**VISVESVARAYA TECHNOLOGICAL**

**UNIVERSITY, BELGAVI**

****

A

Project Report On

**“MUSIC LIBRARY”**

Submitted in partial fulfilment for the DBMS Practical with Mini Project of Bachelor’s Degree in Information Science.

Submitted By

**Sharath S (1JS16IS070)**

**Shyam Sundar Bahety (1JS16IS078)**

Under The Guidance of

**Mrs M B Apsara**  **Mrs Mamatha G**

Assistant Professor, Assistant Professor,

Department of ISE Department of ISE



**Department Of Information Science and Engineering**

**JSS Academy of Technical Education, Bengaluru**



**Department Of Information Science and Engineering**

**JSS Academy of Technical Education, Bengaluru**

**Certificate**

This is to certify that the project work entitled **“Music Library”** is a bonafied work carried out by **Sharath S (1JS16IS070)** and **Shyam Sundar Bahety (1JS16IS078)** in partial fulfilment for the DBMS Laboratory with Mini Project (15CSL58) of 5th semester **Bachelor of Engineering** in **Information Science and Engineering** of the **Visvesvaraya Technological University, Belgavi** during the academic year 2018-2019. It is certified that all corrections and suggestions indicated for internal assessment have been incorporated in the report.

**Mrs M B Apsara Mrs Mamatha G Dr Dayananda P**

(Assistant Professor, (Assistant Professor, (HOD, Department of

Department of ISE) Department of ISE) ISE)

**Name of the examiner**

**ACKNOWLEDGEMENT**

We express my humble pranamas to his holiness **Jagadguru Sri Sri Sri Shivarathri Deshikendra Mahaswamiji** who has showered their blessings on us for framing our career successfully.

The completion of any project involves the efforts of many people. We have been lucky enough to have received a lot of help and support from all quarters during the making of this project, so with gratitude, we take this opportunity to acknowledge all those whose guidance and encouragement helped us emerge successful.

We are thankful to the resourceful guidance, timely assistance and graceful gesture of our guide **Mrs. M B Apsara,** Assistant Professor, Department of Information Science and Engineering and **Mrs. Mamatha G**, Assistant Professor, Department of Information Science and Engineering, who has helped us in every aspect of our project work.

We are also indebted to **Dr Dayananda P,** Head of Department of Information Science and Engineering for the facilities and support extended towards us.

We express our sincere thanks to our beloved principal**, Dr. Mrityunjaya V Latte** for having supported us in our academic endeavors.

And last but not the least, we would be very pleased to express our heart full thanks to all the teaching and non-teaching staff of ISE department and our friends who have rendered their help, motivation and support.

**Sharath S**

**Shyam Sundar Bahety**

**LIST OF CONTENTS**

**Acknowledgement**

**List of Figures**

**1. Introduction**

**LIST OF FIGURES**

1. ER Diagram …
2. Schema Diagram
3. Home Page
4. Login Page/Sign Up Page
5. Albums Page
6. Artist’s Page
7. Search

**ABSTRACT**

This project is to demonstrate the implementation of the concepts of database at an application level. The title of this project is Music Library/Player. We are using HTML, CSS and PHP as the front end tools. The front end will consist of a webpage with categories of Artists, Albums, Songs, Genres and Playlists. A user can dive deep into the various categories mentioned above. The details of all the categories such as artists, albums, songs, genres, and playlists will be stored in the MySQL database. Features like Search, Filter, and Play etc. will be available on the webpage. These features will be implemented with the help of SQL Queries.

**Chapter 1**

**Introduction**

A database is a collection of information that is organized so that it can be easily accessed, managed and updated. A relational database (RDB) is a collective set of multiple data sets organized by tables, records and columns. RDBs establish a well-defined relationship between database tables. Tables communicate and share information, which facilitates data searchability, organization and reporting.

RDBs use Structured Query Language (SQL), which is a standard user application that provides an easy programming interface for database interaction.

“Music Library” is a software designed to support a music player. A user can search for his interests in various genres, artists and albums and listen to those songs that matter, on the go. In the recent times, the entertainment industry is booming at a rapid rate. People want a gateway from their daily routines and listen to classics, new releases and a lot more.

**1.1 Project Aim and Objectives:**

The aim and objectives of the project are:

* Online streaming of music
* Addition of new Artists, Albums and Tracks
* User login and signup page
* A search bar to find songs, albums and artists in the music library.

**Chapter 2**

**Literature Survey**

**2.1 Front End:**

The front end tools used are HTML5, CSS3, PHP, JavaScript and JQuery.

**2.1.1 HTML5:**

Hypertext Markup Language revision 5 (HTML5) is markup language for the structure and presentation of World Wide Web contents. HTML5 supports the traditional HTML and XHTML-style syntax and other new features in its markup, New APIs, XHTML and error handling. The HTML code is written inside the body tag and various headings, paragraphs, links, input fields are included in separate tags themselves inside the body tag. The required styling and scripting files are included in the head tag inside the html tag.

**2.1.2 CSS3:**

Cascading Style Sheet (CSS3) is a style sheet language used for describing the looks and formatting of a document written in markup language. CSS acts as a presentation tool for a website written in HTML5. It was introduced to enable the separation of document content and document presentation where the user can tweak the font, colours, images and a lot more. It can be used to allow the webpages to be displayed adaptively depending upon the screen size.

**2.1.3 PHP:**

PHP is a scripting language designed to produce dynamic webpages. It can also be used as a standalone application providing GUI. It is a widely used scripting language suitable for web development and can be directly embedded into HTML page. It runs on a web server, such as Apache using softwares like Xampp to load content into the web pages.

**2.2 Back End:**

The Backend tools used are MySQL.

**2.2.1 MySQL:**

MySQL is an [open source](https://en.wikipedia.org/wiki/Open-source_software) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). It is written in C and C++. MySQL works on many platform such as macOS, Linux, Windows.

**2.2.2 SQL Server Features:**

* Ease of installation, deployment and use.
* Scalability.
* Data Warehousing.
* System integration with other software.

**2.2.3 Normalization:**

The normalization process takes a relation schema through a series of tests to certify whether it satisfies a certain normal form. It can be considered as a filtering or purification process to make the design have successively better quality. An unsatisfactory relation schema that does not meet the condition for a normal form is decomposed into smaller relation schemas that contain a subset of the attributes and meet the test that was otherwise not met by the original relation.

**1st Normal Form:**

It disallows relations within relations orrelations as attribute values within tuples*.*

**2nd Normal Form:**

A relation schema R is in 2NF if every nonprime attribute A in R is not partially dependent on any key of R.

**3rd Normal Form:**

A relation schema R is in 3NF if, whenever a  
nontrivial functional dependency X *→* A holds in R, either

1. X is a superkey of R.
2. A is prime attribute of R.

**Boyce-Coyd Normal Form:**

A relation schema R is in BCNF if whenever a nontrivial functional dependency X *→* A holds in R, then X is a superkey of R.

**4th Normal Form:**

A relation is in 4th NF if and only if there exists a multi-valued dependency in a relation A ->> B, then all of the relations are also functionally dependant on A.

**5th Normal Form:**

A relation is in 5th NF if and only if every join dependency in R is implied by the candidate key on R. This is called join dependency.

**Chapter 3**

**System Requirements**

**3.1 Non-functional Requirements**

* **Efficiency Requirement**

When a Music Library system is implemented, a user will easily access searching of songs much faster.

* **Reliability Requirement**

The system should accurately perform user registration, user validation, searching etc.

* **Implementation Requirement**

Implementing the whole system uses HTML5, CSS3, and PHP at the server side connecting the database at the backend using MySQL.

**3.2 Functional Requirements**

**3.3 Software Requirements**

* **Operating System:** Windows 7/8/8.1/10 can be used as the operating systems.
* **Database:** MySQL is used as the back end database as it easy to maintain and retrieve records by using simple queries.
* **Markup and Scripting Languages:** HTML is used to write the contents of the front end while CSS3 is used to style the front end, JavaScript provides additional functionality to the contents of the front end and PHP is used as the scripting language. All the codes are written in Sublime Text 3.
* **Xampp:** Xampp is an open-source web server package consisting of Apache Web Servers.

**3.4 Hardware Requirements**

* **Processor**: Any Intel or AMD processor that supports multi-core capabilities is required for the system to work.
* **RAM:** More than 2GB RAM is sufficient for seamless working of the webpage and the music player.
* **Storage:** Approximately 3-4GB of storage is required to store tracks, codes, assets such as images offline.

**Chapter 4**

**System Design**

**4.1 Introduction**

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development.

This Project is implemented using PHP, which is proven to be a very efficient tool in the field of Server side web application. HTML5, CSS3, Materialize is used to implement the user interface and JavaScript and jQuery is used to make the web page dynamic. PHP is used to connect the interface with MySQL.

**4.2 ER Diagram**

An entity–relationship model or the ER Diagram describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types and specifies relationships that can exist between instances of those entity types.

In [software engineering](https://en.wikipedia.org/wiki/Software_engineering) an ER model is commonly formed to represent things that a business needs to remember in order to perform business processes. Consequently, the ER model becomes an abstract [data model](https://en.wikipedia.org/wiki/Data_modeling) that defines a data or information structure that can be implemented in a [database](https://en.wikipedia.org/wiki/Database), typically a [relational database](https://en.wikipedia.org/wiki/Relational_database).

This ER Model represents the various relationships of the Music Player. The tables are Artist, Albums, Tracks, Genre, Playlists and Users. They have a relationship with the table’s media, stem length, dial and brand. The relationships are defined above as has dial size is of brand, uses media etc.

Album\_name

Artist\_name

Album\_art

Album\_id

Artist\_id

Artist\_photo

Makes

Producer

Albums

Artist

Has Sold

Is in

Belongs to

to

to

Contains

Chart\_position

Albums\_sold

Certification

Genre\_name

Genre\_id

Playlist\_name

Playlist\_id

Play\_time

Song\_file

Track\_no

Song

Track\_id

Release\_year

Playlists

Sales\_charts

Genre

Tracks

Tracks

**Album\_id**

**4.3 Schema Diagram**

**Artist**

|  |  |  |
| --- | --- | --- |
| Artist\_id | Artist\_name | Artist\_photo |

**Albums**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Album\_id | Album\_name | Release\_year | Producer | Artist\_id | Genre\_id | Album\_art |

**Tracks**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Track\_id | Track\_no | Song | Play\_time | Album\_id | Playlist\_id | Song\_file |

**Genre**

|  |  |
| --- | --- |
| Genre\_id | Genre\_name |

**Playlists**

|  |  |
| --- | --- |
| Playlist\_id | Playlist\_name |

**Users**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| userid | username | firstName | lastName | email | password | signUpDate | profilePic |

**4.3 Database Design**

The data in the system has to be stored and retrieved from the database. Data structures and data elements to be stored have to been identified at the analysis stage. They are structured and put together to design the data storage and retrieval system.

The general objective is to make the database access easy, quick, inexpensive and flexible for the user. Relationships are established between the data items and unnecessary data items are removed. Normalization is done to get an internal consistency of data and to have minimum redundancy and maximum stability. This ensures minimising data storage required , minimizing chances of data inconsistencies and optimizing updates.

**4.5.1 Creating Tables**

1. **Creating table Artist:**

Create table Artist

(Artist\_id int primary key,

Artist\_name varchar (20),

Artist\_photo varchar (500));

1. **Creating table Genre:**

Create table Genre

(Genre\_id int primary key,

Genre\_name varchar (20));

1. **Creating table Playlists:**

Create table Playlists

(Playlist\_id int primary key,

Playlist\_name varchar (20));

1. **Creating table Albums**

Create table Albums

(Album\_id int primary key,

Album\_name varchar (30),

Release\_year date,

Producer varchar (30),

Album\_art varchar (500),

Artist\_id int not null,

Genre\_id int not null);

Alter table Albums add foreign key one (Artist\_id) references Artist (Artist\_id) on delete cascade;

Alter table Albums add foreign key two (Genre\_id) references Genre (Genre\_id) on delete cascade;

1. **Creating table Tracks**

Create table Tracks

(Track\_id int primary key,

Song varchar (50),

Track\_no int,

Song\_file varchar (500)

Play\_time varchar (10),

Album\_id int not null,

Playlist\_id int not null);

Alter table Tracks add foreign key three (Album\_id) references Albums (Album\_id) on delete cascade;

Alter table Albums add foreign key four (Playlist\_id) references Playlists (Playlist\_id) on delete cascade;

1. **Creating table Sales\_charts**

Create table Sales\_charts

(Album\_id int primary key,

Certification varchar (30),

Albums\_sold int,

Chart\_position int);

Alter table Sales\_charts add foreign key five (Album\_id) references Albums (Album\_id) on delete cascade;

1. **Creating table Users**

Create table Users

(userid int,

username varchar (45),

firstName varchar (45),

lastName varchar (45),

email varchar (200),

password varchar (32),

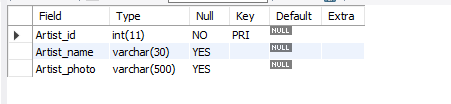
signUpDate datetime,

profilePic varchar (500),

primary key (userid, username));

**4.5.2 Table Description**

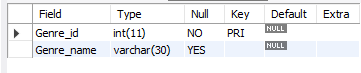
1. **Artist**



This table stores the information related to the various artists in the industry. It consists of 3 attributes.

The primary key is assigned to the Artist\_id attribute and thus cannot store NULL values.

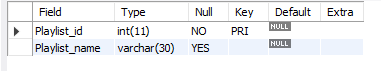
1. **Genre**

****

This table stores the various genres in which artists make music. It consists of 2 attributes.

The primary key is assigned to the Genre\_id attribute and thus cannot store NULL values.

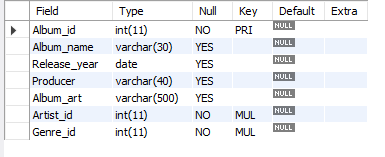
1. **Playlist**



This table stores the various playlists created by the user. It consists of 2 attributes.

The primary key is assigned to the Playlist\_id attribute and thus cannot store NULL values.

1. **Albums**

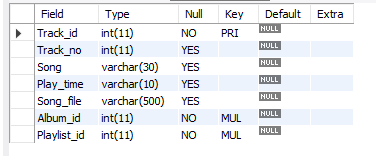


This table stores the various albums created by the artist. It consists of 7 attributes out of which 2 are foreign keys.

The primary key is assigned to the Album\_id attribute and thus cannot store NULL values.

The foreign keys are Artist\_id and Genre\_id of the tables Artist and Genre respectively.

1. **Tracks**

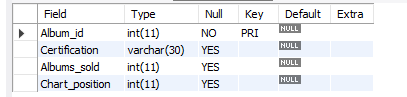


This table stores the various tracks that an album is made up of. It consists of 7 attributes out of which 2 are foreign keys.

The primary key is assigned to the Track\_id attribute and thus cannot store NULL values.

The foreign keys are Album\_id and Playlist\_id of the tables Albums and Playlist respectively.

1. **Sales\_charts**

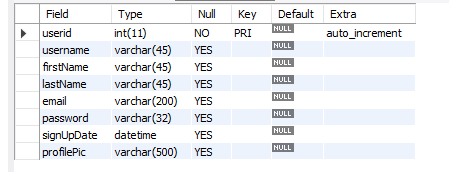


This table stores the details of the performance or success of an album throughout the globe. It consists of 4 attributes out of which 1 is a foreign key.

The primary key is assigned to the Album\_id attribute and thus cannot store NULL values.

The foreign keys is also Album\_id referenced from the Album table.

1. **Users**



This table stores the details of the people who have enrolled in the music library system. It consists of 8 attributes.

The primary key is assigned to the userid attribute and thus cannot store NULL values.

**Chapter 5**

**System Implementation**

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after thorough testing is done and if it is found to work according to the specification.

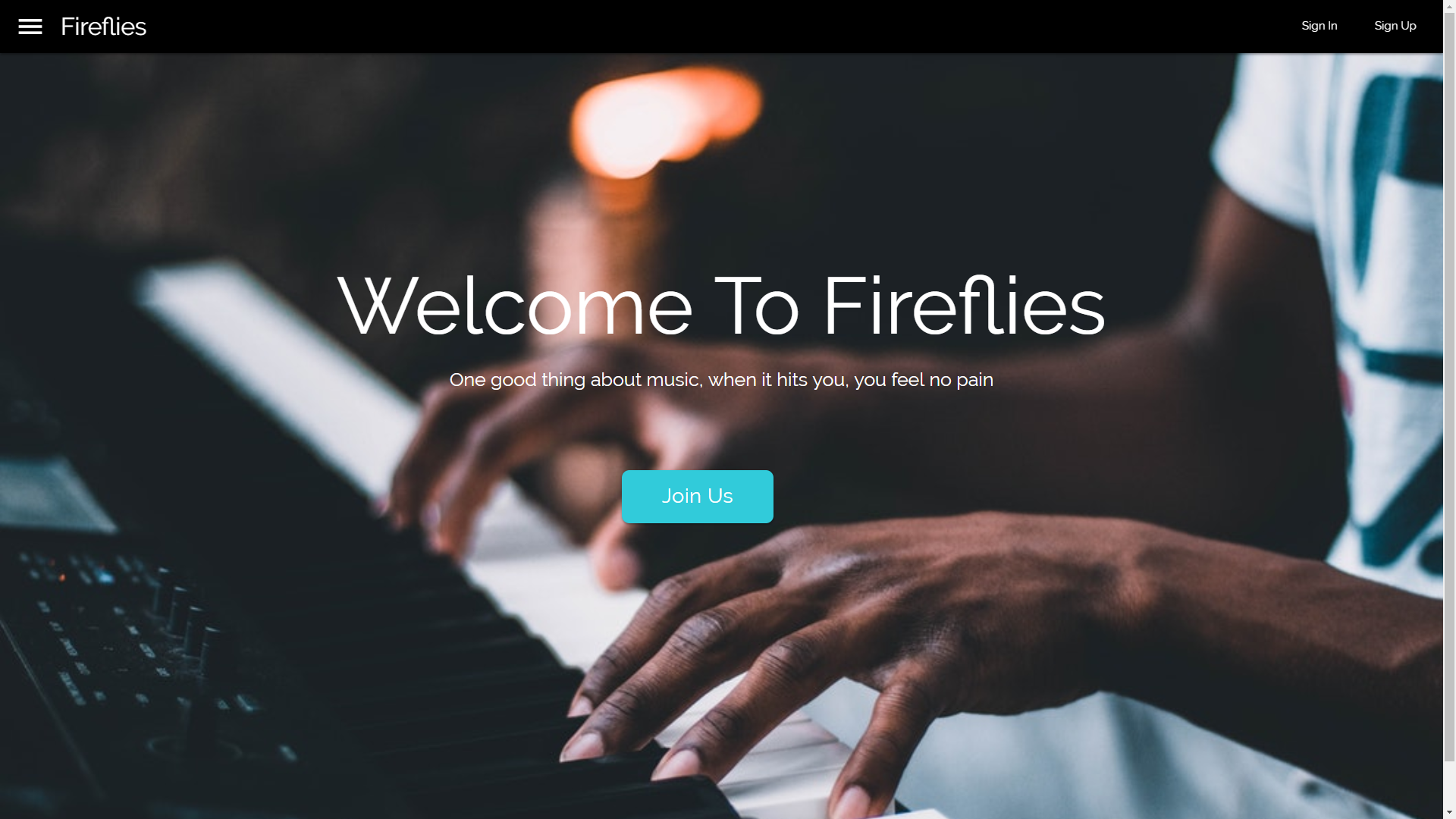
It involves careful planning, investigation of existing systems and its constraints on implementations, design of methods to achieve the change. Two major tasks of preparing the implementation are education and training of the users and testing of the system.

The more complex the system being implemented, the more involved will be system analysis and design effort required just for implementation.

The implementation phase consists of several activities. The required hardware and software acquisition is carried out. The system may require the need to develop a software. For this purpose, programs are written and tested.

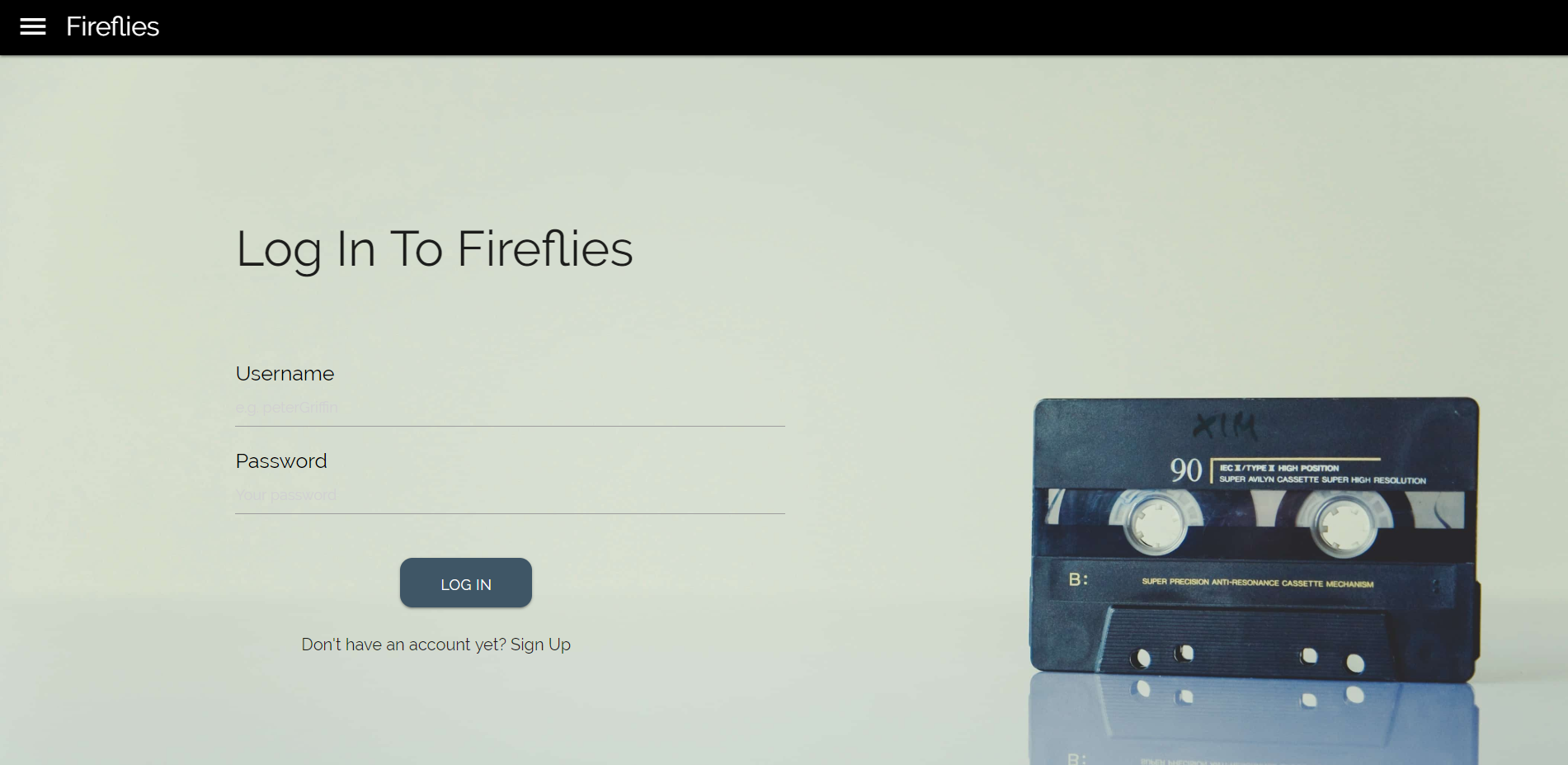
**5.1 Snapshots**

**5.1.1 Home Page**

****

This is the index page of our website: “Music Library”. This page provides the user to register or log in to the website and thus dive deep into the music realm.

**5.1.2 Registration Page**



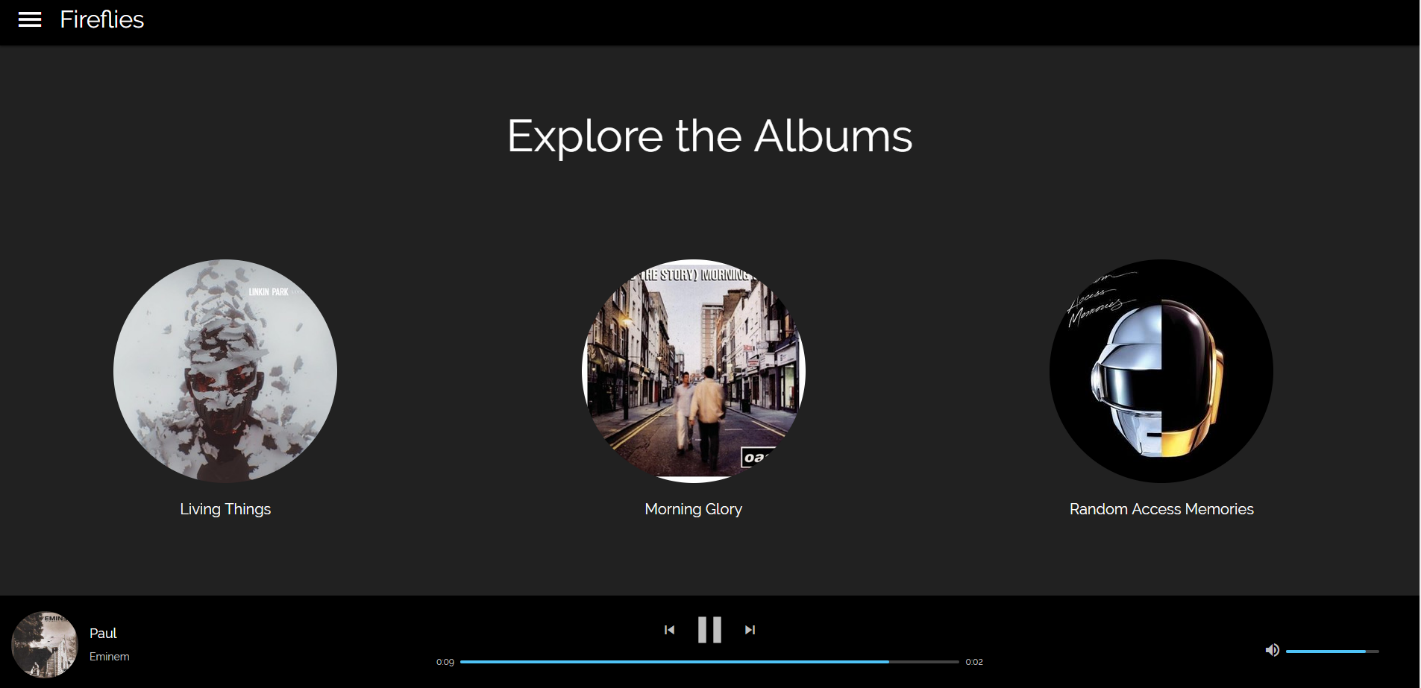


The registration page is required for the user to login to the website, if all already a member or sign up to the website.

For signing up the user has to provide a valid username, email-id and password. If any violations are met, a warning message is sent to the user.

For logging in, the user has to provide his/her username and password to the website and click Submit. Once the user is authenticated, the system redirects the user to the albums page.

**5.1.3 Collective Albums Page**

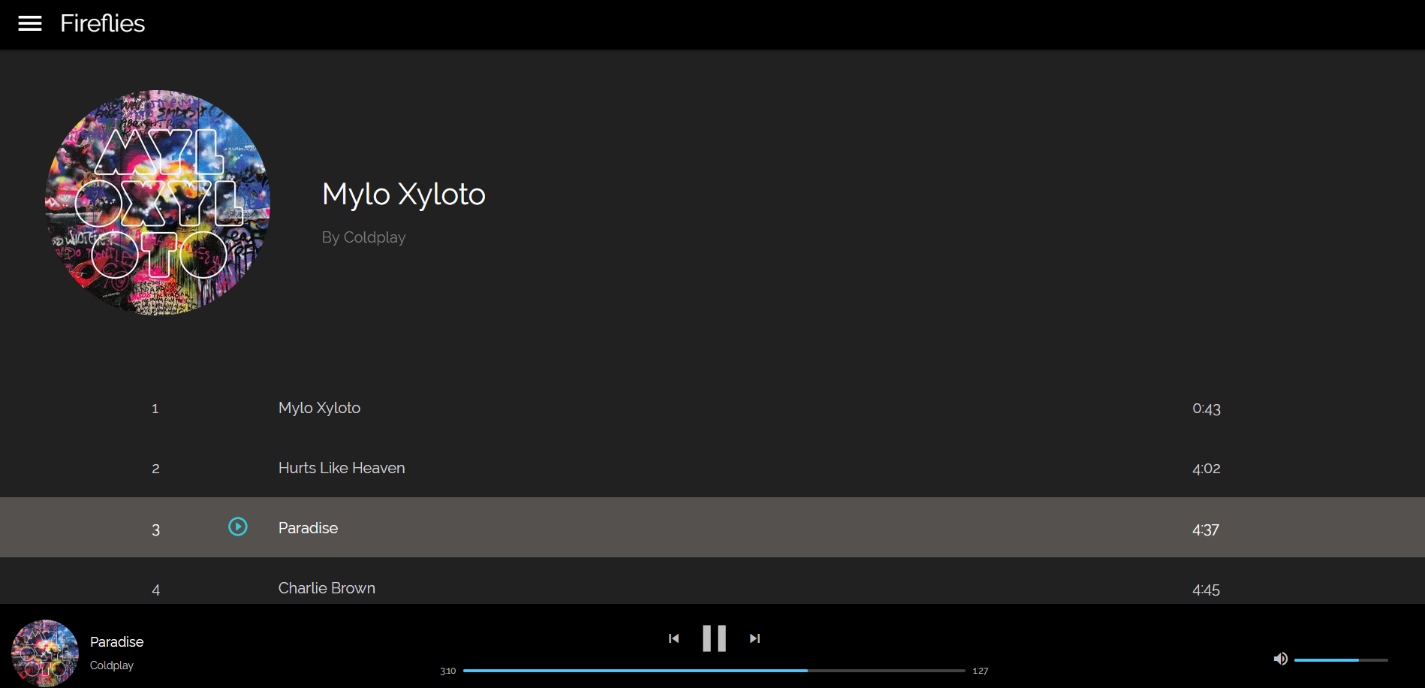


This page displays the grid of all albums in the database. A user can click on particular albums to view the songs in that particular album.

At the bottom of the page lies the playing bar, which can be used to play the songs in the library, skip songs based on your mood etc.

The playing bar consists of the album art, song name and the album name in the left section, provides the user with media controls and on the right hand side of the playing bar lies the volume bar, which can be used to increase/decrease the system volume.

**5.1.4 Album Page**



The album page displays the album name, album art and the artist name at the top section.

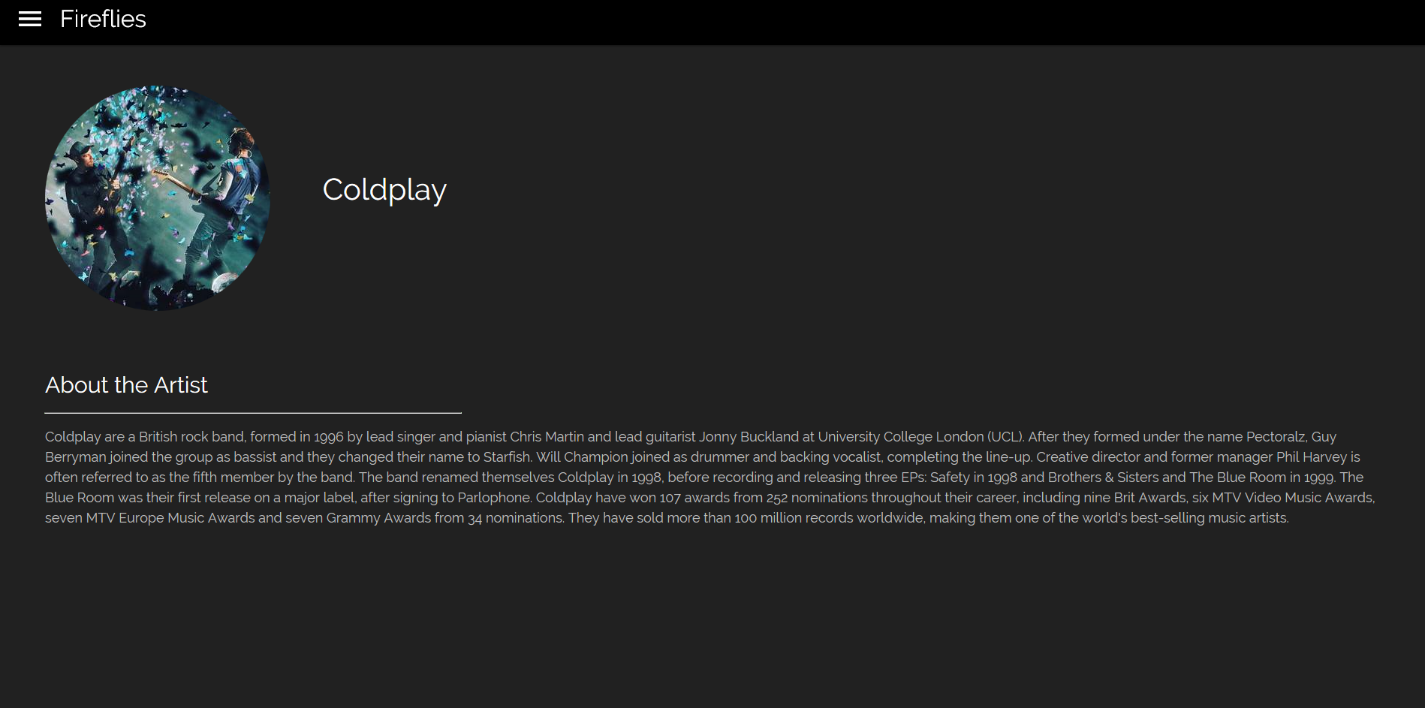
At the next section the page displays the tracks in that particular album, portraying the track number, track name and the playtime.

The system also offers the ability to play particular songs from the album via the play button that is made visible to the user when hovered over the song.

At the bottom of the page, the system displays the playing bar which contains information of the currently playing song, its album name and its album art. It also allows the user to skip to the next o previous song based on his/her likings. At the right section of the playing bar, the system allows the user to change the volume of the system based on the sounds from his/her surroundings.

This webpage also allows the user to explore more about the artist who made the album.0

**5.1.5 Artist Page**



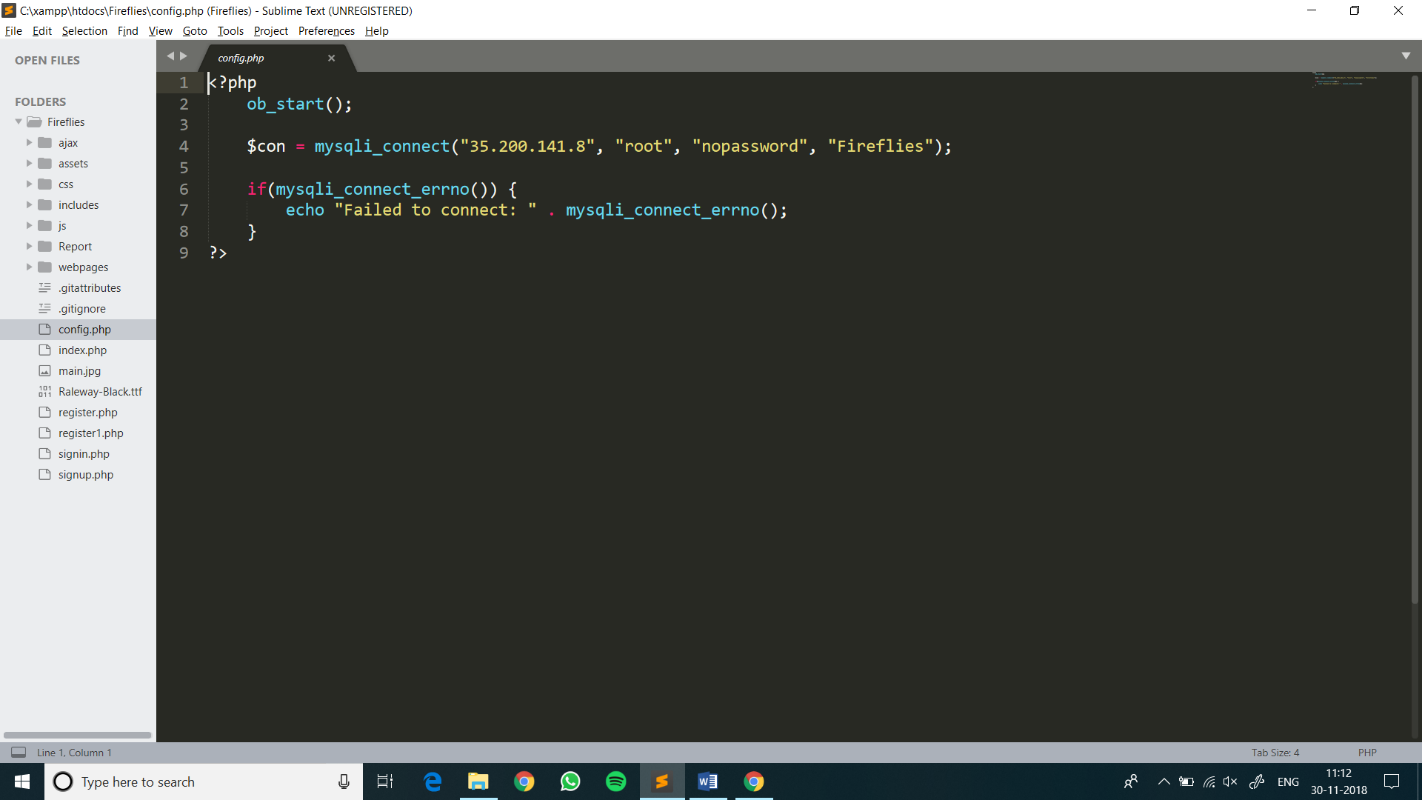
The artist page displays the photo of the band/artist and its name in the top section.

At the lower section it displays some of the interesting facts about the formation and achievements of the band.

**Chapter 6**

**Source Code**

**6.1 Connecting to the Database:**



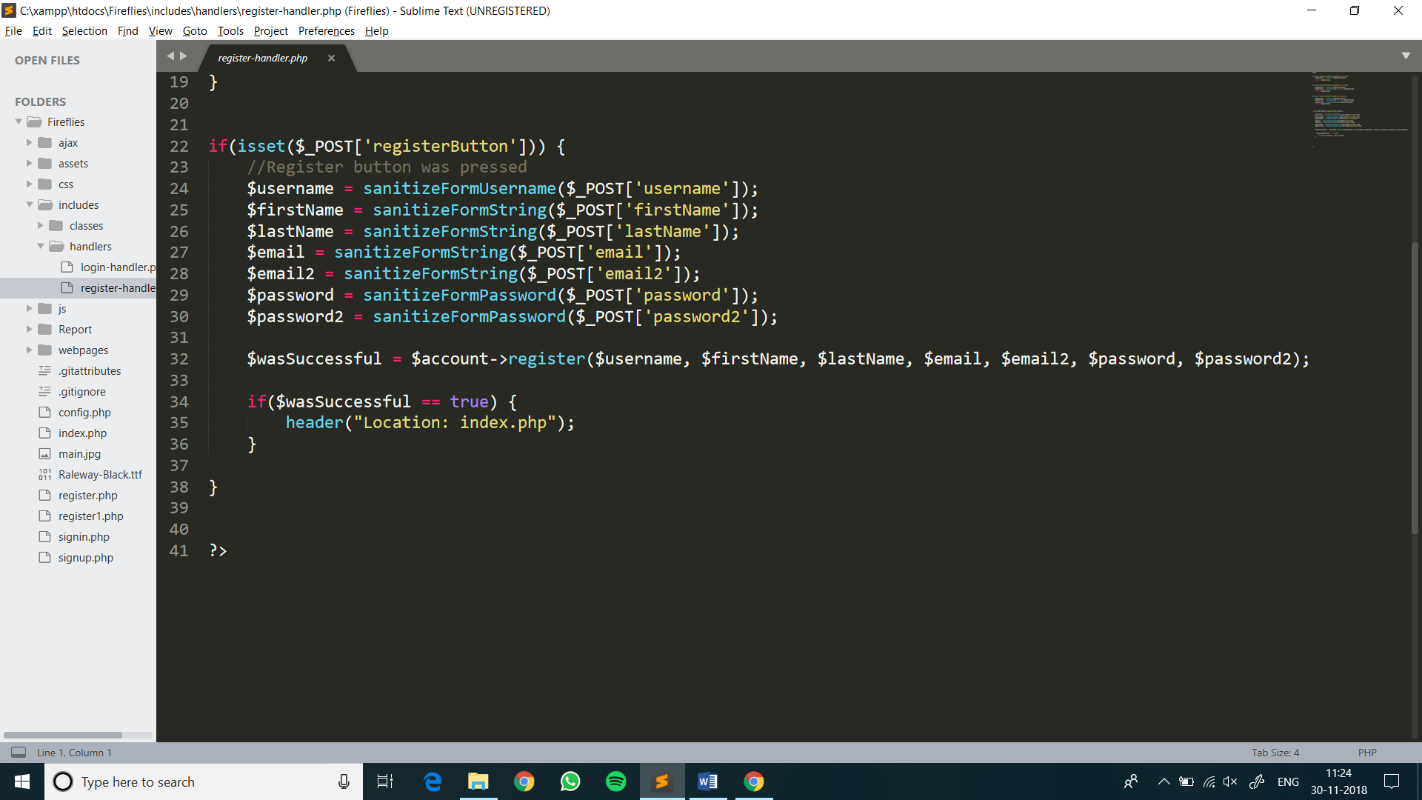
**$con = mysqli\_connect("35.200.141.8", "root", "nopassword", "Fireflies");**

The above function mysqli\_connect() is a build in function in PHP to connect to the MySQL database.  
It accepts 4 arguments, they are:

* Server address
* Username
* Password
* Database Name

This function is the integral part of working on the database from the front end. This function is usually stored in the file called “config.php”.

**6.2 Accepting input from the User:**

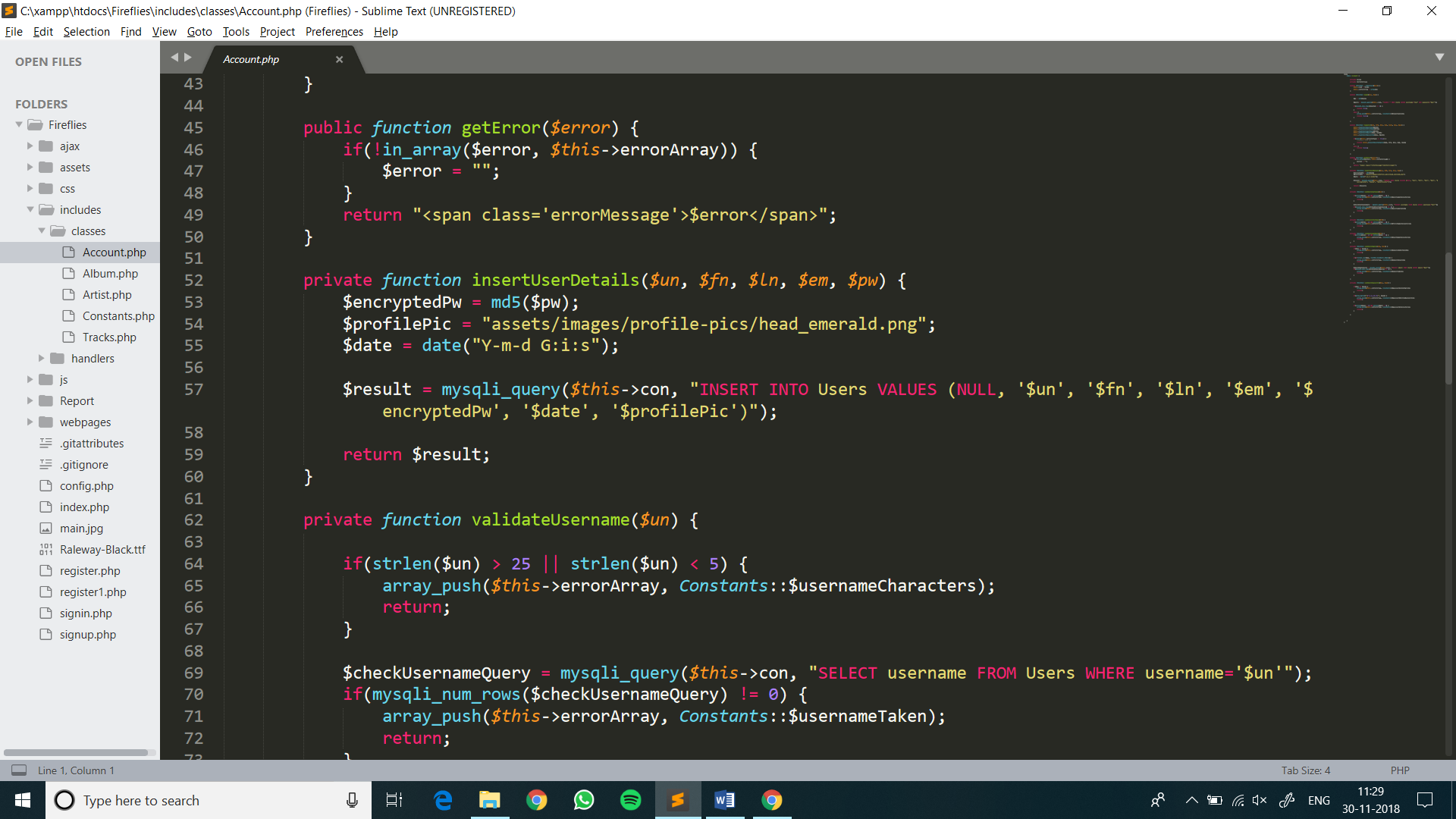
****

In the above code, we are storing the information entered by the user into variables.

**$wasSuccessful = $account->register($username, $firstName, $lastName, $email, $email2, $password, $password2);**

We pass all the variables into the **register()** which is defined in the **Account** class.  
If the registration is successful, then the control is redirected to the main index.php page.

**6.3 Inserting data into the Database:**

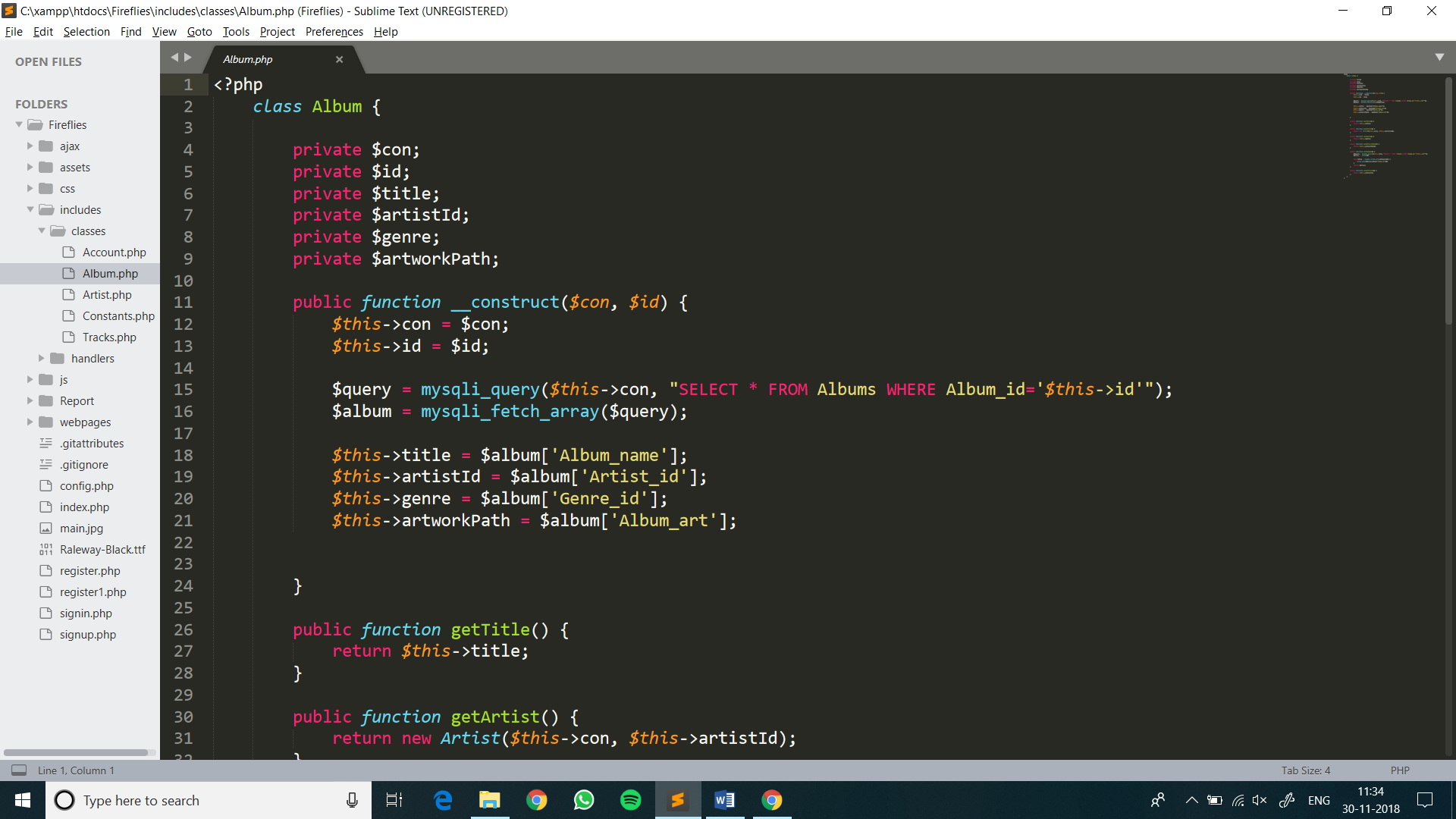


In the above figure, we use the function **insertUserDetails()** to insert the data into the database.

We are passing 8 arguments into the function, as there are 8 attributes in the Users table.   
We encrypt the password using the **md5()** function, so that the password is not visible in the database.

We pass NULL in the 1st argument, as the 1st attribute in the User table is the UserId and it is an incrementing value. Hence, we don’t have to pass any values into it.

**6.4 Extracting info from the database and the Accounts Class:**



In the above figure, we see the **Album** class.

We use this class to extract information about the album from the database and show it on the front end of the project.

We do this using the query:

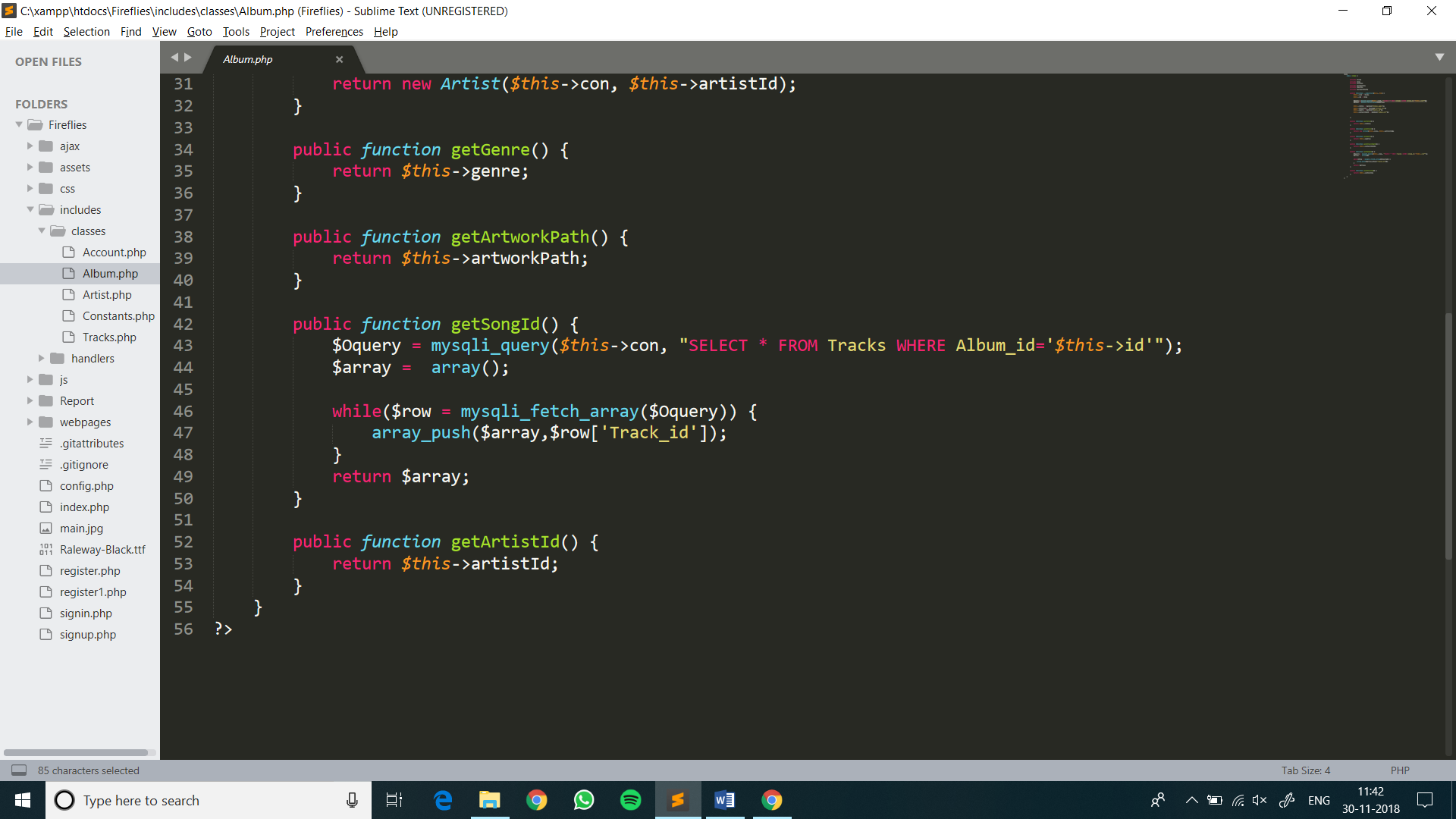
**$query = mysqli\_query($this->con, "SELECT \* FROM Albums WHERE Album\_id='$this->id'");**

where $query is the variable in which the returned values from the queries are stored.

The rows returned from the query are stored in an array called $album.

Using this array we can extract the information like:

* Album\_name
* Artist\_id
* Genre\_id
* Album\_art



In the above figure, we have used another SQL query to extract Track\_id from the Tracks table.

**$Oquery = mysqli\_query($this->con, "SELECT \* FROM Tracks WHERE Album\_id='$this->id'");**

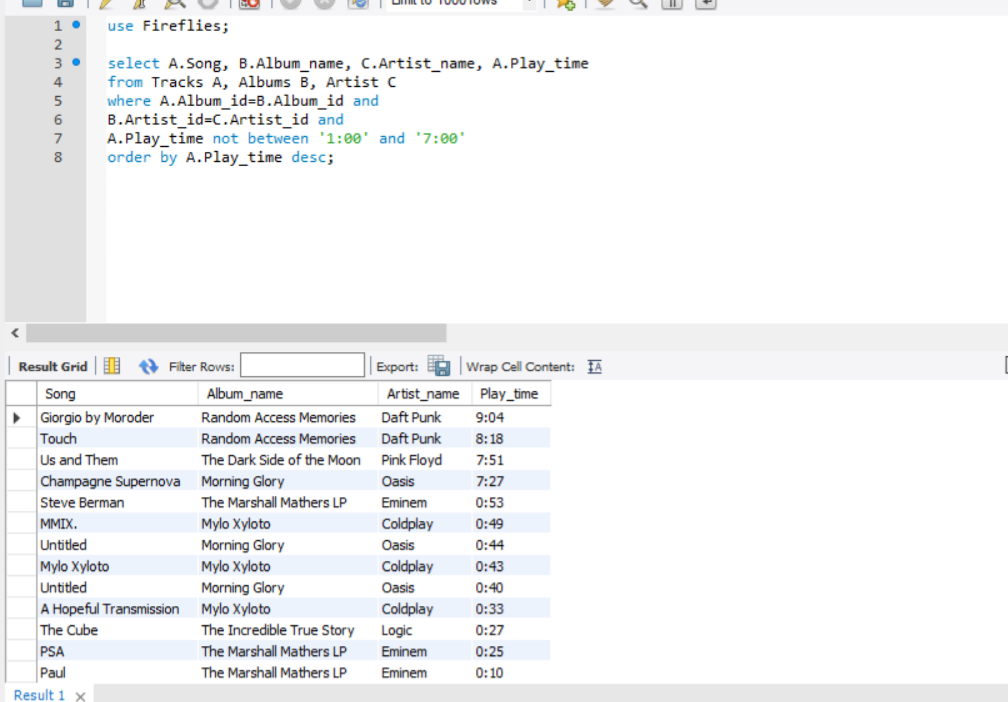
In this query, we pass the Album\_id of the album whose tracks we want to extract.  
$0query stores all the rows returned from the query.

We use the while statement to go through the rows returned. The **mysqli\_fetch\_array()** function fetches the rows from the $0query variable and stores it into the variable $row.  
Extraction of the Track\_id is done using this variable and it is pushed into the array $array to show the information of the songs in the front end.

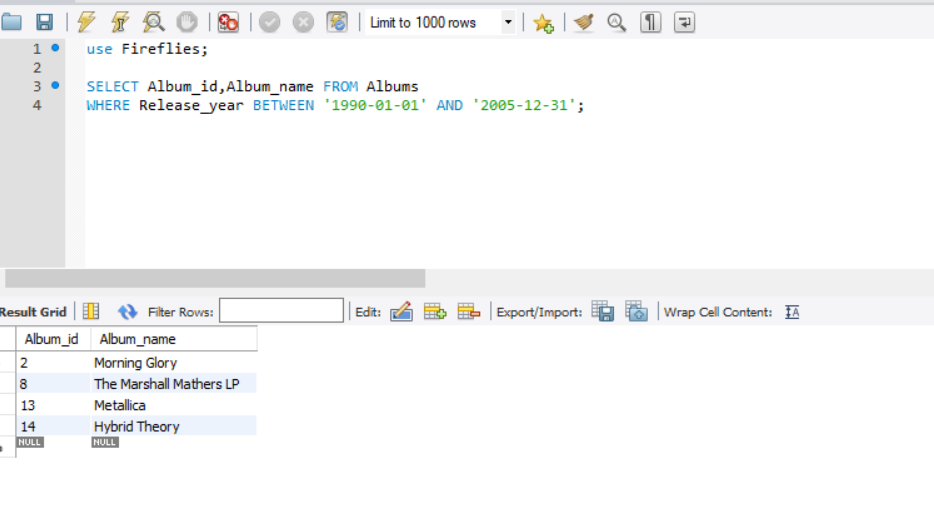
**Chapter 7**

**SQL Queries**

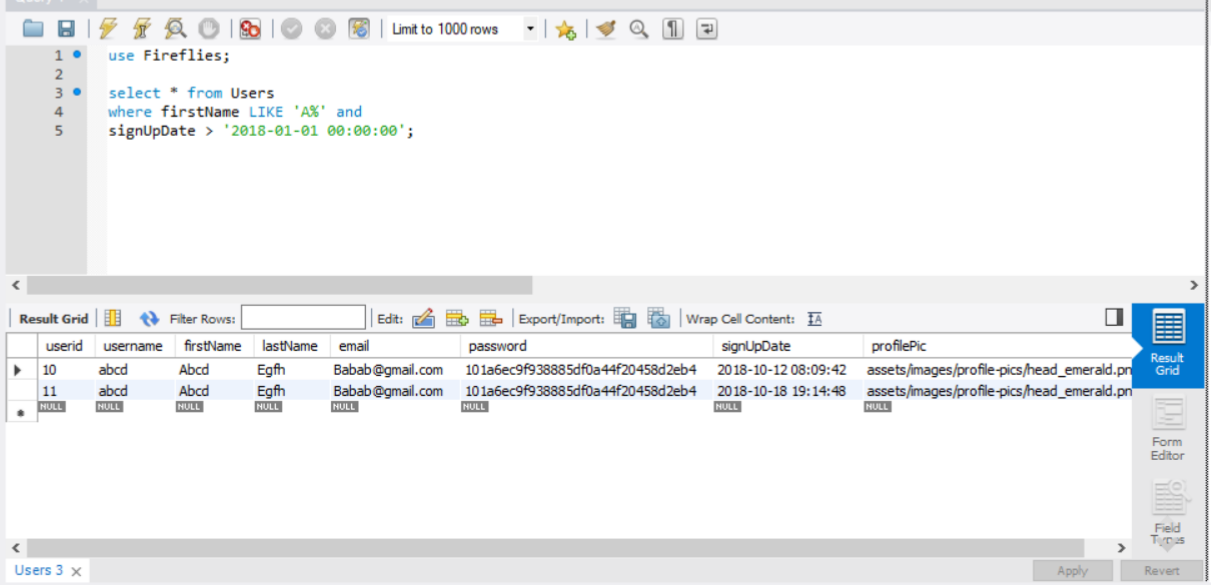
* 1. Select the songs, albums and artists for those songs which have a playtime between 1 minute and 7 minutes.



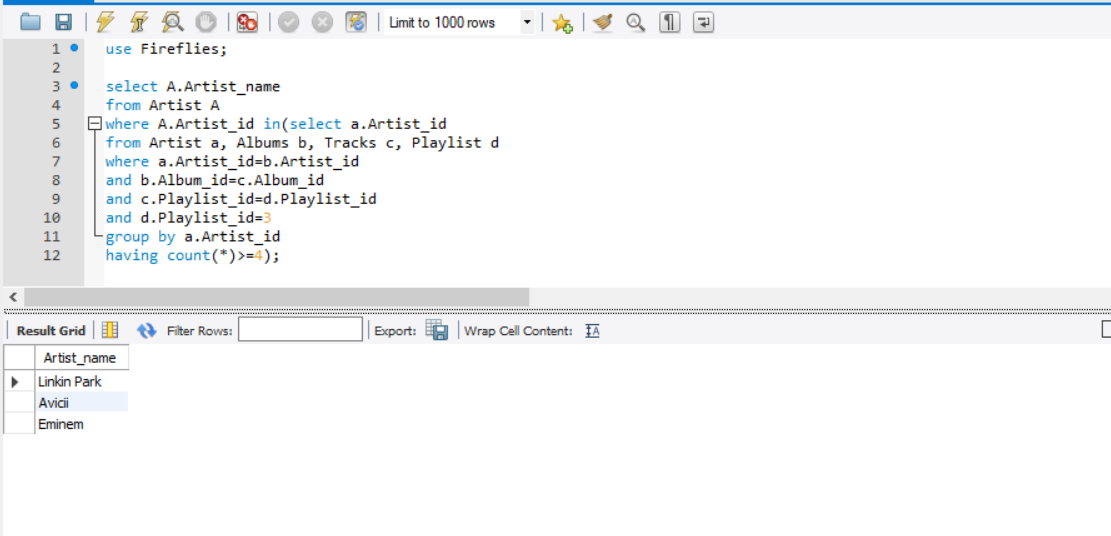
* 1. Find the albums which have been released between 1990 and 2005.



* 1. Determine the user's whose name starts with A and have signed up after 1st January.



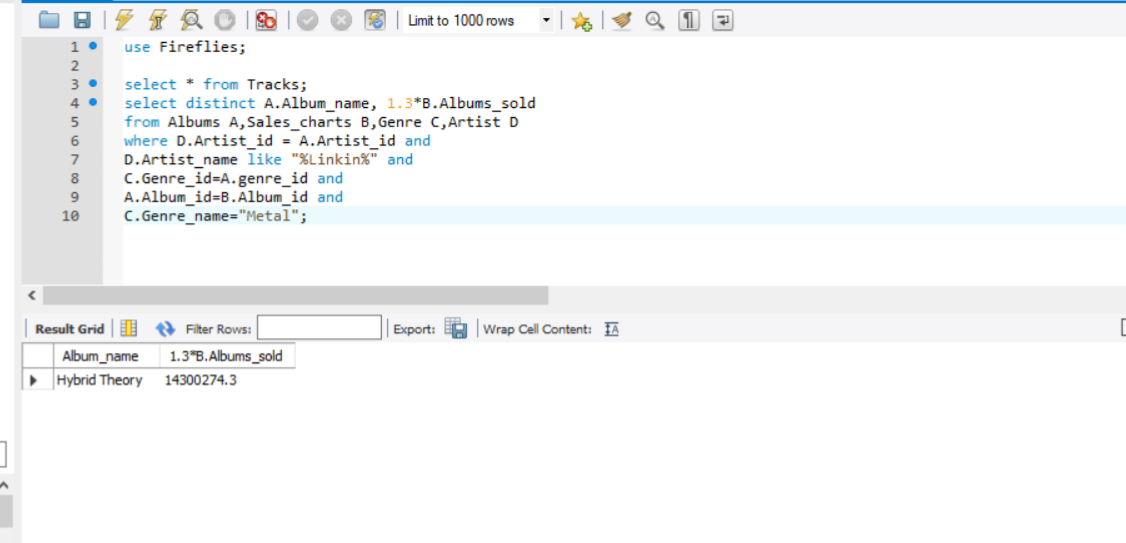
* 1. Find the names of the artists who have 4 or more songs in playlist 3.



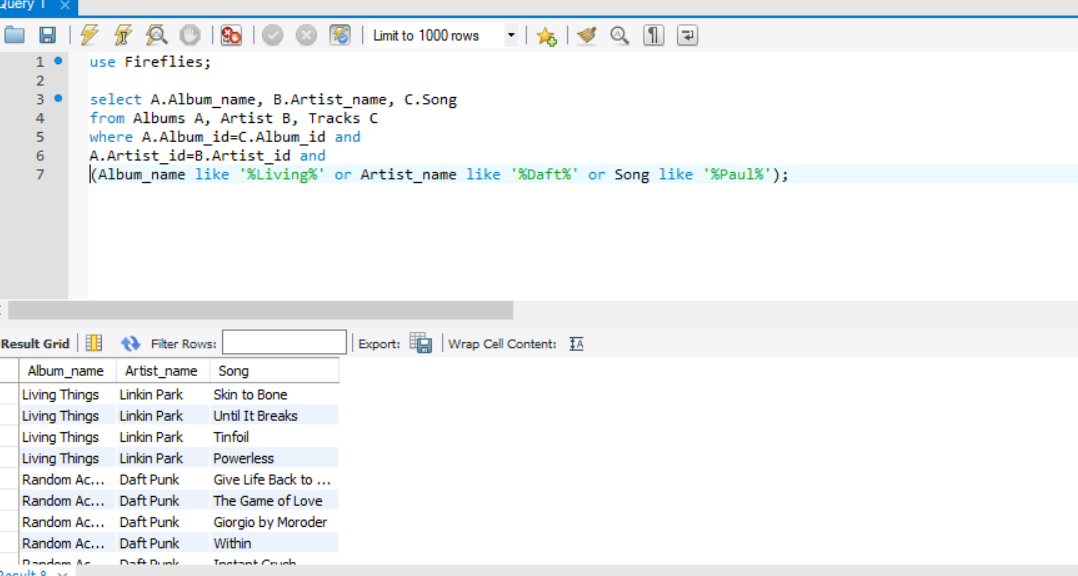
* 1. Find the albums who have sold more than 10000000 records throughout the world and having chart position 1 using inner join.



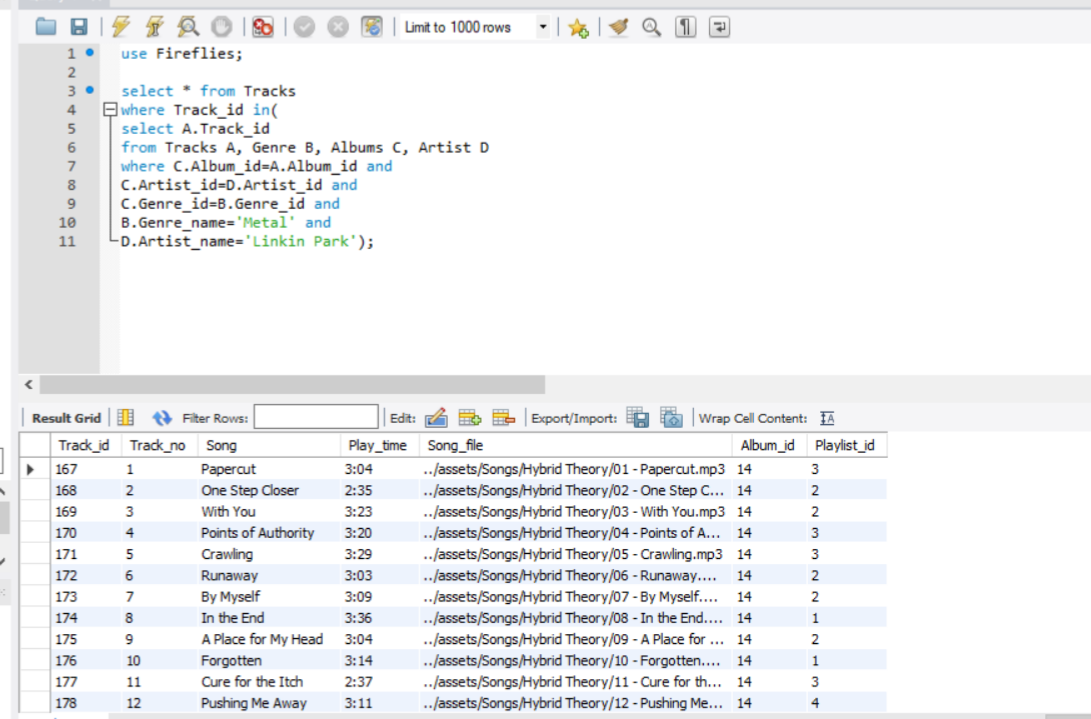
* 1. Increment the number of albums sold by the artist whose name contains linkin by 30% belonging to Metal genre displaying the name of the album and its current albums sold.



* 1. Search and display the Album Name, Artist name and the Track name according to the input.



* 1. Find the songs whose genre is metal and sung by Linkin park.

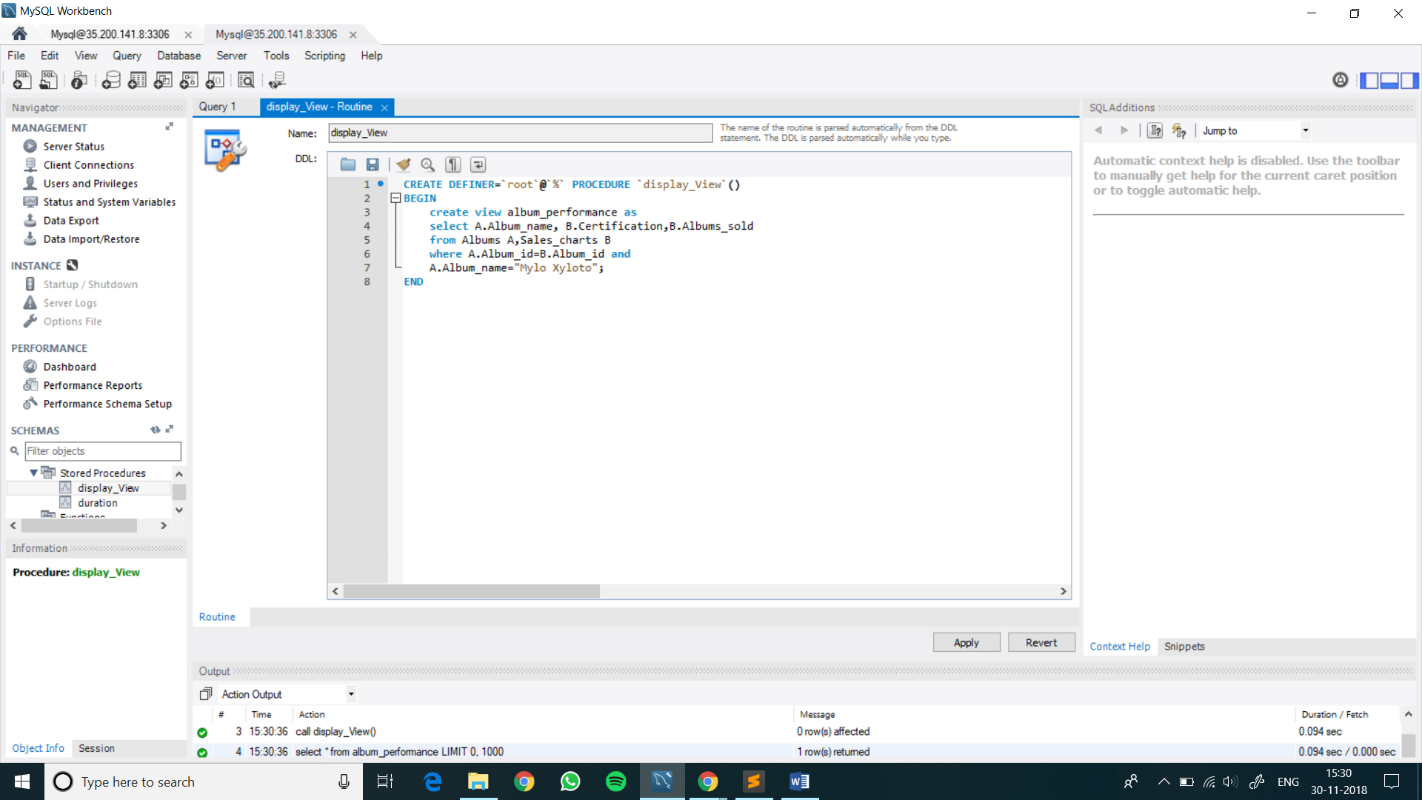


**Chapter 8**

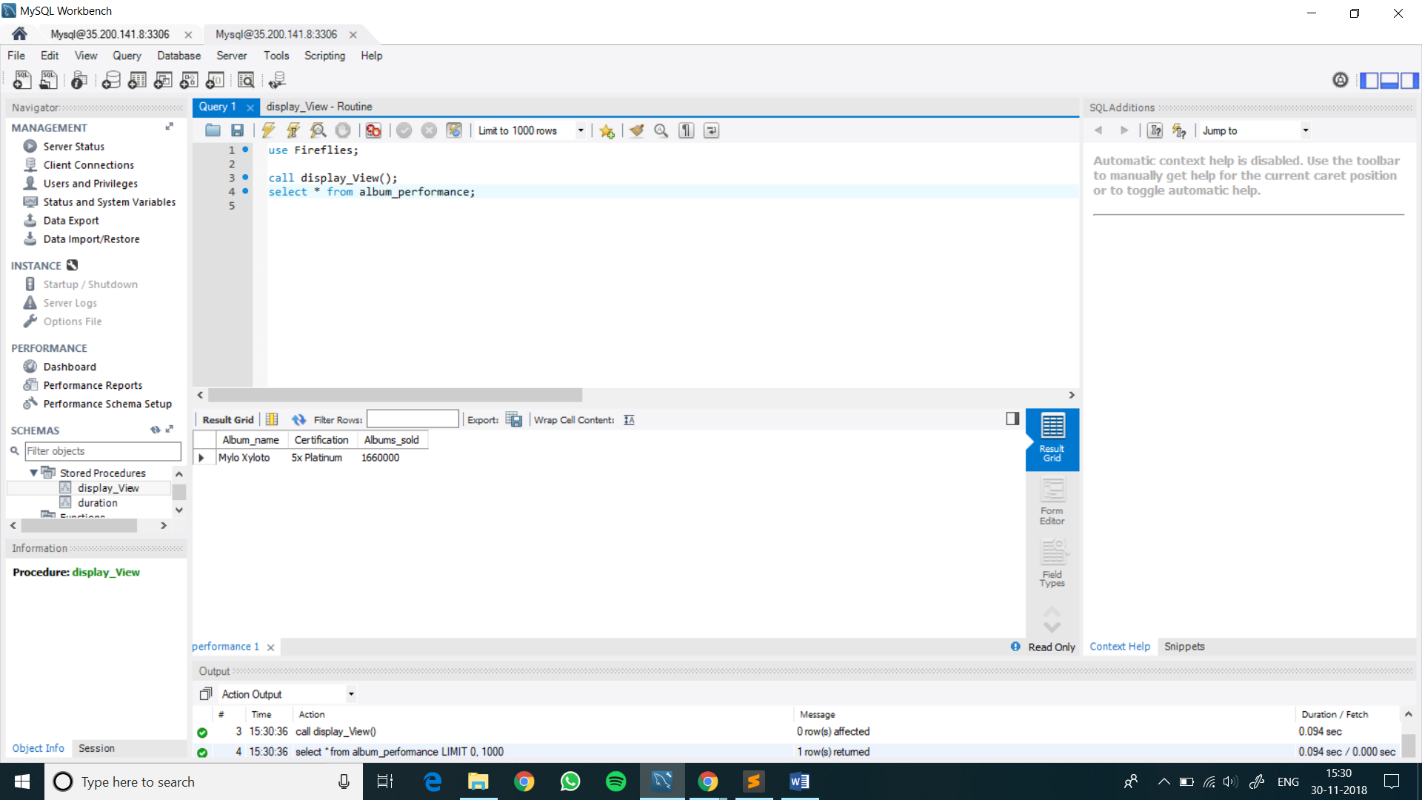
**Stored Procedure**

A stored procedure accepts input parameters and returns multiple output values to the calling procedure.

It contains programming statements that perform operations in the database, including other procedures.



This stored procedure creates a view called album\_performance which displays the Album name, Certification and the albums sold where the album name is “Mylo Xyloto”.



The create procedure in MySQL is called via the “**call *procedure*(parameters);**” statement.

Since we have a view inside the procedure it is called via the select statement.

**Chapter 8**

**Conclusion**

The package was designed in such a way that future modifications could be done easily.

The following conclusions can be made from the development of the above project:

* The music Library system improves the efficiency with which a user listens to music.
* It provides a sophisticated and a user friendly GUI which may be on par with the existing systems.
* It gives appropriate access to authorized users depending upon their permissions.
* It effectively overcomes the delay in accessing files stored locally.
* Updation, insertion and deletion of data stored in the database is made easier for the admin.
* The system has adequate scope for modification in future.

**Chapter 9**

**Bibliography**

The following sources were used to complete the project:

* Fundamentals of Database Systems, by Navathe.
* [www.stackoverflow.com/](http://www.stackoverflow.com/)
* [www.quora.com/](http://www.quora.com/)
* [www.mysqltutorials.com/](http://www.mysqltutorials.com/)
* [www.wikipedia.com/](http://www.wikipedia.com/)
* [www.mdn.org](http://www.mdn.org)
* [www.w3schools.com/](http://www.w3schools.com/)