INTRODUCTION

1.1 OVERVIEW OF THE PROJECT

LAMA music provides access to thousands of music; both mainstream and user submitted, to music lovers around the world. LAMA provides a lightweight online music playing system, where users can easily navigate through the website to find their favorite music. Users have the option of creating playlists with a custom name and adding songs to their playlist, which makes it possible for users to find all their go to songs in one place.

1.2 AIM OF THE PROJECT

Music moves people of all cultures, in a way that doesn't seem to happen with other animals.
LAMA music is a place to bring people from all walks of life for one common goal: Their love of music. It provides a wide array of different genres of music fit for everyone and also in multiple languages.

1.3 SCOPE OF THE PROJECT

The main advantages of the project:

- Simple.
- User friendly
- Easier handling and flexibility
- Use of new technology
- Data can be stored for longer period
- Addition, deletion, modification of records as when needed.
- Seamless page transfer
- Unique temporary playlist for each user in each visit

1.1<u>ABSTRACT</u>

LAMA music is a digital music streaming service that give you access to Millions of songs and other content from artists all over the world. It is a one stop destination for music lovers around the world. It provides access to mainstream and user submitted songs, it provides opportunity to aspiring music creators. It is a platform to both enjoy and share the gift of music with like-minded people. LAMA music provides easy access and navigation within its website, most functionalities within the website can be accessed in few clicks. LAMA music is lightweight as it is not dependent on any framework in both its backend and frontend design. It is the brain child of 4 of its founders Leo, Alex, Melvin and Ajith; coining the name LAMA with their initials.

SYSTEM SPECIFICATION

2.1 SOFTWARE SPECIFICATION

Server side: [MINIMUM]

Operating System	Window, Linux
Web-Technology	PHP
Web server	IIS 7.0,ApachServer
Front-End	HTML, JavaScript, CSS
Back-End	MySQL

Client side: [MINIMUM]

Operating System	Window 7,Linux
Web browser	Any compatible browser.

2.2 HARDWARE SPECIFICATION

Server side: [MINIMUM]

Processor	4.0 GHz or Higher
RAM	4.0 GB
Hard Disk	20 GB

Client side: [MINIMUM]

Processor	2.0 GHz or Higher
RAM	2 GB
Hard Disk	20 GB

DEVELOPING TOOL

3.1 Front End

A front-end developer uses a combination of HTML, CSS, and JavaScript to build everything a user sees and interacts with on a website—that includes everything from front-end features like fonts, sliders, drop-down menus, and buttons, to the overall manner in which web content like photos, videos, and articles are displayed in your web browser.

Front-end developer skills are often thought of in three levels depending on their experience. Developers start at the junior phase for the first few years, although the number of apps they've developed and the apps' complexity are far more important when you determine the right developer for your project. After working alongside more experienced developers, they move to the intermediate phase where they can work on more projects independently. A senior developer can not only code a project, but they can also make decisions about how to design products.

As far as skills in front-end development, look for tools and technologies like:

- Fundamentals like HTML, JavaScript, and CSS
- CSS pre-processors like Sass and LESS
- JavaScript frameworks like Ember, AngularJS, React, etc., or JS-based build tools like Grunt, Gulp, and Bower
- Libraries like jQuery or Backbone.js
- Front-end (CSS) frameworks like Foundation or Bootstrap
- AJAX
- Note: Advanced front-end developers will have a mix of front-end and back-end technology expertise, including solution stacks like MEAN or LAMP, and server-side technologies like Node, Java, or Ruby.

HTML

HTML5 is the latest version of Hypertext Markup Language, the code that describes web pages. It's actually three kinds of code: HTML, which provides the structure; Cascading Style Sheets (CSS), which take care of presentation; and JavaScript, which makes things happen.

HTML5 has been designed to deliver almost everything you'd want to do online without requiring additional software such as browser plugins. It does everything from animation to apps, music to movies, and can also be used to build incredibly complicated applications that run in your browser.

CSS

CSS3 is the latest evolution of the Cascading Style Sheets language and aims at extending CSS2.1. It brings a lot of long-awaited novelties, like rounded corners, shadows, gradients, transitions or animations, as well as new layouts like multi-columns, flexible box or grid layouts. Experimental parts are vendor-prefixed and should either be avoided in production environments, or used with extreme caution as both their syntax and semantics can change in the future.

AJAX

AJAX is about updating parts of a web page, without reloading the whole page. AJAX is the abbreviation for Asynchronous JavaScript and XML. AJAX is a technique for creating fast and dynamic web pages.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page. Classic web pages, (which do not use AJAX) must reload the entire page if the content should change. Examples of applications using AJAX: Google Maps, Gmail, YouTube, and Facebook tabs.

AJAX is based on internet standards, and uses a combination of:

- XML Http Request object (to exchange data asynchronously with a server)
- JavaScript/DOM (to display/interact with the information)
- CSS (to style the data)
- XML (often used as the format for transferring data)

3.2 Back End

PHP

PHP: Hypertext Preprocessor (or simply PHP) is a server-side scripting language designed for web development. It was originally created by Rasmus Lerdorf in 1994; the PHP reference implementation is now produced by The PHP Group. PHP originally stood for Personal Home Page, but it now stands for the recursive initialism PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management systems, and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the de facto standard which other implementations aimed to follow. Since 2014 work has gone on to create a formal PHP specification.

What is the PHP Framework?

Developing an app or a website from scratch involves a lot of work. In many cases, you’ll need to recreate functions that have already been made thousands of times, which is about as efficient as reinventing the wheel. Software frameworks can help you circumvent this problem, by providing you with a foundation you can build upon. Different PHP frameworks are Laravel, Code Igniter , Symphony etc.

Advantages of PHP

1. Cross Platform.

All the PHP based applications can run on various types of platforms. PHP is supported by majority of *Operating Systems*, some of which includes Solaris, UNIX, Windows and Linux. PHP easily interfaces with MySQL and Apache both. And PHP can integrated with various technologies like Java so there is no requirement of re-development. Therefore, saving both time and money, giving it an important advantage.

2. Easy database connection.

PHP has a built-in module that helps it in connecting with database easily. Therefore, PHP has a great demand in the field of web development where a data driven website needs to be developed. PHP significantly *reduces the time* needed in developing the web application that needs an efficient *database management system*.

3. Easy to use.

PHP is widely used because it is easy to use. In contrast with other programming languages that are complex, PHP is simple, fluent, clean and organized, hence it is a boon for the new users. PHP has a well-organized syntax which is logical at the same time.

PHP does not require any intensive studying or manual to *use* it. Command functions of PHP are easily understood as the user can easily figure out from the name of the commands itself what it does. A person who is new to PHP can still code because the syntax is somewhat similar to C.

4. Open source.

One of the important advantages of PHP is that it is *Open Source*. Therefore, PHP is readily available and is entirely free. In contrast to other scripting languages used for web development which requires the user to pay for the *support files*, PHP is open to everyone, anytime and anywhere.

A beginner in PHP need not worry about the support as PHP is maintained and developed by a large group of PHP developers which helps in creating support community of PHP that helps people in PHP implementation and manipulation.

3.3 DESCRIPTION OF DATABASE

MYSQL

MySQL is an open source relational database management system (RDBMS). Its name is a combination of “My”, the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create DB.

MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and Word Press. MySQL is also used by many popular websites, including Google (though not for searches), Facebook, Twitter, Flickr and YouTube.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, Net BSD, Novell NetWare, Open BSD ,Open Solaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO Open Server, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plugins, but otherwise shares the version numbering system and is built from the same code base.

PLATFORM

Windows 8

Windows 8 is a personal computer operating system that is part of the Windows NT family. Windows 8 introduced significant changes to the Windows operating system and its user interface (UI), targeting both desktop computers and tablets. It is a touch-optimized platform based on the modern metro design architecture, which specifies how applications are delivered and rendered in the UI.

Along with having a much different look and feel from its predecessor, Windows 7, Windows 8 also boasted faster startup times and better performance, but it failed to reach critical mass among business and consumer users alike.

Windows 8 major features

The lack of a Windows 8 Start menu marked a major departure from past versions of the operating system. Microsoft replaced the well-known Start menu with a larger Start screen, which provided access to applications and other resources through Live Tiles, a set of icons laid out in a grid. This design offered users an experience similar to that of mobile devices.

The OS also marked the debut of the Charms bar for accessing specific features, such as desktop settings and universal search. The Charms bar appeared on the right side of the screen and was available to users at any time, no matter what they were doing.

Windows 8 also came with USB 3.0 support, an updated Task Manager, Unified Extensible Firmware Interface Secure Boot and a number of preinstalled applications, such as Photos, Mail, Calendar and Messaging. In addition, Windows Explorer was rebranded File Explorer and included a Microsoft Office-like ribbon -- a bar across the top of File Explorer that features items users can click to create a new folder, copy a folder or accomplish other tasks. Windows 8 also provided file backup through the File History feature, which constantly protected users' personal files in several folders, including Favorites and Contacts.

When logging onto Windows 8, users provided their Microsoft online account credentials, unless they specifically overrode this default behavior. This approach allowed Windows 8 to synchronize a user's settings across devices.

The OS also integrated with the Windows Store and the SkyDrive cloud storage service -- now Microsoft One Drive. Other features included improved multimonitor support, ISO mounting, refresh and restart capabilities, and Windows Defender -- Windows' built-in antimalware software.

Windows 8 had the following system requirements:

- Processor: 1 GHz CPU or faster;
- RAM: 1 GB for 32-bit editions and 2 GB for 64-bit editions;
- Storage: 16 GB for 32-bit editions and 20 GB for 64-bit editions; and

- Graphics: Microsoft DirectX 9-capable video card with the Windows Display Driver Model.

Windows 8 product history and reception

Microsoft released Windows 8 to manufacturing in August 2012 and made the OS generally available in October 2012. Upon Windows 8's release, Microsoft invested heavily in getting customers to upgrade from Windows 7 or earlier OSes, spending more than $1.5 billion in marketing, according to Forbes. By the end of January 2013, Microsoft had sold 60 million licenses, which included upgrades and OEM sales.

Device Stage: Whereas Vista barely seems to recognize the presence of cameras, phones, printers, and other external devices, Windows 8's Device Stage treats them like royalty. The operating system devotes a slick-looking status window to each device, so you can browse files, manage media, and perform other device-specific tasks.

Home Group: At long last, Microsoft promises to take the pain and frustration out of home networking for users of its operating system. Set up a Home Group, and then add PCs and other devices--and without further ado you can share files, printers, and the like.

Jump Lists: Like souped-up Recent Documents menus, Jump Lists provide quick access to application-specific documents and/or tasks.

Libraries: Most of us have documents, music, pictures, and video scattered across multiple folders on our PCs. Libraries are special folders in Windows 8 that catalog these items under a single roof, regardless of where you actually store them on your hard drive

One-click Wi-Fi: Unlike Windows Vista, Windows 7 makes choosing a wireless network to connect to simple and convenient: Click the system-tray icon, and choose from the resulting list of available hotspots. Granted, you can find third-party connection managers for Vista, but nothing this streamlined and unobtrusive

SYSTEM ANALYSIS

4.1 INTRODUCTION

System analysis is the performance management and documentation of activities related to the four life cycle phases of any software namely:

- The Study Phase
- The Design Phase
- The Development Phase
- The Operators Phase

System analysis is a vast field of study through which system analyst puts his thoughts and searches for the solution of problem. He has to get a clear idea of what he has in hand and what he has to produce. He has to extract the essence of expectations. He has to satisfy the user in the very possible way. System analysis needs and should include the following steps of study:

➢ Study of current methods, the basic inputs available and output desired.

➢ The splitting of a variable inputs into (.dbf) files so as to reduce redundancy and increase consistency.

➢ Give the idea of key – field (if any).

➢ Ideas regarding code generation.

Software Analysis starts with a preliminary analysis and later switches on to a detailed one. During the preliminary analysis the Analyst takes a quick look at what is needed and whether the cost benefits. Detailed analysis studies in depth all the cornered factors, which builds and strengthens the software. A system study is a step-by-step process used to identify and then developed the software needed to control the processing of specific application. System study is also known as SDLC (Software Development Life Cycle).

Steps of SDLC are:

1. Problem Definition
2. Feasibility Study
3. System Analysis
4. System Design
5. Implementation
6. Post Implementation
7. Maintenance

PROBLEM DEFINITION

The first step in system study is to identify particular problem to be solved or the tasks to be accomplished and setting the system goals to be achieved after defining the problem, the goals of the project are reviewed which has to be fulfilled in order to complete it successfully.

Commencement of my project was done with an objective of developing an application which reduces the work load and increases the system performance.

4.2 EXISTING SYSTEM

Gaana.com is the largest Indian commercial music streaming service with over 150 million monthly users. It was launched in April 2010 by Times Internet and provides both Indian and international music content.

In the existing system, user has to select playlist from predefined albums. Songs are not catered to user's interest. Gaana.com is very heavy to load which bottlenecks devices with slow internet access. Gaana.com is difficult to navigate using its website.

4.3 FEASIBILITY STUDY

After doing the project LAMA Music, study and analyzing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible – given unlimited resources and infinite time

Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

1. TECHNICAL FEASIBILITY

- ✓ Does the necessary technology exist to do what is been suggested.
- ✓ Does the proposed equipment have the technical capacity for using the new system
- ✓ Are there technical guarantees of accuracy, reliability and data security?
- ✓ The environment requires in the development of the system is any windows platform.

- ✓ The observer pattern along with factory pattern will update the results eventually.
- ✓ Around existing environment and to what extend it can support the proposed system. While considering the technical factors of the organization that it presently have is sufficient to implementation of the new system.

2. OPERATIONAL FEASIBILITY

Question that going to be risen in the operational feasibility are:

- ✓ Will the system be used if it developed and implemented.
- ✓ If there was sufficient support for the project from the management and from the users.
- ✓ Have the users been involved in planning and development of the Project.
- ✓ Will the system produce poorer result in any respect or area?

This system can be implemented in the organization because there is adequate support from management and users.

3. ECONOMICAL FEASIBILITY

The system developed and installed will be good benefit to the organization. The system will be developed and operated in the existing hardware and software infrastructure. So there is no need of additional hardware and software for the system. The System developed can reduce the cost overheads of the organization in providing the internet services for establishing the communication channel between the organization staff in conducting the organizational works in smoother and time effective manner.

4.4 PROPOSED SYSTEM

The software is supported to eliminate and in some cases reduce the hardships faced by this existing system. The new system helps to control the problems that are usually happened in any event. The system allows only registered users to login and new users are allowed to register on the application .It allows the users to select from a list of event types like marriage, stage shows, private parties etc. The system then allows the users to select the date and time of event, place and the event equipment's. The system provides each and every information of the event like event calendaring, venue, scheduling etc. The administrator can interact with the client as per his requirements. The system also provides an online chatting facility between client and administrator. The system also calculates cost for all the resources required during the event. The client can also send his requirements and feedback to the administrator though feedback column. The client can update his details at any time. It can also update his booking details before submitting. The system also keeps the clients pervious history of booking and events. It is also linked with Google map which will provides detailed information about the venue location .It is used to know the location over the internet. The system is effective and saves time and cost of the users.

4.4.1 The benefits of the proposed system:

- Simple.
- User friendly
- Easier handling and flexibility
- Use of new technology
- Data can be stored for longer period
- Addition, deletion, modification of records as when needed.
- Seamless page transfer
- Unique temporary playlist created for each user in each visit
- Framework independent backend and front-end design
- High level authentication methods
- SQL injection avoidance methods implemented
- Lightweight website with less stress on user device.

SYSTEM DESIGN

5.1 INTRODUCTION

The most creative and challenging phase of the system development is system design, is a solution to how to approach to the creation of the proposed system. It refers to the technical specification that will be applied. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Design goes through the logical and physical stages of development. At an early stage in designing a new system, the system analyst must have a clear understanding of the objectives, which the design is aiming to fulfill. The first step is to determine how the output is to be produced and in what format. Second input data and master files (database) have to be designed to meet the requirements of the proposed output. The operational (processing) phases are handled through program construction and testing.

The system design includes

- ❖ Modularization
- ❖ Database design
- ❖ Input design
- ❖ Output design

5.2 MODULARIZATION

Structured design partitions a program into small, independent modules. They are arranged in a hierarchy that approximates a model of the business area and is organized in a top-down manner. Structured design is an attempt to minimize complexity and make a problem manageable by subdividing it into smaller segments, which is called modularization or decomposition. The primary advantage of this design is as follows:

- ✓ Critical interfaces are tested first.
- ✓ Early versions of the design, though incomplete, are useful enough to resemble the real system
- ✓ Structuring the design provides control and improves morale.
- ✓ The procedural characteristics define the order that determines processing.
- ✓ Modules that perform only one task are said to be less error-prone than modules that perform multiple tasks.

Administrator Login:

Functions provided for the Admin are:

- ▶ Manage users
- ▶ Manage category of music
- ▶ Manage music categories
- ▶ Manage services provided by them
- ▶ Report generation

User Login:

Functions provided for the customer are:

- ▶ View Manage profile
- ▶ Music player
- ▶ Manage playlist
- ▶ View artist pages
- ▶ Traverse through unique playlist
- ▶ Change credentials

Modules of the system are as follows:

- Registration
- User login
- Search
- Playlist
- History
- Admin panel
- Upload
- User panel
- Music player

1. Registration

New users can use the Registration Module to enter their details and create an account with a unique username

2. User Login

Existing users can login using their unique username and password.

3. Search

Logged in users can search for their favorite songs using the search function,

Search history is logged each time they search for a song

4. Playlist

Users can create a personalized list of songs curated from the songs listed on the website and set the order in which they are to be played.

5. History

All the songs/terms searched in the search module will be logged in the history module and displayed when the user uses the search function again.

6. Admin Panel

When the admin is signed in, the admin can use the admin panel to manipulate Song data, manage creator’s permission and access.

8. Upload

This module can be used to upload songs into the website.

9. Creator Panel

Users with creator privileges can upload songs into the website using the creator panel.

10. Music Player

Users can play, pause or skip songs using the music player.

5.3 INPUT DESIGN

Input design is the process of converting user-oriented inputs to computer based format. It also includes determining the record media, method of input, speed of capture, and entry into system. Inaccurate input data are the most common cause of errors in data processing.

Errors entered by the user can be controlled by input design and control checking. Input design is the process of converting user originated inputs to a computer based format. In the system design phase the Data Flow Diagram (DFD) identifies logical data flows, data stores, source and destinations.

A system flow chart specifies master files (Data Base) transaction files and computer programs. Input data are collected and organized into groups of similar data.

5.4 OUTPUT DESIGN

The outputs can be in the form of operational documents and lengthy reports. The input records have to be validated, edited, organized and accepted by the system before being processed to produce outputs. The most important and direct source of information to user efficient ,intelligible output design should improve the systems relationship with the user and help in decision making. A major form of output is hard copy from the printer. Printouts should be designed around the output requirements of the user. The output devices to consider depends on the factors such as compatibility of the device with the system response time requirements, expected print quantity and quality requirements, expected print quantity and quality.

5.5 DATABASE DESIGN

Data Base design is the logical form of design of data storage in the form of records in a particular structure in the form of tables with fields which is not transparent to the normal user but it actually act as the backbone of the system. As we know database is a collection of which helps the system to manage and store data is called database management system. Data base management system builds some form of constraints like integrity constraints, i.e., the primary key or unique key and referential integrity which help to keep data structure storage and access of data from tables efficiently and accurately and take necessary steps to concurrent access of data and avoid redundancy of data in tables by normalization criterions.

Normalization is the method of breaking down complex table structures into simple table structures by using certain rules thus reduce redundancy and inconsistency and disk space usage and thus increase the performance of the system or application which is directly linked to the database design and also solve the problems of anomalies.

There are different forms of normalization, some are:

- ✓ First normal form (1NF)
- ✓ Second normal form (2NF)
- ✓ Third normal form (3NF)
- ✓ Boyce Codd normal form
- ✓ Forth normal form (4NF)
- ✓ Fifth normal form (5NF)

The data base design of the new system is in second normal form and every non key attribute is functionally depend only on the primary key. The master and transaction tables and their structure are shown below.

TABLE DESIGNS

Table Schema for USERS Table		
Field	**Type/Size**	**Constraint**
ID	INT	NOT NULL
USERNAME	VARCHAR(20)	PRIMARY KEY
FIRSTNAME	VARCHAR(30)	NOT NULL
LASTNAME	VARCHAR(30)	NOT NULL
EMAIL	VARCHAR(20)	NOT NULL
PASSWORD	VARCHAR(256)	NOT NULL
SIGNUPDATE	DATETIME	NOT NULL
STATUS	INT	NOT NULL
PROFPIC	VARCHAR(50)	NOT NULL

Table Schema for SONGS Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
TITLE	VARCHAR(20)	NOT NULL
ARTIST	INT	FORIEGN KEY
ALBUM	INT	FORIEGN KEY
LANGUAGE	INT	FORIEGN KEY
GENRE	INT	FORIEGN KEY
DURATION	TIME	NOT NULL
PATH	VARCHAR(50)	NOT NULL
ALBUMORDER	INT	NOT NULL
PLAYS	INT	NOT NULL

Table Schema for ALBUM Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
TITLE	VARCHAR(20)	NOT NULL
ARTIST	INT	FORIEGN KEY
GENRE	INT	FORIEGN KEY
ARTWORKPATH	VARCHAR(50)	NOT NULL

Table Schema for ARTIST Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
NAME	VARCHAR(20)	NOT NULL

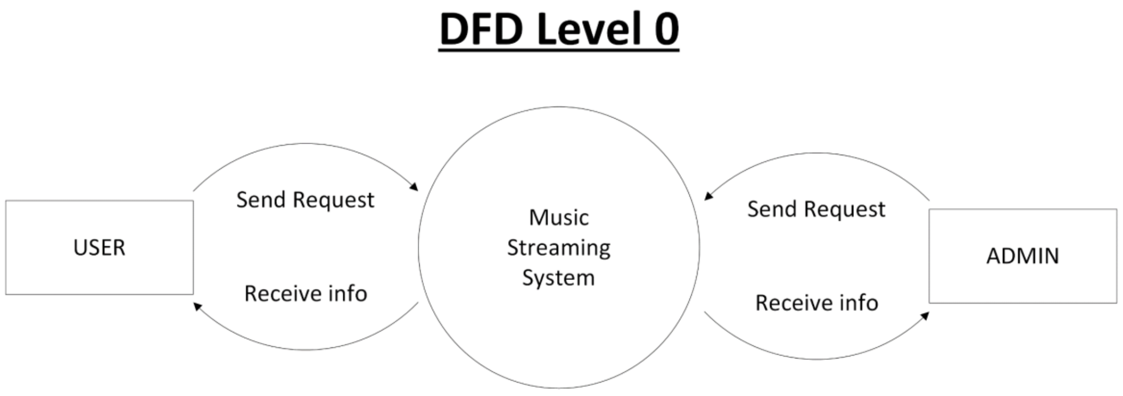
Table Schema for GENRES Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
NAME	VARCHAR(20)	NOT NULL

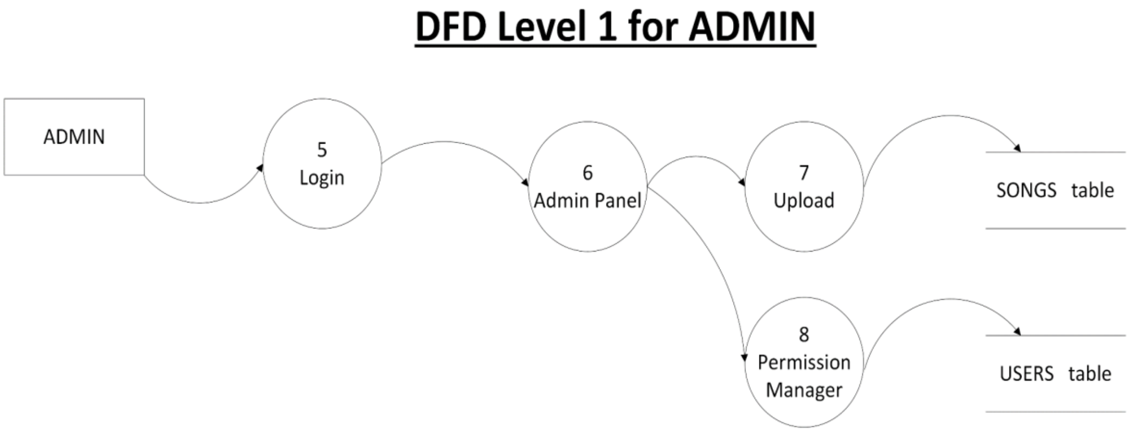
Table Schema for LANGUAGE Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
LANGUAGE	VARCHAR(20)	NOT NULL

Table Schema for PLAYLIST Table		
Field	**Type/Size**	**Constraint**
ID	INT	PRIMARY KEY
PLAY_SONG	INT	FORIEGN KEY
ORDER	INT	NOT NULL

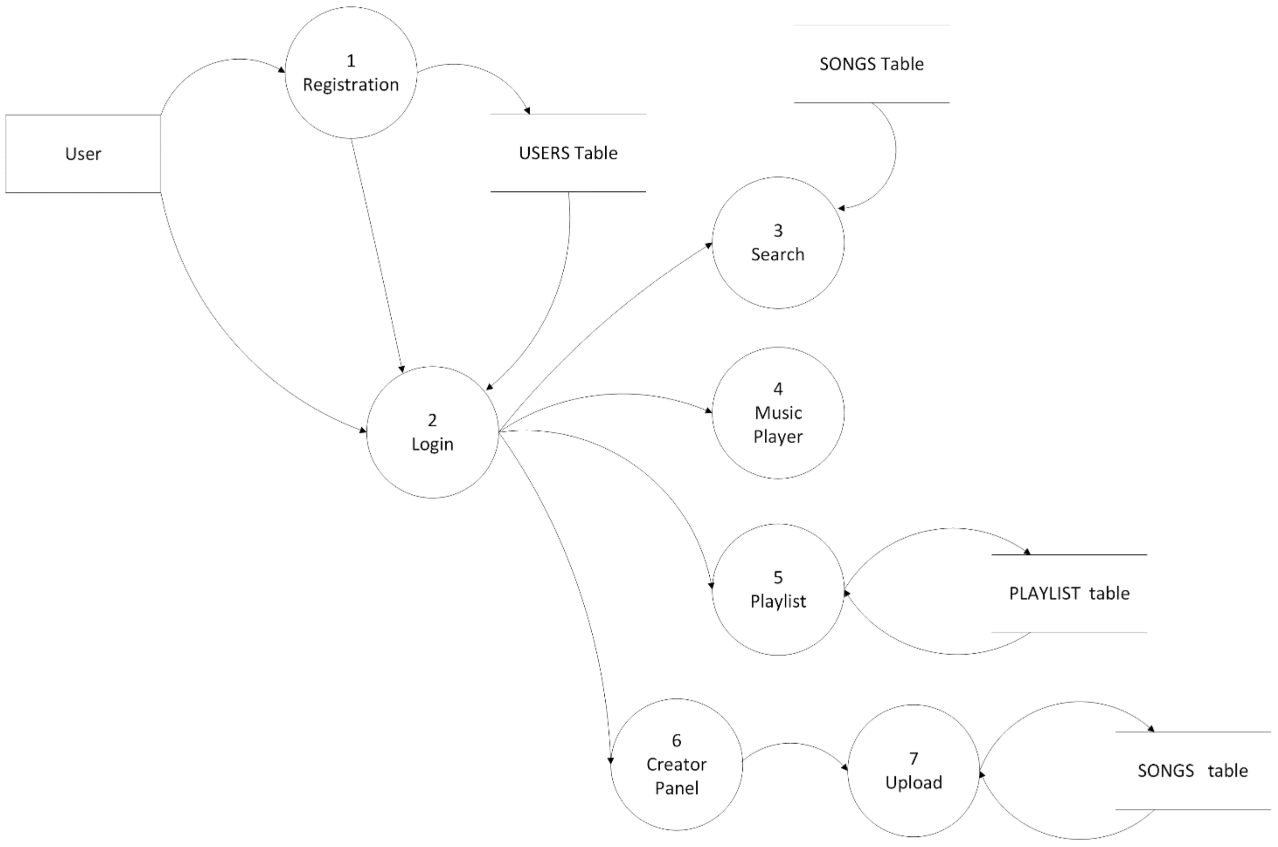
DESIGN TOOL

6.1 DATA FLOW DIAGRAM





DFD Level 1 for USER



SYSTEM TESTING

7.1 SOFTWARE TESTING

Software testing is the critical aspect of software quality assurance and represents the ultimate review of specification, design and coding. Testing is the process of running the program with the explicit intention of finding an error. Testing is vital to the success of the system. During testing the system is used experimentally to ensure that the software does not fail. That is, it will run according to its specifications and in the way user expects too. The system is tested with special test data and the results are examined for their validity.

7.2 CODE TESTING

The code testing strategy examines the logic of the program. To follow this method, the analyst develops test cases that result in executing every instruction in the program or module.

7.3 UNIT TESTING

Unit testing can be performed from the bottom up, starting with the smallest and lowest level modules and routines that are assembled and integrated to perform a specific function.

7.4 SYSTEM TESTING

It tests the integration of each module in the system. It also tests to find discrepancies between the system and its original objective. In this testing analysis we are trying to find areas where modules have been designed with different specification for data length, type etc.

7.5 VALIDATION TESTING

Validation succeeds when the software functions in which the user expects. Validation refers to the process of using software in a live environment in order to find errors. During the course of validating the system, failure may occur and sometimes the coding has to be changed according to the requirement.

7.6 OUTPUT TESTING

The system should produce the required output is on screen and in printed format. The output format on the screen is to be format that designed in the system design phase according to user needs. For the hard copy also, the output comes out as the specified requirements by the user.

7.7 ACCEPTANCE TESTING

User acceptance of a system is the factor for the success of any system. If the users of the system cannot digest the nature of the system, the project would fail**.**

- **Input screen testing**
- **Output screen testing**

Preparation of testing data plays a vital role in the system testing. After preparing the test data the system under study is tested using the test data. While testing the system, errors are again uncovered and corrected by using the above testing steps. Also, it is ideal to note the corrections for future use. The proposed system is tested and finds better results in all the above system tests.

SYSTEM IMPLEMENTATION

8.1 INTRODUCTION TO SYSTEM IMPLEMENTATION

A crucial phase in the system life cycle is the successful implementation of the new system design. Implementation simply means converting a new system design into operation. This involves creating computer compatible files, training, and telecommunication network before the system is up and running. A crucial factor in conversion is not disrupting the functioning of organization. Actual data were input into the program and the working of the system was closely monitored. It is a process of converting a new or revised system into an operational one. It is the essential stage in achieving a successful new system because usually it involves a lot of upheaval in the user. It must therefore be carefully planned and controlled to avoid problems.

The implementation phase involves the following tasks:

1. Careful planning.
2. Investigation
3. Design of methods
4. Training of the staff in the changeover phase.
5. Evaluation of changeover.

We implemented this new system in parallel run plan without making any disruptions to the ongoing system ,but only computerizing the whole system to make the work, evaluation and retrieval of data easier, faster and reliable.

8.2 IMPLEMENTATION PLAN

System implementation is the process of making the newly designed system fully operational and consistent in performance. After the initial design, the system is made published on the internet and the end user can do demonstration. The logical miss-working the system can be identified if any. Various combinations of test data were feed. Each process accuracy/reliability checking was made. After the approval, the system was implemented in the user department.The preparation of implementation of documentation process is often viewed as total sum of the software documentation process. In a well-defined software development environment, however the presentation of implementation documents is essentially an interactive process that synthesis and recognizes document items that were produced during the analysis and design phase for the presentation to user. The following are the three types of implementation documents.

- ❖ Conversion Guide
- ❖ User Guide
- ❖ Operation Guide

8.3 CONVERSION GUIDE

The conversion guide phase of the implementation, process the tasks that are required to place the system into an operation mode. They amplify the conversion lane that was defined during the internal design phase and defines file conversion, file creation and data entry requirements.

8.4 USER GUIDE

The system application and operation functions describes the overall performance capabilities of the system and define procedures the user must follow to operate the system. In the realm of information system, the content of a user guide must be developed to coincide with a criterion that defines the characteristics of one of the following methods of data processing

- ❖ Off-line processing
- ❖ Direct access processing

8.5 OPERATION GUIDE

The function of an operation is to define the control requirements of a system and provide instruction for initializing, running and terminating the system. The items contained in an operation guide may be grouped as follows.

- ❖ General information
- ❖ System overviews
- ❖ Run description

8.6 POST IMPLEMENTATION REVIEW

After the system is implemented and conversion is completed, a review of system is usually conducted by users and analyst this is called post implementation review.

- ♣ The most fundamental concern post implementation review is determining the system has met its objective that is analyst want to know if the performance level of the system has improved and if the system is producing the result intended.

- ♣ If neither is happening, one may question whether the system can be considered successful.

- ♣ By using current system, all the requirements of all users are fulfilled.

SYSTEM MAINTENANCE

9.1 TYPES OF MAINTENANCE

Maintenance is making adaptation of the software for external changes (requirements changes or enhancements) and internal changes (fixing bugs). When changes are made during the maintenance phase all preceding steps of the model must be revisited. There are three types of maintenance:

1. Corrective (Fixing Bugs / errors).
2. Adaptive (Updates due to environment changes).
3. Perfective (Enhancements, requirements changes.

SYSTEM EVALUVATION

10.1 SYSTEM EVALUVATION

System evaluation is the process of assessing the performance of a complete online telephone billing web application to discover how it is likely to perform in online billing system. Systems developers spend a proportion of their trading time trying to improve their existing systems and looking for new and different ideas. It is essential that any proposed system or change is fully evaluated before being used in live situation. System evaluation brings your complete system into play.

System evaluation brings your complete system into play – not just the rules. This is essential because the rules are only a part of your system and arguably they are not the most important part. Your money management techniques are likely to have a much bigger influence on your final results than any other aspect of your system

CONCLUSION

11.1 CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in PHP web based application and know some extent Windows Application and SQL Server, but also about all handling procedure related with LAMA Music. It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

Benefits:

- Simple.
- User friendly
- Easier handling and flexibility
- Use of new technology
- Data can be stored for longer period
- Addition, deletion, modification of records as when needed.
- Seamless page transfer
- Unique temporary playlist for each user in each visit

APPENDIX

12.1 SAMPLE CODE

```
<?php
    $songQuery = mysqli_query($con,"SELECT * FROM songs ORDER BY RAND() LIMIT 10");
    $resultArray = array();

    while($row = mysqli_fetch_assoc($songQuery))
    {
        array_push($resultArray,$row['id']);
    }

    $jsonArray = json_encode($resultArray);

?>
<script>


    $(document).ready(function(){
        newPlaylist = <?php echo $jsonArray ?>;
        audioElement = new Audio();
        console.log(newPlaylist[0]);
        setTrack(newPlaylist[0],newPlaylist,false);
        updateVolumeProgressBar(audioElement.audio);
        //progress bar movement control
```

```
$("#nowPlayingContainer").on("mousedown touchstart mousemove touchmove", function(e){
    e.preventDefault();
})



$(".playbackBar .progressBar").mousedown(function()
{
    mouseDown = true;
});
$(".playbackBar .progressBar").mousemove(function(e)
{
    if(mouseDown)
        {
            timeFromOffset(e,this);
        }

});
$(".playbackBar .progressBar").mouseup(function(e)
{
    timeFromOffset(e,this);

});
$(document).mouseup(function(){
    mouseDown = false;

});


//volume bar movement control

$(".volumeBar .progressBar").mousedown(function()

{

     mouseDown = true;

});

$(".volumeBar .progressBar").mousemove(function(e)

{

    if(mouseDown)

        {

            var percentage = e.offsetX / $(this).width();

            if(percentage >= 0 && percentage <=1)

                {

                    audioElement.audio.volume = percentage;

                }

        }


});

$(".volumeBar .progressBar").mouseup(function(e)

{

    var percentage = e.offsetX / $(this).width();

    if(percentage >= 0 && percentage <=1)
```

```
            {

                audioElement.audio.volume = percentage;

            }


    });

    $(document).mouseup(function(){

        mouseDown = false;

    });


});


function timeFromOffset(mouse,progressBar)

{

    var percentage =    mouse.offsetX / $(progressBar).width() * 100;

    var seconds = audioElement.audio.duration * (percentage/100);

    audioElement.setTime(seconds);

}

function prevSong()

{

    if(audioElement.audio.currentTime >= 3 || currentIndex == 0)

        {

            audioElement.setTime(0);

        }

    else
```

```
          {
              currentIndex--;
              setTrack(currentPlaylist[currentIndex],currentPlaylist,true);
          }
      }
      function nextSong()
      {
         if(repeat)
          {
              audioElement.setTime(0);
              playSong();
              return;
          }
         if(currentIndex == currentPlaylist.length - 1 )
         {
              currentIndex = 0;
         }
         else
         {
            currentIndex++;
         }
         var trackToPlay = shuffle? shufflePlaylist[currentIndex] : currentPlaylist[currentIndex];


         setTrack(trackToPlay,currentPlaylist,true);
```

```
}


function setRepeat()
{
   repeat = !repeat;
   var imageName =  repeat? "repeat-active.png":"repeat.png";
   $(".controlButton.repeat img").attr("src","assets/images/icons/"+imageName);
}



 function setMuted()
{
   audioElement.audio.muted = !audioElement.audio.muted;
   var imageName =  audioElement.audio.muted? "volume-mute.png":"volume.png";
   $(".controlButton.volume img").attr("src","assets/images/icons/"+imageName);
}
 function setShuffle()
{
   shuffle = !shuffle;
   var imageName =  shuffle? "shuffle-active.png":"shuffle.png";
   $(".controlButton.shuffle img").attr("src","assets/images/icons/"+imageName);

   if(shuffle)
       {
```

```
                shuffleArray(shufflePlaylist);

                currentIndex = shufflePlaylist.indexOf(audioElement.currentlyPlaying.id);

            }

            else

                {

                  currentIndex = currentPlaylist.indexOf(audioElement.currentlyPlaying.id);

                }

        }


        function shuffleArray(a) {

        var j, x, i;

        for (i = a.length - 1; i > 0; i--) {

            j = Math.floor(Math.random() * (i + 1));

            x = a[i];

            a[i] = a[j];

            a[j] = x;

        }

        return a;

    }




        function setTrack(trackId,newPlaylist,play)

        {

            if(currentPlaylist != newPlaylist)
            {
                currentPlaylist = newPlaylist;
                shufflePlaylist = currentPlaylist.slice();
                shuffleArray(shufflePlaylist);
            }
            if(shuffle)
            {
                currentIndex = shufflePlaylist.indexOf(trackId);
            }
            else
            {
                currentIndex = currentPlaylist.indexOf(trackId);
            }

            pauseSong();
            $.post("includes/handlers/ajax/getSongJson.php",{songId : trackId },function(data){



            var track = JSON.parse(data);
            $(".trackName span").text(track.title);

            $.post("includes/handlers/ajax/getArtistJson.php",{artistId : track.artist },function(data){
                 var artist = JSON.parse(data);
                $(".trackInfo .artistName span").text(artist.name);
```

```
            $(".trackInfo .artistName span").attr("onclick","openPage('artist.php?id="+artist.id+"')");

        });

        $.post("includes/handlers/ajax/getAlbumJson.php",{albumId : track.album },function(data){

             var album = JSON.parse(data);

            $(".content .albumLink img").attr("src",album.artworkPath);

            $(".content                                                              .albumLink
img").attr("onclick","openPage('album_page.php?id="+album.id+"')");

            $(".trackInfo                                                            .trackName
span").attr("onclick","openPage('album_page.php?id="+album.id+"')");


        });

        audioElement.setTrack(track);

        if(play)

            {

                playSong();

            }

    });


}


function playSong()

{

    if(audioElement.audio.currentTime == 0)

    {
            $.post("includes/handlers/ajax/updatePlays.php",{songId: audioElement.currentlyPlaying.id});

        }

        audioElement.play();

        $('.controlButton.play').hide();

        $('.controlButton.pause').show();

    }

    function pauseSong()

    {

        audioElement.pause();

        $('.controlButton.pause').hide();

        $('.controlButton.play').show();

    }

</script>



  <div id="nowPlayingContainer">

   <div id="nowPlayingBar">

     <div id="nowPlayingLeft">

        <div class="content">

            <span class="albumLink">

                <img role="link" tabindex="0" class = "albumArt" src="" class="albumArtwork" >

            </span>



            <div class="trackInfo">
```

```
                <span class="trackName">
                    <span role="link" tabindex="0"></span>
                </span>
                 <span class="artistName">
                    <span role="link" tabindex="0"></span>
                </span>
            </div>
        </div>
    </div>
    <div id="nowPlayingCenter">
        <div class="content playerControls">
            <div class="buttons">
                <button class="controlButton shuffle" title="Shuffle Button" onclick="setShuffle()">
                    <img src="assets/images/icons/shuffle.png" alt="shuffle">
                </button>
                <button class="controlButton previous" title="Previous Button" onclick="prevSong()">
                    <img src="assets/images/icons/previous.png" alt="previous">
                </button>
                <button class="controlButton play" title="Play Button" onclick = 'playSong()'>
                    <img src="assets/images/icons/play.png" alt="play">
                </button>
                <button class="controlButton pause" title="Pause Button" style="display:none;" onclick =
'pauseSong()'>
                    <img src="assets/images/icons/pause.png" alt="pause" >
                </button>
                <button class="controlButton next" title="Next Button" onclick="nextSong()">
                    <img src="assets/images/icons/next.png" alt="next">
                </button>
                <button class="controlButton repeat" title="Repeat Button" onclick="setRepeat()">
                    <img src="assets/images/icons/repeat.png" alt="repeat">
                </button>
            </div>
             <div class="playbackBar">
                    <span class="progressTime current">0:00</span>
                    <div class="progressBar">
                        <div class="progressbarBg">
                            <div class="progress"></div>
                        </div>
                    </div>
                    <span class="progressTime remaining">0:00</span>
                </div>
        </div>
    </div>
    <div id="nowPlayingRight">
        <div class="volumeBar">
            <button class="controlButton volume" title="Volume button" onclick="setMuted()">
                <img src="assets/images/icons/volume.png" alt="Volume" >
            </button>
```

```
                    <div class="progressBar">

                            <div class="progressbarBg">

                                <div class="progress"></div>

                            </div>

                    </div>

                </div>

            </div>


        </div>

    </div>


    var audioElement;

    var currentPlaylist = [];

    var shufflePlaylist = [];

    var tempPlaylist = [];

    var mouseDown = false;

    var currentIndex = 0;

    var repeat = false;

    var shuffle = false;

    var userLoggedIn;

    var timer;

    var backStack = [];

    var frontStack = [];

    var curUrl="index.php";

    var popFlag = false;
```

```
// making back button work


$(window).on('popstate', function(event) {
  var changePage = backStack.pop();
    frontStack.push(curUrl);
    popFlag = true;
    openPage(changePage);
});
//button work end



$(window).scroll(function(){
    hideOptionsMenu();
})
$(document).click(function(click){
    var target = $(click.target);
    if(!target.hasClass("item") && !target.hasClass("optionsButton"))
        {
            hideOptionsMenu();
        }
})
$(document).on("change","select.playlist",function(){
    var select = this;
    var playlistId = $(select).val();

    var songId = $(select).prev(".songId").val();

    console.log(songId);

    $.post("includes/handlers/ajax/addToPlaylist.php",{playlistId:playlistId,songId:songId})

    .done(function(error){

         if(error!="")

                    {

                        alert(error);

                        return;

                    }

       hideOptionsMenu();

        $(select).val("");


    });

})

function updateEmail(emailClass)

{

    var emailValue = $("."+emailClass).val();

    $.post("includes/handlers/ajax/updateEmail.php",{ email: emailValue, username:
userLoggedIn}).done(function(response){


        $("."+emailClass).nextAll(".message").text(response);


    });

}

function updatePassword(oldPasswordClass,newPasswordClass1,newPasswordClass2)
```

```
{
   var oldPassword = $("."+oldPasswordClass).val();
   var newPassword1 = $("."+newPasswordClass1).val();
   var newPassword2 = $("."+newPasswordClass2).val();


   $.post("includes/handlers/ajax/updatePassword.php",
         {
          oldPassword: oldPassword,
          newPassword1: newPassword1,
          newPassword2: newPassword2,
          username: userLoggedIn
          })
      .done(function(response){


      $("."+oldPasswordClass).nextAll(".message").text(response);


   });
}
function logout()
{
   $.post("includes/handlers/ajax/logout.php",function()
        {
      location.reload();
   });
}
```

```
function openPage(url)
{
    if(timer != null)
        {
            clearTimeout(timer);
        }
    if(url.indexOf("?") == -1)
        {
            url += "?";
        }
    var encodedUrl = encodeURI(url+"&userLoggedIn="+userLoggedIn);
    $("#mainContent").load(encodedUrl);
    $("body").scrollTop(0);

    if(!popFlag)
    {
        backStack.push(curUrl);
    }

    curUrl = url;
    console.log(backStack);
    history.pushState(null,null,url);
    popFlag = false;
}
```

```
function removeFromPlaylist(button,playlistId)
{
    var songId = $(button).prevAll(".songId").val();


      $.post("includes/handlers/ajax/removeFromPlaylist.php", { playlistId: playlistId, songId: songId })
      .done(function(error) {


                  if(error != "") {
                              alert(error);
                              return;
                  }


                  //do something when ajax returns
                  openPage("playlist.php?id=" + playlistId);
      });
}
function createPlaylist()
{
    var popUp = prompt("Enter playlist name:");


    if(popUp != null)
        {

$.post("includes/handlers/ajax/createPlaylist.php",{name:popUp,username:userLoggedIn}).done(funct
ion(error){
               if(error!="")
```

```
                {
                    alert(error);
                    return;
                }
            openPage("yourMusic.php");
          });
        }
}

function  deletePlaylist(playlistId)
{
    var deleteprompt = confirm("Are you sure you want to delete this playlist?");
    if(deleteprompt)
        {

$.post("includes/handlers/ajax/deletePlaylist.php",{playlistId:playlistId}).done(function(error){
            if(error!="")
                {
                    alert(error);
                    return;
                }
            openPage("yourMusic.php");
          });

        }
        else
```

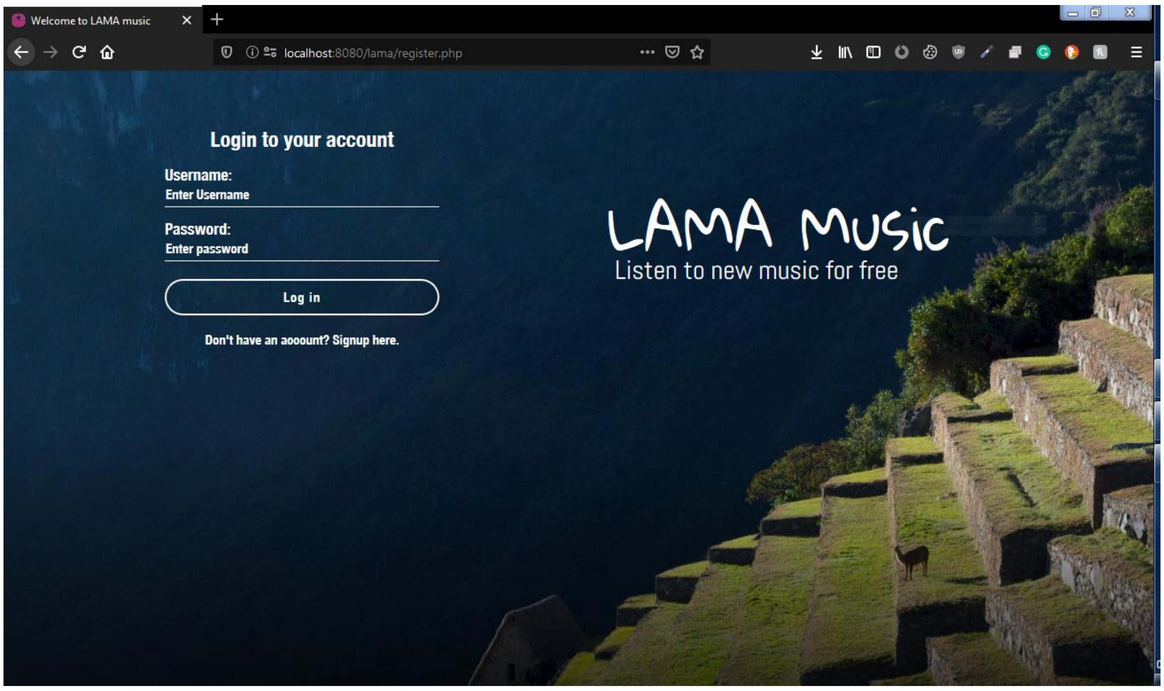
```
            {
                console.log("Dont");
            }
    }
    function hideOptionsMenu()
    {
        var menu = $(".optionMenu");
        if(menu.css("display") != "none")
            {
                menu.css("display","none");
            }
    }
    function showOptionsMenu(button)
    {
        var songId = $(button).prevAll(".songId").val();
        var menu = $(".optionMenu");
        var menuWidth = menu.width();

        menu.find(".songId").val(songId);

        var scrollTop = $(window).scrollTop(); //Distance from top of window to top of document
        var elementOffset = $(button).offset().top //Distance from top of the document to button

        var top = elementOffset - scrollTop;
        var left = $(button).position().left;
        menu.css({ "top":top+"px","left":left - menuWidth+"px","display":"inline",})


    }
    function formatTime(seconds)
    {
        var time = Math.round(seconds);
        var minutes = Math.floor(time/60);
        var seconds = time - (minutes*60);




        var extraZero = (seconds < 10 )? "0" : "";



        return minutes+":"+extraZero+seconds;
    }


    function updateTimeProgressBar(audio)
    {
        $(".progressTime.current").text(formatTime(audio.currentTime));


        var progress = audio.currentTime/audio.duration *100;
        $(".playbackBar .progress").css("width",progress+"%");
    }
```

```
function updateVolumeProgressBar(audio)
{
    var volume = audio.volume *100;
    $(".volumeBar .progress").css("width",volume+"%");
}

function playFirstSong()
{
    setTrack(tempPlaylist[0],tempPlaylist,true);
}
function Audio()
{
    this.currentlyPlaying;
    this.audio = document.createElement('audio');

    this.audio.addEventListener("canplay",function(){
        //'this' refers to 'audio' not 'Audio'
        $(".progressTime.remaining").text(formatTime(this.duration));
    });

    this.audio.addEventListener("timeupdate",function(){
        if(this.duration)
            {
                updateTimeProgressBar(this);
        }
    });

    this.audio.addEventListener("volumechange",function(){
        updateVolumeProgressBar(this);
    });

    this.audio.addEventListener("ended",function(){
        nextSong();
    })
    this.setTrack = function(track)
    {
        this.currentlyPlaying = track;
        this.audio.src = track.path;
    }
    this.play = function(){
        this.audio.play();
    }
    this.pause = function(){
        this.audio.pause();
    }
    this.setTime = function(seconds)
    {
        this.audio.currentTime = seconds;
    }
```

}

12.2 SCREENSHOT

User login page


Welcome to LAMA music
localhost:8080/lama/register.php
LAMA Music
Listen to new music for free

Login to your account

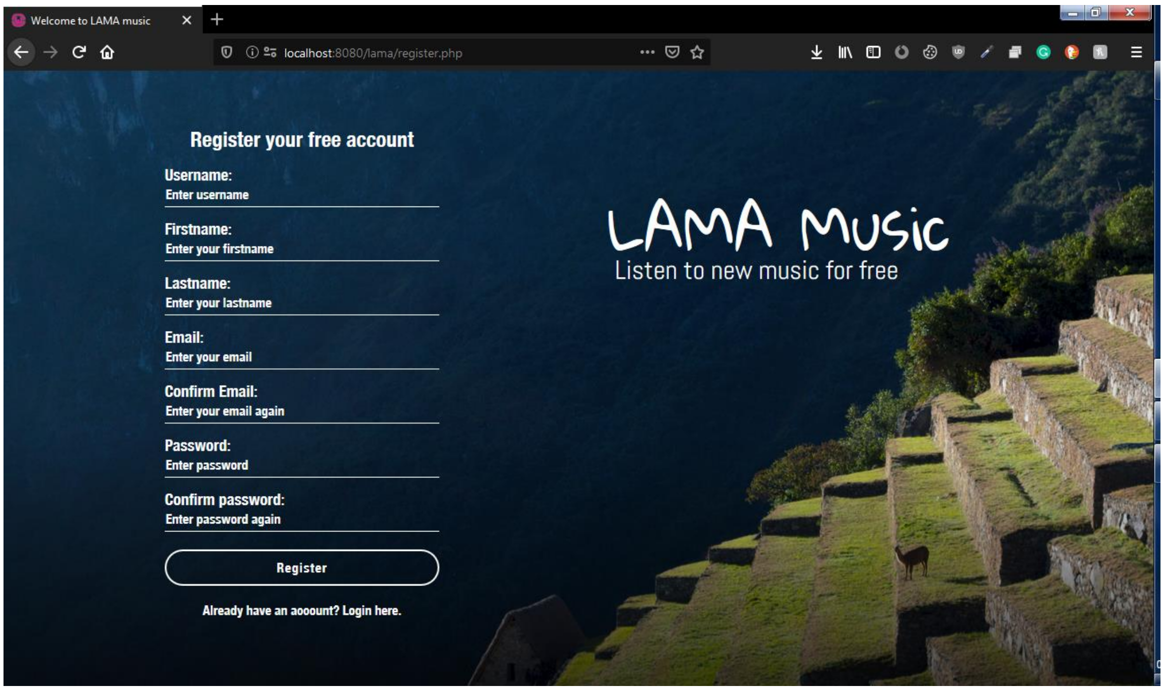
Username:
Enter Username

Password:
Enter password

Log in

Don't have an aooount? Signup here.

User Register page


Welcome to LAMA music
localhost:8080/lama/register.php
LAMA Music
Listen to new music for free

Register your free account

Username:
Enter username

Firstname:
Enter your firstname

Lastname:
Enter your lastname

Email:
Enter your email

Confirm Email:
Enter your email again

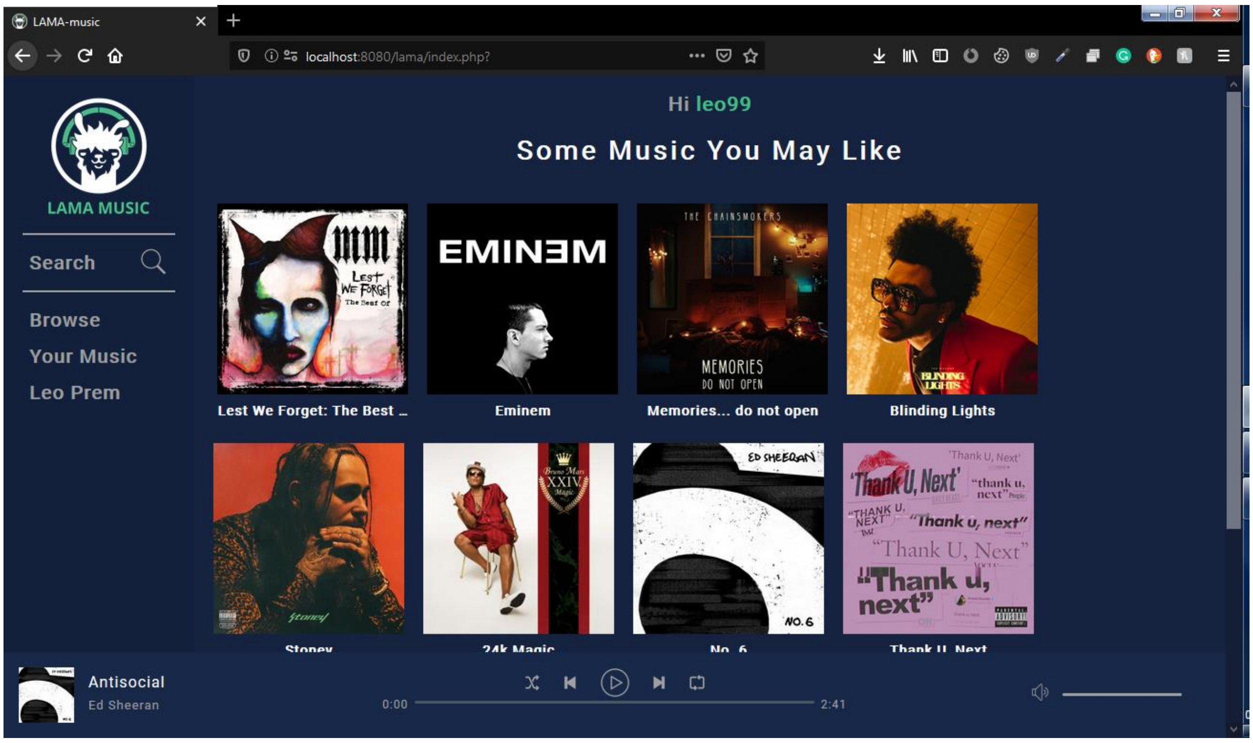
Password:
Enter password

Confirm password:
Enter password again

Register

Already have an aooount? Login here.

User home page



Hi leo99

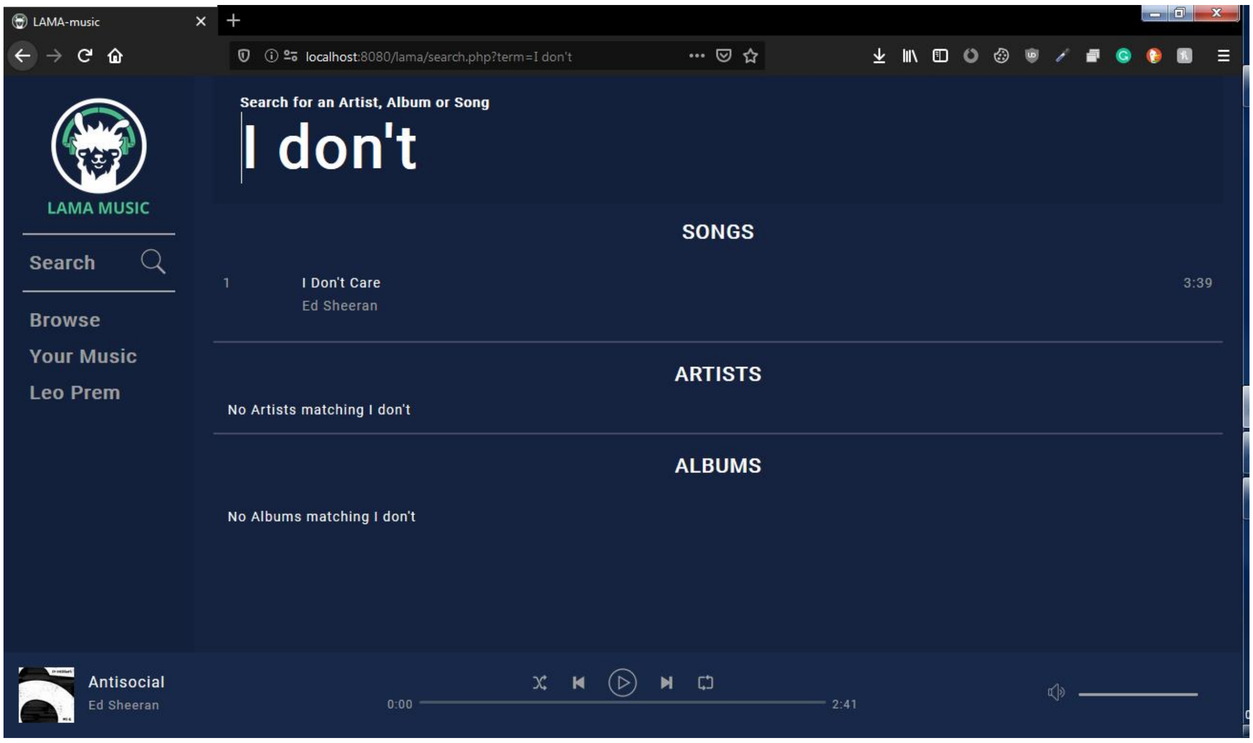
Some Music You May Like

Lest We Forget: The Best ... | Eminem | Memories... do not open | Blinding Lights

Stoney | 24k Magic | No. 6 | Thank U, Next

Antisocial
Ed Sheeran
0:00 2:41

Search page



Search for an Artist, Album or Song

I don't

SONGS

1 I Don't Care
Ed Sheeran 3:39

ARTISTS

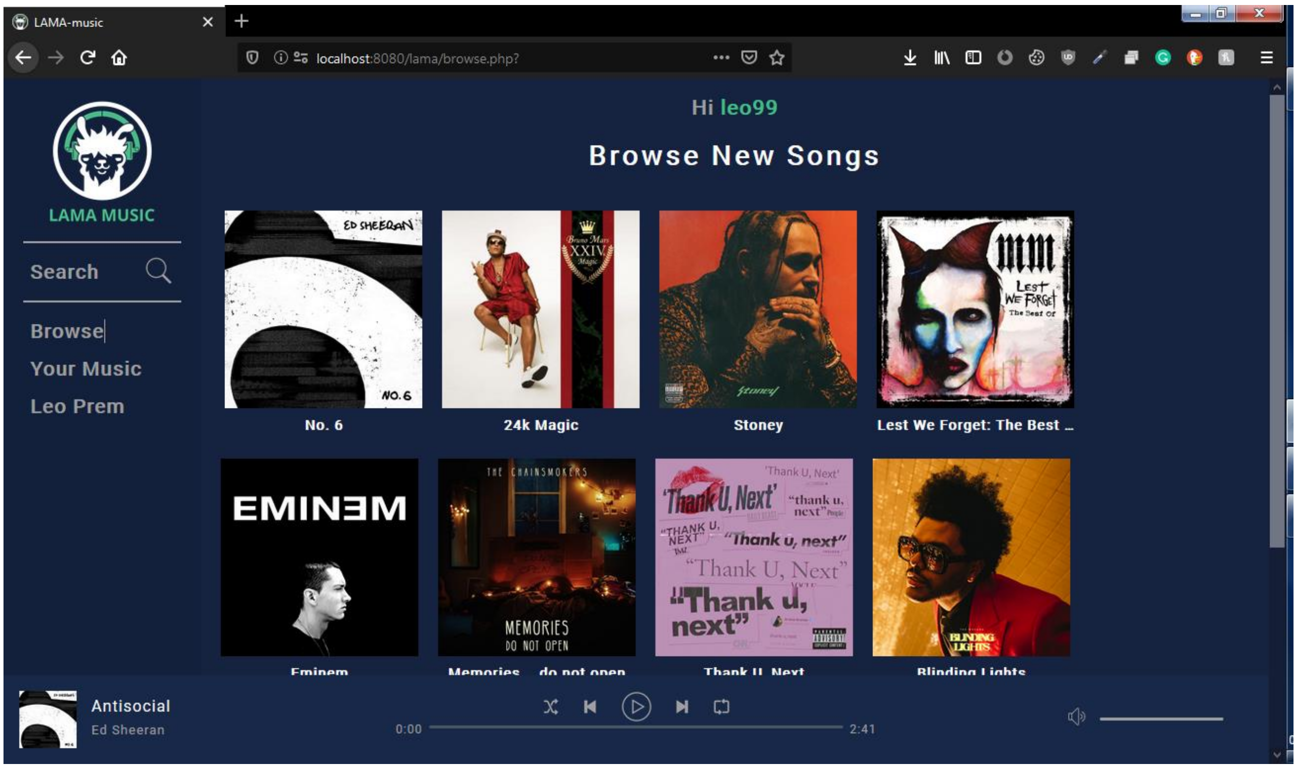
No Artists matching I don't

ALBUMS

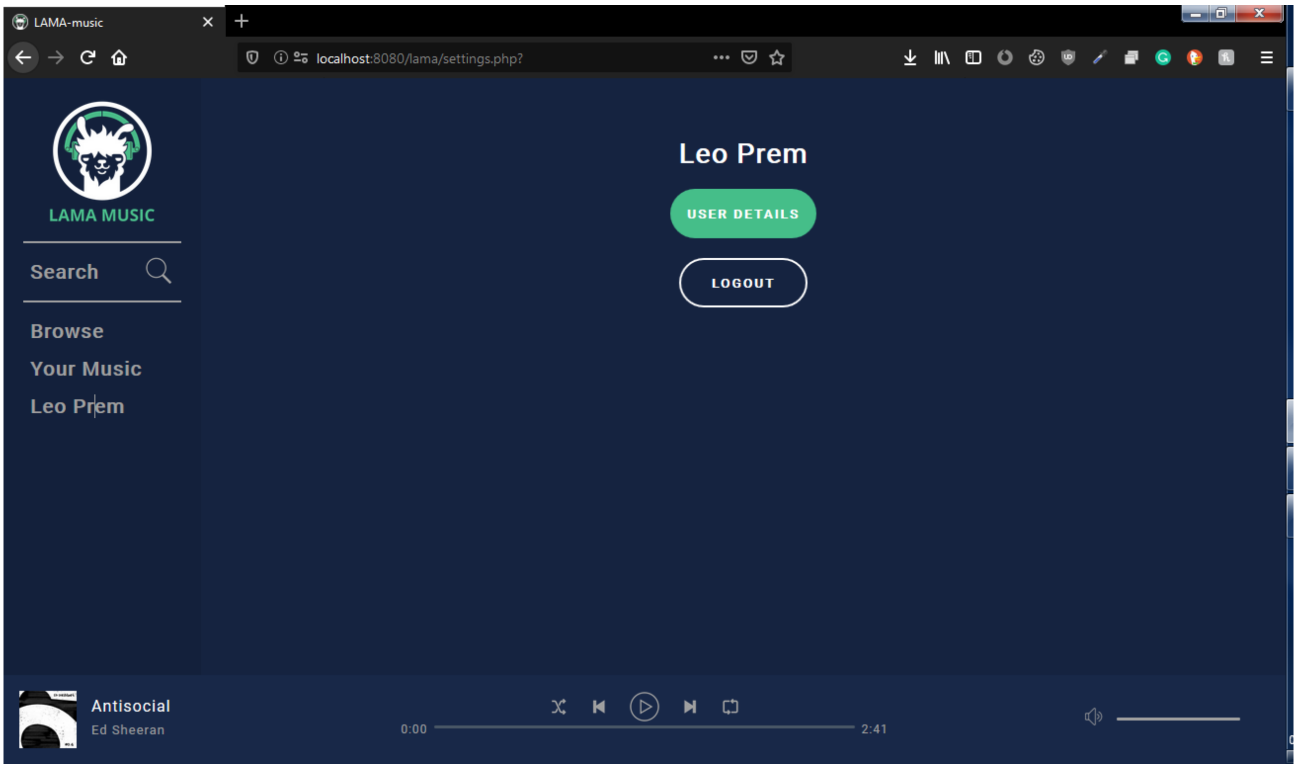
No Albums matching I don't

Antisocial
Ed Sheeran
0:00 2:41

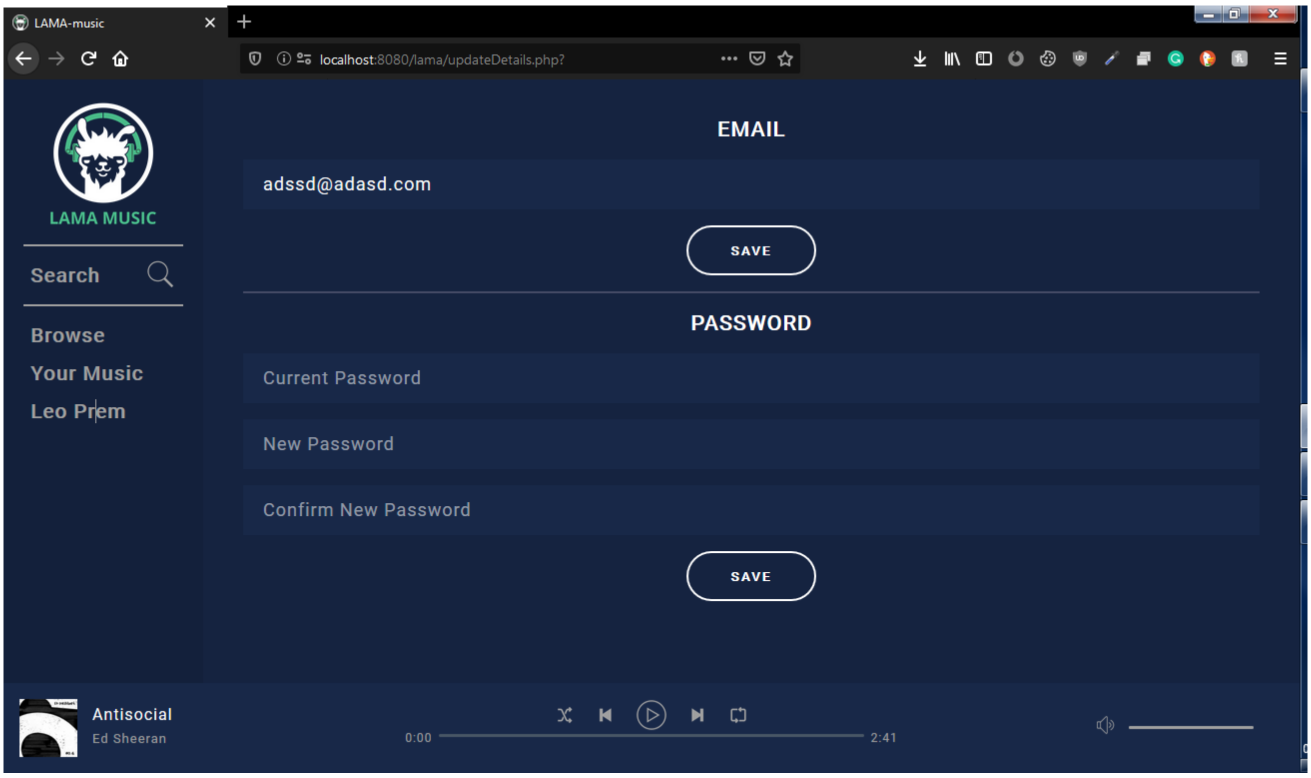
Browse page



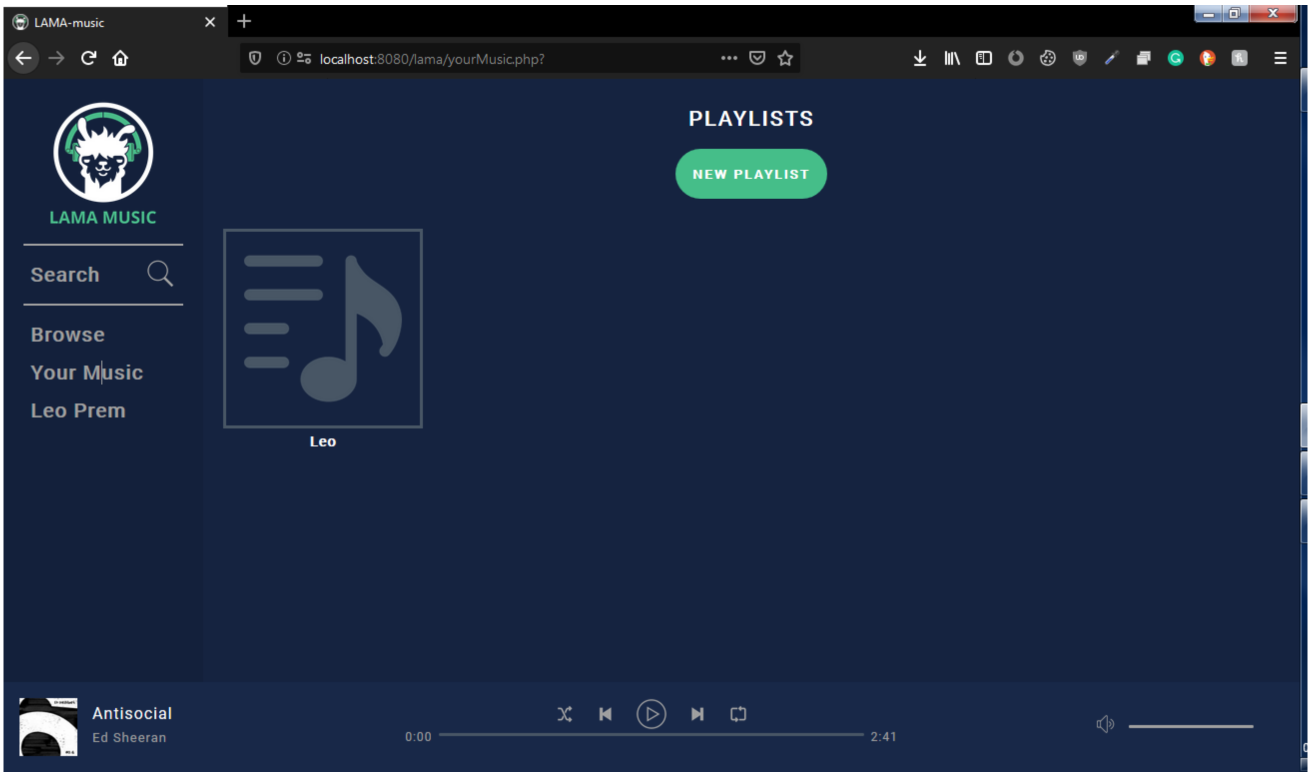
Account page



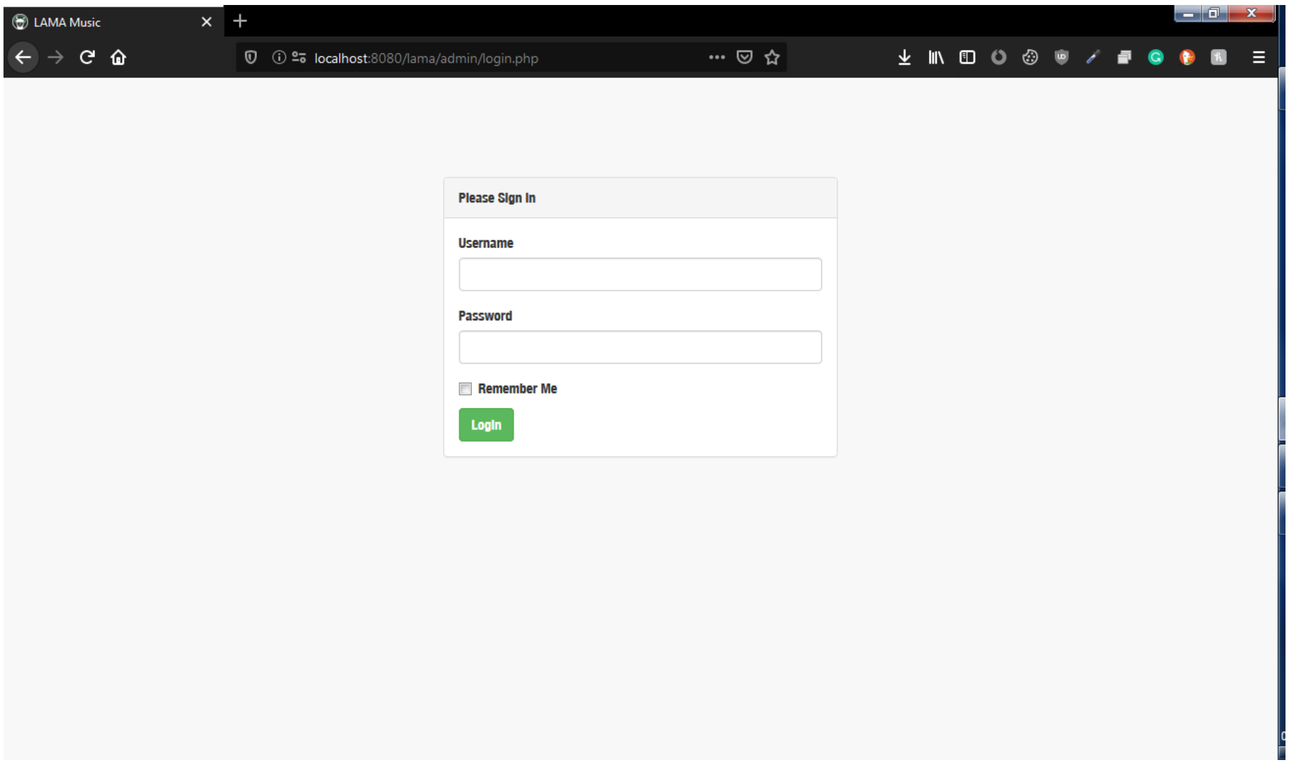
Details page



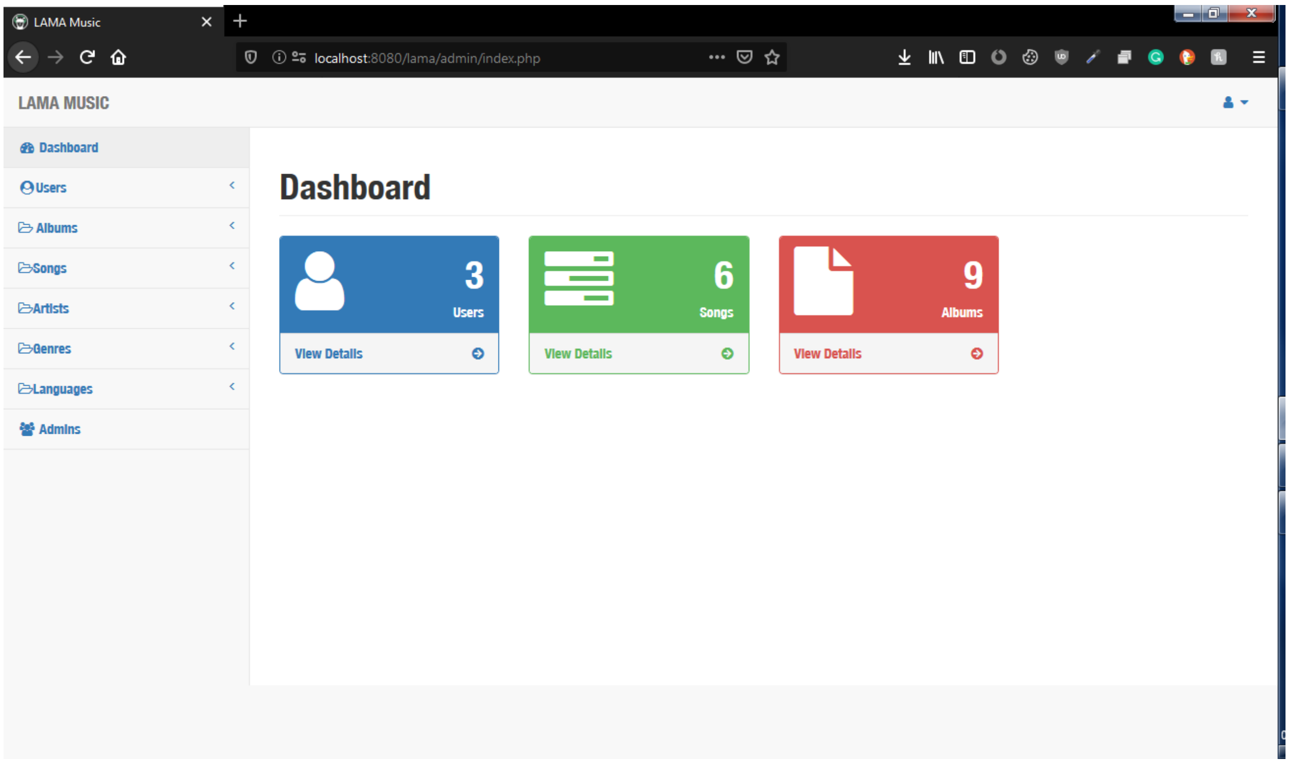
Playlist page



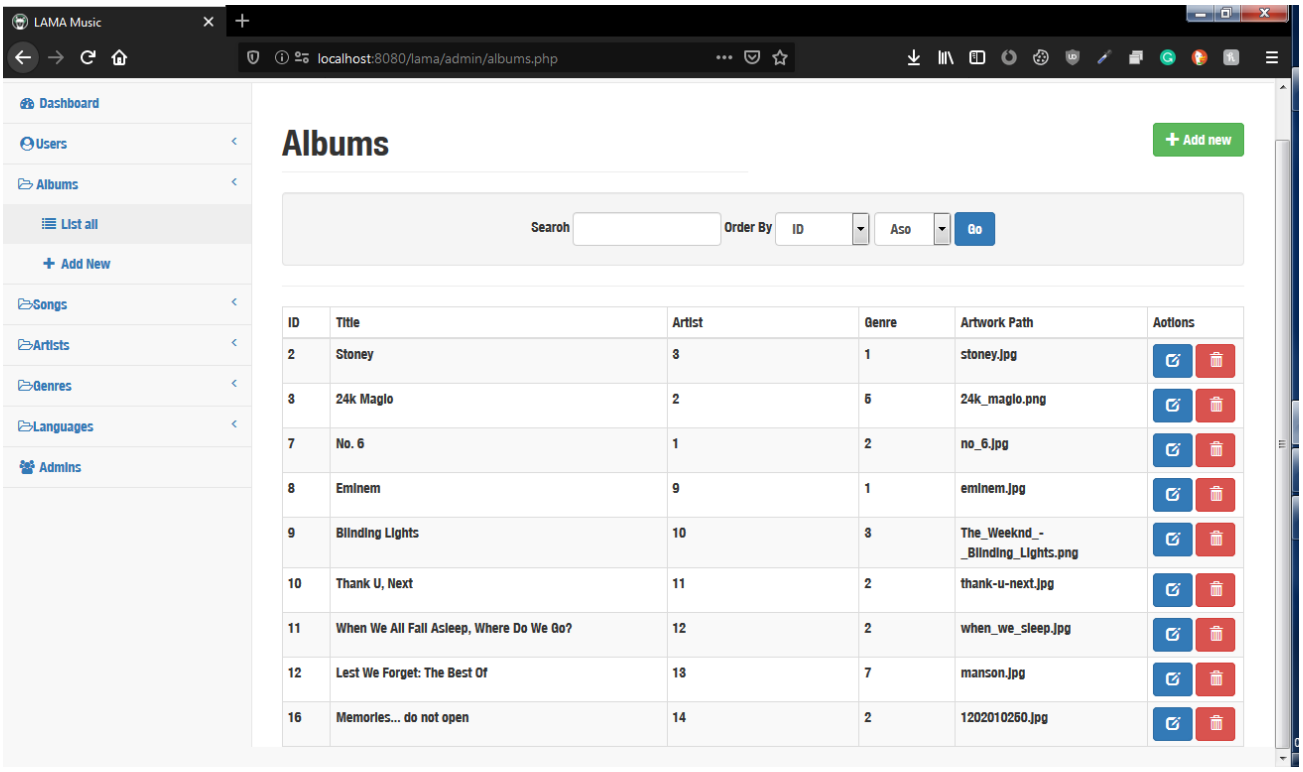
Admin login page



Admin home page



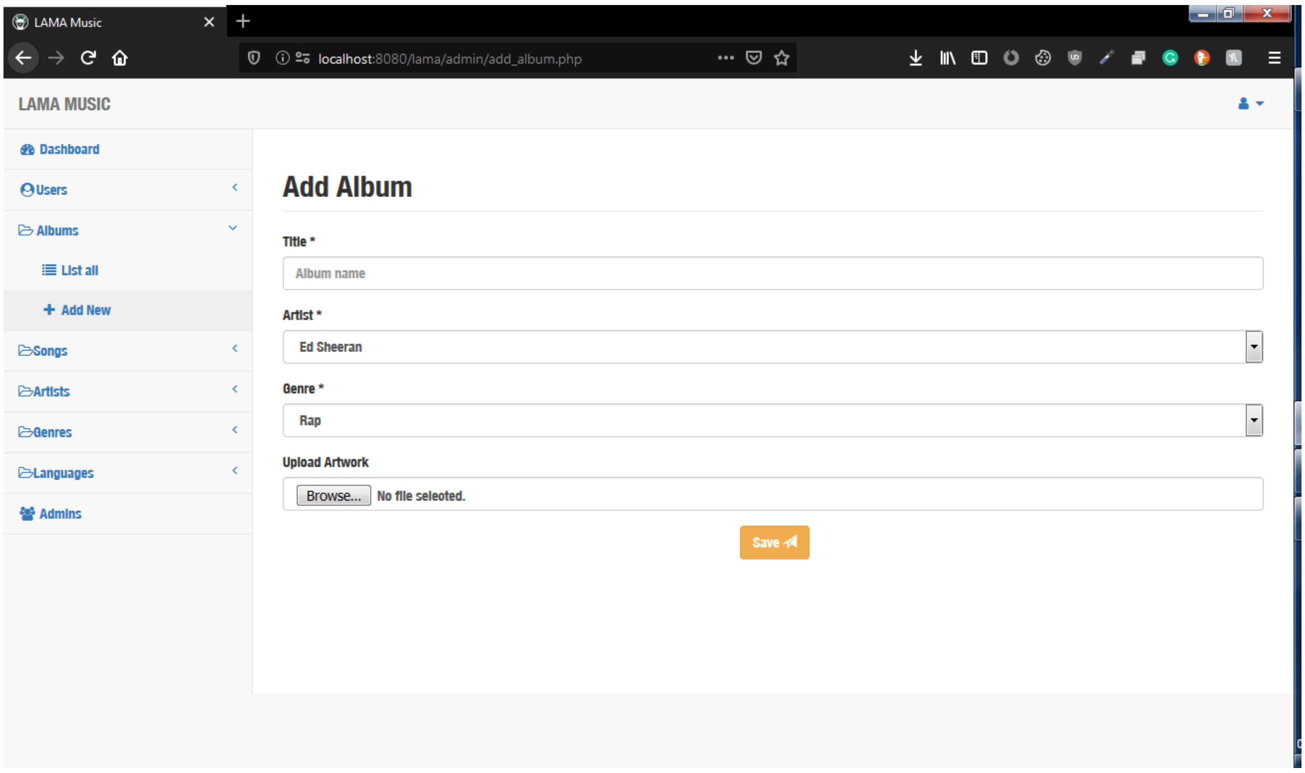
Album list page



Albums

ID	Title	Artist	Genre	Artwork Path	Actions
2	Stoney	3	1	stoney.jpg	
3	24k Magic	2	5	24k_magic.png	
7	No. 6	1	2	no_6.jpg	
8	Eminem	9	1	eminem.jpg	
9	Blinding Lights	10	3	The_Weeknd_-_Blinding_Lights.png	
10	Thank U, Next	11	2	thank-u-next.jpg	
11	When We All Fall Asleep, Where Do We Go?	12	2	when_we_sleep.jpg	
12	Lest We Forget: The Best Of	13	7	manson.jpg	
16	Memories... do not open	14	2	1202010260.jpg	

Add album page



Add Album

Title *
Album name

Artist *
Ed Sheeran

Genre *
Rap

Upload Artwork
No file selected.

BIBLIOGRAPHY

REFERENCES

- PHP: The Complete Reference, Steven Holzner, 2007.
- SQL Server 7: The complete reference, Gayle Coffman, 1990.
- Ajax: the complete Reference, Thomas Powell

WEBSITES

- https://www.w3schools.com/php/default.asp
- http://www.wikipedia.org
- https://www.tutorialspoint.com/ajax/what_is_ajax.htm
- https://stackoverflow.com/
- https://github.com/
- https://www.tutorialspoint.com/
- https://www.youtube.com/
- https://duckduckgo.com/