National College of Ireland

BSc in Computing

2014/2015

Shane O’Brien

X11103400

X11103400@student.ncirl.ie

It’s POPCORNtime!

Technical Report



**Table of Contents**

[Executive Summary 4](#_Toc522377674)

[1 Introduction 5](#_Toc522377675)

[1.1 Background 5](#_Toc522377676)

[1.2 Research and Analysis 5](#_Toc522377677)

[1.3 Aims 6](#_Toc522377678)

[1.3.1 The Security of the Application 7](#_Toc522377679)

[1.3.2 The authentication of Users 8](#_Toc522377680)

[1.3.3 Encryption of the database 8](#_Toc522377681)

[1.3.4 Secure Purchasing Process 9](#_Toc522377682)

[1.4 Technologies 10](#_Toc522377683)

[1.5 Structure 11](#_Toc522377684)

[2 System 12](#_Toc522377685)

[2.1 Requirements 12](#_Toc522377686)

[2.1.1 Main Functional Requirement Details and Main Flows 12](#_Toc522377687)

[2.1.2 Non-Functional Requirements 24](#_Toc522377688)

[2.1.3 Data requirements 24](#_Toc522377689)

[2.1.4 Usability requirements 27](#_Toc522377690)

[2.2 Architecture 28](#_Toc522377691)

[2.2.1 User Requirements and Roles 28](#_Toc522377692)

[2.2.2 Environmental requirements 28](#_Toc522377693)

[2.3 Design 30](#_Toc522377694)

[2.4 Implementation of the Prototype 38](#_Toc522377695)

[2.4.1 Prototype Code 38](#_Toc522377696)

[2.4.2 Prototype Graphical User Interface (GUI) Layout 47](#_Toc522377697)

[2.5 Testing 51](#_Toc522377698)

[2.6 Customer testing 53](#_Toc522377699)

[2.7 Evaluation and Testing results 53](#_Toc522377700)

[3 Conclusions 55](#_Toc522377701)

[4 Further development 56](#_Toc522377702)

[5 References 57](#_Toc522377703)

[6 Appendix 61](#_Toc522377704)

[6.1 Project Proposal 61](#_Toc522377705)

[6.1.1 Introduction 61](#_Toc522377706)

[6.1.2 Background 62](#_Toc522377707)

[6.1.3 Objectives 63](#_Toc522377708)

[6.1.4 Technical Approach 64](#_Toc522377709)

[6.1.5 Special resources required 64](#_Toc522377710)

[6.2 Project Plan 65](#_Toc522377711)

[6.3 Monthly Journals 66](#_Toc522377712)

[6.3.1 Reflective Journal September 2016 66](#_Toc522377713)

[6.3.2 Reflective Journal October 2016 67](#_Toc522377714)

[6.4 Prototype User Guide and Environment Setup 69](#_Toc522377715)

# Executive Summary

The project is a web-based application that can be used to order food in the cinema from their seat in the screen. The prototype is built using the technologies HTML 5, CSS 3, JavaScript, PHP and a PayPal API allowing for mock Payments through a sandboxed Environment.   
There will be a feature for Users to be able to register and log on to the application. Although the prototype only has Login initialized.   
Its main feature will be selecting products, viewing within a cart and then purchasing through a payment method. The order will then be sent for fulfilment by the cinema staff who will deliver it to the user in their seat.

The prototype is more just a proof of concept, as it only has a couple of the overall planned features. There are some design elements, but the main features are the Login and a mock Payment through the PayPal API.

# Introduction

## Background

The idea for my application came whilst visiting the cinema with a friend who is in a wheelchair. Due to complications with travel that day to the cinema we were running late, but we had pre-booked tickets, so we had rushed in to the cinema. We got to our seats just as the movie began, although we had planned to get food we bypassed the counter because we were running late. I then noticed a couple arrive with popcorn and drinks 15 minutes in and had missed the beginning of the movie.

This technology is similar to apps that are currently being used in Cafes e.g. Starbucks, which provide an ‘order and collect’ service via app, but is still a relatively new concept, and it has not been tried in cinemas here in Ireland. Although I thought it was innovative I discovered during my research this technology is currently being developed and tested in cinemas elsewhere.

## Research and Analysis

When I first started this project, there were three cinemas trailing some concept similar to my project. Two are based in the USA:

Regal Cinemas have Express Lane which is a prepay and collect service using the ATOM mobile application.   
AMC Cinemas Have Order Ahead which can now be done live through their website, which is also a Prepay and collect service.

In Canada Landmark Cinemas and expected through Cineplex Cinemas using an app, customers will be able to view movie listings, buy tickets and pre-purchase popcorn, drinks and other concessions.

In Australia Master card trialed QKR an NFC Payment system in Hoyt Cinema’s La Premiere a luxury cinema experience now called LUX. LUX is a dine-in and cinema experience where visitors enjoy a in seat purchase and delivery from a luxury menu. The LUX service is available in several Hoyt Cinemas throughout Australia.  
They have some competition with Event Cinemas that offer Gold class which is also a dine-in and cinema experience like LUX.   
In both the visitors can order and pay from their cinema seat and it is delivered to them in their seat but this is done, to my understanding through a waiter service.

Although these are similar concepts none offer an app where you can order and pay directly from your seat in cinemas. I had discovered in stadiums in the USA there is an app where you can order concessions and pay for them digitally and are delivered by an usher. But seem to only be aimed at stadiums. But to date there is nothing like this in Ireland. So, there is a possible market.

## Aims

My objective is to create a web-based application for consumers to order food from their seat within the cinema screen or to pre-order before they arrive.  
This will be achieve using a web-based application called “It’s POPCORNTime”.

The web application will require Users to register for an account which will save personal information such as their full name, email, contact number, address and credit card information. They will then log on through a Login Page using an email and password.

I also plan on having an Admin Console. This will be a local Administrator view that the cinema or theatre managers will be able to access.   
From here they will be able to create items for sale with a simple form. Also set and change advertisements or set percentage discounts that can be accessed by the user by them inputting a special code. The admin will be able to set this code. The well as create items, Adverts or Promotions, the admin-user will be able to Edit or remove items from displaying on the user’s page, meaning they can change the price of the item, or delete it altogether, making it unavailable to purchase and will not be displayed to the users.  
Another feature will be they can add promotional text to the to display on the Index Page. This text may be able to be transformed using font styling html 5 and CSS 3 tags. The placement on the index page will remain the same.

This will also be tested for cross site scripting and Injection attacks for security. To aid prevention of attacks the tags allowed will be limited and the text will be kept as a string. It will not be linked to a database.

Another potential feature may be to view and collect the anonymous data which could be used for research. This can then be used to view how popular an item is, the number of registered users who access the web application. As well as how many purchases are made after clicking the guest option. This data will be displayed to the Admin in a table or chart to help them know their clientele better and cater to them to provide a better service.

This would be supplemented by an automated user experience survey they can send after purchase.

The main focus for this application will be based on:

1. The Security of the Web Application against attacks,
2. The authentication of the users
3. The secure encryption of the database.
4. The implementing secure payment process.

### The Security of the Application

According to OWASP, 60% of Web based applications are vulnerable to SQL injection, Cross-Site Scripting and other forms of attacks.  
The intention is to make sure that “It’s Popcorn Time!” is secure against such attacks and will be thoroughly tested to detect and eliminate these vulnerabilities.

### The authentication of Users

This will be achieved by users being required to verify their email address after registration. They will also have to set security questions and answers for password recovery during the registration process. The password the user sets will have to meet strict requirements.

### Encryption of the database

This is done to secure the most sensitive user data such as users address, password and contact number but most importantly their credit card information.

The passwords will be hashed before saving to the database. This means the password will be converted in to a indecipherable string of characters that are designed to be extremely difficult and close to impossible to decrypt and convert back in to a readable state. Hashing is a form of encryption designed not to be decrypted instead an entered password is hashed again with the same method and the resulting 2nd hash which is then compared to the original that is stored in the database. If they match then the access is granted, but the user created password in String format is never stored.

The way I plan to do this is using the Default PHP 5-5 “native password hashing API” which uses an implementation of BCrypt hashing, based on the Blowfish Cipher. For security the BCrypt Iterator takes a lot more time to try decrypt and harder to be deciphered by brute force attacks(see the sample code) Unlike the sample code I will not use the username as the salt or link the username to the hash because if the username is changed then the password would be required to be changed and hashed again with the new username as salt, or the username could not be changed at all.

Hashing method could not be used when storing credit card numbers for registered users, the ccv and expiry date would be requested to be input by the user at every transaction, they will never be stored. Although the encrypted credit card number that will be stored would be required to be decrypted to process the transaction without the user having to put it in every time.  
Although this may be unnecessary as the PayPal Payment Process using a PayPal account Login may have the users card linked. Therefore, there is little practical need for the user’s card to be stored. If storing credit card numbers is required, then it is best to use a trusted 3rd party company that is PCI certified or an official local Banking System that you can connect to, to store credit card data and process payments like the PayPal, Stripe or my Easy Pay, as that is best and most secure Practice.

### Secure Purchasing Process

Provided by the PayPal API. The User will add items to the cart then then it’s sent to PayPal where they can log in with a PayPal account or use their card through the PayPal guest Payment function.  
Ideally there Verified by visa will also be implemented during the purchase process.

The challenge of developing this application, will be in implementing the above, as I have had little experience in the secure development of any application. Although I have some previous experience implementing a log in with simple authentication. I may also try setting a log in with google or Microsoft account features.

## Technologies

|  |  |
| --- | --- |
| HTML | Hypertext Markup Language, a web design language. |
| Bootstrap CSS | The term for Mobile CSS that is a form of Responsive Web Design |
| CSS | Cascading Style Sheets, used to style the appearance. |
| JavaScript | Another programming Language I plan to use for Seat Selection. |
| Sublime Text | The Text editor used for development. |
| React JS | A JavaScript Framework. |
| PayPal API | Specifically, the Payment Buttons. |
| XAMPP | The localhost Web server Application used to host the development website. |
| PHP | A general-purpose server scripting language used to link the database with the web site, to create the web application. |

## Structure

The structure of this document is in two parts. First part I talk about the main description for the intended project. I discuss the System and its requirements. I then display a flowchart and have the foundation of tables in the database. I go on to describe the User and environmental requirements, which are the user roles and the platforms it will be used on as well as the development environment.

In the Second part I discuss the Implementation of the prototype. I discuss my code and display examples. I show the prototypes GUI design. I then talk about the testing that was done, by other users and myself. I then evaluated the testing results. I then conclude with my opinion of the overall project outcome, and how I would develop it further.

# System

## Requirements

### Main Functional Requirement Details and Main Flows

#### Requirement 1 <Order Process>

**Description & Priority**

This Requirement is part of the core basis of the application.

Either a Registered user or a guest user will go through this process. The User will input their seat location on the ‘Seat and Screen’ Page form. They will then progress to the ‘Item Selection’ page and select items they wish to purchase which will be added to the ‘cart’ for purchase. The User may then view and edit items in the ‘cart’ before purchasing.

**Use case**

ID 01

**Scope**

The scope of this use case is for a user to complete the three main steps of the web application and progress to the final step.

**Description**

This use case describes how the user starts the main process of the web application, the user will input the seat number and select the screen number, then press the ‘next’ button.

The user will then progress to the ‘item selection’ page where they will then select the items they wish to purchase and then press the ‘next’ button. They can then View the items they selected before purchasing before starting the payment process by clicking the ‘next’(Buy) button

**Flow Description**

**Precondition**

• Mobile device has Internet connection.

**Activation**

The use case starts when a user Logs in or clicks the Guest button.

**Main flow:**

1. The user has progressed to the ‘Screen and Seat Selection’ page.

2. The application displays the ‘Screen and Seat Selection’ page.

3. The user selects, from the dropdown box, the screen they are in.

4. The user enters(selects) their seat location.

5. The user clicks the ‘next’ button.

6. The application displays the ‘Item Select’ page.

7. The User will then locate the items they wish to purchase on the page.

8. The User will select the size of the item using the radio button.

9. Then the user will select how many items they wish to purchase by using the counter on the side.

10. The user will then click the ‘next’ (add to cart) button.

11. The application will display the ‘cart’ page.

12. The user may edit, delete items or items amounts, or click the ‘next’ (Buy) button to purchase items

**Alternate flow**

**A1 : <User did not input screen and/or seat number>**

1. The user does not select a seat and/or screen number on the ‘Screen and Seat Selection’ Page.

2. The Application displays an error message to user stating they Cannot progress without inputting the screen and/or seat number.

3. The use case continues at position 3 of the Main Flow.

**A2 : <User has not selected any items or item Size>**

1. The User has progressed to position 7 of main flow and has not selected an item size and/or any items at all, then tries to progress by clicking the ‘next’ (Add to Cart) Button.

2. The Application displays an error informing the User that they must select an item and/or an item size before continuing.

3. The Use case continues at position 7 of main flow.

**A3 : <User Edits Cart Items: Item Amount>**

1. The user has progressed to position 11 of main flow.

2. The User chooses to add/reduce the amount of an item.

3. The user does this by increasing or decreasing the counter beside the item.

4. The User then progresses to purchase items by clicking the ‘next’ (buy) button.

**A4 : <User Edits the Cart Item: Removes Item>**

1. The User has progressed to position 11 of main flow.

2. The User chooses to remove an item from the cart.

3. The user does this by either decreasing the counter to ‘0’ or by clicking the ‘X’ (delete) button.

4. The Application refreshes the ‘Cart’ Page, and the item has been removed.

5. The User progress to purchase item(s) by clicking the next ‘buy’ Button.

**Exceptional flow**

**E1 : <User Decides to add more items to the cart>**

1. The User has progressed to position 11 of the main flow.

2. The user realises they had to forget to add an item or chooses to add more items, so the User clicks the ‘Back’ Button to return to the ‘Item Selection’ Page.

3. The Application saves the Items already in the cart.

4. The Application displays the ‘Item Selection’ Page.

5. The User Continues from position 7 of the main flow.

**Termination**

The Application closes.

**Post condition**

Once the user has completed this requirement, The Application will display the ‘Payment’ Screen and the user progresses to the ‘Payment Process’.

#### Requirement 2 <Payment Process>

**Description & Priority**

This Requirement is part of the core basis of the App. The User must complete this to purchase their items. The User will input credit cards details e.g. cardholder name, select card type, card number, card expiration date and card security code.

This maybe done using a PayPal API for security and time constraints. The user will then select the ‘pay’ button.

**Use case**

ID 02

**Scope**

The scope of this requirements is that users may pay for their order safely and securely.

**Description**

The use case describes the payment process the user must go through the user must go through to finalize the purchase of the items they selected

**Flow Description**

**Precondition**

• Internet Connection

• Requirement 1 has been completed.

**Activation**

The Use Case starts at when the user presses the ‘next’ button on the ‘cart’ page.

**Main flow:**

1. The user has completed Requirement 1 Main Flow by pressing the ‘next’ button on the cart page.

2. The Application loads the ‘payment’ Page.

3. The user inputs the required details.

4. The user enters Cardholder Details.

1. The user selects the card type from the drop-down box. E.g. Visa, Visa debit, Mastercard.
2. The user enters the card number.
3. The user will input the card expiration date.
4. The user enters the 3-digit Security Code.

5. The User selects the ‘Pay’ button

6. The App then submits the transaction Request along with the credit card information, which is security transmitted to the payment gateway. The details are encrypted so no-one can see the users card details.

7. The card details will be verified, and that the user has sufficient funds to complete the transaction.

8. If the user’s details are valid and there is enough funds, then the transaction amount will be transaction amount will be transferred from users bank account to the Merchants account.

9. Once transfer is complete and confirmed. The Application will display a ‘Confirmation’ Message on the Payment Page.

**Alternate flow**

**A1 : <Details Input Incorrectly>**

1. The user has not filled in any details or left any part blank.

2. The Application displays an error message, informing the user that all details required to progress.

3. The user continues at position 2 of the Payment Page.

**Exceptional flow**

**E1 : <Username already exists>**

1. The use case has completed step 5 of the main flow.

2. The application submits transaction request and card information.

3. The transaction is declined due to invalid details

4. The Application displays an error message to user stating the transaction has been declined due to invalid details.

5. The users continue at position 2 of main flow.

**E2 : <Insufficient Funds>**

1. The use case has completed step 5 of the main flow.

2. The Application submits transaction request and card information.

3. The transaction is processed but is declined due to insufficient funds.

4. The Application displays an error message to user stating the transaction has been declined due to insufficient funds.

5. The App Closes.

**Termination**

The Application processes has completed.

**Post condition**

The items have been paid for and the order will be dispatched and delivered to the user.

#### Requirement 3 < Seasonal Advertisements>

**Description & Priority**

Medium priority.

Throughout the year at specific times/events the cinema will offer promotions or advertisements for movie releases. The Application will have sample time-based Advertisements or promotions for example during summer months (may – july) a promotion of “buy one cold drink, get one half price”, or at Christmas (December) a “two tickets for one” or “kids go free at Easter (March). These Advertisement will be part of the application but will only display in the specific time frame of the time/event, in this case during the calendar months.

**Use case**

ID 03

**Scope**

The scope of this use case is that the Application will display the seasonal advertisements and promotions at the correct time of year.

**Description**

This use case describes how and when the seasonal Advertisements and promotions should be displayed within the application.

**Flow Description**

**Precondition**

• Application must be running.

**Activation**

The use case begins when the application is started by the user.

**Main flow:**

1. The user opens the Application.

2. The Application checks the system Date and Time.

3. The Application compares to current system date and time against the present calendar month that corresponds to the Seasonal Events, and it matches.

4. The Application will then display the Correct ads for that seasonal event on screen to the user.

**Alternate flow**

**A1 : <Not a Month of a seasonal Time of Year>**

1. The user opens the Application.

2. The Application checks the system Date and Time.

3. The Application compares the current system Date and Time to the preset calendar month that corresponds to the seasonal events, but it does not match.

4. Then the Application displays a default advertisement for the cinema for the remaining months until it matches with a month corresponding to a seasonal event.

**Exceptional flow**

**E1 : <Advertisements do not display>**

1. The user opens the Application.

2. The Application checks the system Date and Time.

3. The Application fails to display the advertisements or promotions.

4. The User should still be able to continue to use the Application as normal.

**Termination**

The Application Closes

**Post condition**

None

#### Requirement 2 <Administrator Console>

**Description & Priority**

This Requirement the administrative side of the App. From here the Admin will control the sections shown to the public users and manage, Items, Advertisements, Promotions, and Discounts through the relevant console sections. The Admin user will access these to make changes, whether adding, removing or editing. The admin user will then save, and it will change the display to the public users

**Use case**

ID 04

**Scope**

The scope of this is that the administrator can make changes to the application then publish or save the changes.

**Description**

The use case describes the actions taken for advertisement or promotions, Items and Discounts. The process will be very similar for all.

**Flow Description**

**Precondition**

• Internet Connection

• This will be designed to display desktop computer or laptop.

**Activation**

The Use Case starts at when the Admin signs in with the admin credentials.

**Main flow:**

1. The user logged in.

2. The Application loads the ‘Console’ Page with 4 sections (Promotions, Advertisements, Items, and Discounts.

3. The user will click on a section e.g. the Items Section and a table will display listing the current Items displayed to public users.

4. The user clicks the Add Button.

5. The user inputs the required values e.g. Item name, Item type, check the box for sizes available, and Input price for each size.

6. The user then saves.

7. The Application adds item to database and then displays the Item List with new Item will be added to the bottom of the table list.

8. Then the Application will publish the Item to live web application.

9. The user can repeat this process on all sections.

**Alternate flow**

**A1 : <User wants to remove item>**

1. The user has completed Step 3 of main flow.

2. The User then checks the box(es) at the end of the row of the item they wish to remove.

3. The user scrolls down and clicks on ‘remove’.

4. The Application removes the item from the database and refreshed the Item list removing it from the list.

5. The Application then publishes the changes to the Live web application.

6. The user can repeat this process all sections.

**A2 : <User wants to edit item>**

1. The user has completed Step 3 of main flow.

2. The user scrolls down and clicks on ‘edit.

3. The Application updates the item in the database and refreshed the Item list updating that too on the list.

4. The Application then publishes the changes to the Live web application.

5. The user can repeat this process all sections.

**Exceptional flow**

**E1 : <Item already exists>**

1. The use case has completed step 5 of the main flow.

2. The application tries to add to database.

3. The Application displays an error message to user stating the item with that name already exists.

4. The users continue at position 3 of main flow.

**E2 : <Internet connection disrupted>**

1. The use case has completed step 5 of the main flow.

2. The application tries to add to database.

3. No connection page error on Browser.

4. User must re-establish a stable internet connection to access the application.

**Termination**

The Administration process has completed.

**Post condition**

The items are now published on the live version of the web Application for public users to purchase or see.

### Non-Functional Requirements

#### Performance/Response time requirement

The System will be required to run in real-time and be able to be completed in less than 5 minutes Start to finish, with no more than a few seconds delay.

#### Availability requirement

Initially it should also be available only during the opening time of the cinema e.g. 5pm – 11pm, no purchases should be processed outside of this time.

#### Security requirement

There should be specific roles set with different permissions for accessing, reading, editing and deleting information.

All code must be secured.

#### Maintainability requirement

Admin must have access and permissions to allow them to maintain the system.

### Data requirements

#### User Data

A user email is required for the sign in Process and for invoices to be emailed to.  
It will be used for user validation and all other user data will be linked to it.

Previous Purchase details or Popular selection of Items the user has purchased before when using the application after login will be stored.

At registration users will have the option to add a phone number to link to their account. This is not required, and the application and registration will work the same without it but may be used for marketing purposes.

A user will be required to create a password to gain access to their user account It must adhere to specific rules (8 characters, an Uppercase letter and a lowercase letter, at least one number, and a special character.) for security purposes. This password will be hashed before being stored within the database.

Users will be able to change their password and username from within the Application or delete their account fully.

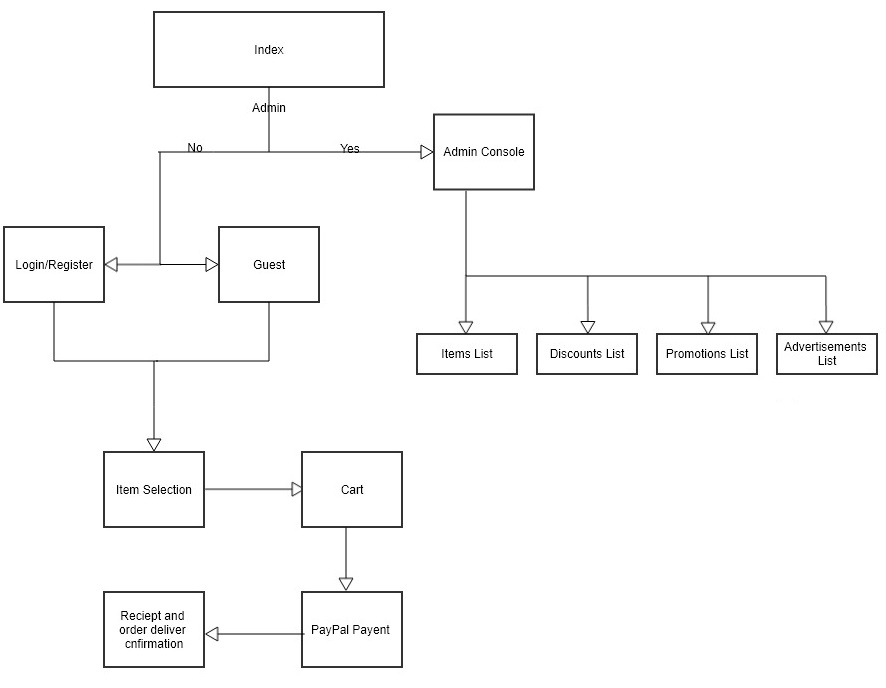
#### Database Backup

The Database must be backed up to a separate SQL file that will be stored on cloud storage.

#### Database Recovery

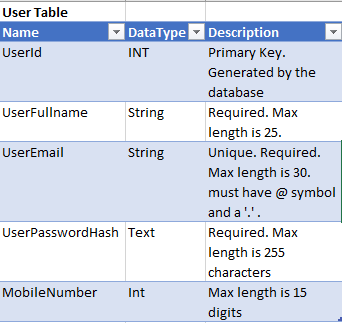
The Database must be able to be restored from the Backup SQL file that stored on cloud storage.

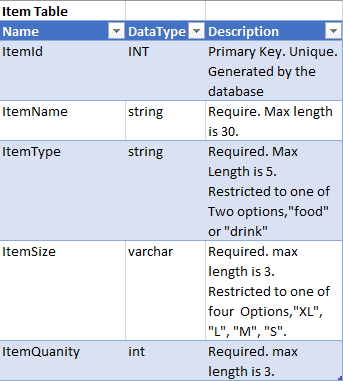
#### Flowchart

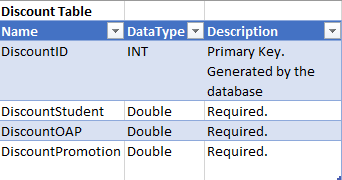
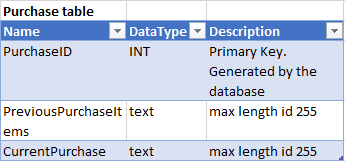
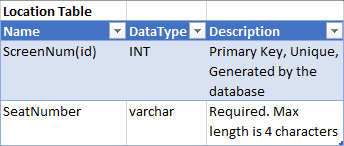


#### Data Diagrams

This is an example of the structure of the data that will be needed within the database.







### Usability requirements

User must be able to use a smart phone, tablet or laptop computer.

The user must be familiar with using an internet browser.

Preferably user will be familiar with online shopping would be beneficial, but this not necessary.

## Architecture

### User Requirements and Roles

**1. Registered User**

The registered user will be able to access the application. When they register and save the details and agree to the terms and conditions they will have extra perks. Like choosing their favorite items and view their previous orders to make the order process faster. They will also gain extra discount codes or promotional content will be displayed to them upon log in.

**2. Unregistered User/Guest**

The Guest User will be able to access and use the standard features like selecting items and purchasing them like registered users but will not be able to apply discount codes or see promotional content. They will be unable to save their favorite items.

**3. Administrator User**

The Administrator user will have full access to the application as well as the administrator console view. Here they will manage Advertisements and Promotional Content Displaying. As well as manage Items for sale and Item Prices.

This will all be done through the administrator console view. In further development they may be able to generate reports based on anonymous data collected during the application use e.g. number if users, the type of users whether guest or registered, the Popular Items etc.

### Environmental requirements

**Operating environment**

The Application is Designed to work on all web capable mobile devices. Specifically, for Smart Phones. The recommended browser is chrome although it will work on any web browser.  
The data will be stored on the server and the web application hosted online so a stable 4G or WIFI connection is required to access and use the web application.

**Development Environment**

The development will be done using a chrome browser, sublime Text editor and XAMPP localhost Server using myphpadmin for database creation and management. It will be developed in HTML 5, CSS 3, Bootstrap CSS, some JS elements (angular JS and React JS) and PHP 5.5.  
I will also be using the PayPal API payment gateway for Security and sandboxed payments in the Prototype to create a complete a complete user process or Journey.

**Assumptions and dependencies**

Without a WIFI or 3G/4G data connection users will not be able to connect to the internet to access the web application therefore an internet connection is required.

If a user doesn’t have either a credit/debit card or a PayPal account, they will be unable to purchase Items. An option for cash payment may be added in further developments, but for now Digital payments are required.

Users must consent to payment and accept terms and conditions of PayPal.

It is assumed that users are on premises of the cinema in in the screen and seat location they have input in to the application. In future development a JS seating map may be implemented.

## Design

The Webpages used in the web application are shown below:

* Start Page
* Login Page
* Registration
* Screen and Seat Selection Page (removed)
* Item Selection Page
* Cart Page
* Payment Page
* My Account Page

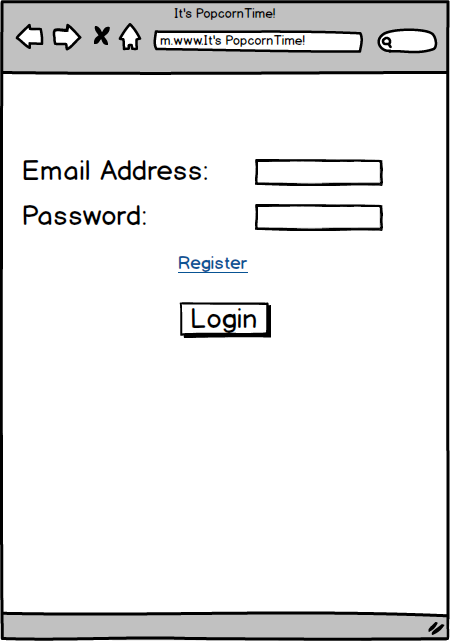
All pages except the start page will also display a menu with buttons to the ‘my account’ page and a log out button, if they have logged on. Also, buttons to the ‘Cart’ and ‘Screen and Seating’ page from which they may start the payment process or view their selected items.

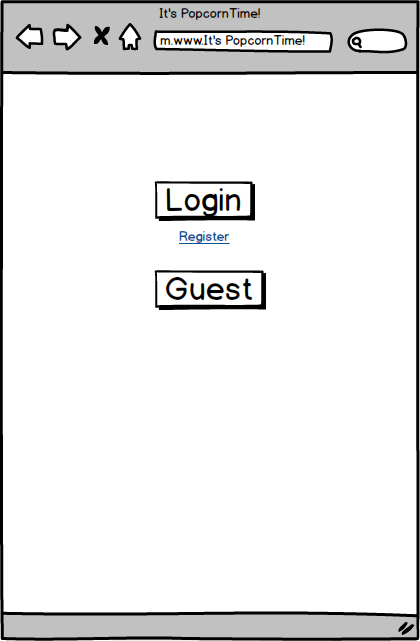
**Start Page**

This is the first page the user sees when they visit the web applications URL. It is simple and straight forward offering the users the options to go to the:

* Login to the Web Application
* To Register to Use the Application
* Or to continue without
* Logging in or registering

Below is an example of what the page may look like:



**Login**

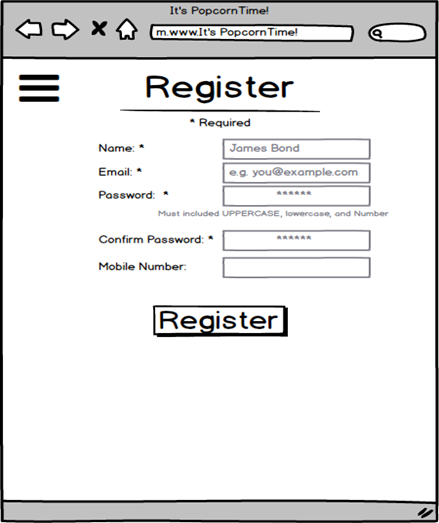
This is an important page in the Application, Here the user will be required to log in to the application using their Email and the Password they set up on Registration. This page has both inputs and a

connection to the Database so it has to provide a secure connection to the database and the input boxes must have protection against Injection attacks. To the right is an example of what the page may look like:

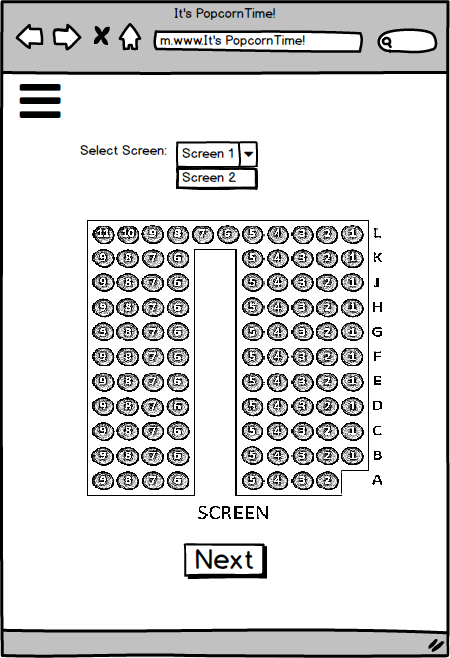
**Registration**

This is another Important page of the application. Here the user will register for an account to use the application and save their details. They will be required to provide a Full name, an Email Address, and to create and confirm a password. They will also be given the option to provide a mobile number. Again, this page also has both inputs and a connection to the Database so it has to provide a secure connection to the database and the input boxes must have protection against Injection attacks.

Below is an example of what the page may look like:

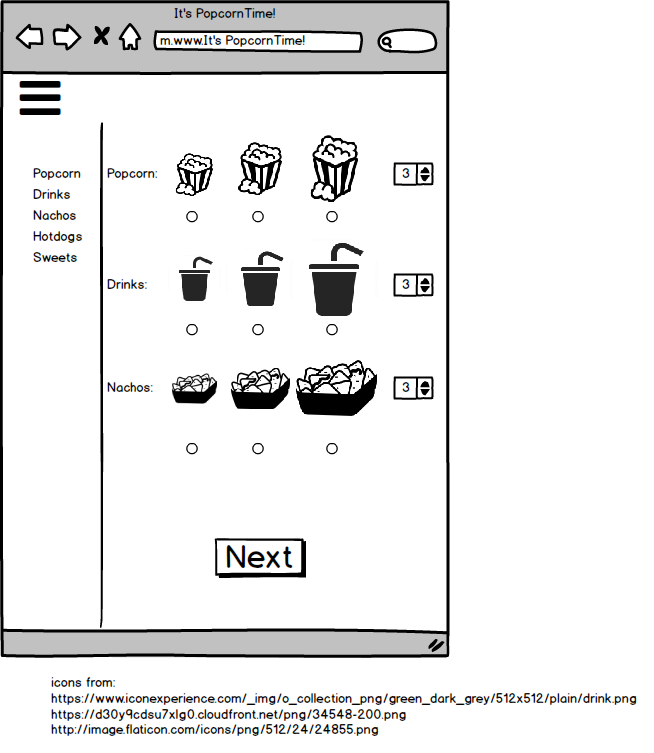


**Screen and Seat Selection**

On this page the user will select the screen that they are in. Once they select a screen a seating map will load below. The user will then select the seat they are sitting in by clicking the seat on the map before continuing to the next page. To the left is an example of what the page may look like:

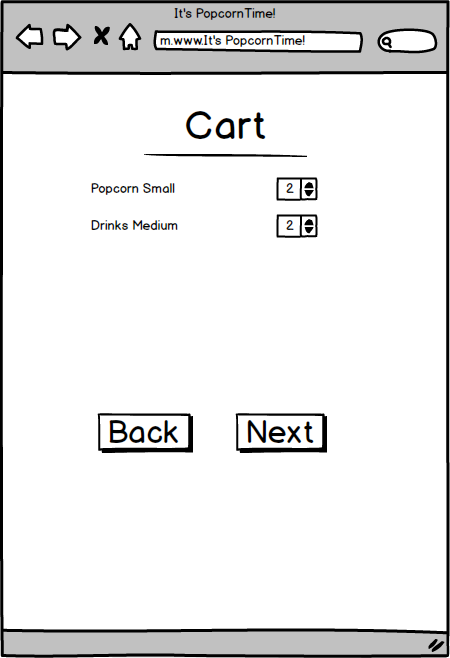
**Item Selection**

This page will be a single page listing the items available for purchase there will be a quick link side bar that will list the chosen specific types of items available e.g. Drinks, Popcorn etc. and the number of that item that they would like.  
The user will also be shown 3 size options for each Item and for drinks the various flavors available. Below is an example of what the page may look like:



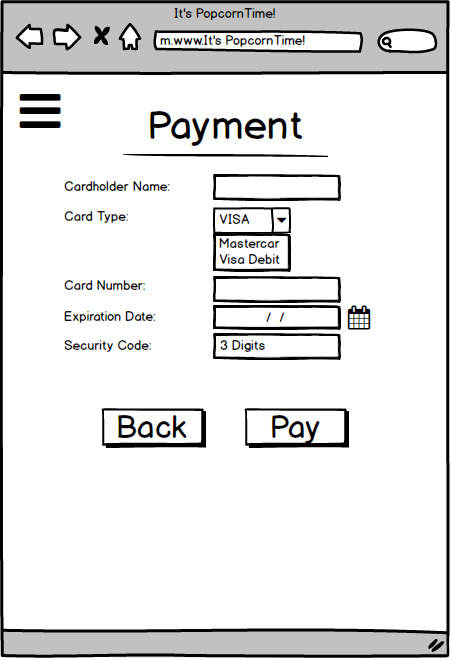
**Cart**

This page will list all the items and the number of each item that the user has selected for purchase. It will also show the breakdown and cost for the total order and the option to return to select more items to add to the cart, delete/remove and item or continue to the payment process (also, where it will give the option to input coupon code or select discount [OAP/Student] which be factored in to the total price.) Below is an

example of what the page may look like:

**Payment**

This is the most important page of the application. Here the user will use their credit card (or PayPal) account to pay for their order. Since this will require the input of Credit Card details and the charging of money for the user’s order it must be the most secure page in the application protected against all forms of attacks with a secure connection to outside resources (PayPal log in, company accounts). Below is an example of what the page may look like:



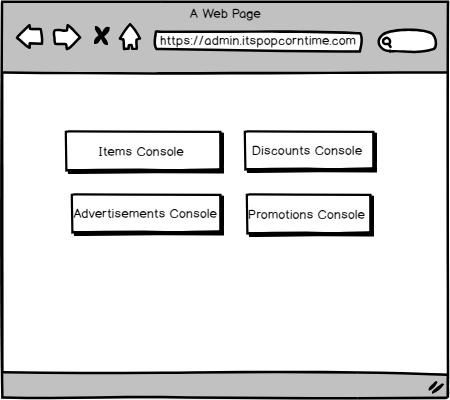
**Menu Bar and My Account Screen**

The Menu bar will appear on all screens as stated above. On the My account page the user will be able to view and edit their account details except their name (email, password, and Mobile number only) that they input into the application when registering. Below is an example of what the page may look like:



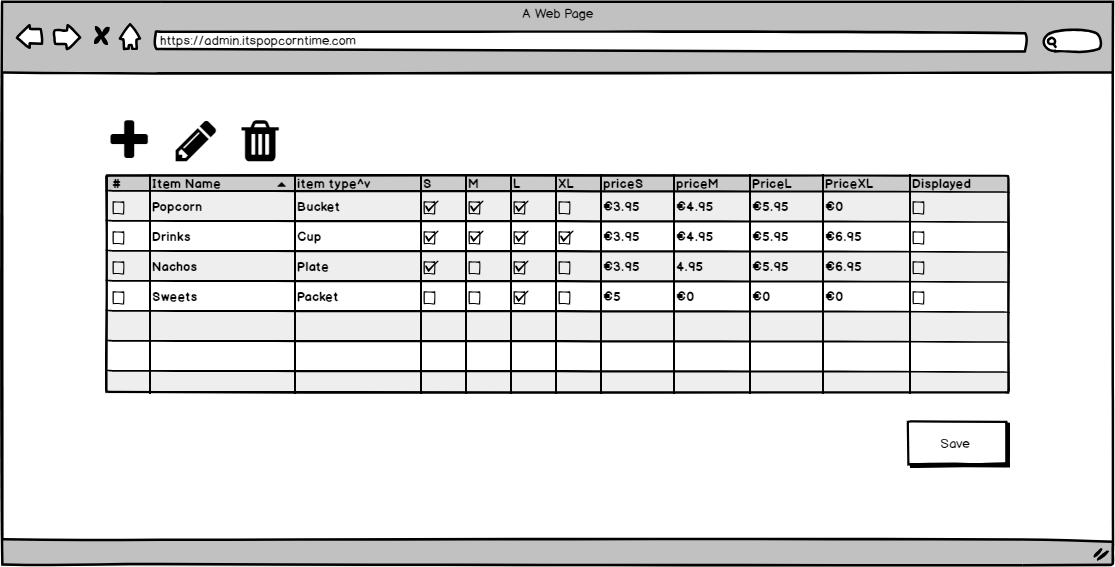
**Administrator Console**

This may only be accessed using a computer. It will be used to manage some items, Advertisements and Discounts and Promotions. Each management screen will display a List. Since all will be the same format I will use the Items Management as an example.



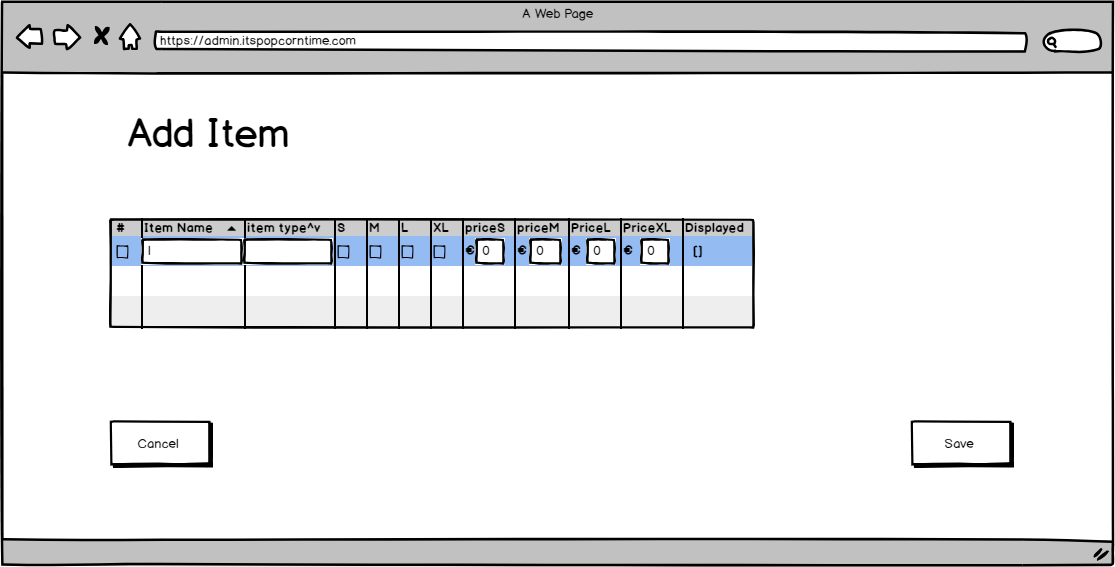
**Item Management List**

This will display a list of all items with various options. Here the administrator can Add, Edit or Remove items.



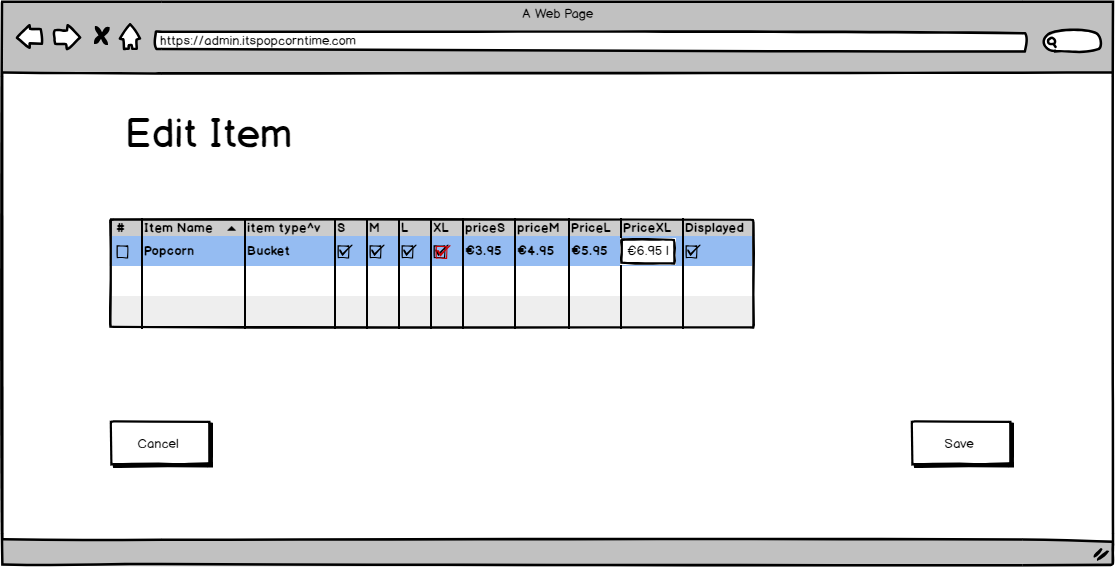
**Add an Item**

If the administrator clicks on the plus on the previous screen they will then be given an input to add a new item. Clicking save will save it to the database.



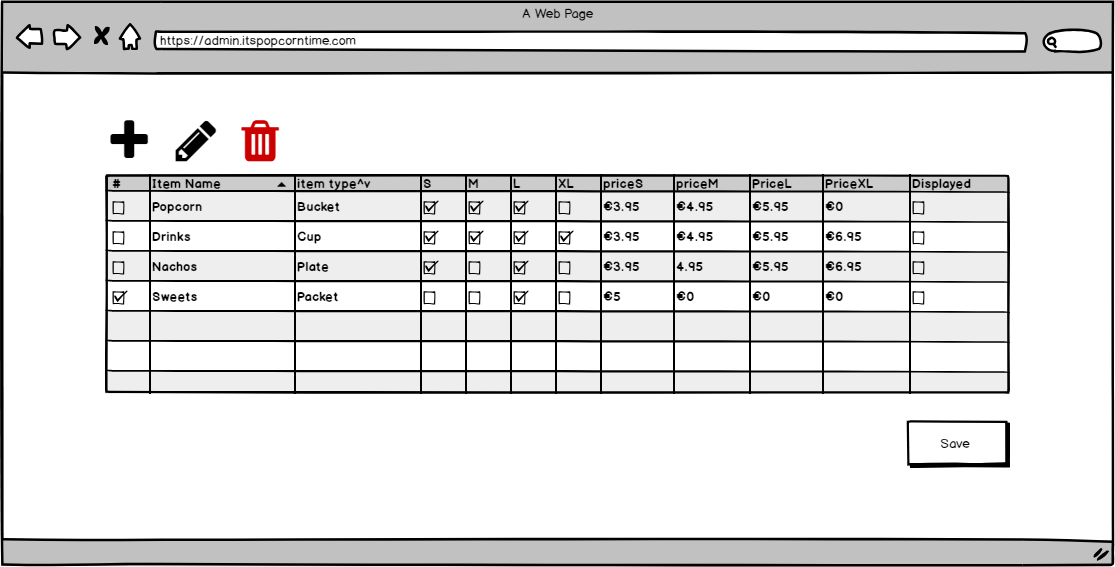
**Edit an Item**

If the administrator checks the checkbox in the first column. Then the administrator clicks on the pencil icon in the list view, they can edit a selected item. They will then see this screen. They will be able to edit multiple items.



**Remove an Item**

If the administrator checks the checkbox in the first column. Then the administrator clicks on the bin icon on the list view, they can then delete an item once removed and they press save it will be deleted from the database. But if they edit the item and uncheck displayed it will be removed from the users view but remain in the database.



## Implementation of the Prototype

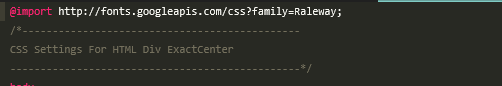
For the prototype I focused on implementing features and the design then debugging as I went along. I focused on trying to get features fully working rather than implementing everything, so I built only the user vie of the application. On top of this I also had setbacks and complications, so it is far from perfect, but it is functioning in a minimal capacity.

### Prototype Code

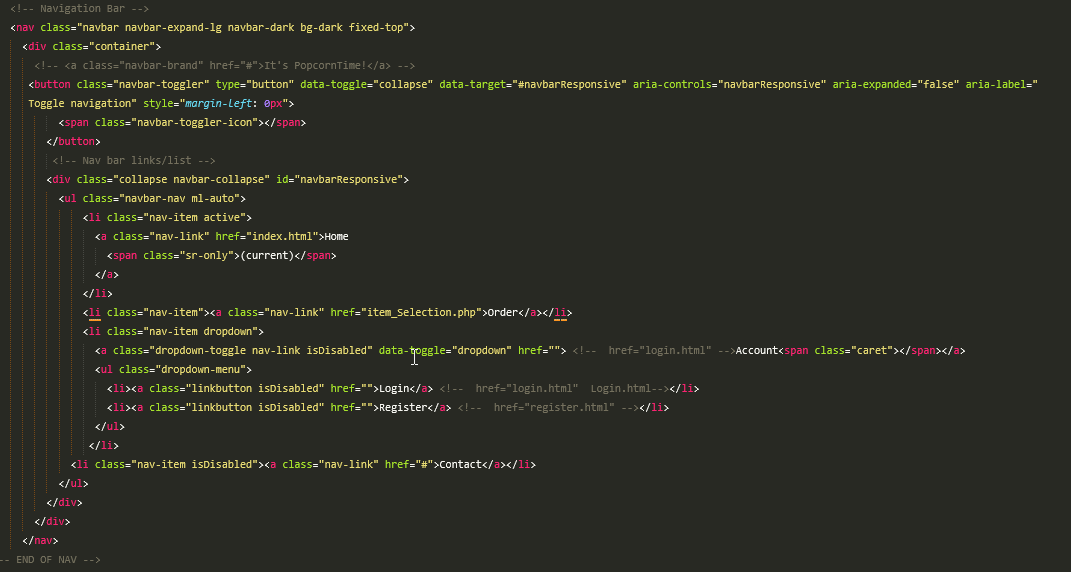
**Bootstrap**



Above I link the bootstrap.min.css in to the file for use. The min file is a file where all the whitespaces are removed to make the file size smaller and faster.

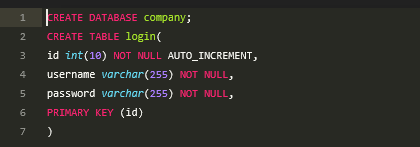


Above I import a font for use within the CSS file, which is then used to display on the index file.



Above I have implemented the Navigation bar to transform in to a hidden menu, when in mobile view. It will, when clicked expand displaying the options.

**Database SQL**



This is the SQL Script I used to create the database. I did this using the XAMPP Localhost Webserver and the built in myphpadmin tool. For the prototype I created a database called company with a table called login. The login table had fields for an: ID, Username and Password. As I didn’t have time to incorporate the password hashing the password is stored in plain text.

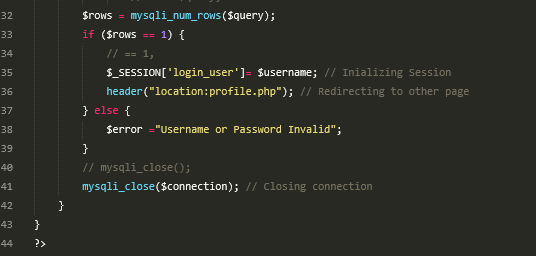
**PHP**



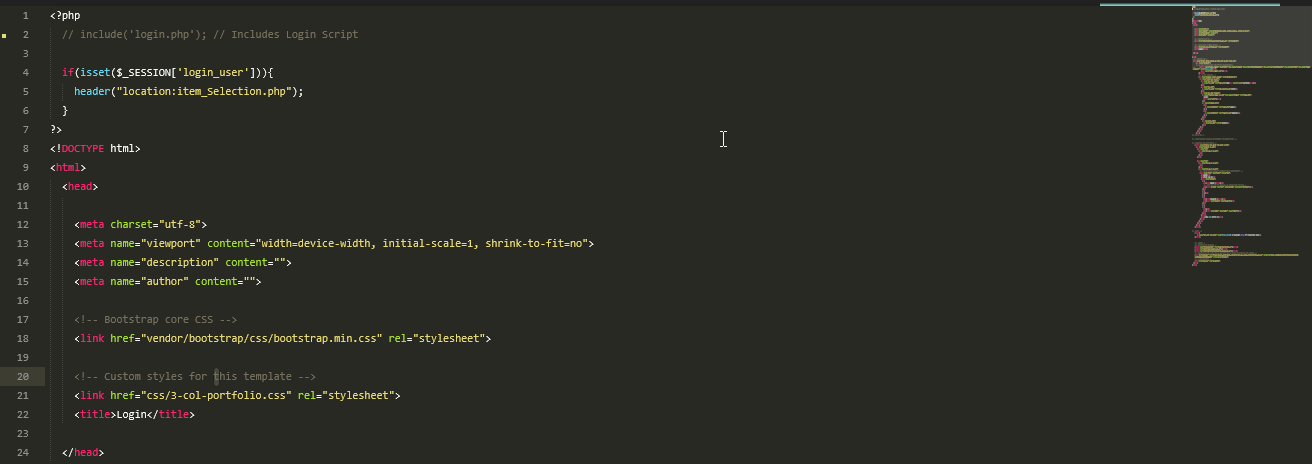
Above is the session.php file. This file is used to create a connection between the application and the database and to create a user session, that is stored in cache. It also has the SQL query to check the database for the user, so they can login to the application. Once this is complete the connection to the database is closed. In the file there are some commented out statements I used while testing to ensure the session data was being stored and the connection was working.



Above is the file that starts the user session. The variables are initialized creating a space for the username and password input after some slight protection against SQL injection using “striplashes” which removes backslashes from the string. And when the submit button is pressed the data is submitted and checked against the data in the database.



Also, if the user input credentials do not match the data in database, then the above error is displayed to the user.

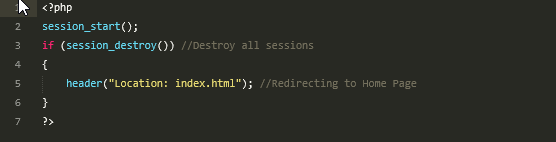


Above is the login\_index.php file. In the head of the file I include a script to take the username and password the user input and to set them in the session.php file which runs the script to check the database and login.

If details are correct user is redirect to the item\_selection.php file when they log in.



In the above is the login form. Here the user inputs their username and password and is then checked in the database through the session file. If the details are wrong though the user is shown an error on the screen. This is taken from the default PHP defined SQLi query error.



Above is the script run when the logout link is hit. It destroys the user session and redirects to the index.html page.



Above is in the Item\_selection.php file. At the start of the document I include the session.php, this means it can be used within this file.

Under the div id profile, I then use the information from the session file to display the username, which in this case is the email so the page shows the user “Welcome: [shane@me.com](mailto:shane@me.com)”. I also create a link to run the script to log the user out.

**PayPal API**



In the above I link the “sandbox.PayPal.com” environment. I set the cart value as well as the Item Options price and the currency that is sent to PayPal. All input types are hidden from user. This is done with all items the user can select.

At the end there is an “add to cart” button that is designed like a standard PayPal button. When pressed it the sends the information to the PayPal cart with the options and the price. Unfortunately, due to time constraints I had to implement an individual button for each in the prototype. Also, for the same reason I used the PayPal cart instead of a locally displayed once that I would send to PayPal.



I also created a separate button at the end for user to click to view the PayPal cart. This does not add anything it simply links to the cart.

**JS Script Links**

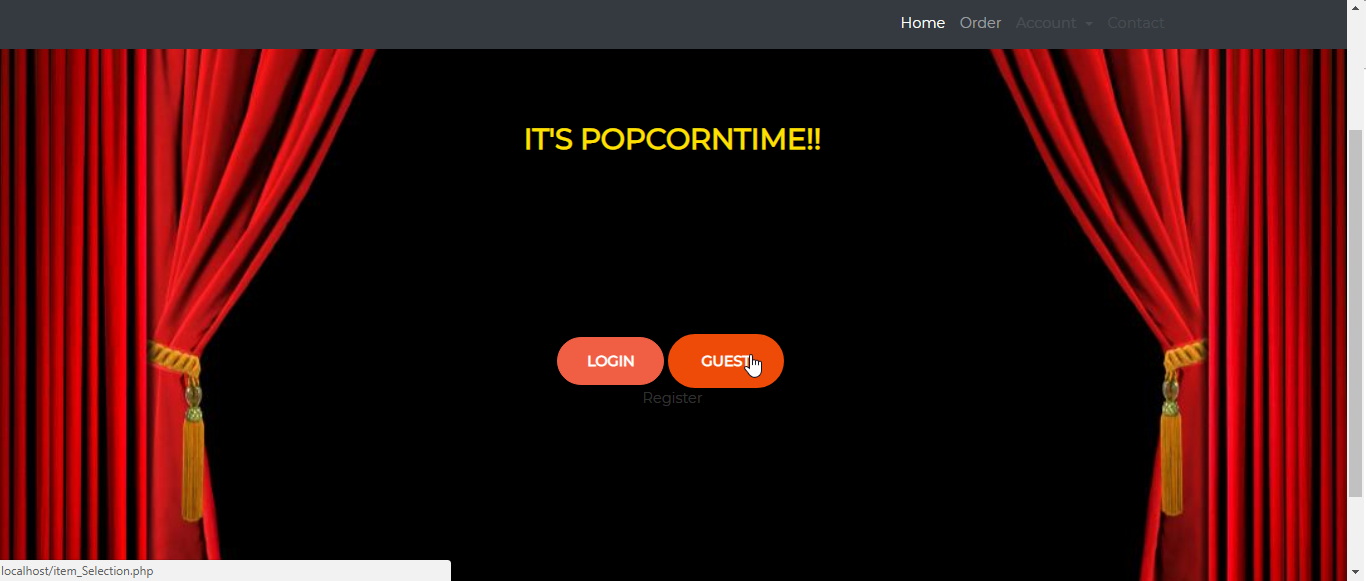


The above links to the Bootstrap JS, the JQuery JS and finally the custom JS file, although the latter is not currently used.

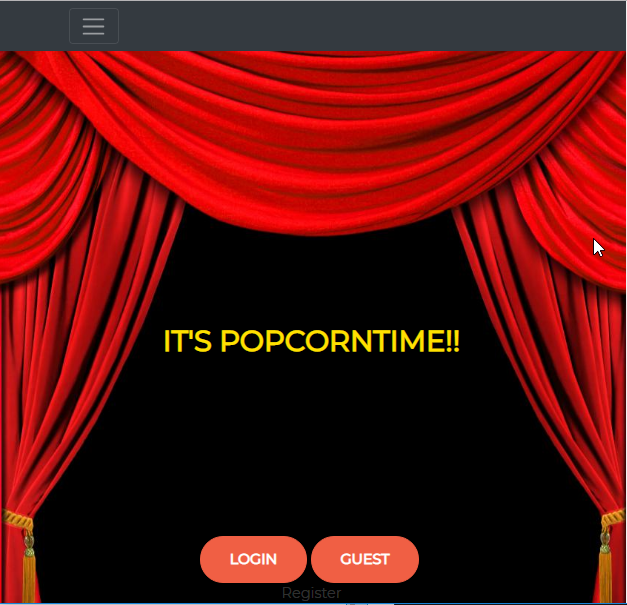
### Prototype Graphical User Interface (GUI) Layout

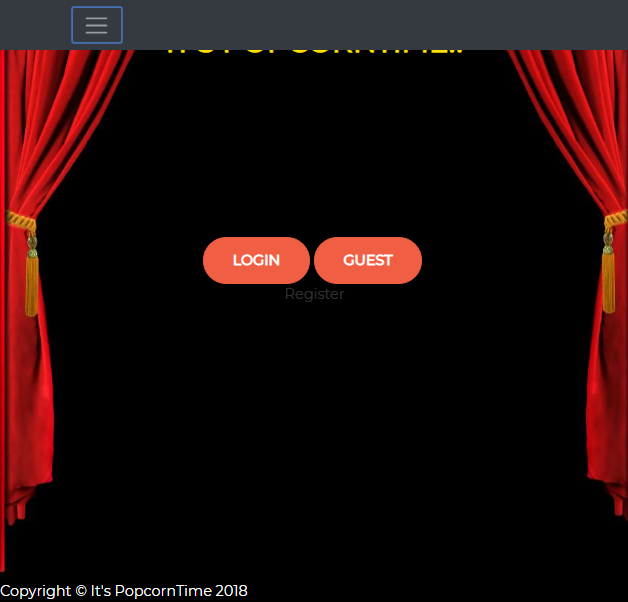
**Index**

This is the first page the user will see it has some basic CSS effect and JavaScript on scroll over on the buttons to make them larger. I also added a disable link when scrolling over a disabled link as shown below.



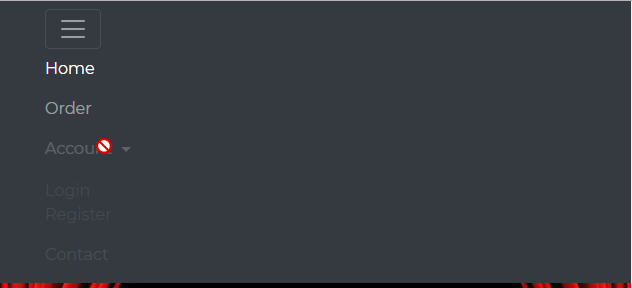
**Mobile Index**



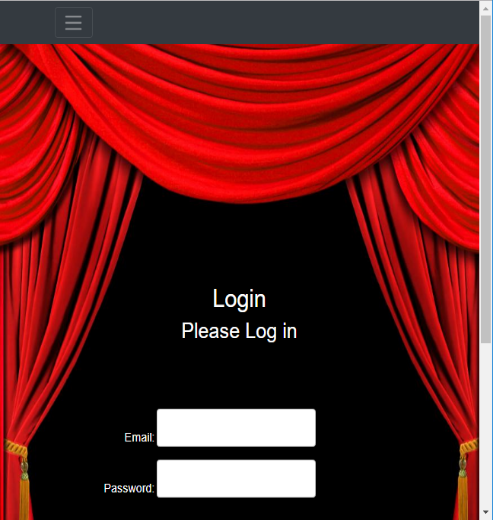


**Mobile Nav Bar**

When viewed on mobile the navigation bar collapsed and changed using the Bootstrap CSS to a 3-bar menu button. When the menu button was clicked it would display the links. For account it was a sub collapsible section so when Account was pressed it would display the Login and Register Link. These are broken links I left in to maintain the feel of the application.

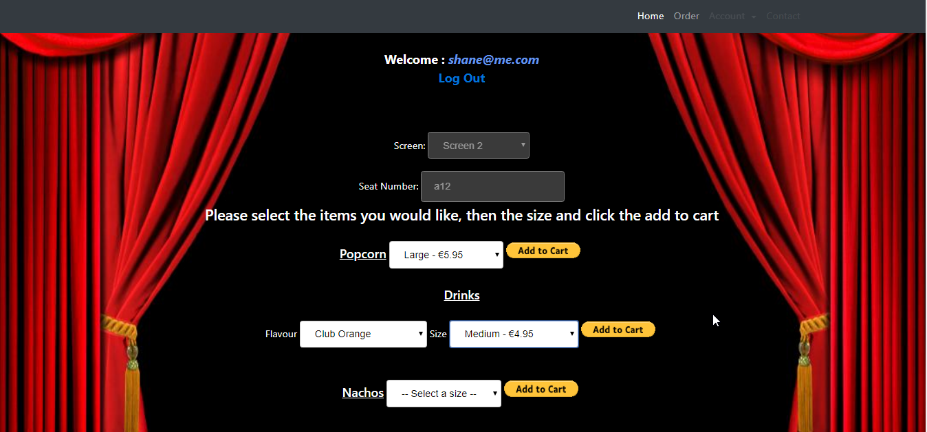


**Mobile Login**

This is the log in page in mobile view the view on a desktop is much the same as like the index page. The user is manually created in the database and the credentials are in the guide. While I tried to implement a registration, I could not get it working. But the login does work and is created using PHP, MySQLi and it also creates a user session. Also, there is some basic validation using HTML 5 input for the Email will show an error if not an email address and if the password is a password input. When trying to log in if the password is wrong an error is displayed.

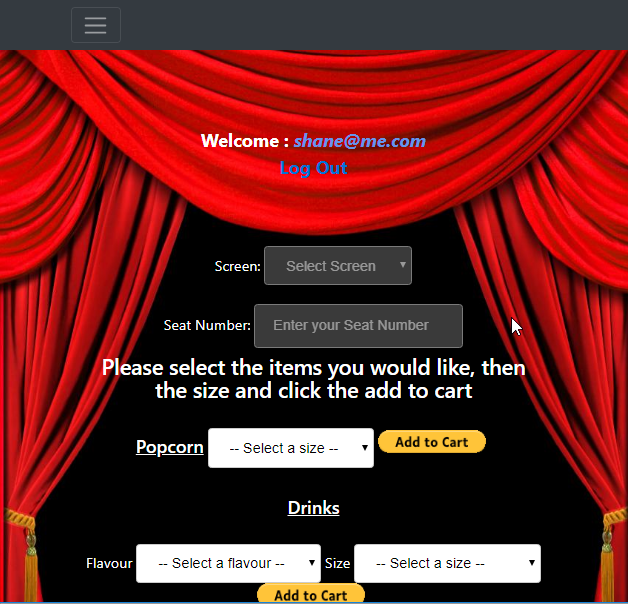
**Items Selection**

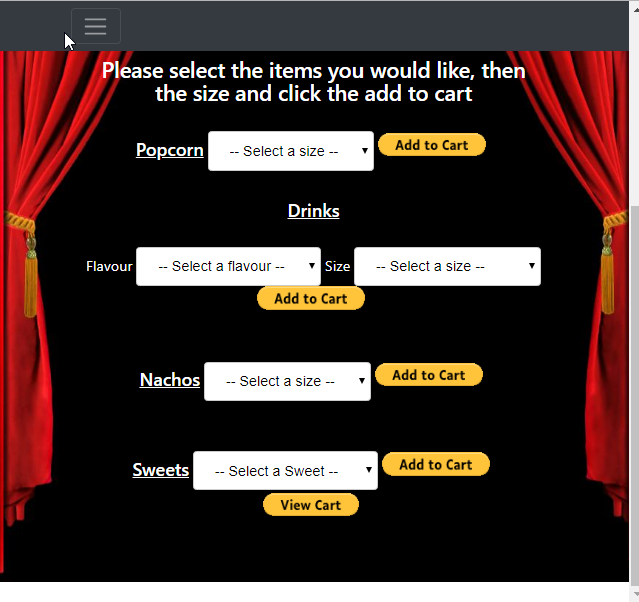
This is how the item selection page displays on a computer browser. Again, not perfect but functions. Currently the Screen and Seat Number inputs are not saved anywhere, and just for show. Each Add to cart is from the PayPal API and link to sandbox.PayPal.com. Unfortunately, they all currently work individually. Say you select a popcorn and the size and click add to cart, it then opens a new tab displaying a cart in PayPal with the item, presently you must close the cart tab and then go back to add the next item. Which will add the 2nd item to the PayPal cart.



**Mobile Items**

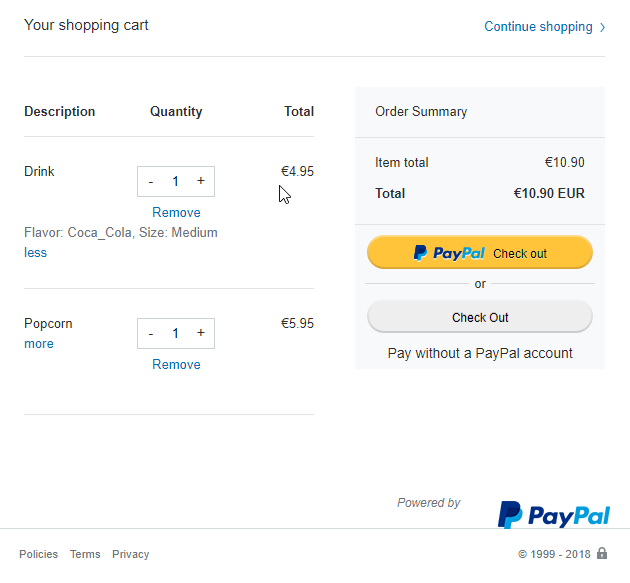
This is how the item selection page displays on the mobile after Login, the session info is then Echoed to display the “Welcome: …”. There is also a button to log back out. Again, some issues when user is a guest without logging in a php error is echoed to the top of the page under the nav bar which causes some conflicts with the CSS.



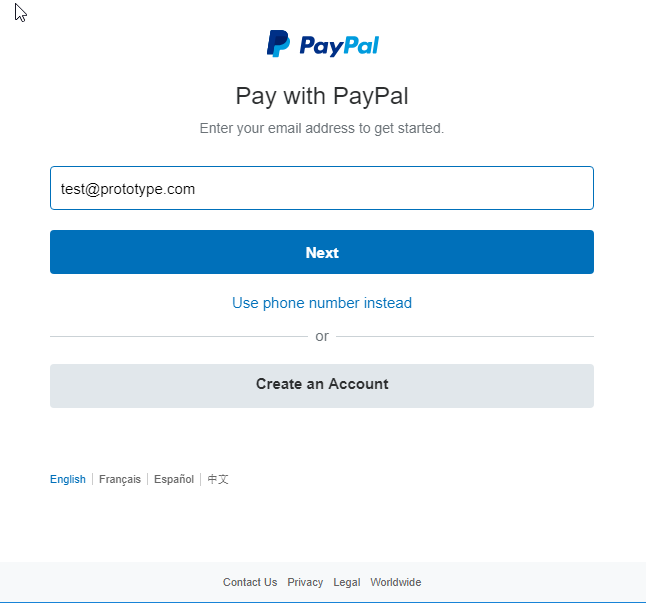


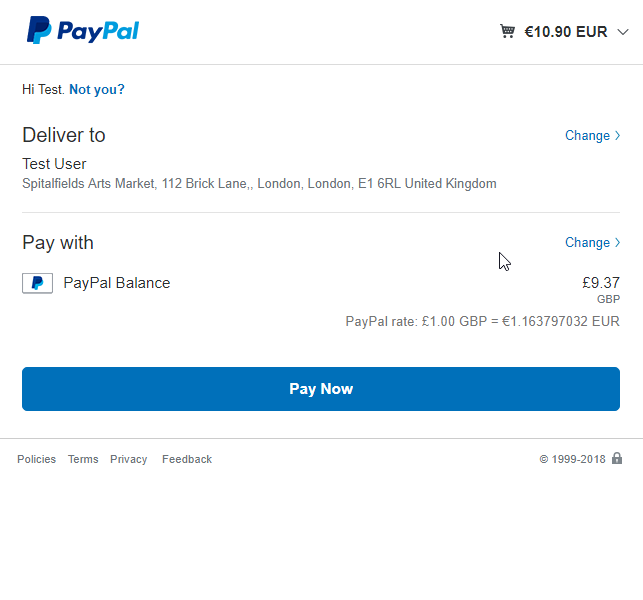
**PayPal Cart**

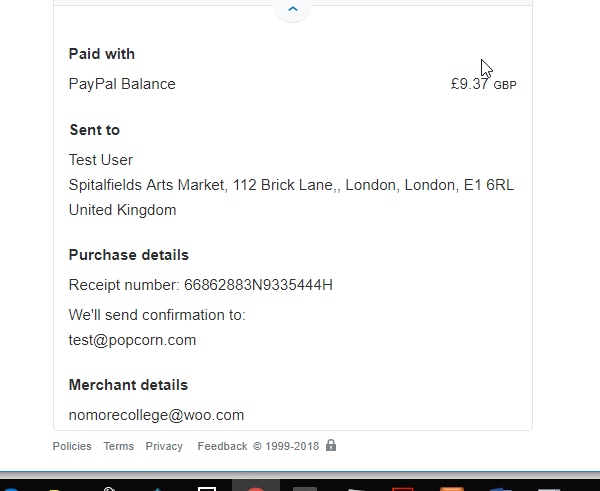
This is the cart displayed on the sandbox.PayPal.com

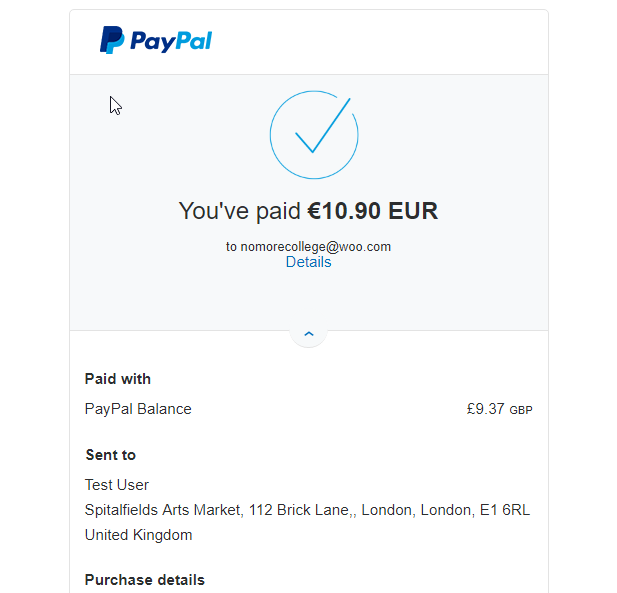


**PayPal Payment Process**









## Testing

I did Unit Testing throughout development of the app. This means I added a feature/aspect, debugged a feature and fixed the issue before moving on to another feature/aspect. The two parts that were most difficult to test were the CSS and Bootstrap conflictions and PHP.

For the CSS and Bootstrap conflictions, I was able to use the developer tools built in to Google Chrome. If there was a default Bootstrap class or id I would remove the conflicting part, in the development tools and then see the affect it would have on the browser without altering the file. Once I was sure I had it figure out I then went back to the Project files and added or removed to whatever class or ID, on the specific file and line number which I was shown in the development tools.

I also used a few other extensions in chrome to complement the design, like Page Ruler, ColorZilla and React Debugging tool.

As for the PHP, that was more difficult. I had created php features in a separate pure PHP projects using online tutorial guides. I then tried merging them before implementing to the prototype (this is also the reason why the project on GitHub is called NewUITest).

When I had first created the login, it hadn’t been working correctly. I had tried adding recommended default MySQL error catchers, but it appeared to be working correctly. The session was created and displaying. I used several echo and print statements to display values to check they were there and being stored. Once it was fixed, I implemented it in to my prototype.

I tried to implement a registration feature. I ended up spending a lot of time going through the apache logs looking at time stamps and debugging every instance or error I found. Until everything seemed to be working and no more errors displayed or in the logs.

The main error was the connection to the database was there when reading from for the login page, but it was not writing data to the database. Instead it would create an empty row in the database. Unfortunately, I never fixed this error. For a brief time, I managed somehow to have it adding 2 rows to the database one empty row and a row with data, but it nullified the Login. Given these issues I never implemented the feature in to the prototype.

## Customer testing

For user testing I had to create a guide as to how to set up and run the prototype. I also had to have them test on a computer as PHP runs on a server, so the testers had to set up a localhost server. In the guide as I had in development I had them set up XAMPP.

Once the user was set up and had the application running I had them go through the Application as they would a normal Application and then answer questions. The majority of users where non-technical but were able to spend the time setting up the environment. Some were used to online shopping others rarely do. They were also of an age range of 22 – 67 years old. Again, this was important for me as I wanted to be sure all visitors to the cinema would be able to use the application.

## Evaluation and Testing results

All users stated it was easy to navigate the application, that they thought it looked good and that the pricing was clear.

As previously stated all used a laptop or desktop to test the application.

A few had errors when trying to log in to PayPal to complete the payment process.

Some saw the email error or the password error on log in.

The technical testers found bugs when the guest clicked in there was no session information so a php error was displayed under the nav bar which affected the CSS on the items page. And some minor display errors which I was able to fix the php error, again due to time I have had to leave in.

In terms of functionality user’s average rating was 62.5% positively.

In terms of the understanding of the elements of the application user’s average rating was 62.5% positively

In terms of reasonability of the time to use the application from entering to competing user’s average rating was 67.5% positively.

In terms of how secure they felt the application was 65% felt it was secure.

When asked what other features they would like to see in the app the majority suggested being able to sign in with Facebook.

Another suggestion was having a wider selection available, being able to pay by cash upon delivery and have discounts.

# Conclusions

From the very start this project was an uphill battle. As I had started to develop the application a lot of changes had been made. I had initially planned to develop in C# with a code first database using visual studio, then after spending months obsessing, trying to get a JavaScript seating map working I ended up thinking I would develop in android natively. That turned out to be a completely different and frustratingly difficult environment to set up without causing issues for other projects.

In the end I settled on the core web development languages with a plan to use Apache Cordova to compile down in to native android and iOS applications.

Again, this changed as I begun to develop and found gaps in my knowledge that delayed me in developing the prototype as I had to refresh everything as I hadn’t done ‘pure’ web development since 2nd year.

I done a course on Udemy.com on web development and again had to refresh my knowledge on software engineering, project management and PHP which I had again previously done through independent learning course on codeacademy.com.

Each time becoming obsessed and spending more time than I should have on each, all outside of working full time. The other distraction. The prototype itself and timescale then influenced me more meaning it is merely a proof of concept more than a prototype both creating more ideas for functions or design or things having to be removed and focused on fully.

Whilst developing I had researched and attempted a number of things for example the failed JS Seating Map, and Registration features, which I spent time on but removed or didn’t include as I felt that a I would rather have something working correctly than only the idea being featured. Although again due to time restraints and delays some error found during user testing have been left in.

Overall it is okay, but I feel had life not got in my way at times then maybe it could have been done better. But I am happy with what I had got done overall.

# Further development

For further development I would have a local cart where users could edit, remove items, the admin console the discounts and all that I had planned in the first place.  
Once the complete prototype is done it could be trialed in a single cinema.

As I said in the research I had come across an App in America doing the same thing for concession in stadiums, so I feel this could easily be scaled to that size.  
I also feel another possible market is theatres. Whether the abbey, smock alley or the grand canal. In these locations it could also offer non-concession items, playbills or other merchandise.

I would also implement a pre-pay and collect feature. Also, I feel this would open the Irish market to the luxury dining and cinema experiences like in Australia that I mentioned during my research.

# References

Agarwal, N. (2018). PHP Login Form with Sessions | FormGet. [online] FormGet. Available at: https://www.formget.com/login-form-in-php/ [Accessed 17 Aug. 2018].

Amctheatres.com. (2018). Order Ahead. [online] Available at: https://www.amctheatres.com/food-and-drink/order-ahead [Accessed 17 Aug. 2018].

Balsamiq.cloud. (2018). Balsamiq Cloud. [online] Available at: https://balsamiq.cloud/ [Accessed 17 Aug. 2018].

Bcrypt.sourceforge.net. (2018). Bcrypt - Blowfish File Encryption. [online] Available at: http://bcrypt.sourceforge.net/ [Accessed 17 Aug. 2018].

Bitnami.com. (2018). Bitnami for XAMPP Application Modules. [online] Available at: https://bitnami.com/stack/xampp [Accessed 17 Aug. 2018].

Codecademy. (2018). Learn to code - for free | Codecademy. [online] Available at: https://www.codecademy.com/ [Accessed 17 Aug. 2018].

Css-tricks.com. (2018). CSS Tricks. [online] Available at: https://css-tricks.com/almanac/ [Accessed 17 Aug. 2018].

Css-tricks.com. (2018). You are being redirected.... [online] Available at: https://css-tricks.com/how-to-disable-links/ [Accessed 17 Aug. 2018].

Designs, A. (2018). Showtime font by ARRF Designs - FontSpace. [online] FontSpace. Available at: http://www.fontspace.com/arrf-designs/showtime [Accessed 17 Aug. 2018].

Designshack.net. (2018). Customize Your Own MailChimp E-Mail Newsletter Signup Form | Design Shack. [online] Available at: https://designshack.net/articles/css/custom-mailchimp-email-signup-form/ [Accessed 17 Aug. 2018].

Developer.PayPal.com. (2018). Add to Cart: Sample HTML button code - PayPal Developer. [online] Available at: https://developer.PayPal.com/docs/classic/PayPal-payments-standard/integration-guide/html\_example\_cart [Accessed 17 Aug. 2018].

Developer.PayPal.com. (2018). Button Manager API - PayPal Developer. [online] Available at: https://developer.PayPal.com/docs/classic/button-manager/integration-guide/ [Accessed 17 Aug. 2018].

Developer.PayPal.com. (2018). PayPal Developer Documentation - PayPal Developer. [online] Available at: https://developer.PayPal.com/ [Accessed 17 Aug. 2018].

Engadget. (2018). MasterCard's QkR mobile payment system enters trial in Australia. [online] Available at: https://www.engadget.com/2012/01/27/mastercards-qkr-mobile-payment-system-enters-trial-in-australia/?guccounter=1 [Accessed 17 Aug. 2018].

FanFood, I. (2018). FanFood - Order your favorite concessions directly from your phone. [online] Fanfoodapp.com. Available at: http://www.fanfoodapp.com/ [Accessed 17 Aug. 2018].

Getbootstrap.com. (2018). Getting Started. [online] Available at: http://getbootstrap.com/getting-started/#grunt [Accessed 17 Aug. 2018].

Iconexperience.com. (2018). [online] Available at: https://www.iconexperience.com/\_img/o\_collection\_png/green\_dark\_grey/512x512/plain/drink.png [Accessed 17 Aug. 2018].

Ie2.php.net. (2018). PHP: Manual Quick Reference. [online] Available at: http://ie2.php.net/manual-lookup.php?pattern=mysqli.mysqli&lang=en&scope=404quickref [Accessed 17 Aug. 2018].

it, m. (2018). mysql\_connect(): No connection could be made because the target machine actively refused it. [online] Stack Overflow. Available at: http://stackoverflow.com/questions/21987746/mysql-connect-no-connection-could-be-made-because-the-target-machine-actively [Accessed 17 Aug. 2018].

Maxcdn.bootstrapcdn.com. (2018). Maxcdn bootstrap cdn. [online] Available at: https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-alpha.6/css/bootstrap.min.css [Accessed 17 Aug. 2018].

Murshed, S. (2018). Encrypt and Decrypt Data with C# - CodeProject. [online] Codeproject.com. Available at: http://www.codeproject.com/Articles/14150/Encrypt-and-Decrypt-Data-with-C [Accessed 17 Aug. 2018].

Neagoie, A. (2018). The Complete Web Developer in 2018. [online] Udemy. Available at: https://www.udemy.com/the-complete-web-developer-in-2018/learn/v4/overview [Accessed 17 Aug. 2018].

Paletton.com. (2018). Paletton - The Color Scheme Designer. [online] Available at: http://paletton.com/#uid=1000u0kllllf5HsidsKotdWrA6x [Accessed 17 Aug. 2018].

PayPal.com. (2018). PayPal UK: Pay, Send Money and Accept Online Payments. [online] Available at: https://www.PayPal.com/cgi-bin/webscr [Accessed 17 Aug. 2018].

Php.net. (2018). PHP: Encrypted Storage Model - Manual. [online] Available at: http://php.net/manual/en/security.database.storage.php [Accessed 17 Aug. 2018].

Php.net. (2018). PHP: Mhash Functions - Manual. [online] Available at: http://php.net/manual/en/ref.mhash.php [Accessed 17 Aug. 2018].

Php.net. (2018). PHP: Password Hashing - Manual. [online] Available at: http://php.net/manual/en/faq.passwords.php [Accessed 17 Aug. 2018].

Php.net. (2018). PHP: password\_hash - Manual. [online] Available at: http://php.net/manual/en/function.password-hash.php [Accessed 17 Aug. 2018].

Retaildive.com. (2018). AMC Theaters offers mobile concession ordering through Atom Tickets app | Retail Dive. [online] Available at: https://www.retaildive.com/ex/mobilecommercedaily/amc-theaters-offers-mobile-concession-ordering-through-atom-tickets-app [Accessed 17 Aug. 2018].

SeatServe - Official Website. (2018). SeatServe - Stay Seated. In-Seat Delivery at Your Stadium.. [online] Available at: https://seatserve.com/ [Accessed 17 Aug. 2018].

Start Bootstrap. (2018). Three Column Portfolio - Free Bootstrap Template. [online] Available at: https://startbootstrap.com/template-overviews/3-col-portfolio [Accessed 17 Aug. 2018].

TODAY.com. (2018). Want to skip the line for movie-theater snacks? There's an app for that. [online] Available at: https://www.today.com/series/wired/want-order-movie-theater-snacks-advance-theres-app-t61581 [Accessed 17 Aug. 2018].

W3schools.com. (2018). Bootstrap Navigation Bar. [online] Available at: https://www.w3schools.com/bootstrap/bootstrap\_navbar.asp [Accessed 17 Aug. 2018].

W3schools.com. (2018). JavaScript Statements. [online] Available at: https://www.w3schools.com/js/js\_statements.asp [Accessed 17 Aug. 2018].

W3schools.com. (2018). PHP 5 Echo and Print Statements. [online] Available at: https://www.w3schools.com/php/php\_echo\_print.asp [Accessed 17 Aug. 2018].

W3schools.com. (2018). PHP 5 include and require. [online] Available at: https://www.w3schools.com/php/php\_includes.asp [Accessed 17 Aug. 2018].

W3schools.com. (2018). PHP mysqli\_real\_escape\_string() Function. [online] Available at: https://www.w3schools.com/php/func\_mysqli\_real\_escape\_string.asp [Accessed 17 Aug. 2018].

W3schools.com. (2018). SQL Tutorial. [online] Available at: https://www.w3schools.com/sql/ [Accessed 17 Aug. 2018].

www.tutorialspoint.com. (2018). Cordova First Application. [online] Available at: https://www.tutorialspoint.com/cordova/cordova\_first\_application.htm [Accessed 17 Aug. 2018].

# Appendix

## Project Proposal

### Introduction

This project proposal is a document which outlines the proposed objectives, background, solutions for the chosen project. It will be used to identify and define what the project proposal is in its entirety.

The outline of the project is to create a web based application solely for the use of customers in a cinema. It will allow the user to:

* Register for an account. Storing their personal information such as:
  + Full name
  + Email address
  + Contact number
  + Home address
  + Credit Card details
* Then use the account to purchase Food items sold by the Cinema using their smart phone, and selecting the screen and exact seat they are sitting in.
  + The food will be then delivered to the customer by an usher.
* There will also be a guest account setting, for one-times users who do not wish to setup an account, but will be able to pay using their credit card.

My idea is for an Web Application App for people to use before or in the cinema.  
In the App the will be able to select and pay for food from the comfort of their cinema seat or in advance, which the usher will deliver directly to them, or when they arrive at the movie.   
This is so that that the users won’t have to miss part of the film if they need to run out for food or drink halfway through or won’t have to waste time queuing.   
Hopefully this technology could, at a later stage, also provide a ticket purchasing system.

The users will be able to order through the App by setting up an account and log in, selecting the food type, and portion size, that the cinema provides e.g. two Large Popcorn. Or by selecting and ordering a deal on offer e.g. Medium popcorn and drink for €10.  
They will select the time and screen of the movie they are in/going to, and then click their exact seat location on a seating map.  Then they will go through a secure payment process and pay using a saved card option or one-time card payment. I also looked in to using NFC payments using a Credit Card with "contact-less" as a possibility. Their order will be sent to the cinema counter to be filled and delivered.

Their account information will securely store their name, address, contact information and credit card details. This and the payment system will be my main focus.

### Background

The idea for my application came whilst visiting the cinema with a friend who is in a wheelchair. Due to complications with travel that day to the cinema we were running late, but we had pre-booked tickets, so we had rushed in to the cinema. We got to our seats just as the movie began, but although we had planned to get food we bypassed the counter because we were running late. I then noticed a couple arrive with popcorn and drinks 15 minutes in and had missed the beginning of the movie.

This technology is similar to apps that are currently being used in Cafes e.g. Starbucks, which provide an ‘order and collect’ service via app, but is still a relatively new concept, and it has not been tried in cinemas here in Ireland. Although I thought it was innovative I discovered during my research this technology is currently being developed and tested in cinemas elsewhere.

There are currently three cinema chains in North America developing and testing a similar concept for use in their cinema’s. Two major cinema Chains in the US AMC Theatres and Regal Entertainment are rolling out this technology out as an ‘order and collect’ by having the prepay and collect at a pickup station.   
In Canada Landmark Cinemas and expected through Cineplex Cinemas using an app, customers will be able to view movie listings, buy tickets and pre-purchase popcorn, drinks and other concessions.  
Also in Australia MasterCard have created an application called QkR which uses NFC or QR codes where the customer can scan the arm rest of the chair to order and pay for food.

### Objectives

My objective is to create a web based application for consumers to order food from their seat within the cinema screen or to pre-order before they arrive.  
This will be achieve using a web based application called “**It’s POPCORNTime**”.

The web application will require Users to register for an account which will save personal information such as their full name, email, contact number, address and credit card information. They will then log on through a

The main focus for this application will be based on:

1. the Security of the Web Application against attacks,
2. the authentication of the users
3. And the secure encryption of the database.
4. And implementing secure payment process.
5. **The Security of the Application**

According to OWASP, 60% of Web based applications are vulnerable to SQL injection, Cross-Site Scripting and other forms of attacks.  
The intention is to make sure that “*It’s Popcorn Time!”* is secure against such attacks and will be thoroughly tested to detect and eliminate these vulnerabilities.

1. **The authentication of Users**

This will be achieved by users being required to verify their email address after registration. They will also have to set security questions and answers for password recovery during the registration process.

1. **Encryption of the database**

This is done to secure the most sensitive user data such as users address, contact number but most importantly their credit card information.

1. **Secure Purchasing Process**

Ideally there Verified by visa will also be implemented during the purchase process.

The challenge of developing this application, will be in implementing the above, as I have had little experience in the secure development of any application. Although I have some previous experience implementing a log in with simple authentication. I may also try setting a log in with google or Microsoft account features.

### Technical Approach

I will be researching methods and different types of Encryption for the Database. I plan to research and look in to tools for encryption, primarily by online research. I plan to research how to create and implement a secure User Authentication System that I will use in the Web Application, like verifying email after registration, protecting from SQL Injection and Cross-Site Scripting attacks. I also plan on building the application by using what I will learn in my secure application programming module.

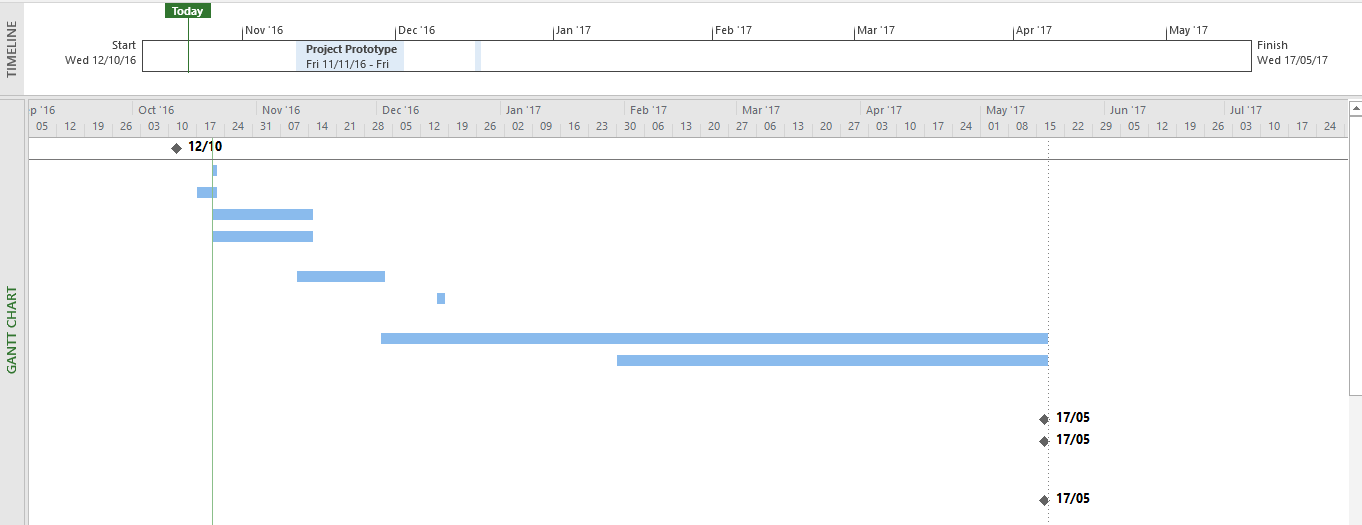
### Special resources required

Due to only starting to fully research the project this week after I received approval to continue I am uncertain of the types of technology that I will need to use for my project. Since I am familiar with using Visual Studio to build a web application from a previous module I think I will probably use this,

## Project Plan

Gantt chart using Microsoft Project with details on implementation steps and timelines

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task Mode | Task Name | Duration | Start | Finish |
| Auto Scheduled | Project Pitch | 0 days | Wed 12/10/16 | Wed 12/10/16 |
| Auto Scheduled | Project Approval | 1 day | Fri 21/10/16 | Fri 21/10/16 |
| Auto Scheduled | Project Proposal | 5 days | Mon 17/10/16 | Fri 21/10/16 |
| Auto Scheduled | Project Research | 17 days | Fri 21/10/16 | Mon 14/11/16 |
| Auto Scheduled | Project Requirement Specs | 17 days | Fri 21/10/16 | Mon 14/11/16 |
| Auto Scheduled | Project Prototype | 16 days | Fri 11/11/16 | Fri 02/12/16 |
| Auto Scheduled | Midpoint Presentation | 2 days | Fri 16/12/16 | Sat 17/12/16 |
| Auto Scheduled | Building Application | 120 days | Fri 02/12/16 | Wed 17/05/17 |
| Auto Scheduled | Final Project Hard Copies Documentation | 78 days | Mon 30/01/17 | Wed 17/05/17 |
| Auto Scheduled | Showcase Materials | 0 days | Wed 17/05/17 | Wed 17/05/17 |
| Auto Scheduled | Software and Documentation Upload | 0 days | Wed 17/05/17 | Wed 17/05/17 |
| Auto Scheduled | Project Presentation | 0 days | Wed 17/05/17 | Wed 17/05/17 |



## Monthly Journals

### Reflective Journal September 2016

Student name: Shane O’Brien

Programme (e.g., BSc in Computing): Cyber Security Stream

Month: September 2016

**My Achievements**

The past few weeks, I came up with an idea for my software project to develop an Android application for cinema customers to use to order and pay food without having to walk out of the movie. I also planned to add a pre-order and pay for customers to use before arriving at the cinema. I also had planned to build a custom payment system in keeping with my stream of Cyber security.

**My Reflection**

I thought I had come up with a unique and innovative idea, but whilst doing research I discovered that there are similar ideas/products being developed in North America and Australia.

I also looked in to using NFC payments using a Credit Card with ‘contactless’ but due to some security concerns about stealing credit card information I decided against it.

I done the project pitch on Wednesday 5th OCT, with Keith Maycock, Adrina Cris and Joe Molumby.

I was a little nervous and I don’t think I explained my idea very well. I surmised my idea for them explaining what it was for and that I had intended to build a custom payment system and develop it with using what I learned in Secure Application Programming. I also forgot to mention my research in to NFC payments. They said my idea wasn’t complex enough to be a 4th year project and that they would discuss it.

Since I am awaiting the final decision on my project idea, I have been unable to start any real work on my software project due to this I have not been assigned a supervisor.

### Reflective Journal October 2016

Student name: Shane O’Brien

Programme (e.g., BSc in Computing): Cyber Security Stream

Month: October 2016

**My Achievements**

I got my project idea approved with some slight enfaces in regard to my stream with help from Sara Kadry, and Eugene McLaughlin. I was also Assigned Eugene as my Supervisor I uploaded my project proposal and currently working on the Requirements Specification for my project.

**My Reflection**

After the project pitch I was told I would find out at a later point if my idea was approved. Unfortunately, due to illness I missed an extra class and meeting held for the cyber security stream on the following Monday 10/10. I then emailed Eamon on Wednesday and also spoke to him in person he suggested I talk to the Cyber Security lectures Eugene and Sara for help. I was under the impression that I needed their approval to continue with my project Idea they both gave me advice and signed off on my Idea.

Although I discovered in class on Monday 17/10, when I went to inform Eamon, that they had approved my idea and he informed me that I wasn’t on the list of people whose ideas were rejected, and my idea had been approved from the very beginning. This cost me time to work on the Project proposal which was due on Friday 21/10.

Under a lot of stress due to a separate assignment due for AI and the time constraints I managed to throw together a project proposal and upload it on time. On the Saturday 22/10 I was assigned Eugene McLaughlin as my Supervisor. Unfortunately, I have yet to speak to him as I missed a Wednesday class and he was absent due to personal reasons on the Thursday.

I emailed him on the Friday about arranging a meeting on reading week and again last Tuesday of reading week but have yet to hear back due to Eugene being out of the office. I’m not too concerned as I know I will hear from him when he returns.

**Intended Changes**

Speaking to Sara Kadry about my project she recommended that there are three security properties within my project that I can focus on:

1. “Strong authentication mechanism to prove the user's identification every time when the user logins.”

2. “Database encryption which is encrypted the user's sensitive information such as credit card details.”

3. “Protection against The SQL injection because your app depends on searching and executing query to database”.

Also it was suggested to develop my project as a web application. I also realised that there were potentially a number of problems of trying to develop it as an Android Application in regards to the payment aspect which would require me to use Google Plays In-App billing system meaning I would be unable to develop my own payment system.

The above suggestions are now part of my main focus for the application.

## Prototype User Guide and Environment Setup

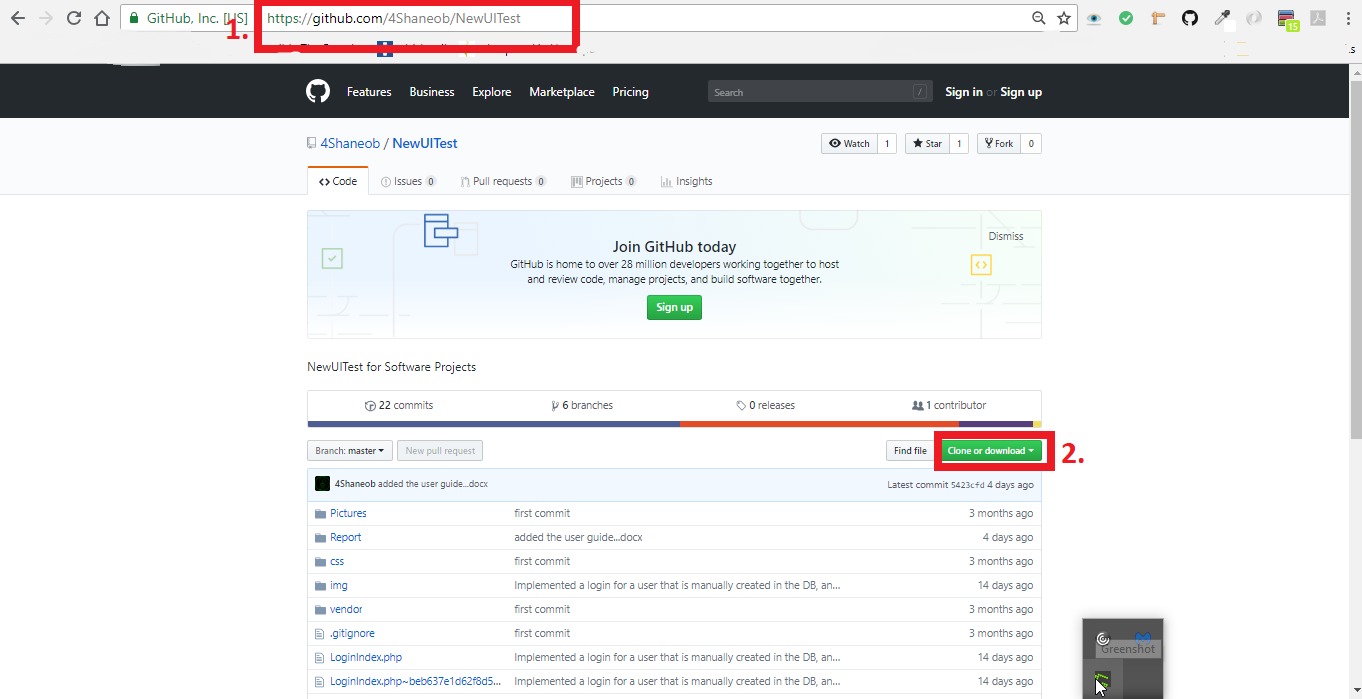
**Environment Needed**

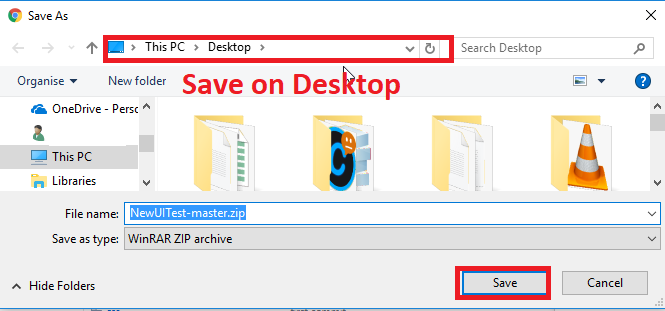
**What You Need:**

* Internet Access
* Laptop or Desktop Computer
* Browser (preferably Chrome and/or another)
* Project Files (available on github <https://github.com/4Shaneob/NewUITest>)
* Xampp Server or Wamp Server
* Database imported from file to localhost server
* User Account Details
* Sandbox PayPal Account Details

**Guide**

For Project Files

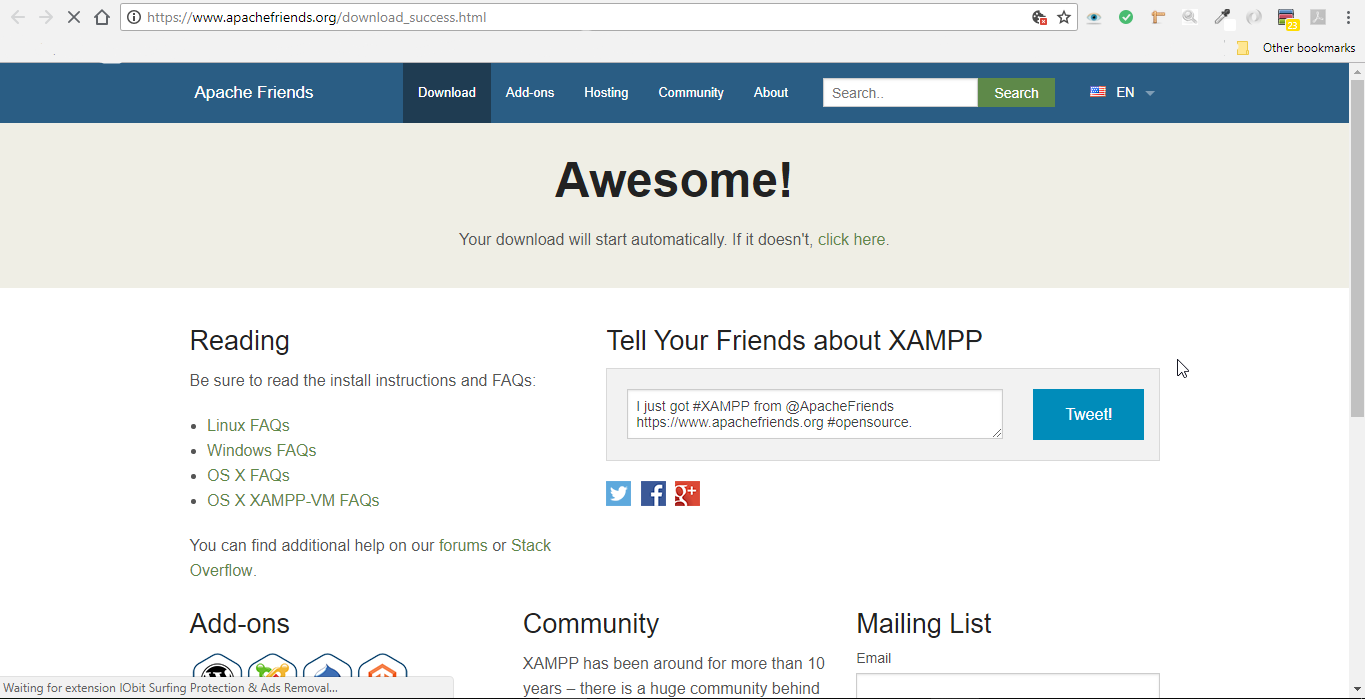
1. Got to <https://github.com/4Shaneob/NewUITest>
2. Click on the “clone or download” button
3. Click on Download ZIP. 
4. Save to Desktop



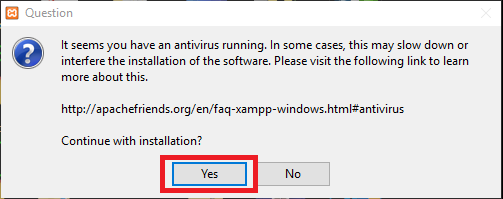
For XAMPP Setup

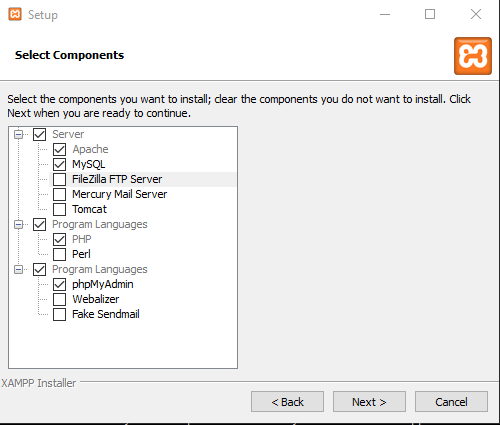
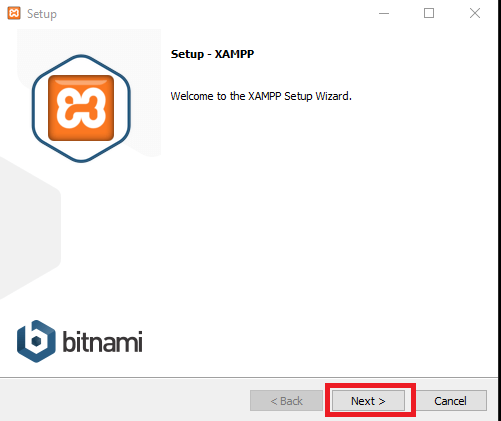
1. Go to <http://www.apachefriends.org/index.html>
2. Select Download for Windows

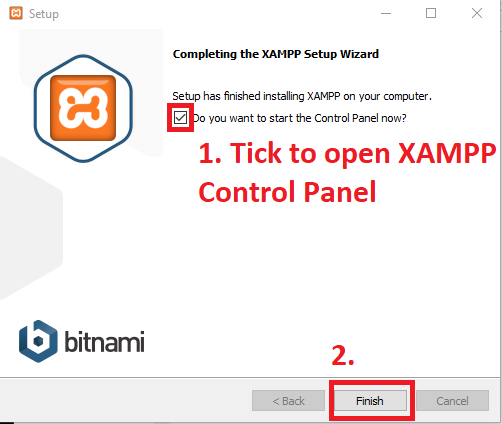
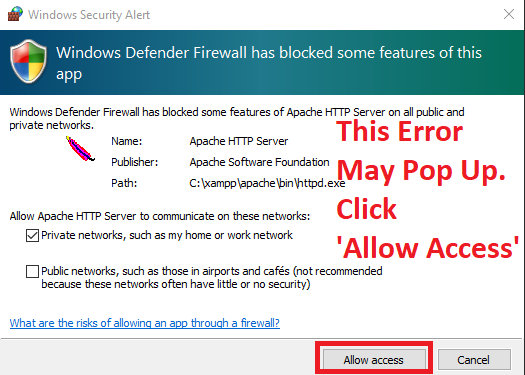
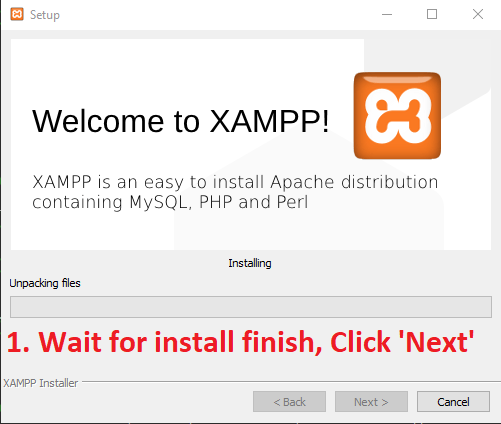
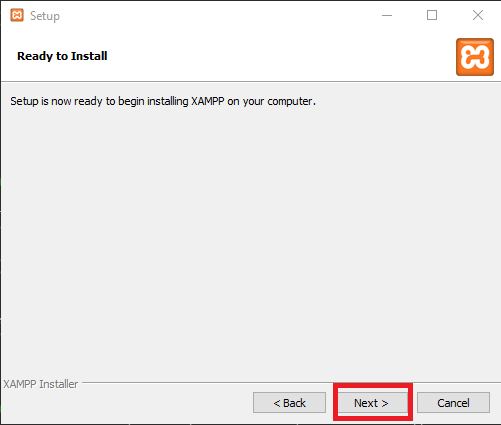
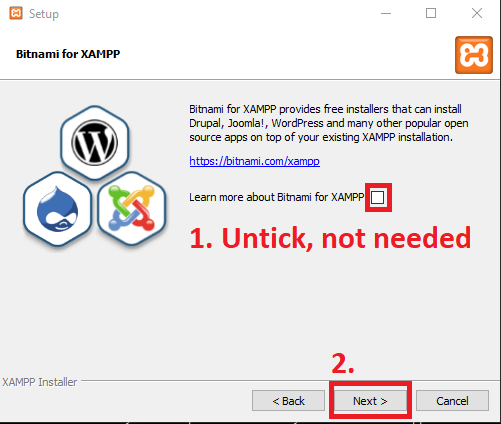
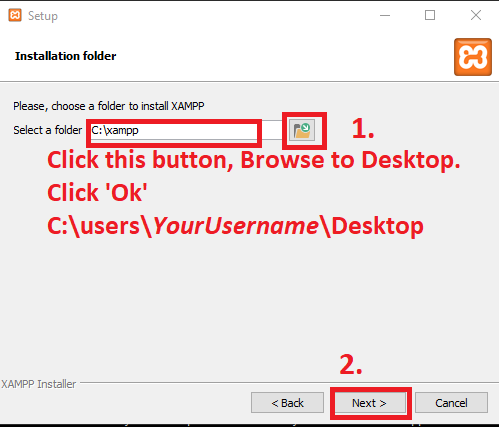




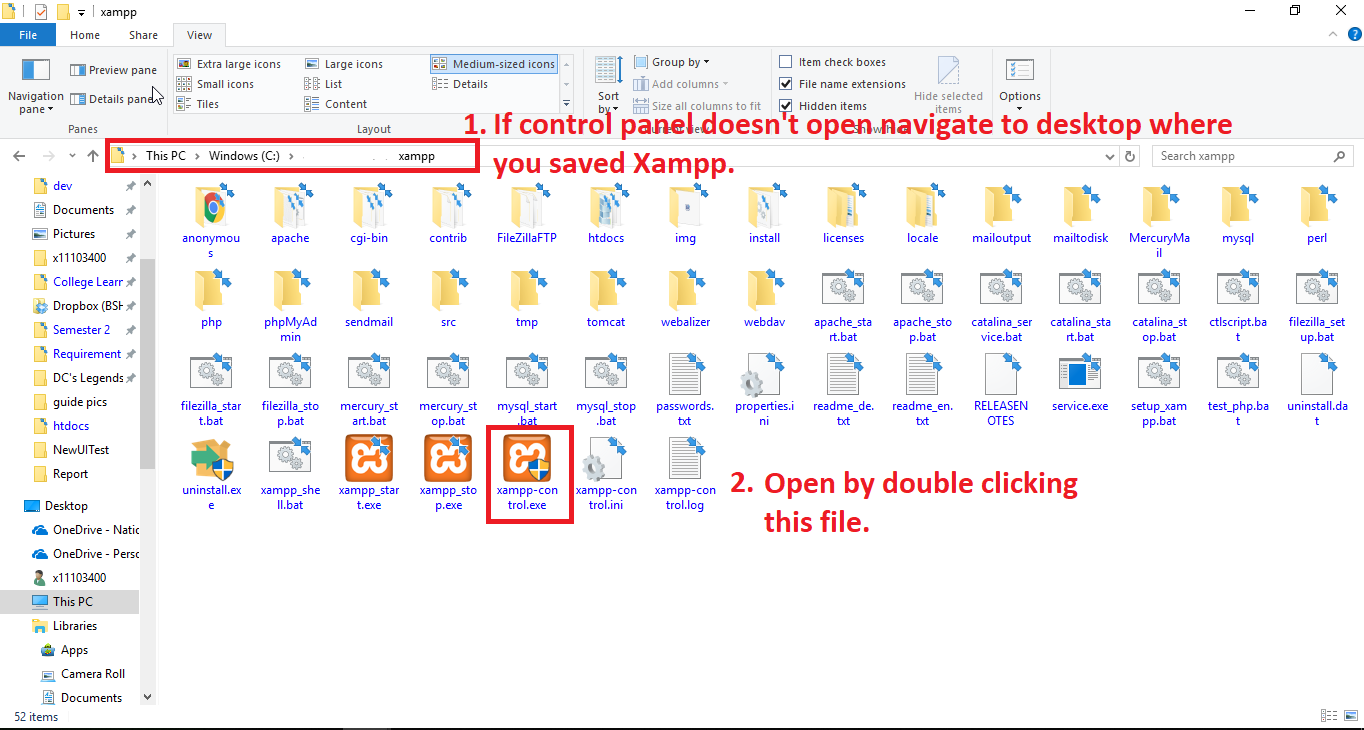
1. Once downloaded, run the executable file.
2. Follow the default Install Wizard Setup

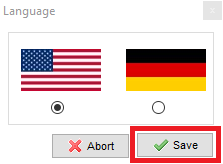




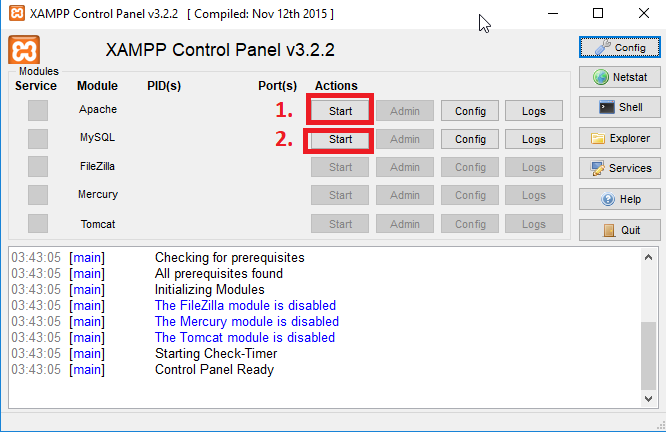


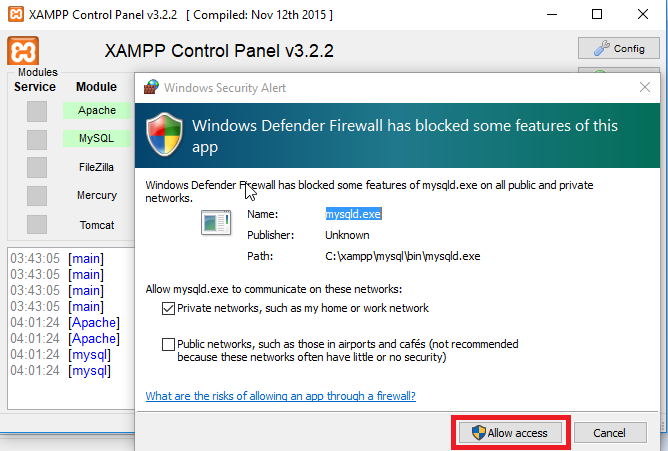
1. Once XAMPP is installed go to location of installation
2. Open “XAMPP-control.exe”



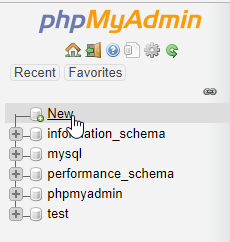


Import Database File

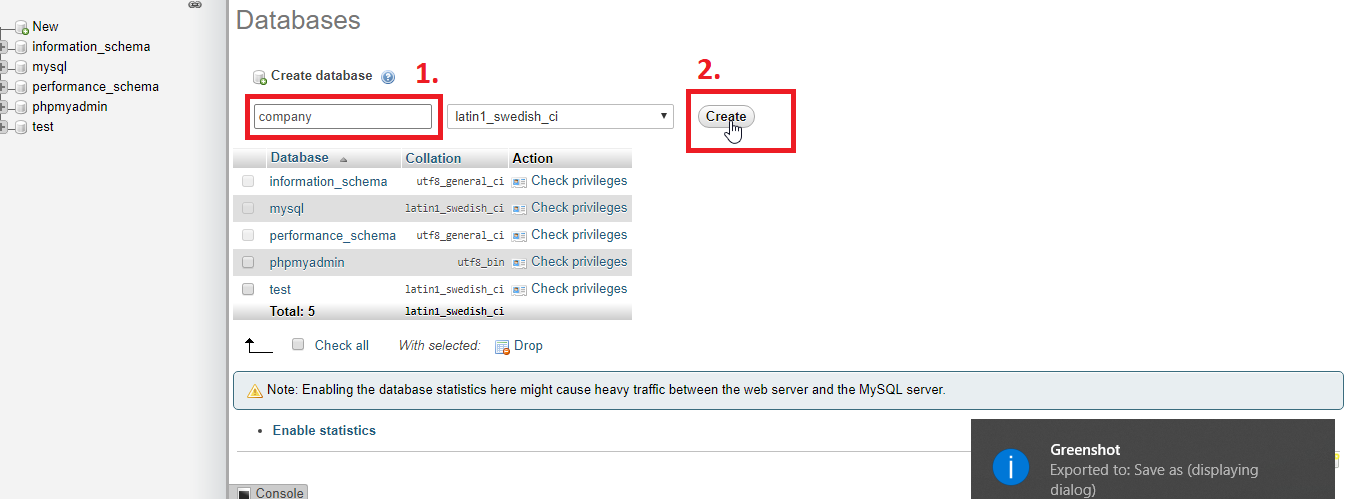
1. Once XAMPP Control is open
2. Click “start” button beside the Apache and MySQL modules. 



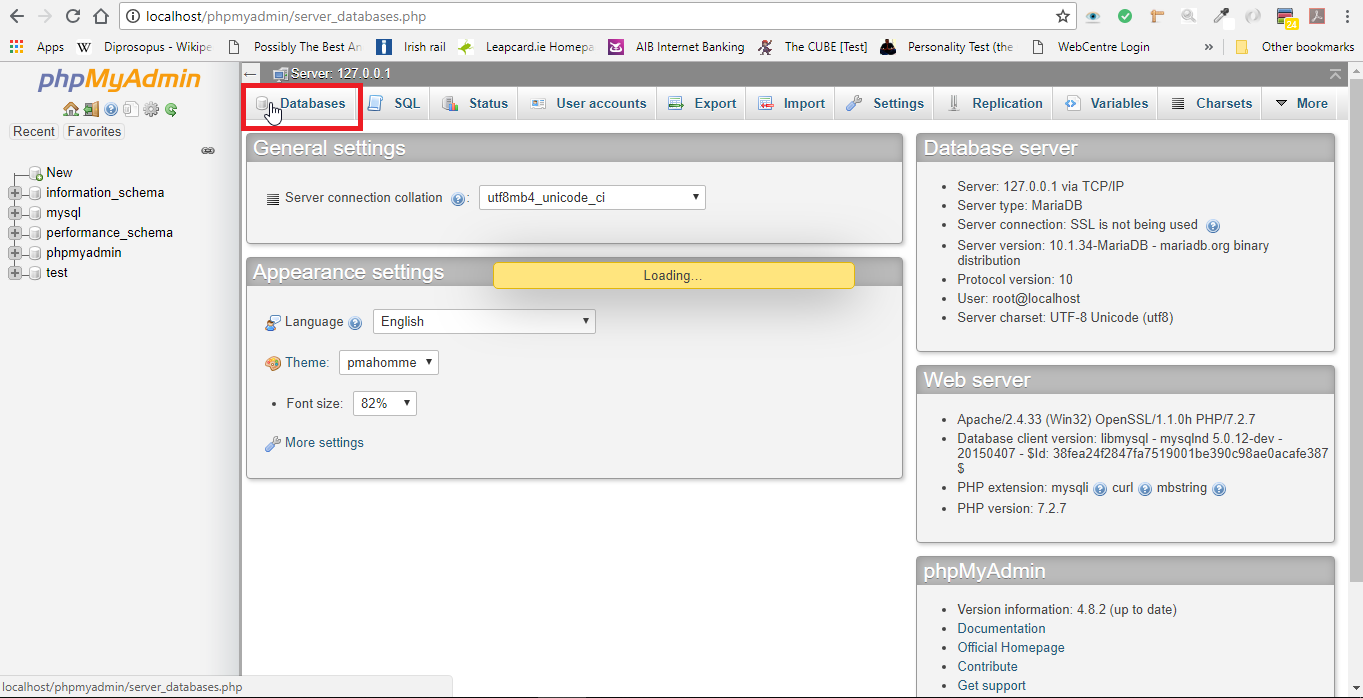
1. The click on ‘Admin’ beside MYSQL Module. This opens PHPmyAdmin in a browser window.
2. Log in to PHPMyAdmin. (Admin root)
3. Click on ‘New’



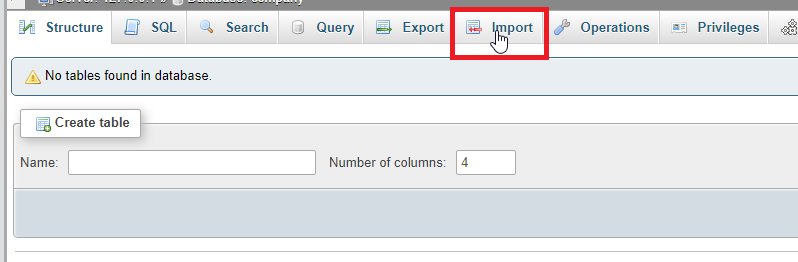
1. Under Create Database type ‘Company’,
2. Then Click ‘Create



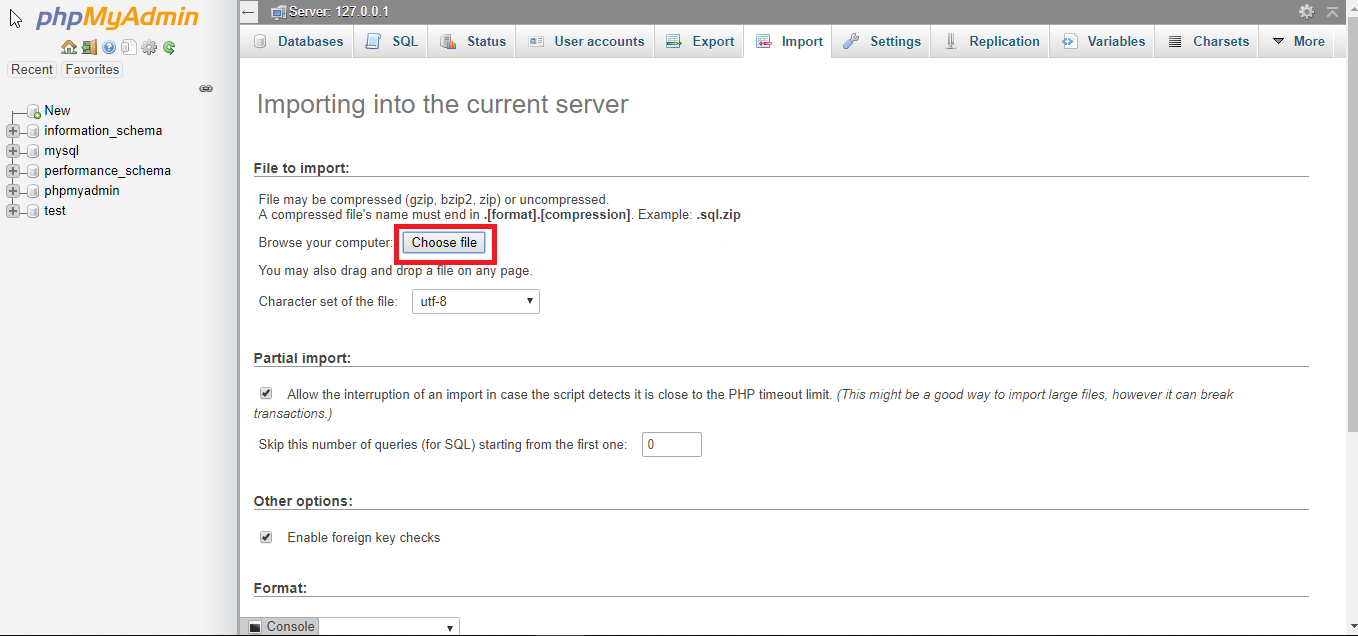
1. Click on the “Databases” Tab

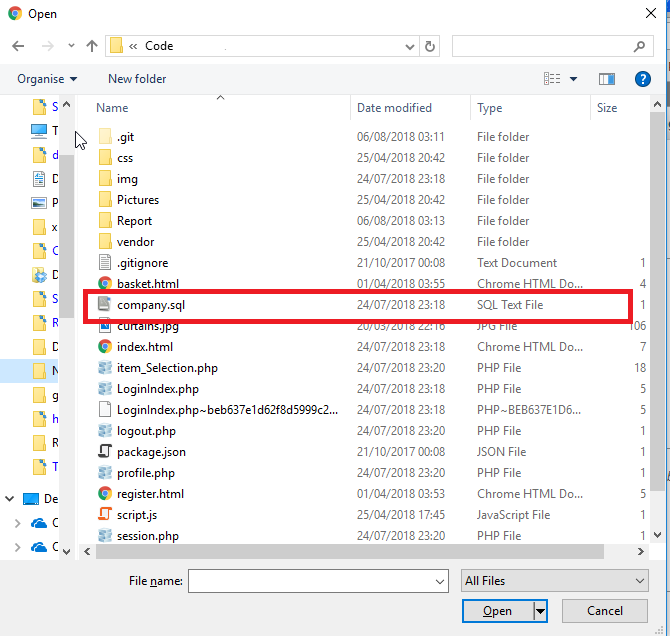


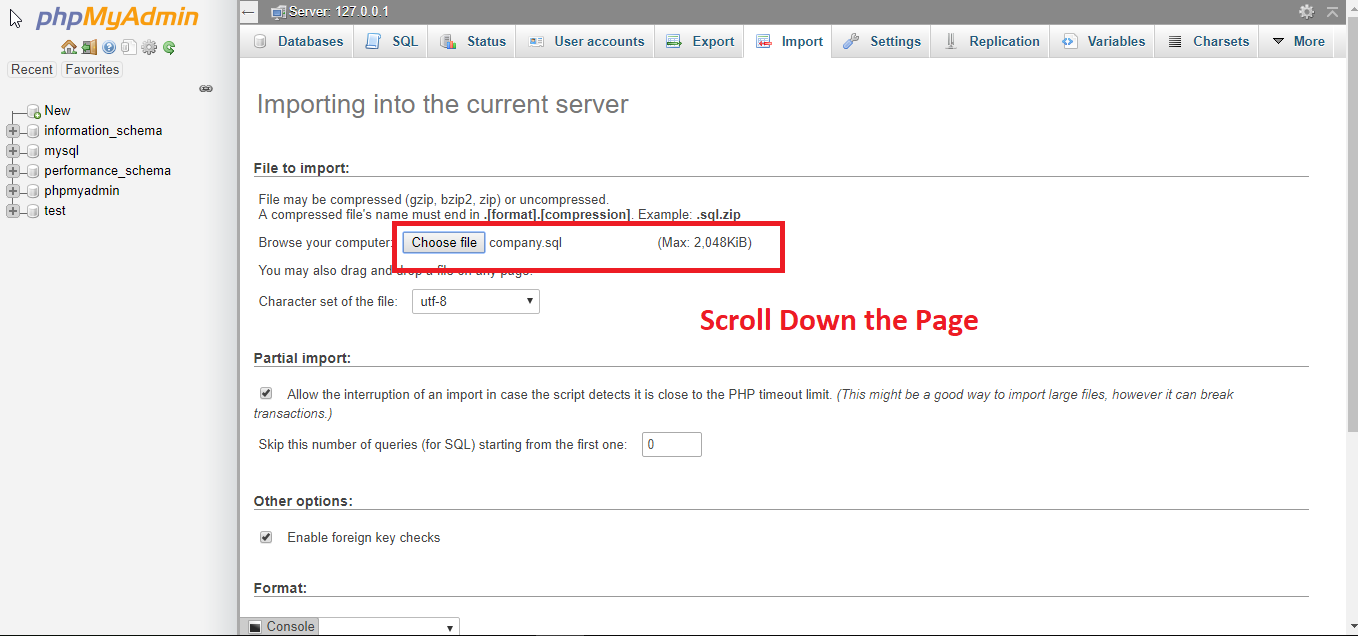
1. Then click on the “Import” Tab

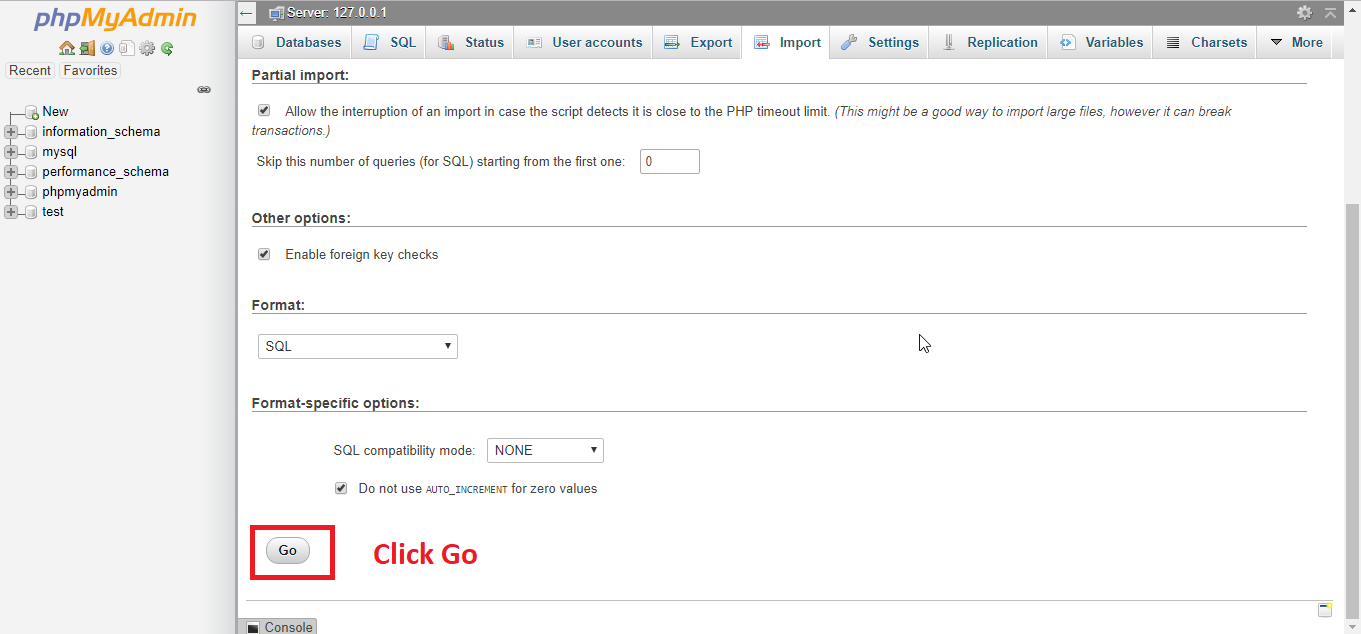


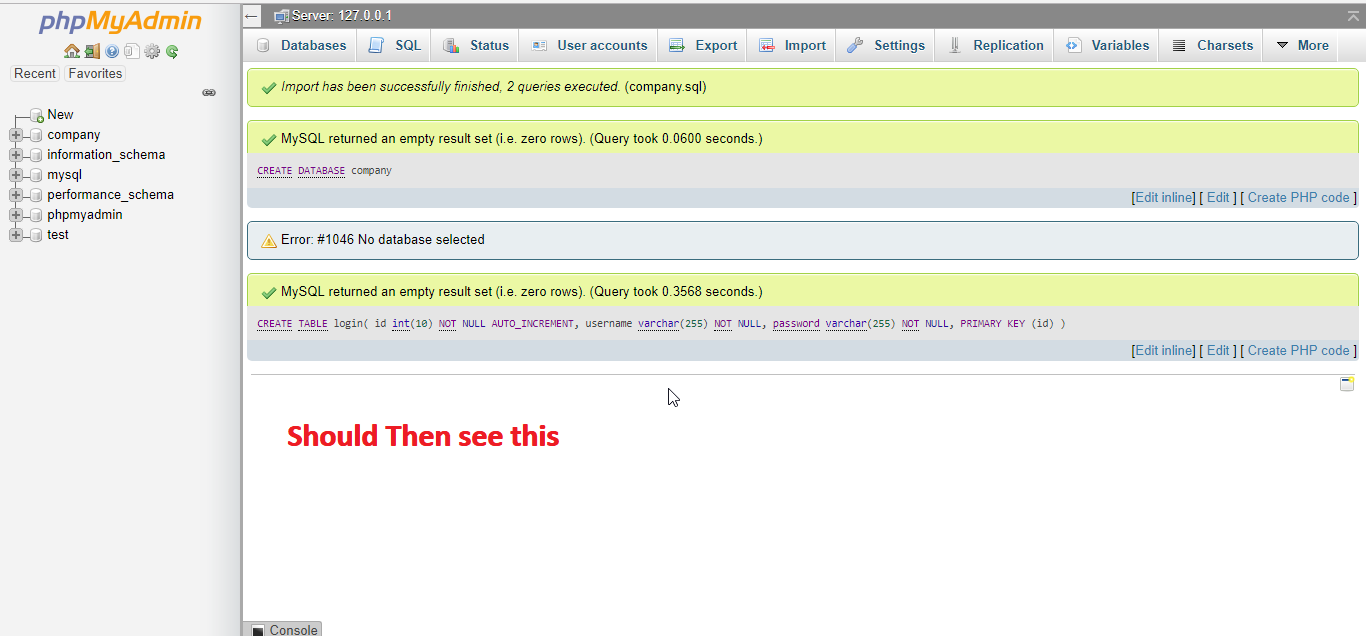
1. Under “***File to import”*** click on the ‘choose File’ Button.



1. Select the “company. sql” file and Open. 



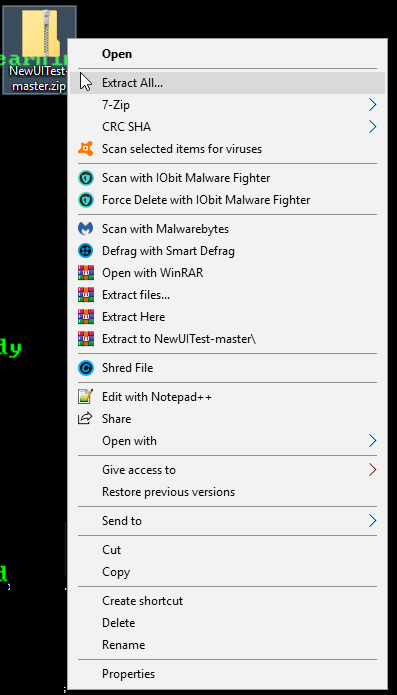






Run the Project

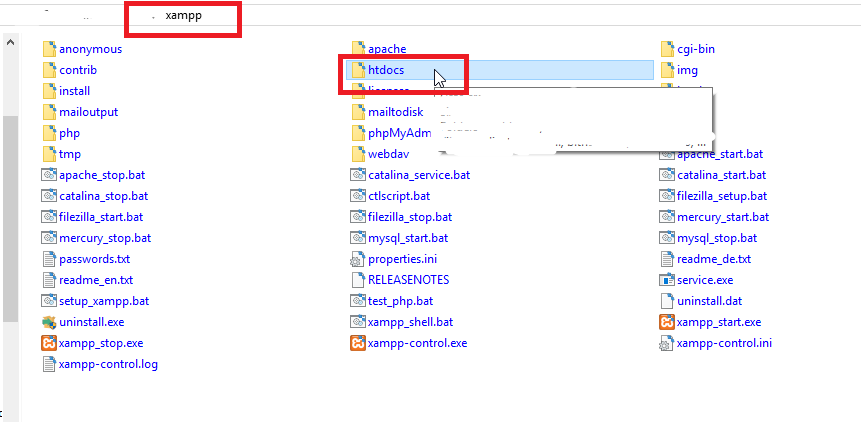
1. In file explorer, browse to the location of the Project files you downloaded right Click the zip file.



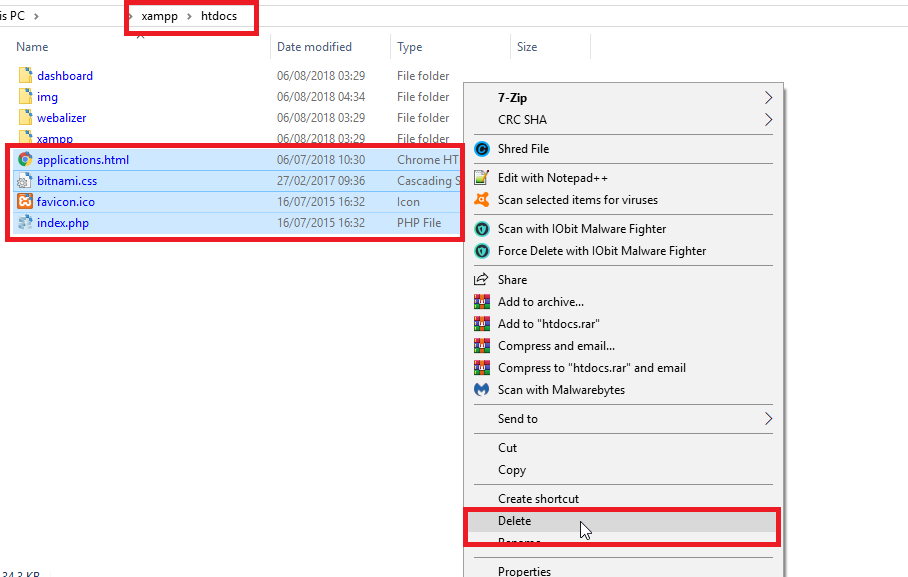
1. Click ‘Extract All’



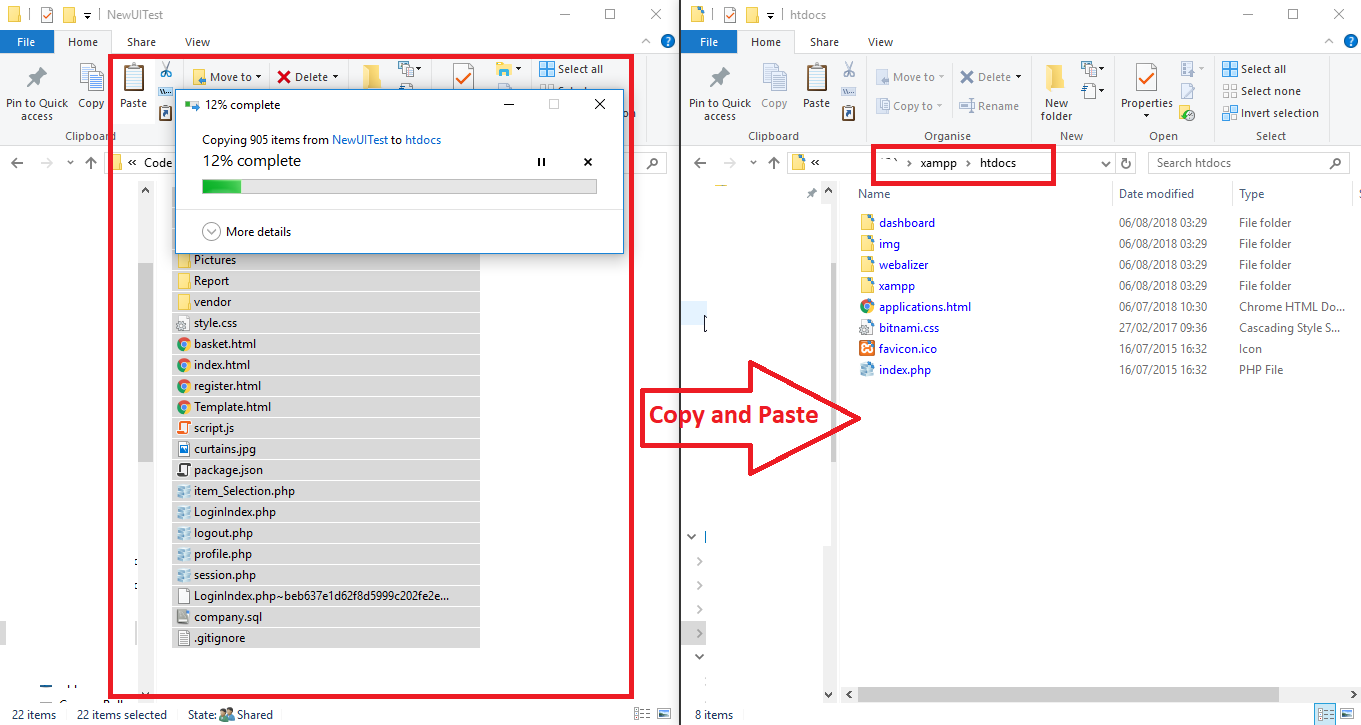
1. Go to \XAMPP\htdocs



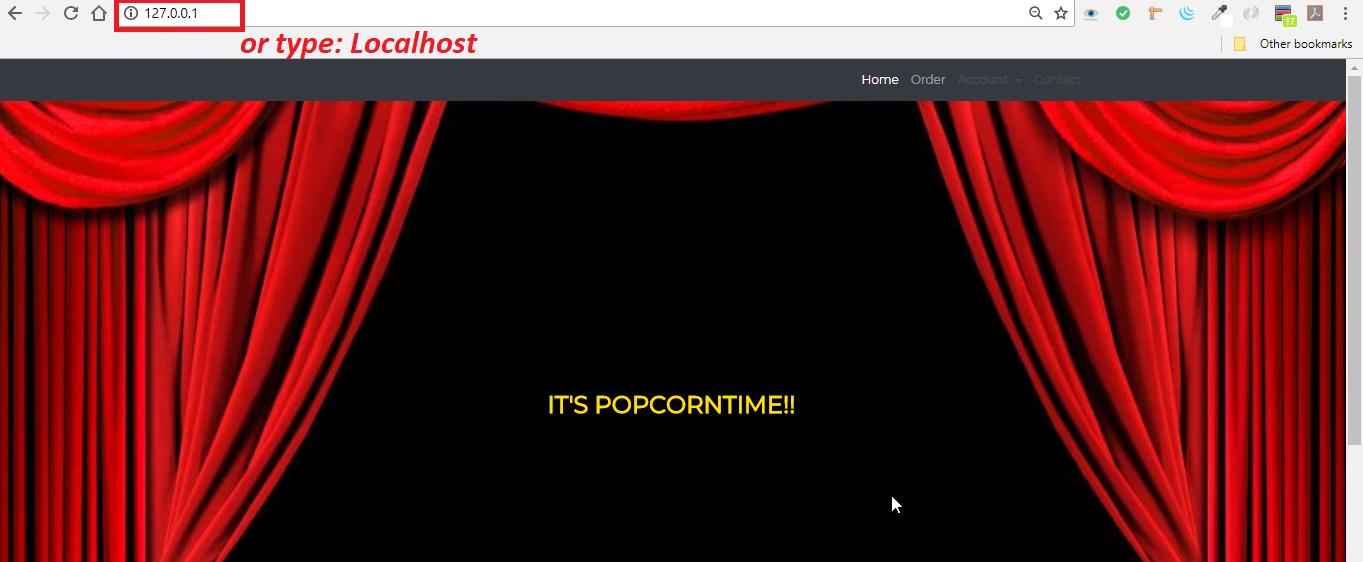
1. Select the default files and right click and delete



1. Copy the Project files to \XAMPP\htdocs



1. Once files have copied open a browser
2. In the URL address bar type: “localhost” or “127.0.0.1”



Login Details

1. For Website:

|  |  |
| --- | --- |
| **Email:** | test@prototype.com |
| **Password:** | P@ssword |

1. For euro(process in French):

|  |  |
| --- | --- |
| **PayPal Username:** | test2@popcorn.com |
| **PayPal Password:** | P0pcorn18 |

For GBP(process in English):

|  |  |
| --- | --- |
| **PayPal Username:** | test@popcorn.com |
| **PayPal Password:** | P0pcorn18 |

The prototype

1. It will open on the index.html page, displaying a Login, Guest and disabled register button. There is also a navigation menu on the top of the page.
   1. The login button redirects to a login page where you will log in using predetermined login credentials.
      1. Type in the username and the password.
      2. If you type either incorrectly then you will receive an error.
      3. If the username is not an email you will receive an error.
      4. Once you log in the page will redirect to the items page which will say welcome “user”.
   2. The guest button should direct to the items page which should only say welcome under the navigation page.
2. You can then choose 1 item fill in the required selection and click the add to cart.
   1. This will redirect in a new browser tab to the PayPal Cart.
   2. To add another close tab and go back to the item selection page.
   3. Choose a different item and repeat until till you have selected as many items as you wish to continue.
3. Once done go back once more select the view cart button on the items page.
4. Click buy now.
5. It will ask for PayPal log in credentials enter the sandbox details I have provided.
6. Continue through the mock payment until complete.
7. On all pages you view resize the browser window to mimic a mobile device.