Lee Healy X00120179  
Conor Griffin X00111602  
Imor Hartnett X00118631

Music FX

2nd Year Project

Table of Contents

[Introduction 3](#_Toc449382755)

[User Requirements 4](#_Toc449382756)

[Key User Interview 4](#_Toc449382757)

[Survey Results 7](#_Toc449382758)

[Summary Of Results 11](#_Toc449382759)

[System Narrative 12](#_Toc449382760)

[Front End 12](#_Toc449382761)

[Online Basket Checkout System 12](#_Toc449382762)

[Stock Control System 12](#_Toc449382763)

[Use Case 13](#_Toc449382764)

[Model 13](#_Toc449382765)

[Descriptions 14](#_Toc449382766)

[Class Diagram 20](#_Toc449382767)

[Pseudocode 21](#_Toc449382768)

[Register 21](#_Toc449382769)

[Login 21](#_Toc449382770)

[Remove Stock 22](#_Toc449382771)

[Place Order 22](#_Toc449382772)

[Pay Order 23](#_Toc449382773)

[Check Stock 24](#_Toc449382774)

[Resupply Stock 24](#_Toc449382775)

[Edit Stock 25](#_Toc449382776)

[Website Storyboard 26](#_Toc449382777)

[Input Design 26](#_Toc449382778)

[Customer Draft Input Screens 26](#_Toc449382779)

[27](#_Toc449382780)

[28](#_Toc449382781)

[Administrator Input Screens 29](#_Toc449382782)

[Output Design 30](#_Toc449382783)

[Website Screenshots 33](#_Toc449382784)

[Home Page 33](#_Toc449382785)

[Register Page 34](#_Toc449382786)

[Login Page 34](#_Toc449382787)

[Products Page 35](#_Toc449382788)

[35](#_Toc449382789)

[Shopping Cart 35](#_Toc449382790)

[Previous Orders 36](#_Toc449382791)

[Administrator Products View 36](#_Toc449382792)

[36](#_Toc449382793)

[Database Design 37](#_Toc449382794)

[ERD – Entity Relationship Diagram 37](#_Toc449382795)

[Table/Record Layouts 38](#_Toc449382796)

[SQL 40](#_Toc449382797)

[Test Case Matrix 42](#_Toc449382798)

[User Manual 45](#_Toc449382799)

[Conclusion 46](#_Toc449382800)

# Introduction

This document is the system documentation for the MusicFX website. It contains all requirements for the site as well as design aspects. The back end of the site was created first, which contained the database and minimal design integration. The basic functionality of the back end was made up of many CRUD functions. Since this is an online shop, we had the common CRUD functions most e-commerce sites would have which would enable administrators to run the site efficiently and effectively. It also has the functions which would enable customers to easily buy products without any hassle or confusion.

After the basic functionality was implemented to the back end of the site, the goal was then to design a site that looked easy on the eye and also welcoming. The design should be able to compete with any other competition online, especially with today’s Bootstrap functionality which can be easily incorporated to HTML and CSS. We decided we would take advantage of Bootstrap and its many commands to make our site look sleek and professional.

Before any development of the site could be done, we needed to find out what requirements we would need for this type of site so we done some research and asked around at some local businesses and also asking colleagues to take a survey.

# User Requirements

Here are the results we got from our requirement research.

## Key User Interview

We undertook interviews about stock systems with the stock administrators of different companies.

**Key User, Interview with Alan McNevin, Stock Administrator at RSL Ireland.**

1. What stock do you sell in your store?

**We are a refrigeration parts wholesaler so we sell items such as, copper coils, copper lengths, leak detectors, compressors, fan motors, condensing units.**

1. Is there a good stock control system implemented in the store?

**There is, everything bought is invoiced and taken off the stock showing on the computer.**

1. Does the system notify staff when stock for certain items is low?

**It does, a small warning symbol appears beside the item.**

1. Is there a way to resupply stock through this stock control system?

**Yes, all the supplier’s details are in with the stock so when it gets low we can easily get the suppliers email or phone number.**

1. If yes, how easy is this system to use?

**Very easy and handy to use.**

1. Is there a login required to manipulate stock?

**Yes there is and we all have different levels of restriction on what we can do**

1. Is there many users on this system? (All staff or managers etc.?)

**Yes about 12-15 users but there is a limit to how many staff can be on the system all at once.**

1. Who of these are able to manipulate stock? (Resupply, edit, add and delete stock?)

**Only staff in the stores can manipulate stock, no one in the office can.**

1. Would you recommend this system for companies to use?

**100%. It’s very effort and time efficient and helps a lot in the store.**

**Key User, Interview with David Moore, Duty Manager at Supervalu Walkinstown.**

1. What stock do you sell in your store?

**We are a retail store called Supervalu Walkinstown, we are Irelands largest grocery and food distributors in Ireland.**

1. Is there a good stock control system implemented in the store?

**Yes, I think our system is good , we use the Musgrave stock control system known as GOLD to keep track of all stock that we buy in and sell. This system is used so that we can keep track of stock flow and everything we stock is then transferred to the computer.**

1. Does the system notify staff when stock for certain items is low?

**It doesn’t quite do that but the system will show us what stock was ordered the previous week and what wasn’t, we can then tell if we need to order more of a certain item of stock or if we have enough of.**

1. Is there a way to resupply stock through this stock control system?

**Yes there is, we scan all the barcodes of stock that is low and this is immediately transferred to the system and all the supplier’s details are in with the stock so when the items are scanned an order is then put through for that item to be resupplied.**

1. If yes, how easy is this system to use?

**Yes it is easy to use, our stock control system does most of the work itself making it a very easy system to use.**

1. Is there a login required to manipulate stock?

**Yes there is a code you must put into each device when logging into the system.**

1. Is there many users on this system? (All staff or managers etc.?)

**It depends we have around 15 portable devices that we use to scan our products if they are low on stock, any staff member can use the system as long as they are authorized by a manager to do so.**

1. Who of these are able to manipulate stock? (Resupply, edit, add and delete stock?)

**All of our staff in the stores can manipulate stock as long as they have been authorized by a manager in the store to do so.**

1. Would you recommend this system for companies to use?

**Yes I definitely I would, It’s easy to use and does most of the work for you.**

**Key User, Interview with Tomasz Maszdur, Manager at Starbucks, Stephens Green.**

1. What stock do you sell in your store?

**We are a coffee shop franchise based in Dublin city centre working as part of the Starbucks Corporation. Our stock sold is mainly comprised of Starbucks branded products like our selection of Arabica coffee and sandwiches / wraps. We also sell outsourced fresh produce like our pastries which are delivered every morning.**

1. Is there a good stock control system implemented in the music store?

**Yes, but for the system to work it needs the supervisor on shift at the end of the week to spend the day taking stock of everything behind the bar and downstairs in the stock room.**

1. Does the system notify staff when stock for certain items was low?

**No there is no automated system the staff must count all stock to find out what produce is low on stock.**

1. Is there a way to resupply stock through this stock control system?

**Yes all our suppliers are entered on the company’s computer so that the manager has direct contact with them.**

1. If yes, how easy is this system to use?

**The system still utilises an emailing process which means the manager must write individual emails to our suppliers which can be a time consuming process.**

1. Is there a login required to manipulate stock?

**Yes but it is very low level security. Everything is controlled by the company’s computer which is locked by login of the user.**

1. Is there many users on this system? (All staff or managers etc?)

**There is only one single profile on the system but this profile is used by seven people. One manager and 6 shift supervisors.**

1. Who of these are able to manipulate stock? (resupply, edit, add and delete stock?)

**All supervisors and the manager are able to manipulate the stock. Barista’s do not have any access to the stock control system.**

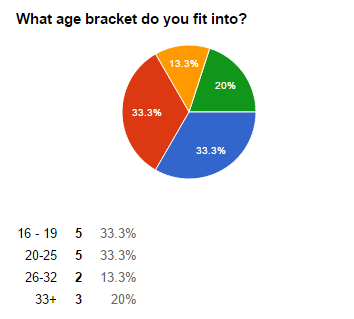
1. Would you recommend this system for companies to use?

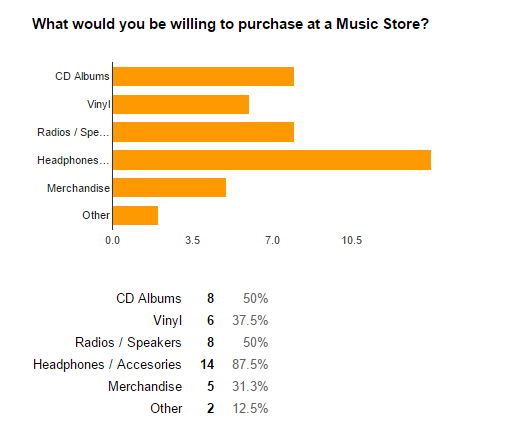
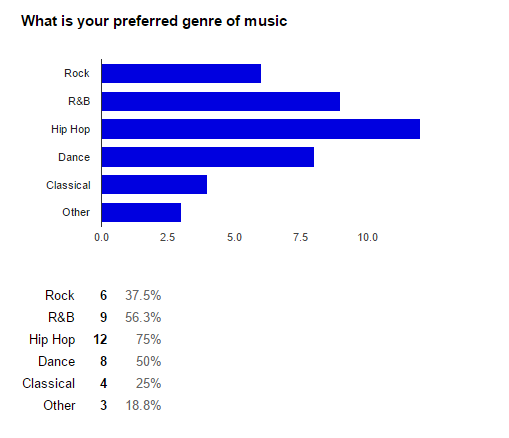
**No as it is not automated enough and relies on lots of staff involvement.**

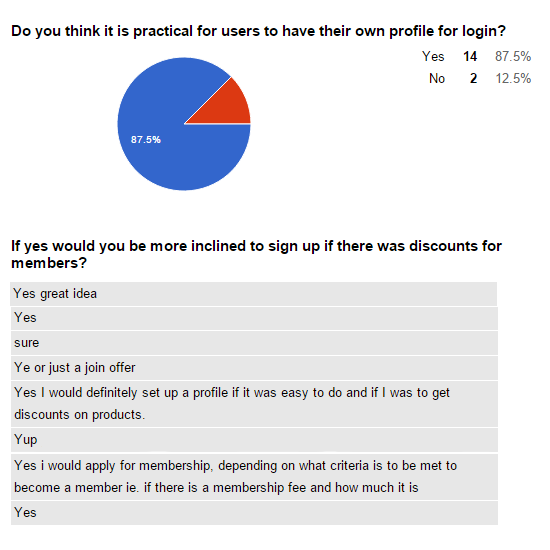
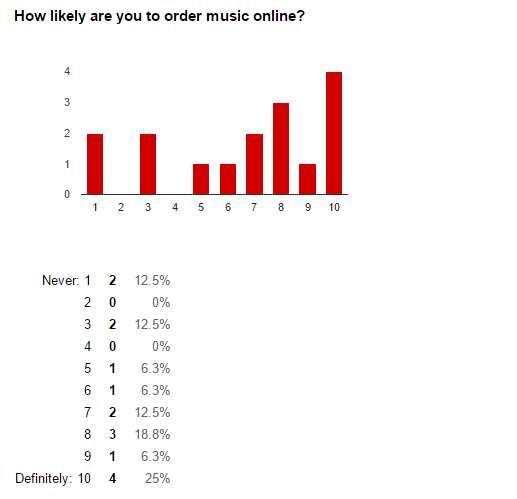
## Survey Results

We also put up a survey online to get some feedback from others on e-commerce websites.

The following are the results from a recent survey we took to see what requirements we would need for our project:







Summary Of Results  
From the results above we established that people from all ages are willing to buy music and music accessories online. So for our Music Store Stock System we decided we should sell a broad range of items from CD’s, Vinyl’s and accessories such as headphones and speakers. Then all 3 of us, went through many different websites and looked at their structure and saw how they worked after which we decided how our own site and system would work and we then came up with a System Narrative.

# 

# System Narrative

Our site is a music site and offers CD’s, Vinyl’s and other music accessories to purchasing customers. All items for sale are physical items and not to be confused with digital online downloads of music such as iTunes and Beatport etc.

## Front End

The customer goes onto the website and the first thing they are presented with is the index page. This page tells the customer a little bit about what we do and what we have for sale. It also includes a button which brings the customer to an online catalogue of items we have for sale. After viewing the index page the customer can then either click the online catalogue button or click any of the other buttons in the navigation bar which includes another catalogue button and other buttons such as contact us and about us.

## Online Basket Checkout System

If the customer would like to buy items from the catalogue whether it be CD’s, Vinyl Record’s or any type of accessory, they would click on the item they would like to buy and it would go into their basket/online shopping cart. This cart would contain all the items they would like to buy much like a real life physical shopping cart/basket. When the customer goes into their online basket it will bring them to a page where they could check out their items. The grand total of all the items would be displayed to the customer and a link to take them to their payment will also be shown on their display.

After the item(s) have been purchased by the customer, they will then be subtracted from their respectful stock amount. For example if there were 10 CD’s and the customer buys 2 CD’s, there will now be 8 CD’s left in stock. If the stock of the items reaches a certain low point the Stock Administrator (Manager) will receive an automatically sent warning about it from the website and then he/she will decide whether the stock is worth resupplying. If the stock is worth resupplying the Manager will send an email to the company’s supplier detailing what stock they need and the quantity of each stock to go with it.

## Stock Control System

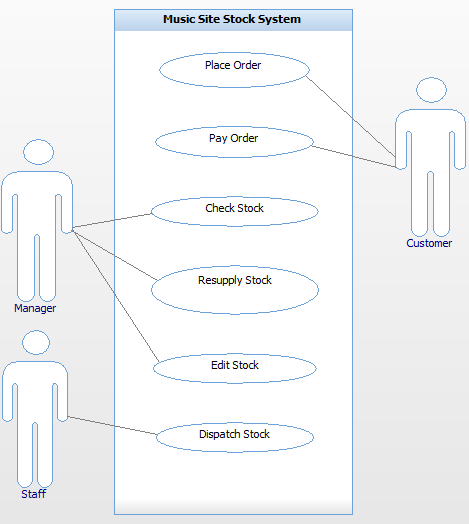
The Stock Administrator will have exclusive access to edit the stock on the system. They will be able to add or delete stock and also edit the name and description. Any other staff will not have these privileges because it could lead to confusion with multiple people manipulating the stock to their liking.

While the system works, in the background on the physical end there is a warehouse which holds all of our stock. The staff work in this warehouse and get orders ready as they come in. It is these members of staff that will send frequent updates of stock levels to the Stock Administrator. When a transaction happens on the front end of our system staff working in the warehouse will be notified what produce has been orders and where it is located in the warehouse. When the stock is located the staff package, label each one with their destination address and get logistics in at the end of the day to deliver all the orders.

# Use Case

The following is the use case model and use case descriptions from our research into the site.

## Model



## Descriptions

|  |  |
| --- | --- |
| Use Case: | **Place Order** |
| Actor(s): | Customer |
| Goal: | For the customer to place an order and the staff to take the order |
| Overview: | The customer enters the shop, proceeds to the counter and places an order. The staff member then takes the customer’s order. |
| Pre-  Condition: | The customer has not yet placed an order. |
| Post-  Condition: | The customer’s order has been taken by the staff member. |
| Successful  Scenario: | Step1: Customer goes onto the online shop website  Step2: Customer browses the site before purchasing  Step3: Website awaits order  Step4: Customer gives the site his/her order  Step5: Website takes the order and enters it into the system. |
| Alternative  Scenario(s): | Website is down  Stock is unavailable to customer  System is down  Connection is lost |

|  |  |
| --- | --- |
| Use Case: | **Pay Order** |
| Actor(s): | Customer |
| Goal: | For the customer to pay for his order |
| Overview: | The customer now wants to pay for the order he/she has previously placed on the website. |
| Pre-  Condition: | The customer has not yet paid for the order. |
| Post-  Condition: | The customer has paid for his/her item. |
| Successful  Scenario: | Step1: Customer proceeds to checkout after placing their order.  Step2: The site displays the total to the customer.  Step3: The site asks the customer to enter their credit card details  Step4: Customer pays by card on the website. |
| Alternative  Scenario(s): | Website is down.  Total price is calculated wrong by the site.  The customer doesn’t have a credit card  Customer has insufficient funds on their credit card. |

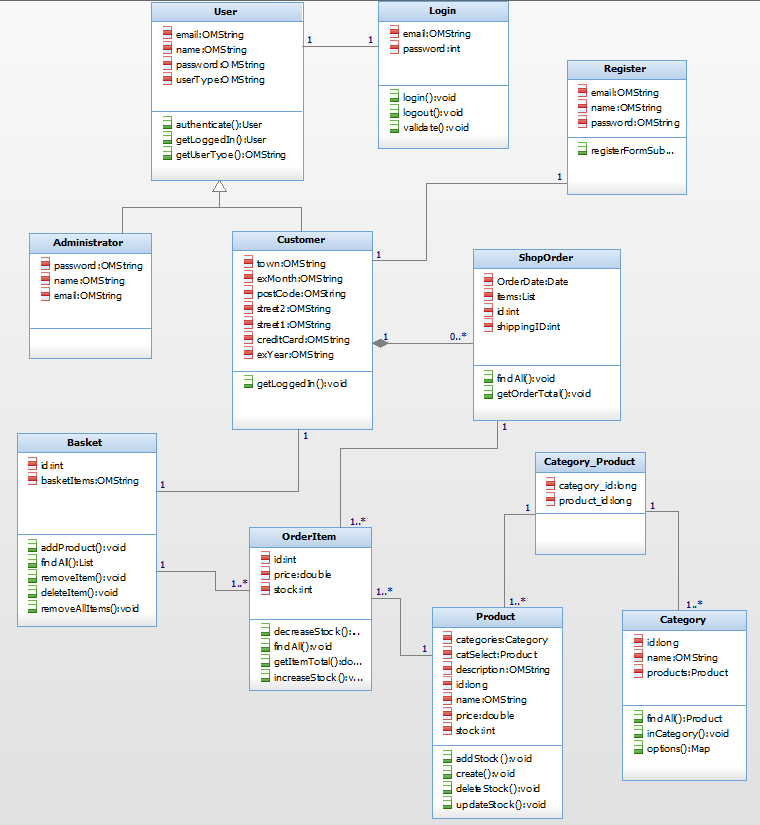
|  |  |
| --- | --- |
| Use Case: | **Dispatch Stock** |
| Actor(s): | Staff |
| Goal: | To export the physical stock from the stockroomto to the customer. |
| Overview: | When stock has been ordered and payment is made the staff member is responsible for physically exporting that item to the customer. |
| Pre-  Condition: | Payment has been made and stock is ready for exportation. |
| Post-  Condition: | The stock has been successfully exported to the customer. |
| Successful  Scenario: | Step1: Staff member is informed that stock has been ordered and payment has been made.  Step2: Stock is prepared by staff for exportation.  Step3: Stock is prepared for physical transportation.  Step4: Stock is exported. |
| Alternative  Scenario(s): |  |

|  |  |
| --- | --- |
| Use Case: | **Check Stock** |
| Actor(s): | Manager |
| Goal: | To look at stock levels and decide whether a certain item of stock is needed. |
| Overview: | The website looks at what items have been sold and if an item goes below a certain level of stock then the system informs the manager that stock is needed. |
| Pre-  Condition: | Stock has not been checked yet. |
| Post-  Condition: | Stock is checked and manager is informed whether certain stock is needed. |
| Successful  Scenario: | Step 1: Stock has not been checked yet.  Step 2: System checks stock when item is ordered.  Step 3: Manager is free to check stock whenever he/she wants.  Step 4: If an item is needed the manager is informed.  Step 5: Stock has been checked. |
| Alternative  Scenario(s): | Stock has not been checked by the manager, and is not ordered. |

|  |  |
| --- | --- |
| Use Case: | **Re-supply Stock** |
| Actor(s): | Manager |
| Goal: | Ordering more stock to be delivered to the warehouse |
| Overview: | Manager re-supplies stock to the store that is selling well. Manager may want to order is seasonal stock as well coming up to different calendar events. |
| Pre-  Condition: | The manager judges how much stock will be needed for the next week and places an order to keep stock levels up for the upcoming week. |
| Post-  Condition: | Stock is ordered before the stock in store in depleted. Stock is then delivered before it is needed. |
| Successful  Scenario: | Stock is delivered before the stores stock is depleted. This enables the store to be able to make the pitched sales figures for the following week. |
| Alternative  Scenario(s): | The stock re-supply is not ordered in time for a sales rush on the weekend coming up to Christmas. This puts the store under pressure to make the pitched sales for the week. |

|  |  |
| --- | --- |
| Use Case: | **Edit Stock** |
| Actor(s): | Manager |
| Goal: | To be able to edit stock (stock item names, descriptions, prices, delete stock) |
| Overview: | The manager edits certain items like the names and descriptions of items and also the changes the prices of certain items |
| Pre-  Condition: | Stock items have not been edited |
| Post-  Condition: | Stock items have been edited |
| Successful  Scenario: | Step1: Managers is on the stock database.  Step2: Manager sees a typo in one of the stock’s description, name or price.  Step3: The manager clicks on the edit button.  Step4: The manager corrects the error in the name, description or price.  Step5: All changes are saved by the manager. |
| Alternative  Scenario | Stock database is down |

# Class Diagram



# Pseudocode

## Register

*//Customer would like to register to our site.*

**Inputs: (Name: Type:)**

            Name :**String**

        Email :**String**

        ConfirmEmail :**String**

        Password :**String**

        ConfirmPassword :**String**

**Begin:**

**OUTPUT** (Please enter your name)

**INPUT** (Name)

**OUTPUT** (Please enter your email)

**INPUT** (Email)

**OUTPUT** (Please confirm your email)

**INPUT** (ConfirmEmail)

**OUTPUT** (Please enter your password)

**INPUT** (Password)

**OUTPUT** (Please confirm your password)

**INPUT** (ConfirmPassword)

*//When ‘****Register’*** *is clicked, user is redirected to* ***‘Login’*** *page*

**End.**

## Login

**Inputs: (Name: Type:)**

Username :**String**

Password :**String**

**Begin:**

**IF** (Username **==** EnteredUsername **&&** Password **==** EnteredPassword) **THEN**

Login Successful

**ELSE**

Login Failed.

**ENDIF**

**End.**

## Remove Stock

*//Customer buys their items and the items are removed from stock*

**Inputs: (Name: Type:)**

stockAmount :**int**

itemsAmount :**int**

stockOnHand :**int**

**Begin:**

**OUTPUT**   (How many would you like to purchase?)

**INPUT**       itemsAmount

stockAmount -= itemsAmount

**IF**(itemsAmount **>** stockOnHand)

**OUTPUT** (to customer) (Sorry, we do not have enough stock)

**ELSE**

Stock removed successfully.

**END IF**

**End.**

## Place Order

*//Customer goes online and views our products*

*//Customer sees a product they like and adds it to the basket*

*//The basket fills up with items that the customer wants*

*//The Customer then proceeds with this basket and makes the order*

**Inputs: (Name: Type:)**

basket :**ArrayList** *//To hold objects (products) that the customer wants*

**Begin:**

**IF** (basket == null) **THEN**

**OUTPUT** Error: Cannot proceed with empty basket

**ELSE IF** (basket != null) **THEN**

**OUTPUT** Proceeding with items…

**End.**

*//All item prices will be added up to a total price and displayed to the Customer*

## Pay Order

*//The total price for all items in the basket is displayed to the customer and they are then asked to enter their card details.*

**Inputs: (Name: Type:)**

Street1         :**String**

        Street2         :**String**

        Town            :**String**

        PostCode     :**String**

cardNum      :**int**

        cardExMonth  :**int** *//Number representation of the month e.g. feb = 2*

        cardExYear :**int** *//Number representation of the year*

**Outputs: (Name: Type)**

totalPrice :**double**

**OUTPUT** (Total price for all items: **totalPrice**)

**OUTPUT** (Please enter your street1 :)

**INPUT** (street1)

**OUTPUT** (Please enter your street2 :)

**INPUT** (street2)

**OUTPUT** (Please enter your town :)

**INPUT** (town)

**OUTPUT** (Please enter your post code:)

**INPUT** (postcode)

**OUTPUT** (Please enter your card number:)

**INPUT** (cardNum)

**OUTPUT** (Please enter your card’s expiry date:)

**INPUT** (cardExMonth) **INPUT** (cardExYear)

**OUTPUT** (Validating card details..)

## Check Stock

***//****If the stock of an item is below 5 then warn the Stock Administrator that it is low, if it is not low //the amount of items left should be displayed to the Stock Administrator*

**Outputs: (Name: Type:)**

certainItemStock :**Object**

amount :**int**   *//Amount of stock of an item (e.g. 7 speakers)*

**Begin:**

**IF** (certainItemStock < 5) **THEN**

**OUTPUT** (Warning: Stock for **certainItemStock** is low!)

**OUTPUT** (There are **amount** left of **certainItemStock**)

**ELSE IF** (certainItemStock > 5) **THEN**

**OUTPUT** There are **amount** left of **certainItemStock**

**End.**

## Resupply Stock

*//Stock Administrator emails the Supplier an email detailing what stock the store needs.*

**Outputs: (Name: Type:)**

stockItem :Object

        supplier :String         *//Suppliers name*

        supplierEmail :String   *//Supplier’s email*

**Begin**

**OUTPUT** (Stock level for **stockItem** is low)

**OUTPUT** (The supplier for this product is: **supplier**)

**OUTPUT** (To resupply stock, email the supplier at: **supplierEmail**)

*//Stock Administrator then Emails the supplier asking for more stock.*

**End.**

## Edit Stock

***//****Stock Administrator goes on and edits the products in the stock.*

*//The Stock Administrator can edit the ID, Name, Description and Price of stock.*

**Inputs: (Name: Type:)**

newProductID :String

        newProductName :String

        newProductDescription :String

        newProductPrice :String

**Outputs: (Name: Type:)**

productID :int

productName :String

productDescription :String

productCategory :String

productStock :int

        productPrice :double

**Begin:**

***//****Displayed in a form*

**OUTPUT** (productID)

**INPUT** (newProductID)

**OUTPUT** (productName)

**INPUT** (newProductName)

**OUTPUT** (productDescription)

**INPUT** (newProductDescription)

**OUTPUT** (productCategory)

**INPUT** (newProductCategory)

**OUTPUT** (productStock)

**INPUT** (newProductStock)

**OUTPUT** (productPrice)

**INPUT** (newProductPrice)

        //*When ‘****Save****’**is clicked*

*//Product edited successfully.*

**OUTPUT** (Saved!)

**End.**

# Website Storyboard

## Input Design

For our input design we chose to show the main input screens for both the customer and the administrator.

For the Customer he must be able to perform actions such as Logging In, Signing Up to the website, entering address details, entering card details to make a purchase, purchasing products.

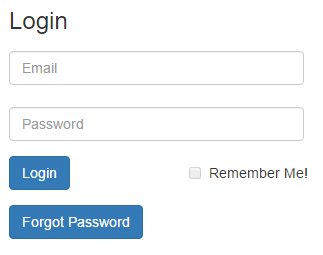
For the Administrator he must also be able to Log in so he can check stock flow and look at reports.

### Customer Draft Input Screens

The following screens are just rough copies and not to be taken as the final implemented design.

#### Logging into the website:

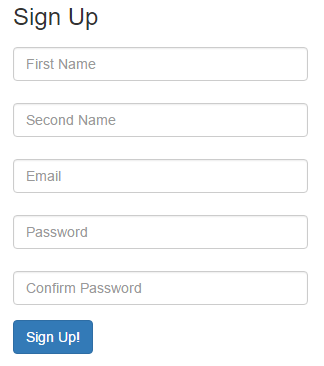
User must enter his/her email and password..



## 

#### Signing up to the website:

User must enter details name, email, password..



User must then enter his/her address details, house number, street name, county, postal code.

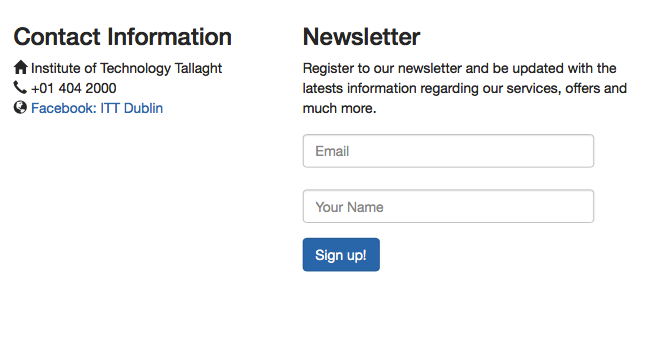
## 

User must enter card details when buying something from the website, name on card, card number, expiry month, year and security number.

## 

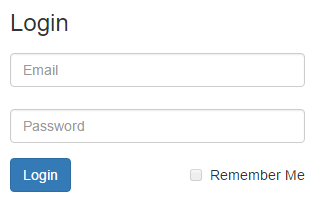
Customer enters amount of product to buy..

User can contact us or they can enter Information and register to our newsletter..

****

### Administrator Input Screens

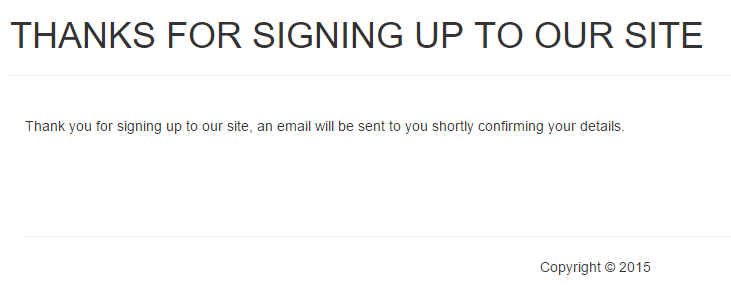
Admin must Login to access Stock details.



## Output Design

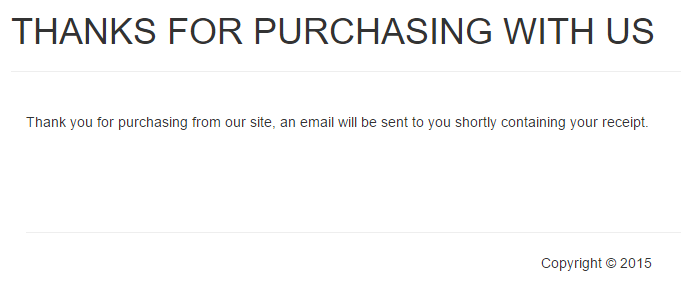
#### Customer Successfully Signs Up

The customer signs up to the site and the following screen is output to the customer:



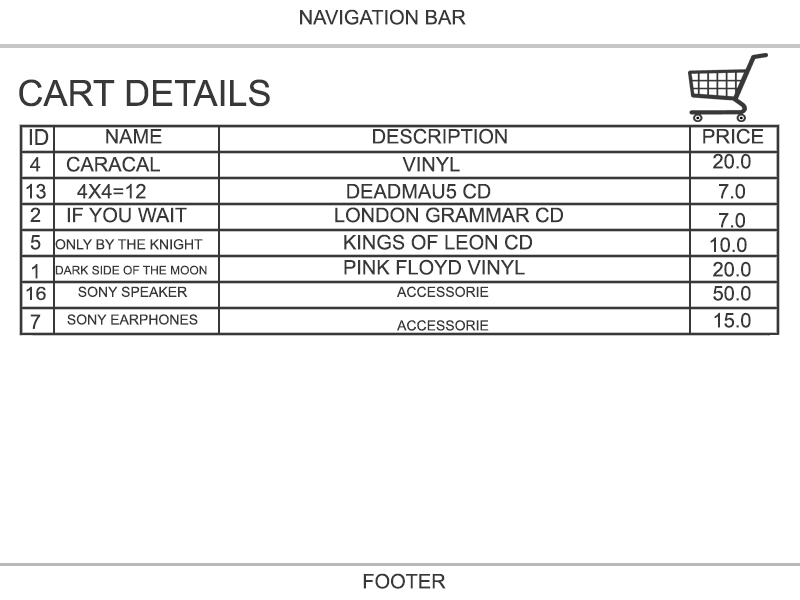
#### Customer Successfully Purchases

The customer purchases an item and the following screen is output to the customer:



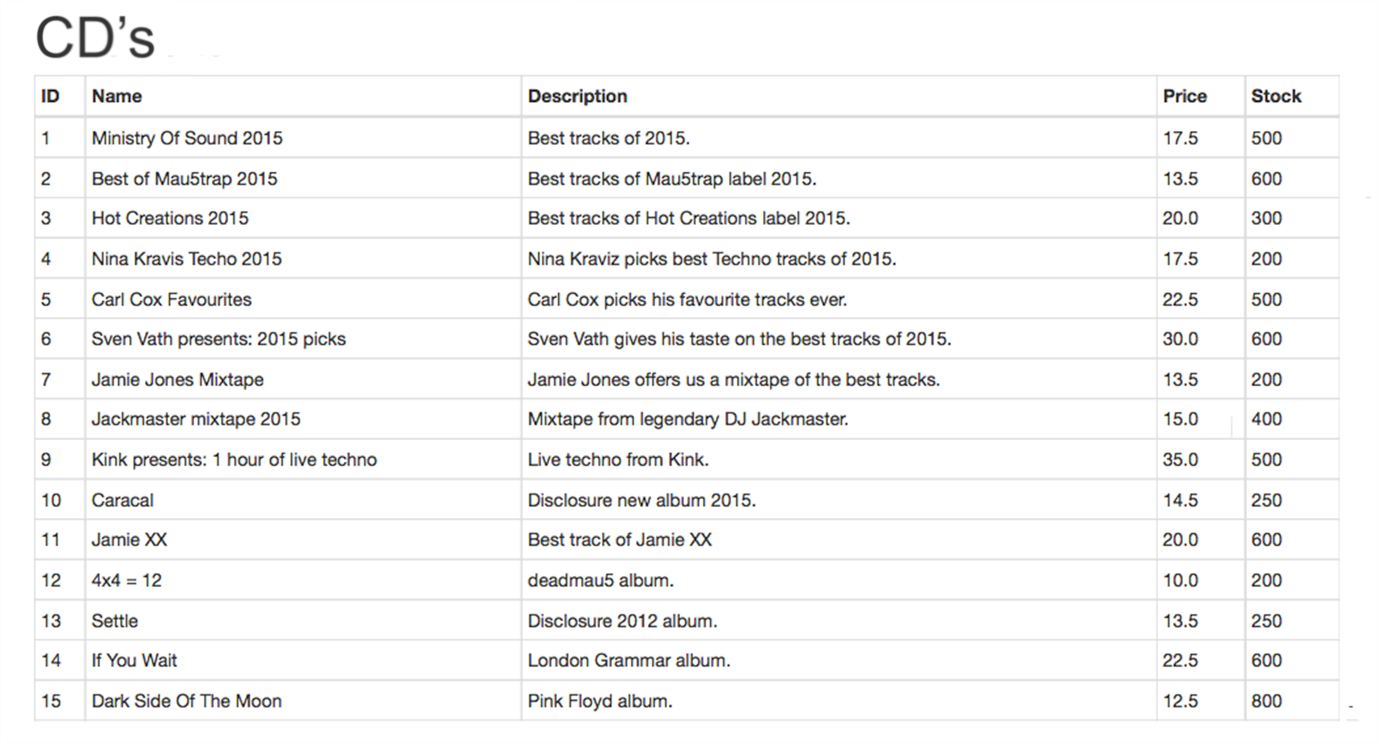
#### Shopping cart output

Customer can look at their cart..



#### Admin Stock Check

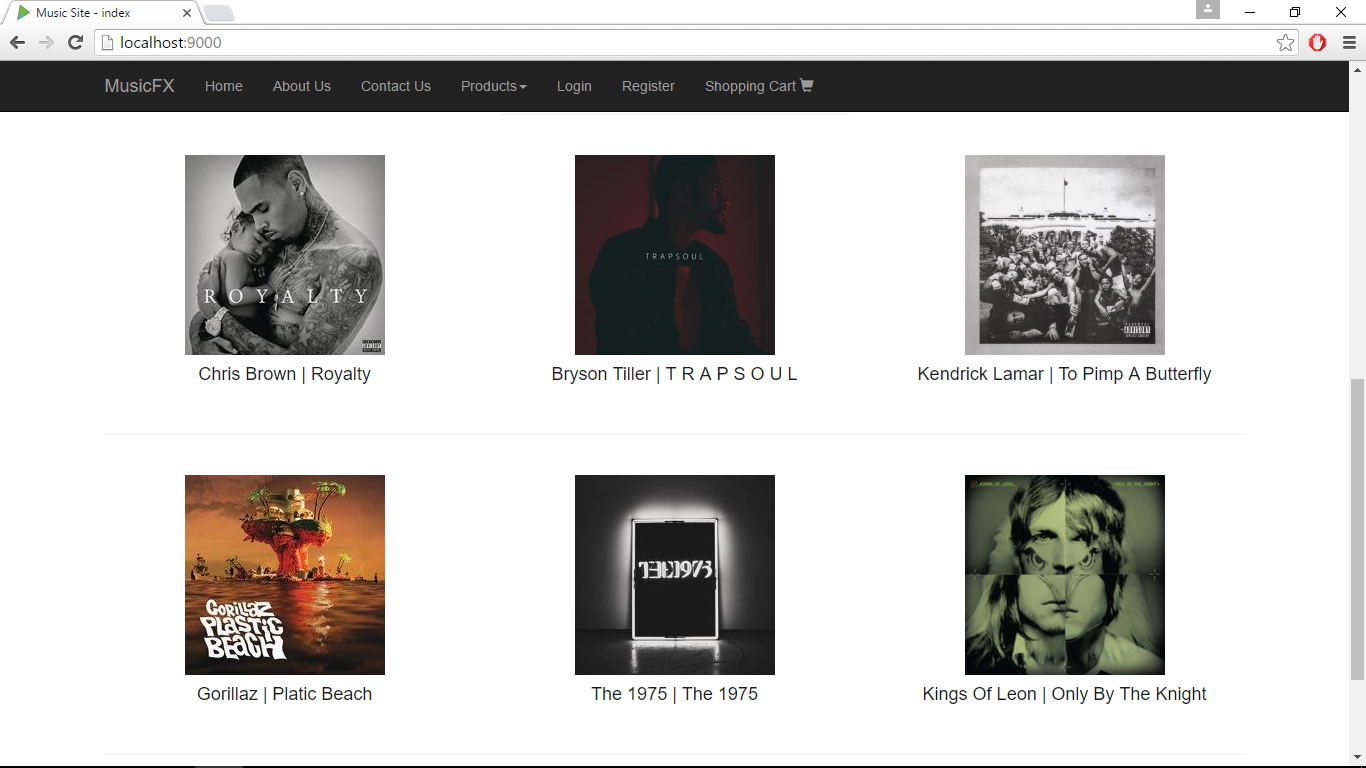
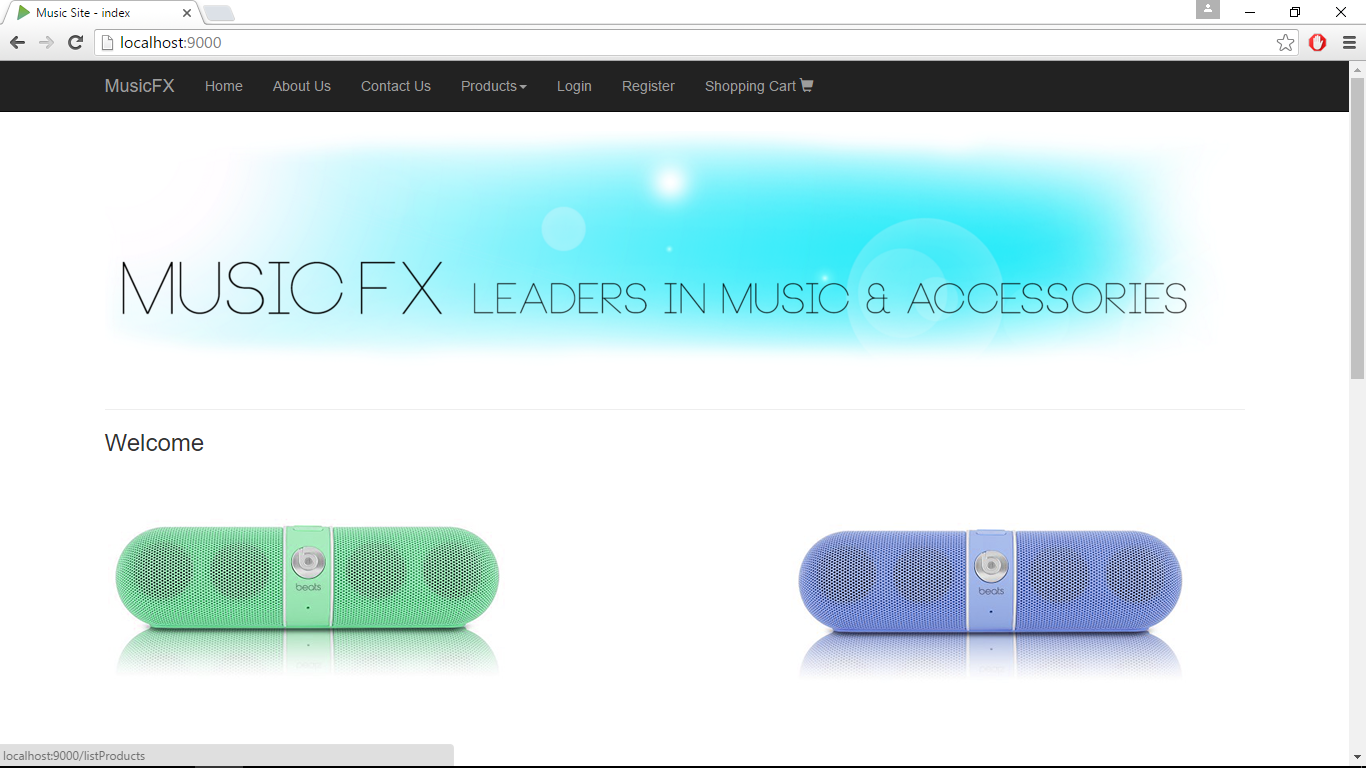
Administrator checks stock



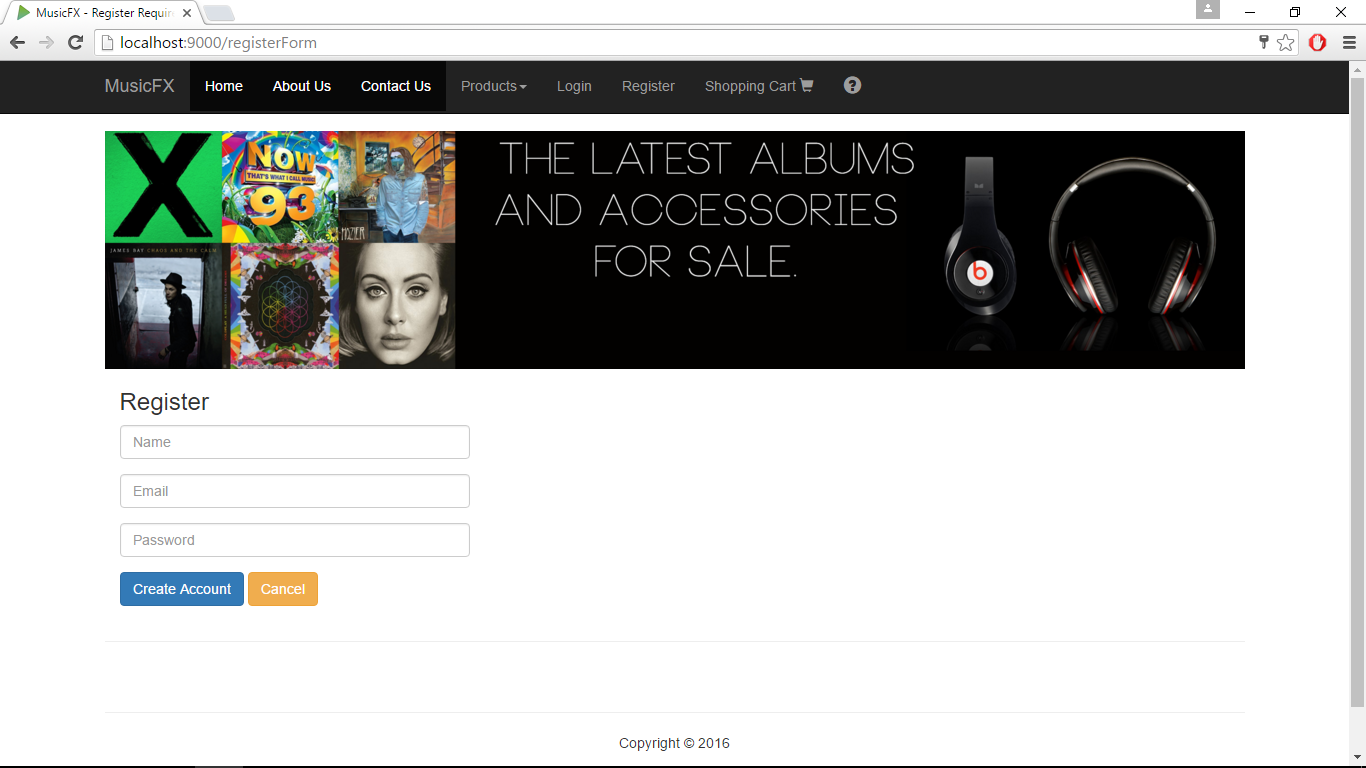
# Website Screenshots

The following are the final versions of the site.

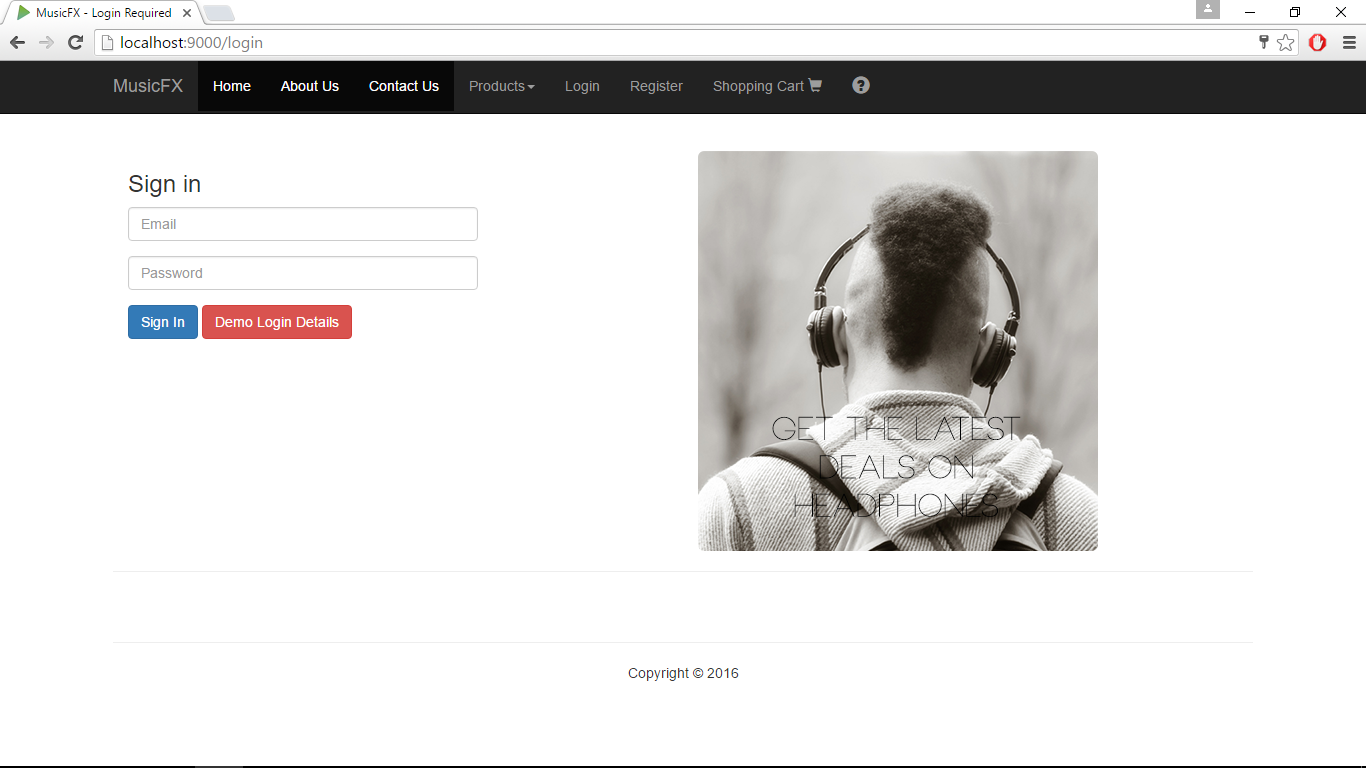
## Home Page



## Register Page



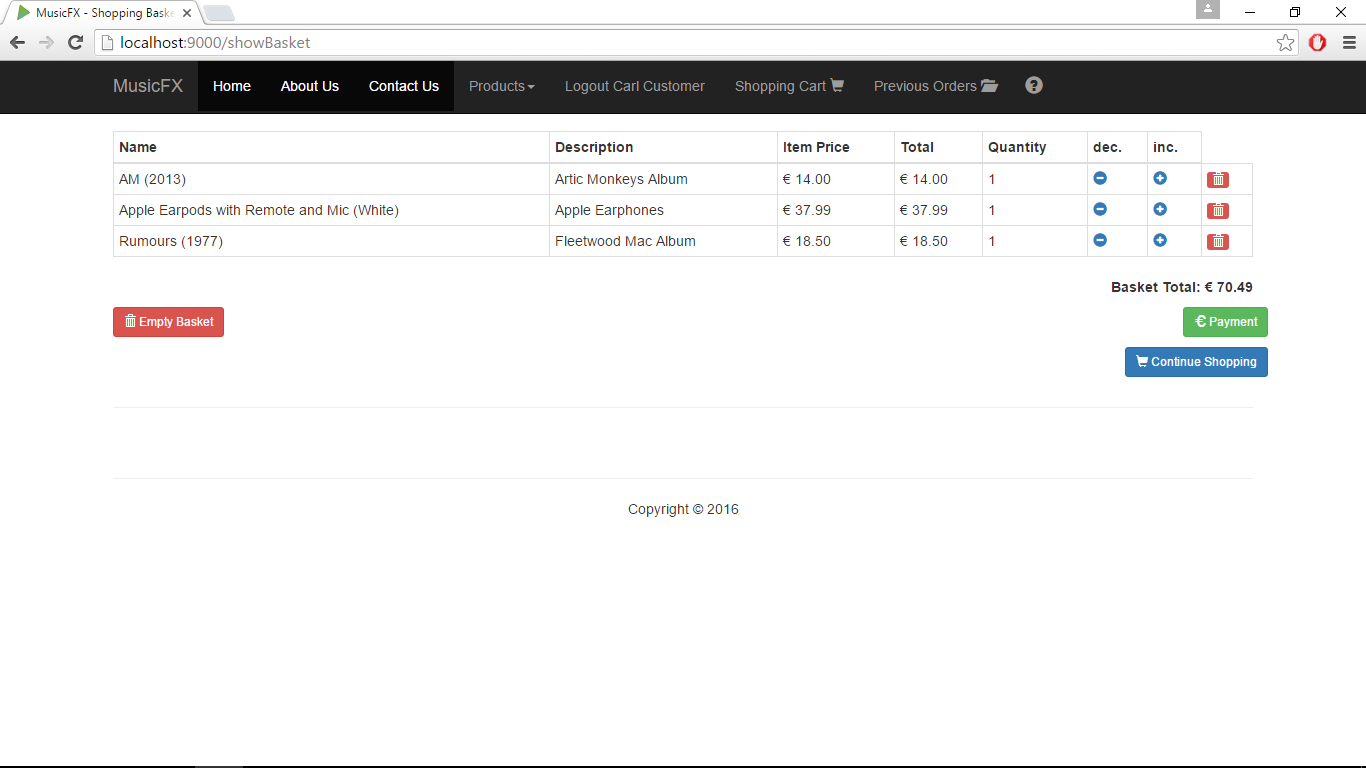
## Login Page



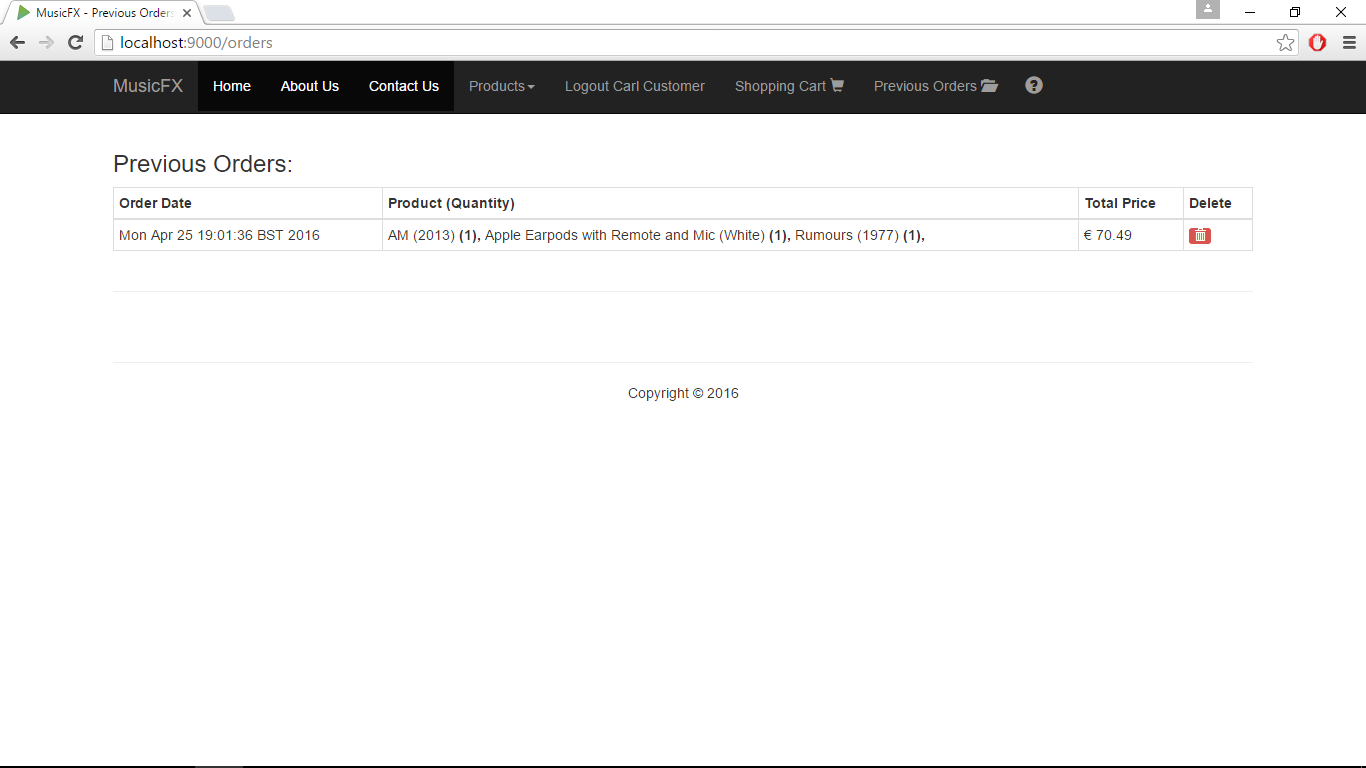
## Products Page

## 

## Shopping Cart



## Previous Orders

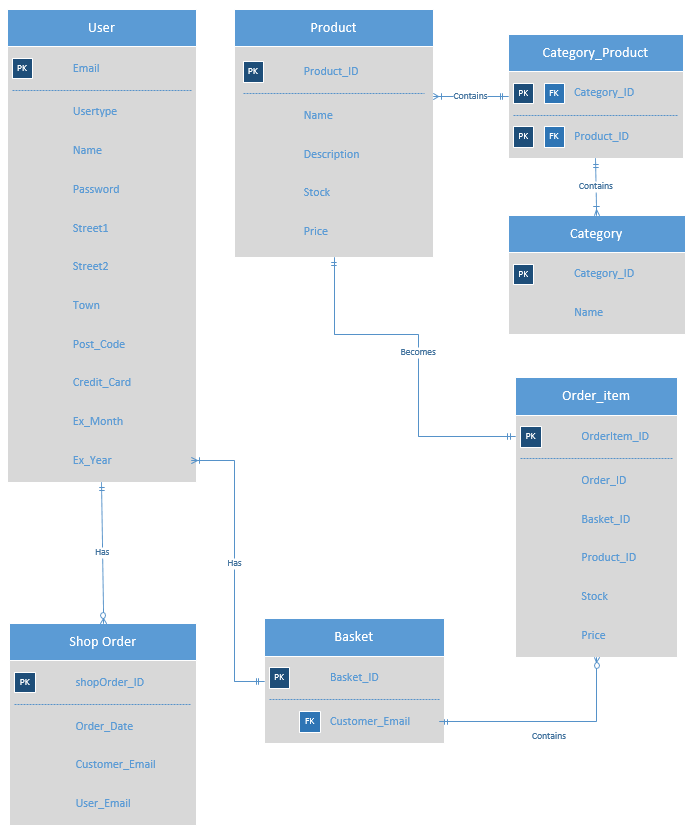


## Administrator Products View

## 

# Database Design

## ERD – Entity Relationship Diagram



## Table/Record Layouts

**User**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Usertype | Email | Name | Password | Street1 | Street2 |
| VARCHAR(255) | VARCHAR(255) | VARCHAR(255) | VARCHAR(255) | VARCHAR(255) | VARCHAR(255) |
|  | Primary Key |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Town | Postcode | Credit\_Card | Ex\_Month | Ex\_Year |
| VARCHAR(255) | VARCHAR(255) | VARCHAR(255) | VARCHAR(255) | VARCHAR(255) |

**Category\_Product**

|  |  |
| --- | --- |
| Category\_ID | Product\_ID |
| BigInt, NOT NULL | BigInt not null |
|  |  |

**Shop\_Order**

|  |  |  |  |
| --- | --- | --- | --- |
| ShopOrder\_ID | Order\_Date | Customer\_Email | User\_Email |
| BigInt, NOT NULL | Timestamp | VARCHAR(255) | VARCHAR(255) |
| Primary Key |  |  |  |

**Product**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Product\_ID | Name | Description | Stock | Price |
| Number, NOT NULL | VARCHAR(255) | VARCHAR(255) | Integer | Double |
| Primary\_Key |  |  |  |  |

**Order\_Item**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OrderItem\_ID | Order\_ID | Basket\_ID | Product\_ID | Stock | Price |
| BigInt, NOT NULL | BigInt | BigInt | BigInt | Integer | Double |
| Primary Key |  |  |  |  |  |

**Category**

|  |  |
| --- | --- |
| Category\_ID | Name |
| BigInt, NOT NULL | VARCHAR(255) |
| Primary Key |  |

**Basket**

|  |  |
| --- | --- |
| Basket\_ID | Customer\_Email |
| BigInt, NOT NULL | VARCHAR(255) |
| Primary Key | Foreign Key |

## SQL

create table basket (

 id                       bigint not null,

 customer\_email          varchar(255),

 constraint uq\_basket\_customer\_email unique (customer\_email),

 constraint pk\_basket primary key (id))

;

create table category (

 id                         bigint not null,

 name                       varchar(255),

 constraint pk\_category primary key (id))

;

create table order\_item (

 id                         bigint not null,

 order\_id                   bigint,

 basket\_id                 bigint,

 product\_id                 bigint,

 stock                      integer,

 price                      double,

 constraint pk\_order\_item primary key (id))

;

create table product (

 id                        bigint not null,

 name                       varchar(255),

 description                varchar(255),

 stock                      integer,

 price                      double,

 constraint pk\_product primary key (id))

;

create table shop\_order (

 id                         bigint not null,

 order\_date                 timestamp,

 customer\_email          varchar(255),

 user\_email                 varchar(255),

 constraint pk\_shop\_order primary key (id))

;

create table user (

 usertype                   varchar(31) not null,

 email                      varchar(255) not null,

 name                       varchar(255),

 password                  varchar(255),

 street1                   varchar(255),

 street2                    varchar(255),

 town                       varchar(255),

 post\_code                 varchar(255),

 credit\_card                varchar(255),

 ex\_month                   varchar(255),

 ex\_year                    varchar(255),

 constraint pk\_user primary key (email))

;

create table category\_product (

 category\_id                bigint not null,

 product\_id bigint not null,

## Test Case Matrix

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case Matrix** | | | | | |
| **Test Case No** | **Test Case Name** | **Input** | **Expected Outcome** | **Actual Outcome** | **Result (Pass/Fail)** |
| 1 | login | Enter email and password. | User is successfully logged in to the profile | Admin is logged in and Customer is logged in successfully. | Pass |
| 2 | logout | Click logout. | Successfully logged out as customer. | Admin and Customers successfully log out. | Pass |
| 3 | register | User enters name, email and password. | Successfully becomes a customer and is brought to login page. | Users can successfully register by entering name, email and password | Pass |
|  | ADMIN |  |  |  |  |
| 4 | editProduct | Admin can edit products, name, description, quantity, type, price. | Product will be edited successfully without error. | Products name, description, quantity, type and price can successfully be updated. | Pass |
| 5 | addProduct | Admin clicks add new product and is brought to a add product form, where they can add name, description, quantity, price, type. | Variables are inputted and a new product is created. | When clicked, Admin enters products name, description, quantity, type and price and the product is successfully added. | Pass |
| 6 | deleteProduct | Admin clicks delete button and product is deleted. | product is deleted from the website and database. | Product successfully deletes from the database and site. | Pass |
| 7 | checkOrders | Admin looks at previous orders. | Admin can look at customer's previous orders. |  | Fail |
|  |  |  |  |  |  |
|  | CUSTOMER |  |  |  |  |
| 8 | updateProfile | Prompt user to enter name and password | Profiles details are successfully changed | Unused function | Fail |
| 9 | selectCategory | User can select between categories. | Each different category is displayed to the User | User can successfully view products by their category if they want | Pass |
| 10 | buyProduct | Add product to basket. | Product is added to basket. | Products successfully adds into the basket | Pass |
| 11 | incrementProductAmount | Customer increases the amount of an item to their basket. | Product amount is increased. | When the plus symbol is clicked the quantity goes up by one on each click. | Pass |
| 12 | decrementProductAmount | Customer decreases the amount of an item to their basket. | Product amount is decreased. | When the minus symbol is clicked the quantity goes down by one on each click. | Pass |
| 13 | stockLimit | Customer tries to order a certain amount of a product. | Is presented with a flash message telling them that there is not enough stock left. | A flash warning appears telling the user they cannot add more stock to their order as there is not enough left. | Pass |
| 14 | updateStock | Updated stock amount after Customer makes a purchase. | Stock is successfully changed and displays the updated correct information. | When an order is put through/placed the quantity on display will change accordingly to the database. | Pass |
| 15 | filterSearch | Customer can look up any item they want. | Product search to be filtered to the customers search. | Customers can successfully search products by name | Pass |
| 16 | deleteCartItem | Customer chooses which item to delete from the basket. | The user's cart is successfully updated with the new subtotal being displayed | Customers can successfully delete a single product from their basket that they dont want. | Pass |
| 17 | emptyBasket | Customer clicks empty basket. | Basket is fully emptied. | When the ‘Empty Basket’ button is clicked the basket successfully empties. | Pass |
| 18 | makePayment | Customer clicks Payment button and is presented with a form to fill in payment details. | Customer enters payment details and there details are added to the database. | Details can be entered into the form but they will not persist to the database. | Fail |
| 19 | placeOrder | User confirms payment of purchase. | After entering payment details customer is presented with confirmed order details. | After entering the payment details and continuing on with the order, an order confirmation will be displayed to the customer | Pass |
| 20 | previousOrders | Customer selects previous orders to look at their orders. | Customer can see all of their orders. | Customer can successfully see all their previous orders when the button is clicked | Pass |
| 21 | deleteOrder | Customer is able to delete theeir order. | If they want to the customer can delete their previous orders. | Orders are successfully deleted | Pass |

# User Manual

The user manual can be found online. Simply visit our site and click the question mark on the right side of the navigation bar. From here you will be displayed with an interactive list which will bring you to each function. You will be shown how to do each function step by step with pictures to help you see what you are doing. Everything being shown will be highlighted by a red box to help you further understand what is going on.

There is manual for all the customers functions and also all the administrators functions.

# Conclusion

As you can see from everything above, we tried to make MusicFX as functional as possible. We tried to cover everything in terms of catering for every aspect of an e-commerce website. We done thorough investigation and research as soon as we decided what type of site we wanted to create. We all went to our own jobs and done interviews with our managers and put an intensive survey online to find out what details people really want to able see and do on an e-commerce site. We found that these interviews and surveys benefited us greatly in the long run as to deciding what to implement into our site.

Creating the site starting with the back end benefited us as it meant we implemented all our functionality first and could then just decorate around it. This made it easier for us to code as we didn’t have to look through hundreds of lines of html to correct our functional code. When designing the site we looked at many popular sites to find common themes and common designs that we could also incorporate to our own site. We also created many custom pictures for the website using Photoshop which give our site a personal touch from ourselves.

After designing the site we feel it looks professional and does its job working as an e-commerce site to the public. We feel most people of all ages would be able to navigate through our site even without the help of the user manual provided.

As far as functionality goes we feel we reached the target of including the basic CRUD features that you would see on most e-commerce websites as a customer and also as an administrator. We added small extra features to the site to make it our own.