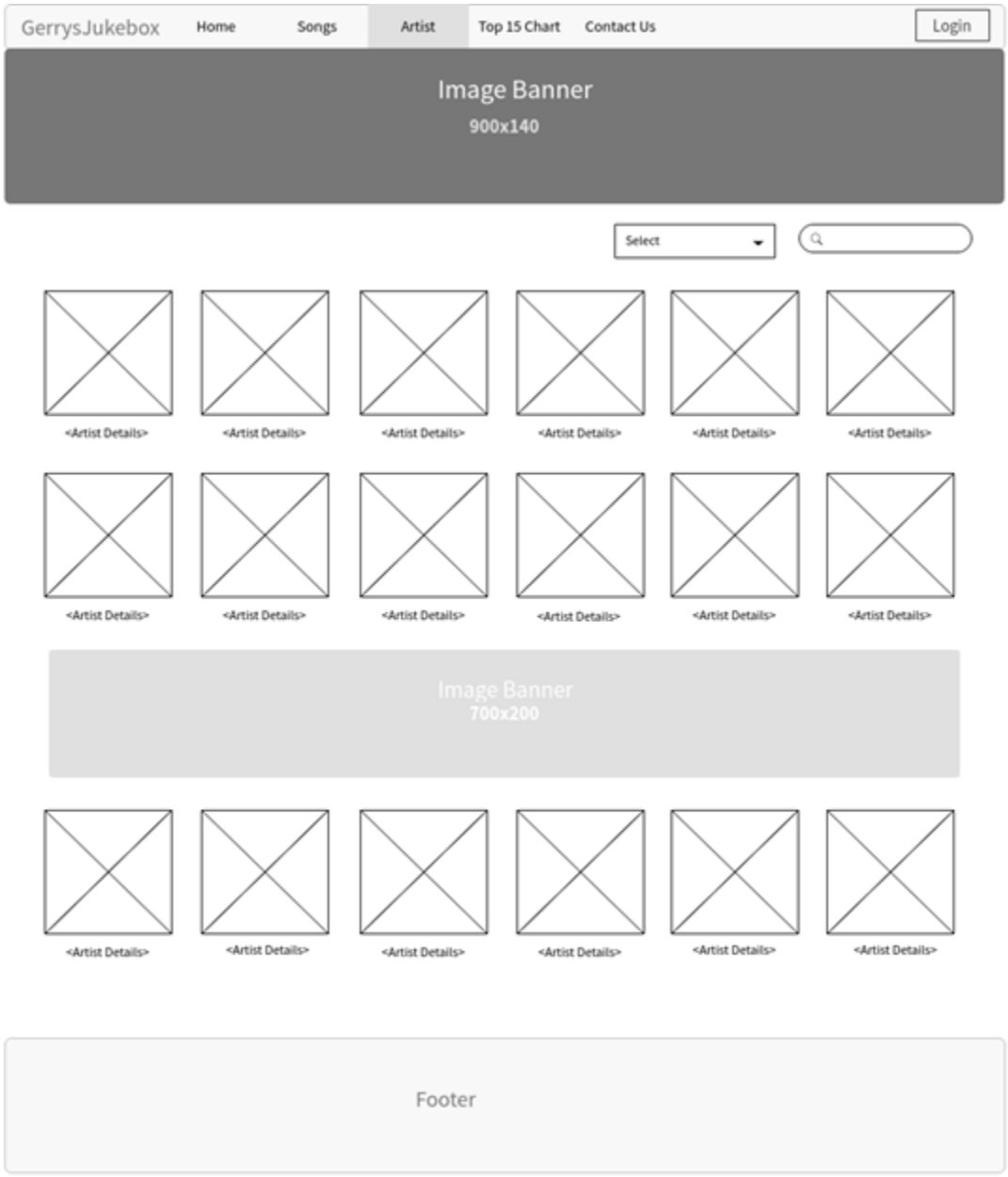


Figure 9 - Artists Page Wireframe

Artist Details Page

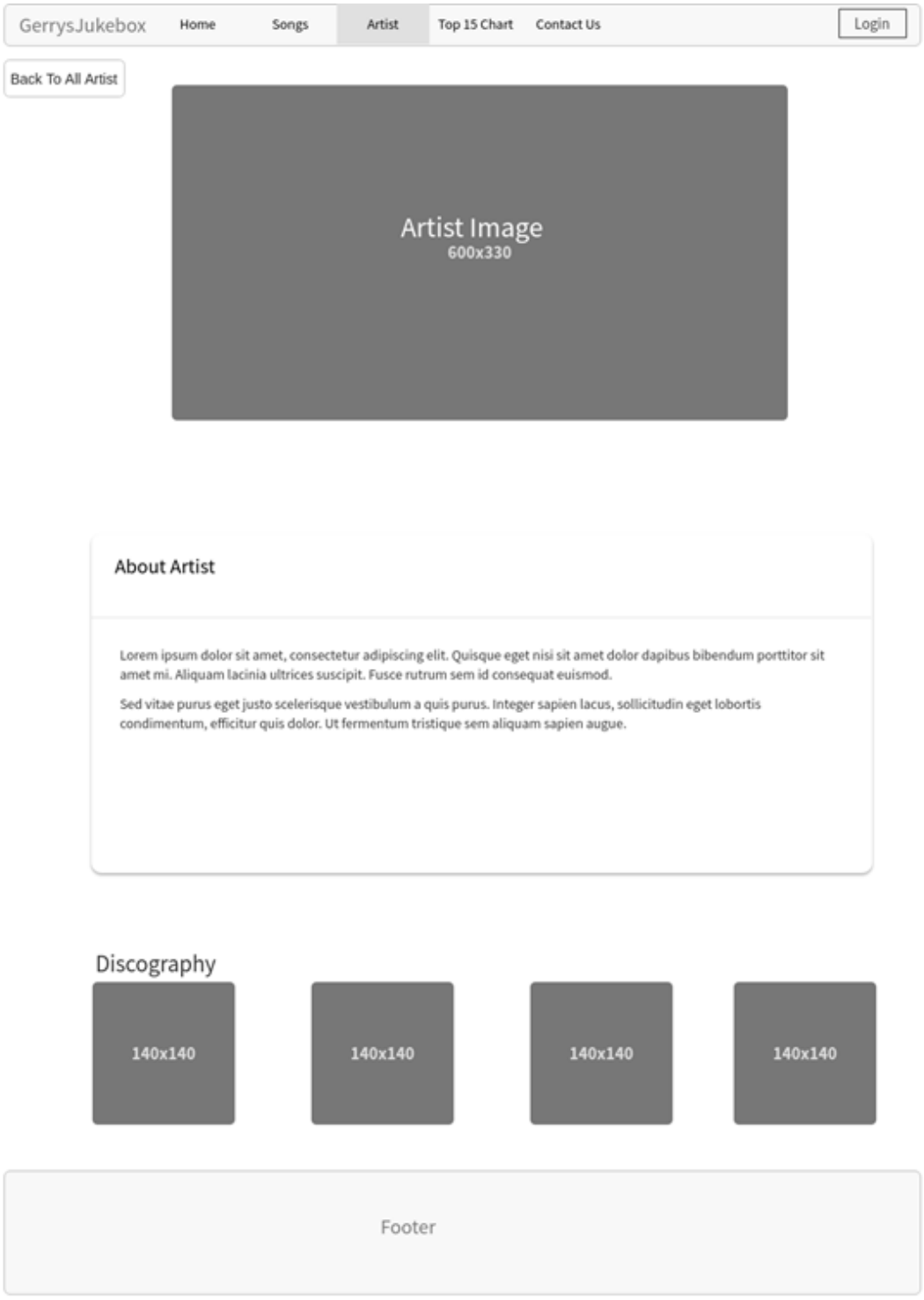


Figure 10 - Artist Details Page Wireframe

Album Details Page

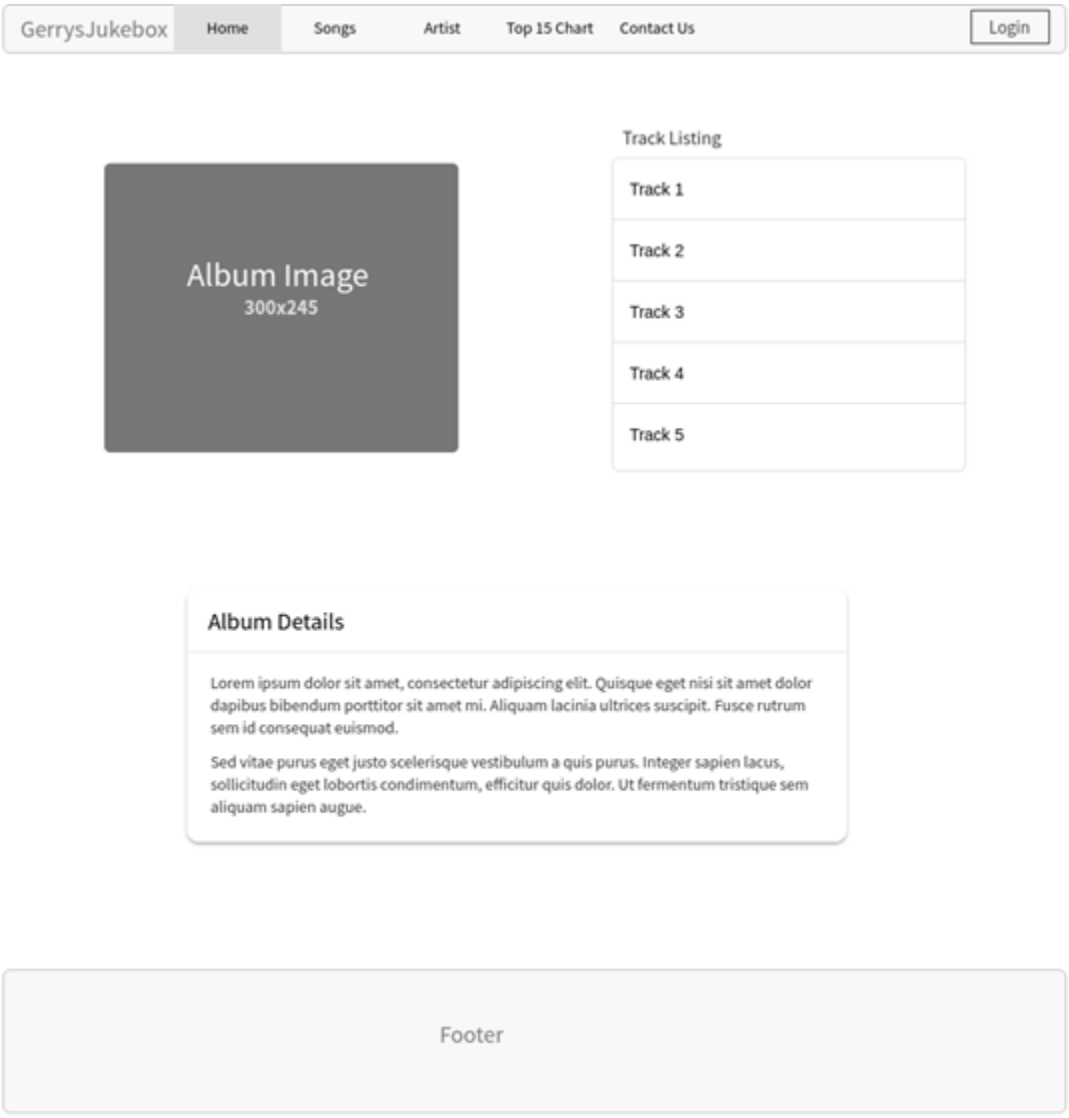


Figure 11 - Album Details Page Wireframe

3.3 Class Diagram

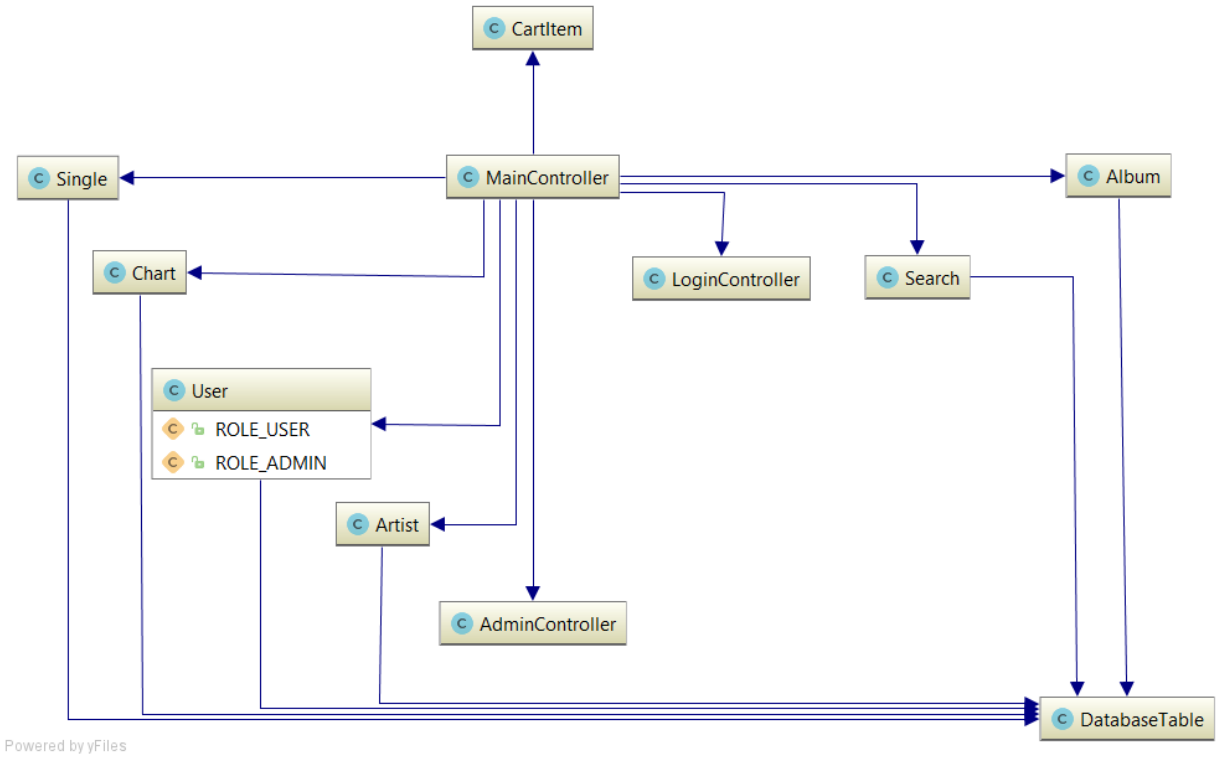


Figure 12 - Class Diagram

The class diagram above illustrates and explains the structure of the website, as well as the relationship between the objects in the code. It shows all the information about class members, methods, and attributes.

3.4 Interaction Diagram

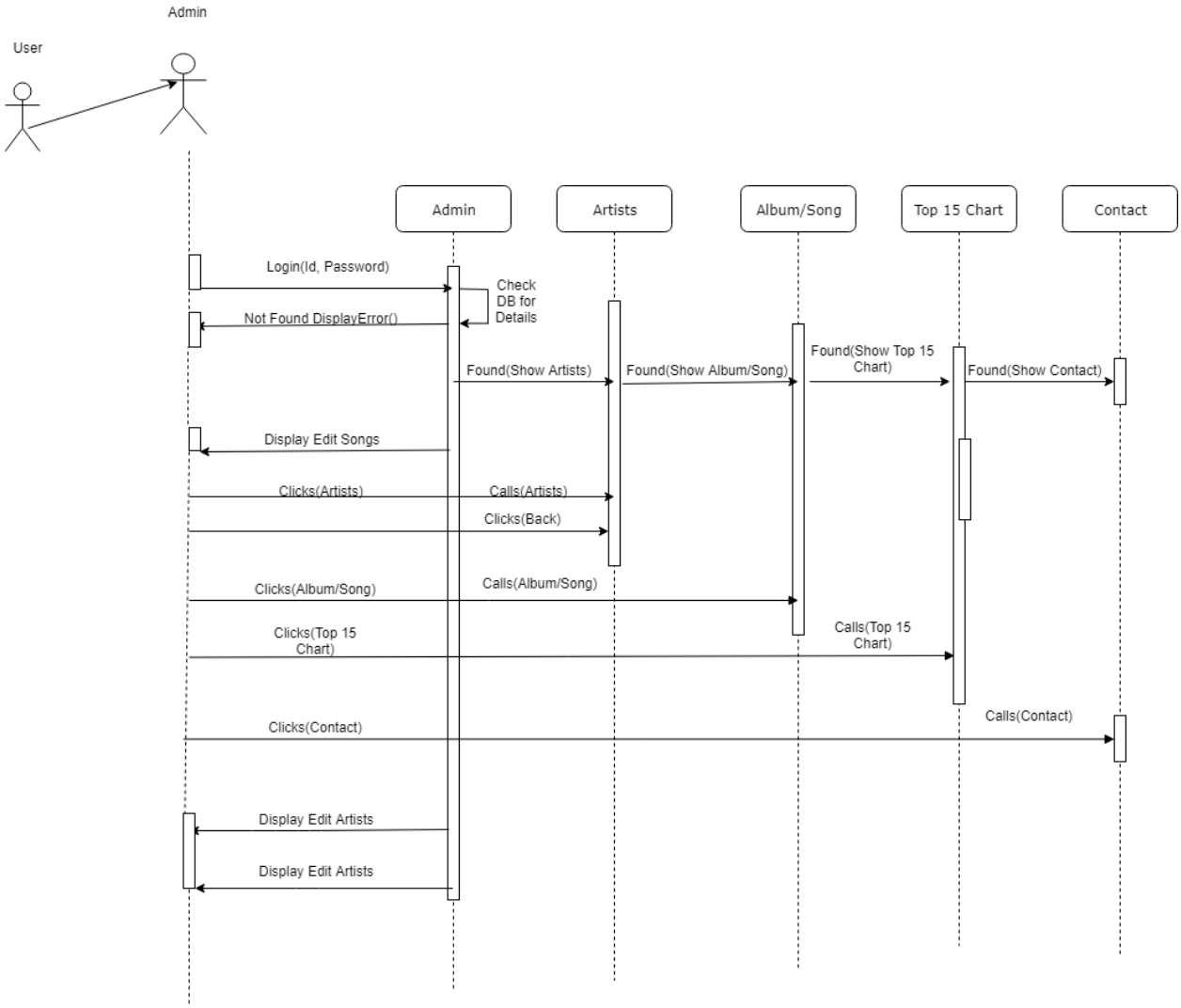


Figure 13 - Interaction Diagram

The diagram above (Interaction Diagram) shows how a group of objects interact with each other and also to an extent the messages being passed from object to object.

3.5 Normalised Database Tables

Normalised Form	1st Normal Form
PK User ID User Name User Role User Image User Email User Password Artist Id Artist Name Artist Image Artist Bio Artist Label Artist Category Album ID Album Name Album Image Album Release Album Video Album Price Song Id Song Track No. Song Name Song Writer Song Length Song Image Song Price Song Category Chart Id Chart Album Chart Artist Chart Weeks	PK Artist ID PK User ID Artist Name Artist Image Artist Bio Artist label Artist Category --- PK User ID User First Name User Surname User Role User Image User Email User Password Album ID Album Name Album Image Album Release Album Video Album Price Song Id Song Track No. Song Name Song Writer Song Length Song Image Song Price Song Category Chart Id Chart Album Chart Artist Chart Weeks

2nd Normal Form	3rd Normal Form
PK Artist ID Artist FirstName Artist Surname Artist Image Artist Bio Artist Label Artist Category	Artists Table PK Artist ID Artist FirstName Artist Surname Artist Image Artist Bio Artist Label Artist Category
PK User ID PK Artist ID	Users Table PK User ID User First Name User Surname User Role User Image User Email User Password *FK Album ID *FK Song ID *FK Chart ID
PK User ID User First Name User Surname User Role User Image User Email User Password Album ID Album Name Album Image Album Release Album Video Album Price Song Id Song Track No. Song Name Song Writer Song Length Song Image Song Price Song Category Chart Id Chart Album Chart Artist Chart Weeks	Album Table PK Album ID Album Name Album Image Album Release Album Video Album Price Song Table PK Song ID Song Track No. Song Name Song Writer Song Length Song Image Song Price Song Category Chart Table PK Chart ID Chart Album Chart Artist Chart Weeks

Chapter 4: Implementation of the System

4.1 Introduction

In this chapter, we will be discussing the implementation of the system. The implementation stage of the project will be broken down into four sections. We will first look at some technical information – this includes system and software design decisions taken and agreed upon by the group, which will be broken down further, looking into project management, design quality and program reliability. We will then dive into the implementation of the prototype and the final build, describing the technologies used and how they were implemented throughout the duration of this project. Finally, this chapter will go into detail about the information about the proposed solutions in earlier chapters and how they were implemented into the final build.

4.2 Technical Information

4.2.1 System and Software Design

The system was developed using the waterfall model. The waterfall model was first used in 1970, introduced by Dr. Winston W. Royce. This software development process model emphasizes that a logical progression of steps must be taken throughout the software development life cycle [3]. We decided that the waterfall model would be the best software development model to use for this project, as we felt it suited our beginning, middle and end goals.

The main advantage of using the waterfall model is that implementing it is a simple task, this is mainly due to it being a simple step-by-step model. The six stages of the model are; Requirements, Analysis, Design, Coding, Testing and Operations.

Using the waterfall model allowed us to make early design changes from the early prototype of the website to the eventual live version. We will discuss, in more detail about the prototype design in a later section of this chapter.

The overall system architecture and walkthrough is shown in fig 14 below.

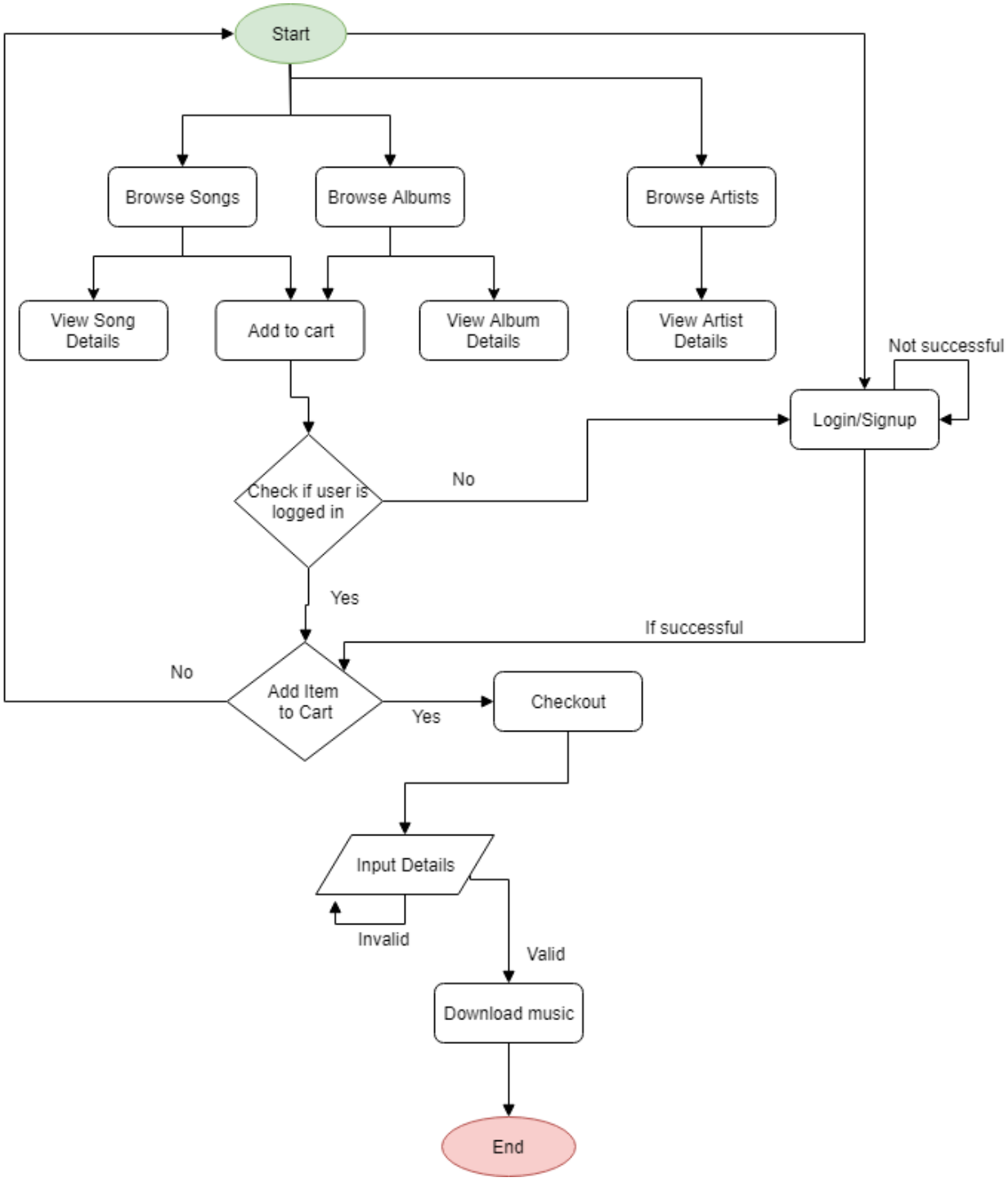
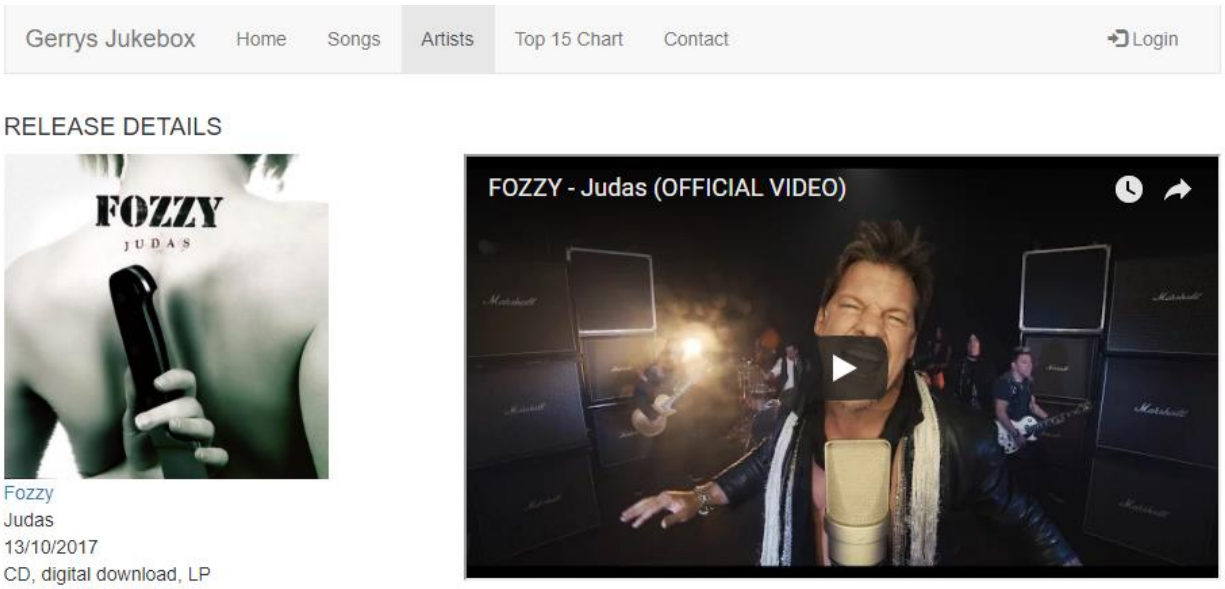


Figure 14 - System Architecture

4.3 Implementation of Prototype

4.3.1 Prototype design

At the beginning of the project, we constructed a simple prototype of the website, in what we hoped would achieve an understanding of what the overall end goal and design of the site would be. The initial prototype was somewhat similar to the wireframes we previously looked at, with slight differences being shown in the album details page.



TRACK LISTING

#	Title	Writer(s)	Length
1	Judas	Rich Ward, Johnny Andrews, Justin Cordle	4:10
2	Drinkin' With Jesus	Rich Ward, Johnny Andrews	3:56
3	Painless	Rich Ward, Johnny Andrews	4:00
4	Weight of My World	Rich Ward, Johnny Andrews	3:18
5	Wordsworth Way	Chris Jericho, Rich Ward, Johnny Andrews	4:48
6	Burn Me Out	Rich Ward, Johnny Andrews	4:04

Figure 15 - Album Details Page

As pictured above in fig 15, instead of the track listing being on the right-hand side of the page, we opted to move it to the bottom. We felt this would give it a cleaner look and feel rather than being bunched up, as this gives the user a clear indication of the title, writer(s) and track length. As we had no option to

preview a sample of the band's music, we decided it would be best to include a music video from the bands single from that album, again, this was mainly due to give the user the optimal experience, as this allows them to listen to the full song rather than a 30 second sound clip.

For the rest of the prototype, we kept the design as close as possible to the wireframes. This is evident with the home page as seen in fig 16 and 17 below.

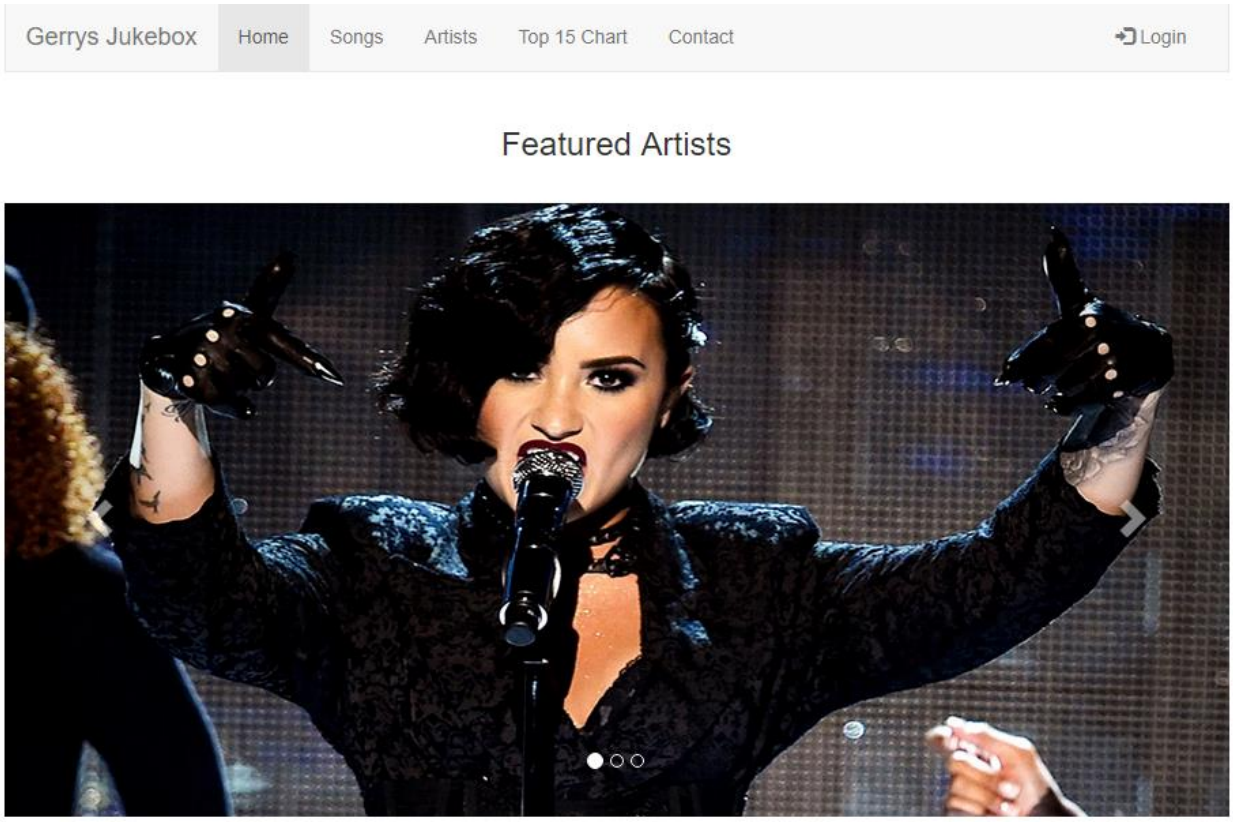


Figure 16 - Home Page Prototype

Figure 17 - Home Page Prototype (continued)

4.3.2 Technologies used for prototype

For the implementation of our prototype, as a group, we agreed that we would have a rather simple website with basic functionality. This was done so we could put together a quick mashup of what we eventually wanted the site to do functionality wise. The prototype was made using Sublime Text 3. Sublime Text is a sophisticated text editor for code, markup, and prose. The reason we chose to code the prototype with Sublime Text was due to its simplistic nature and slick user interface made it our go-to editor over the likes of NotePad and NotePad++. In fig 18, you can see a sample of the code used to create the prototype website.

```
<nav class="navbar navbar-default">
  <div class="container-fluid">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#myNavbar">
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
        <span class="icon-bar"></span>
      </button>
      <a class="navbar-brand" href="#">Gerrys Jukebox</a>
    </div>
    <div class="collapse navbar-collapse" id="myNavbar">
      <ul class="nav navbar-nav">
        <li><a href="../../index.html">Home</a></li>
        <li><a href="#">Songs</a></li>
        <li class="active"><a href="#">Artists</a></li>
        <li><a href="../../chart.html">Top 15 Chart</a></li>
        <li><a href="#">Contact</a></li>
      </ul>
      <ul class="nav navbar-nav navbar-right">
        <li><a href="#"><span class="glyphicon glyphicon-log-in"></span> Login</a></li>
      </ul>
    </div>
  </div>
</nav>
 <br>
<h4>RELEASE DETAILS</h4>
<img src="./images/albums/ballad.png" alt="An image of The Ballad of Tom and Cindy album cover" height="240" width="240"></img>
<iframe width="560" height="315" align="right" src="https://www.youtube.com/embed/cO13lv-mXbQ" frameborder="2" allowfullscreen align=
"right"></iframe>
<!--- ************ -->
```

Figure 18 - Album Details Code Snippet

4.4 Implementation of Final Build

4.4.1 Final build design

With the final build of the project, we wanted to keep it as close to the wireframes and prototype as possible but improve upon them when and where we could. This is most evident with the home page. Originally, we wanted to keep it in line with the prototype, however, after some consideration and after discussing it with our supervisor, we thought it was best to completely revamp the home page. We started by simply having all the albums available on the site scroll through a slideshow, similar to how the featured artists worked on the original incarnation of the home page, however, after heavy discussions and a quick demo with an outside party, we began to think from the perspective of a first-time user. As this was our creation, we knew what to do and where and what it actually is that the website does, but when a first-time user sat down to test the website, they had trouble figuring out what it was and how to do things. With this in mind, we began constructing, what we believe, to be a friendly interface for first time, and potentially all, users visiting the website.

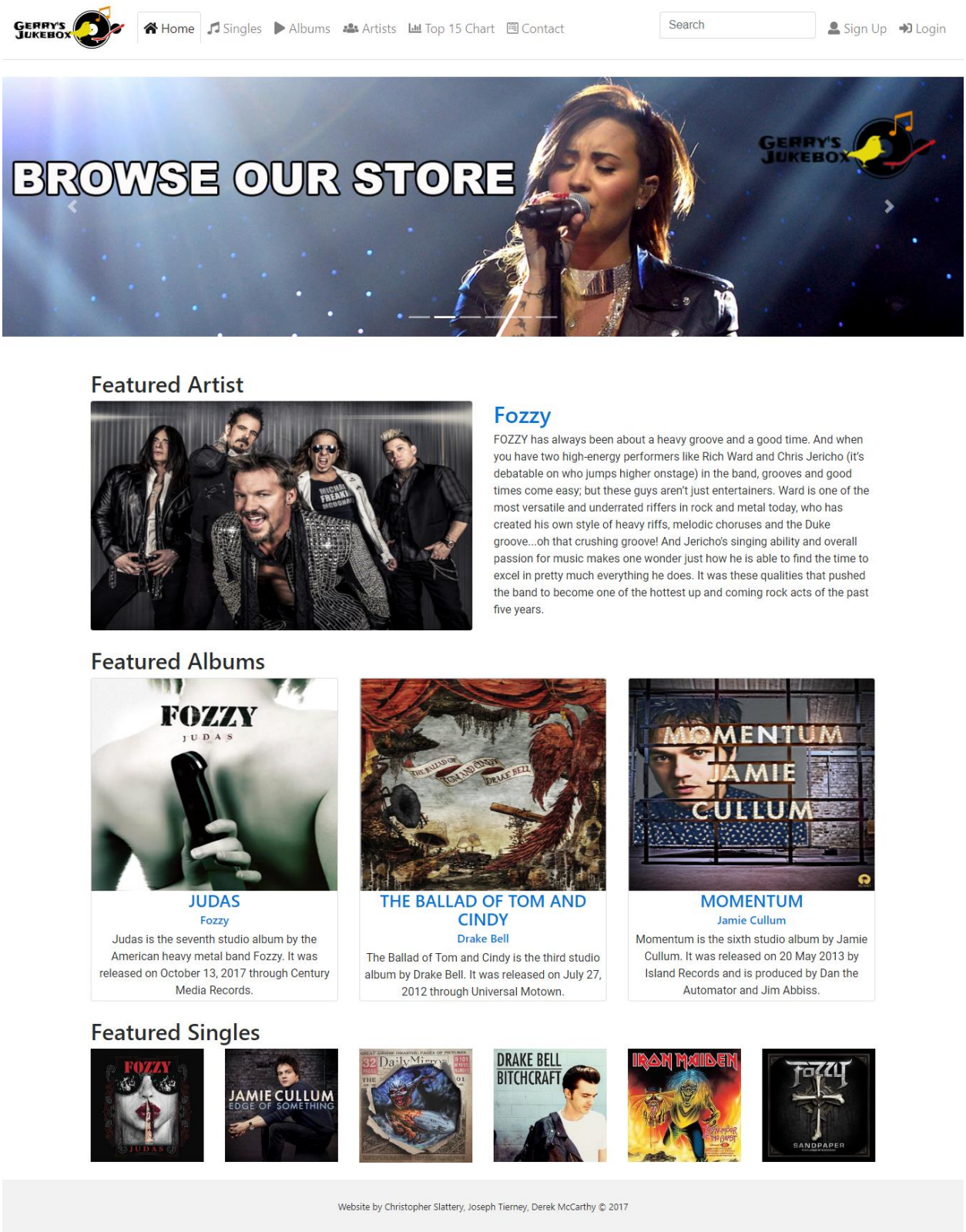


Figure 19 - Home Page

In fig 19 above, you can see the newly designed home page. To start, we kept the slideshow design from previous versions of the project, however, we added banners which clearly indicate to users what they are able to do when visiting the site. These range from browsing the store (as seen in the above fig 19), discovering new music, viewing the top 15 chart, getting in contact with the site administrators and signing up. All of these banners are clickable, giving the user ease of access to each page on the site. We decided to keep the featured artist section from the prototype but only limit it to one artist, with the possibility of updating it in the future as needs be. As an extension to the featured artist section of the home page, we added featured albums and featured singles in order to give the home page that "complete" look and feel of a music website. Based on surveys we carried out with users, the home page was a success and gave users a clear idea of what it is they are supposed to be doing when they visit the site.

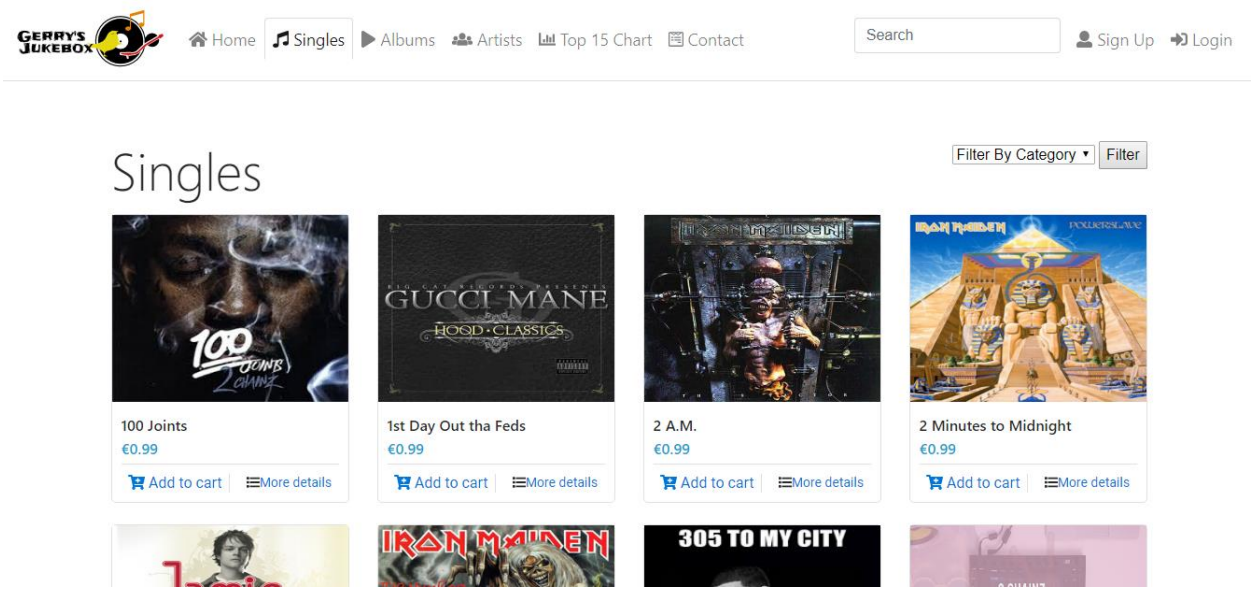


Figure 20 - Singles Page

In fig 20, every song we have stored in the database is displayed, along with an image of the album that song is from. Users are able to buy multiple songs by clicking "Add to cart" and are able to view more details of the song (fig 21).

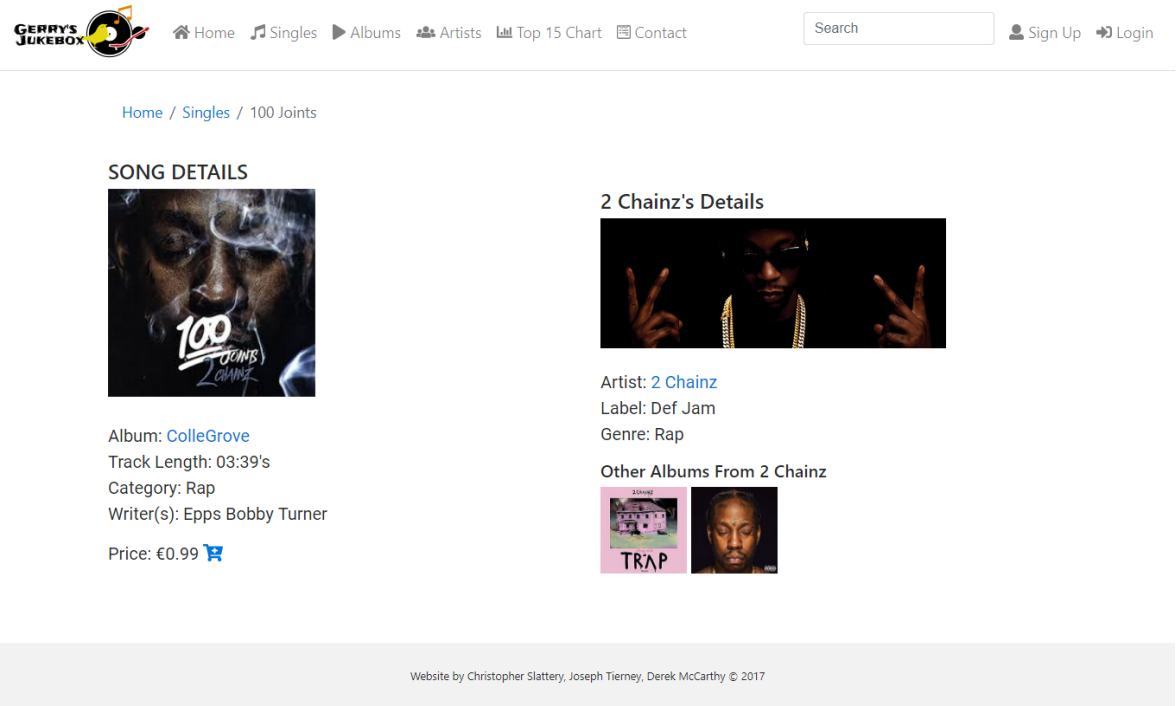


Figure 21 - Song Details Page

In some cases, songs will have a different picture from the album, as these are singles released ahead of the album to promote it and usually come with their own album cover, an example of this can be seen below in fig 22.

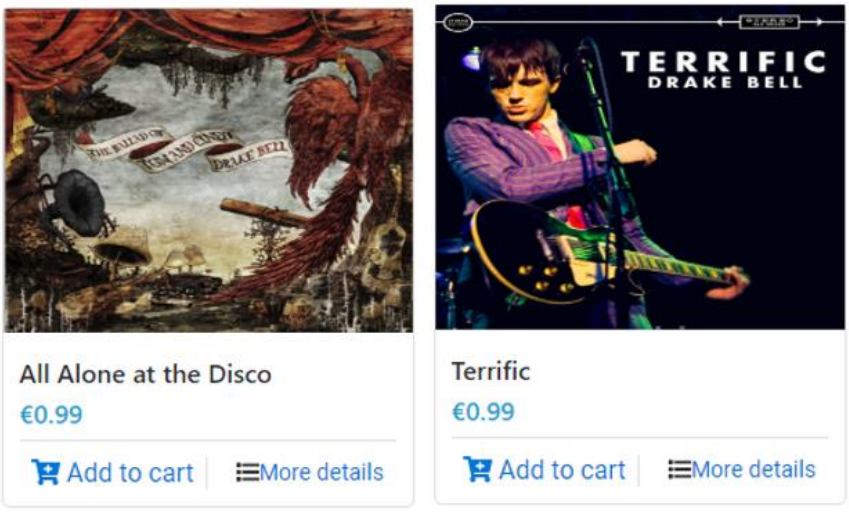


Figure 22 – Song Images from Album

Both of these songs can be found on the album "The Ballad of Tom and Cindy", as seen in fig 23.

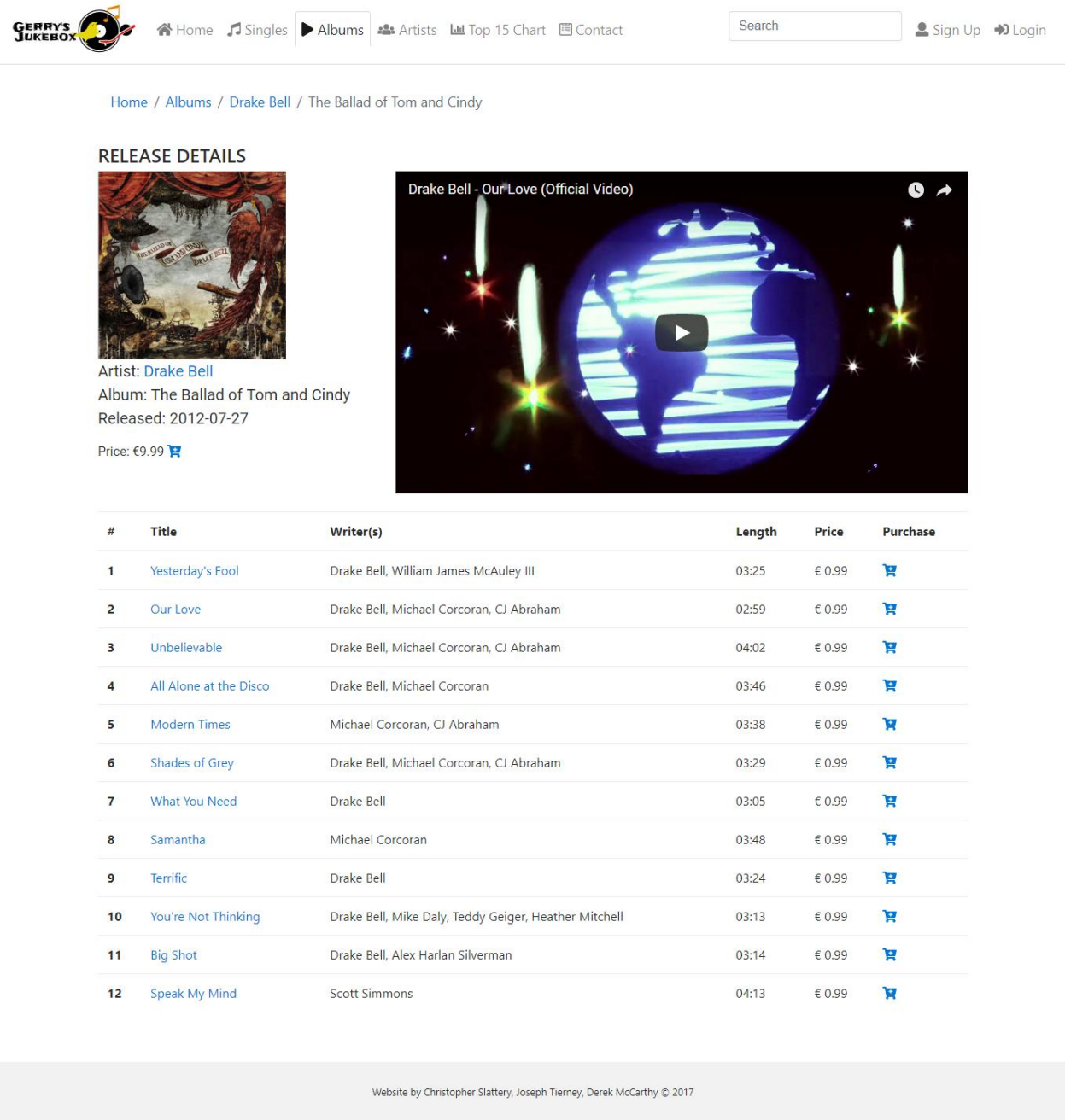


Figure 23 - The Ballad of Tom and Cindy Album Page

The artist page (fig 24 below) displays every artist stored in the database. The user will see an image of the artists, their name and what record label they are currently signed too.

Image	Artist	Label
	Fozzy	Megaforce
	Jamie Cullum	Deckdisc
	Iron Maiden	EMI
	Drake Bell	Motown

Figure 24 - Artist Page

Clicking either their image or name will bring up the artist details page, seen in the fig 25 below.

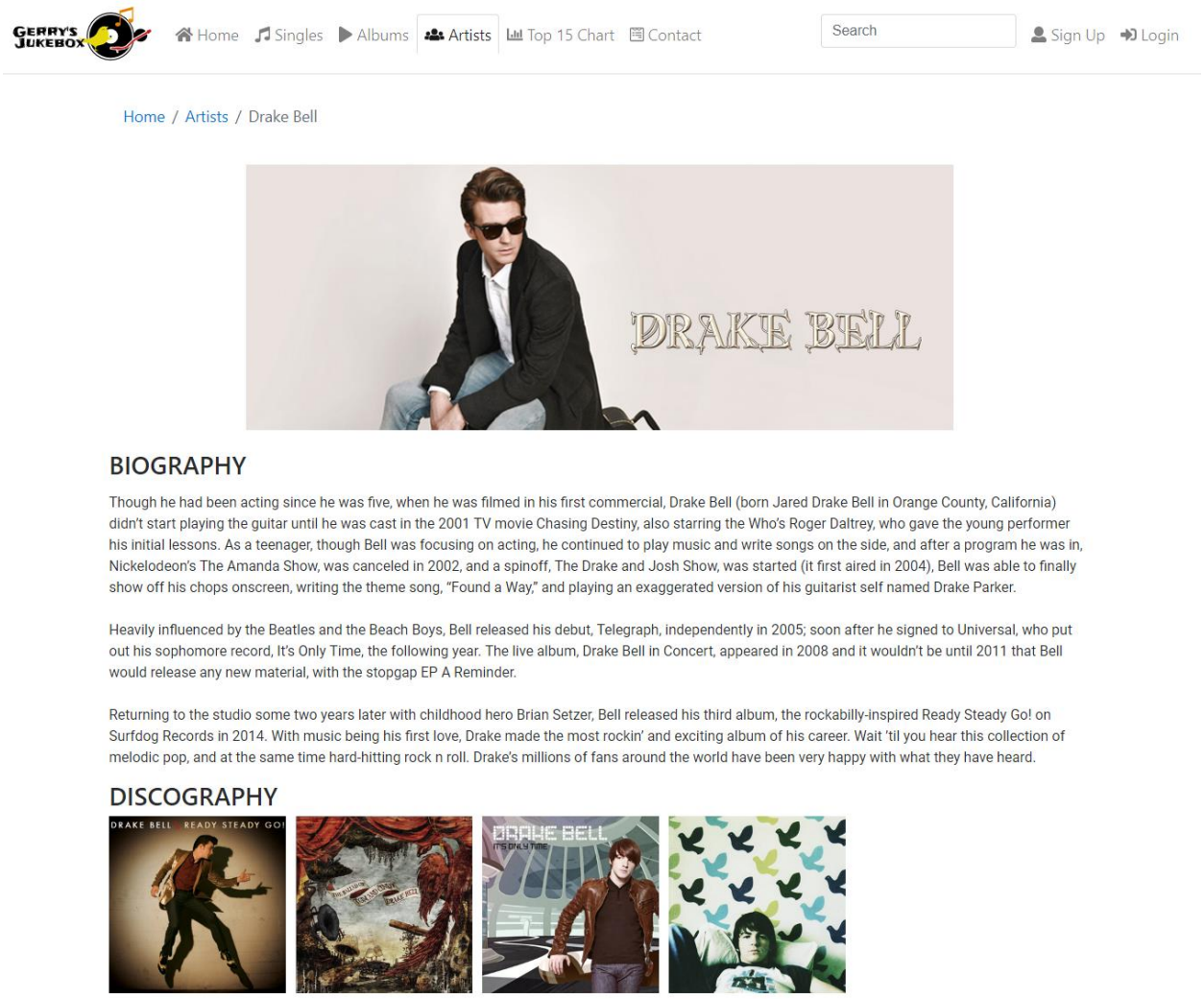


Figure 25 - Drake Bell Details Page

4.4.2 Technologies used for final build

For the implementation of our final build of the project, we used a variety of technologies in order to achieve the end result we wanted. For the coding, we used JetBrains PhpStorm IDE. We decided to use this as we felt it was necessary for the functionality of the website, and it played a part in the validity testing of the final product. Many of the functions included in the final website would have been near impossible to implement if we continued using a simple text editor such as Sublime or NotePad++, so making the transition over to PhpStorm was really an obvious choice that we all agreed on. PhpStorm allowed us to use many functions and imports, such as the Add to cart and login features that wouldn't have worked otherwise. PhpStorm also allowed us to easily implement a database, so that we could store

all the artist information and user information and call on it when needed for the website. Below is a fragment of code used to call the database, displaying the albums by the artists ID. To further explain, Fozzy have the artist ID of 1, therefore, on the albums page, their albums will be displayed first. This is just one of many examples as to why we used PhpStorm.

```php
public static function getAlbumsByArtistID($artistID)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();

    $sql = 'SELECT * FROM albums WHERE artistID=:artistID';
    $statement = $connection->prepare($sql);
    $statement->bindParam(':artistID', $artistID, \PDO::PARAM_STR);
    $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
    $statement->execute();

    if ($object = $statement->fetchAll()) {
        return $object;
    } else {
        return null;
    }
}
```

Figure 26 - SQL Query in PHP to Return an Artists Album

For the database, we used MySQL Workbench. We chose MySQL Workbench over other database software tools like Microsoft Access or PHP MyAdmin as we are more familiar with Workbench and how to properly implement it with PhpStorm for a working website. For editing the banners on the home page, we used Adobe Photoshop.

Chapter 5: Testing and Evaluation

In this chapter we will be discussing the results from our testing and evaluation of the website. For this we have divided the tasks into six different phases. For these phases we will be testing for Functionality, Usability. Interface, Compatibility, Performance and Security.

5.1 Functional Testing

As part of the functionality testing we will be testing all links on the website, adding music to the shopping cart, confirming the purchase, database connections as well as checking all forms used for submitting data.

5.1.1 Testing Links

To thoroughly test all active links on the website we carried out the following tests,

- Testing links to and from all internal pages.
- Testing links jumping to the same page, which involved checking the link which brought you to the webpage, to ensure it just reloaded the page you are on.
- Testing links for orphan/dead end pages, which are pages with no direct reference to other pages on the website.
- Testing for broken links.

Having extensively carried out these tests a number of times in different browsers we did not find any issues with and links.

5.1.2 Testing Forms

As forms are central to the backend/server side of our website for passing information we conducted the following tests,

- Tested all input fields for validation. This included testing inputs would not accept empty inputs or fields which required a certain type of data. The types of data include numeric and date types in input fields.
- Tested form submissions for the validity of data received to the data sent in the form submission.
- Tested input fields for validation such as, passwords should contain at least one uppercase and one lowercase letter and be of length 8+.
- Tested all admin forms for creating, updating, and deleting of albums, artists, and songs.
- Tested the default values in the input fields of the admin update forms were correct.

During the form testing we encountered a number of problems with the forms on the website. These are the problems we found and the solutions to fixing them,

- On the user profile page, the change password form had no validation on the password input field. Which allowed users to change their password to an empty string. To address this issue, we added a validation rule which required passwords to contain at least 8 characters with at least one uppercase and one lowercase letter and at least one number.
- The signup form had no validation on the first and surname name. To fix this we added a validation rule which required the names to contain 3 or more characters.
- The admin forms for adding and updating artists, albums and songs had no validation. We fixed this by adding validation to all input fields on these forms such as the date input field would only accept dates, the track number input field would only accept numbers etc.
- The default input field on the artist update page contained a misspelled word, we addressed this issue by correcting the spelling.

After documenting our findings from our form testing and addressing the issues we had discovered. We then re-tested all the forms on the website and found no issues with these forms.

5.1.3 Validity of HTML and CSS

For the validation testing of code, we used two different methods. The first method was using the PhpStorm IDE built in html validator which once enabled highlights unused tags and brackets. And the other method was using the W3C [2] online validator which is used by web developers for finding syntax errors in code.

During the validity testing we first fixed issues we had with missing or unnecessary syntax in PhpStorm. Some of the issues we had in PhpStorm are highlighted in fig 27.

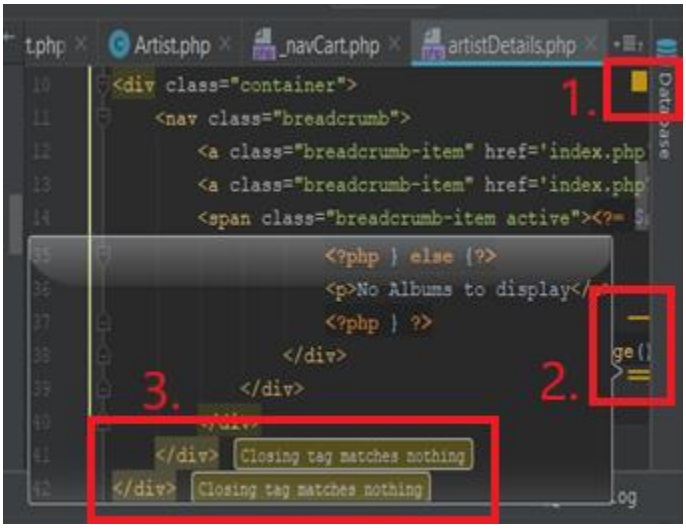


Figure 27 - PhpStorm Validity Testing Errors

From fig 27 we can see the validation taking place in PhpStorm. There are three items of interest we've highlighted with red rectangles and number. The first item is of an amber box which indicates there's

some issue or issues with syntax on the given web page. Next, we have the three amber lines which indicate there are three issues with two of them being highlighted by the third red rectangle. The two issues highlighted were of two unnecessary tags while the third issue was caused because there was no space between two attributes within a tag.

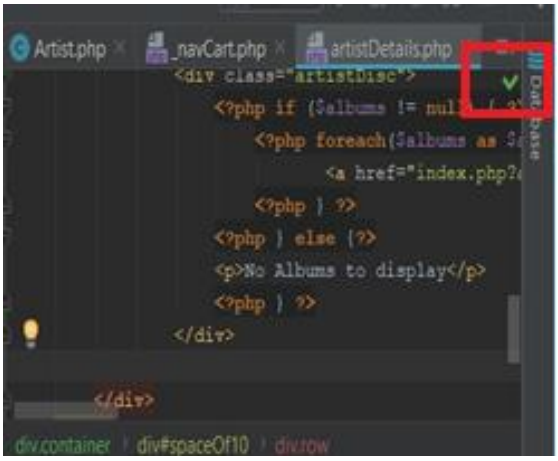


Figure 28 - PhpStorm Validity Testing Errors Fixed

From this image fig 28 we can see the amber issue indicator from the previous image has now being replaced by a green tick. This was as a result of removing the two unnecessary tags and adding a space between the attributes. We have carried out the same test on all pages within our website.

The other method for validating our html was using the W3C online validator. The W3C online validator is use in industry for syntax validation. There are a number of ways in which you can use the validator tool. The first is by using you Uniform Resource Locator (URI) address. The second way is by uploading your html file and finally the last method and the one we use is by copying and pasting your html into the provided textbox.

Figure 29 - Syntax Errors

As you can see from fig 29 the W3C validator highlights your errors and gives you a good explanation on what caused the error. Once the error has being rectified and the validation re-run you should she no errors.

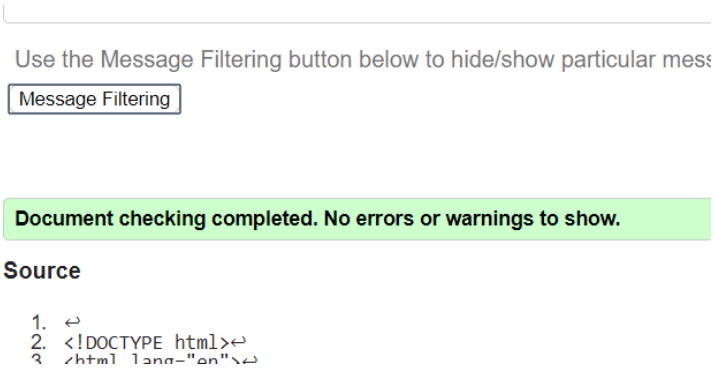


Figure 30 - Syntax Validation

This fig 30 is the result of the three previous errors being fixed.

5.1.4 Database Testing

For the database testing we did two types of testing. The first test was to test the data mappings between our user interface (UI) and the database. The second test was to test the data integrity between the UI and the database.

Data mapping

To test the integrity of data mappings of our website we tested that the fields in our UI matched those of our database and where of the correct data type and size. We carried out a number of these tests by entering both valid and invalid data. For the valid data we compared the data in our database to that of the data we had just created, modified, or deleted to ensure it matched. To test the invalid data, we used the wrong data types in the forms of the website such as entering letters in a number field and entering data that was larger than what we had defined in the database field. If the mappings are correct the entered data should be rejected, and no changes should take place in our tests it was rejected. During our testing both the valid and invalid data tests passed.

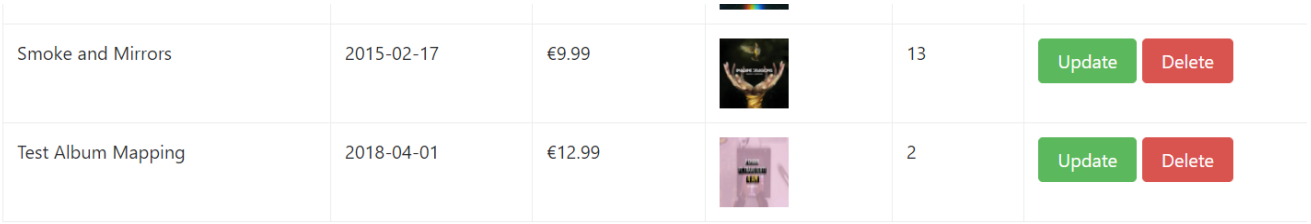


Figure 31 - Data Mapping of New Album

In this fig 31 we are testing data mapping of adding a new album to the database.

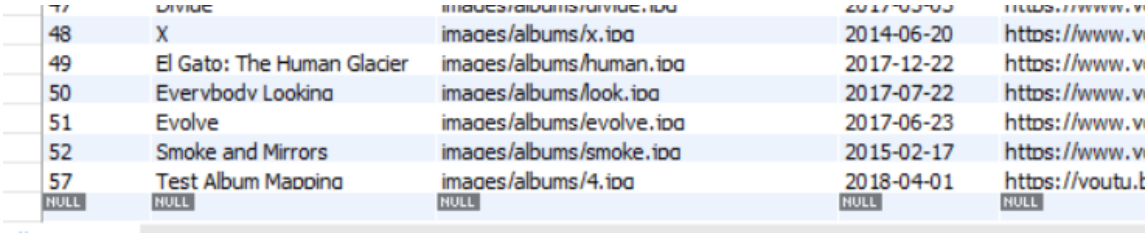


Figure 32 - New Album Viewed in Database

As you can see from fig 32 the data mapping is correct. We also carried out the same test's for adding new songs and artists.

Data Integrity

For the testing of the data integrity we carried out the same tests as we had in the previous test. Unlike using the UI of the website in the previous test we directly ran our tests in SQL.

#	Time	Action	Message	Duration / Fetch
11	03:40:29	insert into artists values(null,null,'af','adfsf','fassss...	Error Code: 1048. Column 'artistName' cannot be...	0.000 sec
12	03:41:28	insert into artists values('safsaf',null,'adfsf','fassss...	Error Code: 1136. Column count doesn't match v...	0.016 sec
13	03:41:56	insert into artists values('sdfs','safsaf',null,'adfsf','f...	Error Code: 1366. Incorrect integer value: 'sdfs' f...	0.000 sec
14	03:42:29	insert into artists values(null,null,'safsaf','adfsf','fa...	Error Code: 1048. Column 'artistName' cannot be...	0.000 sec
15	03:42:48	insert into artists values(null,'sf',null,'adfsf','fassss...	Error Code: 1048. Column 'artistImage' cannot be...	0.000 sec
16	03:43:06	insert into artists values(null,'sf','fsd',null,'fasssss...	Error Code: 1048. Column 'artistBio' cannot be null	0.000 sec

Figure 33 - Data Integrity Testing in MySQL

As you can see from our results in fig 33 the invalid data we tested for was rejected. Whereas the valid data was accepted by our database thus not affecting the data integrity of the database.

5.2 User Experience (UX) Usability Testing

For the UX testing we used 15 different people to test the website. What we were hoping to achieve from this testing was, how someone with no previous knowledge of our website managed to navigate, how easy/difficult they found the website to use and their experience of using the website. For the test cases we formulated a plan in which none of the testers were to be given access to the website before the test or given instructions on how to use the website. For the test we gave each tester access to the live version of the website and asked them to fill in a questionnaire (see Appendix B) after they had finished using the website.

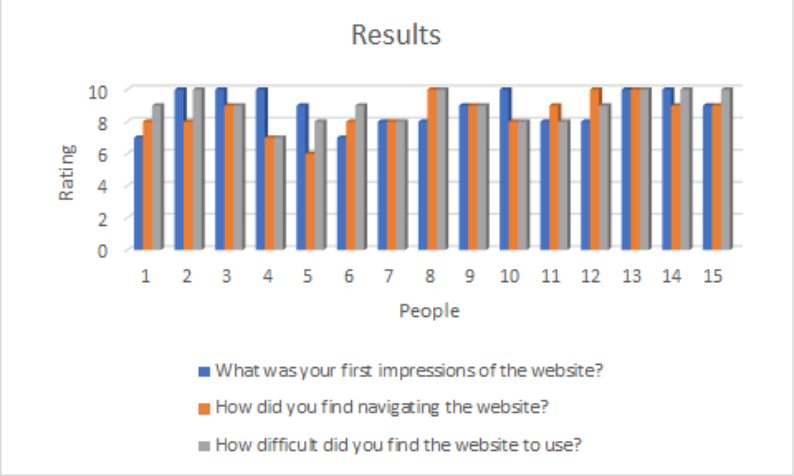


Figure 34 - Results from Our User Survey

In fig 34 we can see the results of the questions and its ratings by each user. As you can see the vast majority users enjoyed using our website based on the ratings they gave. As for the total rating figures from the questionnaire for the question about the first impressions of the website we had an 87% rating. For the question about navigating the website we had an 85% rating and finally for the question about, how difficult the website was to use we had an 89% rating overall.

As for the two yes or no answer questions from the questionnaire all 15-people testing the website said they would recommend the website to a friend. For the last question of the questionnaire about improvements to the website 7 people answered no to the question and 8 people said we could make improvements. The improvements that was recommended that could be improved upon were,

- Don't show all songs and albums on the one page. This issue we didn't address as we didn't have the time to implement but, in the future, we would could add page pagination were only a select number of items would be displayed per page.
- Have a previous purchase page where you could see a list of all albums and songs that have being previously purchased. This is another issue we didn't address due to time constraints, but we could at a later date add this functionality to the user's profile page.
- Improve navigation when viewing a particular artist album as to go back to view other albums from the artist you needed go to the artist page then navigate to that particular artist. We addressed this issue by adding a breadcrumb menu back to the previous pages.

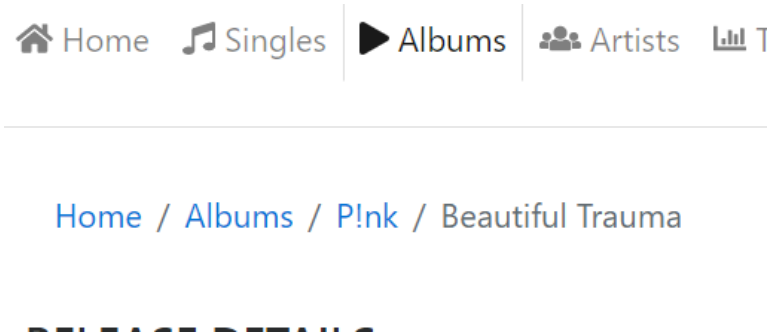


Figure 35 - Breadcrumb Menu from The Album Details Page

5.3 Compatibility Testing

To test the compatibility of our website we used to use three different methods. The first method is browser compatibility where we tested the website in a variety of different browsers. The second method we used was, mobile testing where we tested the website on an android mobile phone. And finally, the third method we used was operating system (OS) compatibility where we tested the website on two different OS's (Windows 10, Linux).

5.3.1 Browser Compatibility

For the browser compatibility testing we tested the website in the three most popular web browsers for windows. During our testing we found a number of differences in the look of the website.

Figure 36 - Image Upload on Different Browsers

From fig 36 we can see the change profile option on the user's profile page. As you can see there is somewhat a difference in the Microsoft Edge browser than the other two browsers. While both Google Chrome and Firefox use two buttons and a label for the uploading of images the Microsoft Edge browser uses two buttons and an input box. Also, the input box on the Edge browse is located to the left of the top button whereas the other two browsers have the label to the right of the button. This difference can be seen on six other pages on the website. The pages were these differences occur are, add new album, edit album, add new song, edit song, add new artist and the edit artist pages.

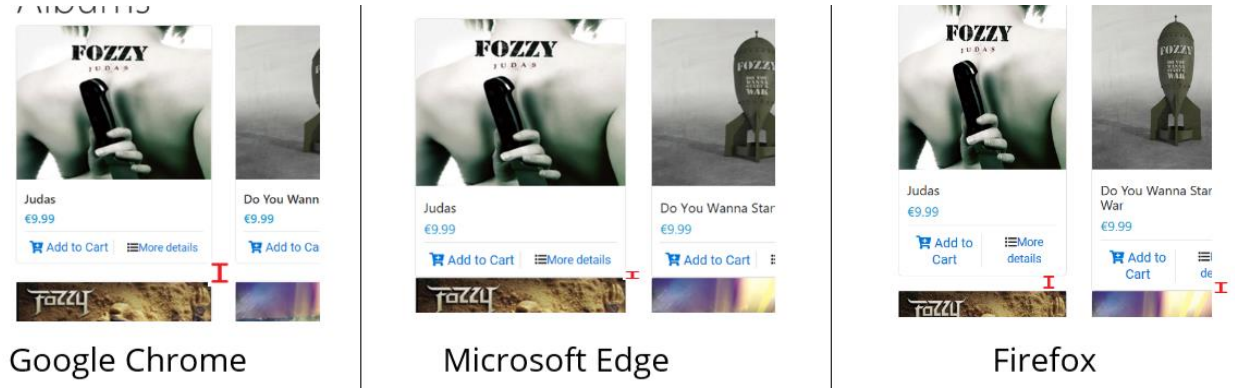


Figure 37 - Padding Between Items in Different Browsers

Another thing we noticed during the testing was the spacing (Padding) between components on the view album/song pages where different from browser to browser. As you can see from fig 37 the padding of 2% on the bottom of each component is consistent on Google Chrome the browser we used to develop the website. Whereas on Microsoft Edge there is no padding at the bottom of components and on Firefox the padding fluctuates between components, some components have the correct padding while others have none or very little.

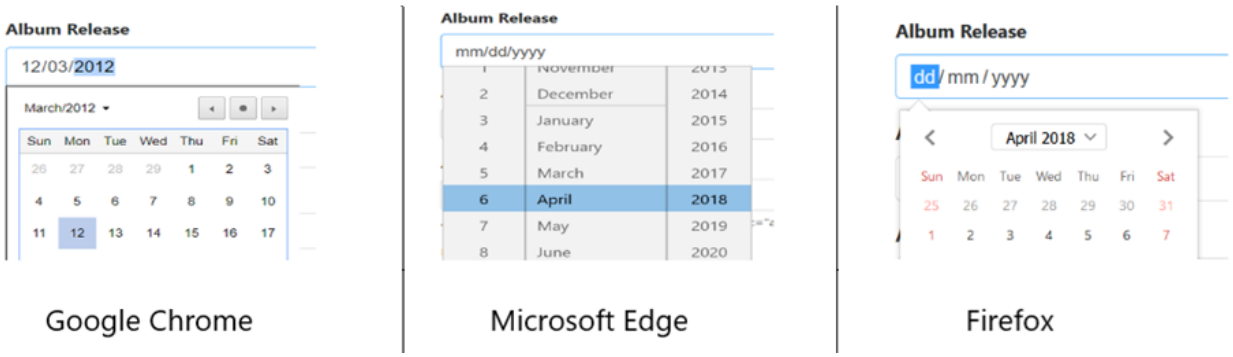


Figure 38 - Calendar Display in Different Browsers

Finally, the only other difference we found during testing was how the calendar is displayed on the add album and edit album pages.

During our thorough testing between browsers we did not find any functional differences between the browsers. Every feature of our website did what it supposed to. The only difference between browsers where the visual differences described earlier.

5.3.2 Operating System Compatibility

For the testing of the website on different Operating Systems (OS) we tested the live website on a variety of different OS's which included Windows 10, 8, 7 and on Linux Mageia. From our testing we found the website function similarly in each web browser on the different OS's.

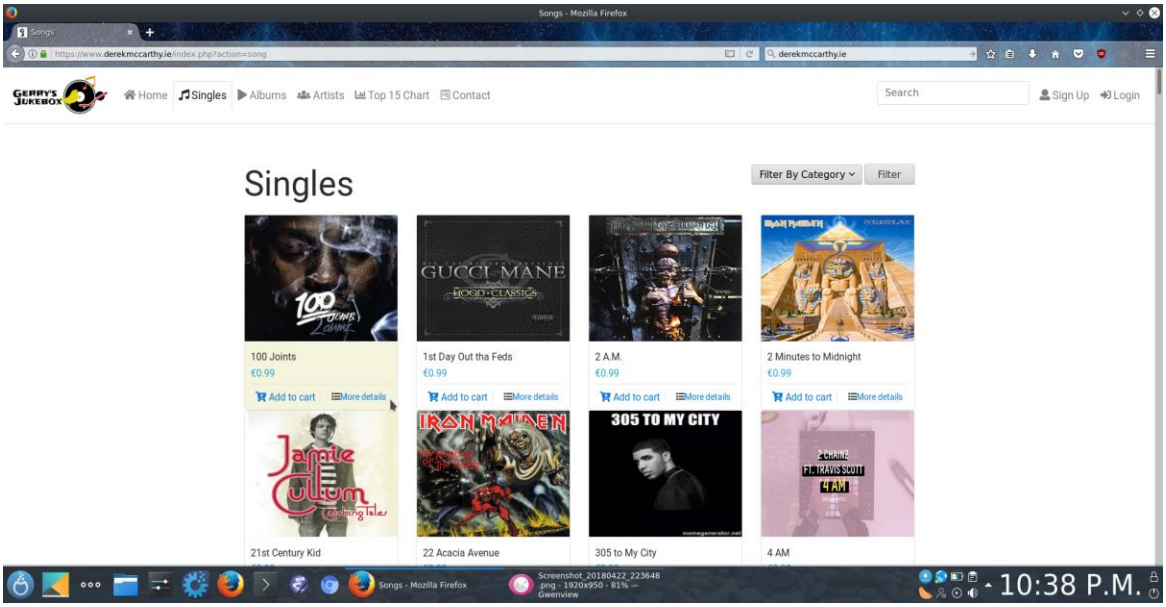


Figure 39 - Website Running in Mageia

In fig 39 we can see the website in the Firefox browser on the Linux Mageia OS.

5.3.3 Mobile Compatibility

To test how the website functions on mobile devices we used two different mobile OS's, Android from Google, and iOS from Apple.

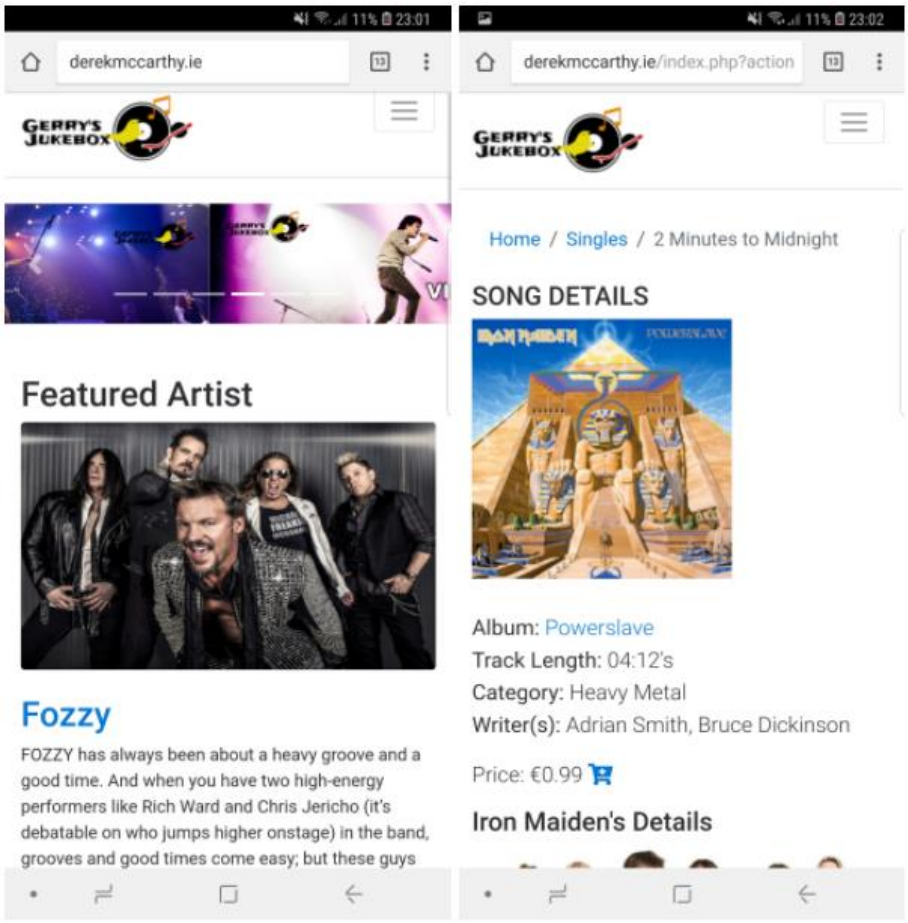


Figure 40 - Website Running on Android

In the above image we can see the home page and the song details page on an android mobile device. As you can see the website is 100% mobile compatible. This was achieved by using a grid system which moves content down as the screen size gets smaller as well as this, images resize dependent on the size of the screen size viewing the webpage. For example, if viewing the website on a laptop or PC the websites images will be full size with the content going horizontally across the screen. But if you view the website on a mobile device the images resize to fit the width of the device and the content that was going horizontally across the screen moves down vertically. This is done to avoid side scrolling when viewing on mobile devices.

5.4 Performance Testing

To test the performance of the website we used an online website speed test (https://www.webpagetest.org/). Which test the rendering times of the pages on the website and the rendering of each individual element of the website such as images, video, and html etc.

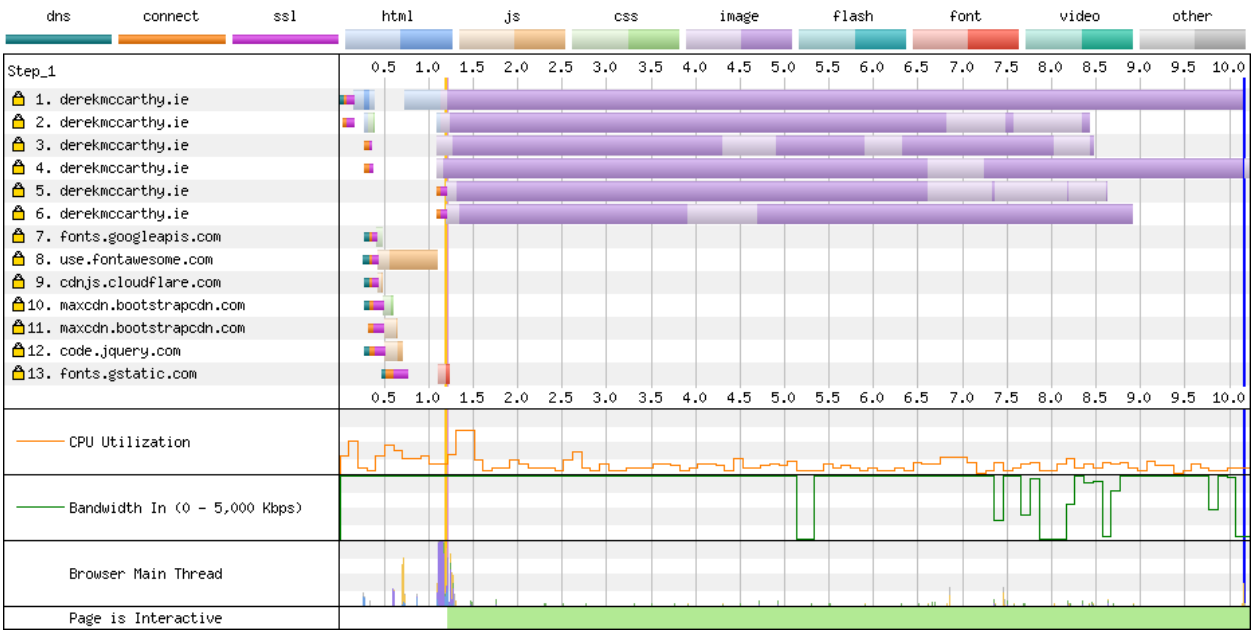


Figure 41 - Loading Time Performance

From fig 41 we can see the loading times of the elements of the website. From this we can see that images take the longest time to load with JavaScript taking the second longest.

5.5 Security Testing

For security testing we tested trying to access pages that were reserved for users and admins while we were not logged in. When trying to access any of these pages while not logged in we were met with the below image denying us access to these pages.

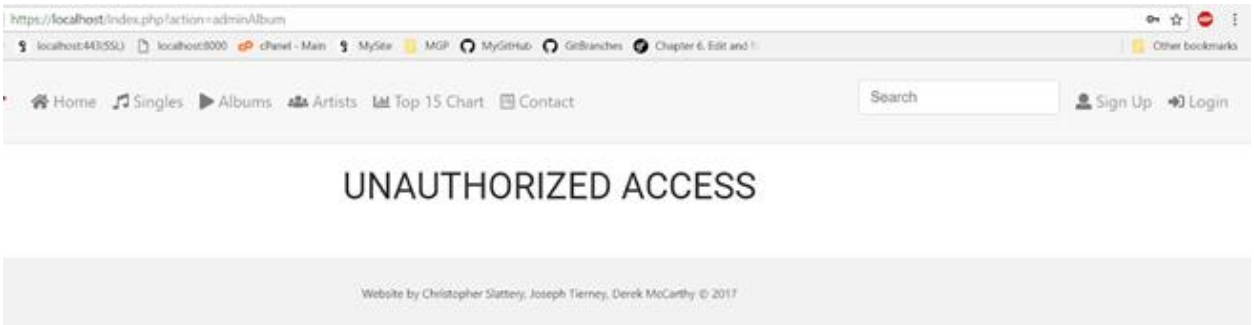


Figure 42 - Denied Access Page

Another form of security testing we performed was SQL injection attacks on user inputs. What an SQL injection attack is, when a person tries to gain access to the contents of a SQL database by inserting SQL queries into user inputs.

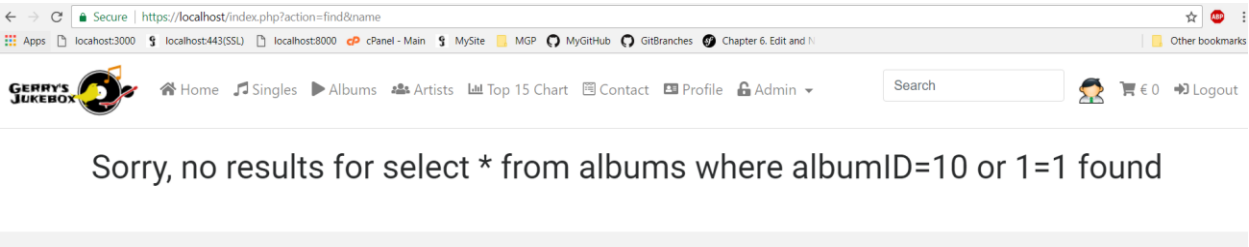


Figure 43 - Attempted SQL Injection Query

In fig 43 we see an attempted SQL injection attack on a search input box. But because we sanitize user inputs the SQL query is treated as a text string and not as a SQL query. We carried out the same tests on all user inputs on the website.

Chapter 6 Security

As this is an online website where users are submitting sensitive data security is paramount in protecting this data. To protect user's data, we implemented a number of security measures which included creating our own SSL certificates, only storing the hashed value of user's passwords and restricting access to certain aspects of the website such as admin features etc.

6.1 SSL Certificates

For the creation of the SSL certificates we used openssl which is a commercial-grade SSL and TLS toolkit for creating SSL certificates. We also had to enable the SSL on our apache web server and add the SSL cert and key to the apache config file.

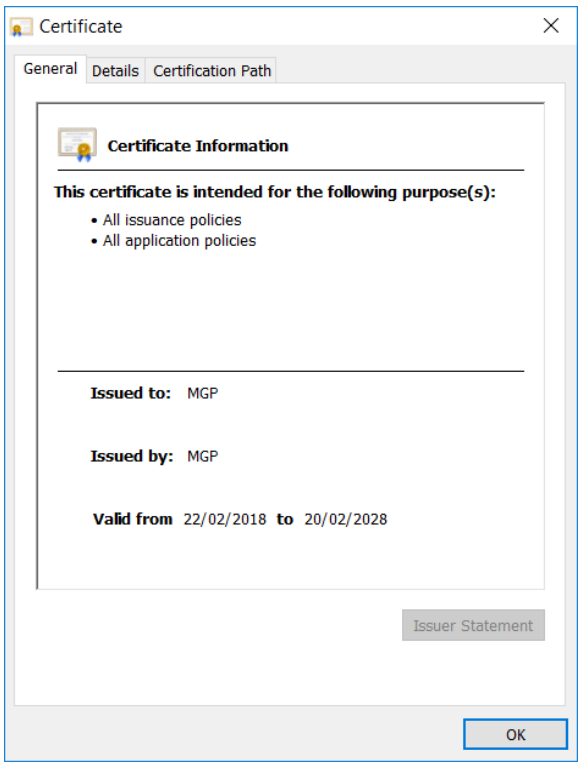


Figure 44 - SSL Certificate We Created

In the above fig 44 we can see the SSL cert we have created for use on the localhost.

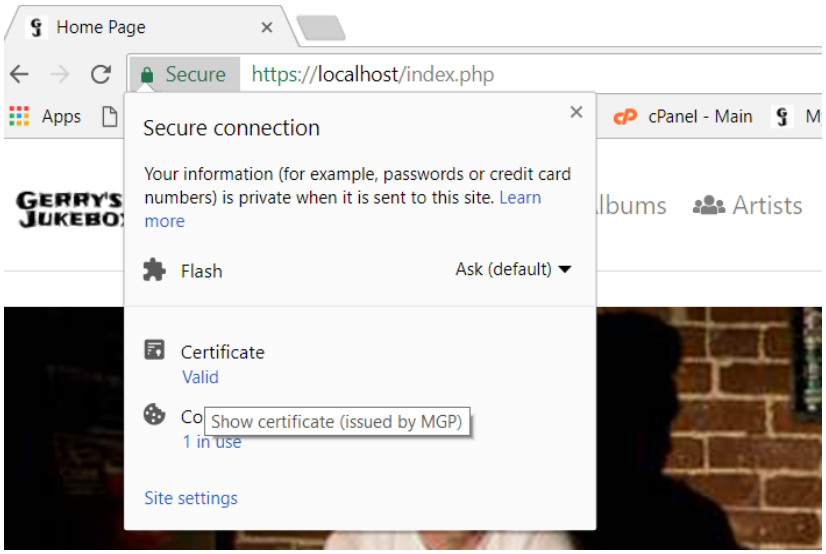


Figure 45 - Website Using SSL Certificate on Localhost

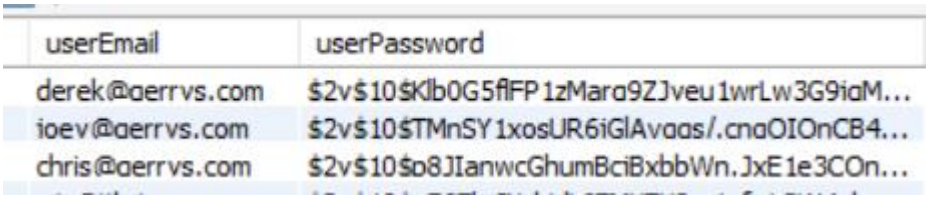
In the above fig 45 we can see that the localhost is using https on port 443 with the certificate we created.

6.2 Password Hashing

```
public function setPassword($userPassword)
{
    $hashedPassword = password_hash($userPassword, algo: PASSWORD_DEFAULT);
    $this->userPassword  = $hashedPassword;
}
```

Figure 46 - Password Hashing in PhpStorm

In order to protect user's login credentials in the event of a database breach only the hash value of the user's password will be stored in the database. To hash the user password, we used the built-in password hashing function in PHP 'password_hash' see fig 46. Which uses a strong one-way hashing algorithm to hash the user's passwords.



userEmail	userPassword
derek@gerrys.com	$2y$10$Klb0G5flFP1zMara9ZJveu1wrLw3G9iaM...
joev@gerrys.com	$2y$10$TMnSY1xosUR6iGlAvaas/.cnaOIOnCB4...
chris@gerrys.com	$2y$10$p8JIanwcGhumBciBxbbWn.JxE1e3COn...

Figure 47 - Passwords Stored in Database

In fig 47 we can see that only the hashed value of the user's password is stored in the database.

6.3 Restricted Access

In order to restrict access to certain pages on the website we blocked unregistered users from purchasing music and their access to certain pages that are intended for logged in users.
Once a user is logged in we further restricted access for users by only allowing administrators to add, edit and remove items on the website. In order to distinguish between regular users and administrators we gave the users a role ‘ROLE_ADMIN’ and ‘ROLE_USER’. The members with the role USER have access to purchasing albums, access to their profile page and all the access rights of a non-registered user. While members with the role ADMIN have access to add, remove, and edit items on the website as well as all the access rights of a regular user.

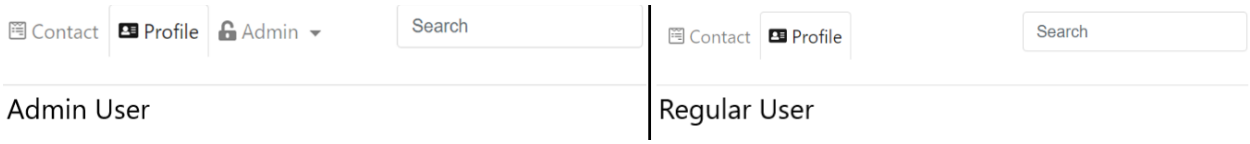


Figure 48 - Admin and Regular User Navigation View

Chapter 7: Conclusion

In this chapter, we will be summarizing the thesis and outline possible directions for future work.

7.1 Summary of Thesis

This thesis has introduced the importance of having a fully functional, user friendly and secure website. Chapter 1 gave a brief overview as to why we chose a music website, what the project contained, the skills needed in order to achieve a fully functional website and a brief overview of what is included in the thesis.

Chapter 2 discussed the functional requirements of the project, going into detail with various class diagrams. Chapter 3 went into further detail of the system design. In this chapter, we covered the user interface and overall website design using wireframes, which show the various web pages and what we had hoped they would look like at the end product. Chapter 3 also contained database normalization and how that was handled in the project.

Chapter 4 brought us to the implementation of both the prototype system and the final build. This chapter went into some technical information of the site, including system and software design, further explaining the software development process model we had chosen to work with. Chapter 4 also went into details of the prototypes design and what technologies had been used in order to create it, and, if necessary, explained the differences between the wireframes and prototype if there were any. Finally, this chapter also discussed the implementation of the final build, going into various details about the final design choices, how they differ from the prototype and the technologies used and why we used them for the final version and not the prototype.

Chapter 5 introduced testing and evaluation. Chapter 5 went into details about the results from various tests we conducted with the website and the six different phases. It discussed what we tested, these being functionality testing, usability testing, compatibility testing, performance testing and security testing. Each individual test was broken down further, explaining in detail what exactly it was that was being tested and the end result of the tests.

Finally, chapter 6 covered security in detail. This chapter contains information about the SSL certificate used for the project and how it was implemented, while also going into detail about how we kept the user's passwords protected in the off chance there is a database breach. Chapter 6 also discussed how we kept certain features of the website hidden from regular users and made them only accessible to the administrators of the website.

7.2 Future Work

For future work on this assignment, we would like to address some common issues among users. The first of these issues being usernames and passwords. If given more time, we would like to have implemented a "Forgotten username or password" feature, which will allow the user to send us an email / message, informing us of the situation so we are able to correct it by resetting their username and / or password.

Another issue we would like to expand on, is found on the songs and albums page. Currently, when visiting these pages, they display every single song and album that we have stored in the database,

making the page quite long and tedious to scroll through. Instead of this, we would like to have added page pagination, making it so that only a select number of songs / albums are displayed per page, ultimately making it neat and concise, adding to the user-friendly experience.

We would also like to have implemented a feature that allowed users to see their purchase history, again, we could not add this due to time constraints, but it is a feature we discussed.

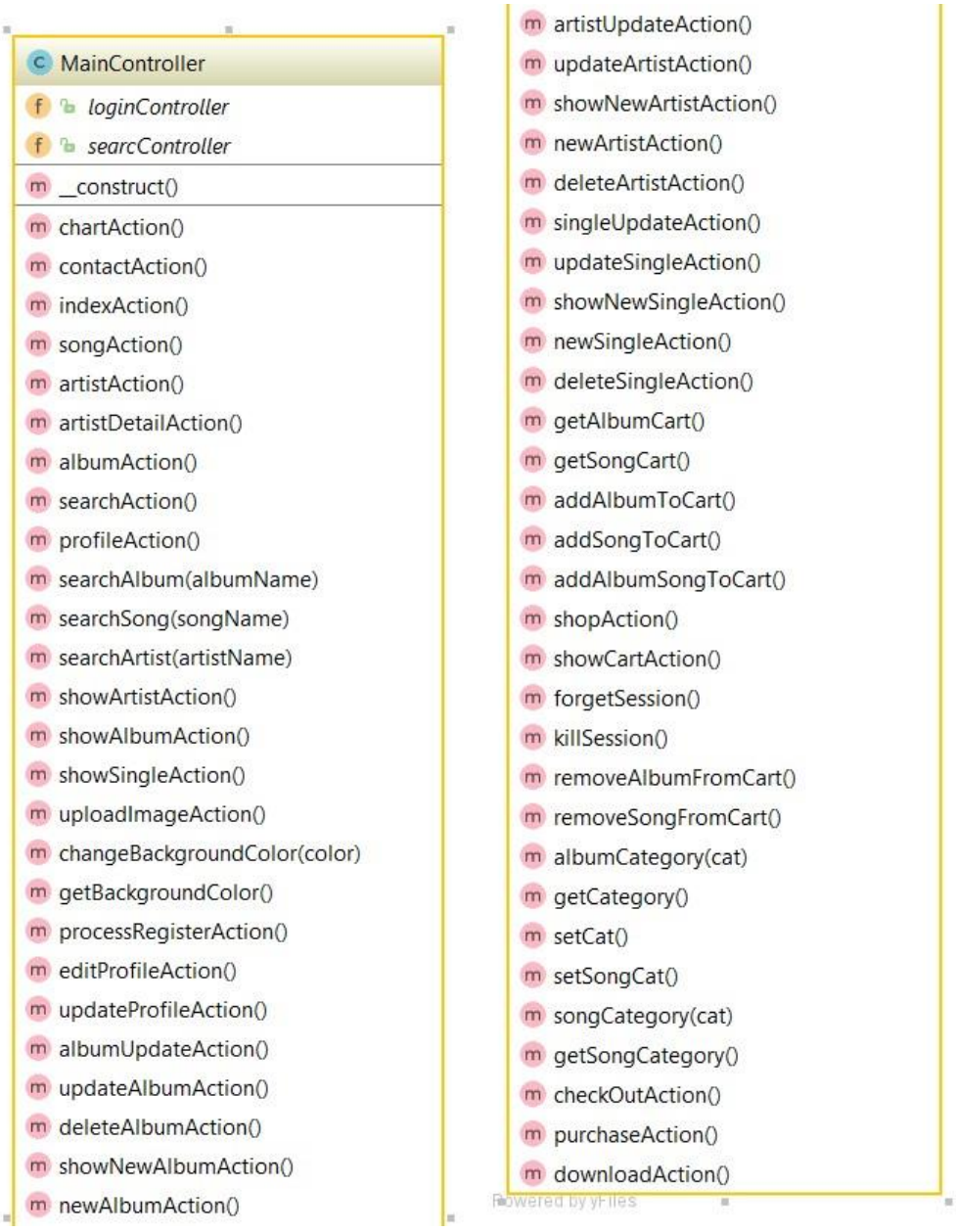
7.3 Final Conclusion

Overall, we feel that our project has met the criteria we set out to achieve, and we have accomplished the goals we set for ourselves in the early beginnings of the project.

Appendices

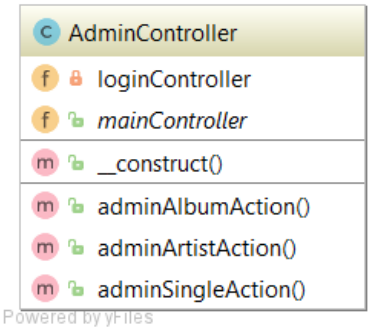
Appendix A: Class Diagram Details

Main Controller Class



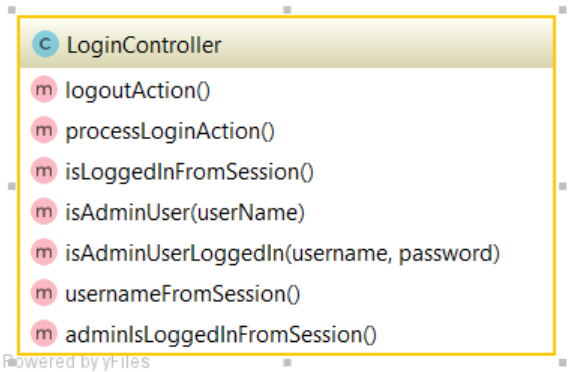
The main controller class is the main class in the website. It manages and delegates tasks to all other classes as well as performing its own tasks such as changing the background colour.

Admin Class



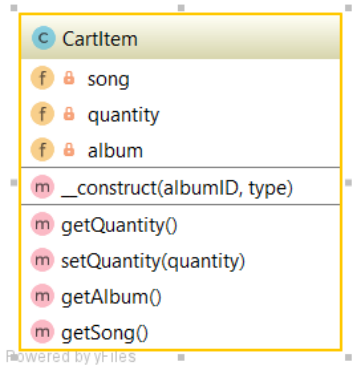
The admin controller class is used for granting access to administrators to modify data on the website.

Login Controller



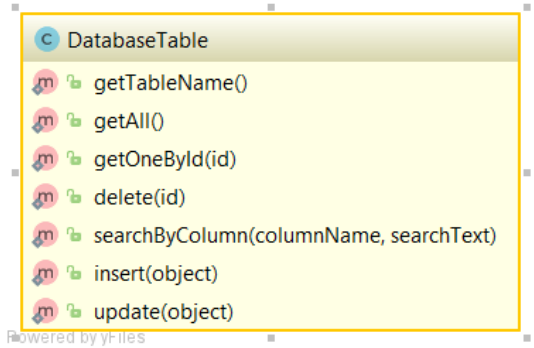
The login controller is a class which compares the data received through the login form for a match in the database. It does this by comparing the users inputted username and password. If the username and password are found in the database, they are granted access to purchase music and modify their profile. If the user in question happens to be an administrator, they are granted the same access privileges as a regular user plus the administrator privileges.

Cart Item Class



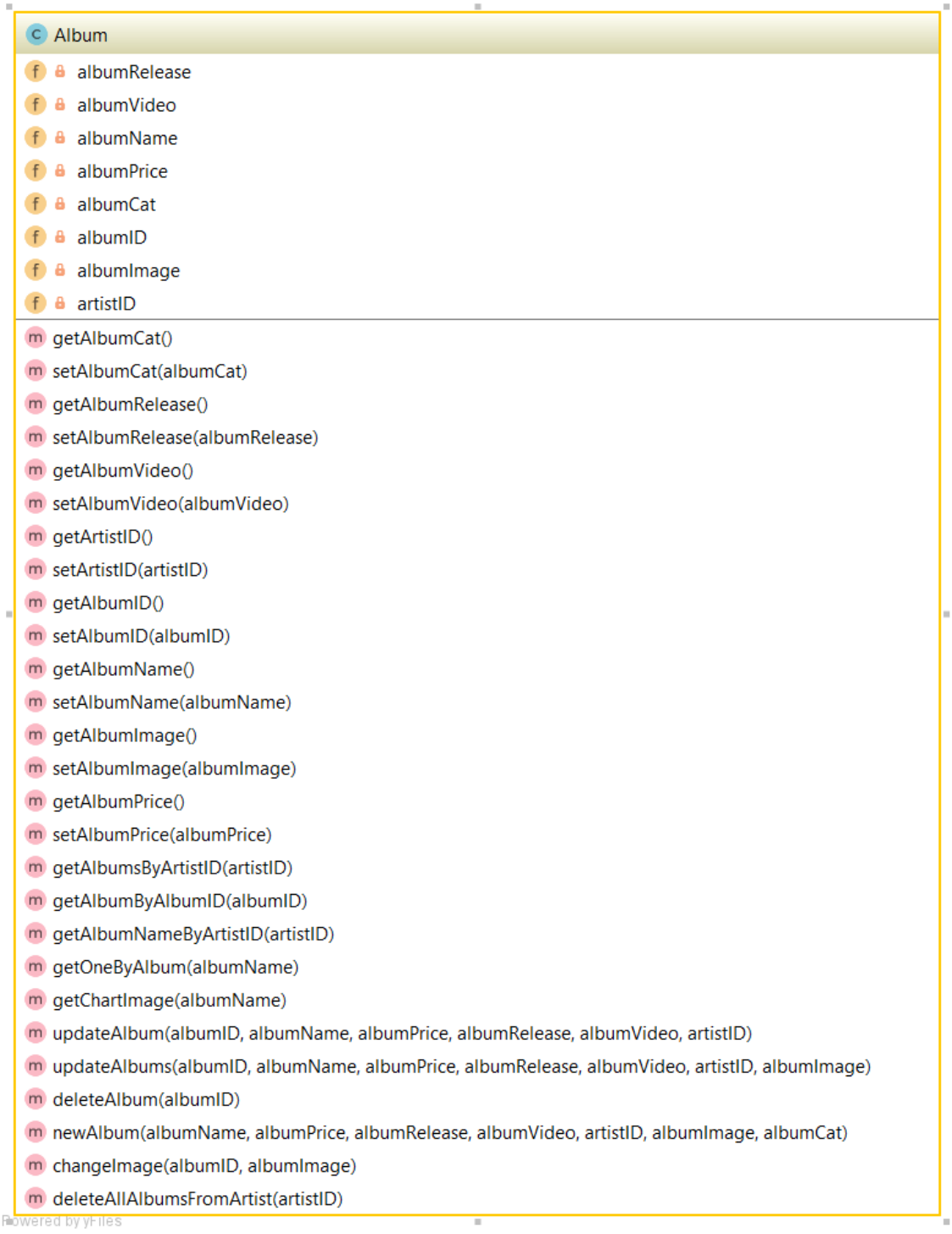
The cart item class is used to manage the shopping cart and its contents.

Database Table Class



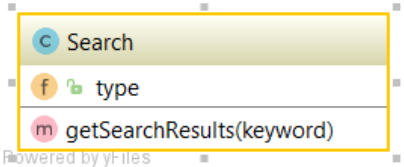
The database table class is used for managing common database SQL queries.

Album Class



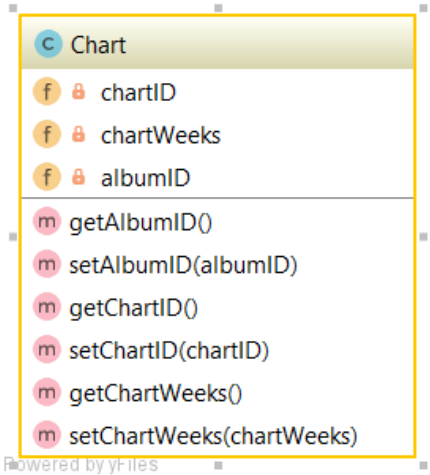
The album class is used to managing all aspects relating to albums on the website.

Search Class



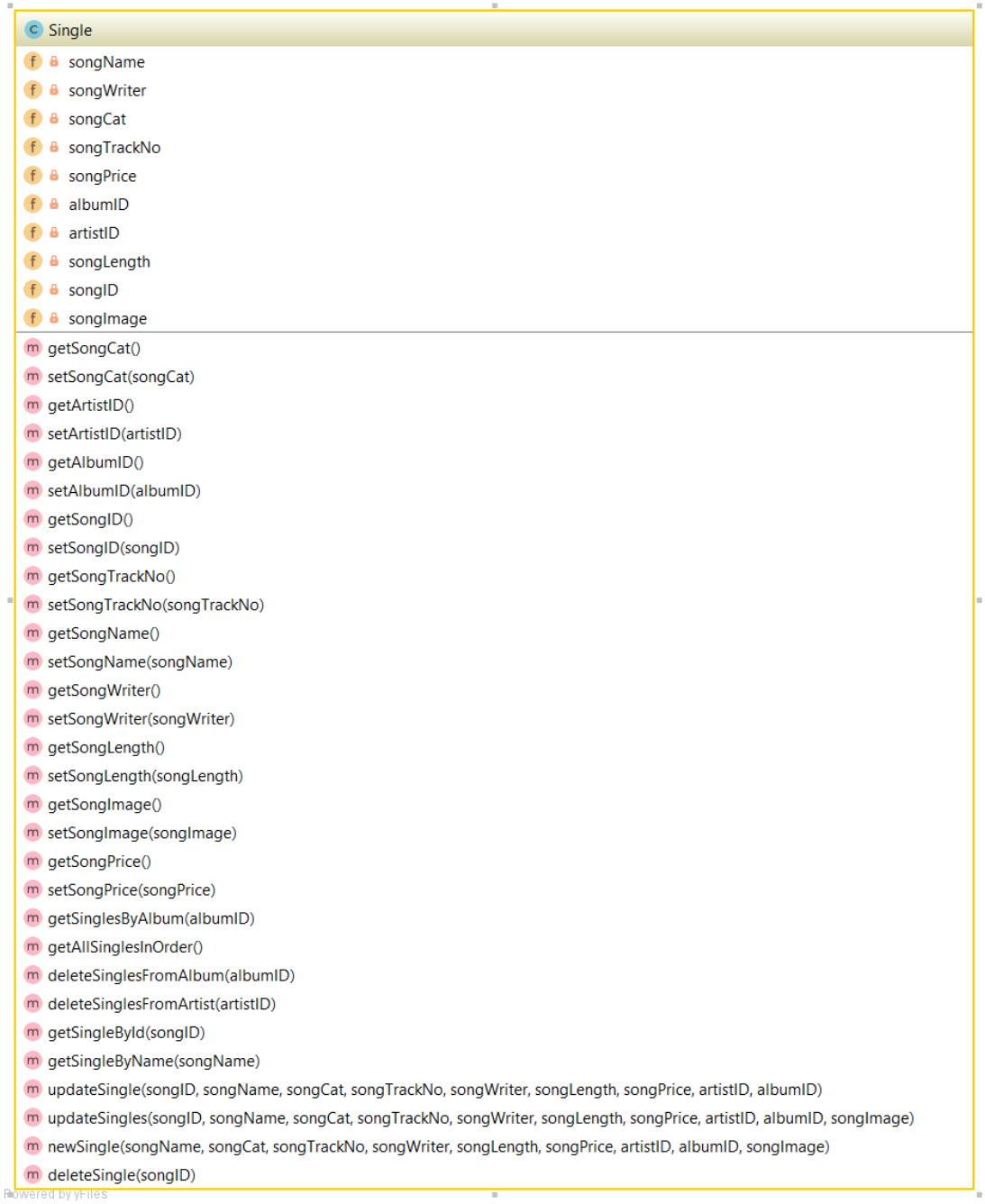
The search class basic function is to search the database for data that matches the users search query data.

Chart Class



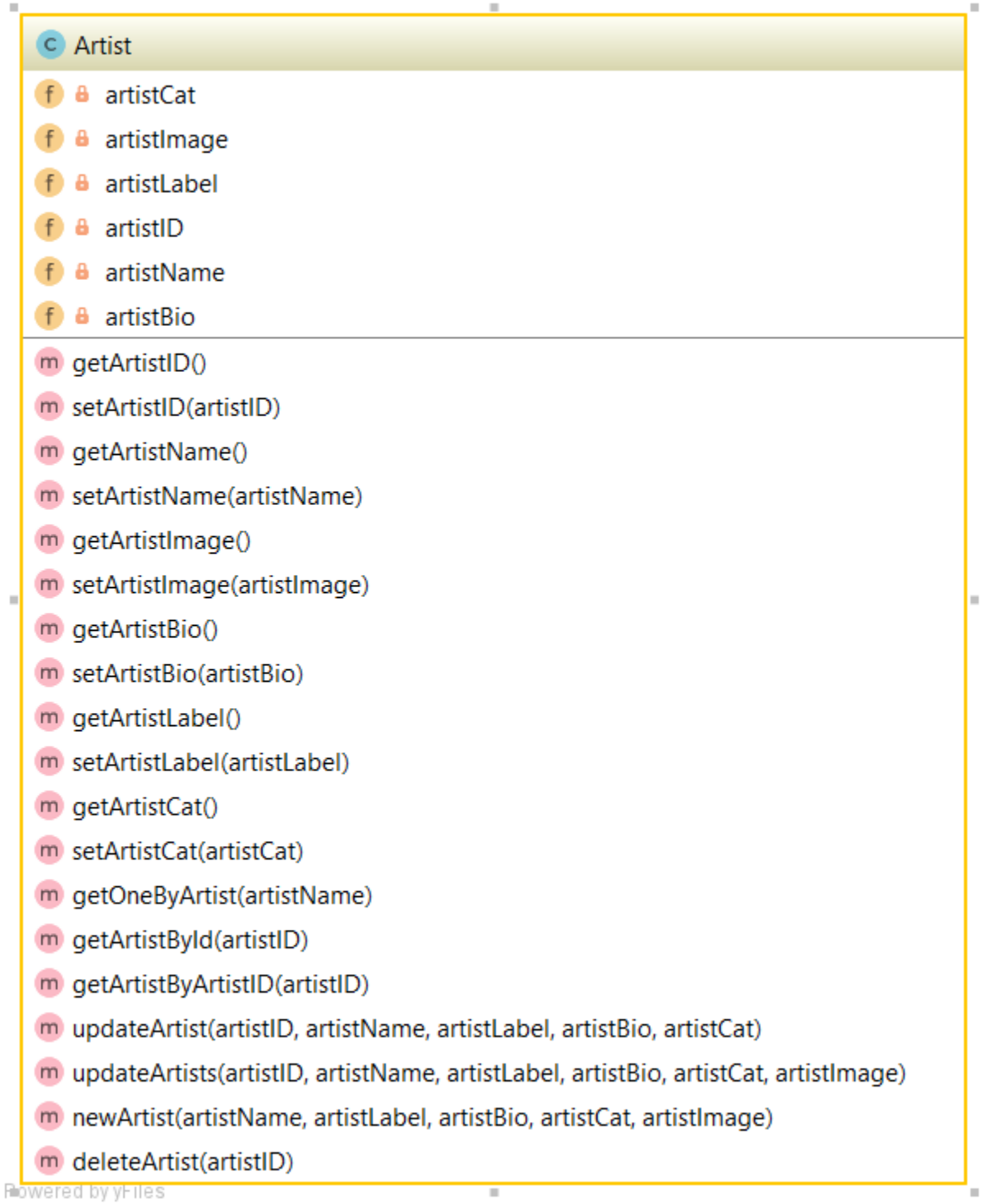
The chart class is responsible for managing and manipulating the album chart on the website.

Single Class



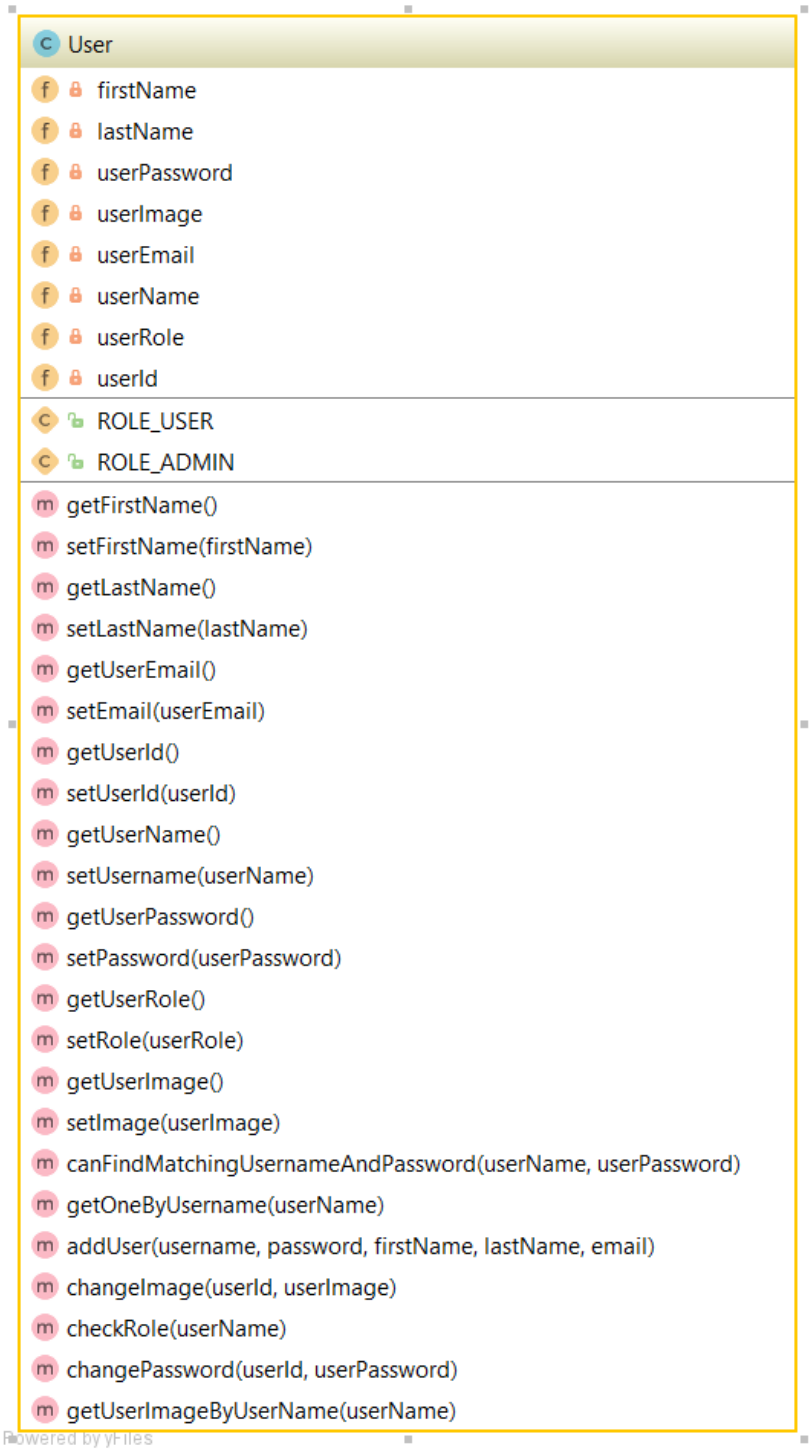
The single class is used to managing all aspects relating to songs on the website.

Artist Class



The artist class is used to managing all aspects relating to artists on the website.

User Class



The user class is used to managing all aspects relating to users on the website.

Appendix B: Questionnaire feedback form

Please take the time to fill in this quick survey.

1) What was your first impressions of the website?

1	2	3	4	5	6	7	8	9	10
Very Poor									Excellent

2) How did you find navigating the website?

1	2	3	4	5	6	7	8	9	10
Very Hard									Very Easy

3) How difficult did you find the website to use?

1	2	3	4	5	6	7	8	9	10
Very Hard									Very Easy

4) Would you recommend the website to a friend?

☐ **Yes** ☐ **No**

5) Could we add or improve any functionality of the website if yes please explain?

☐ **Yes** ☐ **No**

Figure 49 - Questionnaire Form

Appendix C: Literature Review

User Experience

User experience is when a company provides a service for its consumers and the experience the user feels from using the service. For example, for a website to provide an excellent UX (User Experience) they must not only meet the exact needs of the consumer it also needs to be easy to use and navigate. This can be achieved by a seamless merging of different aspects of the website such as coding, designing, graphics and user interface. A bad user experience could arise if a website has a difficult to use interface, which the user may find difficult to navigate on the website.

Although UI (User Interface) is a very important aspect of web design it differs from UX in the sense that the user interface is the way in which the website is designed and how a user interacts with it. In recent years there has been an increased demand for a seamless integration of both UX and UI into web design. This can be achieved by using a variety of programming tools such as HTML5, CSS, Bootstrap, Javascript and PHP.

The HTML will be used in the creation of the web pages with CSS used to set the style of how the HTML looks. If we decide to include a slide shows in our design we plan on using Javascript and CSS to create it. As for the frontend framework we plan on using a Bootstrap grid system to arrange our web page components and Bootstrap navigation menu which collapses when the website is accessed from devices with smaller screens. Having looked at other front end frameworks such as Foundation and Semantic UI we feel Bootstrap is best suited for our needs in the creation of a more responsive website. And finally for the backend framework we will incorporate PHP to handle all the web pages backend/server side needs such as accessing a database, adding a song to your cart and login and logout etc.

The term usability is a narrower concept of UX. It mainly focusses on quality of the UI and whether the website is easy to use and navigate. The author goes on to explain how UX surpasses usability by not only focusing on an easy to use website but to also include issues such as usefulness, desirability, credibility, and accessibility.

The main UX methodologies we will be incorporating into our website will be to make the user experience as positive as possible while using our website. We will achieve this using the five UX goals set out by the author they are: Brand, Theory, Empathy, Technology, and Vision.
Brand - The representation of our website in the form of the website name and logo we will use.

Theory - The knowledge available to us on the user's behavior while using our website. The researching of other music website and general market research can achieve this.
Empathy - Stepping into the shoes of a user using our website and understanding their needs.
Technology - Looking at different forms of technology and the impact it has on the user's experience. Such as when using a mobile device does the user have the same UX as when using a PC.
Vision - Looking to the future and what other UX's we could use and implement [1].

Web Security

It is very important to have web security as it is of paramount importance when surfing the internet and visiting different websites. The need for web security grows even more when there are personal details involved. This applies to personal details which throw up a risk of identity theft and of course if you are purchasing things from the internet. If there was no web security for a website, there is a high chance the website will be attacked, and if it is a website that stores people's details, well then that is not good for any party involved as it will also throw up a lot of legal issues with the host/hosts of the website.

One of the methods we have chosen to secure our website is Secure Socket Layer (SSL). We think secure socket layer certificates will be enough to secure out website from unwanted malicious attacks. The reason we think this is because of the mechanism that they use to secure websites. How they work will be explained in detail below in the methodology section of the paper.

The main methodology used in our project for web security was Secure Socket Layer (SSL). Secure Socket Layer Certificates are created to help an online business have a dependable environment to enable a user to be secure when surfing through the online music store. This is the most important thing to have when creating a website that will be selling products. This mechanism works by creating two keys - a Public key and a Private key. The creator of the keys lets the public key available to anybody and the private key is only seen by the creator/creators. If data has been encrypted with a public key, it can only be decrypted by the corresponding private key and vice versa. This technique is known as cryptography. Another way of securing a web application would be to use SSH (Secure Shell). Secure shell uses the same technique as SSL (Cryptography) to protect web applications. SSH was introduced to help connect and SSH client application to a SSH server. It provides a secure channel for on an unsecured network to facilitate the flow of private data. We decided to go with SSL for our project because our website is a

website that requires the transmission of private data like banking and personal details. SSL is used more than SSH for such websites. We feel that this is much more efficient for what we want to achieve [2].

Web Design and Development

Good web design is of paramount importance. One of the most important if not the most important part of web design is good navigation. This is important as users should be able to navigate their way through the website and also understand what part of the website they are navigating to with ease. The navigation should be easy for users to see and understand. This might also persuade the user to stay on the website for longer than they intended to.

In website design, it is important to have a website designed and developed specifically with the users in mind. The author discusses the idea of empathizing with the customers, and understanding their needs and the technologies that they use. This brings up a simple, yet effective question of "Why?" – imagine having a website fully completed, but it's nearly impossible to navigate and find exactly what you're looking for. Websites like this can cause customers to turn away from the website and decide to take their services elsewhere, which will also give you as a website designer and developer a bad reputation and cost the company money. The author gives an example of a website that was poorly designed and did not consider the customer's/user's needs. IBM's website was designed without considering the above-mentioned factors, they soon realized that their site was not doing as well as they thought it should have been. Some quick research carried out by the company showed that users were using the search feature the most on the site, indicating that the IBM website was so confusing that their customers could not navigate their site with ease. The second most popular feature on their website was the help section.

A website is essentially the face of a business on the internet, which is why focusing on customer needs and getting the right web design is just as important as anything else involving the business – it makes you stand out from the crowd and helps strengthen the brand you are trying to sell. This will be taken under consideration throughout the project.

Web accessibility is the practice of making a properly designed website for people who have disabilities – this has been taken under consideration during our website design. With web accessibility, all images on the website will include alternative text in the code, making the images information easily accessible for people who need to use a screen reader – allowing the reader to read the alternate text information of the image (and the page itself) aloud.

When using alternative text, the information is available to blind people, as well as people who may have turned images off on their mobile devices, or for people who live in rural areas with low bandwidth. Alternative text also makes the information available to technologies that may not be able to see the image, like search engines.

A good website will not rely on the use of a mouse. This falls into the category of web accessibility, allowing many older users, who may have limited motor control, be able to use the website without using the mouse, as it should provide its functionality by the keyboard. This will also allow people who need to use technologies, like speech input, to mimic the keyboard.

While there are multiple factors to consider when designing and developing a website, here we will look at what are considered to be the most important ones, according to the author.

Graphic Design: As a way to get attention and give customers a clear understanding of who you are and what it is you do and provide. This includes images, a brand logo, colours, interactive features for your website and other easy on the eye graphics that can give brand recognition to customers.

Website Layout: This is the basic layout of how the website's pages are presented to a visitor/customer. They should be easy to read and easy to learn how to use.

Navigation: The navigation of a website should be found user friendly and easy to traverse the website from all pages.

Fonts: These should simply look good and be easily readable by any user visiting the website [3].

Literature Review Conclusion

Good visualization, a structured layout, sensible navigation, and easy to understand instructions are all important parts of having a good user experience(UX). These parts also help with the implementation of a good user interface. Both of these tie in with each other. Implementing these ideas and techniques will help our website thrive. A major problem in computing is hacking and data being stolen or put into the hands of the wrong person. SSL in web security is a very important tool in combating that. There are also many more web security methodologies that help combat it, but we feel that SSL will best suit our website project.

As previously discussed, basing a website's design around what is best for the user is of vital importance, not only to build a reasonable reputation, but to improve business and build on a brand. Easy to navigate and easy on the eye websites are the key factors in keeping returning customers and adding the possibility of potential customers – people should not feel bombarded with images, complex features and hard to read fonts, otherwise they may take their business elsewhere, which is a major factor we intended to avoid when undertaking this project.

Literature Review References

User Experience

[1] **Title:** Behavior & Information Technology, Volume. 34 Issue 10, October 2015, pages 949-951.
Author(s): Tom Stewart
Available online at: http://zl6es4wn5y.search.serialssolutions.com/?ctx_ver=Z39.88-2004&ctx_enc=info%3Aofi%2Fenc%3AUTF-8&rfr_id=info%3Asid%2Fsummon.serialssolutions.com&rft_val_fmt=info%3Aofi%2Ffmt%3Akev%3Amtx%3Ajournal&rft.genre=article&rft.atitle=User+experience&rft.jtitle=Behaviour+%26+Information+Technology&rft.au=Stewart%2C+Tom&rft.date=2015-10-03&rft.issn=0144-929X&rft.eissn=1362-3001&rft.volume=34&rft.issue=10&rft.spage=949&rft.epage=951&rft_id=info:doi/10.1080%2F0144929X.2015.1077578&rft.externalDBID=n%2Fa&rft.externalDocID=10_1080_0144929X_2015_1077578&paramdict=en-UK

Web security

[2] **Title:** Computer Networks, Volume 48, Issue 5, 5 August 2005, pages 697-699.
Author(s): P. McDaniel, Aviel D. Rubin.

Available online at: http://0-www.sciencedirect.com.acpmil02web.ancheim.ie/science/journal/13891286/48/5?sdc=2

It also appears in the ITB library. - http://0-www.sciencedirect.com.acpmil02web.ancheim.ie/science/article/pii/S1389128605000423

Web Design and Development

[3] **Title:** The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience

Author(s): Douglas K. Van Duyne, James Landay and Jason I. Hong

Publication: Addison-Wesley Longman Publishing Co., Inc.

ISBN: 020172149X

Available online at:

https://books.google.ie/books/about/The_Design_of_Sites.html?id=CyT089J95agC&redir_esc=y

Appendix D: Project Planning

The project plan for the rest of our project will begin the week after christmas is finished, while we are on our break from college. This week begins on January 1st.

Week Commencing January 1st - Continue working on Homepage and create and implement a Contact Us page - these pages are imperative to our website.

Week Commencing January 8th - Finalise Homepage and Continue working on Contact Us page.

Week Commencing January 15th - Start creating and implementing the remaining HTML pages. This includes the artist page, songs page and charts page.

Week Commencing January 22nd - Continue creating and implementing the HTML pages that we started the week before.

Week Commencing January 29th - Start front end process of bootstrapping our website in order to help with the layout of the website.

Week Commencing February 5th - Create Database and add user and admin details.

Week Commencing February 12th - Continue creating database and move our hard-coded data to the database - this includes song infor and artist details.

Week Commencing February 19th - Start to template out HTML into PHP - this also includes making a cart for the users to shop.

Week Commencing February 26th - Configure and establish a connection with the the database through PHP.

Week Commencing March 5th - Create and implement the admin options - this will include adding songs, removing songs, editing songs, edit user, delete user etc.

Week Commencing March 12th - Create and implement SSL certificates into our website.

Week Commencing March 19th - Test and make sure SSL certificates are secure on our website.

Week Commencing March 26th - Start testing our website. Running our website in all of the web browser that are used on the internet.

Week Commencing April 2nd - Test website on mobile device web browsers.

Week Commencing April 9th – Start Thesis Write Up

Week Commencing April 16th – Continue Thesis Write Up

Week Commencing April 23rd – Finish Thesis Write Up

Appendix D: Project Diary (Parts 1 and 2)

Christopher Slattery

Week Beginning September 28th

Go through list of projects available to us and decide what project was suitable and would be achievable for us. This was a joint decision between the 3 of us and the 3 of us looked over the projects together.

Week Beginning October 2nd

This week we were tasked with doing research on the project. As we were doing a music website, it was essential to research music websites and software. We set out what websites we were going to research and allocated them to each member of the team. It was my job to research Apple iTunes software.

Week Beginning October 9th

For this week, I was tasked with doing my part of the project proposal. We broke the proposal down into different parts and as there was 9 parts we thought that 3 parts each would suffice. The parts I was tasked with were - Abstract, Contents, References, Justification/Benefit and Expected Results.

Week Beginning October 16th

This week I continued on and finished my 3 allocated sections of the project proposal.

Week Beginning October 23rd

This week was the week to start doing the research and a literature review for our project. We had two weeks to do this so I decided to break them down. For this week I decided to do the research that goes into a literature review and the research into the topic (Security).

Week Beginning October 30th

This week I started my required parts for the literature review.

Week Beginning November 6th

Continuation of the literature review and writing about my required topic.

Week Beginning November 13th

Putting the literature review together with my team to make it read as one cohesive document.

Week Beginning November 20th

This week we started on the Wireframe for the project - this helped us get a good skeleton to our project and helped with the structure that we would like to have and helped with the functionality that was already implemented.
I was tasked with wireframing 3 pages. The homepage, the contact page and the album details page.

Week Beginning November 27th

This week I was tasked with coding some HTML pages for our website. We started to create the Homepage, one artist page and one songs page. I was tasked with creating the Homepage.

Week Beginning December 4th
This week I was tasked with continuing on coding HTML.

Week Beginning December 11th
This week I was tasked with writing my part of the report. My part of the report consisted of an Overview/System Design and writing part of the project plan for what and when future work will take place on this project.

Week Beginning December 18th
This week I continued on with writing the report.

Week Beginning December 25th
I added more to the report this week and added some more songs to the website.

Week Beginning January 8th
Started researching about Secure Socket Layer(SSL). This included researching different ways to acquire SSL. It also included the different software programs that are available to make SSL Certificates with.

Week Beginning January 15th
This week I created the SSL Certificates for the website.

Week Beginning January 22nd
Implemented SSL certificates on localhost. This included creating a bash script.

Week Beginning January 29th
As we had most of the main parts of the website complete, I started to add songs and albums through the SQL database.

Week Beginning February 5th
This week I worked on the filters for the categories part of the website. I also continued adding songs and albums.

Week Beginning February 12th
This week was another week of adding songs as we did not think we had enough songs in the website. I also finished off working on the category filters for the website.

Week Beginning February 19th
Began upgrading the website from bootstrap 3 to bootstrap 4.

Week Beginning February 26th
This week the team held a meeting with our supervisor for the project and showed him how far along in the project we are and got feedback on it. I also completed the upgrading of the website from bootstrap 3 to bootstrap 4.

Week Beginning March 5th
This week I added a checkout page to the website which is used for when users are purchasing songs. I also started to validate the input fields on the website.

Week Beginning March 12th
Completed validating the input fields on the website. This took a bit of time as we had a lot of input fields to validate.

Week Beginning March 19th
This week I started to implement the add to cart functionality. We needed this as the website will make songs available for people to purchase.

Week Beginning March 26th
Completed the adding to cart functionality.

Week Beginning April 2nd
This week i added more songs and albums to the website using SQL.

Week Beginning April 9th
This week I worked on the thesis write up for the website project. I also got some family and friends to use the website and get some feedback.

Week Beginning April 16th
Continued working on thesis write up for this week.

Week Beginning April 23rd
Finished up working on thesis write up.

Joseph Tierney
Week Beginning September 25th
The group looked through the list of available projects, after narrowing it down to either an Online Music Store or a banking application, we ultimately decided to go with the Online Music Store, as it was something we believed could be achievable.

Week Beginning October 2nd
This week, each of us decided to research music websites, specifically the main three music websites; Apple iTunes, Google Play and Spotify. I was given Spotify to research. Along with my research of Spotify, I also researched some award-winning websites that had a similar design to what we wanted to achieve, these can be seen at the links below.
https://www.awwwards.com/sites/nubikk
https://www.awwwards.com/sites/pictanovo

Week Beginning October 9th
The group began to work on the project proposal. Firstly, as a group, we divided up the project proposal, with myself working on the Main Research Questions, Methodologies and the Project Plan. During the rest of the week, I managed to finish the three topics of the project proposal I was tasked to work on.

Week Beginning October 16th
We combined all our parts of the project proposal this week and submitted it to our supervisor.

Week Beginning October 23rd
This week was the start of our literature review for the project. My chosen area to research and review was Web design and development, this was mainly due to the fact that good web design is the key to a successful website and/or business in general. I felt that this would help us make the best website we possibly could. I started looking around at various websites I enjoyed visiting in order to see what it was that drew me in as a user to keep coming back.

Week Beginning October 30th
This week, I started reading multiple journals, academic publications and books, trying to determine which I felt was best suited for the project. Eventually, I found The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centred Web Experience by Douglas K. Van Duyne, James Landay and Jason I. Hong. What made me chose this book was the fact they hit the major points of creating a good website for the users, whether they have a disability or not.

Week Beginning November 6th
During this week, I began to write and finish my review on The Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centred Web Experience.

Week Beginning November 13th
This week, as a group, we began to put the literature review together in order to make it read as one cohesive document and sent it to our supervisor for feedback.

Week Beginning November 20th
We began to plan the design of our website using wireframes. We divided this up between the three of us, and I was tasked with designing the songs and charts wireframe, along with planning out the use case diagram for our project.

Week Beginning November 27th
Continued to code the album details pages, I also coded the top 15 chart as a way to navigate to the artists pages for the prototype.

Week Beginning December 4th
Continued coding album details and styling.

Week Beginning December 11th
Continued coding album details.

Week Beginning January 22nd
After going through multiple websites, I began coding the search bar for our website.

Week Beginning January 29th
Continued coding the search bar.

Week Beginning February 5th
Finished coding the search bar.

Week Beginning February 12th

This week, I began working on the contact page, trying to keep it as close to the wireframes as I possibly can.

Week Beginning February 19th
During this week, I finished coding the contact page.

Week Beginning February 26th
This week, we showed our supervisor our project and got appropriate feedback, as well as information regarding the testing of the site. He recommended we change our home page.

Week Beginning March 5th
Began asking other people (friends, family members) to look at the home page of the website and see what we can improve on. After some feedback, most of which was about not knowing what the site was on their first visit, I began discussing with my group about a new home page. Also started my part of the thesis - chapter 4 and chapter 7.

Week Beginning March 12th
During this week, I began looking at various home page designs, before finally settling on a simple banner-style home page, meaning that the site will have a few banners in a slideshow format, telling users what they can do on our site.

Week Beginning March 19th
This week, I discussed the new home page idea with the group. Once it was agreed, I began coding the home page and designing the banners. Continued working on the thesis.

Week Beginning March 26th
Finished coding the home page during this week. Also during this week, I was tasked with making the users purchase download a song. Continued working on the thesis.

Week Beginning April 2nd
Finished adding music to the website, and making it be available to download on purchase. This week I also began doing some general styling for the website.

Week Beginning April 9th
During this week, we dedicated it to mainly testing the site with other people. I had five of my friends test the website and fill out the survey. Continued working on the thesis.

Week Beginning April 16th
Finished a logo for the website, and added some icons to the navbar as well as a favicon. Began working on breadcrumb navigation for the site. Continued working on the thesis.

Week Beginning April 23rd
Finished the breadcrumb navigation for the site during this week. We also demoed the final version of the website to our supervisor and finished writing the thesis.

Derek McCarthy
Week Beginning September 25th
This week we were given a list of proposed projects and had to choose. After some deliberation we decided on an Online Music Store from supervisor Michael O'Donnell.

Week Beginning October 2nd
This week I began to research Google Play to see what musical features they offer. I also began researching other award-winning websites for inspiration of features we could use and improve on for our own project. Below is some of the links to the websites I researched. On Thursday the 5th of October we met with our supervisor Michael O'Donnell and we discussed what features we will include in the website as well as what we discussed whether these features are feasible or not.

https://www.awwwards.com/sites/201-brand-site
https://play.google.com/music/listen
https://www.awwwards.com/sites/elegant-seagulls

Week Beginning October 9th
This week we began the project proposal we first divided the workload up between the three of us. For my part of the proposal I have to do the background of the project, Feasibility and the conclusion. I finished the project background and the feasibility but will have to finish conclusion when we put the proposal together with all our parts. On Thursday the 12th of October we met with our supervisor Michael O'Donnell and we talked about how we're getting along with the proposal and also that we should to send him a copy of the proposal for him to give us feedback.

Week Beginning October 16th
This week we combined all our parts of the project proposal on Google Docs and i finished the conclusion. On Thursday the 19th of October we met with our supervisor Michael O'Donnell and he gave us feedback from the proposal and improvements we could make.

Week Beginning October 23rd
This week we started our literature review and we all chose an area from the project development to review. For my part I chose User Experience. As i felt this is one of the main elements of not only web design and program design in general. I started researching lots of scholarly papers/journals on google scholar and ITB online library service. To date I haven't found one that appeals to me of yet.

Week Beginning October 30th
This week I found a paper on user experience I can relate to. And I am now in the process of writing my review.

Week Beginning November 6th
This week I finished my review of Tom Stewart's, Behaviour & Information Technology, Volume. 34 Issue 10, October 2015 User Experience chapter.

Week Beginning November 13th
This we added the parts we deemed necessary from our individual reviews to the groups literature review. We had to make it to read as if it was one document so in order to do this I had to change some sections of my review to keep with the theme of the review. As well as uploading the literature review to Moodle we sent a copy to our supervisor Michael who is going to read it and give us feedback at our next meeting.

Week Beginning November 20th
Having finished our research this week we began the actual design phase of our website. At the start of the week we gave ourselves two weeks to finish the design phase of the website. This included the wireframes of the webpages, use case scenario and ERD. For my part of the design I had to create a wireframe of the Artists and Artist details pages as well as create the ERD for our database. Having found an excellent website for creating wireframes called mockflow [1] we actually finished the design phase this week. As for the creation of the ERD I used MySQL Workbench 6.3. On Thursday 24th of October we met with our supervisor and he gave us feedback from the literature review and talked to us on area where we could improve on in the review.

[1] - https://mockflow.com/

Week Beginning November 27th
This week we began the coding of the website**.** For this I have to create a web page for each artist.

Week Beginning December 4th
This week I've been continuing creating the artists pages.

Week Beginning December 11th
This week I've being preparing the deliverables for project. And finalizing the artists pages for the demo.

Week Beginning December 18th
This week I created the main controller class in PHP and implemented the PHP functionality into the websites html pages. I also re-coded the index page to use a PHP MVC model with the index page been our view.

Week Beginning January 8th
This week I created the artists table in the database and populated it with the artists data from our html website. I also created the artist class in PHP and implemented the artist table within the website using object oriented PHP . I also implemented the artist details page and database connection.

Week Beginning January 15th
This week I created the database for the albums and added some albums we have so far to it. I also created the Album class in PHP for accessing the albums from the database. As we have a lot of albums and have to still get more I didn't get them all added to the database.

Week Beginning January 22nd
This week I finished adding the albums and their information we have gathered so far to the database.

Week Beginning January 29th
This week I created the chart class in PHP and created the top 15 chart table in MySQL. And also implemented the chart database table with the website. I also converted the artist and album images from jpeg format to a more web friendly form PNG and also resized them.

Week Beginning February 5th
This week I created the Singles class in PHP and also created the singles database table and implemented the singles database connection with the website. Also, I began adding some of the singles that make up the albums in our website to the database but did not get them all added as we have over 350 singles so far.

Week Beginning February 12th
This week i finished adding all of the singles to the singles table in the database. And also implemented bootstrap to the components that didn't already have it done.

Week Beginning February 19th
This week I created the login controller in PHP and implemented the database connection for them. I also created the login and signup models where you can login to the website or signup if you are not a member.

Week Beginning February 26th
This week I created the user profile page the page you see when you login. On this page you can edit you profile such as changing profile image and password and viewing your details etc. We also met with our supervisor and showed him the progress we have made so far on the project.

Week Beginning March 5th
This week I created the admin controller and the edit album page. The edit album is an admin option for adding new albums and their details in the database. Deleting albums and also updating albums details in the database such as changing album image, name and price etc.

Week Beginning March 12th
This week I created the edit artist page and the subsequent pages associated with it such as updating, deleting and adding new artist pages.

Week Beginning March 19th
This week I created the edit singles page and the subsequent pages associated with it such as updating, deleting and adding new singles pages.

Week Beginning March 26th
This week I redid the chart page and created a chart database table to manage the album charts. I also started the compatibility testing of the website this included testing the functionality of the website on different browsers, operating systems and mobile devices.

Week Beginning April 2nd
This week i finished the compatibility testing of the website. And also started the functionality testing. This included testing forms, links and syntax validity.

Week Beginning April 9th
This week i finished the functionality testing. And also finished the database testing. We also got 15 of our friends and family to test the website and to fill in a questionnaire after testing it.

Week Beginning April 16th
This week i started my chapters of the thesis write up System Analysis, Testing and Security.

Week Beginning April 23rd
This week i finished my sections of the thesis write up.

Appendix E: Code listings

Admin Controller Class

```
<?php
namespace Itb;
class AdminController
{
    private $loginController;
    public function __construct()
    {
        $this->loginController = new LoginController();
        $this->mainController = new MainController();
    }
    public function adminAlbumAction()
    {
        $pageTitle = "Edit Albums";
        $adminAlbumLinkStyle = 'nav-link active';
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        if ($isLoggedIn){
            $username = $this->loginController->usernameFromSession();
            $isAdmin = $this->loginController->isAdminUser($username);
            $backgroundColor = $this->mainController->getBackgroundColor();
            $albumCart = $this->mainController->getAlbumCart();
            $songCart = $this->mainController->getSongCart();
            $select = $this->mainController->getCategory();
            $select1 = $this->mainController->getSongCategory();
            require_once __DIR__ . '/../templates/CRUD/albumCrud.php';
        } else {
            $message = 'UNAUTHORIZED ACCESS';
            require_once __DIR__ . '/../templates/message.php';
        }
    }

    public function adminArtistAction()
    {
        $pageTitle = "Edit Artist";
        $adminAlbumLinkStyle = 'nav-link active';
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        if ($isLoggedIn){
            $username = $this->loginController->usernameFromSession();
            $isAdmin = $this->loginController->isAdminUser($username);
            $backgroundColor = $this->mainController->getBackgroundColor();;
            $albumCart = $this->mainController->getAlbumCart();
            $songCart = $this->mainController->getSongCart();
```

```
            $select = $this->mainController->getCategory();
            $select1 = $this->mainController->getSongCategory();
            require_once __DIR__ . '/../templates/CRUD/artistCrud.php';
        } else {
            $message = 'UNAUTHORIZED ACCESS';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function adminSingleAction()
    {
        $pageTitle = "Edit Singles";
        $adminAlbumLinkStyle = 'nav-link active';
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        if ($isLoggedIn){
            $username = $this->loginController->usernameFromSession();
            $isAdmin = $this->loginController->isAdminUser($username);
            $backgroundColor = $this->mainController->getBackgroundColor();
            $albumCart = $this->mainController->getAlbumCart();
            $songCart = $this->mainController->getSongCart();
            $select = $this->mainController->getCategory();
            $select1 = $this->mainController->getSongCategory();
            require_once __DIR__ . '/../templates/CRUD/singleCrud.php';
        } else {
            $message = 'UNAUTHORIZED ACCESS';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
}
```

Album Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class Album extends DatabaseTable
{
    private $albumID;
    private $albumName;
    private $albumImage;
    private $albumPrice;
    private $artistID;
    private $albumRelease;
    private $albumVideo;
    private $albumCat;
```

```
public function getAlbumCat()
{
    return $this->albumCat;
}
public function setAlbumCat($albumCat)
{
    $this->albumCat = $albumCat;
}
public function getAlbumRelease()
{
    return $this->albumRelease;
}
public function setAlbumRelease($albumRelease)
{
    $this->albumRelease = $albumRelease;
}
public function getAlbumVideo()
{
    return $this->albumVideo;
}
public function setAlbumVideo($albumVideo)
{
    $this->albumVideo = $albumVideo;
}
public function getArtistID()
{
    return $this->artistID;
}
public function setArtistID($artistID)
{
    $this->artistID = $artistID;
}
public function getAlbumID()
{
    return $this->albumID;
}
public function setAlbumID($albumID)
{
    $this->albumID = $albumID;
}
public function getAlbumName()
{
    return $this->albumName;
}
```

```
    public function setAlbumName($albumName)
    {
        $this->albumName = $albumName;
    }
    public function getAlbumImage()
    {
        return $this->albumImage;
    }
    public function setAlbumImage($albumImage)
    {
        $this->albumImage = $albumImage;
    }
    public function getAlbumPrice()
    {
        return $this->albumPrice;
    }
    public function setAlbumPrice($albumPrice)
    {
        $this->albumPrice = $albumPrice;
    }
    public static function getAlbumsByArtistID($artistID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = 'SELECT * FROM albums WHERE artistID=:artistID';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':artistID', $artistID, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetchAll()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function getAlbumByAlbumID($albumID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = 'SELECT * FROM albums WHERE albumID=:albumID';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':albumID', $albumID, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
```

```
    if ($object = $statement->fetch()) {
        return $object;
    } else {
        return null;
    }
}
public static function getAlbumNameByArtistID($artistID)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = 'SELECT albumName FROM albums WHERE artistID=:artistID';
    $statement = $connection->prepare($sql);
    $statement->bindParam(':artistID', $artistID, \PDO::PARAM_STR);
    $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
    $statement->execute();
    if ($object = $statement->fetch()) {
        return $object;
    } else {
        return null;
    }
}
public static function getOneByAlbum($albumName)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = 'SELECT * FROM albums WHERE albumName=:albumName';
    $statement = $connection->prepare($sql);
    $statement->bindParam(':albumName', $albumName, \PDO::PARAM_STR);
    $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
    $statement->execute();
    if ($object = $statement->fetch()) {
        return $object;
    } else {
        return null;
    }
}
public function getChartImage($albumName)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = "SELECT * FROM albums WHERE albumName=:albumName";
    $statement = $connection->prepare($sql);
    $statement->bindParam(':albumName', $albumName, \PDO::PARAM_STR);
    $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
```

```
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function
updateAlbum($albumID,$albumName,$albumPrice,$albumRelease,$albumVideo,$artistID,$alb
umCat)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE albums SET albumName = '$albumName', albumPrice = $albumPrice,
albumRelease = '$albumRelease', albumVideo = '$albumVideo', albumCat = '$albumCat',
        artistID = $artistID WHERE albumID = $albumID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function
updateAlbums($albumID,$albumName,$albumPrice,$albumRelease,$albumVideo,$artistID,$al
bumImage,$albumCat)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE albums SET albumName = '$albumName', albumPrice = $albumPrice,
albumRelease = '$albumRelease', albumVideo = '$albumVideo',
        artistID = $artistID, albumImage = 'images/albums/$albumImage', albumCat = '$albumCat'
WHERE albumID = $albumID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function deleteAlbum($albumID)
    {
```

```
        Single::deleteSinglesFromAlbum($albumID);
        Chart::deleteAlbumFromChart($albumID);
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "DELETE FROM albums WHERE albumID=$albumID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function
newAlbum($albumName,$albumPrice,$albumRelease,$albumVideo,$artistID,$albumImage,$al
bumCat)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "INSERT into albums (albumName, albumPrice, albumRelease, albumVideo,
artistID, albumImage,albumCat) VALUES
('$albumName',$albumPrice,'$albumRelease','$albumVideo','$artistID','images/albums/$albumI
mage','$albumCat')";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public function changeImage($albumID, $albumImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE albums SET albumImage = '/images/albums/$albumImage' WHERE
albumID=$albumID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;// = true;
```

```
    }
    public static function deleteAllAlbumsFromArtist($artistID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "DELETE FROM albums WHERE artistID=$artistID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
}
```

Artist Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class Artist extends DatabaseTable
{
    private $artistID;
    private $artistName;
    private $artistImage;
    private $artistBio;
    private $artistLabel;
    private $artistCat;
    public function getArtistID()
    {
        return $this->artistID;
    }
    public function setArtistID($artistID)
    {
        $this->artistID = $artistID;
    }
    public function getArtistName()
    {
        return $this->artistName;
    }
    public function setArtistName($artistName)
    {
        $this->artistName = $artistName;
```

```
    }
    public function getArtistImage()
    {
        return $this->artistImage;
    }
    public function setArtistImage($artistImage)
    {
        $this->artistImage = $artistImage;
    }
    public function getArtistBio()
    {
        return $this->artistBio;
    }
    public function setArtistBio($artistBio)
    {
        $this->artistBio = $artistBio;
    }
    public function getArtistLabel()
    {
        return $this->artistLabel;
    }
    public function setArtistLabel($artistLabel)
    {
        $this->artistLabel = $artistLabel;
    }
    public function getArtistCat()
    {
        return $this->artistCat;
    }
    public function setArtistCat($artistCat)
    {
        $this->artistCat = $artistCat;
    }
    public static function getOneByArtist($artistName)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = 'SELECT * FROM artists WHERE artistName=:artistName';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':artistName', $artistName, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
```

```
        } else {
            return null;
        }
    }
    public static function getArtistById($artistID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "SELECT artistName FROM artists WHERE artistID=$artistID";
        $statement = $connection->prepare($sql);
        $statement->bindParam(':artistID', $artistID, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function getArtistByArtistID($artistID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "SELECT * FROM artists WHERE artistID=$artistID";
        $statement = $connection->prepare($sql);
        $statement->bindParam(':artistID', $artistID, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function updateArtist($artistID,$artistName,$artistLabel,$artistBio, $artistCat)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE artists SET artistName = '$artistName', artistBio = '$artistBio', artistLabel
= '$artistLabel',
        artistCat = '$artistCat' WHERE artistID = $artistID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
```

```
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function updateArtists($artistID,$artistName,$artistLabel,$artistBio, $artistCat,
$artistImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE artists SET artistName = '$artistName', artistBio = '$artistBio', artistLabel
= '$artistLabel',
        artistCat = '$artistCat', artistImage = 'images/artists/$artistImage' WHERE artistID =
$artistID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function newArtist($artistName,$artistLabel,$artistBio, $artistCat, $artistImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "INSERT into artists (artistName, artistLabel, artistBio, artistCat, artistImage)
VALUES ('$artistName','$artistLabel','$artistBio','$artistCat','images/artists/$artistImage')";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function deleteArtist($artistID)
    {
        Single::deleteSinglesFromArtist($artistID);
        Album::deleteAllAlbumsFromArtist($artistID);
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "DELETE FROM artists WHERE artistID=$artistID";
        $numRowsAffected = $connection->exec($sql);
```

```
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
}
```

Cart Item Class

```
<?php
namespace Itb;
class CartItem
{
    private $album;
    private $song;
    private $quantity = 1;
    public function __construct($albumID, $type)
    {
        if ($type == 'Album') {
            $this->album = Album::getAlbumByAlbumID($albumID);
        } else {
            $this->song= Single::getSingleById($albumID);
        }
    }
    public function getQuantity()
    {
        return $this->quantity;
    }
    public function setQuantity($quantity)
    {
        $this->quantity = $quantity;
    }
    public function getAlbum()
    {
        return $this->album;
    }
    public function getSong()
    {
        return $this->song;
    }
}
```

Chart Class

```php
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class Chart extends DatabaseTable
{
    private $chartID;
    private $albumID;
    private $chartWeeks;
    public function getAlbumID()
    {
        return $this->albumID;
    }
    public function setAlbumID($albumID)
    {
        $this->albumID = $albumID;
    }
    public function getChartID()
    {
        return $this->chartID;
    }
    public function setChartID($chartID)
    {
        $this->chartID = $chartID;
    }
    public function getChartWeeks()
    {
        return $this->chartWeeks;
    }
    public function setChartWeeks($chartWeeks)
    {
        $this->chartWeeks = $chartWeeks;
    }
    public static function deleteAlbumFromChart($albumID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "DELETE FROM charts WHERE albumID=$albumID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
```

```
        return $queryWasSuccessful;
    }
}
```

Login Controller Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class LoginController
{
    public function logoutAction()
    {
        $_SESSION = [];
        if (ini_get('session.use_cookies')){
            $params = session_get_cookie_params();
            setcookie(
                session_name(),
                '',
                time() - 42000,
                $params['path'],
                $params['domain'],
                $params['secure'],
                $params['httponly']
            );
        }
        session_destroy();
        $mainController = new MainController();
        $mainController->indexAction();
    }
    public function processLoginAction()
    {
        // default is bad login
        $isLoggedIn = false;
        $isAdmin = false;
        $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);
        $password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
        $isLoggedIn = User::canFindMatchingUsernameAndPassword($username, $password);
        $role = '';
        $isAdmins = $this->isAdminUser($username);
        if($isLoggedIn){
            $_SESSION['user'] = $username;
            if ($isAdmins == User::ROLE_ADMIN){
                $_SESSION['admin'] = $role;
```

```
            $isAdmin = true;
        }else{
            $isAdmin = false;
        }
        $mainController = new MainController();
        $mainController->profileAction();
    } else {
        $pageTitle = "Incorrect Details";
        $message = 'Incorrect username or password, please try again';
        require_once __DIR__ . '/../templates/message.php';
    }
}
public function isLoggedInFromSession()
{
    $isLoggedIn = false;
    if(isset($_SESSION['user'])){
        $isLoggedIn = true;
    }
    return $isLoggedIn;
}
public static function isAdminUser($userName)
{
    $isAdmin = false;
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = "SELECT userRole FROM users WHERE userName='$userName'";
    $statement = $connection->query($sql);
    $row = $statement->fetchColumn();
    if ($row == User::ROLE_ADMIN) {
        $isAdmin = true;
    }
    return $isAdmin;
}
public function isAdminUserLoggedIn($username, $password)
{
    $isAdmin = false;
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = "SELECT role FROM users WHERE username='$username' AND
password='$password'";
    $statement = $connection->prepare($sql);
    $statement->fetch();
    $statement->execute();
    if ($row = $statement->fetch()) {
```

```
            return $row;
        } else {
            return null;
        }
    }
    public function usernameFromSession()
    {
        $username = '';
        if (isset($_SESSION['user'])) {
            $username = $_SESSION['user'];
        }
        return $username;
    }
    public function adminIsLoggedInFromSession()
    {
        $isAdmin = false;

        if(isset($_SESSION['admin'])){
            $isAdmin = true;
        }

        return $isAdmin;
    }
}
```

Main Controller Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseManager;
class MainController
{
    public function __construct()
    {
        $this->loginController = new LoginController();
        $this->searcController = new Search();
    }
    function chartAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
```

```
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Top 15 Chart';
        $chartLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/chart.php';
    }
    function contactAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Contact Us';
        $contactLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/contact.php';
    }
    function indexAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Home Page';
        $indexLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/index.php';
    }
    function songAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select1 = $this->getSongCategory();
        $select = $this->getCategory();
```

```
        $select1 = $this->getSongCategory();
        $pageTitle = 'Songs';
        $songLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/singles.php';
    }
    function artistAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Artists';
        $artistLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/artists.php';
    }
    function artistDetailAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Artist Details';
        $artistDetailsLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/artistDetails.php';
    }
    function albumAction()
    {
        $select1 = $this->getSongCategory();
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $pageTitle = 'Albums';
```

```
        $albumLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/albums.php';
    }
    function searchAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Search';
        if (isset($_POST['search']) || !empty($_POST['search'])) {
            $keyword = $_POST['search'];
        }
        $search = $this->searcController->getSearchResults($keyword);
        if($check = Artist::getOneByArtist($search) != null){
            $this->searchArtist($keyword);
        } else if (($check = Album::getOneByAlbum($keyword)) != null){
            $this->searchAlbum($keyword);
        } else if (($check = Single::getSingleByName($keyword)) != null){
            $this->searchSong($keyword);
        }else {
            $message = 'Sorry, no results for ' . $keyword . ' found';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    function profileAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $this  ->killSession();
        $pageTitle = 'Profile';
        $profileLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/profile.php';
    }
```

```
    function searchAlbum($albumName)
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $album = Album::getOneByAlbum($albumName);
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = "$albumName Details";
        $albumLinkStyle = 'nav-link active';
        if(null == $album){
            $message = 'Sorry, no Album with name = ' . $albumName . ' could be retrieved from
the database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/albumDetails.php';
        }
    }
    function searchSong($songName)
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $song = Single::getSingleByName($songName);
        $songID = $song->getSongID();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = "$songName Details";
        $songLinkStyle = 'nav-link active';
        if(null == $song){
            $message = 'Sorry, no Song with name = ' . $songName . ' could be retrieved from the
database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/singleDetails.php';
        }
    }
    function searchArtist($artistName)
```

```
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = "$artistName Details";
        $artistLinkStyle = 'nav-link active';
        $artist= Artist::getOneByArtist($artistName);
        if(null == $artist){
            $message = 'Sorry, no Artist with name = ' . $artistName . ' could be retrieved from the
database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/artistDetails.php';
        }
    }
    function showArtistAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $artistName = filter_input(INPUT_GET, 'artistName', FILTER_SANITIZE_STRING);
        $pageTitle = "$artistName Details";
        $artistLinkStyle = 'nav-link active';
        $artist= Artist::getOneByArtist($artistName);
        if(null == $artist){
            $message = 'Sorry, no Artist with name = ' . $artistName . ' could be retrieved from the
database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/artistDetails.php';
        }
    }
    function showAlbumAction()
    {
```

```
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $albumName = filter_input(INPUT_GET, 'albumName', FILTER_SANITIZE_STRING);
        $album = Album::getOneByAlbum($albumName);
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = "$albumName Details";
        $albumLinkStyle = 'nav-link active';
        if(null == $album){
            $message = 'Sorry, no Album with name = ' . $albumName . ' could be retrieved from
the database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/albumDetails.php';
        }
    }
    function showSingleAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $songID = filter_input(INPUT_GET, 'songID', FILTER_SANITIZE_STRING);
        $song = Single::getSingleById($songID);
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = "Song Details";
        $singleLinkStyle = 'nav-link active';
        if(null == $song){
            $message = 'Sorry, no Song with the ID = ' . $songID. ' could be retrieved from the
database';
            // output the detail of artist in HTML table
            require_once __DIR__ . '/../templates/message.php';
        } else {
            // output the detail of artist in HTML table
            require_once __DIR__ . '/../templates/singleDetails.php';
        }
    }
```

```
public function uploadImageAction()
{
    $target_path = "../public/images/userProfilePics";
    $id = filter_input(INPUT_POST, 'id', FILTER_SANITIZE_NUMBER_INT);
    $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
    $target_path = "../public/images/userProfilePics/";
    $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
    if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
        $image = new User();
        $images = basename( $_FILES['uploadedfile']['name']);
        $image->changeImage($id, $images);
        $mainController = new MainController();
        $mainController->profileAction();
    } else{
        $message = "There was an error uploading the file, please try again!";
        require_once __DIR__ . '/../templates/message.php';
    }
}
public function changeBackgroundColor($color) {
    $_SESSION['backgroundColor'] = $color;
    $this->profileAction();
}
public function getBackgroundColor()
{
    if (isset($_SESSION['backgroundColor'])){
        return $_SESSION['backgroundColor'];
    } else {
        return 'white';
    }
}
public function processRegisterAction()
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);
    $password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
    $firstName = filter_input(INPUT_POST, 'firstName', FILTER_SANITIZE_STRING);
    $lastName = filter_input(INPUT_POST, 'lastName', FILTER_SANITIZE_STRING);
    $email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_STRING);
    $success = User::addUser($username, $password,$firstName, $lastName, $email);
    if($success){
        $id = $connection->lastInsertId();
        $loginController = new LoginController();
        $loginController->processLoginAction();
```

```
        } else {
            $message = 'Sorry, there was a problem creating an account';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    function editProfileAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $pageTitle = 'Edit Profile';
        $editProfileLinkStyle = 'nav-link';
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
       // require_once __DIR__ . '/../templates/changePassword.php';
    }
    public function updateProfileAction()
    {
        $id = filter_input(INPUT_POST, 'userId', FILTER_SANITIZE_NUMBER_INT);
        $password = filter_input(INPUT_POST, 'userPassword', FILTER_SANITIZE_STRING);
        $success = new User();
        $success->changePassword($id, $password);
        if($success){
            $message = "Your Password has been changed";
        } else {
            $message = 'Change Password action canceled no changes took place';
        }
        $mainController = new MainController();
        $mainController->profileAction();
    }
    public function albumUpdateAction()
    {
        $albumID = filter_input(INPUT_GET, 'albumID', FILTER_SANITIZE_NUMBER_INT);
        $album = Album::getAlbumByAlbumID($albumID);
        $adminAlbumLinkStyle = 'nav-link active';
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
```

```
        $select1 = $this->getSongCategory();
        $pageTitle = 'Edit Album';
        if(null == $album){
            $message = 'Sorry, no Album with id = ' . $albumID . ' could be retrieved from the
database';

            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/CRUD/updateAlbum.php';
        }
    }
    public function updateAlbumAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Edit Albums';
        $adminAlbumLinkStyle = 'nav-link active';
        $albumID = filter_input(INPUT_POST, 'albumID', FILTER_SANITIZE_NUMBER_INT);
        $albumName = filter_input(INPUT_POST, 'albumName', FILTER_SANITIZE_STRING);
        $albumRelease = filter_input(INPUT_POST, 'albumRelease',
FILTER_SANITIZE_STRING);
        $albumVideo = filter_input(INPUT_POST, 'albumVideo', FILTER_SANITIZE_STRING);
        $albumCat = filter_input(INPUT_POST, 'albumCat', FILTER_SANITIZE_STRING);
        $albumPrice = filter_input(INPUT_POST, 'albumPrice',
FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_FRACTION);
        $artistID = filter_input(INPUT_POST, 'artistID', FILTER_SANITIZE_STRING);
        $albumImage = basename( $_FILES['uploadedfile']['name']);
        $success =
Album::updateAlbum($albumID,$albumName,$albumPrice,$albumRelease,$albumVideo,$artist
ID, $albumCat);
        if (null != $albumImage){
            $target_path = "../public/images/albums";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            $target_path = "../public/images/albums/";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
                $albumImage = basename( $_FILES['uploadedfile']['name']);
                $success =
```

```
Album::updateAlbums($albumID,$albumName,$albumPrice,$albumRelease,$albumVideo,$artis
tID,$albumImage,$albumCat);
            if($success){
                require_once __DIR__ . '/../templates/CRUD/albumCrud.php';
            } else {
                $message = 'There was a problem updating the Album';
            }
        } else{
            $message = "There was an error uploading the file, please try again!";
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    if($success){
        require_once __DIR__ . '/../templates/CRUD/albumCrud.php';
    }  else {
        $message = 'No changes took place, Action cancelled';
        require_once __DIR__ . '/../templates/message.php';
    }
}
public function deleteAlbumAction()
{
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
    $isAdmin = $this->loginController->isAdminUser($username);
    $backgroundColor = $this->getBackgroundColor();
    $albumCart = $this->getAlbumCart();
    $songCart = $this->getSongCart();
    $select = $this->getCategory();
    $select1 = $this->getSongCategory();
    $pageTitle = 'Edit Albums';
    $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
    $success = Album::deleteAlbum($id);
    if($success){
        $message = 'Album with id = ' . $id . ' was deleted successfully';
    } else {
        $message = 'Sorry, Album with id = ' . $id . ' could not be deleted';
    }

    require_once __DIR__ . '/../templates/CRUD/albumCrud.php';
}
public function showNewAlbumAction()
{
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
```

```
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $pageTitle = 'Add New Album';
        $adminAlbumLinkStyle = 'nav-link active';
        require_once __DIR__ . '/../templates/CRUD/newAlbum.php';
    }
    public function newAlbumAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Albums';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $albumName = filter_input(INPUT_POST, 'albumName', FILTER_SANITIZE_STRING);
        $albumRelease = filter_input(INPUT_POST, 'albumRelease',
FILTER_SANITIZE_STRING);
        $albumVideo = filter_input(INPUT_POST, 'albumVideo', FILTER_SANITIZE_STRING);
        $albumPrice = filter_input(INPUT_POST, 'albumPrice',
FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_FRACTION);
        $artistID = filter_input(INPUT_POST, 'artistID', FILTER_SANITIZE_STRING);
        $albumCat = filter_input(INPUT_POST, 'albumCat', FILTER_SANITIZE_STRING);
        $target_path = "../public/images/albums";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
        $target_path = "../public/images/albums/";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
        if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
            $albumImage = basename( $_FILES['uploadedfile']['name']);
            $success =
Album::newAlbum($albumName,$albumPrice,$albumRelease,$albumVideo,$artistID,$albumIm
age,$albumCat);

            if($success){
                require_once __DIR__ . '/../templates/CRUD/albumCrud.php';
            } else {
```

```
            $message = 'Sorry, there was a problem creating new Album';
        }
    } else{
        $message = "There was an error uploading the file, please try again!";
        require_once __DIR__ . '/../templates/message.php';
    }
}
public function artistUpdateAction()
{
    $artistID = filter_input(INPUT_GET, 'artistID', FILTER_SANITIZE_NUMBER_INT);
    $artist = Artist::getArtistByArtistID($artistID);
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
    $isAdmin = $this->loginController->isAdminUser($username);
    $albumCart = $this->getAlbumCart();
    $songCart = $this->getSongCart();
    $pageTitle = 'Edit Artist';
    $adminAlbumLinkStyle = 'nav-link active';
    $select = $this->getCategory();
    $select1 = $this->getSongCategory();
    if(null == $artist){
        $message = 'Sorry, no Artist with id = ' . $artistID . ' could be retrieved from the
database';

        require_once __DIR__ . '/../templates/message.php';
    } else {
        require_once __DIR__ . '/../templates/CRUD/updateArtist.php';
    }
}
public function updateArtistAction()
{
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
    $isAdmin = $this->loginController->isAdminUser($username);
    $backgroundColor = $this->getBackgroundColor();
    $albumCart = $this->getAlbumCart();
    $songCart = $this->getSongCart();
    $pageTitle = 'Edit Artist';
    $adminAlbumLinkStyle = 'nav-link active';
    $select = $this->getCategory();
    $select1 = $this->getSongCategory();
    $artistID = filter_input(INPUT_POST, 'artistID', FILTER_SANITIZE_NUMBER_INT);
    $artistName = filter_input(INPUT_POST, 'artistName', FILTER_SANITIZE_STRING);
    $artistBio = filter_input(INPUT_POST, 'artistBio', FILTER_SANITIZE_STRING);
```

```
        $artistLabel = filter_input(INPUT_POST, 'artistLabel', FILTER_SANITIZE_STRING);
        $artistCat = filter_input(INPUT_POST, 'artistCat', FILTER_SANITIZE_STRING);
        $artistImage = basename( $_FILES['uploadedfile']['name']);
        if (null != $artistImage) {
            $target_path = "../public/images/artists";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            $target_path = "../public/images/artists/";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            if (move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
                $artistImage = basename( $_FILES['uploadedfile']['name']);
                $success = Artist::updateArtists($artistID, $artistName, $artistLabel, $artistBio,
$artistCat, $artistImage);
                if ($success) {
                    $artistImage = null;
                    require_once __DIR__ . '/../templates/CRUD/artistCrud.php';
                } else {
                    $message = 'There was a problem updating the Artist';
                    require_once __DIR__ . '/../templates/message.php';
                }
            } else {
                $message = "There was an error uploading the file, please try again!";
                require_once __DIR__ . '/../templates/message.php';
            }
        }
        $success = Artist::updateArtist($artistID, $artistName, $artistLabel, $artistBio, $artistCat);
        if($success){
            require_once __DIR__ . '/../templates/CRUD/artistCrud.php';

        }  else {
            $message = 'No changes took place, Action cancelled';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function showNewArtistAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Add New Artist';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
```

```
        $select1 = $this->getSongCategory();
        require_once __DIR__ . '/../templates/CRUD/newArtist.php';
    }
    public function newArtistAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Artist';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $artistName = filter_input(INPUT_POST, 'artistName', FILTER_SANITIZE_STRING);
        $artistBio = filter_input(INPUT_POST, 'artistBio', FILTER_SANITIZE_STRING);
        $artistLabel = filter_input(INPUT_POST, 'artistLabel', FILTER_SANITIZE_STRING);
        $artistCat = filter_input(INPUT_POST, 'artistCat', FILTER_SANITIZE_STRING);
        $artistImage = $_FILES['uploadedfile'];
        $target_path = "../public/images/artists";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
        $target_path = "../public/images/artists/";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
        if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
            $artistImage = basename( $_FILES['uploadedfile']['name']);
            $success = Artist::newArtist($artistName, $artistLabel, $artistBio, $artistCat,
$artistImage);
            if($success){
                require_once __DIR__ . '/../templates/CRUD/artistCrud.php';
            } else {
                $message = 'Sorry, there was a problem creating new Artist';
                require_once __DIR__ . '/../templates/message.php';
            }
        } else{
            $message = "There was an error uploading the file, please try again!";
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function deleteArtistAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
```

```
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Artists';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
        $success = Artist::deleteArtist($id);
        if($success){
            $message = 'Artist with id = ' . $id . ' was deleted successfully';
            require_once __DIR__ . '/../templates/CRUD/artistCrud.php';
        } else {
            $message = 'Sorry, Artist with id = ' . $id . ' could not be deleted';
            require_once __DIR__ . '/../templates/message.php';
        }
    }

    public function singleUpdateAction()
    {
        $songID = filter_input(INPUT_GET, 'songID', FILTER_SANITIZE_NUMBER_INT);
        $song = Single::getSingleById($songID);
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Single';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        if(null == $song){
            $message = 'Sorry, no Single with id = ' . $songID . ' could be retrieved from the
database';
            require_once __DIR__ . '/../templates/message.php';
        } else {
            require_once __DIR__ . '/../templates/CRUD/updateSingle.php';
        }
    }
    public function updateSingleAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
```

```
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Single';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $songID = filter_input(INPUT_POST, 'songID', FILTER_SANITIZE_NUMBER_INT);
        $songName = filter_input(INPUT_POST, 'songName', FILTER_SANITIZE_STRING);
        $songCat = filter_input(INPUT_POST, 'songCat', FILTER_SANITIZE_STRING);
        $songTrackNo = filter_input(INPUT_POST, 'songTrackNo', FILTER_SANITIZE_STRING);
        $songWriter = filter_input(INPUT_POST, 'songWriter', FILTER_SANITIZE_STRING);
        $songLength = filter_input(INPUT_POST, 'songLength', FILTER_SANITIZE_STRING);
        $songPrice = filter_input(INPUT_POST, 'songPrice',
FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_FRACTION);
        $artistID = filter_input(INPUT_POST, 'artistID', FILTER_SANITIZE_NUMBER_INT);
        $albumID = filter_input(INPUT_POST, 'albumID', FILTER_SANITIZE_NUMBER_INT);
        $songImage = basename( $_FILES['uploadedfile']['name']);
        if (null != $songImage) {
            $target_path = "../public/images/singles";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            $target_path = "../public/images/singles/";
            $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
            if (move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
                $songImage = basename( $_FILES['uploadedfile']['name']);
                $success = Single::updateSingles($songID, $songName, $songCat, $songTrackNo,
$songWriter, $songLength, $songPrice,$artistID,$albumID, $songImage);
                if ($success) {
                  // $songImage = null;
                    require_once __DIR__ . '/../templates/CRUD/singleCrud.php';
                } else {
                    $message = 'There was a problem updating the Single';
                    require_once __DIR__ . '/../templates/message.php';
                }
            } else {
                $message = "There was an error uploading the file, please try again!";
                require_once __DIR__ . '/../templates/message.php';
            }
        }
        $success = Single::updateSingle($songID, $songName, $songCat, $songTrackNo,
$songWriter, $songLength, $songPrice,$artistID,$albumID);
        if($success){
```

```
            require_once __DIR__ . '/../templates/CRUD/singleCrud.php';
        } else {
            $message = 'No changes took place, Action cancelled';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function showNewSingleAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Add New Single';
        $adminAlbumLinkStyle = 'nav-link active';
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        require_once __DIR__ . '/../templates/CRUD/newSingle.php';
    }
    public function newSingleAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Single';
        $adminAlbumLinkStyle = 'nav-link active';
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $songName = filter_input(INPUT_POST, 'songName', FILTER_SANITIZE_STRING);
        $songCat = filter_input(INPUT_POST, 'songCat', FILTER_SANITIZE_STRING);
        $songTrackNo = filter_input(INPUT_POST, 'songTrackNo', FILTER_SANITIZE_STRING);
        $songWriter = filter_input(INPUT_POST, 'songWriter', FILTER_SANITIZE_STRING);
        $songLength = filter_input(INPUT_POST, 'songLength', FILTER_SANITIZE_STRING);
        $songPrice = filter_input(INPUT_POST, 'songPrice',
FILTER_SANITIZE_NUMBER_FLOAT, FILTER_FLAG_ALLOW_FRACTION);
        $artistID = filter_input(INPUT_POST, 'artistID', FILTER_SANITIZE_NUMBER_INT);
        $albumID = filter_input(INPUT_POST, 'albumID', FILTER_SANITIZE_NUMBER_INT);
        $songImage = $_FILES['uploadedfile'];
        $target_path = "../public/images/singles";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
```

```
        $target_path = "../public/images/singles/";
        $target_path = $target_path . basename( $_FILES['uploadedfile']['name']);
        if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
            $songImage = basename( $_FILES['uploadedfile']['name']);
            $success = Single::newSingle($songName, $songCat, $songTrackNo, $songWriter,
$songLength, $songPrice,$artistID,$albumID, $songImage);
            if($success){
                require_once __DIR__ . '/../templates/CRUD/singleCrud.php';
            } else {
                $message = 'Sorry, there was a problem creating new Single';
                require_once __DIR__ . '/../templates/message.php';
            }
        } else{
            $message = "There was an error uploading the file, please try again!";
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function deleteSingleAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $pageTitle = 'Edit Single';
        $adminAlbumLinkStyle = 'nav-link active';
        $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
        $success = Single::deleteSingle($id);
        if($success){
            require_once __DIR__ . '/../templates/CRUD/singleCrud.php';
        } else {
            $message = 'Sorry, Artist with id = ' . $id . ' could not be deleted';
            require_once __DIR__ . '/../templates/message.php';
        }
    }
    public function getAlbumCart()
    {
        if (isset($_SESSION['albumCart'])){
            return $_SESSION['albumCart'];
        } else {
            return [];
        }
    }
```

```
public function getSongCart()
{
    if (isset($_SESSION['songCart'])){
        return $_SESSION['songCart'];
    } else {
        return [];
    }
}
public function addAlbumToCart()
{
    $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
    $type = filter_input(INPUT_GET, 'type', FILTER_SANITIZE_STRING);
    $select = $this->getCategory();
    $albumCart = $this->getAlbumCart();
    $cartItem = new CartItem($id, $type);
    if(isset($albumCart[$id])){
        $cartItem = $albumCart[$id];
        $oldQuantity = $cartItem->getQuantity();
        $newQuantity = $oldQuantity + 1;
        $cartItem->setQuantity($newQuantity);
    }
    $albumCart[$id] = $cartItem;
    $_SESSION['albumCart'] = $albumCart;
    $this->shopAction();
}
public function addSongToCart()
{
    // get the ID of product to add to cart
    $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
    $type = filter_input(INPUT_GET, 'type', FILTER_SANITIZE_STRING);
    $select1 = $this->getSongCategory();
    $songCart = $this->getSongCart();
    $songCartItem = new CartItem($id, $type);
    if(isset($songCart[$id])){
        $songCartItem = $songCart[$id];
        $oldQuantity = $songCartItem->getQuantity();
        $newQuantity = $oldQuantity + 1;
        $songCartItem->setQuantity($newQuantity);
    }
    $songCart[$id] = $songCartItem;
    $_SESSION['songCart'] = $songCart;
    $this->songAction();
}
public function addAlbumSongToCart()
```

```
{
    $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
    $albumName = filter_input(INPUT_GET, 'albumName', FILTER_SANITIZE_STRING);
    $type = filter_input(INPUT_GET, 'type', FILTER_SANITIZE_STRING);
    $select1 = $this->getSongCategory();
    $songCart = $this->getSongCart();
    $songCartItem = new CartItem($id, $type);
    if(isset($songCart[$id])){
        $songCartItem = $songCart[$id];
        $oldQuantity = $songCartItem->getQuantity();
        $newQuantity = $oldQuantity + 1;
        $songCartItem->setQuantity($newQuantity);
    }
    $songCart[$id] = $songCartItem;
    $_SESSION['songCart'] = $songCart;
    $this->showAlbumAction();
}
function shopAction()
{
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
    $isAdmin = $this->loginController->isAdminUser($username);
    $backgroundColor = $this->getBackgroundColor();
    $pageTitle = 'Albums';
    $albumLinkStyle = 'nav-link active';
    $albumCart = $this->getAlbumCart();
    $songCart = $this->getSongCart();
    $select = $this->getCategory();
    $select1 = $this->getSongCategory();
    $albums = Album::getAll();
    $singles = Single::getAll();
    require_once __DIR__ . '/../templates/albums.php';
}
function showCartAction()
{
    $isLoggedIn = $this->loginController->isLoggedInFromSession();
    $username = $this->loginController->usernameFromSession();
    $isAdmin = $this->loginController->isAdminUser($username);
    $backgroundColor = $this->getBackgroundColor();
    $select = $this->getCategory();
    $pageTitle = 'View Cart';
    $albumCart = $this->getAlbumCart();
    $songCart = $this->getSongCart();
    $select = $this->getCategory();
```

```
        $select1 = $this->getSongCategory();
        $albums = Album::getAll();
        $singles = Single::getAll();
        require_once __DIR__ . '/../templates/cart.php';
    }

    public function forgetSession()
    {
        $this->killSession();

        // redirect to shop
        $this->indexAction();
    }
    public function killSession()
    {
        unset($_SESSION["albumCart"]);
        unset($_SESSION["songCart"]);
    }
    public function removeAlbumFromCart()
    {
        $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
        $albumCart = $this->getAlbumCart();
        unset($albumCart[$id]);
        $_SESSION['albumCart'] = $albumCart;
        $this->showCartAction();
    }
    public function removeSongFromCart()
    {
        $id = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT);
        $songCart = $this->getSongCart();
        unset($songCart[$id]);
        $_SESSION['songCart'] = $songCart;
        $this->showCartAction();
    }
    public function albumCategory($cat) {
        $_SESSION['select1'] = $cat;
        $this->albumAction();
    }
    public function getCategory()
    {
        if (isset($_SESSION['select1'])){
            return $_SESSION['select1'];
        } else {
            return 'Filter By Category';
```

```
        }
    }
    public function setCat()
    {
        $select = $_POST["select1"];
        if($select == null){
            $select = 'Filter By Category';
        }
        $this->albumCategory($select);
    }
    public function setSongCat()
    {
        $select1 = $_POST["select"];
        $this->songCategory($select1);
    }
    public function songCategory($cat) {
        $_SESSION['select'] = $cat;
        $this->songAction();
    }
    public function getSongCategory()
    {
        if (isset($_SESSION['select'])){
            return $_SESSION['select'];
        } else {
            return 'Filter By Category';
        }
    }

    public function checkOutAction()
    {
        $total = filter_input(INPUT_GET, 'total', FILTER_SANITIZE_NUMBER_FLOAT,
FILTER_FLAG_ALLOW_FRACTION);
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $select = $this->getCategory();
        $pageTitle = 'Checkout';
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        require_once __DIR__ . '/../templates/checkout.php';
    }
```

```
    public function purchaseAction()
    {
        $this->killSession();
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $select = $this->getCategory();
        $pageTitle = 'Thank You For Your Purchase';
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        require_once __DIR__ . '/../templates/purchase.php';
    }
    public function downloadAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $select = $this->getCategory();
        $pageTitle = 'Thank You For Your Purchase';
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        require_once __DIR__ . '/../templates/download.php';
    }

    public function contactMsgAction()
    {
        $isLoggedIn = $this->loginController->isLoggedInFromSession();
        $username = $this->loginController->usernameFromSession();
        $isAdmin = $this->loginController->isAdminUser($username);
        $backgroundColor = $this->getBackgroundColor();
        $select = $this->getCategory();
        $albumCart = $this->getAlbumCart();
        $songCart = $this->getSongCart();
        $select = $this->getCategory();
        $select1 = $this->getSongCategory();
        $message = 'Thank you for your for contacting us, we will reply as soon as possible';
        require_once __DIR__ . '/../templates/message.php';
```

```
    }
}
```

Purchase Album Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class PurchaseAlbum extends DatabaseTable
{
    private $purchaseAlbumID;
    private $albumID;
    private $userID;
    public function getPurchaseAlbumID()
    {
        return $this->purchaseAlbumID;
    }
    public function setPurchaseAlbumID($purchaseAlbumID)
    {
        $this->purchaseAlbumID = $purchaseAlbumID;
    }
    public function getAlbumID()
    {
        return $this->albumID;
    }
    public function setAlbumID($albumID)
    {
        $this->albumID = $albumID;
    }
    public function getUserID()
    {
        return $this->userID;
    }
    public function setUserID($userID)
    {
        $this->userID = $userID;
    }
    public static function newArtist($albumID, $userID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "INSERT into purchaseAlbums (albumID, userID) VALUES (3,3)";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
```

```
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
}
```

Search Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class Search extends DatabaseTable
{
    public $type;
    public static function getSearchResults($keyword)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "(SELECT artistName, 'art' as type FROM artists WHERE artistName LIKE
'$keyword')
        UNION
        (SELECT albumName, 'alb' as type FROM albums WHERE albumName LIKE '$keyword')
        UNION
        SELECT songName, 'son' as type FROM singles WHERE songName LIKE '$keyword'";
        $statement = $connection->prepare($sql);
        $statement->setFetchMode(\PDO::FETCH_CLASS, '\\' .  static::class);
        $statement->execute();
        $objects = $statement->fetchColumn();
        return $objects;
    }
}
```

Single Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class Single extends DatabaseTable
{
    private $songID;
    private $songTrackNo;
    private $songName;
```

```
    private $songWriter;
    private $songLength;
    private $songImage;
    private $songPrice;
    private $songCat;
    private $artistID;
    private $albumID;
    public function getSongCat()
    {
        return $this->songCat;
    }
    public function setSongCat($songCat)
    {
        $this->songCat = $songCat;
    }
    public function getArtistID()
    {
        return $this->artistID;
    }
    public function setArtistID($artistID)
    {
        $this->artistID = $artistID;
    }
    public function getAlbumID()
    {
        return $this->albumID;
    }
    public function setAlbumID($albumID)
    {
        $this->albumID = $albumID;
    }
    public function getSongID()
    {
        return $this->songID;
    }
    public function setSongID($songID)
    {
        $this->songID = $songID;
    }
    public function getSongTrackNo()
    {
        return $this->songTrackNo;
    }
    public function setSongTrackNo($songTrackNo)
```

```
    {
        $this->songTrackNo = $songTrackNo;
    }
    public function getSongName()
    {
        return $this->songName;
    }
    public function setSongName($songName)
    {
        $this->songName = $songName;
    }
    public function getSongWriter()
    {
        return $this->songWriter;
    }
    public function setSongWriter($songWriter)
    {
        $this->songWriter = $songWriter;
    }
    public function getSongLength()
    {
        return $this->songLength;
    }
    public function setSongLength($songLength)
    {
        $this->songLength = $songLength;
    }
    public function getSongImage()
    {
        return $this->songImage;
    }
    public function setSongImage($songImage)
    {
        $this->songImage = $songImage;
    }
    public function getSongPrice()
    {
        return $this->songPrice;
    }
    public function setSongPrice($songPrice)
    {
        $this->songPrice = $songPrice;
    }
    public static function getSinglesByAlbum($albumID)
```

```
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = 'SELECT * FROM singles WHERE albumID=:albumID';
    $statement = $connection->prepare($sql);
    $statement->bindParam(':albumID', $albumID, \PDO::PARAM_STR);
    $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
    $statement->execute();
    if ($object = $statement->fetchAll()) {
        return $object;
    } else {
        return null;
    }
}
public static function getAllSinglesInOrder()
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = 'SELECT * FROM singles ORDER BY songname';
    $statement = $connection->prepare($sql);
    $statement->setFetchMode(\PDO::FETCH_CLASS, '\\' .  static::class);
    $statement->execute();
    $objects = $statement->fetchAll();
    return $objects;
}
public static function deleteSinglesFromAlbum($albumID)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = "DELETE FROM singles WHERE albumID=$albumID";
    $numRowsAffected = $connection->exec($sql);
    if($numRowsAffected > 0){
        $queryWasSuccessful = true;
    } else {
        $queryWasSuccessful = false;
    }
    return $queryWasSuccessful;
}
public static function deleteSinglesFromArtist($artistID)
{
    $db = new DatabaseManager();
    $connection = $db->getDbh();
    $sql = "DELETE FROM singles WHERE artistID=$artistID";
    $numRowsAffected = $connection->exec($sql);
```

```
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function getSingleById($songID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "SELECT * FROM singles WHERE songID=$songID";
        $statement = $connection->prepare($sql);
        $statement->bindParam(':songID', $songID, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function getSingleByName($songName)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql ='SELECT * FROM singles WHERE songName=:songName';
        $statement = $connection->prepare($sql);
        $statement->bindParam(":songName", $songName, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function updateSingle($songID, $songName, $songCat, $songTrackNo,
$songWriter, $songLength, $songPrice,$artistID,$albumID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE singles SET songName = '$songName', songCat = '$songCat',
songTrackNo = $songTrackNo, albumID = $albumID,
```

```
        songCat = '$songCat',songWriter = '$songWriter',songLength = '$songLength',songPrice =
$songPrice, artistID = $artistID WHERE songID = $songID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function updateSingles($songID, $songName, $songCat, $songTrackNo,
$songWriter, $songLength, $songPrice,$artistID,$albumID, $songImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE singles SET songName = '$songName', songCat = '$songCat',
songTrackNo = $songTrackNo, albumID = $albumID, songImage =
'images/singles/$songImage',
        songCat = '$songCat',songWriter = '$songWriter',songLength = '$songLength',songPrice =
$songPrice, artistID = $artistID WHERE songID = $songID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function newSingle($songName, $songCat, $songTrackNo, $songWriter,
$songLength, $songPrice,$artistID,$albumID, $songImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "INSERT into singles (songName, songPrice, songTrackNo, songCat, songWriter,
songLength, artistID, albumID, songImage) VALUES
('$songName',$songPrice,$songTrackNo,'$songCat','$songWriter','$songLength',$artistID,$albu
mID,'images/singles/$songImage')";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
```

```
    }
    public static function deleteSingle($songID)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "DELETE FROM singles WHERE songID=$songID";
        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
}
```

User Class

```
<?php
namespace Itb;
use Mattsmithdev\PdoCrud\DatabaseTable;
use Mattsmithdev\PdoCrud\DatabaseManager;
class User extends DatabaseTable
{
    const ROLE_USER = 1;
    const ROLE_ADMIN = 2;
    private $userId;
    private $userName;
    private $userPassword;
    private $userRole;
    private $userImage;
    private $userEmail;
    private $firstName;
    private $lastName;
    public function getFirstName()
    {
        return $this->firstName;
    }
    public function setFirstName($firstName)
    {
        $this->firstName = $firstName;
    }
    public function getLastName()
    {
        return $this->lastName;
```

```
}
public function setLastName($lastName)
{
    $this->lastName = $lastName;
}
public function getUserEmail()
{
    return $this->userEmail;
}
public function setEmail($userEmail)
{
    $this->userEmail = $userEmail;
}
public function getUserId()
{
    return $this->userId;
}
public function setUserId($userId)
{
    $this->userId = $userId;
}
public function getUserName()
{
    return $this->userName;
}
public function setUsername($userName)
{
    $this->userName = $userName;
}
public function getUserPassword()
{
    return $this->userPassword;
}
public function setPassword($userPassword)
{
    $hashedPassword = password_hash($userPassword, PASSWORD_DEFAULT);
    $this->userPassword  = $hashedPassword;
}
public function getUserRole()
{
    return $this->userRole;
}
public function setRole($userRole)
{
```

```
        $this->userRole = $userRole;
    }
    public function getUserImage()
    {
        return $this->userImage;
    }
    public function setImage($userImage)
    {
        $this->userImage = $userImage;
    }
    public static function canFindMatchingUsernameAndPassword($userName,
$userPassword)
    {
        $user = User::getOneByUsername($userName);
        if(null == $user){
            return false;
        }
        $hashedPassword = $user->getUserPassword();
        $passwordWasVerified = password_verify($userPassword, $hashedPassword);
        return $passwordWasVerified;
    }
    public static function getOneByUsername($userName)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = 'SELECT * FROM users WHERE userName=:userName';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':userName', $userName, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $object;
        } else {
            return null;
        }
    }
    public static function addUser($username, $password, $firstName, $lastName, $email)
    {
        $register = new User();
        $register->setUsername($username);
        $register->setPassword($password);
        $register->setFirstName($firstName);
        $register->setLastName($lastName);
        $register->setEmail($email);
```

```
        $register->setRole(1);
        $register->setImage('/images/userProfilePics/user.ico');
        self::insert($register);
        if(null == $register){
            return false;
        }
        return true;
    }
    public function changeImage($userId, $userImage)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        $sql = "UPDATE users SET userImage = '/images/userProfilePics/$userImage' WHERE
userId=$userId";

        $numRowsAffected = $connection->exec($sql);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;// = true;
    }
    public static function checkRole($userName)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        //$this->getRole();
        $sql = 'SELECT userRole FROM users WHERE userName=$userName';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':userName', $userName, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetch()) {
            return $sql;
        } else {
            return null;
        }
    }
    public function changePassword($userId, $userPassword)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        self::setPassword($userPassword);
```

```
        $hashedPassword = $this->getUserPassword();
        $sql = "UPDATE users SET userPassword = '$hashedPassword' WHERE
userId='$userId'";
        $numRowsAffected = $connection->exec($sql);
        $queryWasSuccessful = ($numRowsAffected > 0);
        if($numRowsAffected > 0){
            $queryWasSuccessful = true;
        } else {
            $queryWasSuccessful = false;
        }
        return $queryWasSuccessful;
    }
    public static function getUserImageByUserName($userName)
    {
        $db = new DatabaseManager();
        $connection = $db->getDbh();
        //$this->getRole();
        $sql = 'SELECT userImage FROM users WHERE userName=:userName';
        $statement = $connection->prepare($sql);
        $statement->bindParam(':userName', $userName, \PDO::PARAM_STR);
        $statement->setFetchMode(\PDO::FETCH_CLASS, __CLASS__);
        $statement->execute();
        if ($object = $statement->fetchColumn()) {
            return $object;
        } else {
            return null;
        }
    }
}
```

Index Page (Front Loader)

```
<?php
require_once __DIR__ . '/../vendor/autoload.php';
require_once __DIR__ . '/app/config_db.php';
session_start();
use Itb\MainController;
use Itb\LoginController;
$action = filter_input(INPUT_GET, 'action', FILTER_SANITIZE_STRING);
$mainController = new MainController;
$loginController = new LoginController();
$adminController = new \Itb\AdminController();
switch ($action){
    case 'chart' :
        $mainController->chartAction();
```

```
        break;
    case 'contact' :
        $mainController->contactAction();
        break;
    case 'song' :
        $mainController->songAction();
        break;
    case 'show' :
        $mainController->showArtistAction();
        break;
    case 'artist' :
        $mainController->artistAction();
        break;
    case 'artistDetails' :
        $mainController->artistDetailAction();
        break;
    case 'showAlbum' :
        $mainController->showAlbumAction();
        break;
    case 'showSong' :
        $mainController->showSingleAction();
        break;
    case 'albums' :
        $mainController->albumAction();
        break;
    case 'find' :
        $mainController->searchAction();
        break;
    case 'processLogin':
        $loginController->processLoginAction();
        break;
    case 'editProfile':
        $mainController->editProfileAction();
        break;
    case 'logout':
        $loginController->logoutAction();
        break;
    case 'profile':
        $mainController->profileAction();
        break;
    case 'imageUpload':
        $mainController->uploadImageAction();
        break;
    case 'setBackgroundColorBlue':
```

```
        $mainController->changeBackgroundColor('#ccffff');
        break;
    case 'setBackgroundColorGreen':
        $mainController->changeBackgroundColor('#f2ffe6');
        break;
    case 'setBackgroundColorDefault':
        $mainController->changeBackgroundColor('#ffffff');
        break;
    case 'processRegister':
        $mainController->processRegisterAction();
        break;
    case 'updateProfile':
        $mainController->updateProfileAction();
        break;
    case 'adminAlbum':
        $adminController->adminAlbumAction();
        break;
    case 'adminArtist':
        $adminController->adminArtistAction();
        break;
    case 'adminSingle':
        $adminController->adminSingleAction();
        break;
    case 'albumUpdate':
        $mainController->albumUpdateAction();
        break;
    case 'updateAlbum':
        $mainController->updateAlbumAction();
        break;
    case 'deleteAlbum':
        $mainController->deleteAlbumAction();
        break;
    case 'showNewAlbum':
        $mainController->showNewAlbumAction();
        break;
    case 'newAlbum':
        $mainController->newAlbumAction();
        break;
    case 'artistUpdate':
        $mainController->artistUpdateAction();
        break;
    case 'updateArtist':
        $mainController->updateArtistAction();
        break;
```

```
    case 'showNewArtist':
        $mainController->showNewArtistAction();
        break;
    case 'newArtist':
        $mainController->newArtistAction();
        break;
    case 'deleteArtist':
        $mainController->deleteArtistAction();
        break;
    case 'singleUpdate':
        $mainController->singleUpdateAction();
        break;
    case 'updateSingle':
        $mainController->updateSingleAction();
        break;
    case 'showNewSingle':
        $mainController->showNewSingleAction();
        break;
    case 'newSingle':
        $mainController->newSingleAction();
        break;
    case 'deleteSingle':
        $mainController->deleteSingleAction();
        break;
    case 'addToCart':
        $mainController->addAlbumToCart();
        break;
    case 'removeAlbumFromCart':
        $mainController->removeAlbumFromCart();
        break;
    case 'removeSongFromCart':
        $mainController->removeSongFromCart();
        break;
    case 'emptyCart':
        $mainController->forgetSession();
        break;
    case 'showCart':
        $mainController->showCartAction();
        break;
    case 'addSongToCart':
        $mainController->addSongToCart();
        break;
    case 'addAlbumSongToCart':
        $mainController->addAlbumSongToCart();
```

```
        break;
    case 'setCat':
        $mainController->setCat();
        break;
    case 'setSongCat':
        $mainController->setSongCat();
        break;
    case 'checkOut':
        $mainController->checkOutAction();
        break;
    case 'purchase':
        $mainController->purchaseAction();
        break;
    case 'download':
        $mainController->downloadAction();
        break;
    case 'contactMsg' :
        $mainController->contactMsgAction();
        break;
    default:
        // default is home page ('index' action)
        $mainController->indexAction();
}
```

User Seed Page (Sets up initial admin users)

```
<?php
session_start();
require_once __DIR__ . '/../vendor/autoload.php';
use Itb\User;
define('DB_HOST','localhost');
define('DB_USER', 'root');
define('DB_PASS', 'root');
define('DB_NAME', 'gerrys_db');
$derek = new User();
$derek->setFirstName('Derek');
$derek->setLastName('McCarthy');
$derek->setUsername('derek');
$derek->setPassword('derek');
$derek->setImage('images/userProfilePics/user.ico');
$derek->setEmail('derek@gerrys.com');
$derek->setRole(User::ROLE_ADMIN);
$joey = new User();
$joey->setFirstName('Joseph');
$joey->setLastName('Tierney');
```

```
$joey->setUsername('joey');
$joey->setPassword('joey');
$joey->setImage('images/userProfilePics/user.ico');
$joey->setEmail('joey@gerrys.com');
$joey->setRole(User::ROLE_ADMIN);
$chris = new User();
$chris->setFirstName('Christopher');
$chris->setLastName('Slattery');
$chris->setUsername('chris');
$chris->setPassword('chris');
$chris->setImage('images/userProfilePics/user.ico');
$chris->setEmail('chris@gerrys.com');
$chris->setRole(User::ROLE_ADMIN);
User::insert($derek);
User::insert($joey);
User::insert($chris);
$users = \Itb\User::getAll();
var_dump($users);
?>
```

_checkOutCart Template

```
<?php
$total = 0;
$songTotal = 0;
   foreach ($albumCart as $cartItem):
      $album = $cartItem->getAlbum();
      $subTotal = $album->getAlbumPrice() * $cartItem->getQuantity();
      $total += $subTotal;
   endforeach;
   foreach ($songCart as $cartIte):
      $song = $cartIte->getSong();
      $songSubTotal = $song->getSongPrice() * $cartIte->getQuantity();
      $songTotal += $songSubTotal;
   endforeach;
$total += $songTotal;
?>
&euro; <?= $total ?>
```

_footer Template

```
</article>
<footer class="container-fluid text-center">
   <small>Website by
      Christopher Slattery,
```

```
        Joseph Tierney,
        Derek McCarthy
        &copy; 2017</small>
</footer>
</body>
</html>
```

_head Template

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title><?= $pageTitle ?></title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="Music website to purchase music">
    <meta name="author" content="Christopher Slattery,Joseph Tierney,Derek McCarthy">
    <link rel="icon" href="../images/favicon.ico">
    <script defer src="https://use.fontawesome.com/releases/v5.0.6/js/all.js"></script>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
alpha.6/css/bootstrap.min.css" integrity="sha384-
rwoIResjU2yc3z8GV/NPeZWAv56rSmLldC3R/AZzGRnGxQQKnKkoFVhFQhNUwEyJ"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.1.1.slim.min.js" integrity="sha384-
A7FZj7v+d/sdmMqp/nOQwliLvUsJfDHW+k9Omg/a/EheAdgtzNs3hpfag6Ed950n"
crossorigin="anonymous"></script>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/tether/1.4.0/js/tether.min.js"
integrity="sha384-
DztdAPBWPRXSA/3eYEEUWrWCy7G5KFbe8fFjk5JAIxUYHKkDx6Qin1DkWx51bBrb"
crossorigin="anonymous"></script>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-alpha.6/js/bootstrap.min.js"
integrity="sha384-
vBWWzlZJ8ea9aCX4pEW3rVHjgjt7zpkNpZk+02D9phzyeVkE+jo0ieGizqPLForn"
crossorigin="anonymous"></script>
    <script src="/assets/js/bootstrap.min.js"></script>
    <link rel="stylesheet" href="/css/style.css">
    <style>
        @import "/../public/css/style.css";
    </style>
    <style>
        @import url('https://fonts.googleapis.com/css?family=Roboto');
    </style>
</head>
```

_nav Template

```
<?php
$indexLinkStyle = isset($indexLinkStyle) ? $indexLinkStyle : 'nav-link';
$chartLinkStyle = isset($chartLinkStyle) ? $chartLinkStyle : 'nav-link';
$contactLinkStyle = isset($contactLinkStyle) ? $contactLinkStyle : 'nav-link';
$artistLinkStyle = isset($artistLinkStyle) ? $artistLinkStyle : 'nav-link';
$albumLinkStyle = isset($albumLinkStyle) ? $albumLinkStyle : 'nav-link';
$profileLinkStyle = isset($profileLinkStyle) ? $profileLinkStyle : 'nav-link';
$adminAlbumLinkStyle = isset($adminAlbumLinkStyle) ? $adminAlbumLinkStyle : 'nav-link';
$adminArtistLinkStyle = isset($adminArtistLinkStyle) ? $adminArtistLinkStyle : 'nav-link';
$adminSingleLinkStyle = isset($adminSingleLinkStyle) ? $adminSingleLinkStyle : 'nav-link';
$songLinkStyle = isset($songLinkStyle) ? $songLinkStyle : 'nav-link';
$image = \ltb\User::getUserImageByUserName($username);
?>
<body>
<style>
   body{
      background-color: <?= $backgroundColor ?>;
   }
   .navbar {
      background-color: <?= $backgroundColor ?>;
   }
</style>
<nav class="navbar navbar-toggleable-md navbar-light bg-faded nav-tabs">
   <button class="navbar-toggler navbar-toggler-right" type="button" data-toggle="collapse"
data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
      <span class="navbar-toggler-icon"></span>
   </button>
   <a class="navbar-brand" href="index.php"><img src="../images/gerrys.png"
alt="GerrysJukebox"></a>
   <div class="collapse navbar-collapse" id="navbarSupportedContent">
      <ul class="navbar-nav mr-auto">
         <li class="nav-item">
            <a href='index.php' class="<?= $indexLinkStyle ?>"><i class="fas fa-home"></i>
Home</a>
         </li>
         <li class="nav-item">
            <a href="index.php?action=song" class="<?= $songLinkStyle ?>"><i class="fas fa-
music"></i> Singles</a>
         </li>
         <li class="nav-item">
            <a href='index.php?action=albums' class="<?= $albumLinkStyle ?>"><i class="fas fa-
play"></i> Albums</a>
         </li>
```

```
        <li class="nav-item">
            <a href='index.php?action=artist' class="<?= $artistLinkStyle ?>"><i class="fas fa-
users"></i> Artists</a>
        </li>
        <li class="nav-item">
            <a href='index.php?action=chart' class="<?= $chartLinkStyle ?>"><i class="fas fa-
chart-bar"></i> Top 15 Chart</a>
        </li>
        <li class="nav-item">
            <a href='index.php?action=contact' class="<?= $contactLinkStyle ?>"><i class="fab
fa-wpforms"></i> Contact</a>
        </li>
        <?php
        if($isLoggedIn):?>
            <li class="nav-item">
                <a href='index.php?action=profile' class="<?= $profileLinkStyle ?>"><i class="fas
fa-address-card"></i> Profile</a>
            </li>
        <?php endif; ?>
        <?php if($isAdmin): ?>
            <li class="nav-item dropdown">
                <a class="<?= $adminAlbumLinkStyle ?> dropdown-toggle"
href="http://example.com" id="navbarDropdownMenuLink" data-toggle="dropdown" aria-
haspopup="true" aria-expanded="false">
                    <i class="fas fa-unlock-alt"></i> Admin
                </a>
                <div class="dropdown-menu" aria-labelledby="navbarDropdownMenuLink">
                    <a class="dropdown-item" href='index.php?action=adminAlbum'>Edit
Albums</a>
                    <a class="dropdown-item" href='index.php?action=adminArtist'>Edit Artist</a>
                    <a class="dropdown-item" href='index.php?action=adminSingle'>Edit
Singles</a>
                </div>
            </li>
        <?php endif; ?>
    </ul>
    <ul class="navbar-nav ml-auto">
        <li>
            <form class="form-inline my-2 my-lg-0 pull-right"
action="index.php?action=find&name" method="post">
                <input class="form-control mr-sm-2" type="text" placeholder="Search"
name="search">
            </form>
        </li>
```

```
        <?php
        if($isLoggedIn && $albumCart != 0):
            ?>
            <li class="nav-brand">
                <a href='index.php?action=profile'><img class="nav-link" src="<?= $image ?>"
alt="" width="50" height="50"></a>
            </li>
            <li class="navbar-item">
                <a href="index.php?action=showCart" class="nav-link" data-toggle="tooltip" data-
placement="right" title="Check Out"><i class="fas fa-shopping-cart"></i><?php require_once
'_navCart.php' ?></a>
            </li>
            <li class="navbar-item">
                <a href="index.php?action=logout" class="nav-link"><i class="fas fa-sign-in-
alt"></i> Logout</a>
            </li>
            <?php
        else:
            ?>
            <li class="nav-item">
                <a class="nav-link" href="#" data-toggle="modal" data-target="#register-form"><i
class="fas fa-user"></i> Sign Up</a>
            </li>
            <li>
            <div class="modal fade" id="register-form" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
                <div class="modal-dialog" role="document">
                    <div class="modal-content">
                        <div class="modal-header">
                            <h5 class="modal-title" id="exampleModalLabel">Sign Up</h5>
                            <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                                <span aria-hidden="true">&times;</span>
                            </button>
                        </div>
                        <div class="modal-body">
                            <div class="container-fluid">
                                <form action="index.php?action=processRegister" method="post">
                                    <br>
                                    <div class="row">
                                        <div class="col-3"><label>First Name</label></div>
                                        <div class="col-9 col-md-offset-4"><input class="form-control"
type="text" name="firstName" placeholder="First Name" required="required"></div>
                                    </div>
```

```
                        <br>
                        <div class="row">
                            <div class="col-3"><label>Last Name</label></div>
                            <div class="col-9 col-md-offset-4"><input class="form-control"
type="text" name="lastName" placeholder="Last Name" required="required"></div>
                        </div>
                        <br>
                        <div class="row">
                            <div class="col-3"><label>Email</label></div>
                            <div class="col-9 col-md-offset-4"><input class="form-control"
type="email" name="email" placeholder="Email" required="required"></div>
                        </div>
                        <br>
                        <div class="row">
                            <div class="col-3"><label>Username</label></div>
                            <div class="col-9 col-md-offset-4"><input class="form-control"
type="text" name="username" placeholder="Username" required="required"
pattern=".{5,}"></div>
                        </div>
                        <br>
                        <div class="row">
                            <div class="col-3"><label>Password</label></div>
                            <div class="col-9 col-md-offset-4"><input class="form-control"
type="password" name="password" placeholder="Password" required="required"
pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}" title="Must contain at least one number and one
uppercase and lowercase letter, and at least 8 or more characters"></div>
                        </div>
                        <br>
                        <hr>
                        <br>
                        <input type="submit" name="register" class="btn btn-primary btn-lg
btn-block" value="Register">
                    </form>
                </div>
            </div>
        </div>
    </div>
    <li class="navbar-item">
        <a class="nav-link" href="#" data-toggle="modal" data-target="#login-modal"><i
class="fas fa-sign-in-alt"></i> Login</a>
    </li>
    <li>
    <div class="modal fade" id="login-modal" tabindex="-1" role="dialog" aria-
```

```
labelledby="myModalLabel" aria-hidden="true" style="display: none;">
                <div class="modal-dialog" role="document">
                    <div class="modal-content">
                        <div class="modal-header">
                            <h5 class="modal-title" id="myModalLabel">Login to Your Account</h5>
                            <button type="button" class="close" data-dismiss="modal" aria-
label="Close">
                                <span aria-hidden="true">&times;</span>
                            </button>
                        </div>
                        <div class="modal-body">
                            <div class="container-fluid">
                                <form action="index.php?action=processLogin" method="post">
                                    <br>
                                    <div class="row">
                                        <div class="col-3"><label>Username</label></div>
                                        <div class="col-9 col-md-offset-4"><input class="form-control"
type="text" name="username" placeholder="Username" required="required"></div>
                                    </div>
                                    <br>
                                    <div class="row">
                                        <div class="col-3"><label>Password</label></div>
                                        <div class="col-9 col-md-offset-4"><input class="form-control"
type="password" name="password" placeholder="Password" required="required"></div>
                                    </div>
                                    <br>
                                    <hr>
                                    <br>
                                    <input type="submit" name="login" class="btn btn-primary btn-lg btn-
block" value="Login">
                                </form>
                            </div>
                        </div>
                    </div>
                </div>
                </li>
                <?php
            endif;
            ?>
        </ul>
    </div>
</nav>
<script>
```

```
    $(document).ready(function(){
        $('[data-toggle="tooltip"]').tooltip();
    });
</script>
<script>
    $(document).ready(function () {
        var links = $('.navbar ul li a');
        $.each(links, function (key, va) {
            if (va.href == document.URL) {
                $(this).addClass('active');
            }
        });
    });
    $('#myModal').modal({
        keyboard: false
    });
    $('#myTab').on('shown.bs.tab', function(evt) {
        $(evt.target).parent().addClass('active');
        $(evt.relatedTarget).parent().removeClass('active');
    });
</script>
<article>
```

_navCart Template

```
<?php
$total = 0;
$songTotal = 0;
    foreach ($albumCart as $cartItem):
        $album = $cartItem->getAlbum();
        $subTotal = $album->getAlbumPrice() * $cartItem->getQuantity();
        $total += $subTotal;
    endforeach;
    foreach ($songCart as $cartIte):
        $song = $cartIte->getSong();
        $songSubTotal = $song->getSongPrice() * $cartIte->getQuantity();
        $songTotal += $songSubTotal;
    endforeach;
$total += $songTotal;
?>
&euro; <?= $total ?>
```

Album Details Page

```
<?PHP
require_once __DIR__ . '/../templates/_head.php';
```

```
require_once __DIR__ . '/../templates/_nav.php';
$album = \Itb\Album::getOneByAlbum($albumName);
$artist = \Itb\Artist::getArtistById($album->getArtistID());
$songs = \Itb\Single::getSinglesByAlbum($album->getAlbumID());
;?>
   <br>
   <div class="container">
      <nav class="breadcrumb">
         <a class="breadcrumb-item" href='index.php'>Home</a>
         <a class="breadcrumb-item" href='index.php?action=albums'>Albums</a>
         <a class="breadcrumb-item" href="index.php?action=show&artistName=<?= $artist-
>getArtistName() ?>"><?= $artist->getArtistName() ?></a>
         <span class="breadcrumb-item active"><?= $album->getAlbumName() ?></span>
      </nav>
      <div id="spaceOf10">
      <h4>RELEASE DETAILS</h4>
      <div class="row">
         <div class="col-sm-4">
             <img src="<?= $album->getAlbumImage() ?>" width="240" height="240">
               <p class="lead"><b>Artist: </b><a href="index.php?action=show&artistName=<?=
$artist->getArtistName() ?>"><?= $artist->getArtistName() ?></a>
               <br>
               <b>Album: </b><?= $album->getAlbumName() ?>
               <br>
               <b>Released: </b><?= $album->getAlbumRelease() ?>
               <br>
                  <?php if ($isLoggedIn) {?>
               <b>Price: €</b><?= $album->getAlbumPrice() ?> <a
href="index.php?action=addToCart&id=<?= $album->getAlbumID() ?>&type=Album" data-
toggle="tooltip" title="Add to Cart" data-placement="right"><i class="fas fa-cart-
plus"></i></a></p>
                  <?php } else {?>
            <p>Price: €<?= $album->getAlbumPrice() ?> <a href="#" data-toggle="modal" data-
target="#id01"><span data-toggle="tooltip" data-placement="right" title="Add to Cart"><i
class="fas fa-cart-plus"></i></span></a></p>

            <?php } ?></p>
         </div>
         <div class="col-sm-8">
            <div align="left" class="embed-responsive embed-responsive-16by9">
               <iframe width="560" height="315" align="right" src="<?= $album-
>getAlbumVideo()?>" frameborder="2" allowfullscreen></iframe>
            </div>
         </div>
```

```
        </div>
    </div>
<br>
<table class="table">
    <thead class="thead-light">
    <tr>
        <th scope="col">#</th>
        <th scope="col">Title</th>
        <th scope="col">Writer(s)</th>
        <th scope="col">Length</th>
        <th scope="col">Price</th>
        <th scope="col">Purchase</th>
    </tr>
    </thead>
    <tbody>
<?php if($songs != null) {
    foreach ($songs as $song) { ?>
        <tr>
            <th scope="row"><?= $song->getSongTrackNo() ?></th>
            <td><a href="index.php?action=showSong&songID=<?= $song->getSongID() ?>"><?=
$song->getSongName() ?></a>
            </td>
            <td><?= $song->getSongWriter() ?></td>
            <td><?= $song->getSongLength() ?></td>
            <td>€ <?= $song->getSongPrice() ?></td>
            <td><?php if ($isLoggedIn) {
                    ?>
                    <a href="index.php?action=addAlbumSongToCart&id=<?= $song->getSongID()
?>&type=Single&albumName=<?= $album->getAlbumName() ?>"
                       data-toggle="tooltip" data-placement="right" title="Add to Cart"><i class="fas fa-
cart-plus fa-1x"></i></a>
                    <?php
                } else { ?>
                    <a href="#" data-toggle="modal" data-target="#id01"><span data-toggle="tooltip"
data-placement="right" title="Add to Cart"><i class="fas fa-cart-plus fa-1x"></i></span></a>
                    <?php
                }
                ?></td>
        </tr>
    <?php } 
    }?>
    </tbody>
</table>
<div class="modal fade" id="id01" tabindex="-1" role="dialog" aria-labelledby="myModalLabel"
```

```
aria-hidden="true" style="display: none;">
   <div class="modal-dialog" role="document">
      <div class="modal-content">
         <div class="modal-header">
            <h5 class="modal-title">User Not Logged In</h5>
            <button type="button" class="close" data-dismiss="modal" aria-label="Close">
               <span aria-hidden="true">&times;</span>
            </button>
         </div>
         <div class="modal-body">
            <h2>You need to be logged in in order to purchase music.</h2>
            <br>
            <small><b>Not A Member</b></small>
            <small>You can simply sign up for free account by clicking on the sign-up button in
the top right corner of the page.</small>
         </div>
         <div class="modal-footer">
            <button type="button" class="btn btn-primary" data-dismiss="modal">Ok</button>
         </div>
      </div>
   </div>
   </div>
</div>
<br><br>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Albums Page

```
<?php require_once __DIR__ . '/../templates/_head.php'; ?>
<?php require_once __DIR__ . '/../templates/_nav.php'; ?>
<?php $albums = \Itb\Album::getAll() ?>
<?php $counter = 1; $flag = false;
if($select != 'Filter By Category'){
   $flag = true;
}
?>
<br>
<style>
   .well form {
      text-align: right;
   }
   .well {
      margin-top: 1em;
   }
</style>
```

```
<br>
<div class="container">
   <div class="well well-sm">
      <form action="index.php?action=setCat" method="post">
         <h2 class="display-4 float-left">Albums</h2>
         <?php if($select != 'Filter By Category'){?>
         <p>Results For <?= $select ?>
            <?php }?>
            <select id="category" name="select1"">
            <option value="Filter By Category"><?php if($select != 'Filter By Category'){?>
                  All
               <?php } 
               else{ ?>
                  <?=$select ?><?php
               }?></option>
            <option value="Pop">Pop</option>
            <option value="Jazz">Jazz</option>
            <option value="Rap">Rap</option>
            <option value="Heavy Metal">Heavy Metal</option>
            </select>
            <input type="submit" name="submit" value="Filter"/></p>
      </form>
   </div>
   <br>
   <div class="row">
      <?php if ($flag == true){
         foreach($albums as $album) {
            if ($album->getAlbumCat() == $select) {?>
               <?php $artist = \Itb\Artist::getArtistById($album->getArtistID()); ?>
               <div class="col-sm-3" id="id1">
                  <div class="col-item">
                     <div class="photo">
                        <a href="index.php?action=showAlbum&albumName=<?= $album-
>getAlbumName() ?>"> <img src="<?= $album->getAlbumImage() ?>" width="200"
height="200" class="img-responsive"></a>
                     </div>
                     <div class="info">
                        <div class="row">
                           <div class="price col-md-12">
                              <h6><?= $album->getAlbumName() ?></h6>
                              <h6 class="price-text-color">€<?= $album->getAlbumPrice() ?></h6>
                           </div>
                        </div>
                        <div class="separator clear-left">
```

```
                        <p class="btn-add">
                            <?php if($isLoggedIn){ ?>
                            <a href="index.php?action=addToCart&id=<?= $album->getAlbumID()
?>&type=Album" data-toggle="tooltip" title="Add to Cart" data-placement="right"><i class="fas
fa-cart-plus"></i> Add to Cart</p></a>
                        <?php } else { ?>
                            <a href="#" class="hidden-sm"  data-toggle="modal" data-
target="#id01"><span data-toggle="tooltip" data-placement="right" title="Add to Cart"><i
class="fas fa-cart-plus"></i> Add to cart</a></span></p>
                        <?php } ?>
                        <p class="btn-details">
                            <i class="fa fa-list"></i><a
href="index.php?action=showAlbum&albumName=<?= $album->getAlbumName() ?>">More
details</a></p>
                    </div>
                    <div class="clearfix">
                    </div>
                </div>
            </div>
            <?php $counter++ ?>
        <?php } ?>
    <?php } ?>
<?php } else {
    foreach($albums as $album) {
        $artist = \Itb\Artist::getArtistById($album->getArtistID()); ?>
        <div class="col-sm-3" id="id1">
            <div class="col-item">
                <div class="photo">
                    <a href="index.php?action=showAlbum&albumName=<?= $album-
>getAlbumName() ?>"> <img src="<?= $album->getAlbumImage() ?>" width="200"
height="200" class="img-responsive"></a>
                </div>
                <div class="info">
                    <div class="row">
                        <div class="price col-md-12">
                            <h6><?= $album->getAlbumName() ?></h6>
                            <h6 class="price-text-color">€<?= $album->getAlbumPrice() ?></h6>
                        </div>
                    </div>
                    <div class="separator clear-left">
                        <p class="btn-add">
                            <?php if($isLoggedIn){ ?>
                            <a href="index.php?action=addToCart&id=<?= $album->getAlbumID()
```

```
?>&type=Album" data-toggle="tooltip" title="Add to Cart" data-placement="right"><i class="fas
fa-cart-plus"></i> Add to Cart</p></a>
                        <?php } else { ?>
                           <a href="#" class="hidden-sm"  data-toggle="modal" data-
target="#id01"><span data-toggle="tooltip" data-placement="right" title="Add to Cart"><i
class="fas fa-cart-plus"></i> Add to cart</a></span></p>
                        <?php } ?>
                        <p class="btn-details">
                           <i class="fa fa-list"></i><a
href="index.php?action=showAlbum&albumName=<?= $album->getAlbumName() ?>">More
details</a></p>
                     </div>
                     <div class="clearfix">
                     </div>
                  </div>
               </div>
               <?php $counter++ ?>
            <?php } ?>
         <?php } ?>
      </div>
   </div>
   </div>
   <div class="modal fade" id="id01" tabindex="-1" role="dialog" aria-labelledby="myModalLabel"
   aria-hidden="true" style="display: none;">
      <div class="modal-dialog" role="document">
         <div class="modal-content">
            <div class="modal-header">
               <h5 class="modal-title">User Not Logged In</h5>
               <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                  <span aria-hidden="true">&times;</span>
               </button>
            </div>
            <div class="modal-body">
               <h2>You need to be logged in in order to purchase music.</h2>
               <br>
               <small><b>Not A Member</b></small>
               <small>You can simply sign up for free account by clicking on the sign-up button in
the top right corner of the page.</small>
            </div>
            <div class="modal-footer">
               <button type="button" class="btn btn-primary" data-dismiss="modal">Ok</button>
            </div>
         </div>
```

```
    </div>
</div>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Artist Details Page

```
<?PHP
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
$artist = \Itb\Artist::getOneByArtist($artistName);
$albums = \Itb\Album::getAlbumsByArtistID($artist->getArtistID());
;?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=artist'>Artists</a>
        <span class="breadcrumb-item active"><?= $artist->getArtistName() ?></span>
    </nav>
    <div id="spaceOf10">
        <div class="text-center">
            <img src="<?= $artist->getArtistImage() ?>" class="img-fluid" width="800" height="300"
alt="<?= $artist->getArtistName() ?>">
        </div>
        <br>
        <div class="row">
            <div class="col-xs-12 col-sm-12 col-lg-12"">
                <h3>BIOGRAPHY</h3>
                <p><?= $artist->getArtistBio() ?></p>
            </div>
        </div>
        <div class="row">
            <div class="col-xs-7 col-sm-6 col-lg-12"">
                <h3>DISCOGRAPHY</h3>
                <div class="artistDisc">
                    <?php if ($albums != null) { ?>
                        <?php foreach($albums as $album) {?>
                            <a href="index.php?action=showAlbum&albumName=<?= $album-
>getAlbumName() ?>"><img src="<?= $album->getAlbumImage() ?> "class="img-responsive"
width="200" height="200" border=2 style="float: left; margin-right: 1%; margin-bottom:
0.5em;"></a>
                        <?php } ?>
                    <?php } else {?>
                    <p>No Albums to display</p>
                    <?php } ?>
```

```
            </div>
          </div>
        </div>
    </div>
</div>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Artist Page

```
<?php
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
$artists = \Itb\Artist::getAll();
?>
<br>
<div class="container">
    <div class="table-responsive">
        <table class="table table-hover">
            <thead>
                <th>Image</th>
                <th>Artist</th>
                <th class="hideCol">Label</th>
            </thead>
            <tbody>
            <?php foreach($artists as $artist) {?>
            <tr>
                <td id="tdFont"><a href="index.php?action=show&artistName=<?= $artist->getArtistName() ?>"><img src="<?= $artist->getArtistImage() ?>" width="220" height="120" class="img-fluid"></a></td>
                <td id="tdFont"><a href="index.php?action=show&artistName=<?= $artist->getArtistName() ?>"><?= $artist->getArtistName() ?></a></td>
                <td id="tdFont" class="hideCol"><?= $artist->getArtistLabel()?></td>
            </tr>
            <?php } ?>
            </tbody>
        </table>
    </div>
</div>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Cart Page

```
<?PHP
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
```

```
;?>
<br>
<div class="well well-lg"><h1>Shopping Cart</h1></div>
<div class="container">
<table class="table table-striped">
   <tr>
      <th>Album</th>
      <th>Price</th>
      <th>Quantity</th>
      <th class="hideCol">Sub Total</th>
      <th width="10%">Remove Album</th>
   </tr>
<?php
$total = 0;
foreach($albumCart as $cartItem):
   $album = $cartItem->getAlbum();
   $subTotal = $album->getALbumPrice() * $cartItem->getQuantity();
   $total += $subTotal;
?>
   <tr>
      <td><?= $album->getAlbumName() ?></td>
      <td>&euro;<?= $album->getAlbumPrice() ?></td>
      <td><?= $cartItem->getQuantity() ?></td>
      <td class="hideCol">€ <?= $subTotal ?></td>
      <td width="10%"><a href="index.php?action=removeAlbumFromCart&id=<?= $album-
>getAlbumID() ?>"class="btn btn-warning">Remove</a></td>
   </tr>
<?php
endforeach;
?>
</table>
<table class="table table-striped">
   <tr>
      <th>Song</th>
      <th>Price</th>
      <th>Quantity</th>
      <th class="hideCol">Sub Total</th>
      <th width="10%">Remove Song</th>
   </tr>
   <?php
   foreach($songCart as $cartItem):
      $song = $cartItem->getSong();
      $subTotal = $song->getSongPrice() * $cartItem->getQuantity();
      $total += $subTotal;
```

```
        ?>
        <tr>
            <td><?= $song->getSongName() ?></td>
            <td>&euro;<?= $song->getSongPrice() ?></td>
            <td><?= $cartItem->getQuantity() ?></td>
            <td class="hideCol">€ <?= $subTotal ?></td>
            <td width="10%"><a href="index.php?action=removeSongFromCart&id=<?= $song->getSongID() ?>" class="btn btn-warning">Remove</a></td>
        </tr>
        <?php
    endforeach;
    ?>
    <tr class="success">
        <td colspan="3"><b>Total</td>
        <td><b>&euro; <?= $total ?></td>
    </tr>
</table>
<a href="index.php?action=emptyCart"  class="btn btn-danger">EMPTY CART</a>
<?php if ($total != 0) { ?>
    <a href="index.php?action=checkOut&total=<?= $total ?>"  class="btn btn-success">CHECK OUT</a>
<?php } ?>
</div>
<br><br>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Chart Page

```
<?php
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
$indexLinkStyle = isset($indexLinkStyle) ? $indexLinkStyle : 'nav-link';
$chartLinkStyle = isset($chartLinkStyle) ? $chartLinkStyle : 'nav-link active';
$contactLinkStyle = isset($contactLinkStyle) ? $contactLinkStyle : 'nav-link';
$artistLinkStyle = isset($artistLinkStyle) ? $artistLinkStyle : 'nav-link';
$albumLinkStyle = isset($albumLinkStyle) ? $albumLinkStyle : 'nav-link';
$profileLinkStyle = isset($profileLinkStyle) ? $profileLinkStyle : '';
$adminAlbumLinkStyle = isset($adminAlbumLinkStyle) ? $adminAlbumLinkStyle : 'nav-link';
$adminArtistLinkStyle = isset($adminArtistLinkStyle) ? $adminArtistLinkStyle : 'nav-link';
$adminSingleLinkStyle = isset($adminSingleLinkStyle) ? $adminSingleLinkStyle : 'nav-link';
$songLinkStyle = isset($songLinkStyle) ? $songLinkStyle : 'nav-link';
$image = \Itb\User::getUserImageByUserName($username);
$charts = \Itb\Chart::getAll();
$albums = \Itb\Album::getAll();
$artists = \Itb\Artist::getAll();
```

```
$counter = 1;
?>
<br>
<div class="container">
<table class="table table-striped">
   <tr>
      <th>Rank</th>
      <th>Album</th>
      <th>Artist</th>
      <th>Weeks on Chart</th>
   </tr>
   <?php foreach($charts as $chart) {
      $album = Itb\Album::getAlbumByAlbumID($chart->getAlbumID());
      $artist = Itb\Artist::getArtistByArtistID($album->getArtistID());
      if ($counter <= 15){
      ?>
      <tr>
         <td><b><?= $counter ?></b></td>
         <td><a href="index.php?action=showAlbum&albumName=<?= $album-
>getAlbumName() ?>"><?= $album->getAlbumName() ?></a></td>
         <td><a href="index.php?action=show&artistName=<?= $artist->getArtistName() ?>">
               <?= $artist->getArtistName() ?></a></td>
         <td><?= $chart->getChartWeeks() ?></td>
      </tr>
   <?php
         $counter++;
      }
   } ?>
</table>
<?
//php require_once __DIR__ . '/_footer.php'; ?>
</div>
<script>
   $(document).ready(function () {
      var links = $('.navbar ul li a');
      $.each(links, function (key, va) {
         if (va.href == document.URL) {
            $(this).addClass('active');
         }
      });
   });
   $('#myModal').modal({
      keyboard: false
```

```
   })
</script>
```

Contact Page

```
<?php
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/_nav.php';?>
<br>
<div class="jumbotron jumbotron-sm">
   <div class="container">
      <div class="row">
         <div class="col-sm-12 col-lg-12">
            <h1>Contact Us</h1>
         </div>
      </div>
   </div>
</div>
<div class="container">
   <div class="row">
      <div class="col-md-8">
         <div class="well well-sm">
            <form action="index.php?action=contactMsg"
                  method="POST" enctype="multipart/form-data">
               <div class="row">
                  <div class="col-md-6">
                     <div class="form-group">
                        <label for="name">Name</label>
                        <input type="text" class="form-control" id="name" placeholder="Enter
name" required="required" />
                     </div>
                     <div class="form-group">
                        <label for="email">Email Address</label>
                        <div class="input-group">
                        <span class="input-group-addon"><span class="fas fa-envelope"></span>
                        </span>
                           <input type="email" class="form-control" id="email" placeholder="Enter
email" required="required" /></div>
                     </div>
                     <div class="form-group">
                        <label for="subject">Subject</label>
                        <select id="subject" name="subject" class="form-control"
required="required">
                           <option value="na" selected="">Choose One:</option>
```

```
                        <option value="service">General Customer Service</option>
                        <option value="suggestions">Suggestions</option>
                        <option value="product">Music Support</option>
                     </select>
                  </div>
               </div>
               <div class="col-md-6">
                  <div class="form-group">
                     <label for="name">Message</label>
                     <textarea name="message" id="message" class="form-control" rows="9"
cols="25" required="required"
                              placeholder="Message"></textarea>
                  </div>
               </div>
               <div class="col-md-12">
                  <button type="submit" class="btn btn-primary pull-right"
id="btnContactUs">Send Message</button>
               </div>
            </div>
         </form>
      </div>
   </div>
   <div class="col-md-4">
      <form>
         <legend><span class="glyphicon glyphicon-globe"></span> Our office</legend>
         <address>
            <strong>GerrysJukebox,</strong><br>
            Blanchardstown Rd N,<br>
            Blanchardstown,<br>
            Dublin 15<br>
            <abbr title="Phone" class="glyphicon glyphicon-earphone"></abbr>
            (123) 456-7890
         </address>
         <address>
            <strong>Email</strong><br>
            <a href="mailto:#">admin@gerrysjukebox.com</a>
         </address>
      </form>
   </div>
</div>
</div>
   <br><br>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Download Page

```
<?php
$file_url = 'music/P!nk.mp3';
header('Content-Type: application/octet-stream');
header("Content-Transfer-Encoding: Binary");
header("Content-disposition: attachment; filename=\"" . basename($file_url) . "\"");
readfile($file_url);
?>
```

Index Page

```
<?php require_once __DIR__ . '/../templates/_head.php'; ?>
<?php require_once __DIR__ . '/../templates/_nav.php'; ?>
<br>
<div id="carouselExampleIndicators" class="carousel slide" data-ride="carousel">
   <ol class="carousel-indicators">
      <li data-target="#carouselExampleIndicators" data-slide-to="0" class="active"></li>
      <li data-target="#carouselExampleIndicators" data-slide-to="1"></li>
      <li data-target="#carouselExampleIndicators" data-slide-to="2"></li>
      <li data-target="#carouselExampleIndicators" data-slide-to="3"></li>
      <li data-target="#carouselExampleIndicators" data-slide-to="4"></li>
      <li data-target="#carouselExampleIndicators" data-slide-to="5"></li>
   </ol>
   <div class="carousel-inner">
      <div class="carousel-item active">
         <a href="index.php"><img class="d-block w-100" src="images/homeslide.png"
alt="Home slide"></a>
      </div>
      <div class="carousel-item">
         <a href="index.php?action=song"><img class="d-block w-100"
src="images/storeslide.png" alt="Store slide"></a>
      </div>
      <div class="carousel-item">
         <a href="index.php?action=albums"><img class="d-block w-100"
src="images/albumslide.png" alt="Album slide"></a>
      </div>
      <div class="carousel-item">
         <a href="index.php?action=chart"><img class="d-block w-100"
src="images/chartslide.png" alt="Chart slide"></a>
      </div>
      <div class="carousel-item">
         <a href="index.php?action=contact"><img class="d-block w-100"
src="images/contactslide.png" alt="Contact slide"></a>
      </div>
      <div class="carousel-item">
```

```
        <a href="index.php"><img class="d-block w-100" src="images/signupslide.png"
alt="Sign up slide"></a>
      </div>
    </div>
    <a class="carousel-control-prev" href="#carouselExampleIndicators" role="button" data-
slide="prev">
      <span class="carousel-control-prev-icon" aria-hidden="true"></span>
      <span class="sr-only">Previous</span>
    </a>
    <a class="carousel-control-next" href="#carouselExampleIndicators" role="button" data-
slide="next">
      <span class="carousel-control-next-icon" aria-hidden="true"></span>
      <span class="sr-only">Next</span>
    </a>
</div>
<br>
<br>
<!-- Page Content -->
<div class="container">
    <h2>Featured Artist</h2>
    <div class="row">
      <div class="col-lg-6">
        <img class="img-fluid rounded mb-4" src="/images/artists/fozzyhomepage.png" alt="">
      </div>
      <div class="col-lg-6">
        <h2><a href="index.php?action=show&artistName=Fozzy">Fozzy</a></h2>
        <p>FOZZY has always been about a heavy groove and a good time. And when you
have two high-energy performers like Rich Ward and Chris Jericho (it’s debatable on who jumps
higher onstage) in the band, grooves and good times come easy; but these guys aren’t just
entertainers. Ward is one of the most versatile and underrated riffers in rock and metal today,
who has created his own style of heavy riffs, melodic choruses and the Duke groove...oh that
crushing groove! And Jericho's singing ability and overall passion for music makes one wonder
just how he is able to find the time to excel in pretty much everything he does. It was these
qualities that pushed the band to become one of the hottest up and coming rock acts of the past
five years.</p>
      </div>
    </div>
    <h2>Featured Albums</h2>
    <div class="row">
      <div class="col-lg-4 mb-4">
        <div class="card h-100 text-center">
          <img class="card-img-top" src="/images/albums/fozzyjudas.png" alt="">
          <div class="card-body">
            <h4 class="card-title"><a
```

```
href="index.php?action=showAlbum&albumName=Judas">JUDAS</a></h4>
                <h6 class="card-subtitle mb-2 text-muted"><a
href="index.php?action=show&artistName=Fozzy">Fozzy</a></h6>
                <p class="card-text">Judas is the seventh studio album by the American heavy
metal band Fozzy. It was released on October 13, 2017 through Century Media Records.</p>
            </div>
          </div>
        </div>
        <div class="col-lg-4 mb-4">
          <div class="card h-100 text-center">
            <img class="card-img-top" src="/images/albums/ballad.png" alt="">
            <div class="card-body">
              <h4 class="card-title"><a href="index.php?action=showAlbum&albumName=The
Ballad of Tom and Cindy">THE BALLAD OF TOM AND CINDY</a></h4>
                <h6 class="card-subtitle mb-2 text-muted"><a
href="index.php?action=show&artistName=Drake Bell">Drake Bell</a></h6>
                <p class="card-text">The Ballad of Tom and Cindy is the third studio album by
Drake Bell. It was released on July 27, 2012 through Universal Motown.</p>
            </div>
          </div>
        </div>
        <div class="col-lg-4 mb-4">
          <div class="card h-100 text-center">
            <img class="card-img-top" src="/images/albums/momentum.png" alt="">
            <div class="card-body">
              <h4 class="card-title"><a
href="index.php?action=showAlbum&albumName=Momentum">MOMENTUM</a></h4>
                <h6 class="card-subtitle mb-2 text-muted"><a
href="index.php?action=show&artistName=Jamie Cullum">Jamie Cullum</a></h6>
                <p class="card-text">Momentum is the sixth studio album by Jamie Cullum. It was
released on 20 May 2013 by Island Records and is produced by Dan the Automator and Jim
Abbiss.</p>
            </div>
          </div>
        </div>
    </div>
    <h2>Featured Singles</h2>
    <div class="row">
        <div class="col-lg-2 col-sm-4 mb-4">
          <a href="index.php?action=showSong&songID=1"> <img class="img-fluid"
src="/images/singles/JudasSingle.jpg" alt=""></a>
        </div>
        <div class="col-lg-2 col-sm-4 mb-4">
          <a href="index.php?action=showSong&songID=90"><img class="img-fluid"
```

```
src="/images/singles/EdgeOfSomething.jpg" alt=""></a>
        </div>
        <div class="col-lg-2 col-sm-4 mb-4">
            <a href="index.php?action=showSong&songID=212"><img class="img-fluid"
src="/images/singles/Iron_Maiden_Empire_of_the_Clouds.jpg" alt=""></a>
        </div>
        <div class="col-lg-2 col-sm-4 mb-4">
            <a href="index.php?action=showSong&songID=158"><img class="img-fluid"
src="/images/singles/Bitchcraft.jpg" alt=""></a>
        </div>
        <div class="col-lg-2 col-sm-4 mb-4">
            <a href="index.php?action=showSong&songID=332"><img class="img-fluid"
src="/images/singles/Notb.jpg" alt=""></a>
        </div>
        <div class="col-lg-2 col-sm-4 mb-4">
            <a href="index.php?action=showSong&songID=25"> <img class="img-fluid"
src="/images/singles/Sandpaper.jpg" alt=""></a>
        </div>
    </div>
</div>
<!-- Bootstrap core JavaScript -->
<script src="/vendor/bootstrap/jquery.min.js"></script>
<script src="/vendor/bootstrap/bootstrap.bundle.min.js"></script>
<?php require_once __DIR__ . '/../templates/_footer.php'; ?>
```

Message Page

```
<?php
require_once __DIR__ . '/_head.php';
require_once __DIR__ . '/_nav.php';
?>
<br>
<h1>
    <?= $message ?>
</h1>
    <br><br>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Payment Page

```
<?PHP
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
?>
<br>
    <style>
```

```
.paymentWrap {
    padding: 50px;
}
.paymentWrap .paymentBtnGroup {
    max-width: 800px;
    margin: auto;
}
.paymentWrap .paymentBtnGroup .paymentMethod {
    padding: 40px;
    box-shadow: none;
    position: relative;
}
.paymentWrap .paymentBtnGroup .paymentMethod.active {
    outline: none !important;
}
.paymentWrap .paymentBtnGroup .paymentMethod.active .method {
    border-color: #4cd264;
    outline: none !important;
    box-shadow: 0px 3px 22px 0px #7b7b7b;
}
.paymentWrap .paymentBtnGroup .paymentMethod .method {
    position: absolute;
    right: 3px;
    top: 3px;
    bottom: 3px;
    left: 3px;
    background-size: contain;
    background-position: center;
    background-repeat: no-repeat;
    border: 2px solid transparent;
    transition: all 0.5s;
}
.paymentWrap .paymentBtnGroup .paymentMethod .method.visa {
    background-image: url("data:image.png");
}

.paymentWrap .paymentBtnGroup .paymentMethod .method.master-card {
    background-image: url("data:image.png");
}
.paymentWrap .paymentBtnGroup .paymentMethod .method.amex {
    background-image: url("http://www.paymentscardsandmobile.com/wp-
content/uploads/2015/08/Amex-icon.jpg");
}
.paymentWrap .paymentBtnGroup .paymentMethod .method.vishwa {
```

```
            background-image: url("http://i.imgur.com/VkiM7PL.jpg");
        }
        .paymentWrap .paymentBtnGroup .paymentMethod .method.ez-cash {
            background-image:
url("http://www.busbooking.lk/img/carousel/BusBooking.lk_ezCash_offer.png");
        }
        .paymentWrap .paymentBtnGroup .paymentMethod .method:hover {
            border-color: #4cd264;
            outline: none !important;
        }
    </style>
<div class="container">
    <div class="row">
        <div class="paymentCont">
            <div class="headingWrap">
                <h3 class="headingTop text-center">Select Your Payment Method</h3>
            </div>
            <div class="paymentWrap">
                <div class="btn-group paymentBtnGroup btn-group-justified" data-toggle="buttons">
                    <label class="btn paymentMethod active">
                        <div class="method visa"></div>
                        <input type="radio" name="options" checked>
                    </label>
                    <label class="btn paymentMethod">
                        <div class="method master-card"></div>
                        <input type="radio" name="options">
                    </label>
                    <label class="btn paymentMethod">
                        <div class="method amex"></div>
                        <input type="radio" name="options">
                    </label>
                    <label class="btn paymentMethod">
                        <div class="method vishwa"></div>
                        <input type="radio" name="options">
                    </label>
                    <label class="btn paymentMethod">
                        <div class="method ez-cash"></div>
                        <input type="radio" name="options">
                    </label>
                </div>
            </div>
            <div class="footerNavWrap clearfix">
                <div class="btn btn-success pull-left btn-fyi"><span class="glyphicon glyphicon-
chevron-left"></span> CONTINUE SHOPPING</div>
```

```
            <div class="btn btn-success pull-right btn-fyi">CHECKOUT<span class="glyphicon
glyphicon-chevron-right"></span></div>
        </div>
    </div>

  </div>
</div>
<br>
<?php require_once __DIR__ . '/../templates/_footer.php'; ?>
```

User Profile Page

```
<?php
require_once __DIR__ . '/_head.php';
require_once __DIR__ . '/_nav.php';
$user = \Itb\User::getOneByUsername($username);
?>
<br>
<div class="container">
<h2 class="display-4"><?= $user->getFirstName() ?> <?= $user->getLastName() ?>'s
Details</h2>
<br>
<br>
<dl>
  <dt><img src="<?= $user->getUserImage() ?>" alt=""  class="img-responsive" width="280"
height="280"></dt>
  <dt>ID:</dt>
  <dd><?= $user->getUserId() ?></dd>
  <dt>First Name:</dt>
  <dd><?= $user->getFirstName() ?></dd>
  <dt>Last Name:</dt>
  <dd><?= $user->getLastName() ?></dd>
  <dt>Username:</dt>
  <dd><?= $user->getUserName() ?></dd>
</dl>
  <form enctype="multipart/form-data" action="index.php?action=imageUpload"
method="POST">
    <input type="hidden" name="id" value="<?= $user->getUserId() ?>" required="required">
    <input type="hidden" name="MAX_FILE_SIZE" value="1000000" required="required">
    <h4>Change Profile image</h4>
    <input name="uploadedfile" type="file" accept=".png, .jpg, .jpeg" aria-
describedby="fileHelp" required="required">
    <br>
    <br>
    <input type="submit" value="Upload File"/>
```

```
        <br>
        <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload profile
image of type .png, .jpg, .jpeg.</small>
    </form>
    <br>
    <h4>Change Background Colour</h4>
    <a href='index.php?action=setBackgroundColorBlue'>Blue</a>
    <br>
    <a href='index.php?action=setBackgroundColorGreen'>Green</a>
    <br>
    <a href='index.php?action=setBackgroundColorDefault'>Default</a>
    <br>
    <br>
    <h4>Change Your Password</h4>
    <a href="#" data-toggle="modal" data-target="#change-password" class="btn btn-info btn-
lg"><span class="glyphicon glyphicon-lock"></span> Change Password</a>
    <div class="modal fade" id="chane-password" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel" aria-hidden="true" style="display: none;">
        <div class="modal-dialog">
            <div class="loginmodal-container">
                <h1>Change Password</h1><br>
                <form action="index.php?action=updateProfile" method="post">
                    <h1>Change Password</h1>
                    <label>Enter New Password</label>
                    <input type="password" name="userPassword" placeholder="Password">
                    <input type="hidden" name="userId" value="<?= $user->getUserId() ?>">
                    <input type="submit" name="register" class="login loginmodal-submit"
value="Change Password" onclick="myFunction()">
                </form>
            </div>
        </div>
    </div>
    <!-- Modal -->
    <div class="modal fade" id="change-password" tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel" aria-hidden="true">
        <div class="modal-dialog" role="document">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title" id="exampleModalLabel">Change Password</h5>
                    <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                        <span aria-hidden="true">&times;</span>
                    </button>
                </div>
                <div class="modal-body">
```

```
                <form action="index.php?action=updateProfile" method="post" onsubmit="return
myFunction()">
                    <label>Enter New Password</label>
                    <input class="form-control" type="password" name="userPassword"
placeholder="Password" required="required" pattern="(?=.*\d)(?=.*[a-z])(?=.*[A-Z]).{8,}"
title="Must contain at least one number and one uppercase and lowercase letter, and at least 8
or more characters">
                    <input type="hidden" name="userId" value="<?= $user->getUserId() ?>">
                    <br>
                    <hr>
                <div class="col-sm-12 text-center">
                    <button id="submit" type="submit" name="register" class="btn btn-outline-
success btn-lg btn-block">Change Password</button>
                    <button type="button" class="btn btn-outline-danger btn-lg btn-block" data-
dismiss="modal">Close</button>
                </div>
                </form>
            </div>
        </div>
    </div>
</div>
    <script>
        function myFunction() {
            alert("Password Successfully Changed");
        }
    </script>
<br>
<br>
<?php require_once __DIR__ . '/_footer.php';
```

Purchase Page

```
<?php
require_once __DIR__ . '/_head.php';
require_once __DIR__ . '/_nav.php';
?>
<br>
<div class="container">
    <h1 id="h1Left">Thank you for your purchase</h1>
    <br>
    <br>
    <br>
    <p>Your download will start in a moment. If it doesn't, use this <a
href="index.php?action=download">direct link.</a></p>
```

```
</div>
<?php require_once __DIR__ . '/_footer.php'; ?>
<script type="text/javascript">
   window.onload = startDownload();
   function startDownload() {
      window.location = "index.php?action=download";
   }
</script>
```

Single Details Page

```
<?PHP
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/../templates/_nav.php';
$song = \Itb\Single::getSingleById($songID);
$artist = \Itb\Artist::getArtistByArtistID($song->getArtistID());
$album = Itb\Album::getAlbumByAlbumID($song->getAlbumID());
$albums = \Itb\Album::getAlbumsByArtistID($artist->getArtistID());
?>
<br>
<div class="container">
   <nav class="breadcrumb">
      <a class="breadcrumb-item" href='index.php'>Home</a>
      <a class="breadcrumb-item" href='index.php?action=song'>Singles</a>
      <span class="breadcrumb-item active"><?= $song->getSongName() ?></span>
   </nav>
   <div id="spaceOf10">
<h4>SONG DETAILS</h4>
   <div class="row">
      <div class="col-sm-6">
         <img src="<?= $song->getSongImage() ?>" width="240" height="240" class="img-
responsive">
         <p class="lead">
         <br><b>Album: </b><a href="index.php?action=showAlbum&albumName=<?= $album-
>getAlbumName() ?>"><?= $album->getAlbumName() ?></a>
         <br><b>Track Length: </b><?= $song->getSongLength() ?>'s
         <br><b>Category: </b><?= $song->getSongCat() ?>
         <br><b>Writer(s): </b><?= $song->getSongWriter() ?></p>
         <?php if($isLoggedIn){
            ?>
            <p class="lead">Price: €<?= $song->getSongPrice() ?> <a
href="index.php?action=addSongToCart&id=<?= $song->getSongID() ?>&type=Single" data-
toggle="tooltip" data-placement="right" title="Add to Cart"><i class="fas fa-cart-
plus"></i></a></p>
            <?php
```

```
        }else{?>
            <p class="lead">Price: €<?= $song->getSongPrice() ?> <a href="#" data-
toggle="modal" data-target="#id01"><span data-toggle="tooltip" data-placement="right"
title="Add to Cart"><i class="fas fa-cart-plus"></i></span></a></p>
            <?php
        }
        ?>
      </div>
      <div class="col-sm-6">
        <h4><?= $artist->getArtistName() ?>'s Details</h4>
        <img src="<?= $artist->getArtistImage() ?>" width="400" height="150" alt="$artistName"
class="img-responsive">
        <br><br>
        <p class="lead"><b>Artist: </b><a href="index.php?action=show&artistName=<?=
$artist->getArtistName() ?>"><?= $artist->getArtistName() ?></a>
        <br><b>Label: <?= $artist->getArtistLabel() ?></b>
          <br><b>Genre: <?= $artist->getArtistCat() ?></b></p>
        <h5>Other Albums From <?= $artist->getArtistName() ?></h5>
        <?php foreach($albums as $album) {?>
          <a href="index.php?action=showAlbum&albumName=<?= $album->getAlbumName()
?>"><img src="<?= $album->getAlbumImage() ?>" class="img-responsive" width="100"
height="100" border=2 style="float: left; margin-right: 1%; margin-bottom: 0.5em;"></a>
        <?php } ?>
      </div>
    </div>
</div>
<div class="modal fade" id="id01" tabindex="-1" role="dialog" aria-labelledby="myModalLabel"
aria-hidden="true" style="display: none;">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">User Not Logged In</h5>
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
          <span aria-hidden="true">&times;</span>
        </button>
      </div>
      <div class="modal-body">
        <h2>You need to be logged in in order to purchase music.</h2>
        <br>
        <small><b>Not A Member</b></small>
        <small>You can simply sign up for free account by clicking on the sign-up button in
the top right corner of the page.</small>
      </div>
      <div class="modal-footer">
```

```
            <button type="button" class="btn btn-primary" data-dismiss="modal">Ok</button>
          </div>
        </div>
    </div>
    </div>
</div>
<br>
<br><br>
<script>
    $(document).ready(function(){
        $('[data-toggle="tooltip"]').tooltip();
    });
</script>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Singles Page

```
<?php
require_once __DIR__ . '/../templates/_head.php';
require_once __DIR__ . '/_nav.php';
$singles = \Itb\Single::getAllSinglesInOrder();
$flag = false;
$id1=1;
$id2=1;
if($select1 != 'Filter By Category'){
    $flag = true;
}
?>
    <br>
    <style>
        .well form {
            text-align: right;
        }
        .well {
            margin-top: 1em;
        }
    </style>
    <br>
    <div class="container">
        <div class="well well-sm">
            <form action="index.php?action=setSongCat" method="post">
                <h2 class="display-4 float-left">Singles</h2>
                <?php if($select1 != 'Filter By Category'){?>
                <p>Results For <?= $select1 ?>
                    <?php }?>
```

```
                <select id="category" name="select" title="Filter">
                <option value="Filter By Category"><?php if($select1 != 'Filter By Category'){?>
                    All
                  <?php } 
                  else{ ?>

                    <?=$select1 ?><?php
                  }?></option>
                <option value="Pop">Pop</option>
                <option value="Jazz">Jazz</option>
                <option value="Rap">Rap</option>
                <option value="Heavy Metal">Heavy Metal</option>
                </select>
              <input type="submit" name="submit" value="Filter"/>
            </form>
        </div>
        <br>
        <br>
        <div class="row">
          <?php if ($flag == true){
            foreach($singles as $single) {
              if ($single->getSongCat() == $select1) {?>
                <?php $artist = \Itb\Artist::getArtistById($single->getArtistID()) ?>
                <?php $album = \Itb\Album::getAlbumNameByArtistID($single->getArtistID()) ?>
                <div class="col-sm-3" id="id<?=$id1++?>" style="padding-bottom: 2%">
                  <div class="col-item">
                    <div class="photo">
                      <a href="index.php?action=showSong&songID=<?= $single-
>getSongID()?>"><img src="<?= $single->getSongImage() ?>" width="200" height="200"
class="img-responsive" alt="<?=$single->getSongName()?>"></a>
                    </div>
                    <div class="info">
                      <div class="row">
                        <div class="price col-md-12">
                          <h6><?= $single->getSongName() ?></h6>
                          <h6 class="price-text-color">€<?= $single->getSongPrice() ?></h6>
                        </div>
                      </div>
                      <div class="separator clear-left">
                        <p class="btn-add">
                          <?php if($isLoggedIn){ ?>
                          <a href="index.php?action=addSongToCart&id=<?= $single-
>getSongID() ?>&type=Single" data-toggle="tooltip" title="Add to Cart" data-
```

```
placement="right"><i class="fas fa-cart-plus"></i> Add to Cart</a>
                                <?php } else { ?>
                                    <a href="#" class="hidden-sm"  data-toggle="modal" data-
target="#id01"><i class="fas fa-cart-plus"></i><span data-toggle="tooltip" data-
placement="right" title="Add to Cart"> Add to cart</a></span></p>
                                <?php } ?>
                                <p class="btn-details">
                                    <i class="fa fa-list"></i><a
href="index.php?action=showSong&songID=<?= $single->getSongID()?>">More
details</a></p>
                            </div>
                            <div class="clearfix">
                            </div>
                        </div>
                    </div>
                </div>
                <?php } ?>
            <?php } ?>
        <?php } else {
            foreach($singles as $single) {?>
                <?php $artist = \Itb\Artist::getArtistById($single->getArtistID()) ?>
                <?php $album = \Itb\Album::getAlbumNameByArtistID($single->getArtistID()) ?>
                <div class="col-sm-3" id="id<?=$id1++?>" style="padding-bottom: 2%">
                    <div class="col-item">
                        <div class="photo">
                            <a href="index.php?action=showSong&songID=<?= $single->getSongID()
?>"> <img src="<?= $single->getSongImage() ?>" width="200" height="200" class="img-
responsive" alt="<?= $single->getSongName() ?>"></a>
                        </div>
                        <div class="info">
                            <div class="row">
                                <div class="price col-md-12">
                                    <h6><?= $single->getSongName() ?></h6>
                                    <h6 class="price-text-color">€<?= $single->getSongPrice() ?></h6>
                                </div>
                            </div>
                            <div class="separator clear-left">
                                <p class="btn-add">
                                    <?php if($isLoggedIn){ ?>
                                    <a href="index.php?action=addSongToCart&id=<?= $single-
>getSongID() ?>&type=Single" data-toggle="tooltip" title="Add to Cart" data-
placement="right"><i class="fas fa-cart-plus"></i> Add to Cart</a>
                                <?php } else { ?>
                                    <a href="#" class="hidden-sm"  data-toggle="modal" data-
```

```
target="#id01"><i class="fas fa-cart-plus"></i><span data-toggle="tooltip" data-
placement="right" title="Add to Cart"> Add to cart</span></a></p>
                        <?php } ?>
                        <p class="btn-details">
                            <i class="fa fa-list"></i><a
href="index.php?action=showSong&songID=<?= $single->getSongID()?>">More
details</a></p>
                    </div>
                    <div class="clearfix">
                    </div>
                </div>
            </div>
        </div>
        <?php } ?>
    <?php } ?>
  </div>
  <div class="modal fade" id="id01" tabindex="-1" role="dialog" aria-
labelledby="myModalLabel1" aria-hidden="true" style="display: none;">
    <div class="modal-dialog" role="document">
      <div class="modal-content">
        <div class="modal-header">
          <h5 class="modal-title" id="myModalLabel1">User Not Logged In</h5>
          <button type="button" class="close" data-dismiss="modal" aria-label="Close">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>
        <div class="modal-body">
          <h2>You need to be logged in in order to purchase music.</h2>
          <br>
          <small><b>Not A Member</b></small>
          <small>You can simply sign up for free account by clicking on the sign-up button in
the top right corner of the page.</small>
        </div>
        <div class="modal-footer">
          <button type="button" class="btn btn-primary" data-dismiss="modal">Ok</button>
        </div>
      </div>
    </div>
  </div>

<?php require_once __DIR__ . '/_footer.php'; ?>
```

Album CRUD Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$albums = \Itb\Album::getAll()
?>
<br>
<div class="container">
   <div class="row">
      <h3>Edit Album Details</h3>
   </div>
   <div class="row">
      <label>Add a New Album</label>
   </div>
   <div class="row"><a href="index.php?action=showNewAlbum" class="btn btn-outline-primary
btn-lg">Create</a></div>
      <br>
   <div class="row">
      <div class="table-responsive">
         <table class="table table-bordered table-responsive">
            <thead>
               <tr>
                  <th>Album Name</th>
                  <th class="hideCol">Album Release</th>
                  <th class="hideCol">Album Price</th>
                  <th class="hideCol">Album Image</th>
                  <th class="hideCol">Artist ID</th>
                  <th>Action</th>
               </tr>
            </thead>
            <tbody>
            <?php
            foreach ($albums as $album) {
               ?>
               <tr>
                  <td width="25%"><?=$album->getAlbumName()?></td>
                  <td class="hideCol"><?=$album->getAlbumRelease()?></td>
                  <td class="hideCol">€<?= $album->getAlbumPrice()?></td>
                  <td class="hideCol"><img src="<?= $album->getAlbumImage() ?>" alt=""
class="img-responsive" width="60" height="60"></td>
                  <td class="hideCol"><?= $album->getArtistID()?></td>
                  <td>
                  <a class="btn btn-success"
href="index.php?action=albumUpdate&albumID=<?=$album->getAlbumID()?>">Update</a>
                  <a class="btn btn-danger" href="index.php?action=deleteAlbum&id=<?=$album-
```

```
>getAlbumID() ?>" onclick="myFunction()">Delete</a>
                    </td>
                </tr>
            <?php
            }
            ?>
            </tbody>
        </table>
    </div>
</div>
</div> <!-- /container -->
    <script>
        function myFunction() {
            alert("Album Deleted Successfully");
        }
    </script>
    <style>
        .table-responsive {
            display: table;
        }
    </style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Artist CRUD Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$artists = \Itb\Artist::getAll()
?>
<br>
<div class="container">
    <div class="row">
        <h3>Edit Artists Details</h3>
    </div>
    <div class="row">
        <label>Add a New Artist</label>
    </div>
    <div class="row">
        <a href="index.php?action=showNewArtist" class="btn btn-outline-primary btn-
lg">Create</a>
    </div>
    <br>
        <div class="row">
            <div class="table-responsive">
```

```
            <table class="table table-bordered table-responsive">
                <thead>
                <tr>
                    <th>Artist Name</th>
                    <th class="hideCol">Artist Image</th>
                    <th class="hideCol">Artist Label</th>
                    <th>Action</th>
                </thead>
                <tbody>
                <?php
                foreach ($artists as $artist) {
                    ?>
                    <tr>
                        <td><?=$artist->getArtistName()?></td>
                        <td class="hideCol"><img src="<?= $artist->getArtistImage() ?>" alt=""
class="img-responsive" width="120" height="60"></td>
                        <td class="hideCol"><?= $artist->getArtistLabel()?></td>
                        <td>
                            <a class="btn btn-success"
href="index.php?action=artistUpdate&artistID=<?=$artist->getArtistID()?>">Update</a>
                            <a class="btn btn-danger"
href="index.php?action=deleteArtist&id=<?=$artist->getArtistID() ?>"
onclick="myFunction()">Delete</a>
                        </td>
                    </tr>
                    <?php
                }
                ?>
                </tbody>
            </table>
        </div>
    </div>
</div> <!-- /container -->
</div>
    <script>
        function myFunction() {
            alert("Artist Deleted Successfully");
        }
    </script>
    <style>
        .table-responsive {
            display: table;
        }
```

```
    </style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Singles CRUD Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$singles = \Itb\Single::getAll()
?>
<br>
    <div class="container">
        <div class="row">
            <h3>Edit Single Details</h3>
        </div>
    <div class="row">
        <label>Add a New Single</label>
    </div>
    <div class="row">
        <a href="index.php?action=showNewSingle" class="btn btn-outline-primary btn-
lg">Create</a>
    </div>
        <br>
        <div class="row">

            <div class="table-responsive">
                <table class="table table-bordered table-responsive">
                    <thead>
                    <tr>
                        <th>Name</th>
                        <th class="hideCol">Track No</th>
                        <th class="hideCol">Writers</th>
                        <th class="hideCol">Length</th>
                        <th class="hideCol">Price</th>
                        <th class="hideCol">Category</th>
                        <th>Update</th>
                        <th>Delete</th>
                    </thead>
                    <tbody>
                    <?php
                    foreach ($singles as $single) {
                        ?>
                        <tr>
                            <td><?=$single->getSongName()?></td>
                            <td class="hideCol"><?=$single->getSongTrackNo()?></td>
```

```
                    <td class="hideCol" width="25%"><?=$single->getSongWriter()?></td>
                    <td class="hideCol"><?=$single->getSongLength()?></td>
                    <td class="hideCol">€<?=$single->getSongPrice()?></td>
                    <td class="hideCol"><?=$single->getSongCat()?></td>
                    <td><a class="btn btn-success"
href="index.php?action=singleUpdate&songID=<?=$single->getSongID()?>">Update</a></td>
                    <td><a class="btn btn-danger"
href="index.php?action=deleteSingle&id=<?=$single->getSongID() ?>"
onclick="myFunction()">Delete</a></td>
                </tr>
                <?php
            }
            ?>
            </tbody>
        </table>
    </div>
</div>
</div> <!-- /container -->
<script>
    function myFunction() {
        alert("Single Deleted Successfully");
    }
</script>
<style>
    .table-responsive {
        display: table;
    }
</style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

New Album Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$allArtists = \Itb\Artist::getAll();
?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=adminAlbum'>Edit Albums</a>
        <span class="breadcrumb-item active">Add New Album</span>
    </nav>
    <div id="spaceOf10">
    <h2>Add New Album</h2>
```

```
   <form action="index.php?action=newAlbum"
         method="POST" enctype="multipart/form-data">
       <input type="hidden" name="albumID" value="">
       <div class="form-group">
          <label><b>Album Name</b></label>
          <input type="text" class="form-control" id="albumName" placeholder="Enter Album
Name" name="albumName" required="required">
       </div>
       <div class="form-group">
          <label><b>Album Release</b></label>
          <input type="date" class="form-control" id="albumRelease" placeholder="Enter Album
Release Date" name="albumRelease" required="required">
       </div>
       <div class="form-group">
          <label><b>Album Category</b></label>
          <input type="text" class="form-control" id="albumCat" placeholder="Enter Album
Category" name="albumCat" required="required">
       </div>
       <div class="form-group">
          <label><b>Album Video</b></label>
          <input type="text" class="form-control" id="albumVideo" placeholder="Enter Album
Video Link" name="albumVideo" required="required">
          <small id="fileHelp" class="form-text text-muted">Add the video source section from the
iframe e.g. src="abc", just the abc part</small>
       </div>
       <div class="form-group">
          <label><b>Upload Album Image</b></label>
          <br>
          <input type="hidden" name="id">
          <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
          <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg,
.jpeg" required="required">
          <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload album
image of type .png, .jpg, .jpeg and size of either 300x300 or 200x200</small>
       </div>
       <div class="form-group">
          <label><b>Album Price</b></label>
          <input type="number" min="0.00" max="10000.00" step="any" class="form-control"
id="albumPrice" placeholder="Enter Album Price" name="albumPrice" aria-
describedby="fileHelp" required="required">
          <small id="fileHelp" class="form-text text-muted">Price needs to be in float format e.g
9.99, 10.00, 1.50 etc.</small>
       </div>
       <br>
```

```
        <label><b>List of Artist ID's</b></label>
        <div class="dropup">
            <button class="btn btn-info dropdown-toggle" type="button" data-
toggle="dropdown">Artist ID's
                <span class="caret"></span></button>
            <ul class="dropdown-menu scrollable-menu">
                <li><b>ID   Artist Name</b></li>
                <?php foreach($allArtists as $allArtist) {?>
                    <li><?= $allArtist->getArtistID() ?>  |  <?= $allArtist->getArtistName() ?></li>
                <?php } ?>
            </ul>
        </div>
        <br>
        <div class="form-group">
            <label><b>Artist ID</b></label>
            <input type="number" min="1" max="<?= sizeof($allArtists)?>" class="form-control"
id="artistID" placeholder="Enter Artist ID" name="artistID" aria-describedby="fileHelp">
            <small id="fileHelp" class="form-text text-muted">If the Album artist is not displayed in
the drop down above, you need to add the Artist before adding the Album</small>

        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
        <a href="index.php?action=adminAlbum" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>

    </form>
        </div>
</div>
    <br>
    <br>
<style>
    .scrollable-menu {
        height: auto;
        max-height: 200px;
        overflow-x: hidden;
    }
</style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

New Artist Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
?>
```

```
<br>
<div class="container">
   <nav class="breadcrumb">
      <a class="breadcrumb-item" href='index.php'>Home</a>
      <a class="breadcrumb-item" href='index.php?action=adminArtist'>Edit Artist's</a>
      <span class="breadcrumb-item active">Add New Artist</span>
   </nav>
   <div id="spaceOf10">
   <h2>Add New Artist</h2>
   <form action="index.php?action=newArtist"
        method="POST" enctype="multipart/form-data">
      <input type="hidden" name="artistID">
      <div class="form-group">
         <label><b>Artist(s) Name</b></label>
         <input type="text" class="form-control" id="artistName" placeholder="Enter Artist Name"
name="artistName" required="required">
      </div>
      <div class="form-group">
         <label><b>Artist Record Label</b></label>
         <input type="text" class="form-control" id="artistLabel" placeholder="Enter Artist Record
Label" name="artistLabel" required="required">
      </div>
      <div class="form-group">
         <label><b>Artist Music Category</b></label>
         <input type="text" class="form-control" id="artistCat" placeholder="Enter Artist Music
Category" name="artistCat" aria-describedby="fileHelp" required="required">
         <small id="fileHelp" class="form-text text-muted">Artist music category e.g. Rock, Rap,
Heavy Metal etc.</small>
      </div>
      <div class="form-group">
         <label><b>Upload Artist Image</b></label>
         <br>
         <input type="hidden" name="id">
         <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
         <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg,
.jpeg" required="required">
         <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload album
image of type .png, .jpg, .jpeg and of size 800x300px.</small>
      </div>
      <div class="form-group">
         <label><b>Artist Biography</b></label>
         <textarea class="form-control" rows="8" name="artistBio" id="artistBio"
placeholder="Enter Artist Bio" required="required"></textarea>
      </div>
```

```
        <button type="submit" class="btn btn-primary">Submit</button>
        <a href="index.php?action=adminArtist" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>
    </form>
        </div>
    </div>
    <br>
    <br>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

New Single Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$artists = \Itb\Artist::getAll();
$albums = \Itb\Album::getAll();
?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=adminSingle'>Edit Singles</a>
        <span class="breadcrumb-item active">Add New Single</span>
    </nav>
    <div id="spaceOf10">
    <h2>Add New Single</h2>
    <form action="index.php?action=newSingle"
          method="POST" enctype="multipart/form-data">
        <div class="form-group">
            <label><b>Single Name</b></label>
            <input type="text" class="form-control" id="songName" placeholder="Enter single name"
name="songName" required="required">
        </div>
        <div class="form-group">
            <label><b>Writer(s)</b></label>
            <input type="text" class="form-control" id="songWriter" placeholder="Enter single
writer(s)" name="songWriter" aria-describedby="fileHelp" required="required">
            <small id="fileHelp" class="form-text text-muted">If theres more than one writer
seperate name by a comma e.g. Paul Deitel, Harvey Deitel etc.</small>
        </div>
        <div>
        </div>
        <div class="form-group">
            <label><b>Upload Single Image</b></label>
```

```
            <br>
            <input type="hidden" name="id">
            <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
            <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg,
.jpeg" required="required">
            <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload song
image of type .png, .jpg, .jpeg.</small>
        </div>
        <div class="form-group">
            <label><b>Price</b></label>
            <input type="number" min="0.00" max="10000.00" step="any" class="form-control"
id="songPrice" placeholder="Enter the singles price" name="songPrice" aria-
describedby="fileHelp" required="required">
            <small id="fileHelp" class="form-text text-muted">Price needs to be in float format e.g
1.99, 10.00, 1.50 etc.</small>
        </div>
        <div class="form-group">
            <label><b>Length</b></label>
            <input type="time" class="form-control" id="songLength" placeholder="Enter the singles
play length" name="songLength" aria-describedby="fileHelp" required="required">
            <small id="fileHelp" class="form-text text-muted">Length needs to be in the type double
format e.g 3.47, 5.10, 1.36 etc.</small>
        </div>
        <div class="form-group">
            <label><b>Category</b></label>
            <input type="text" class="form-control" id="songCat" placeholder="Enter the music
category of the single" name="songCat" aria-describedby="fileHelp" required="required">
            <small id="fileHelp" class="form-text text-muted">Music category e.g. Rock, Rap, Heavy
Metal etc.</small>
        </div>
        <div class="form-group">
            <label><b>Track Number</b></label>
            <input type="number" class="form-control" id="songTrackNo" placeholder="Enter track
number" name="songTrackNo" aria-describedby="fileHelp" required="required">
            <small id="fileHelp" class="form-text text-muted">Track number needs to be of type
integer.</small>
        </div>

        <label><b>List of Artist ID's</b></label>
        <div class="dropup">
            <button class="btn btn-info dropdown-toggle" type="button" data-
toggle="dropdown">Artist ID's
                <span class="caret"></span></button>
            <ul class="dropdown-menu scrollable-menu">
```

```
            <li><b>ID   Artist Name</b></li>
            <?php foreach($artists as $artist) {?>
               <li><?=$artist->getArtistID() ?> |  <?=$artist->getArtistName() ?></li>
            <?php } ?>
         </ul>
      </div>
      <br>
      <div class="form-group">
         <label><b>Artist ID</b></label>
         <input type="number" min="1" max="<?= sizeof($artists)?>" class="form-control"
id="artistID" placeholder="Enter the Artist ID" name="artistID" aria-describedby="fileHelp"
required="required">
         <small id="fileHelp" class="form-text text-muted">If the Artist is not displayed in the drop
down above, you need to add the Artist before adding the Single</small>
      </div>
      <label><b>List of Album ID's</b></label>
      <div class="dropup">
         <button class="btn btn-info dropdown-toggle" type="button" data-
toggle="dropdown">Album ID's
            <span class="caret"></span></button>
         <ul class="dropdown-menu scrollable-menu">
            <li><b>ID   Album Name</b></li>
            <?php foreach($albums as $album) {?>
               <li><?=$album->getAlbumID() ?> |  <?=$album->getAlbumName() ?></li>
            <?php } ?>
         </ul>
      </div>
      <br>
      <div class="form-group">
         <label><b>Album ID</b></label>
         <input type="number" min="1" max="<?= sizeof($albums)?>" class="form-control"
id="albumID" placeholder="Enter Album ID" name="albumID" aria-describedby="fileHelp"
required="required">
         <small id="fileHelp" class="form-text text-muted">If the Album artist is not displayed in
the drop down above, you need to add the Artist before adding the Album</small>

      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
      <a href="index.php?action=adminSingle" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>
   </form>
      </div>
   </div>
   <br>
```

```
    <br>
    <style>
        .scrollable-menu {
            height: auto;
            max-height: 200px;
            overflow-x: hidden;
        }
    </style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Update Album Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$artist = \Itb\Artist::getArtistById($album->getArtistID());
$allArtists = \Itb\Artist::getAll();
?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=adminAlbum'>Edit Albums</a>
        <span class="breadcrumb-item active">Update Album Details</span>
    </nav>
    <div id="spaceOf10">
<h2>Edit <?=$album->getAlbumName() ?> Album</h2>
<form action="index.php?action=updateAlbum"
      method="POST" enctype="multipart/form-data">
    <input type="hidden" name="albumID" value="<?= $album->getAlbumID() ?>" >
        <div class="form-group">
            <label><b>Album Name</b></label>
            <input type="text" class="form-control" id="albumName" value="<?= $album-
>getAlbumName() ?>" name="albumName" required="required">
        </div>
        <div class="form-group">
            <label><b>Album Release</b></label>
            <input type="date" class="form-control" id="albumRelease" value="<?= $album-
>getAlbumRelease() ?>" name="albumRelease" required="required">
        </div>
        <div class="form-group">
            <label><b>Album Category</b></label>
            <input type="text" class="form-control" id="albumCat" value="<?= $album-
>getAlbumCat() ?>" name="albumCat">
        </div>
```

```
        <div class="form-group">
            <label><b>Video</b> </label>
            <input type="text" class="form-control" id="albumVideo" value="<?= $album->getAlbumVideo() ?>" name="albumVideo" required="required">
            <small id="fileHelp" class="form-text text-muted">Add the video source section from the iframe e.g. src="abc", just the abc part</small>
        </div>
        <div>
            <label><b>Album Image</b></label>
        </div>
        <br>
        <img src="<?= $album->getAlbumImage() ?>" width="240" height="240" class="img-responsive">
        <br>
        <br>
        <div class="form-group">
            <label><b>Upload New Image</b></label>
            <input type="hidden" name="id">
            <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
            <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg, .jpeg">
            <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload album image of type .png, .jpg, .jpeg.</small>
        </div>
        <div class="form-group">
            <label><b>Album Price</b></label>
            <input type="number" min="0.00" max="10000.00" step="any" class="form-control" id="albumPrice" value="<?= $album->getAlbumPrice() ?>" name="albumPrice" required="required">
        </div>
        <br>
        <label><b>List of Artist ID's</b></label>
        <div class="dropup">
            <button class="btn btn-info dropdown-toggle" type="button" data-toggle="dropdown">Artist ID's
                <span class="caret"></span></button>
            <ul class="dropdown-menu scrollable-menu">
                <li><b>ID   Artist Name</b></li>
                <?php foreach($allArtists as $allArtist) {?>
                <li><?= $allArtist->getArtistID() ?>  |  <?= $allArtist->getArtistName() ?></li>
                <?php } ?>
            </ul>
        </div>
        <br>
```

```
        <div class="form-group">
            <label><b>Artist ID</b></Artist></label>
            <input type="number" min="1" max="<?= sizeof($allArtists)?>" class="form-control"
id="artistID" value="<?= $album->getArtistID() ?>" name="artistID" aria-describedby="fileHelp"
required="required">
            <small id="fileHelp" class="form-text text-muted">If the Album artist is not displayed in
the drop down above, you need to add the Artist before adding the Album</small>
        </div>
        <button type="submit" class="btn btn-primary">Submit</button>
        <a href="index.php?action=adminAlbum" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>
</form>
    </div>
</div>
<br>
<br>
    <style>
        .scrollable-menu {
            height: auto;
            max-height: 200px;
            overflow-x: hidden;
        }
    </style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Update Artist Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$artist = \Itb\Artist::getArtistByArtistID($artist->getArtistID());
?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=adminArtist'>Edit Artist's</a>
        <span class="breadcrumb-item active">Update Artist Details</span>
    </nav>
    <div id="spaceOf10">
    <h2>Edit <?=$artist->getArtistName() ?>'s Details</h2>
    <form action="index.php?action=updateArtist"
          method="POST" enctype="multipart/form-data">
        <input type="hidden" name="artistID" value="<?= $artist->getArtistID() ?>">
        <div class="form-group">
```

```
        <label><b>Artist(s) Name</b></label>
        <input type="text" class="form-control" id="artistName" value="<?= $artist-
>getArtistName() ?>" name="artistName" required="required">
    </div>
    <div class="form-group">
        <label><b>Artist Record Label</b></label>
        <input type="text" class="form-control" id="artistLabel" value="<?= $artist-
>getArtistLabel() ?>" name="artistLabel" required="required">
    </div>
    <div class="form-group">
        <label><b>Artist Music Category</b></label>
        <input type="text" class="form-control" id="artistCat" value="<?= $artist->getArtistCat()
?>" name="artistCat" aria-describedby="fileHelp" required="required">
        <small id="fileHelp" class="form-text text-muted">Artist music category e.g. Rock, Rap,
Heavy Metal etc.</small>
    </div>
    <div>
        <label><b>Artist Image</b></label>
    </div>
    <img src="<?= $artist->getArtistImage() ?>" width="360" height="160" class="img-
responsive">
    <br>
    <br>
    <div class="form-group">
        <label><b>Upload New Image</b></label>
        <br>
        <input type="hidden" name="id">
        <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
        <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg,
.jpeg">
        <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload album
image of type .png, .jpg, .jpeg.</small>
    </div>
    <div class="form-group">
        <label><b>Artist Biography</b></label>
        <textarea class="form-control" rows="8" name="artistBio" id="artistBio"
required="required"><?= $artist->getArtistBio() ?></textarea>
    </div>
    <button type="submit" class="btn btn-primary">Submit</button>
    <a href="index.php?action=adminArtist" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>
  </form>
  </div>
</div>
```

```
    <br>
    <br>
    <script src="http://js.nicedit.com/nicEdit-latest.js" type="text/javascript"></script>
    <script type="text/javascript">bkLib.onDomLoaded(nicEditors.allTextAreas);</script>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Update Singles Page for Admins

```
<?php
require_once __DIR__ . '/../_head.php';
require_once __DIR__ . '/../_nav.php';
$artists = \Itb\Artist::getAll();
$albums = \Itb\Album::getAll();
?>
<br>
<div class="container">
    <nav class="breadcrumb">
        <a class="breadcrumb-item" href='index.php'>Home</a>
        <a class="breadcrumb-item" href='index.php?action=adminSingle'>Edit Singles</a>
        <span class="breadcrumb-item active">Update Single Details</span>
    </nav>
    <div id="spaceOf10">
    <h2>Edit Single</h2>
    <form action="index.php?action=updateSingle"
          method="POST" enctype="multipart/form-data">
        <input type="hidden" name="songID" value="<?=$song->getSongID() ?>">
        <div class="form-group">
            <label><b>Single Name</b></label>
            <input type="text" class="form-control" id="songName" value="<?=$song-
>getSongName() ?>" name="songName" required="required">
        </div>
        <div class="form-group">
            <label><b>Writer(s)</b></label>
            <input type="text" class="form-control" id="songWriter" value="<?=$song-
>getSongWriter() ?>" name="songWriter" required="required">
        </div>
        <div>
            <label><b>Single Image</b></label>
        </div>
        <img src="<?=$song->getSongImage() ?>" width="240" height="240" class="img-
responsive">
        <br>
        <br>
        <div class="form-group">
            <label><b>Upload New Image</b></label>
```

```
            <br>
            <input type="hidden" name="id">
            <input type="hidden" name="MAX_FILE_SIZE" value="1000000">
            <input name="uploadedfile" type="file" aria-describedby="fileHelp" accept=".png, .jpg,
.jpeg">
            <small id="fileHelp" class="form-text text-muted">Click "Choose File" to upload song
image of type .png, .jpg, .jpeg.</small>
        </div>
        <div class="form-group">
            <label><b>Price</b></label>
            <input type="number" min="0.00" max="10000.00" step="any" class="form-control"
id="songPrice" value="<?=$song->getSongPrice() ?>" name="songPrice" required="required">
        </div>
        <div class="form-group">
            <label><b>Length</b></label>
            <input type="time" class="form-control" id="songLength" value="<?=$song-
>getSongLength() ?>" name="songLength" required="required">
        </div>
        <div class="form-group">
            <label><b>Category</b></label>
            <input type="text" class="form-control" id="songCat" value="<?=$song->getSongCat()
?>" name="songCat" required="required">
        </div>
        <div class="form-group">
            <label><b>Track Number</b></label>
            <input type="number" class="form-control" id="songTrackNo" value="<?=$song-
>getSongTrackNo() ?>" name="songTrackNo" required="required">
        </div>
        <label><b>List of Artist ID's</b></label>
        <div class="dropup">
            <button class="btn btn-info dropdown-toggle" type="button" data-
toggle="dropdown">Artist ID's
                <span class="caret"></span></button>
            <ul class="dropdown-menu scrollable-menu">
                <li><b>ID   Artist Name</b></li>
                <?php foreach($artists as $artist) {?>
                    <li><?=$artist->getArtistID() ?> |  <?=$artist->getArtistName() ?></li>
                <?php } ?>
            </ul>
        </div>

        <br>
        <div class="form-group">
            <label><b>Artist ID</b></label>
```

```
        <input type="number" min="1" max="<?= sizeof($artists)?>" class="form-control"
id="artistID" value="<?=$song->getArtistID() ?>" name="artistID" required="required">
      </div>
      <br>
      <label><b>List of Album ID's</b></label>
      <div class="dropup">
        <button class="btn btn-info dropdown-toggle" type="button" data-
toggle="dropdown">Album ID's
          <span class="caret"></span></button>
        <ul class="dropdown-menu scrollable-menu">
          <li><b>ID   Album Name</b></li>
          <?php foreach($albums as $album) {?>
            <li><?=$album->getAlbumID() ?> |  <?=$album->getAlbumName() ?></li>
          <?php } ?>
        </ul>
      </div>
      <br>
      <div class="form-group">
        <label><b>Album ID</b></label>
        <input type="number" min="1" max="<?= sizeof($albums)?>" class="form-control"
id="albumID" value="<?=$song->getAlbumID() ?>" name="albumID" required="required">
      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
      <a href="index.php?action=adminSingle" id="cancel" name="cancel" class="btn btn-
danger">Cancel</a>
    </form>
    </div>
</div>
    <br>
    <br>
<style>
    .scrollable-menu {
        height: auto;
        max-height: 200px;
        overflow-x: hidden;
    }
</style>
<?php require_once __DIR__ . '/../_footer.php'; ?>
```

Payment Page

```
<?php
require_once __DIR__ . '/../templates/_head.php';
?>
    <link href="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/css/bootstrap.min.css"
```

```
rel="stylesheet" id="bootstrap-css">
   <script src="//maxcdn.bootstrapcdn.com/bootstrap/3.3.0/js/bootstrap.min.js"></script>
   <script src="//code.jquery.com/jquery-1.11.1.min.js"></script>
   <link rel="stylesheet" href="http://maxcdn.bootstrapcdn.com/font-awesome/4.3.0/css/font-
awesome.min.css">
   <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery-
validate/1.13.1/jquery.validate.min.js"></script>
   <script type="text/javascript"
src="https://cdnjs.cloudflare.com/ajax/libs/jquery.payment/1.2.3/jquery.payment.min.js"></script
>
   <script type="text/javascript" src="https://js.stripe.com/v2/"></script>
   <br><br>
   <div class="container">
      <div class="row">
         <div id="cc" class="col-xs-12 col-md-4">
            <div class="panel panel-default credit-card-box">
               <div class="panel-heading display-table" >
                  <div class="row display-tr" >
                     <h3 class="panel-title display-td" >Payment Details</h3>
                     <div class="display-td" >
                        <img class="img-responsive pull-right"
src="http://i76.imgup.net/accepted_c22e0.png">
                     </div>
                  </div>
               </div>
               <div class="panel-body">
                  <form role="form" id="payment-form" method="POST"
action="javascript:void(0);">
                     <div class="row">
                        <div class="col-xs-12">
                           <div class="form-group">
                              <label for="cardNumber">CARD NUMBER</label>
                              <div class="input-group">
                                 <input
                                       type="tel"
                                       class="form-control"
                                       name="cardNumber"
                                       placeholder="Valid Card Number"
                                       autocomplete="cc-number"
                                       required autofocus
                                 />
                                 <span class="input-group-addon"><i class="fa fa-credit-
card"></i></span>
                              </div>
```

```
                    </div>
                </div>
            </div>
            <div class="row">
                <div class="col-xs-7 col-md-7">
                    <div class="form-group">
                        <label for="cardExpiry"><span class="hidden-
xs">EXPIRATION</span><span class="visible-xs-inline">EXP</span> DATE</label>
                        <input
                            type="tel"
                            class="form-control"
                            name="cardExpiry"
                            placeholder="MM / YY"
                            autocomplete="cc-exp"
                            required
                        />
                    </div>
                </div>
                <div class="col-xs-5 col-md-5 pull-right">
                    <div class="form-group">
                        <label for="cardCVC">CV CODE</label>
                        <input
                            type="tel"
                            class="form-control"
                            name="cardCVC"
                            placeholder="CVC"
                            autocomplete="cc-csc"
                            required
                        />
                    </div>
                </div>
            </div>
            <div class="row">
                <div class="col-xs-12">
                    <button class="subscribe btn btn-success btn-lg btn-block" type="button"
onclick="openPurchase()">Confirm Purchase</button>
                    <button class="cancel btn btn-danger btn-lg btn-block" type="button"
onclick="openCancel()">Cancel</button>
                </div>
            </div>
            <div class="row" style="display:none;">
                <div class="col-xs-12">
                    <p class="payment-errors"></p>
                </div>
```

```
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>
<style>
    #cc {
        float: none;
        margin: 0 auto;
    }
    /* Padding - just for asthetics on Bootsnipp.com */
    body { margin-top:20px; }
    .credit-card-box .panel-title {
        display: inline;
        font-weight: bold;
    }
    .credit-card-box .form-control.error {
        border-color: red;
        outline: 0;
        box-shadow: inset 0 1px 1px rgba(0,0,0,0.075),0 0 8px rgba(255,0,0,0.6);
    }
    .credit-card-box label.error {
        font-weight: bold;
        color: red;
        padding: 2px 8px;
        margin-top: 2px;
    }
    .credit-card-box .payment-errors {
        font-weight: bold;
        color: red;
        padding: 2px 8px;
        margin-top: 2px;
    }
    .credit-card-box label {
        display: block;
    }
    .credit-card-box .display-table {
        display: table;
    }
    .credit-card-box .display-tr {
        display: table-row;
    }
```

```
    .credit-card-box .display-td {
        display: table-cell;
        vertical-align: middle;
        width: 50%;
    }
    .credit-card-box .panel-heading img {
        min-width: 180px;
    }
</style>
<script>
    var $form = $('#payment-form');
    $form.find('.subscribe').on('click', payWithStripe);
    function payWithStripe(e) {
        e.preventDefault();
        if (!validator.form()) {
            return;
        }
        $form.find('.subscribe').html('Validating <i class="fa fa-spinner fa-
pulse"></i>').prop('disabled', true);
        var PublishableKey = 'pk_test_6pRNASCoBOKtIshFeQd4XMUh'; // Replace with your API
publishable key
        Stripe.setPublishableKey(PublishableKey);
        var expiry = $form.find('[name=cardExpiry]').payment('cardExpiryVal');
        var ccData = {
            number: $form.find('[name=cardNumber]').val().replace(/\s/g,''),
            cvc: $form.find('[name=cardCVC]').val(),
            exp_month: expiry.month,
            exp_year: expiry.year
        };
        Stripe.card.createToken(ccData, function stripeResponseHandler(status, response) {
            if (response.error) {
                $form.find('.subscribe').html('Try again').prop('disabled', false);
                $form.find('.payment-errors').text(response.error.message);
                $form.find('.payment-errors').closest('.row').show();
            } else {
                $form.find('.subscribe').html('Processing <i class="fa fa-spinner fa-pulse"></i>');
                $form.find('.payment-errors').closest('.row').hide();
                $form.find('.payment-errors').text("");
                // response contains id and card, which contains additional card details
                console.log(response.id);
                console.log(response.card);
                var token = response.id;
                // AJAX - you would send 'token' to your server here.
                $.post('/account/stripe_card_token', {
```

```
                token: token
            })
            // Assign handlers immediately after making the request,
                .done(function(data, textStatus, jqXHR) {
                    $form.find('.subscribe').html('Payment successful <i class="fa fa-check"></i>');
                })
                .fail(function(jqXHR, textStatus, errorThrown) {
                    $form.find('.subscribe').html('There was a
problem').removeClass('success').addClass('error');
                    $form.find('.payment-errors').text('Try refreshing the page and trying again.');
                    $form.find('.payment-errors').closest('.row').show();
                });
        }
    });
}
$('input[name=cardNumber]').payment('formatCardNumber');
$('input[name=cardCVC]').payment('formatCardCVC');
$('input[name=cardExpiry').payment('formatCardExpiry');
jQuery.validator.addMethod("cardNumber", function(value, element) {
    return this.optional(element) || Stripe.card.validateCardNumber(value);
}, "Please specify a valid credit card number.");
jQuery.validator.addMethod("cardExpiry", function(value, element) {
    value = $.payment.cardExpiryVal(value);
    return this.optional(element) || Stripe.card.validateExpiry(value.month, value.year);
}, "Invalid expiration date.");
jQuery.validator.addMethod("cardCVC", function(value, element) {
    return this.optional(element) || Stripe.card.validateCVC(value);
}, "Invalid CVC.");
validator = $form.validate({
    rules: {
        cardNumber: {
            required: true,
            cardNumber: true
        },
        cardExpiry: {
            required: true,
            cardExpiry: true
        },
        cardCVC: {
            required: true,
            cardCVC: true
        }
    },
    highlight: function(element) {
```

```
            $(element).closest('.form-control').removeClass('success').addClass('error');
        },
        unhighlight: function(element) {
            $(element).closest('.form-control').removeClass('error').addClass('success');
        },
        errorPlacement: function(error, element) {
            $(element).closest('.form-group').append(error);
        }
    });
    paymentFormReady = function() {
        if ($form.find('[name=cardNumber]').hasClass("success") &&
            $form.find('[name=cardExpiry]').hasClass("success") &&
            $form.find('[name=cardCVC]').val().length > 1) {
            return true;
        } else {
            return false;
        }
    };
    $form.find('.subscribe').prop('disabled', true);
    var readyInterval = setInterval(function() {
        if (paymentFormReady()) {
            $form.find('.subscribe').prop('disabled', false);
            clearInterval(readyInterval);
        }
    }, 250);
    function openPurchase(){
        if (paymentFormReady()) {
            window.location.href = "index.php?action=purchase";
        }
    }
    function openCancel() {
        window.location.href = "index.php?action=showCart";
    }
</script>
<?php require_once __DIR__ . '/_footer.php'; ?>
```

Database Table Setup

```
CREATE TABLE users (
    userId int NOT NULL AUTO_INCREMENT,
    firstName varchar(50),
    lastName varchar(50),
    userName varchar(50),
    userRole int NOT NULL,
    userImage text(255),
```

```
    userEmail varchar(90),
    userPassword text(255),
    PRIMARY KEY(userId)
);
CREATE TABLE artists (
    artistID int NOT NULL AUTO_INCREMENT,
    artistName varchar(70) NOT NULL,
    artistImage text(255) NOT NULL,
    artistBio text(255) NOT NULL,
    artistLabel varchar(70) NOT NULL,
    artistCat varchar(70) NOT NULL,
    PRIMARY KEY(artistID)
);
CREATE TABLE albums (
    albumID int NOT NULL AUTO_INCREMENT,
    albumName varchar(70) NOT NULL,
    albumImage text(255) NOT NULL,
    albumRelease varchar(70) NOT NULL,
    albumVideo text(255) NOT NULL,
    albumPrice float NOT NULL,
    artistID int NOT NULL,
    albumCat varchar(70) NOT NULL,
    PRIMARY KEY(albumID),
    CONSTRAINT albums_artistID_FK FOREIGN KEY (artistID) REFERENCES artists(artistID)
);
CREATE TABLE singles (
        songID int NOT NULL AUTO_INCREMENT,
    songTrackNo int,
    songName varchar(70) NOT NULL,
    songWriter text(255),
    songLength varchar (70),
    songImage text(255),
    songPrice float NOT NULL,
    songCat varchar (70) NOT NULL,
    artistID int,
    albumID int,
    PRIMARY KEY(songID),
    CONSTRAINT songs_artistID_FK FOREIGN KEY (artistID) REFERENCES artists(artistID),
    CONSTRAINT songs_albumID_FK FOREIGN KEY (albumID) REFERENCES
albums(albumID)
);
create table charts (
        chartID int NOT NULL AUTO_INCREMENT,
    chartAlbum varchar(70) NOT NULL,
```

```
    chartArtist varchar(70) NOT NULL,
    chartWeeks int NOT NULL,
    PRIMARY KEY(chartID)
);
```

List of References

https://noisey.vice.com/en_us/article/6vapxr/go-aerosmith-how-head-first-became-the-first-song-available-for-digital-download-20-years-ago-today - [1] source for first ever download of music.
https://validator.w3.org/ [2]
https://airbrake.io/blog/sdlc/waterfall-model [3]