www.HEYDJPLAYMYSONG.com

CS-670-A Research Project Seminar

Sacred Heart University

5151 Park Avenue, Fairfield, CT 06825-1000

Friday the 6th, December 2013

Ali B. Kaba

Master of Science in Computer Information Science (MSCIS)

Information Technology Track

Francesco Sardo

Table of Contents

1 Abstract....6

2 Introduction....6

2.1 Summary of Problems, Opportunities and Directives....6

2.2 Scope....7

2.3 Intended Audience....7

2.4 Computer Applications Used....7

2.5 Web Applications Used....7

2.6 Languages Used....7

2.7 Plugins Used....8

3 Analysis Overview....8

3.1 System Usage....8

3.2 Assumptions, Dependencies and Constraints....8

3.3 Development Methods....9

4 Requirements....9

4.1 Inputs – Data....9

4.2 Outputs – Information....9

4.3 Processes – Manual/Automatic....9

4.4 Storage – Database....10

4.5 Control – Interfaces....10

4.6 Ease of use....10

4.7 Timelines and deadlines....10

4.8 Training....10

4.9 Quality Management....10

4.10 Security and Audits....10

4.11 Use Cases....11

4.12 Assess Project Worth in Terms of Cost vs. Value....11

4.12.1 Estimated Costs....11

4.12.2 Value....11

4.13 Preliminary Project Plan....11

4.13.1 Master Schedule....11

4.13.2 Resource Assignment....12

5 Design....13

5.1 Use Case Diagram....13
5.2 Context Data Flow Diagram....15
5.3 Entity Relationship Diagram....16
5.4 Class Diagram....16
5.4.1 Database Stored Procedures....17
5.5 Data Structured Diagram....22
5.5.1 Register....22
5.5.2 Search....23
5.5.3 Activate account....23
5.5.4 Reset passcode....24
5.5.5 Reset password....24
5.5.6 Change password....25
5.5.7 Login....25
5.5.8 Manage playlist....26
5.5.9 Create playlist....26
5.5.10 Logout....27
5.5.11 Upload playlist....27
5.6 Dataflow Diagram....28
5.7 Activity Diagram....28
5.8 Sequence Diagram....28
5.8 Conceptual Website Diagram....28
5.0 Screenshots....29
5.1 index.php....29
5.2 dashboard.php....29
5.3 manageplaylist.php....30
5.4 createplaylist.php....30
5.5 uploadplaylist.php....31
5.6 activate.php....31
5.7 changepassword.php....32
5.8 resetpassword.php....32
5.9 forgotpassword.php....33
5.10 music.php....33
6 English Instruction Manual....34

6.1 Using Hey DJ Play My Song 34
6.2 Getting Started 34
6.2.1 Getting/creating the CSV document 34
6.2.2 Getting the music 34
6.2.3 Formatting the CSV for HDJPMS system 35
6.3 Setting up an account 36
6.3.1 Visiting HeyDJPlayMySong.com 36
6.3.2 Creating an account 37
6.3.3 Verifying your account 37
6.4 Dashboard 39
6.4.1 Change password 39
6.4.2 Manage playlist 40
6.4.3 Create playlist 41
6.5 Search 42
6.5.1 Music 43
6.5.2 Voting 44
6.6 Homepage 45
7.0 Enhancements 46
8.0 Bibliography 46
8.1 Books 46
8.2 Internet Sites 46
9.0 Conclusion 47

This page was purposely left blank

1 Abstract

Hey DJ Play My Song (HDJPMS) is a music voting system web application. The intended audience of this product is for working professional Disc Jockeys (DJs) however it can be used by friends and family or even someone who plans to hire a DJ. It is a completely separate system from the actual device that is used to use to play the song.

Join the fun, place your playlist online for your friends and family to vote on! Where ever they are!

2 Introduction

2.1 Summary of Problems, Opportunities and Directives

HDJPMS's goal is to do the same outcome task as other music voting system on the market and it is as followed:

- Ability to create a playlist with favorite songs.
- Ability to search a song and request it to be played.
- Ability to vote up or down on an already requested song.

However some of the current systems require the following before voting:

- A setup fee.
- Personal information such as name, address.
- Ownership of the songs which can be a legal issue.
- Direct connection to the music system being used to play the song which is a potential threat.

The risk assessment here is that with guest connected to the music systems with users which can be a potential cause the following depending on the value of the playlist to the DJ/host:

- Negligible.
- Slight loss of competitive advantage.
- Significant loss of competitive advantage.
- Significant financial loss.
- Significant business profit.
- Serious loss of life.

The opportunity HDJPMS tackles is as followed:

- Free to use.
- Complete anonymous voting system.
- Fast registration with email for verification only, no name or address needed.
- HDJPMS is an entity of its own, doesn't connect directly to music system.
- This potential works well with legacy systems or closed systems.
- Peace of mind from malicious software.

With the user creating their own playlist, whether they are a DJ or hiring one for an event, this application can be used as a music database which can later on be used during the event for guest to vote on.

Potential directives would be to partner with a paid music record pool and be offered as an incentive for being part of that record pool.

2.2 Scope

This document contains a detail description of HDJPMS. This system is includes the integration of PHP, JavaScript, jQuery Mobile on the front end and MySQL on the backend. This document does not describe in detail the 3rd party technologies being used to implement the system.

2.3 Intended Audience

The audience for this document is anyone looking to understand this project and how it is designed.

2.4 Computer Applications Used

- Agent Ransack
- Apple iTunes
- Fiddler2
- FileZilla
- Gadwin PrintScreen
- Google Chrome Version 31.0.1650.57
- Internet Explorer 11
- jZip
- Microsoft Excel
- Microsoft PowerPoint
- Microsoft Visio
- Microsoft Word
- MySQL Workbench 6.0 CE
- Notepad++
- PuTTY

2.5 Web Applications Used

- CPanel
- Dropbox
- GiTHub
- JSFiddle
- JustHost.com

2.6 Languages Used

- Ajax
- Coding language used solutions
- CSS3
- HTML 5
- JavaScript
- PHP 5.4

2.7 Plugins Used

- Bootstrap
- Joyride (ZURB University)
- jQuery
- jQuery Mobile
- Uploadifive
- Voting (Dynamic Voting System with JQuery, PHP and MySQL)

3 Analysis Overview

3.1 System Usage

For a DJ or host to use this application, they first must get a comma-separated value (CSV) document, also freely available on www.heygodjplaymysong.com/about.php. Once the document is downloaded, they must register & activate their account. Once activated, they must create a playlist name and upload the CSV document onto the system. Once that is complete, they must announce to the guests on about the playlist name.

For a guest to use this application, they must first have a playlist name provided to them by the host. With the playlist name, they must visit www.heygodjplaymysong.com and type in the passcode in the search field. Once it is entered, they must hit search to be presented with two list:

- Top 20 Requested Songs
    - Ability to vote up or down songs.
- All Songs
    - Ability to search for songs and vote them up or down.

3.2 Assumptions, Dependencies and Constraints

This system is composed of many separate distributed components. The website is hosted with www.JustHost.com and therefore the up-time of the system is on their end. I decided to use PHP because it was the only language that would work with MySQL which was already available. I was also required to use Ajax as a solution of updating MySQL for the voting to be updated live I decided to use jQuery and Bootstrap because they were made designed for mobile devices.

I am limited to what my host offers however as far as jQuery and Bootstrap; I can make some advancement since they are built on JavaScript. Uploadifive is an upload plugin that was very limited on customization as well as Joyride which was a plugin build on top of jQuery which is also built on top of JavaScript.

More importantly the website is not using a Secure Sockets Layer (SSL) therefore it is possible to exploit the system and perform multiple votes. My preventions cannot secure users from a session hijacking however due to the nature of the system, I do not keep personal information except email.

3.3 Development Methods

The system is developed on a custom machine:

Machine	
OS Name	Microsoft Windows 7 Professional
Version	6.1.7601 Service Pack 1 Build 7601
OS Manufacturer	Microsoft Corporation
System Manufacturer	INTEL_
System Model	DZ68BC__
System Type	x64-based PC
Processor	Intel(R) Core(TM) i7-2600K CPU @ 3.40GHz, 3401 Mhz, 4 Core(s), 8 Logical Processor(s)
BIOS Version/Date	Intel Corp. BCZ6810H.86A.0021.2011.0831.1555, 8/31/2011
SMBIOS Version	2.6
Total Physical Memory	16.0 GB
Total Virtual Memory	32.0 GB
C Drive	238 GB

The software I used to create the database is MySQL Workbench 6.0 CE, and the software I used to write the the webpages is Notepad++. I also used some web applications to test some coding in the applications (see 2.5).

4 Requirements

4.1 Inputs – Data.

User:

- Enter playlist.
- Register.
- Upload playlist.
- Reset playlist.
- Delete playlist
- Reset song.
- Delete song.

4.2 Outputs – Information.

- Playlist.

4.3 Processes – Manual/Automatic.

Manual:

- User search song.
- User builds playlist file.
- User uploads playlist file.

Automatic:

- Playlist is collected from Database.
- User inputs are subsequently modifying database.

4.4 Storage – Database

- All data will be stored in MySQL database.

4.5 Control – Interfaces

- Administrator can modify playlist and its songs.
- User can vote anonymously without registering via the homepage

4.6 Ease of use

- Documentation will be downloadable from the website
- Tour is implemented on the website for ease of understanding.

4.7 Timelines and deadlines

The design and development of HDJPMS with the scope currently presented would be 4 months or less.

	Iteration 1	
Phase	Start Date	End Date
Inception	01/11/13	02/22/13
	Iteration 2	
Inception	02/22/13	03/15/13
Elaboration	03/15/13	05/25/13
	Iteration 3	
Inception	05/25/13	06/01/13
Elaboration	06/01/13	07/01/13
Construction	07/01/13	11/14/13
	Iteration 4	
Construction	11/14/13	11/30/13
Transition	11/30/13	11/30/13

4.8 Training

- Documentation is available on the website for training.
- Tour is also available for basic understanding.

4.9 Quality Management

- Data is user derived therefore training will be as details as possible.

4.10 Security and Audits

- System has not direct interaction with music system.
- Website doesn’t used HTTPS/SSL however it uses sessions.

- Certain pages include a session page which checks if it is a registered user.

4.11 Use Cases

Use Case Name	
Description	
Actors	
Pre-Conditions	
Basic Flow	
Post Conditions	
Alternate Flows	
Notes	

4.12 Assess Project Worth in Terms of Cost vs. Value

4.12.1 Estimated Costs

Operational Cost = $13,000.00

- Computer hardware & software = $2,100.00.
  - Desktops, laptops, keyboards, software used to develop or maintain system.
- Hosting = $400.00 (3 years)
  - Database, website at external location.
- Books = $500.00
- Others = $10,000.00
  - Includes utilities, rent, travel, etc.

Personnel Cost = $0.00

- Myself = $0.00
  - Programmer, Database Administrator, System Analyst.

Total Development Cost Estimate: $13,000.00

4.12.2 Value

This application value is the ability is to allow a host to present a music selection to their company in a voting system which can be used with closed systems.

- Hey DJ Play My Song is a free web application.
- Users can save time with this application.

4.13 Preliminary Project Plan

4.13.1 Master Schedule

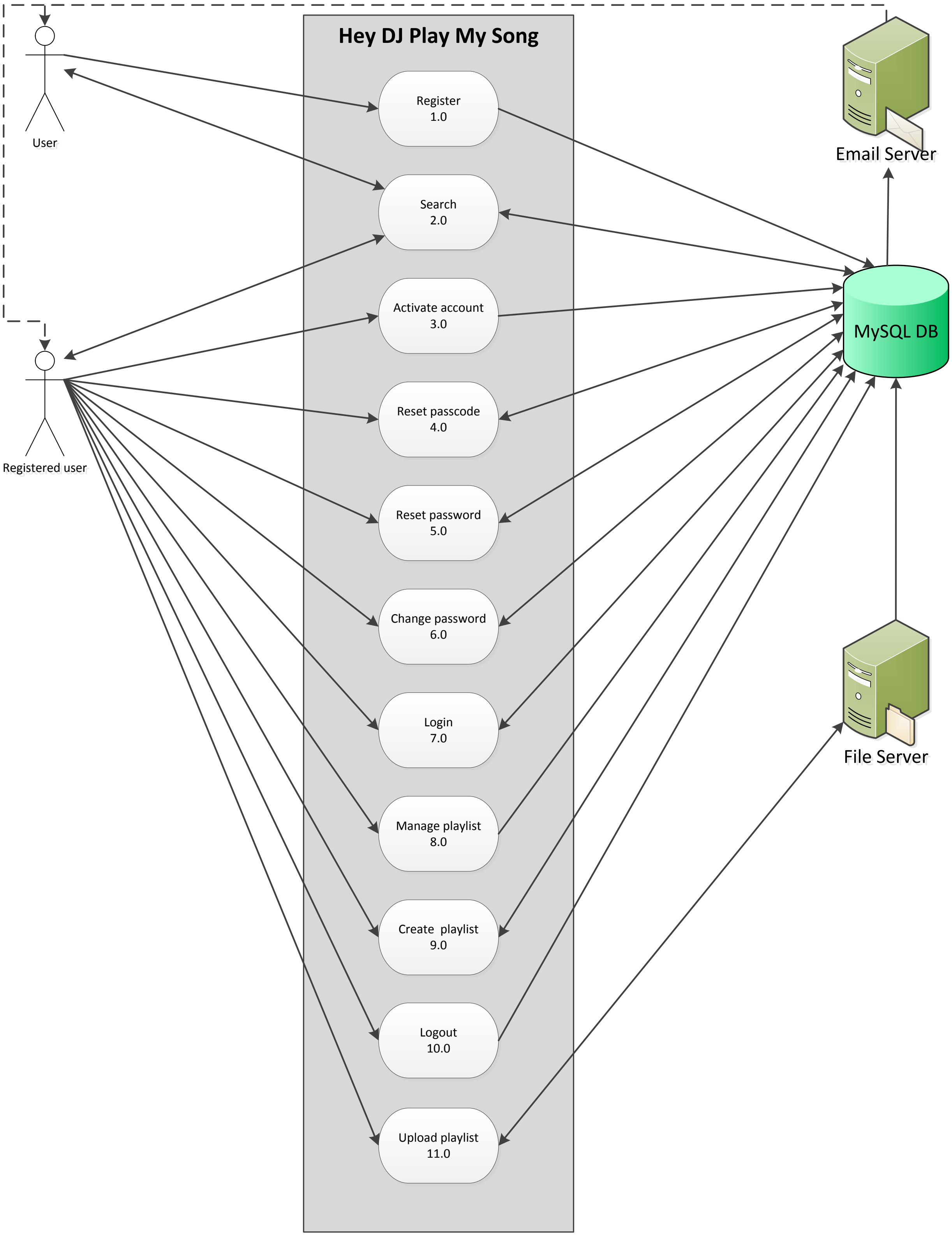
This system will consists of approximately ~2,000 lines of codes, the effort and schedule required to complete the system is estimated to be attainable in one year.

4.13.2 Resource Assignment

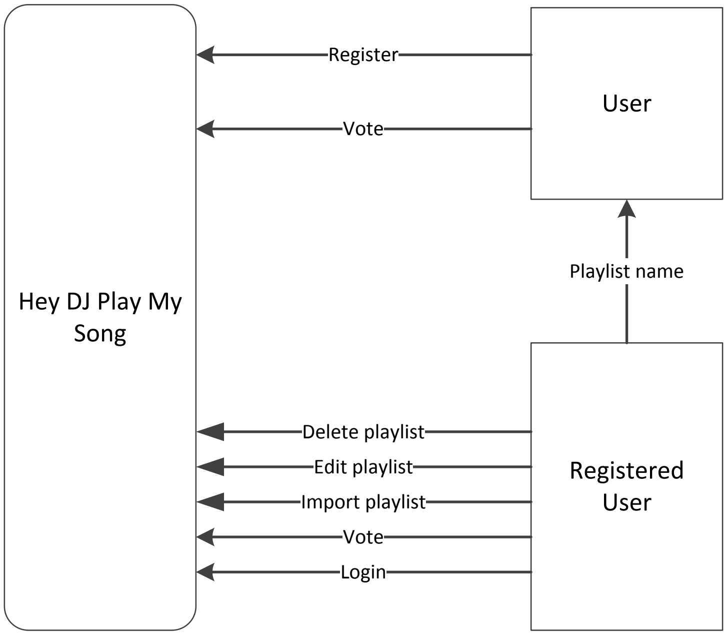
- **Personnel:**
  - 1 Programmer/DBA/System Analysts.

5 Design

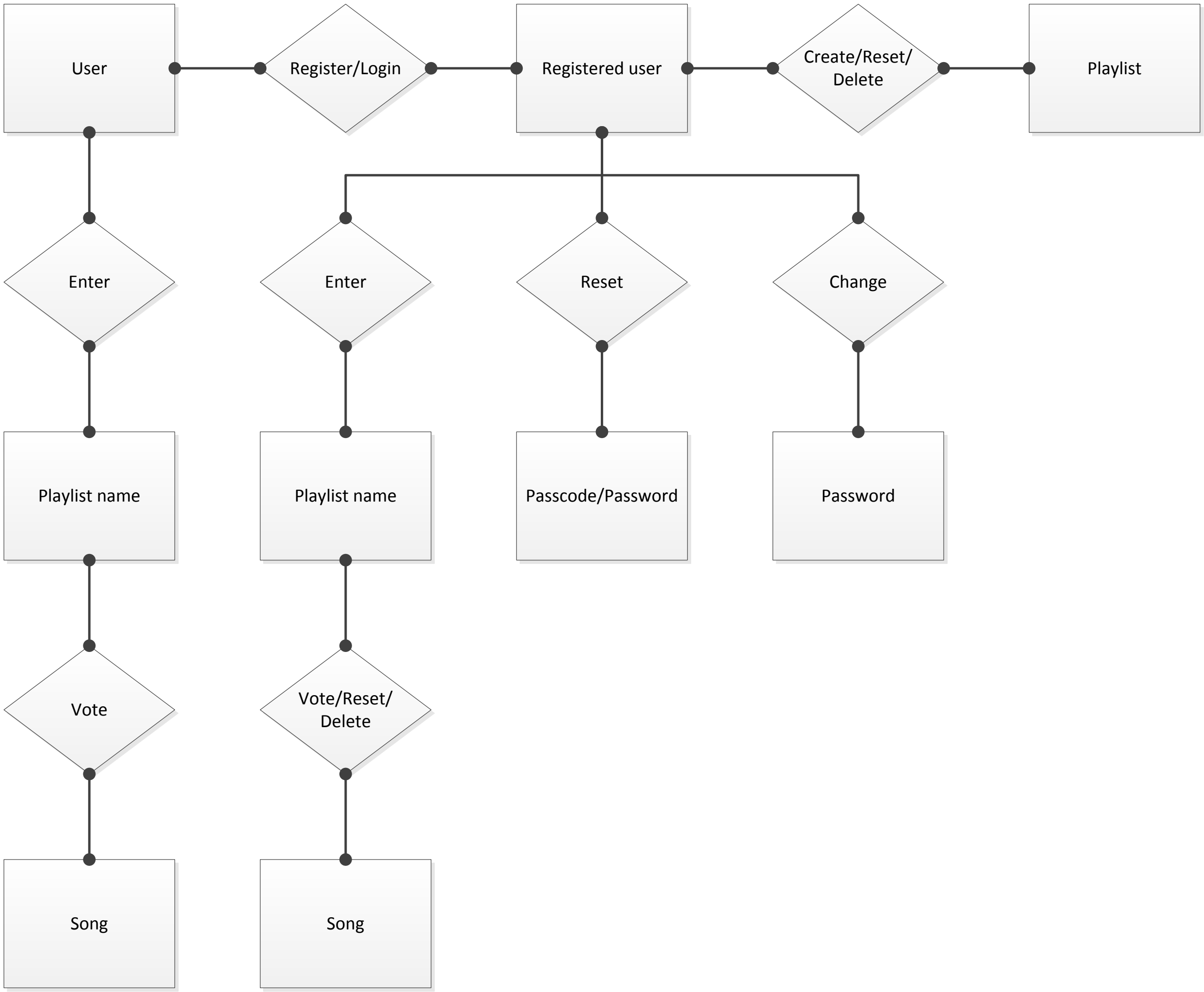
5.1 Use Case Diagram


Hey DJ Play My Song
User
Registered user
Register
1.0
Search
2.0
Activate account
3.0
Reset passcode
4.0
Reset password
5.0
Change password
6.0
Login
7.0
Manage playlist
8.0
Create playlist
9.0
Logout
10.0
Upload playlist
11.0
Email Server
MySQL DB
File Server

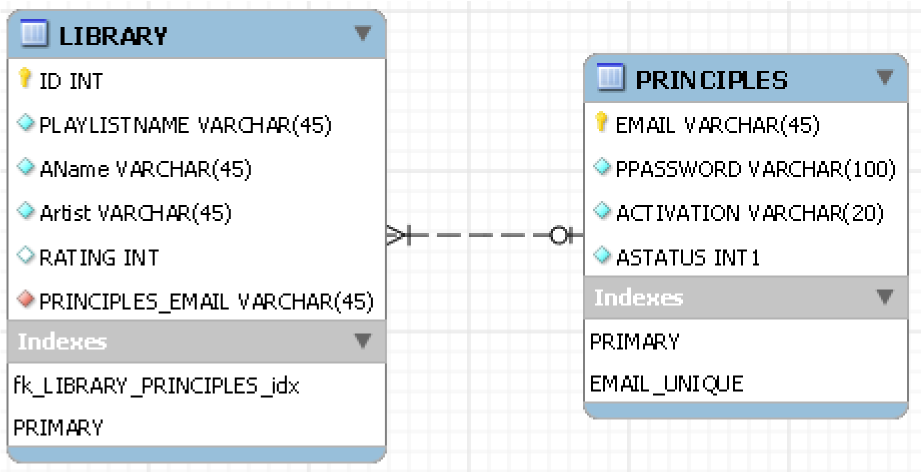
5.2 Context Data Flow Diagram



5.3 Entity Relationship Diagram



5.4 Class Diagram

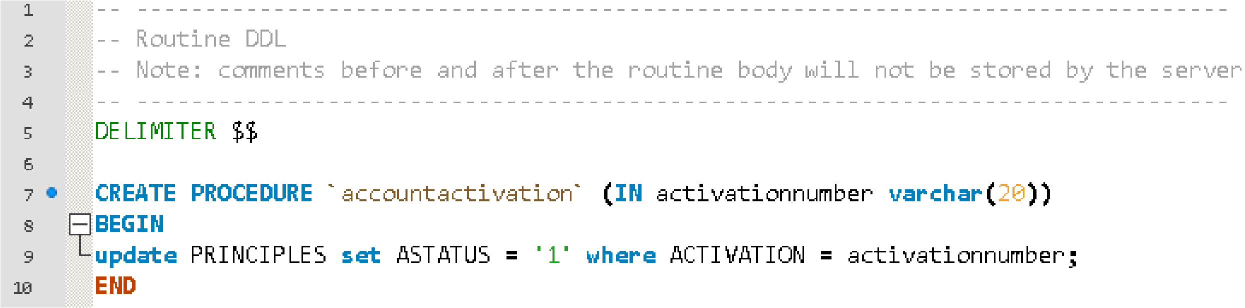


LIBRARY
ID INT
PLAYLISTNAME VARCHAR(45)
AName VARCHAR(45)
Artist VARCHAR(45)
RATING INT
PRINCIPLES_EMAIL VARCHAR(45)
Indexes
fk_LIBRARY_PRINCIPLES_idx
PRIMARY

PRINCIPLES
EMAIL VARCHAR(45)
PPASSWORD VARCHAR(100)
ACTIVATION VARCHAR(20)
ASTATUS INT1
Indexes
PRIMARY
EMAIL_UNIQUE

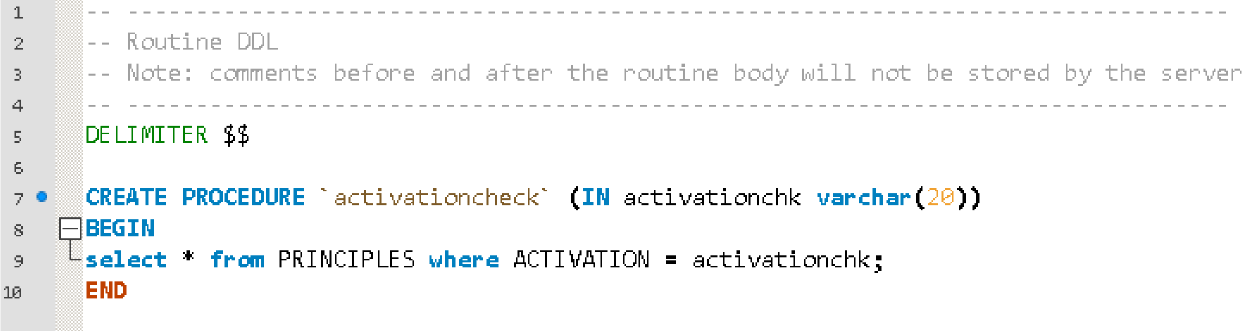
5.4.1 Database Stored Procedures

5.4.1.1 Account Activation



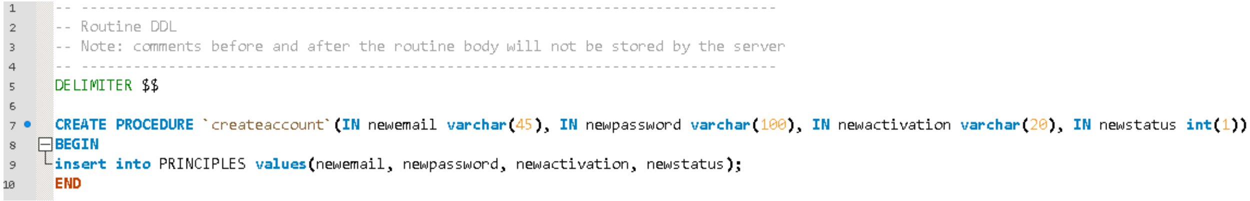
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `accountactivation` (IN activationnumber varchar(20))
BEGIN
update PRINCIPLES set ASTATUS = '1' where ACTIVATION = activationnumber;
END
```

5.4.1.2 Activation Check



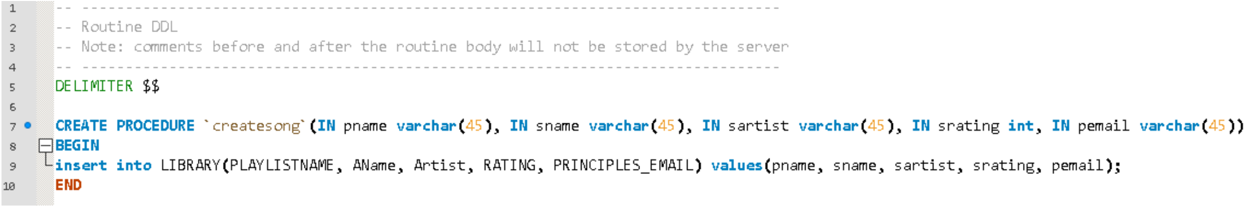
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `activationcheck` (IN activationchk varchar(20))
BEGIN
select * from PRINCIPLES where ACTIVATION = activationchk;
END
```

5.4.1.3 Create Account



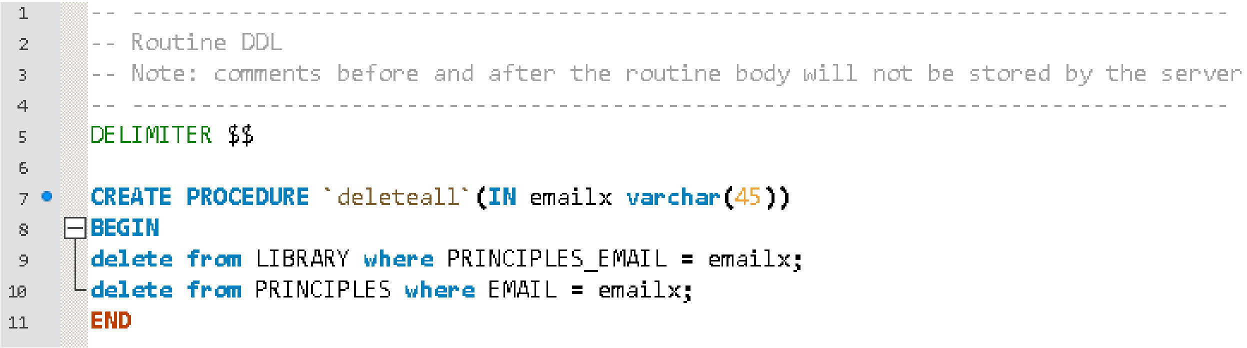
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `createaccount`(IN newemail varchar(45), IN newpassword varchar(100), IN newactivation varchar(20), IN newstatus int(1))
BEGIN
insert into PRINCIPLES values(newemail, newpassword, newactivation, newstatus);
END
```

5.4.1.4 Create Song



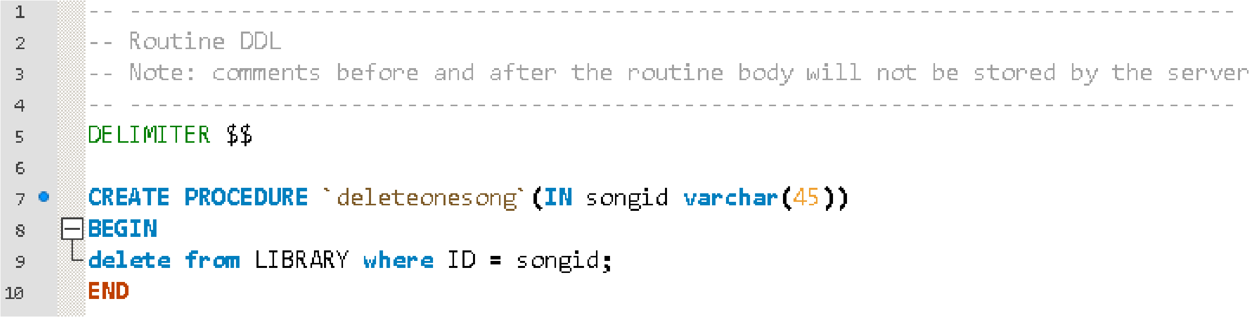
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `createsong`(IN pname varchar(45), IN sname varchar(45), IN sartist varchar(45), IN srating int, IN pemail varchar(45))
BEGIN
insert into LIBRARY(PLAYLISTNAME, AName, Artist, RATING, PRINCIPLES_EMAIL) values(pname, sname, sartist, srating, pemail);
END
```

5.4.1.5 Delete All



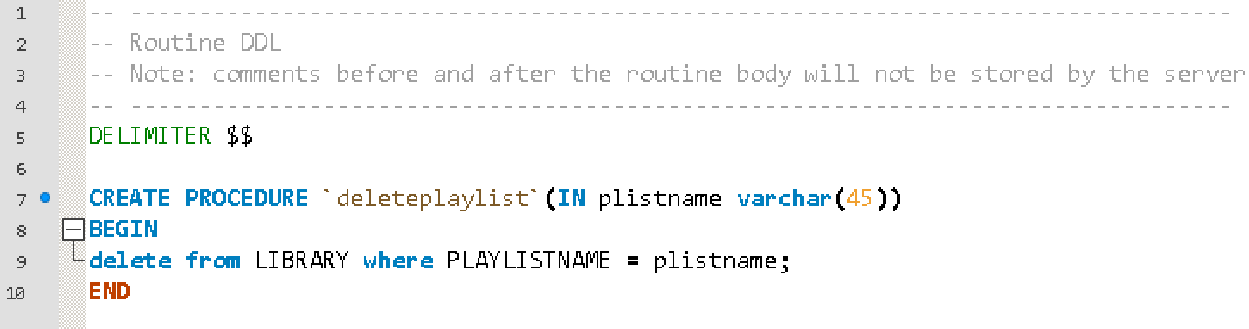
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `deleteall`(IN emailx varchar(45))
BEGIN
delete from LIBRARY where PRINCIPLES_EMAIL = emailx;
delete from PRINCIPLES where EMAIL = emailx;
END
```

3.4.1.6 Delete One Song



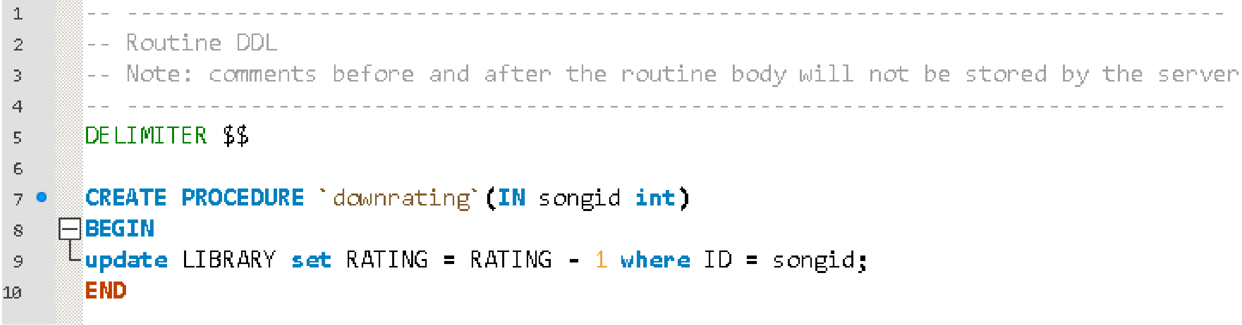
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `deleteonesong`(IN songid varchar(45))
BEGIN
delete from LIBRARY where ID = songid;
END
```

5.4.1.7 Delete Playlist



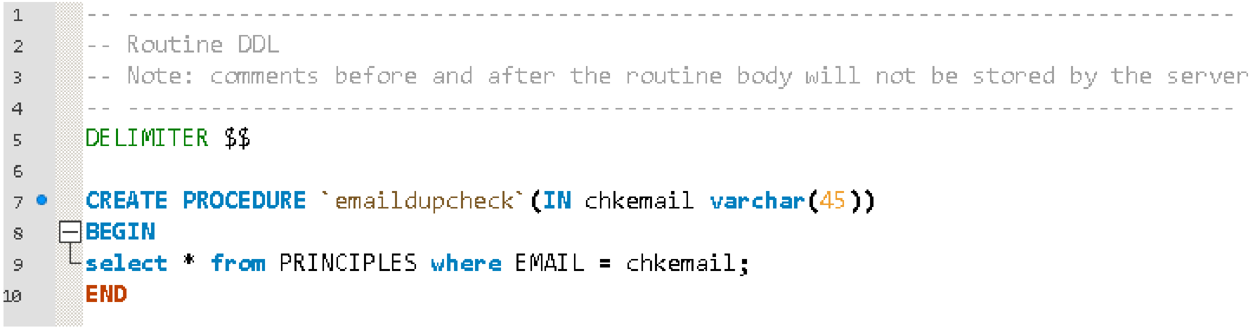
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `deleteplaylist`(IN plistname varchar(45))
BEGIN
delete from LIBRARY where PLAYLISTNAME = plistname;
END
```

5.4.1.8 Down Rating



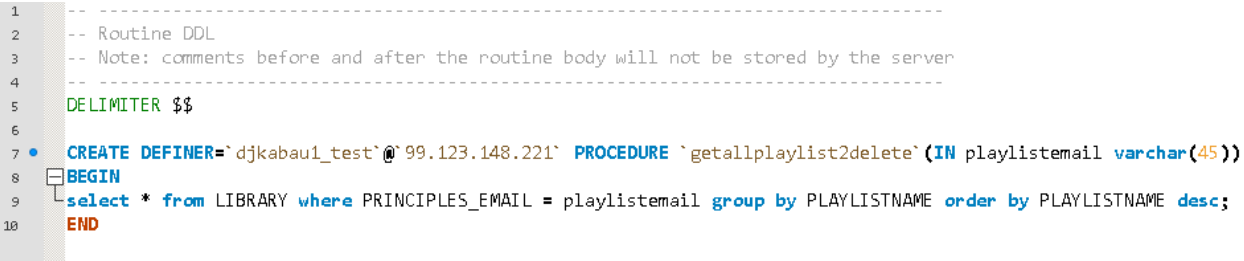
```sql
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `downrating`(IN songid int)
BEGIN
update LIBRARY set RATING = RATING - 1 where ID = songid;
END
```

5.4.1.9 Email Duplicate Check



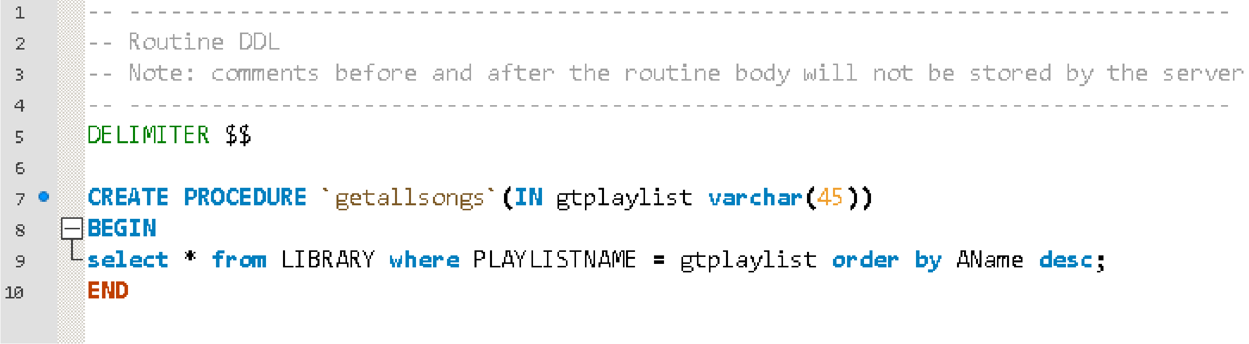
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `emaildupcheck`(IN chkemail varchar(45))
BEGIN
select * from PRINCIPLES where EMAIL = chkemail;
END
```

5.4.1.10 Get All Playlist To Delete



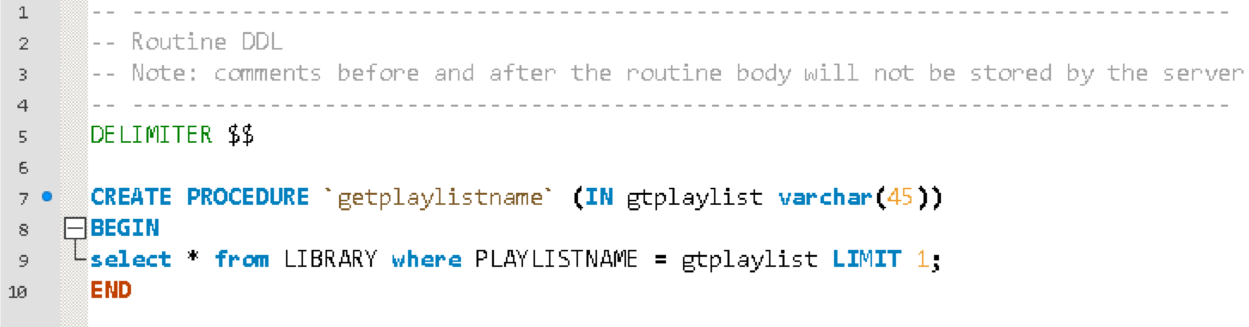
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE DEFINER=`djkabau1_test`@`99.123.148.221` PROCEDURE `getallplaylist2delete`(IN playlistemail varchar(45))
BEGIN
select * from LIBRARY where PRINCIPLES_EMAIL = playlistemail group by PLAYLISTNAME order by PLAYLISTNAME desc;
END
```

5.4.1.11 Get All Songs



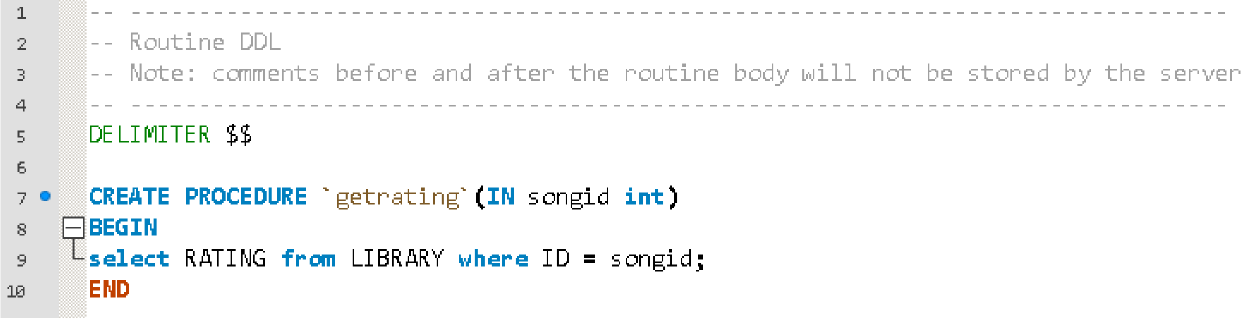
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `getallsongs`(IN gtplaylist varchar(45))
BEGIN
select * from LIBRARY where PLAYLISTNAME = gtplaylist order by AName desc;
END
```

5.4.1.12 Get Playlist Name



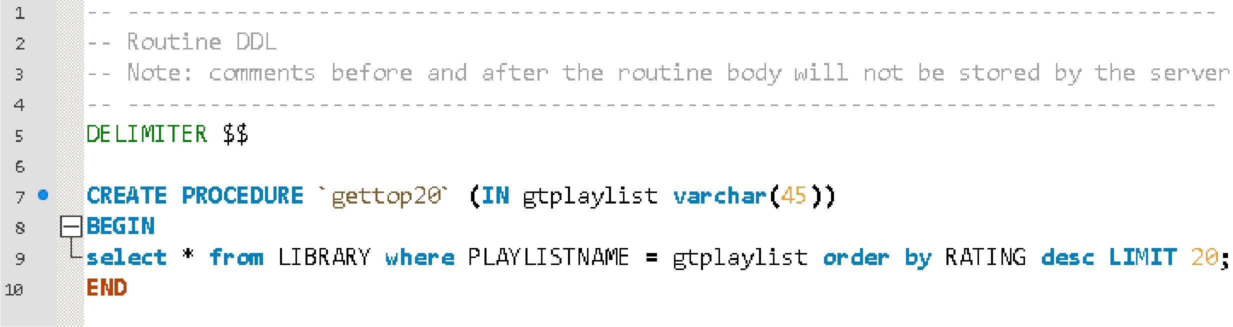
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `getplaylistname` (IN gtplaylist varchar(45))
BEGIN
select * from LIBRARY where PLAYLISTNAME = gtplaylist LIMIT 1;
END
```

5.4.1.13 Get Rating



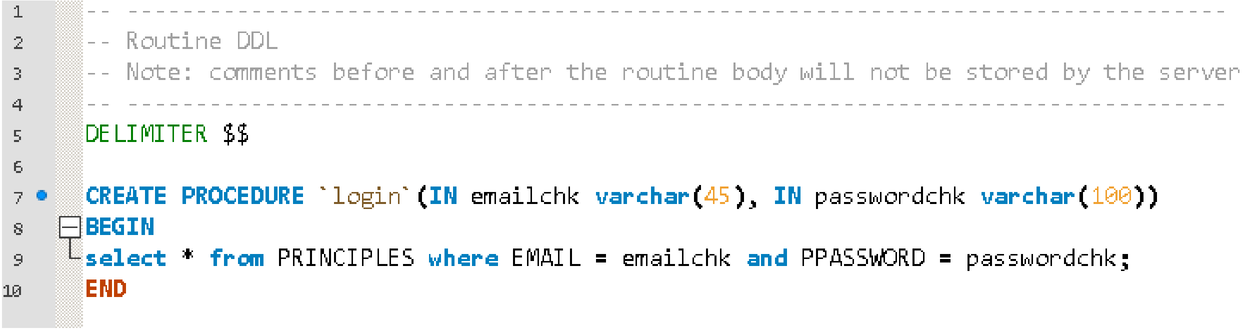
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `getrating`(IN songid int)
BEGIN
select RATING from LIBRARY where ID = songid;
END
```

5.4.1.14 Get Top 20



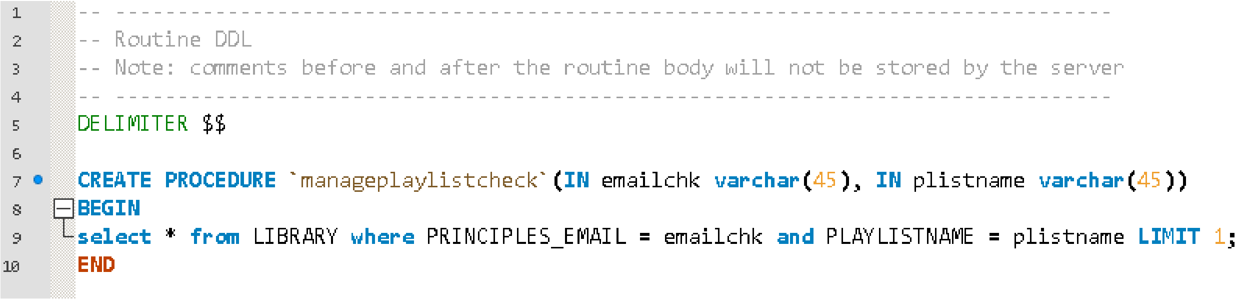
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `gettop20` (IN gtplaylist varchar(45))
BEGIN
select * from LIBRARY where PLAYLISTNAME = gtplaylist order by RATING desc LIMIT 20;
END
```

5.4.1.15 Login



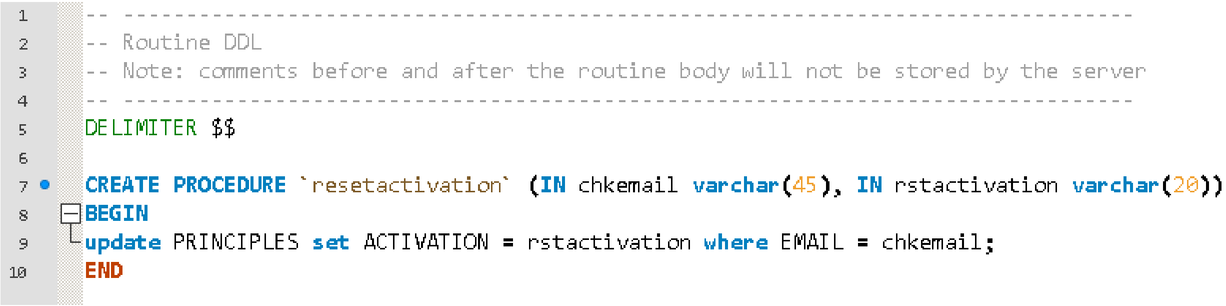
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `login`(IN emailchk varchar(45), IN passwordchk varchar(100))
BEGIN
select * from PRINCIPLES where EMAIL = emailchk and PPASSWORD = passwordchk;
END
```

5.4.1.16 Manage Playlist Check



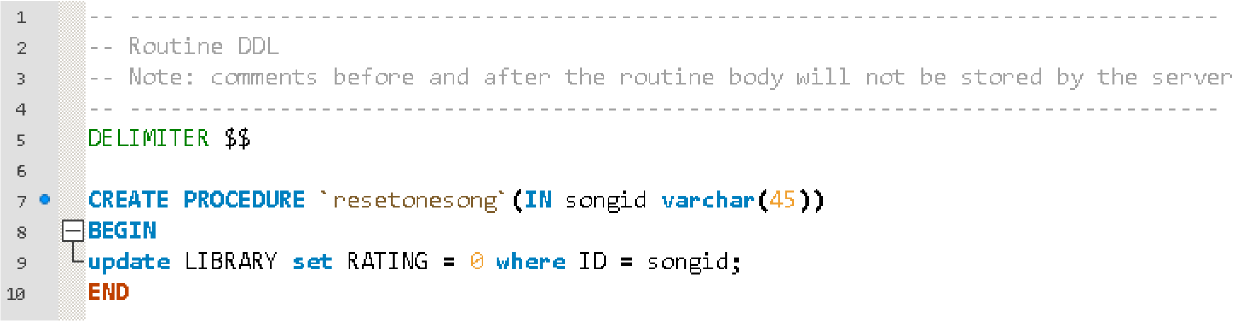
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `manageplaylistcheck`(IN emailchk varchar(45), IN plistname varchar(45))
BEGIN
select * from LIBRARY where PRINCIPLES_EMAIL = emailchk and PLAYLISTNAME = plistname LIMIT 1;
END
```

5.4.1.17 Reset Activation



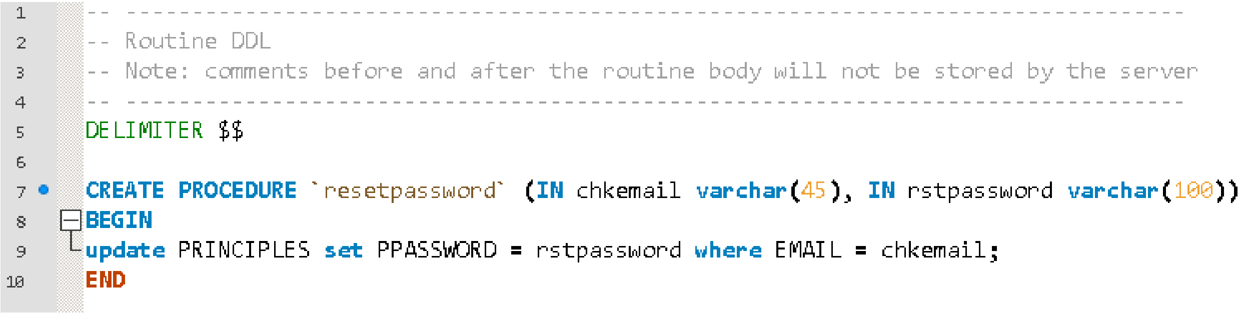
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `resetactivation` (IN chkemail varchar(45), IN rstactivation varchar(20))
BEGIN
update PRINCIPLES set ACTIVATION = rstactivation where EMAIL = chkemail;
END
```

5.4.1.18 Reset One Song



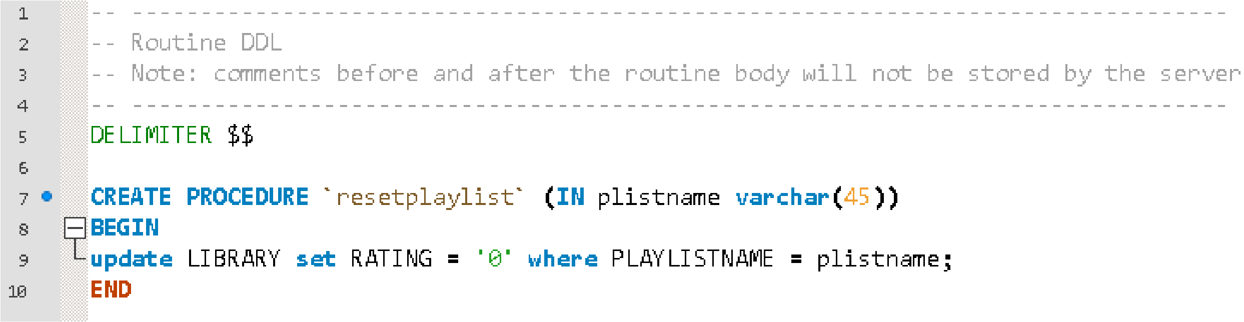
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `resetonesong`(IN songid varchar(45))
BEGIN
update LIBRARY set RATING = 0 where ID = songid;
END
```

5.4.1.19 Reset Password



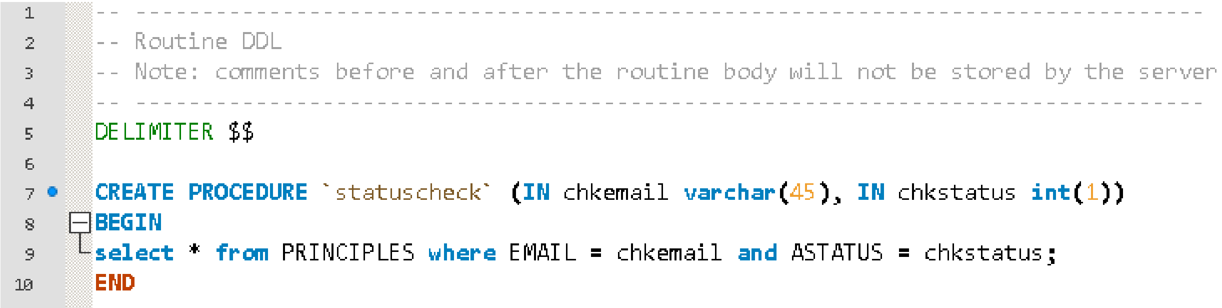
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `resetpassword` (IN chkemail varchar(45), IN rstpassword varchar(100))
BEGIN
update PRINCIPLES set PPASSWORD = rstpassword where EMAIL = chkemail;
END
```

5.4.1.20 Reset Playlist



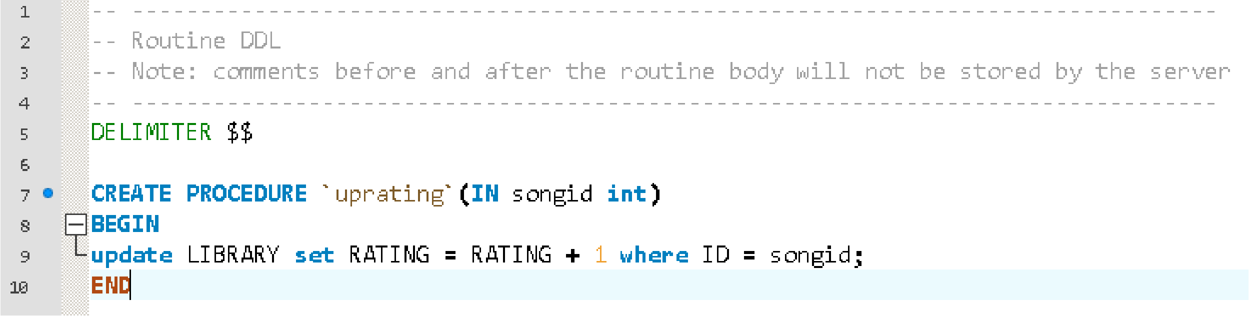
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `resetplaylist` (IN plistname varchar(45))
BEGIN
update LIBRARY set RATING = '0' where PLAYLISTNAME = plistname;
END
```

5.4.1.21 Status Check



```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `statuscheck` (IN chkemail varchar(45), IN chkstatus int(1))
BEGIN
select * from PRINCIPLES where EMAIL = chkemail and ASTATUS = chkstatus;
END
```

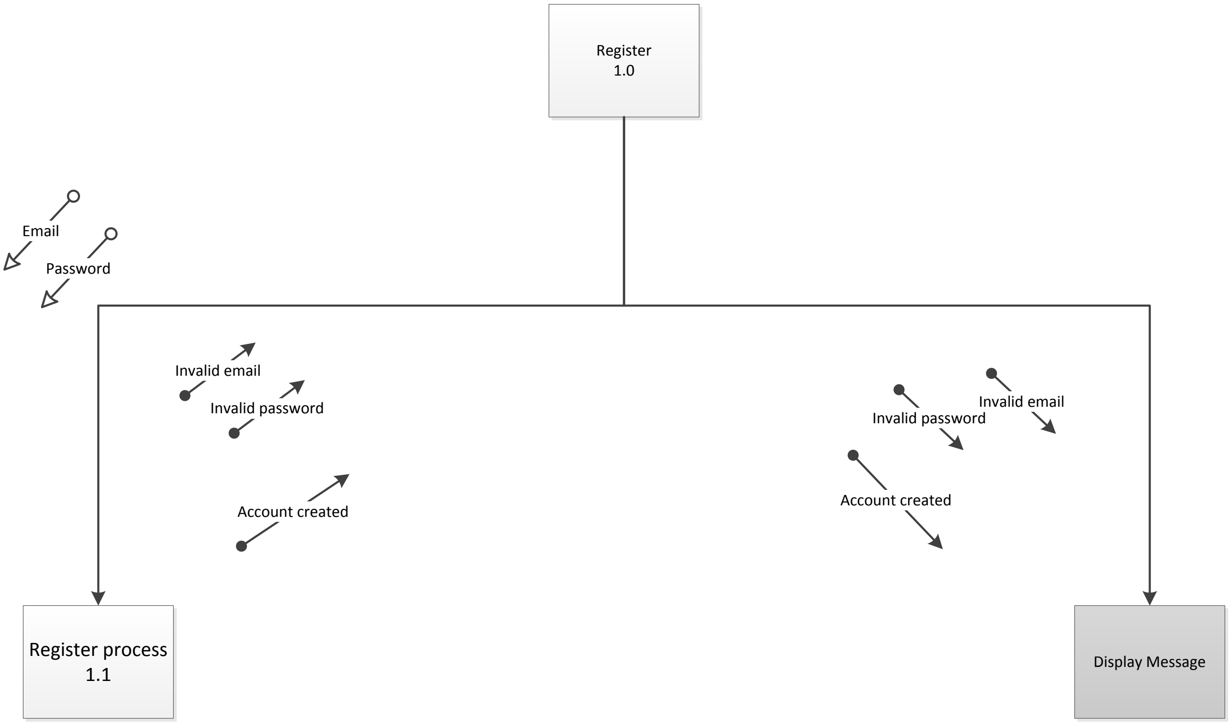
5.4.1.22 Up Rating



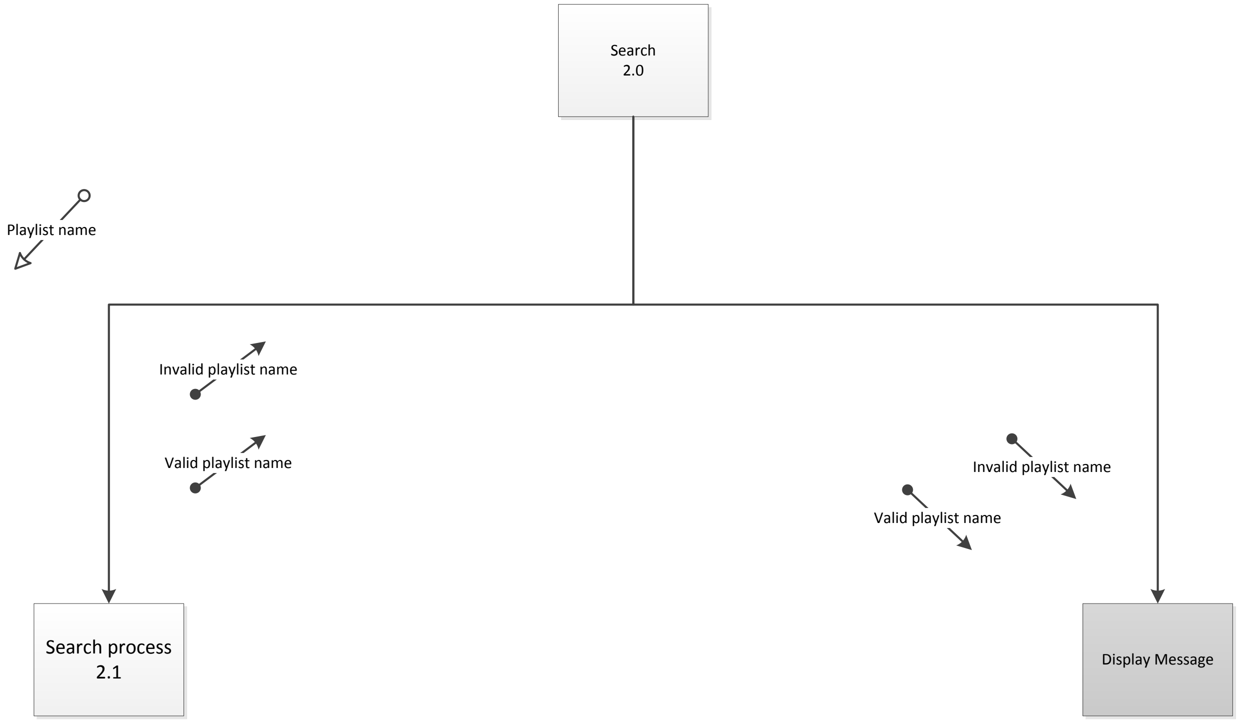
```
-- --------------------------------------------------------------------------------
-- Routine DDL
-- Note: comments before and after the routine body will not be stored by the server
-- --------------------------------------------------------------------------------
DELIMITER $$

CREATE PROCEDURE `uprating`(IN songid int)
BEGIN
update LIBRARY set RATING = RATING + 1 where ID = songid;
END
```

5.5 Data Structured Diagram

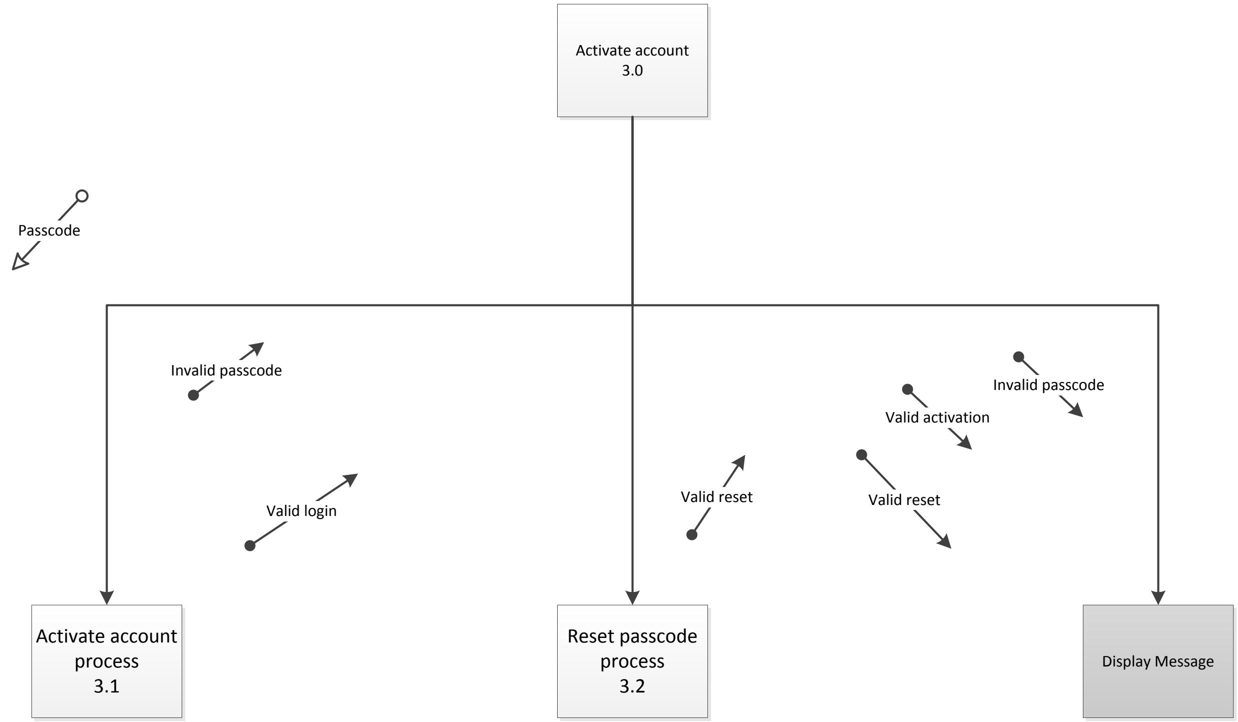
5.5.1 Register



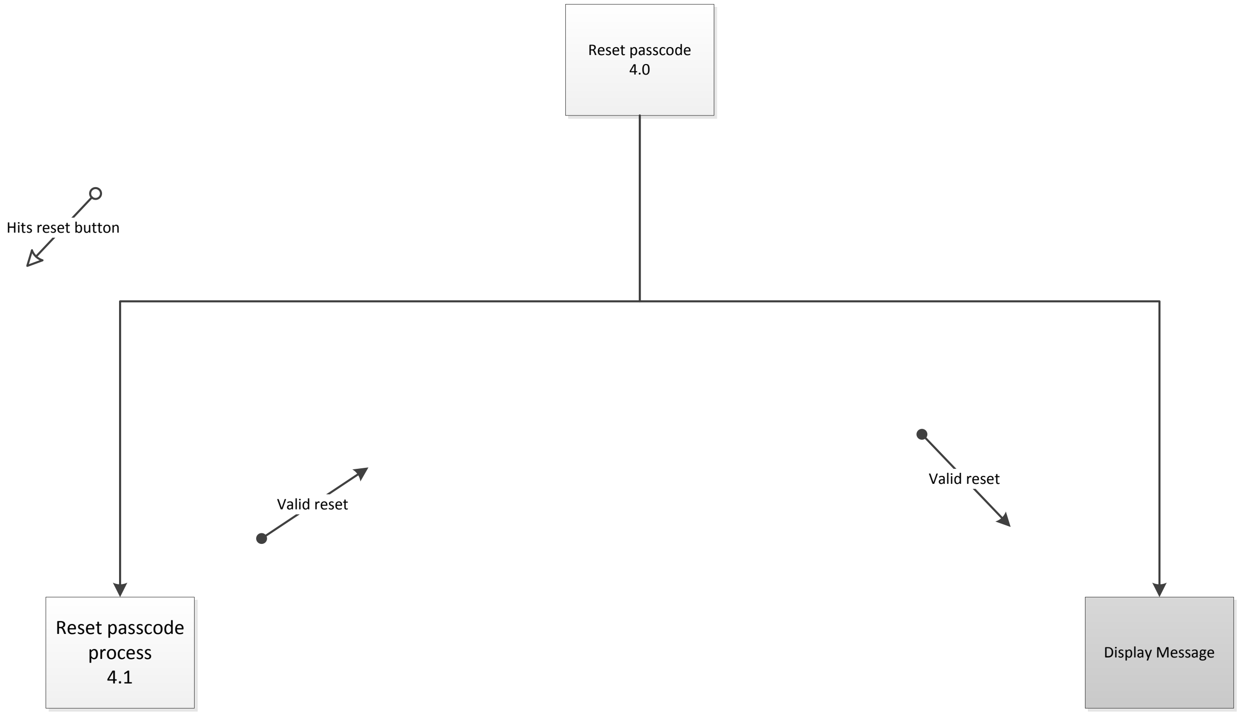
5.5.2 Search



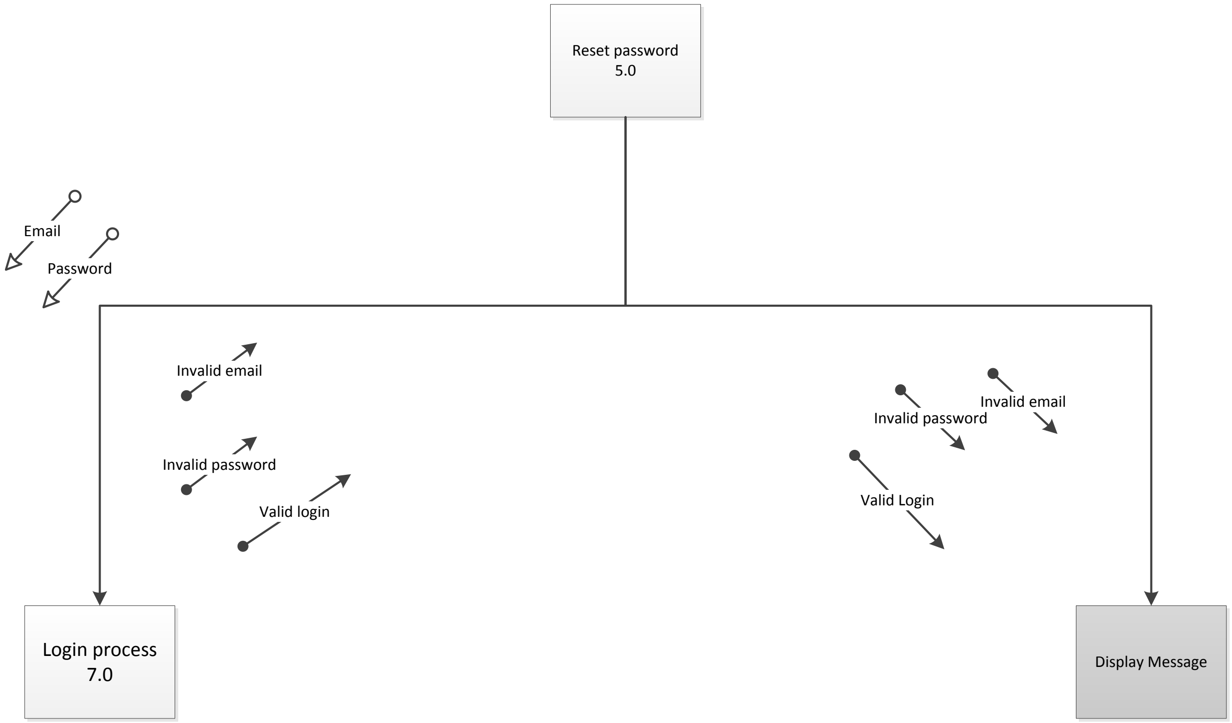
5.5.3 Activate account



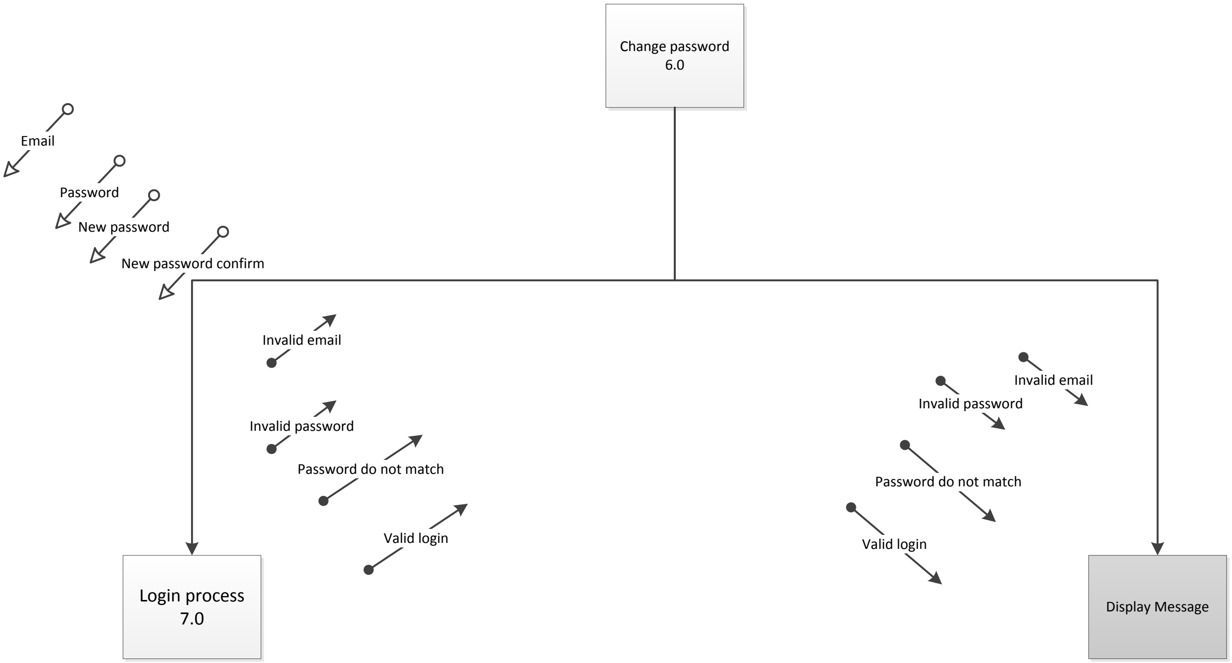
5.5.4 Reset passcode



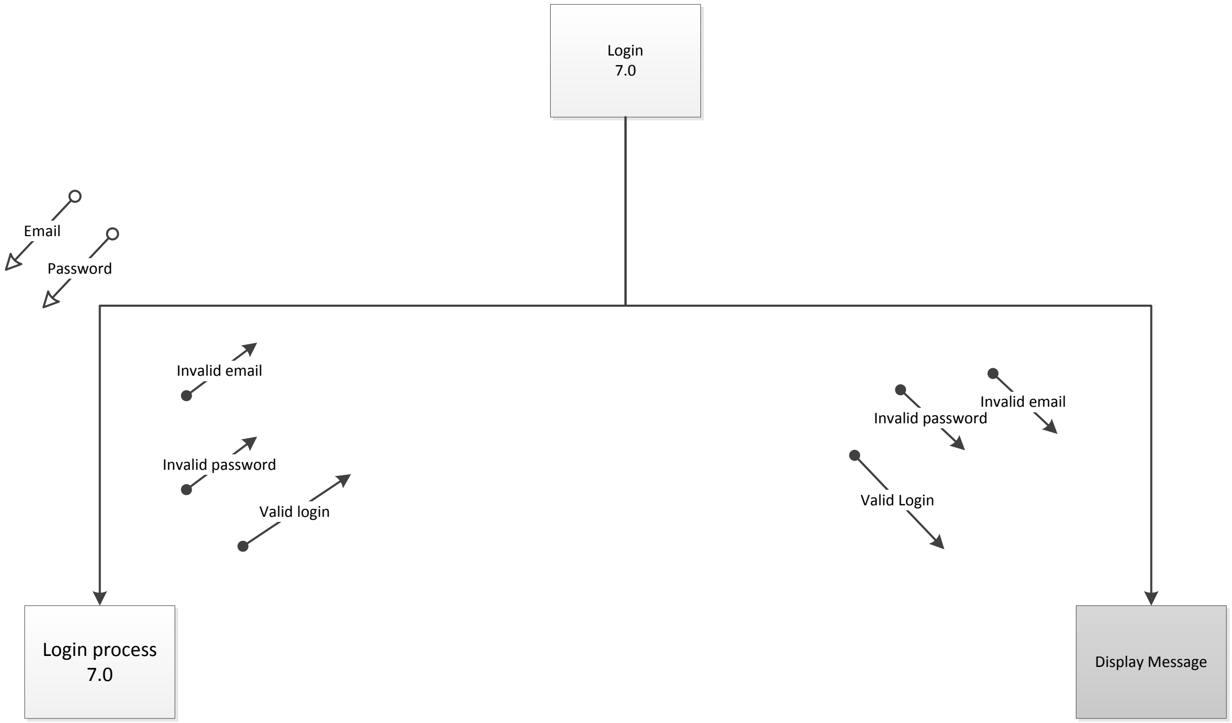
5.5.5 Reset password



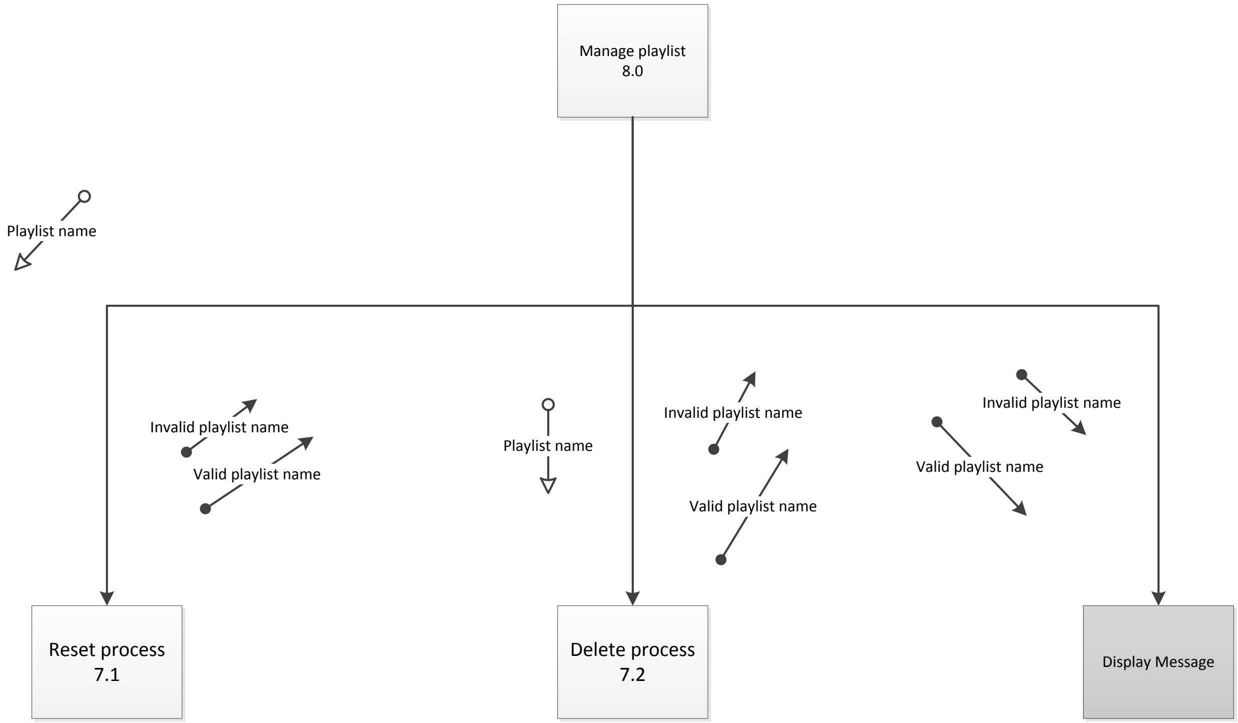
5.5.6 Change password



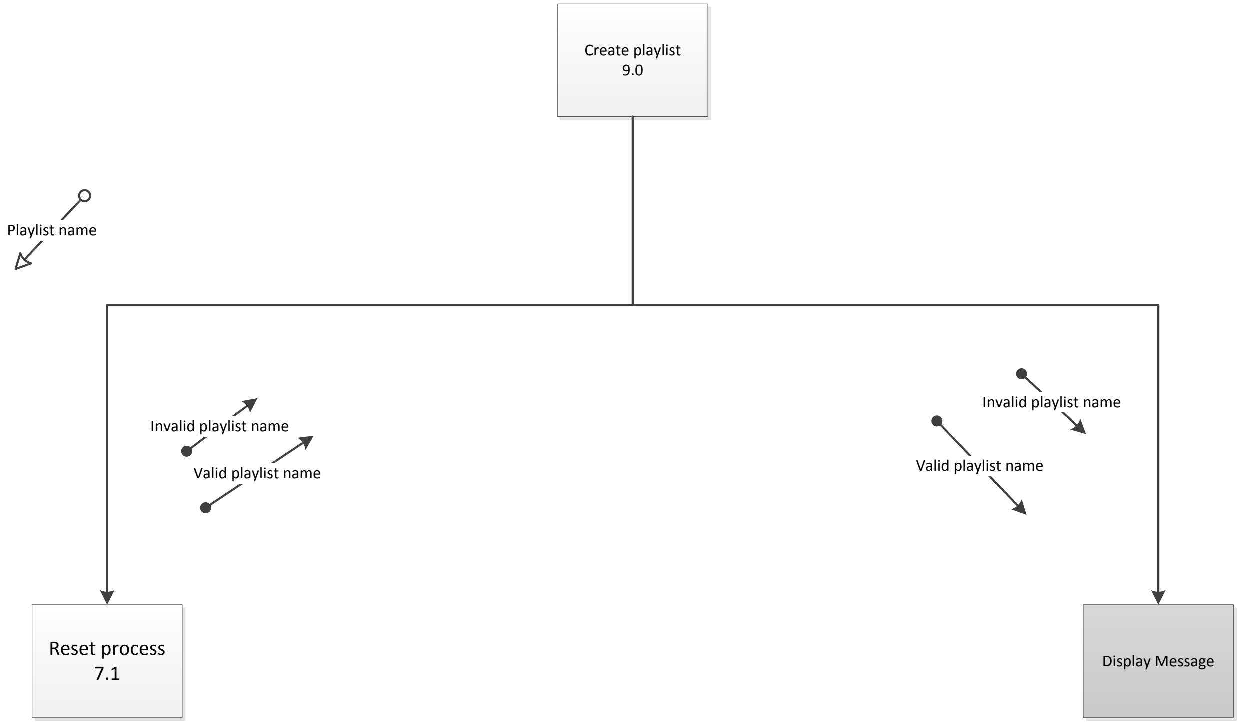
5.5.7 Login



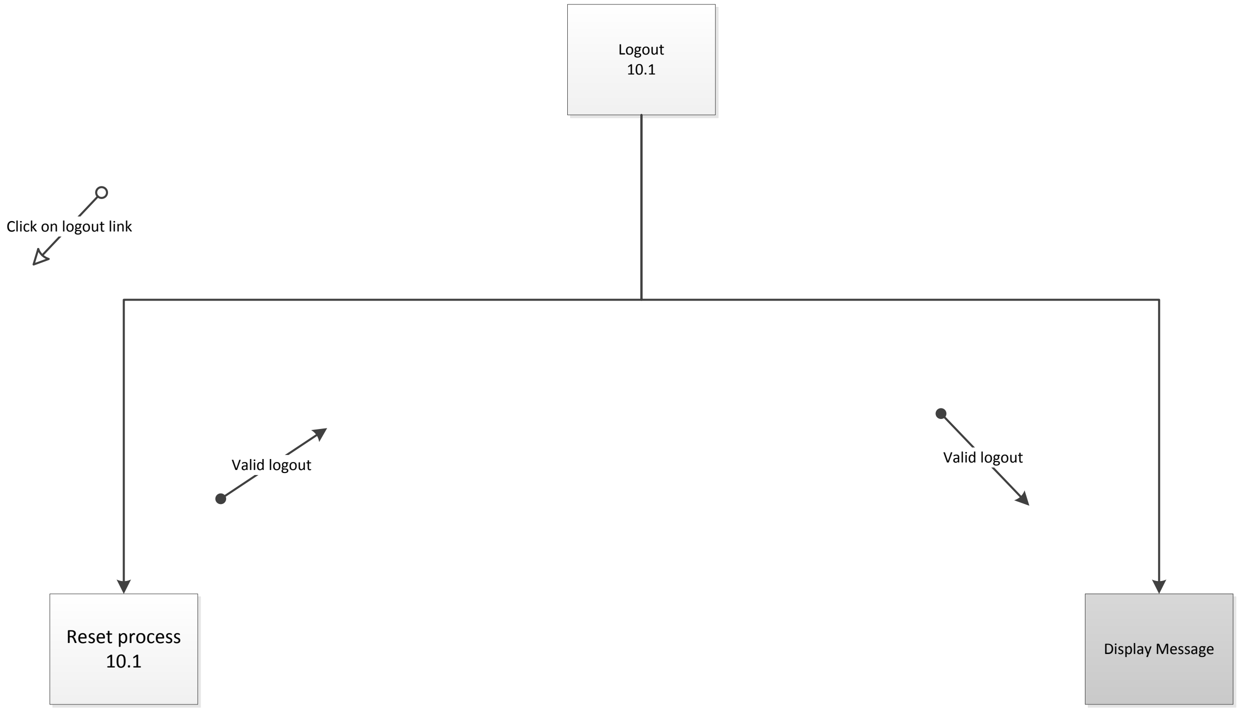
5.5.8 Manage playlist



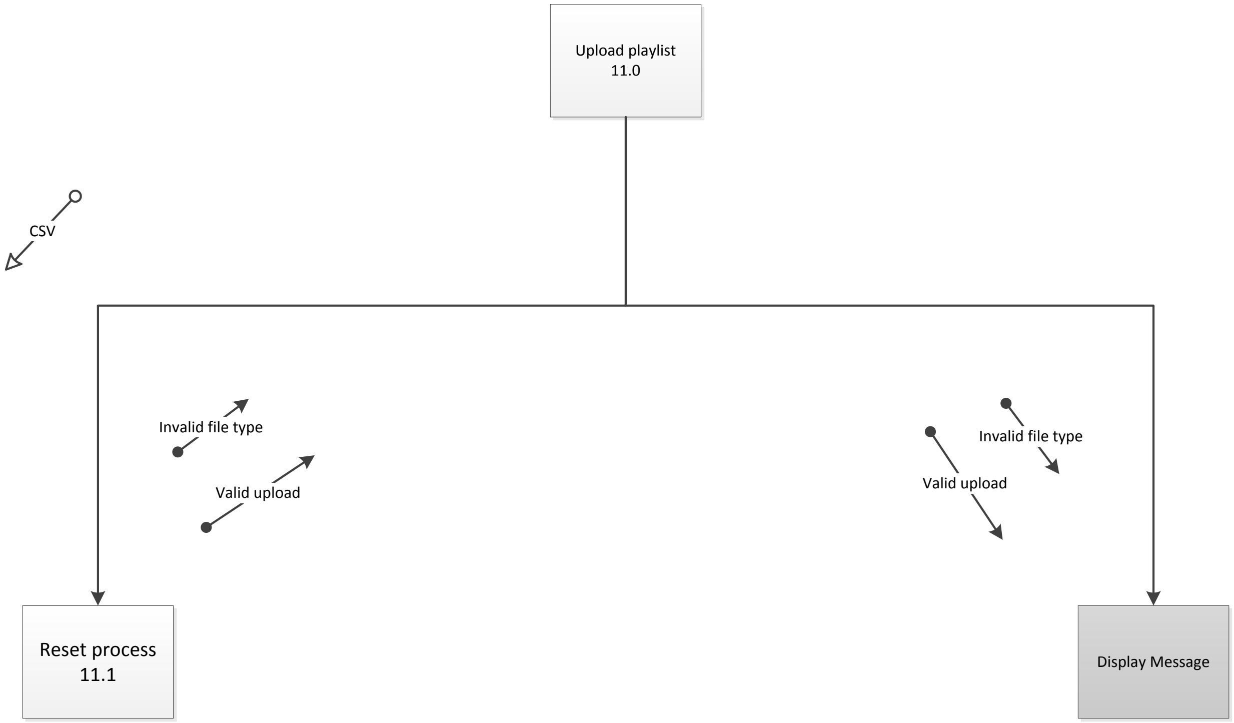
5.5.9 Create playlist



5.5.10 Logout



5.5.11 Upload playlist

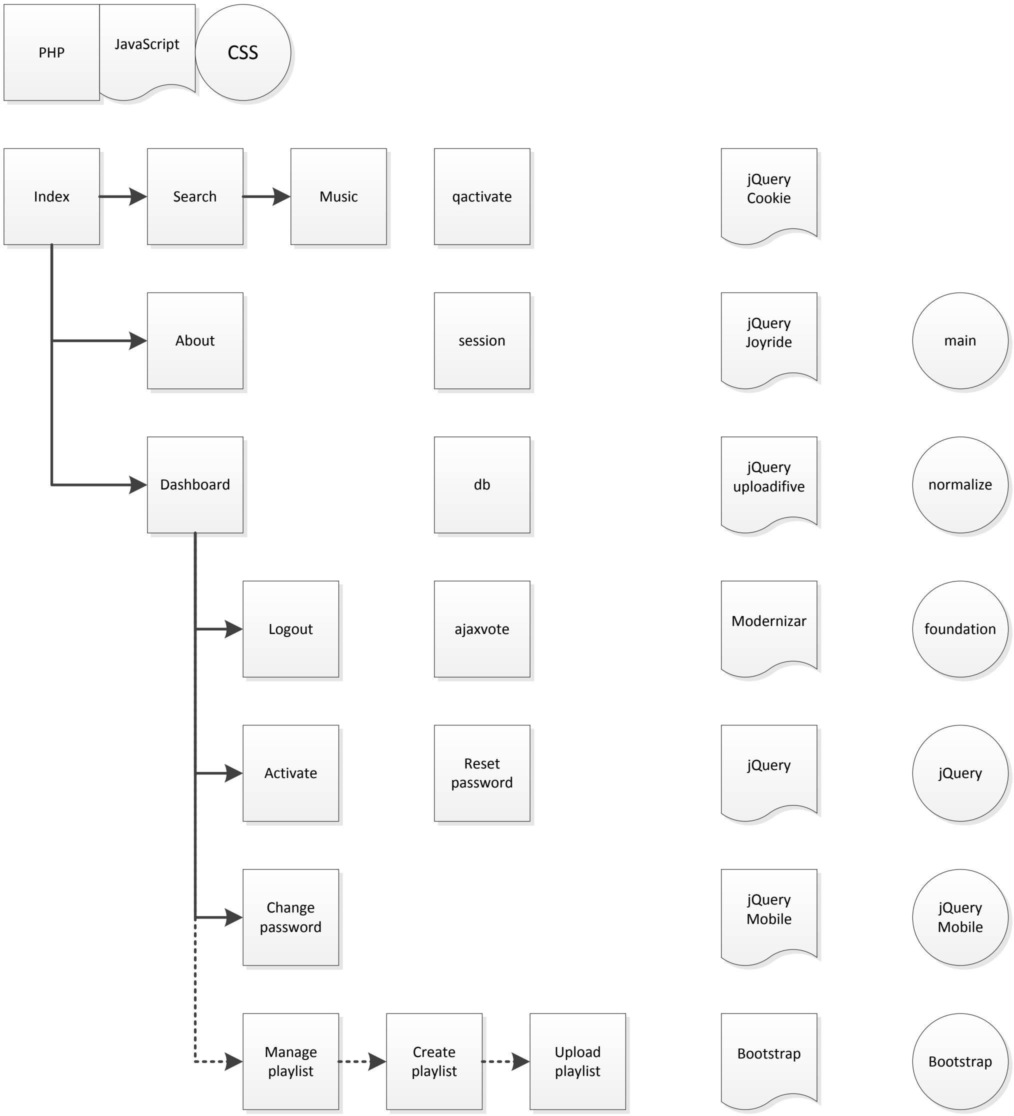


5.6 Dataflow Diagram

5.7 Activity Diagram

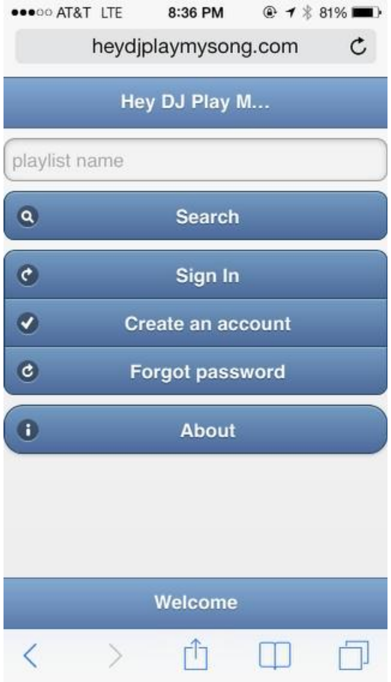
5.8 Sequence Diagram

5.8 Conceptual Website Diagram

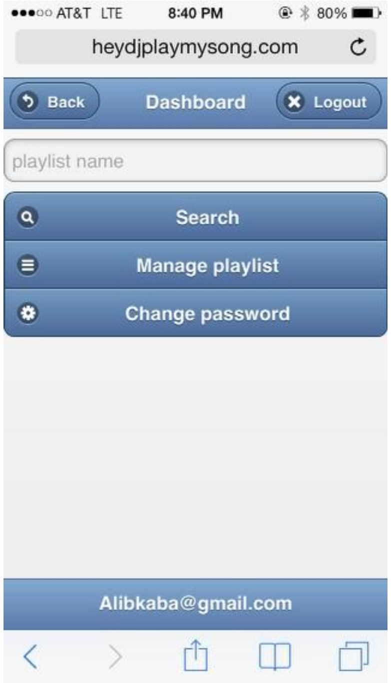


5.0 Screenshots

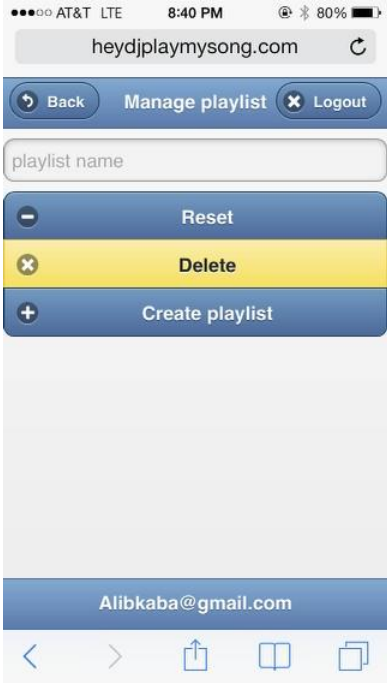
5.1 index.php



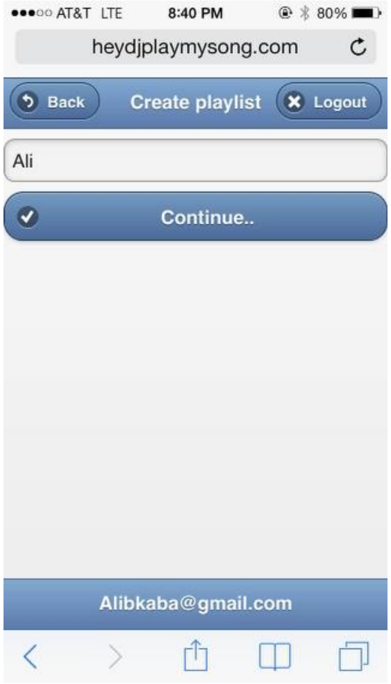
5.2 dashboard.php



5.3 manageplaylist.php

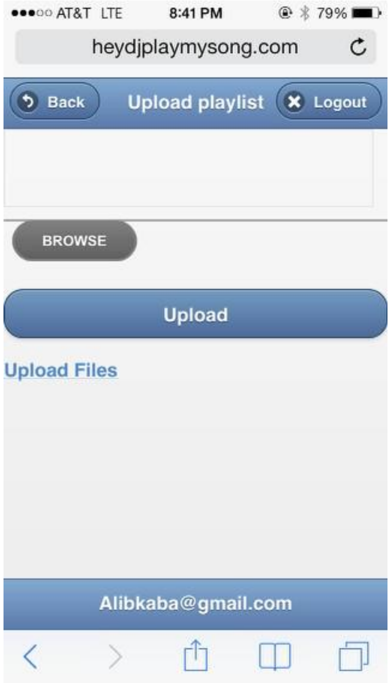


5.4 createplaylist.php

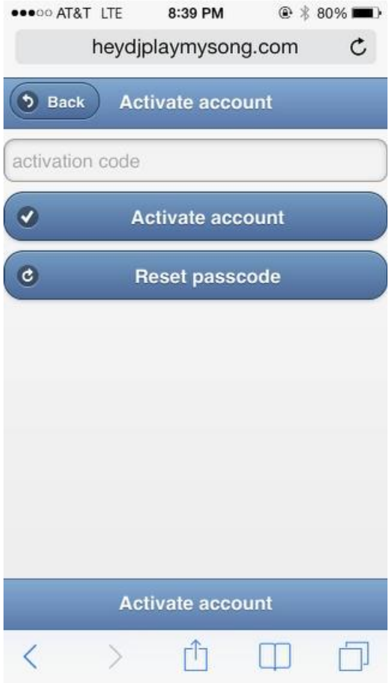


Ali

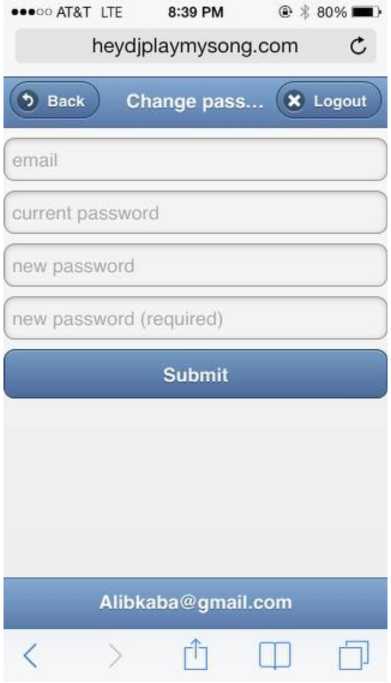
5.5 uploadplaylist.php



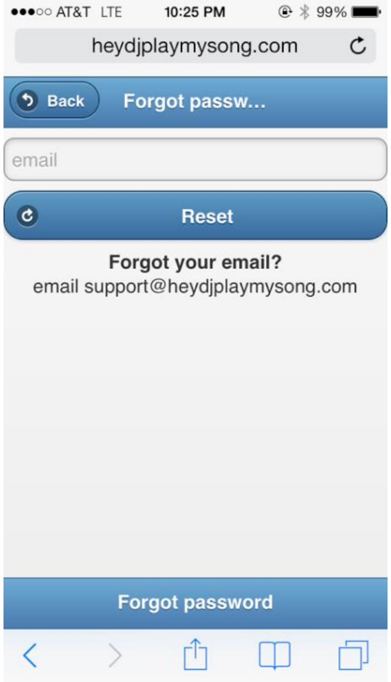
5.6 activate.php



5.7 changepassword.php

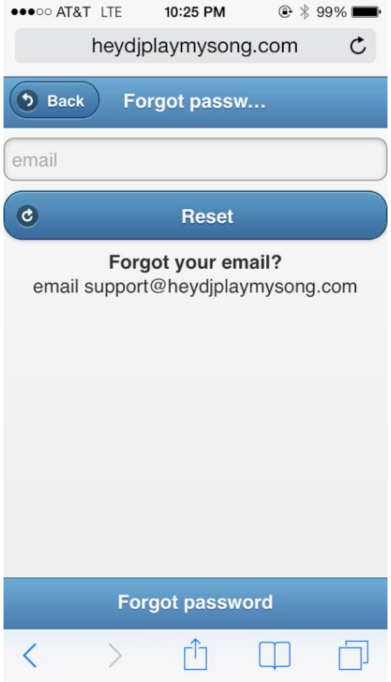


5.8 resetpassword.php



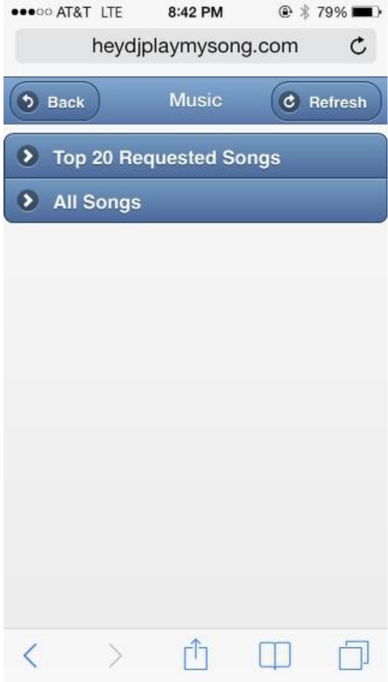
Forgot your email?
email support@heydjplaymysong.com

5.9 forgotpassword.php



Forgot your email?
email support@heydjplaymysong.com

5.10 music.php



reset

-

6 English Instruction Manual

6.1 Using Hey DJ Play My Song

Dear Customer,

Thank you for using Hey DJ Play My Song system.

Hey DJ Play My Song (HDJPMS) is a 21st century solution of requesting songs. The system is an online tool that consists of playlist created by your host, DJs and anyone looking to give their company a great time.

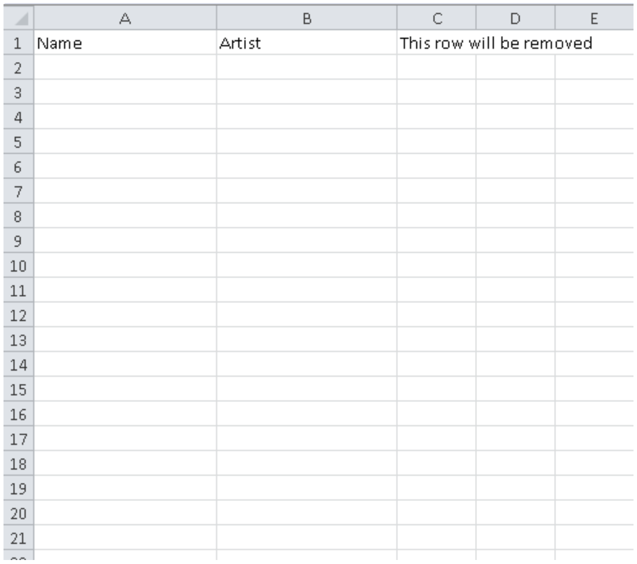
6.2 Getting Started

Every playlist available on HDJPMS was imported using comma-separated values (CSV) document. A CSV is document format that can be made from a numerous applications.

6.2.1 Getting/creating the CSV document

A template is available for download on the swebsite **About** page and should be followed accordingly for the songs to be uploaded correctly.

Figure 1 below is how the document should look like.

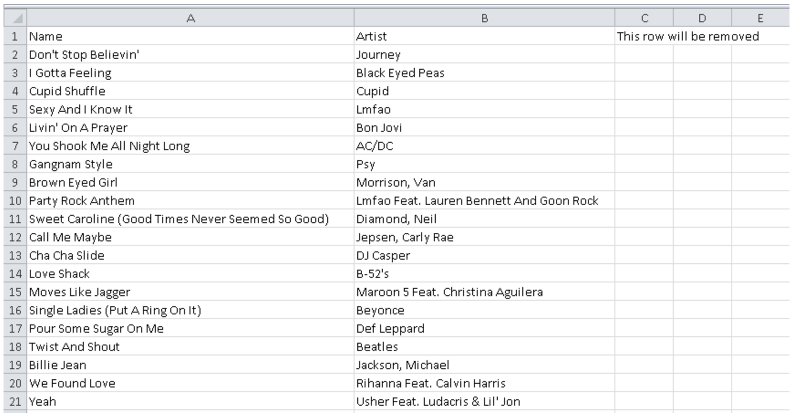


	A	B	C	D	E
1	Name	Artist	This row will be removed		
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
19					
20					
21					

Figure 1

6.2.2 Getting the music

Populate the field with data as it is visually shown on Figure 2.

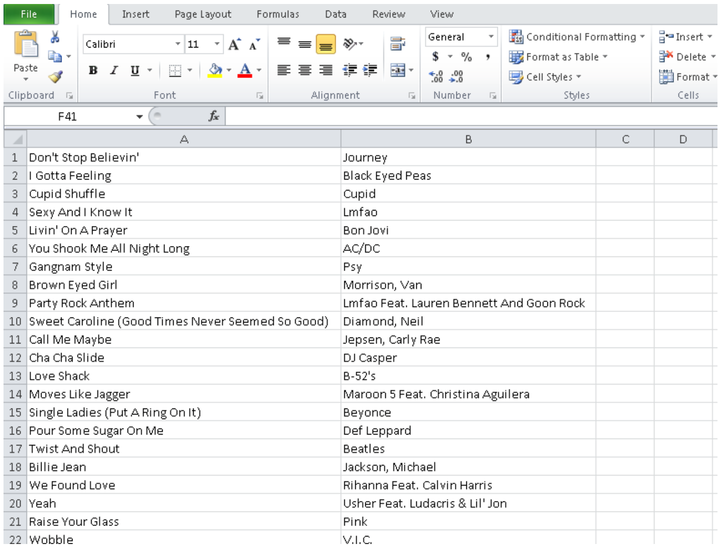


	A	B	C	D	E
1	Name	Artist	This row will be removed		
2	Don't Stop Believin'	Journey			
3	I Gotta Feeling	Black Eyed Peas			
4	Cupid Shuffle	Cupid			
5	Sexy And I Know It	Lmfao			
6	Livin' On A Prayer	Bon Jovi			
7	You Shook Me All Night Long	AC/DC			
8	Gangnam Style	Psy			
9	Brown Eyed Girl	Morrison, Van			
10	Party Rock Anthem	Lmfao Feat. Lauren Bennett And Goon Rock			
11	Sweet Caroline (Good Times Never Seemed So Good)	Diamond, Neil			
12	Call Me Maybe	Jepsen, Carly Rae			
13	Cha Cha Slide	DJ Casper			
14	Love Shack	B-52's			
15	Moves Like Jagger	Maroon 5 Feat. Christina Aguilera			
16	Single Ladies (Put A Ring On It)	Beyonce			
17	Pour Some Sugar On Me	Def Leppard			
18	Twist And Shout	Beatles			
19	Billie Jean	Jackson, Michael			
20	We Found Love	Rihanna Feat. Calvin Harris			
21	Yeah	Usher Feat. Ludacris & Lil' Jon			

Figure 2

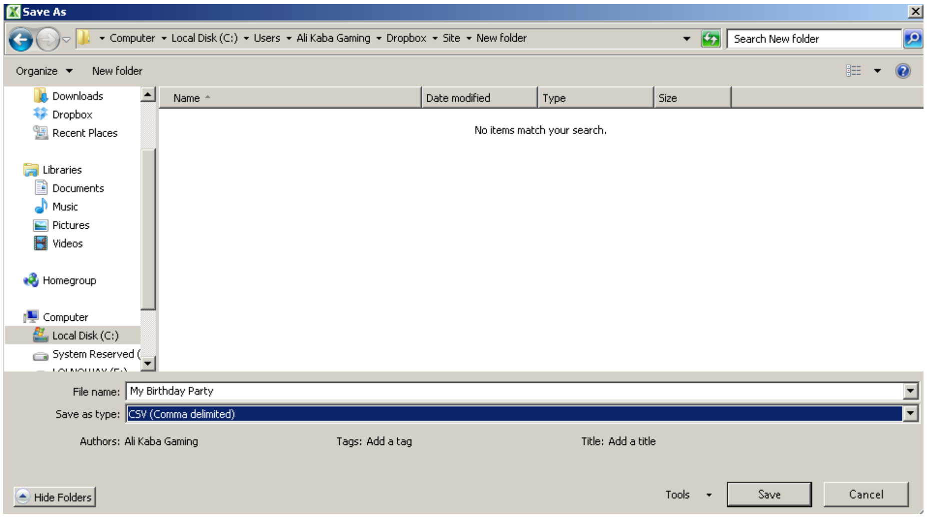
6.2.3 Formatting the CSV for HDJPMS system

Remove the first row and save the file as it is visually shown on Figure 3 & 4.



	A	B	C	D
1	Don't Stop Believin'	Journey		
2	I Gotta Feeling	Black Eyed Peas		
3	Cupid Shuffle	Cupid		
4	Sexy And I Know It	Lmfao		
5	Livin' On A Prayer	Bon Jovi		
6	You Shook Me All Night Long	AC/DC		
7	Gangnam Style	Psy		
8	Brown Eyed Girl	Morrison, Van		
9	Party Rock Anthem	Lmfao Feat. Lauren Bennett And Goon Rock		
10	Sweet Caroline (Good Times Never Seemed So Good)	Diamond, Neil		
11	Call Me Maybe	Jepsen, Carly Rae		
12	Cha Cha Slide	DJ Casper		
13	Love Shack	B-52's		
14	Moves Like Jagger	Maroon 5 Feat. Christina Aguilera		
15	Single Ladies (Put A Ring On It)	Beyonce		
16	Pour Some Sugar On Me	Def Leppard		
17	Twist And Shout	Beatles		
18	Billie Jean	Jackson, Michael		
19	We Found Love	Rihanna Feat. Calvin Harris		
20	Yeah	Usher Feat. Ludacris & Lil' Jon		
21	Raise Your Glass	Pink		
22	Wobble	V.I.C.		

Figure 3



File name: My Birthday Party

Save as type: CSV (Comma delimited)

Authors: Ali Kaba Gaming

Tags: Add a tag

Title: Add a title

Figure 4

6.3 Setting up an account

An account is required to be able to create a playlist.

6.3.1 Visiting HeyDJPlayMySong.com

HDJPMS website as it is visually shown on Figure 5.

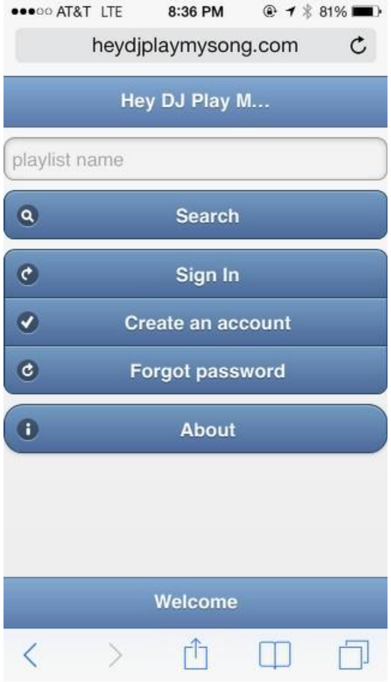
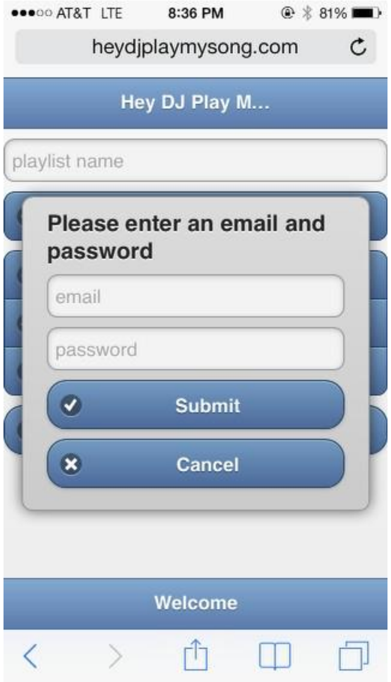


Figure 5

6.3.2 Creating an account

Click on **Create an account** and enter your email and password as it is visually shown on Figure 6. Password must be between 6 to 12 characters long.



Please enter an email and password

Figure 6

6.3.3 Verifying your account

After creating your account, you will be prompt a message to activate your account. Not activating your account will not permit you of using the manage playlist features which allows you to create a playlist as it is visually shown on Figure 7. To activate your account you either have to click on the link that was emailed to the email used to create the account or use the code listed on that same email and go to Activate account and enter it there as it is visually shown on Figure 8. Hit reset passcode to have a new one emailed again. Once your account is verified, you will access Manage playlist as it is visually shown on Figure 9.

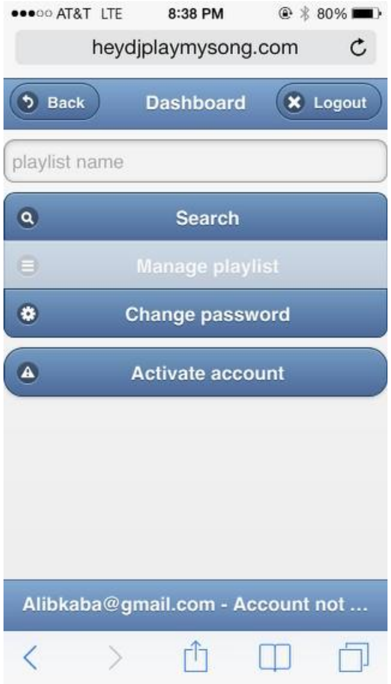


Figure 7

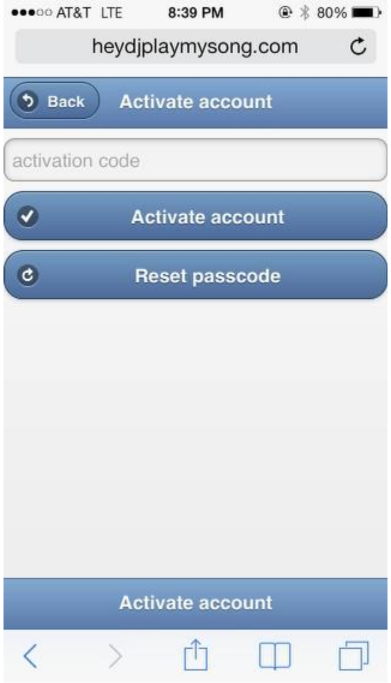


Figure 8

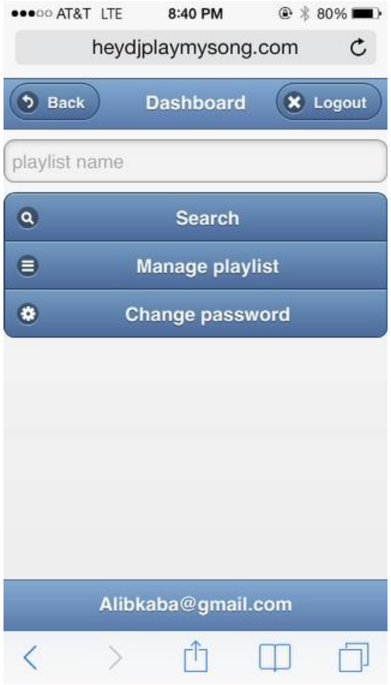


Figure 9

6.4 Dashboard

Here you can change your password, manage your playlist and search a playlist to vote on.

6.4.1 Change password

Requirements are the email, current password and typing your new password in both fields as it is visually shown on Figure 10.

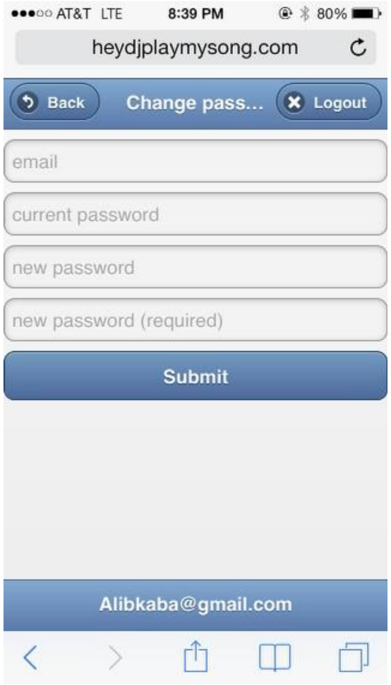


Figure 10

6.4.2 Manage playlist

HDJPMS values playlist and understands the pain of accidents. In order to prevent an accidental reset or delete, we require that you type in your playlist name. Figure 11 doesn't have any playlist listed however skip to Figure 14 to see one playlist that is listed.

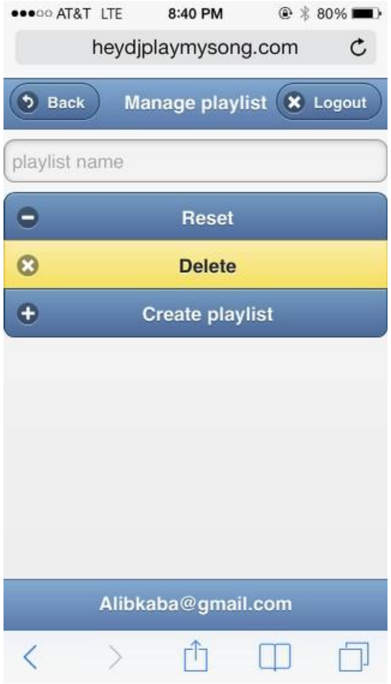
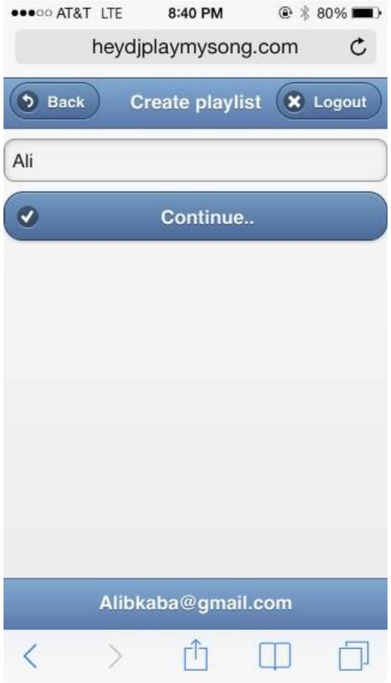


Figure 11

6.4.3 Create playlist

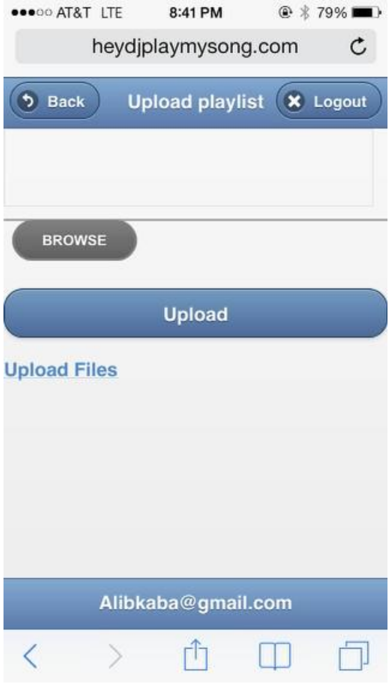
Once Create playlist is selected on Figure 11, you will be required to enter a playlist name as it is visually shown on Figure 12. If the playlist name is available, you will proceed to Figure 13 to finalize the upload process where you will click on browse, select your .CSV file and click upload where you will be re-directed back Figure 14.



Ali

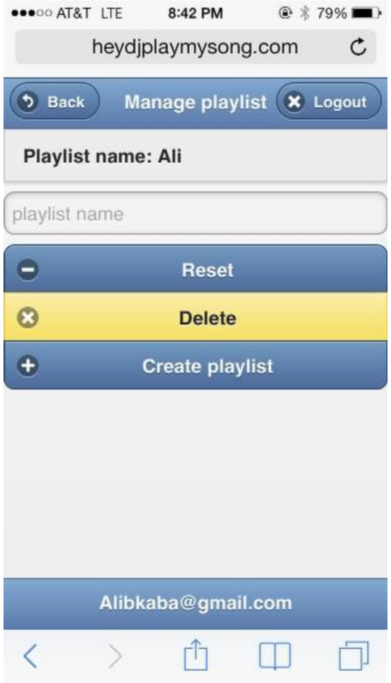
Alibkaba@gmail.com

Figure 12



Alibkaba@gmail.com

Figure 13



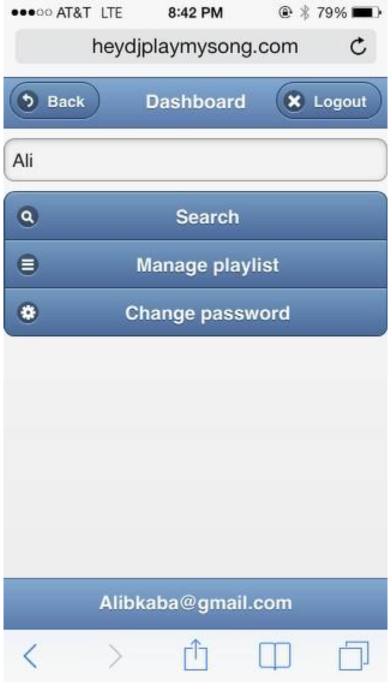
Playlist name: Ali

playlist name

Figure 14

6.5 Search

Here you can perform a search from Figure 1 or Figure 15.



Ali

Figure 15

6.5.1 Music

The music page consists of Top Requested 20 Songs and All songs as it is visually shown on Figure 16.

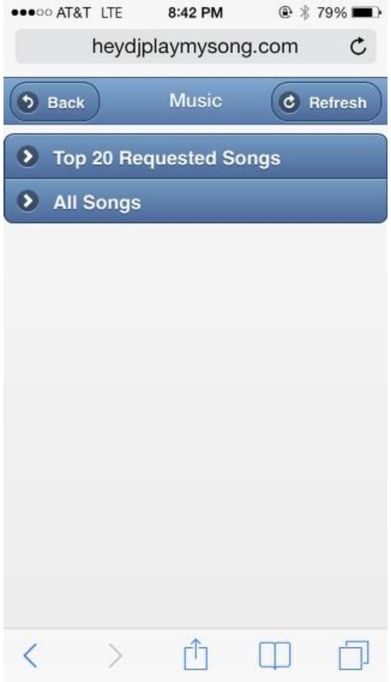


Figure 16

6.5.2 Voting

On Figure 17 and 18 you can vote for the song with the up or down arrow.   If you are the owner of the playlist, you will be able to reset or delete individual songs as it is visually shown on Figure 19.

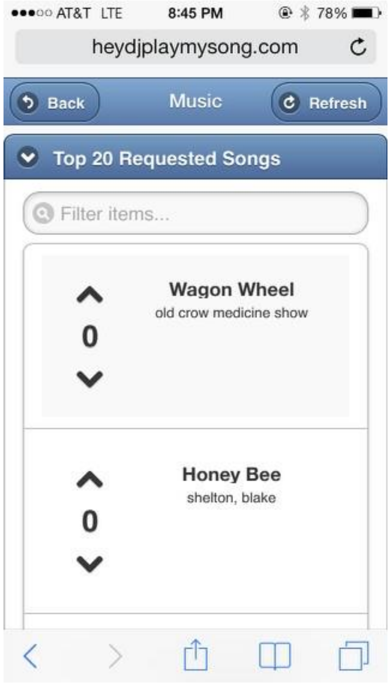


Figure 17

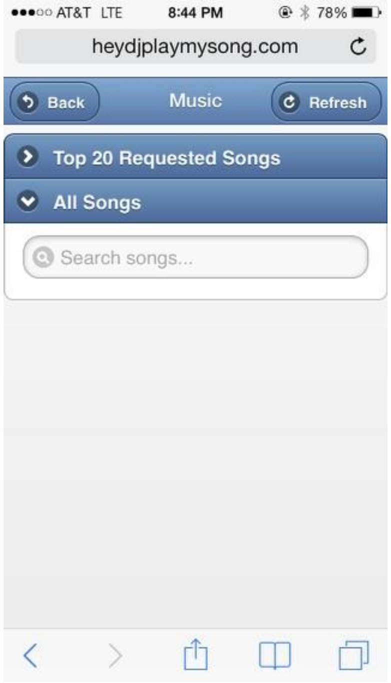


Figure 18

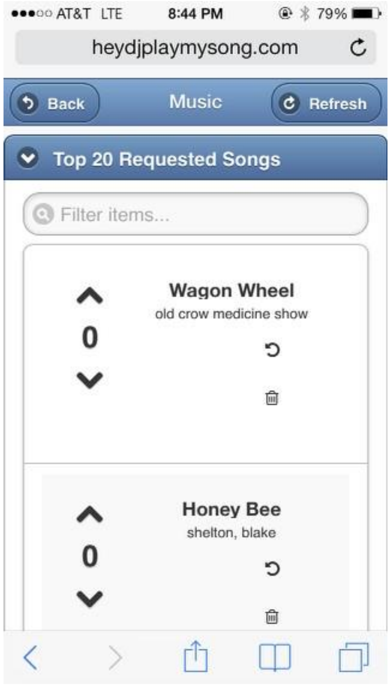


Figure 19

6.6 Homepage

One you are logged, the homepage will change from Figure 1 to Figure 20.

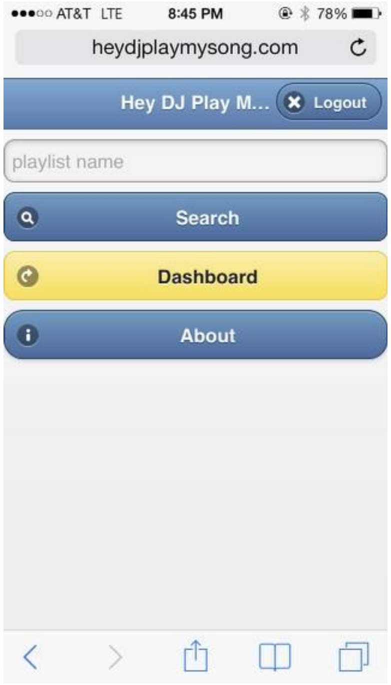


Figure 20

7.0 Enhancements

8.0 Bibliography

8.1 Books

Meloni, Julie C. *Sams Teach Yourself PHP, MySQL and Apache: All in One*. Indianapolis, IN: Sams Pub., 2012. Print.

Meloni, Julie C., and Matthew A. Telles. *PHP 6 Fast & Easy Web Development*. Australia: Course Technology PTR, Cengage Learning, 2007. Print.

Murach, Joel, and Ray Harris. *Murach's PHP and MySQL: Training & Reference*. [Fresno, Calif.]: Mike Murach & Associates, 2010. Print.

Ullman, Larry E. *PHP and MySQL for Dynamic Web Sites*. Berkeley, CA: Peachpit, 2012. Print.

8.2 Internet Sites

"Alias.io." *— How to Store Passwords Safely with PHP and MySQL*. N.p., n.d. Web. 19 Nov. 2013. <http://alias.io/2010/01/store-passwords-safely-with-php-and-mysql/>.

"Developing PHP/MySQL Webapps." *Developing PHP/MySQL Webapps*. N.p., n.d. Web. 21 Nov. 2013. <http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/php2_webapps.html>.

"Dynamic Voting System with JQuery, PHP and MySQL." *Programming and Designing Blog*. N.p., n.d. Web. 01 Dec. 2013. <http://www.w3bees.com/2013/09/voting-system-with-jquery-php-and-mysql.html>.

"Ircmaxell's Blog." *Seven Ways To Screw Up BCrypt*. N.p., n.d. Web. 21 Nov. 2013. <http://blog.ircmaxell.com/2012/12/seven-ways-to-screw-up-bcrypt.html>.

"MySQL :: The World's Most Popular Open Source Database." *MySQL :: The World's Most Popular Open Source Database*. N.p., n.d. Web. 19 Nov. 2013. <http://www.mysql.com/>.

"MySQL Tutorial." *MySQL Tutorial*. N.p., n.d. Web. 19 Nov. 2013. <http://www.mysqltutorial.org/>.

"PDO Tutorial for MySQL Developers." *Hashphp.org*. N.p., n.d. Web. 21 Nov. 2013. <http://wiki.hashphp.org/PDO_Tutorial_for_MySQL_Developers>.

"PHP Freaks." *PHP Freaks*. N.p., n.d. Web. 19 Nov. 2013. <http://forums.phpfreaks.com/>

"PHP: Hypertext Preprocessor." *PHP: Hypertext Preprocessor*. N.p., n.d. Web. 19 Nov. 2013.

<http://www.php.net/>.

"PHP Login Page Example." *PHP Login Page Example.* N.p., n.d. Web. 19 Nov. 2013. <http://www.9lessons.info/2009/09/php-login-page-example.html>

"Top Questions." *Stack Overflow*. N.p., n.d. Web. 19 Nov. 2013. <http://stackoverflow.com/>

"ZURB University." *JQuery Joyride Plugin*. N.p., n.d. Web. 02 Dec. 2013. <http://zurb.com/playground/jquery-joyride-feature-tour-plugin>.

9.0 Conclusion