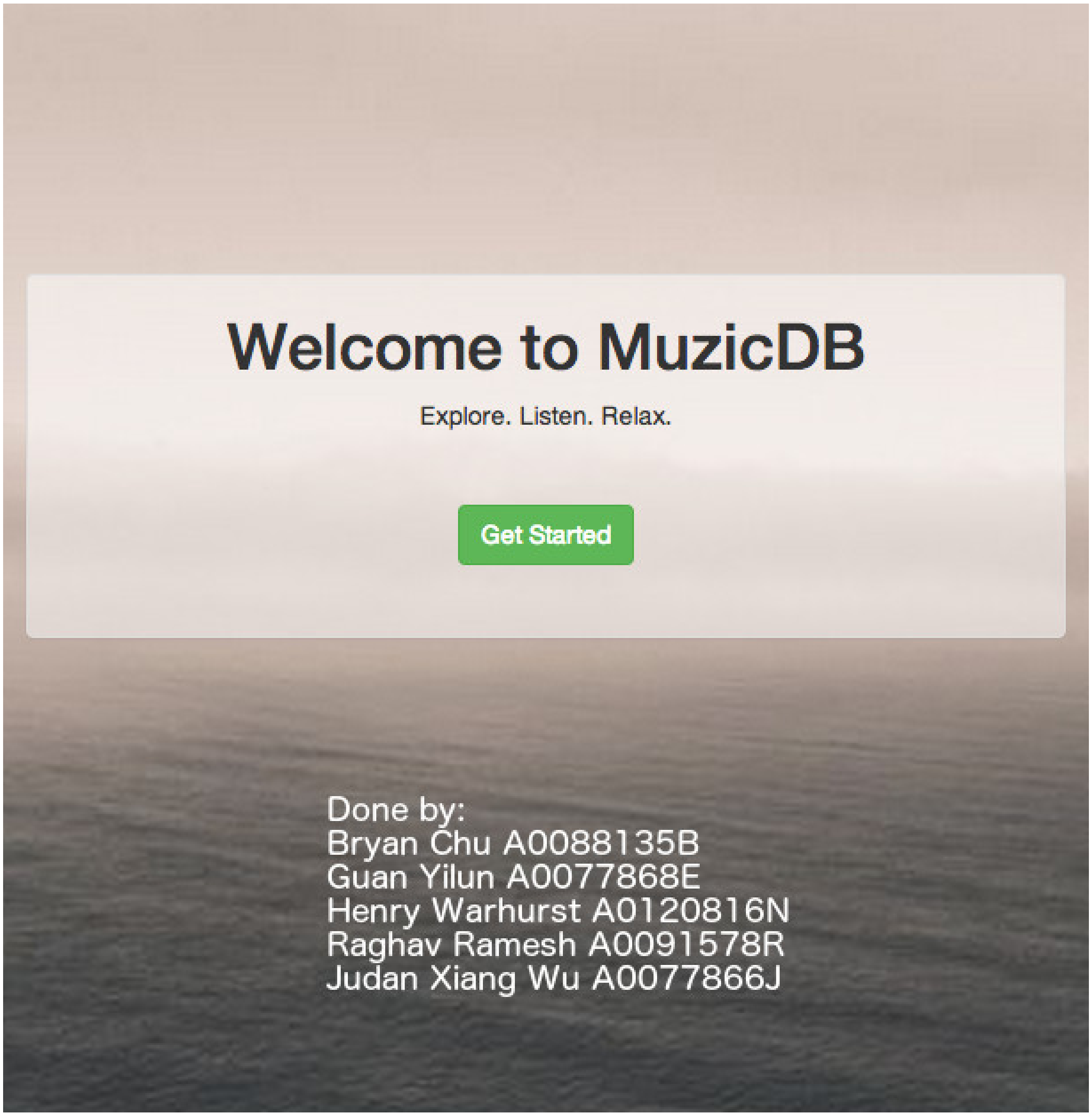


Welcome to MuzicDB

Explore. Listen. Relax.

Done by:
Bryan Chu A0088135B
Guan Yilun A0077868E
Henry Warhurst A0120816N
Raghav Ramesh A0091578R
Judan Xiang Wu A0077866J

1. Introduction

Our project aims to create an online music catalog where people can browse songs, albums and make purchases. The website can be found at muzicdb.hostei.com.

Our website is a user-based website. Registration is needed for all users. We allow two types of users: normal users and the administrator. Their privileges are summarized a table.

Normal User	Administrator
● Browse songs ● Browse albums ● Browse artists ● Purchase songs ● Purchase albums	● Browse songs ● Browse albums ● Browse artists ● Add/Remove/Update songs ● Add/Remove/Update albums ● Add/Remove/Update artists ● Add/Remove/Update users

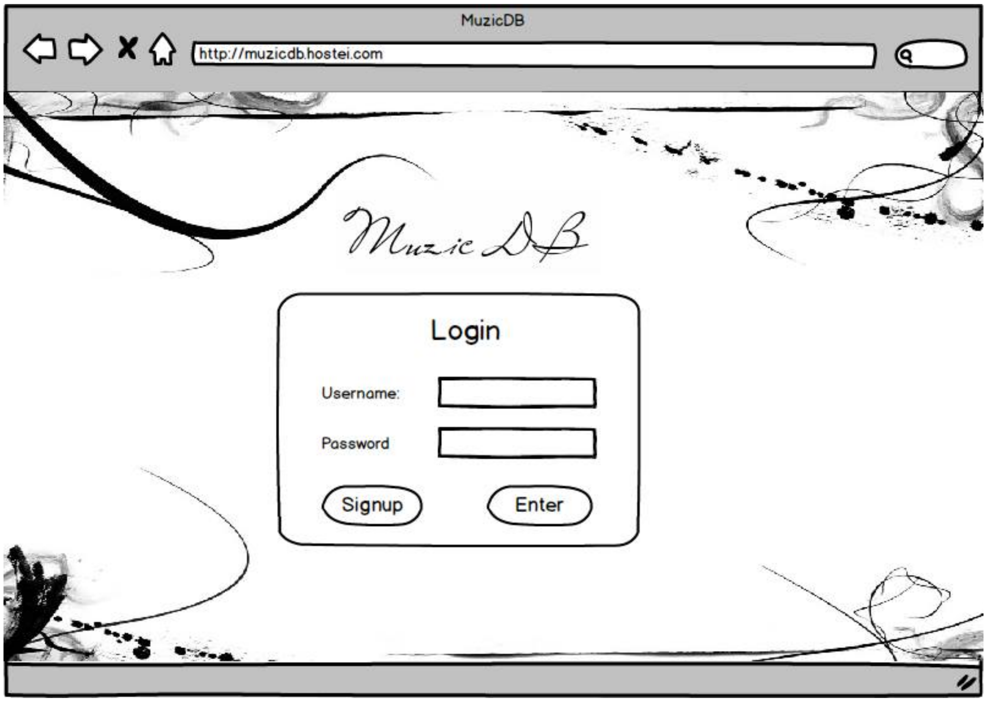
The technologies that we used:

Hosting Service:	000webhost.com
Server Languages	PHP
Database Management System (DBMS)	MySQL
Mock-up	Balsamiq

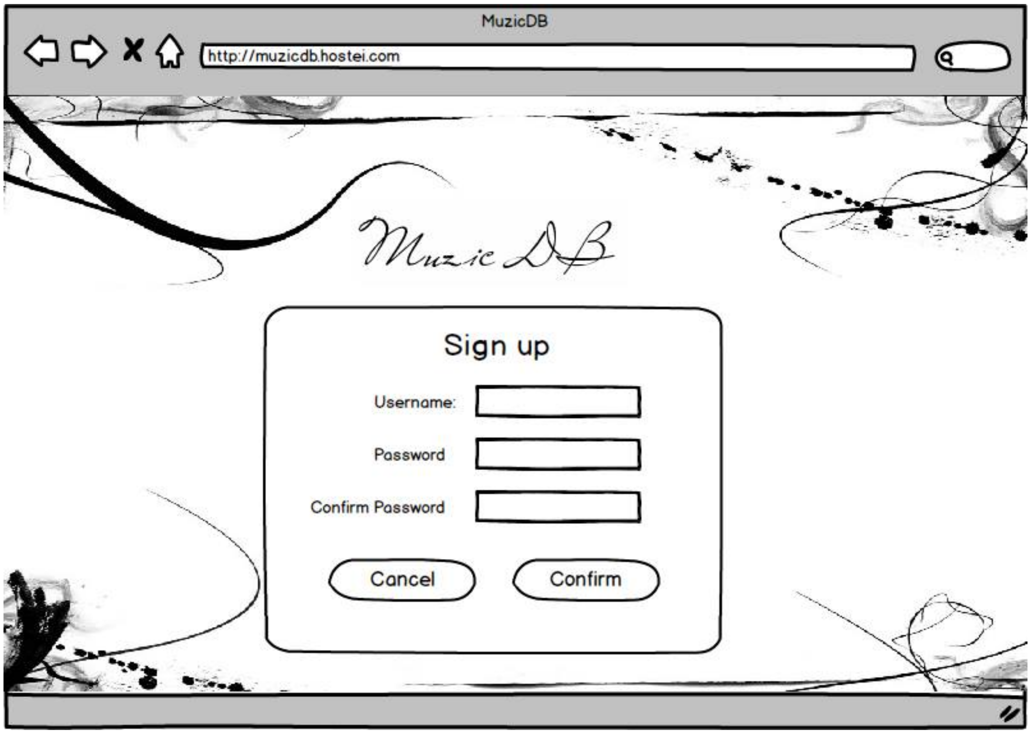
2. Initial Design

The initial design is made through mock-up website “Balsamiq”. They are shown below:

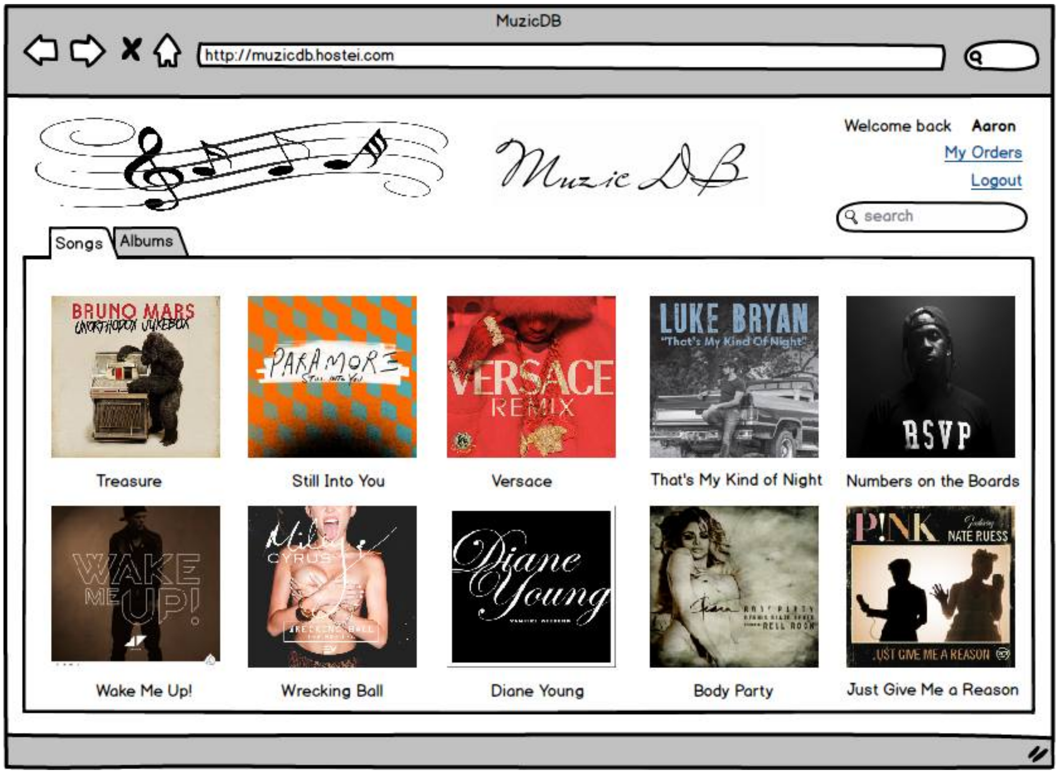
2.1 Login page



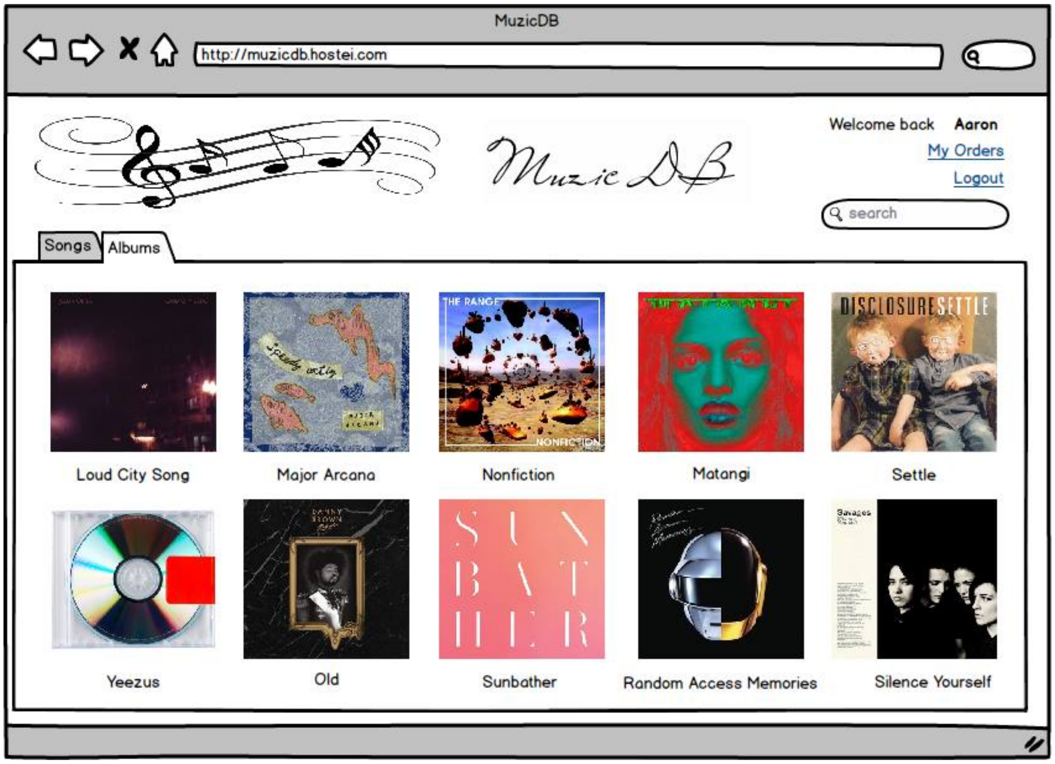
2.2 Sign up page



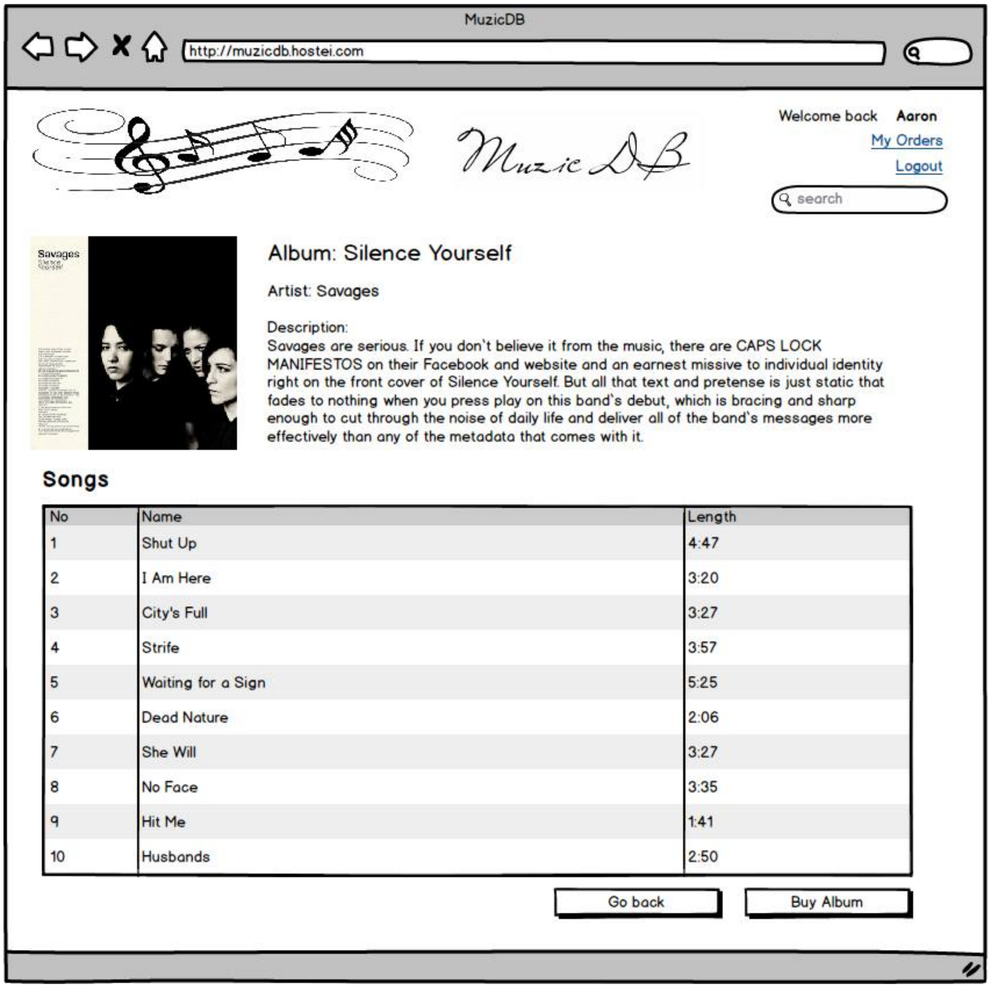
2.3 Songs page



2.4 Album page



2.5 Purchase album page



Album: Silence Yourself

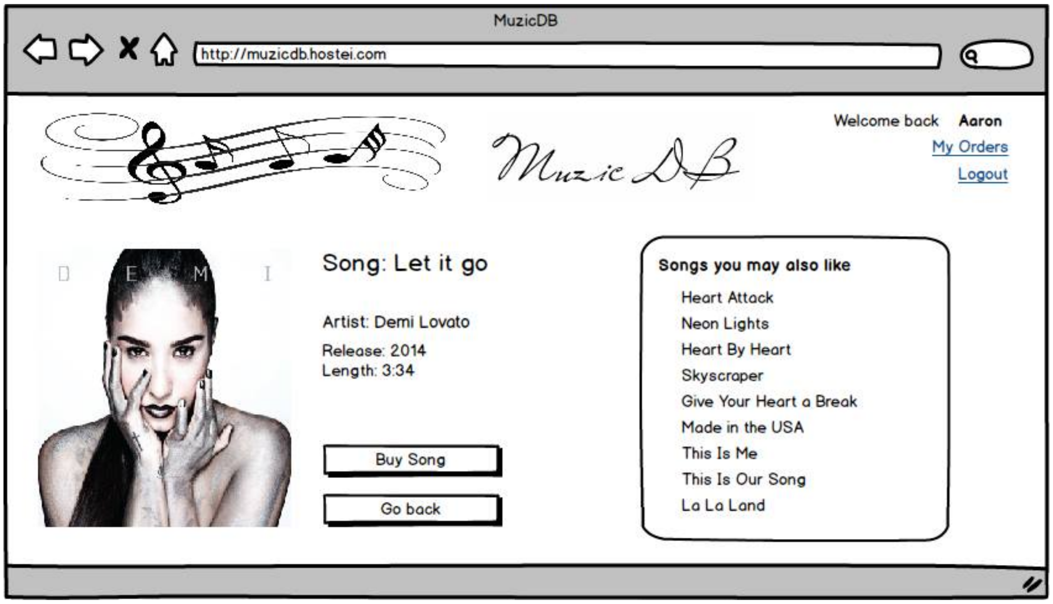
Artist: Savages

Description:
Savages are serious. If you don't believe it from the music, there are CAPS LOCK MANIFESTOS on their Facebook and website and an earnest missive to individual identity right on the front cover of Silence Yourself. But all that text and pretense is just static that fades to nothing when you press play on this band's debut, which is bracing and sharp enough to cut through the noise of daily life and deliver all of the band's messages more effectively than any of the metadata that comes with it.

Songs

No	Name	Length
1	Shut Up	4:47
2	I Am Here	3:20
3	City's Full	3:27
4	Strife	3:57
5	Waiting for a Sign	5:25
6	Dead Nature	2:06
7	She Will	3:27
8	No Face	3:35
9	Hit Me	1:41
10	Husbands	2:50

2.6 Purchase Song page



Song: Let it go

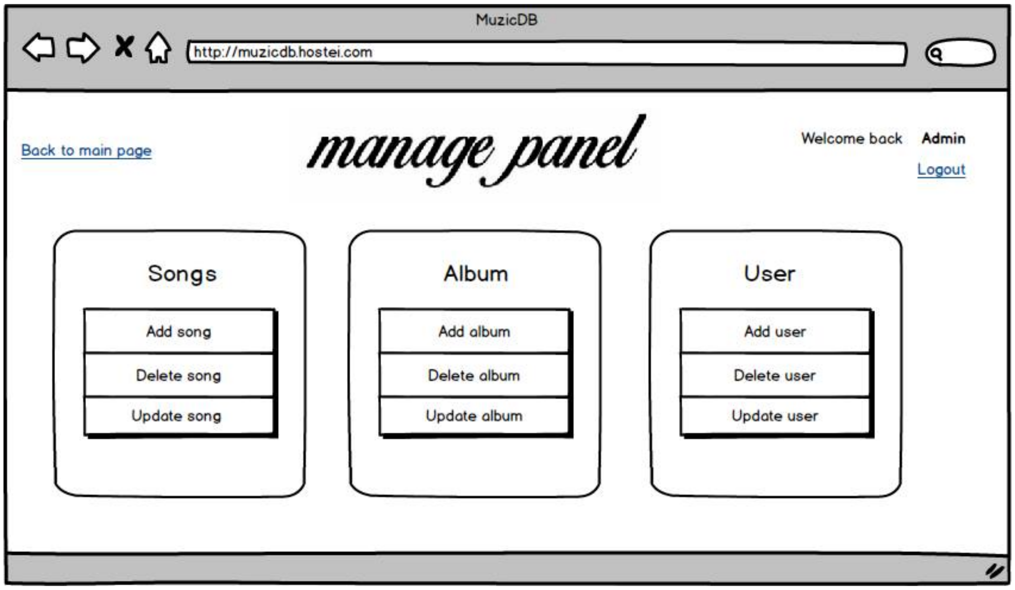
Artist: Demi Lovato

Release: 2014
Length: 3:34

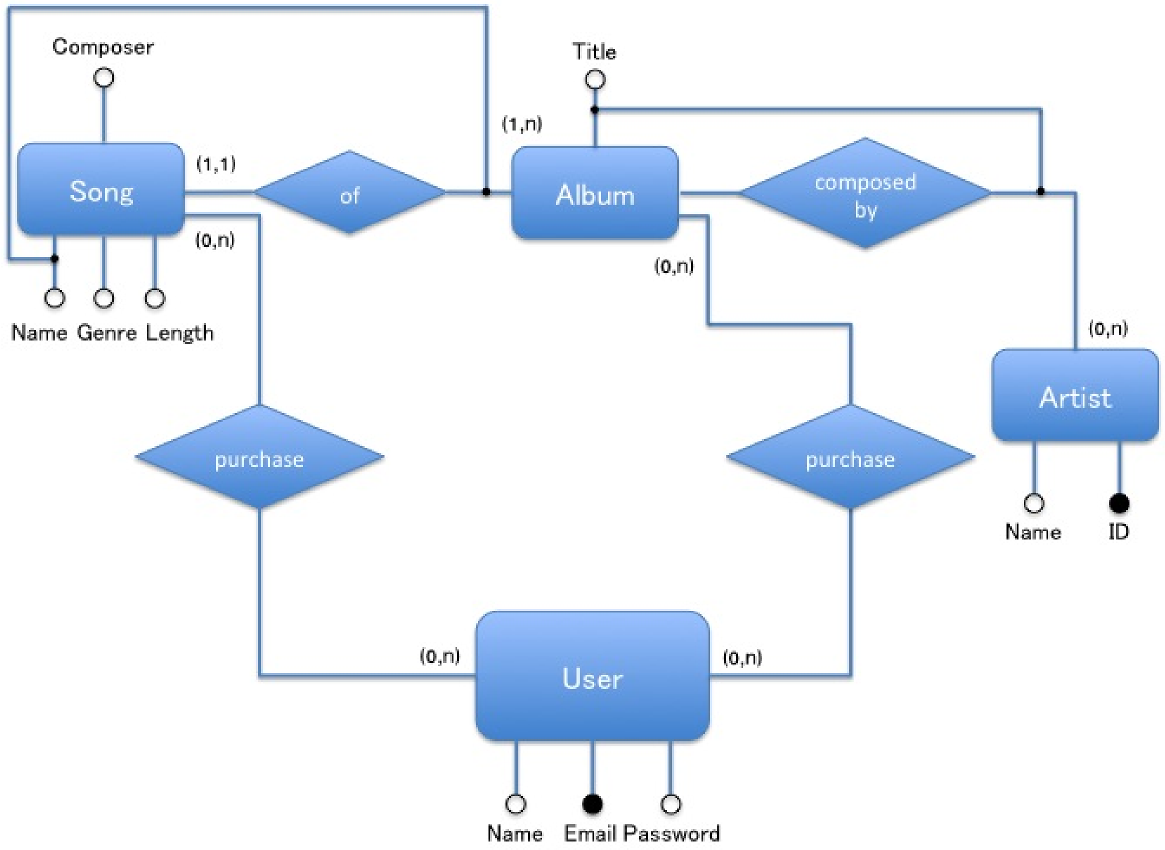
Songs you may also like

- Heart Attack
- Neon Lights
- Heart By Heart
- Skyscraper
- Give Your Heart a Break
- Made in the USA
- This Is Me
- This Is Our Song
- La La Land

2.7 Admin page



3. ER Diagram



Above is the ER diagram of our system. We have chosen to implement Song as a weak entity of Album, and Album as a weak entity of Artist.

4. Explanation of MuzicDB schema

SQL **DDL** Statement	Description

CREATE TABLE artist (
artistName VARCHAR(64) NOT NULL UNIQUE,
id VARCHAR(32) PRIMARY KEY
);
	Creates artist *entity, with artist's name and ID.*

CREATE TABLE album (title VARCHAR(64), artistID VARCHAR(32), PRIMARY KEY (title, artistID), FOREIGN KEY (artistID) REFERENCES artist(id) ON DELETE CASCADE ON UPDATE CASCADE);	***Creates album*** *entity, with album title and artist ID.* *Primary key is a composite key of album title and artistID.* *Upon deletion or update to artistID, the album will be deleted or updated correspondingly*
CREATE TABLE song (name VARCHAR(64), title VARCHAR(64), artistID VARCHAR(32), composer VARCHAR(64), genre VARCHAR(64), length INTEGER, PRIMARY KEY (name, title, artistID), FOREIGN KEY (title, artistID) REFERENCES album(title, artistID) ON DELETE CASCADE ON UPDATE CASCADE, FOREIGN KEY (artistID) REFERENCES artist(id) ON DELETE CASCADE ON UPDATE CASCADE);	***Creates song*** *entity, with song name, album title and artist ID.* *Other attributes include composer, genre and length. Length is represented by an integer that measures in seconds.* *Song has a foreign key that references album, as well as a foreign key that references artist ID.* *Deletion/updating of either album or artist will affect the song.*
CREATE TABLE user (name VARCHAR(64), UNIQUE email VARCHAR(64) PRIMARY KEY, password VARCHAR(32));	*Creates user entity. Contains username, password and email. Email is the primary key. Username must be unique.*
CREATE TABLE song_purchase (name VARCHAR(64), title VARCHAR(64), artistID VARCHAR(32), price FLOAT(5,2), email VARCHAR(64),	*Creates table representing a song purchase.* *Primary key is the composite of song primary key and email of the user who purchased the song.*

PRIMARY KEY (name, title, artistID, email), FOREIGN KEY (name, title, artistID) REFERENCES song(name, title, artistID) ON UPDATE CASCADE ON DELETE CASCADE, FOREIGN KEY (email) REFERENCES user(email) ON UPDATE CASCADE ON DELETE CASCADE);	*Also contains the float attribute price.* *Song purchase is deleted when user or song is deleted.* *Song purchase is updated if song primary key attributes or email is updated.*
CREATE TABLE album_purchase (title VARCHAR(64), artistID VARCHAR(32), price FLOAT(5,2), email VARCHAR(64), PRIMARY KEY (title, artistID, email), FOREIGN KEY (title, artistID) REFERENCES album(title, artistID) ON UPDATE CASCADE ON DELETE CASCADE, FOREIGN KEY (email) REFERENCES user(email) ON UPDATE CASCADE ON DELETE CASCADE);	*Creates table representing an album purchase.* *Primary key is the composite of album primary key and the email of the user who purchased the album.* *Also contains the float attribute price.* *Album purchase is deleted when user or album is deleted.* *Album purchase is updated if album title or email is updated.*

5. SQL **DML** statements

SQL **DML** Statement	Description

INSERT INTO artist(artistName, id) VALUES('Regina Spektor', '001');
INSERT INTO artist(artistName, id) VALUES('Capital Cities', '002');
INSERT INTO artist(artistName, id) VALUES('Tensnake', '003');
INSERT INTO artist(artistName, id) VALUES('2NE1', '004');
INSERT INTO artist(artistName, id) VALUES('Yiruma', '005');
INSERT INTO artist(artistName, id) VALUES('Skrillex', '006');
INSERT INTO artist(artistName, id) VALUES('Jon McLaughlin', '007');
INSERT INTO artist(artistName, id) VALUES('Fire Away', '008');
INSERT INTO artist(artistName, id) VALUES('Yngwie J. Malmsteen', '009');
	*The code snippet on the left illustrates the **insertion of artists** into the artist table.*
	Value pairs are <artist name, artist ID>.

DELETE FROM artist
WHERE artistName='Yiruma';
 	
DELETE FROM artist
WHERE artistName='Skrillex';
	The code snippet on the left illustrates the ***deletion of artist*** *from the artist table, by artist name.*
	Note that this removes associated songs and albums too.

UPDATE artist
SET artistName='SKRLX'
WHERE artistName='Skrillex';
	The code snippet on the left illustrates the ***editing of artist name****.*

INSERT INTO album(title, artistID)
VALUES('In a Tidal Wave of Mystery','002');
INSERT INTO album(title, artistID)
VALUES('Glow','003');
INSERT INTO album(title, artistID)
VALUES('Crush','004');
INSERT INTO album(title, artistID)
VALUES('Healing Piano','005');
INSERT INTO album(title, artistID)
VALUES('Stay In Memory','005');
INSERT INTO album(title, artistID)
VALUES('Recess','006');
INSERT INTO album(title, artistID)
VALUES('Holding My Breath','007');
INSERT INTO album(title, artistID)
VALUES('The Greatest Hits - EP - Single','008');
INSERT INTO album(title, artistID)
VALUES('Spellbound','009');
	The code snippet on the left illustrates the ***insertion of albums*** *into the album table.*
	
	Value pairs are <album title, artist ID>.
	
	Note: the artist entry has to be inserted into the table before an album can be assigned to that artist ID.

DELETE FROM album
WHERE EXISTS (
      SELECT *
      FROM artist
      WHERE artist.id = album.artistID
      AND artist.artistName = 'Yiruma')
AND album.title = 'Healing Piano';
	The code snippet on the left illustrates the ***deletion of an album*** *in the album table.*

UPDATE album al
SET al.title = 'new_name'
WHERE EXISTS (
      SELECT *
      FROM artist ar
      WHERE ar.artistName = '2NE1'
      AND ar.id = al.artistID)
AND al.title = 'Crush';
	The code snippet on the left illustrates the ***editing of an album*** *in the album table.*

INSERT INTO song(name, title, artistID, composer, genre, length)	*The code snippet on the left illustrates the* ***insertion of songs*** *into the song table.*

```
VALUES('I Sold My Bed, But Not My Stereo',
'In a Tidal Wave of Mystery', '002',
'Capital Cities', 'Alternative', 183);

INSERT INTO song(name, title, artistID,
composer, genre, length)
  VALUES('Come Back Home', 'Crush', '004',
'2NE1', 'Pop', 232);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('River Flows in You', 'Healing
Piano', '005', 'Yiruma', 'New Age', 184);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('May Be', 'Healing Piano', '005',
'Yiruma', 'New Age', 184);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('Silver Line', 'Stay In Memory',
'005', 'Yiruma', 'New Age', 194);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('Recess', 'Recess', '006',
'Skrillex', 'Dance', 170);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('Fire Away', 'Recess', '006',
'Skrillex', 'Dance', 200);

INSERT INTO song(name, title, artistID,
composer, genre, length)
VALUES('Electric Duet', 'Spellbound',
'009', 'Yngwie J. Malmsteen', 'Rock',
220);
```

Values are <song name, album title, artist ID, composer, genre, length>.

Note: the artist and album entry has to be inserted into their respective tables before a song can be assigned to that album and artist ID.

```
DELETE FROM song
WHERE EXISTS (
    SELECT *
    FROM artist, album
    WHERE artist.id = album.artistID
    AND album.title = song.title
    AND artist.artistName = 'Capital
Cities'
    AND album.title = 'In a Tidal Wave of
Mystery')
AND song.name = 'Kangaroo Court';
```

*The code snippet on the left illustrates the **deletion of a song** from the song table.*

UPDATE song s
SET s.name = 'Go Back Home'
WHERE s.name = 'Come Back Home';
	*The code snippet on the left illustrates the **editing of a song** in the song table.*

INSERT INTO user VALUES('John Crew',
'john@crew.com', 'bigjohncrew');
INSERT INTO user VALUES('John',
'john@j.com', 'bigjohn');
	*The code snippet on the left illustrates the **insertion of new users** into the user table.*
	Values are <user name, email, password>.

DELETE FROM user
WHERE email='john@crew.com';
	*The code snippet on the left illustrates the **deletion of users** in the user table.*

UPDATE user
SET email='johncrew@gmail.com'
WHERE email='john@crew.com';
	*The code snippet on the left illustrates the **editing of a user's email** address in the user table.*

6. Example Queries

Query	Description
SELECT s.title AS 'Album Name', a.artistName AS 'Artist' FROM song s, artist a WHERE UPPER(s.title) LIKE UPPER('%php_search_query%') #Note: the % signs are at the front and back for AND a.id = s.artistID # padding, so you can search for a string GROUP BY s.title, a.artistName;	To search for music by Album Name

SELECT s.name AS 'Song Title',
a.artistName AS 'Artist', s.title AS
'Album Name',
       s.composer, s.genre, s.length
FROM song s, artist a
WHERE UPPER(s.name) LIKE
UPPER('%php_search_query%')
      AND a.id = s.artistID;
	To search for music by song title.

SELECT a.artistName AS 'Artist'
FROM artist a
WHERE UPPER(a.artistName) LIKE
UPPER('%ns%')
GROUP BY a.artistName, a.id;
	To search music by artist name.

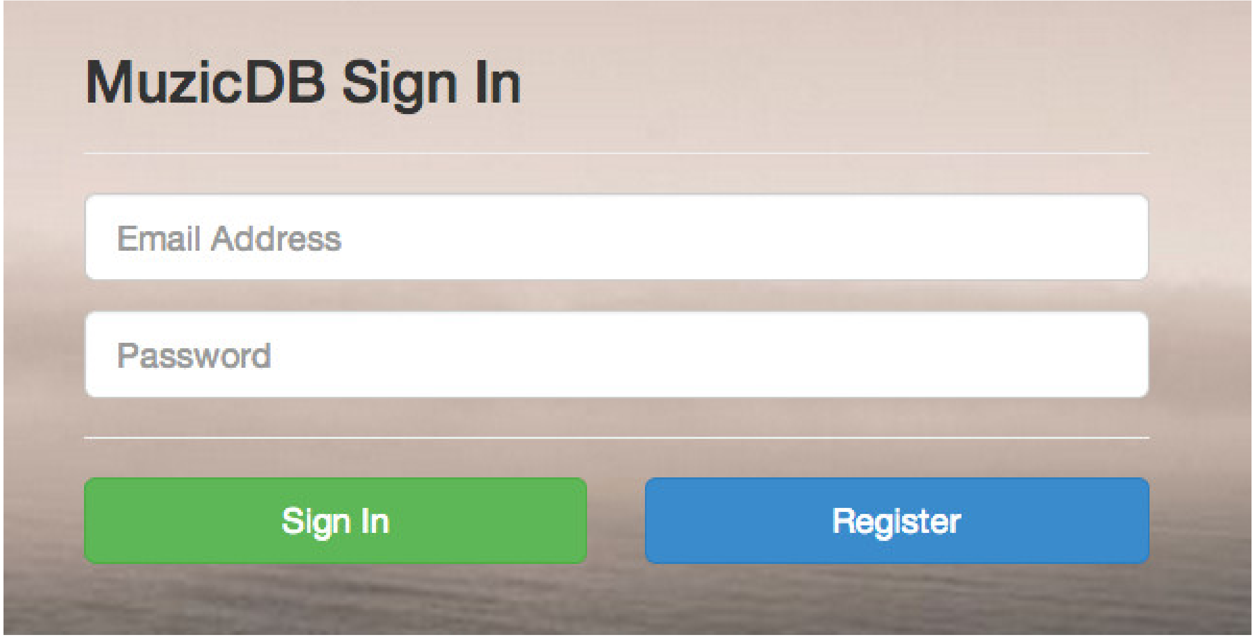
`SELECT s.name AS 'Song Title',`
`a.artistName AS 'Artist', s.title AS`
`'Album Name',`
`       s.composer, s.genre, s.length`
`FROM song s, artist a`
`WHERE UPPER(s.composer) LIKE`
`UPPER('%php_search_query%')`
`      AND a.id = s.artistID;`
	To search music by composer name.

`SELECT s.name AS 'Song Title',`
`a.artistName AS 'Artist', s.title AS`
`'Album Name',`
`       s.composer, s.genre, s.length`
`FROM song s, artist a`
`WHERE UPPER(s.genre) LIKE`
`UPPER('%php_search_query%')`
`      AND a.id = s.artistID;`
	To search music by genre.

`SELECT s.name AS 'Song Title',`
`a.artistName AS 'Artist', s.title AS`
`'Album Name',`
`       s.composer, s.genre, s.length`
`FROM song s, artist a`
`WHERE a.id = s.artistID;`
	To find all songs.

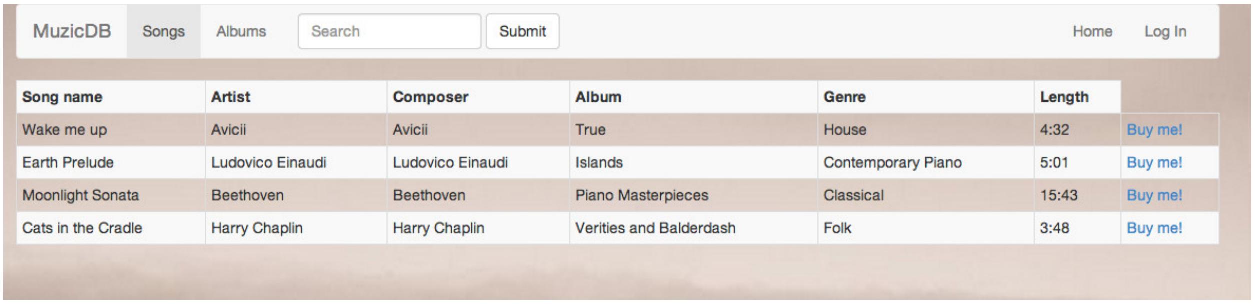
`SELECT s.title AS 'Album Name',`
`a.artistName AS 'Artist'`
`FROM song s, artist a`
`WHERE a.id = s.artistID`
`GROUP BY s.title, a.artistName;`
	To find all albums.

7. Screenshots of our website



MuzicDB Sign In

Main Interface



Song name	Artist	Composer	Album	Genre	Length	
Wake me up	Avicii	Avicii	True	House	4:32	Buy me!
Earth Prelude	Ludovico Einaudi	Ludovico Einaudi	Islands	Contemporary Piano	5:01	Buy me!
Moonlight Sonata	Beethoven	Beethoven	Piano Masterpieces	Classical	15:43	Buy me!
Cats in the Cradle	Harry Chaplin	Harry Chaplin	Verities and Balderdash	Folk	3:48	Buy me!

8. Learning points/Conclusions

Through this project we have gained practical knowledge of building a website from scratch. We also learnt how to utilize the database knowledge taught in class to solve a real world problem and how to collaborate as a team.

Another lesson learnt was the importance of proper work allocation. Some of us handled the database aspect of the project, while some handled web or GUI mockups. We also learnt how to use new technologies not taught in class as none of us had experience with php prior to this.