**CT4009 Introduction to Web Development**

**Assignment 001**

Technical Report

for the offline store catalogue

**Student Name**: Nathaniel Clarke

**Student Number**: s1906563

**Course**: Computer and Cyber Security

**Submission Date**: Monday, 13 January 2020

**University of Gloucestershire**

Contents

[Website Details 4](#_Toc29379568)

[List of Functional and Non-functional pages 5](#_Toc29379569)

[Testing 6](#_Toc29379570)

[User Manual 12](#_Toc29379571)

[Interesting Use of HTML/CSS/JavaScript 22](#_Toc29379572)

[Shortcomings 23](#_Toc29379573)

[References 24](#_Toc29379574)

[Appendix 1 27](#_Toc29379575)

[Appendix 2 27](#_Toc29379576)

[2.1 styles.css 27](#_Toc29379577)

[2.2 HomePage.html 27](#_Toc29379578)

[2.3 catTemplate.html 27](#_Toc29379579)

[2.4 contact.html 27](#_Toc29379580)

[2.5 deals.html 27](#_Toc29379581)

[2.6 crud.html 27](#_Toc29379582)

[2.7 watch.html 27](#_Toc29379583)

[2.8 EventList.html 27](#_Toc29379584)

[2.9 results.html 27](#_Toc29379585)

[2.10 AddCategory.js 27](#_Toc29379586)

[2.11 AddItem.js 27](#_Toc29379587)

[2.12 AddItemDAO.js 27](#_Toc29379588)

[2.13 AddSubcategory.js 27](#_Toc29379589)

[2.14 indexeddb.js 27](#_Toc29379590)

[2.15 initCompare.js 27](#_Toc29379591)

[2.16 listCat.js 27](#_Toc29379592)

[2.17 ListItems.js 27](#_Toc29379593)

[2.18 ListItemsDAO.js 27](#_Toc29379594)

[2.19 updateCat.js 27](#_Toc29379595)

[2.20 updateCatDAO.js 27](#_Toc29379596)

[2.21 UpdateItem.js 27](#_Toc29379597)

[2.22 UpdateItemDAO.js 27](#_Toc29379598)

[2.23 updateMap.js 27](#_Toc29379599)

[2.24 updateMapDAO.js 27](#_Toc29379600)

[2.25 updateSubcat.js 27](#_Toc29379601)

[2.26 updateSubcatDAO.js 27](#_Toc29379602)

[2.27 accordion.js 27](#_Toc29379603)

[2.28 main.js 27](#_Toc29379604)

[2.29 EventList.js 27](#_Toc29379605)

[2.30 EventListDAO.js 27](#_Toc29379606)

[2.31 InsertEvent.js 27](#_Toc29379607)

[2.32 InsertEventDAO.js 27](#_Toc29379608)

[2.33 listLocations.js 27](#_Toc29379609)

[2.34 Update.html 27](#_Toc29379610)

[2.35 updateCat.html 27](#_Toc29379611)

[2.36 updateMap.html 27](#_Toc29379612)

[2.37 updateSubcat.html 27](#_Toc29379613)

# Website Details

**Plesk**

Username: s1906563

Password: HJ55xH%nMbDv#t9?

Domain name: s1906563-ct4009.uogs.co.uk

Plesk URL for Web Hosting Management: https://uogs.co.uk:8443/login\_up.php

**FTP** Login: s1906563-ct4009 Password: Dey$146g

# List of Functional and Non-functional pages

\*JavaScript/IndexedDB only needed for login/nav bar

\*\*functional but nothing on the page except nav bar

|  |  |  |
| --- | --- | --- |
| Page | HTML/CSS | JavaScript/IndexedDB |
| Home – carousel/login\* | ✓ | ✓🗶 |
| Store locator - map | ✓ | ✓ |
| Contact – email/map link\* | ✓ | ✓🗶 |
| Shop - view cat/subcat/items/compare/add to watchlist | ✓ | ✓ |
| Hot Deals\*\* | ✓ | ✓🗶 |
| Watchlist - view | ✓ | ✓ |
| Search Results\*\* | ✓ | ✓🗶 |
| Admin Controls – add cat/add subcat/add item/add location   * Delete & link to update cat/subcat/item/location | ✓ | ✓ |
| Update Map | ✓ | ✓ |
| Update Category | ✓ | ✓ |
| Update Subcategory | ✓ | ✓ |
| Update Item | ✓ | ✓ |

# Testing

All login, IndexedDB tests use /js/lib/indexeddb.js, /js/main.js

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Ref.** | **Page Area** | **Scope** | **Expected Results** | **File ref** | **Actual Results** | **Fixes** |
| 001 | Home Login | Login function   1. Login clicked 2. Invalid credentials entered | Login field appears, Access denied, user alerted  Code: respond to incorrect input | /home/HomePage.html  /js/main.js | As expected | N/A |
| 002 | Home Login | Login function admin   1. Correct credentials entered | User redirected to admin page  Code: match valid credentials, redirect to target URL | /home/HomePage.html  /js/main.js | As expected | N/A |
| 003 | Admin Controls Add item, subcategory | Empty selection input   1. User adds item without selecting subcategory or dB has no values to display 2. User adds subcategory without selecting category or dB has no values to display | Form not submitted; user alerted  Code: handle empty input, prevent form submitting | /login/crud.html  /js/AddFunc/AddItem.js  /js/AddFunc/AddItemDAO.js  /js/AddSubcat/AddSubcategory.js | As expected | N/A |
| 04 | Add Category | New Category with name and description   1. User completes category form | Category added to dB, user alerted  Code: enter input into category dB store | /login/crud.html  /js/lib/indexeddb.js  /js/AddCat/AddCategory.js | As expected | N/A |
| 05 | Add Subcategory | Subcategory with parent category   1. Select category from selection and submit | Subcategory added to dB, user alerted  Code: enter input into subcategory dB store with according category id | /login/crud.html  /js/lib/indexeddb.js  /js/AddSubcat/AddSubcategory.js | As expected | N/A |
| 006 | Add item | Item with image, subcategory parent id   1. Enter values, select subcategory 2. Upload image | Item added to dB  Code: enter input into item dB store with subcategory id and image file | /login/crud.html  /js/lib/indexeddb.js  /js/AddFunc/AddItem.js  /js/AddFunc/AddItemDAO.js | As expected | N/A |
| 007 | Add location | Add location to db   1. Enter store name 2. Drag marker to desired location and submit | Latitude/longitude of marker saved to dB, user alerted  Code: save data to events store, alert user, handle empty inputs | /login/crud.html  /js/lib/indexeddb.js  /map/InsertEvent.js  /map/InsertEventDAO.js | Location added in centre of map | EventList.js was causing issues being linked in the same page, location list now doesn’t have map with markers |
| 008 | Update/delete locations | List of locations viewable with update/delete buttons | Location list from events dB store viewable  Code: get all values from events store, display with links to update/delete | /login/crud.html  /js/lib/indexeddb.js  /map/listLocations.js | Item values are “undefined” | Function had to be separated from InsertEvent.js file, js link in html needed to be within accordion div |
| 009 | Update/delete locations | Deleting entry from dB   1. Click delete button of desired location | Location deleted from db, user alerted  Code: remove entry with matching id from events store, alert user | /login/crud.html  /js/lib/indexeddb.js  /map/listLocations.js | Data not deleted | Delete function needed to be within function to build html div populated by db |
| 010 | Update/delete categories with subcategories & items | Viewing items indented in subcategories indented in categories   1. Click update categories button | Objects added to db viewable, items under correct subcategories, under correct categories  Code: build html with matching id’s and informating showing | /login/crud.html  /js/lib/indexeddb.js  /js/ListCat/listCat.js | As expected | N/A |
| 011 | Update/delete categories with subcategories & items | Update category, subcategory and item successfully   1. Click update button underneath desired category, subcategory or item | User taken to update page for category, subcategory or item according to which clicked. Current object info filled out, able to edit and submit  Code: Redirect to target URL, obtain correct db entry. Change id values in store | /login/crud.html  /js/lib/indexeddb.js  /js/ListCat/listCat.js  /js/UpdateFunc/updateCat.js  /js/UpdateFunc/updateCatDAO.js  /js/UpdateFunc/updateSubcat.js  /js/UpdateFunc/updateSubcatDAO.js  /js/UpdateFunc/UpdateItem.js  /js/UpdateFunc/UpdateItemDAO.js | 1. Buttons don’t do anything 2. Id values ‘null’ when #1 fixed | 1. Function calls had to be within html build function 2. Needed to call function to get values of objects 3. in array and parse id from array as int |
| 012 | Update/delete Locations | Update store locations successfully   1. Click update button of specific location | User taken to map update page, current info filled out, able to edit and submit  Code: Redirect to target URL, obtain correct db entry. Change id values in store | /login/crud.html  /map/listLocations.js  /js/UpdateFunc/updateMap.js  /js/UpdateFunc/updateMapDAO.js | As expected | N/A |
| 013 | Store Locator | View Store locations | User can see map with markers and titles for each store  Code: retrieve and display all marker information from db | /map/EventList/EventList.html  /map/EventList/EventList.js  /map/EventList/EventListDAO.js | Location in centre of map | Fixed in test #007 |
| 014 | Shop | Navigate to specific category and view subcategories and items   1. Click link to category in nav dropdown | User taken to clicked category on shop page, subcategories and items displayed  Code: redirect to specific element on page, retrieve objects from db and populate html | /CatTemplate/catTemplate.html  /js/ListCat/listCat.js  /js/ListCat/initCompare.js | Images not displaying | Image folder path had to be in image source – images must be uploaded from that folder |
| 015 | Shop | Add 2 items to compare   1. Click compare checkbox on two items | Both items displayed with remove button, without compare or watchlist checkboxes  Code: get correct id and display in compare box without checkboxes, allow no more than two items | /CatTemplate/catTemplate.html  /js/ListCat/listCat.js  /js/ListCat/initCompare.js | As expected | N/A |
| 016 | Shop | Remove items from compare   1. Click remove under compare item | Item deleted from compare,  Code: remove item from div, hide compare div if empty | /CatTemplate/catTemplate.html  /js/ListCat/listCat.js  /js/ListCat/initCompare.js | As expected | Added code to uncheck checkbox when removed |
| 017 | Shop | Add item to watchlist   1. Click add to watchlist checkbox on item | Item added to watchlist db store, user alerted  Code: get item info and add to watchlist store | /CatTemplate/catTemplate.html  /js/ListCat/listCat.js | ID and item info not being found and added | Needed to split id to parse as int |
| 018 | Any | Login to public user page   1. Enter valid credentials and submit | User taken to account/watchlist page, watchlist items viewable  Code: match valid credentials, redirect to target URL | /js/main.js  /WatchList/watch.html  /js/ListFunc/ListItems.js  /js/ListFunc/ListItemsDAO.js | Items not being found | Incorrect object store being named |
| 019 | Watchlist | Delete item from watchlist   1. Click remove button on item | Item removed, user alerted  Code: delete item from db, alert user and refresh page | /WatchList/watch.html  /js/ListFunc/ListItems.js  /js/ListFunc/ListItemsDAO.js | As expected | N/A |
| 020 | Mobile Site | Test site loads on mobile   1. Load site on mobile device | Site should format properly, nav condensed to burger, site functions same as desktop view  Code: distinguish type of device user is using and format | /home/HomePage.html | As expected | N/A |

# User Manual

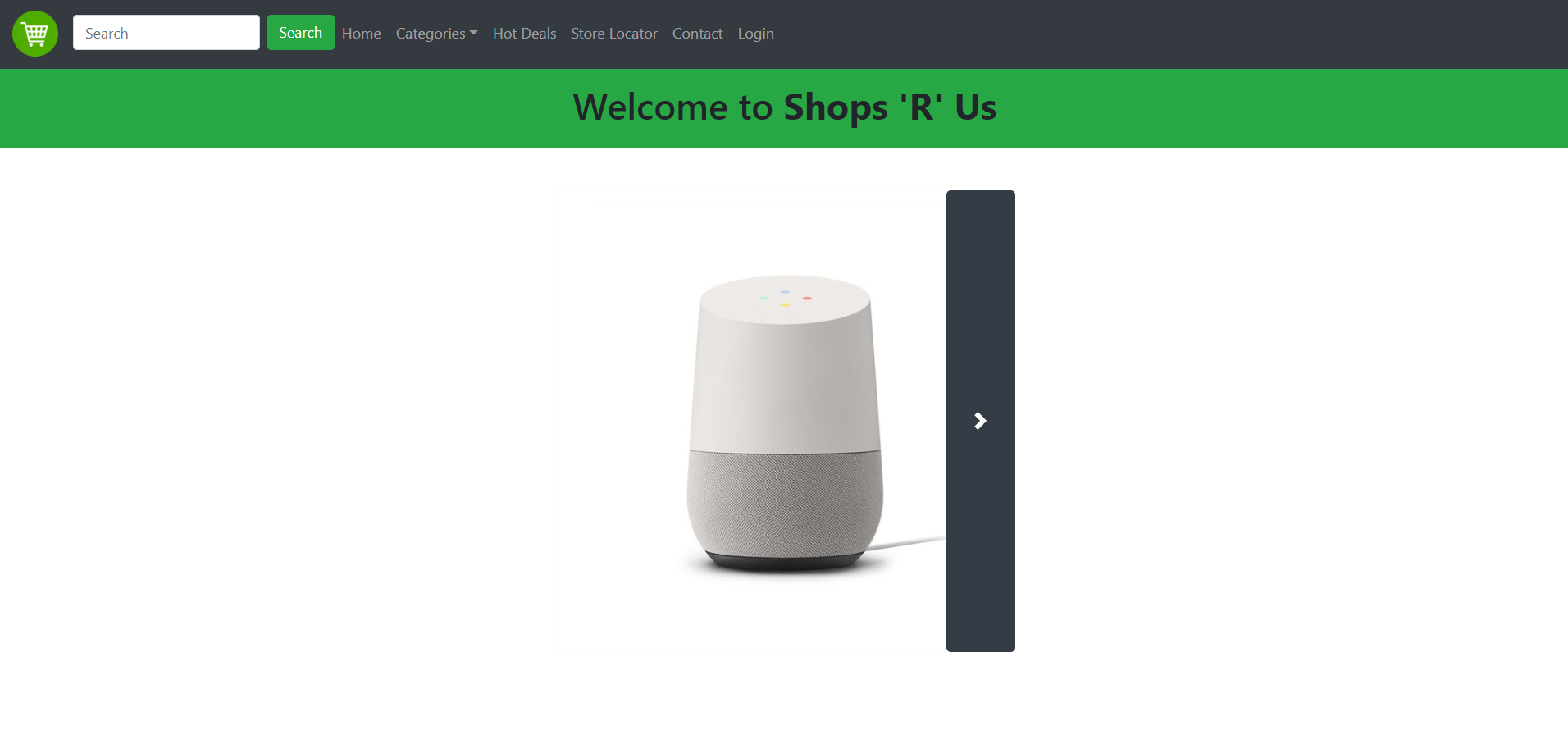
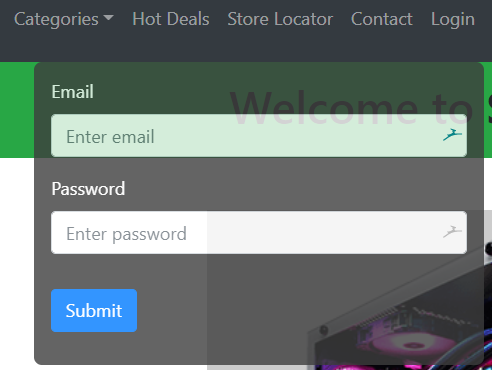
In this manual you will find a detailed explanation of every functionality of the site.

Admin controls: p12 – 17

Contact/Store Locator: p18

Shop/Compare: p19

Watchlist: p20



Click on the login button and this box will appear

The admin login will take a user to the Admin Controls page.

Email: admin@test.com

Password: password123

The public login will take a user to their account or watchlist.

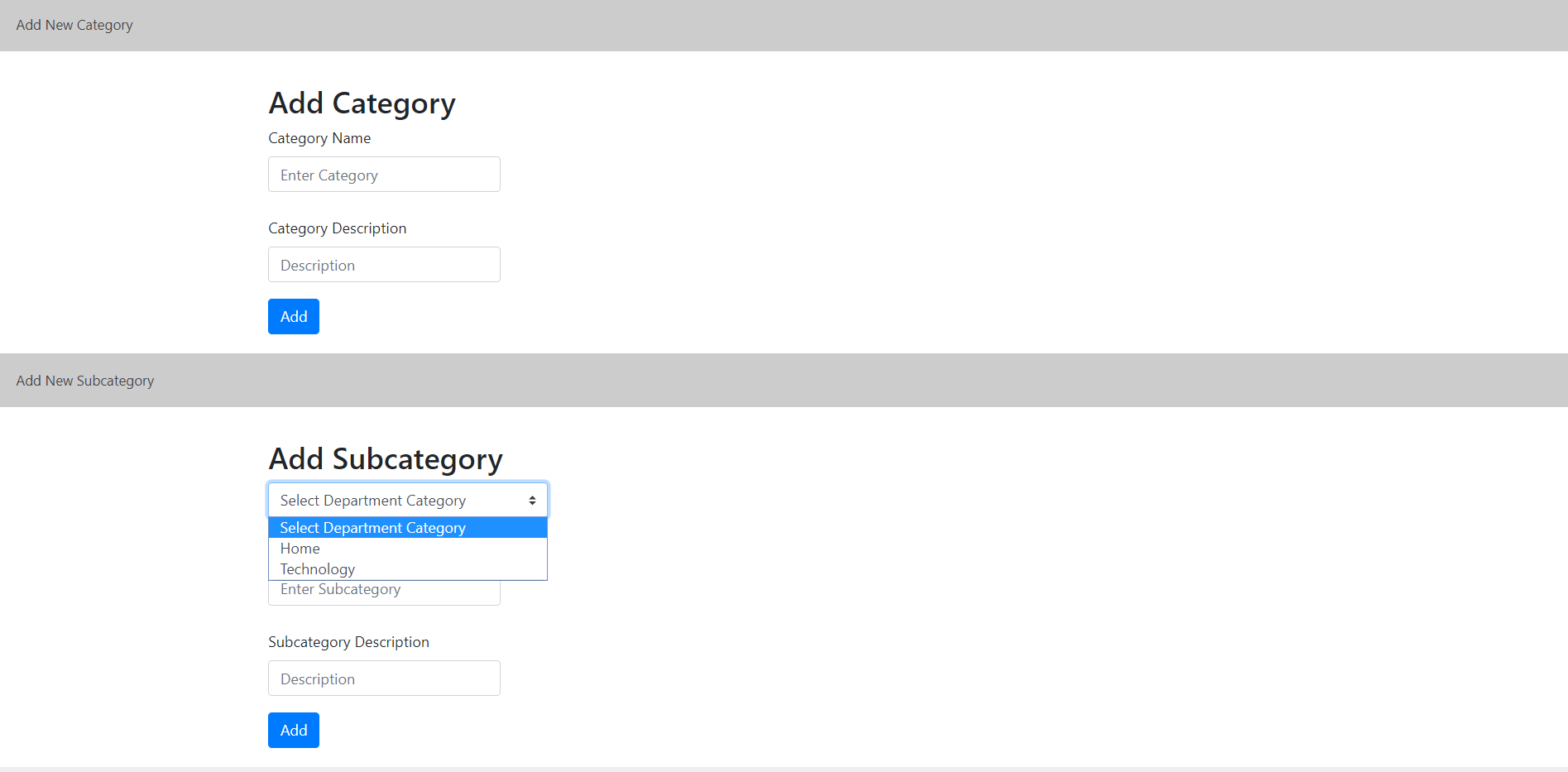
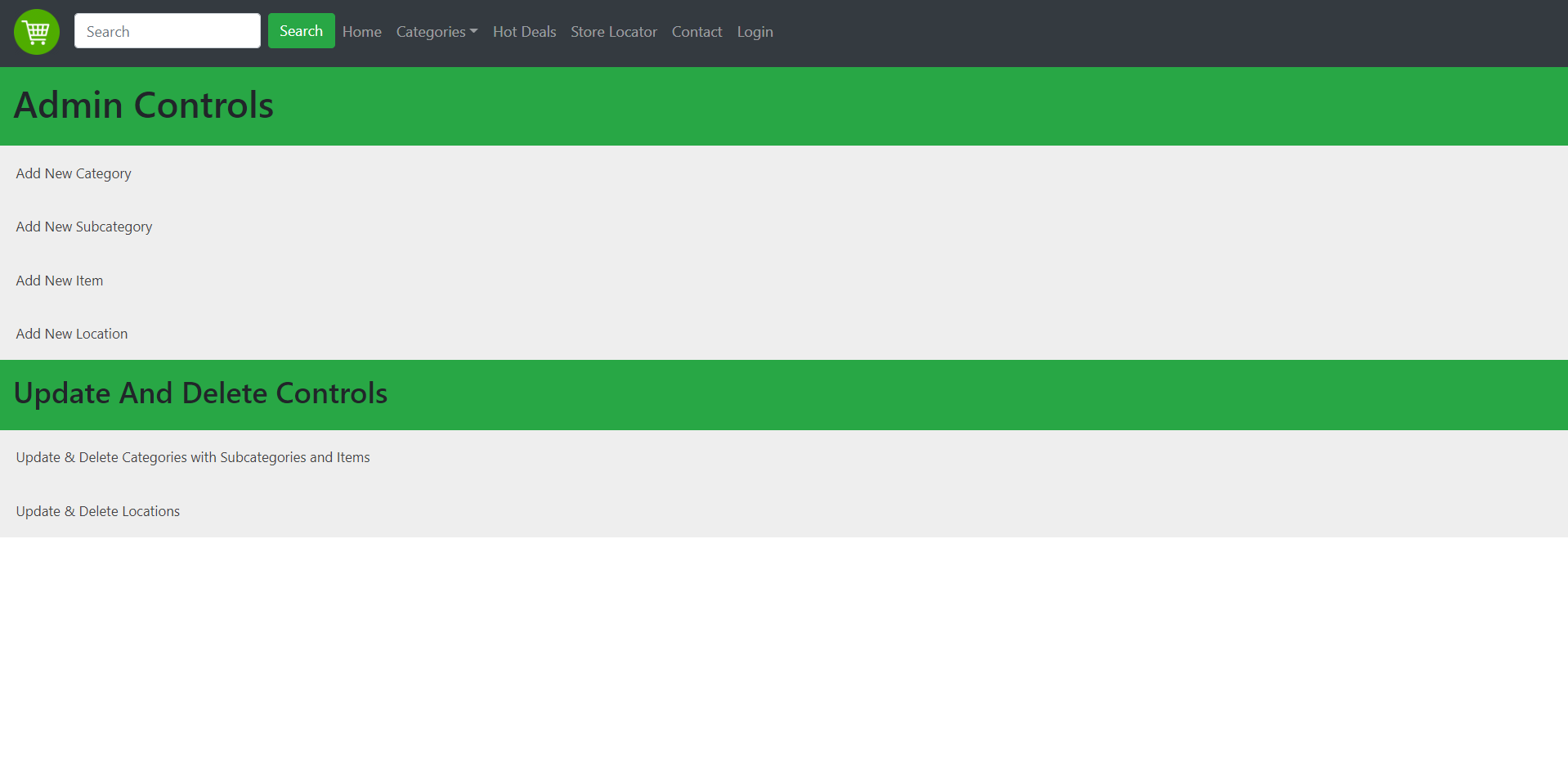
Email: public@test.com

Password: 321password

This is the home page a user will first see.

At top is the navigation bar with the links to the shop page, hot deals (non-functional), store locator, contact and login. There is also a search bar, but this functionality has not been implemented and it will take the user to a blank page.

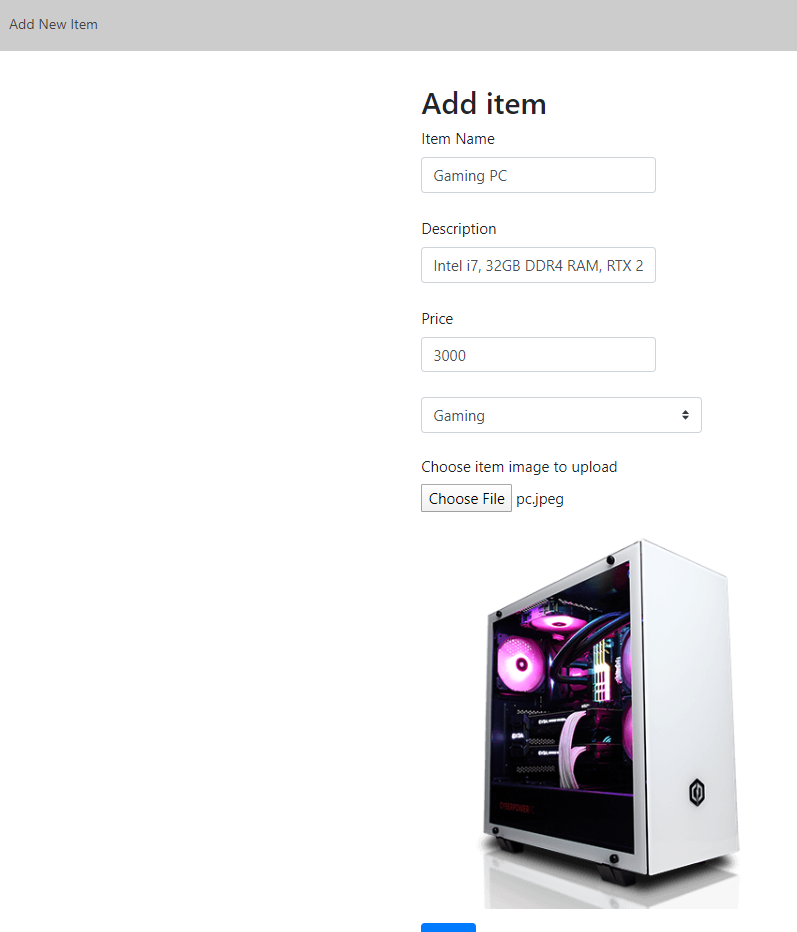
This carousel currently cycles through 3 images however it’s only fed from the file path, not from the database.



This is the page where administrators can add a new category, subcategory and item that all relate to each other. These can all be updated and deleted too. Store locations can also be added which public users can view in the store locator page. These too can be updated and deleted.

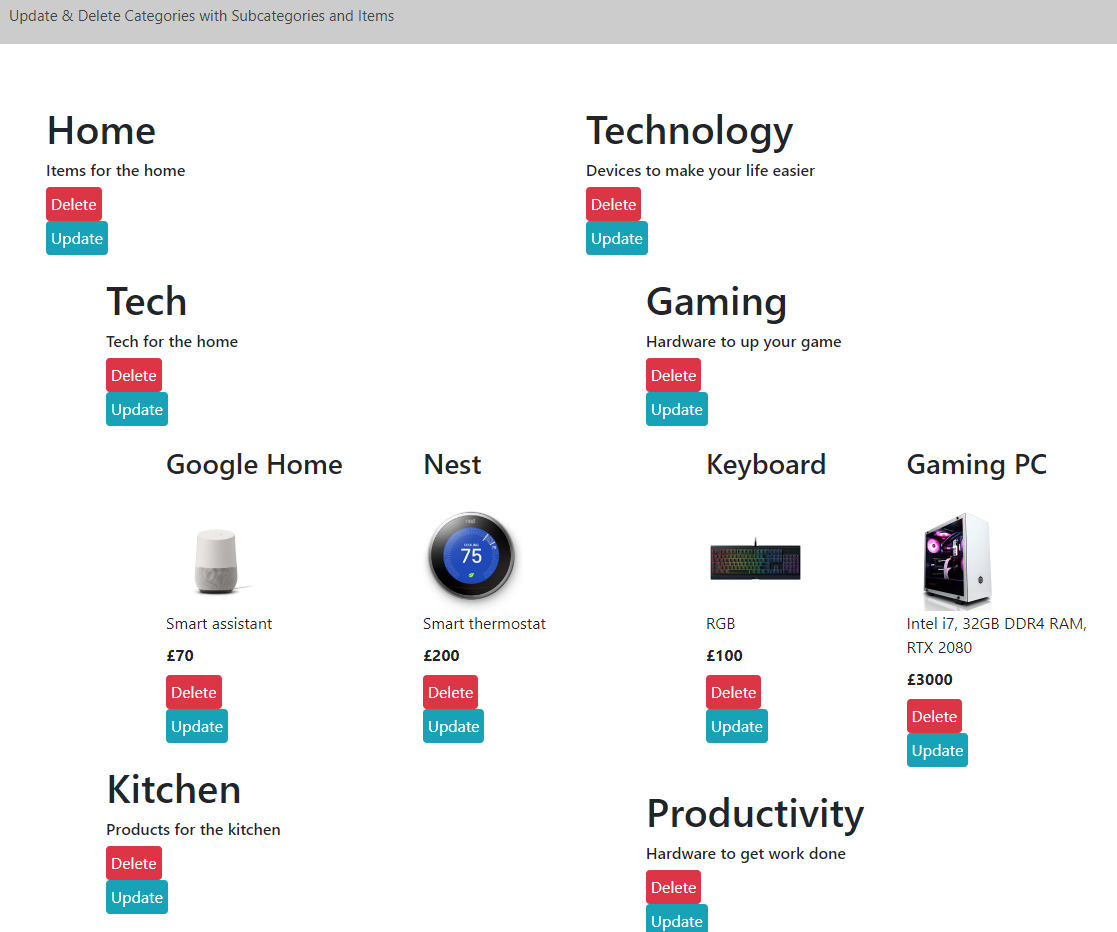
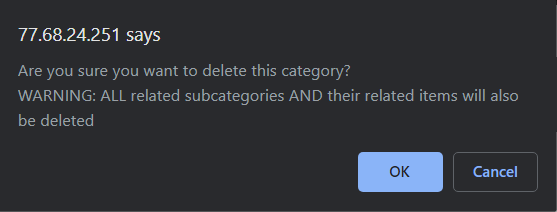
To access a function, click the one you want, and it will expand. This is to keep the site looking clean and tidy.

A category must be added before a subcategory, and a subcategory before an item. This is because the dropdown requires an input that comes from the function preceding it. If a user skips a field and submits, they will be alerted that all fields are required.

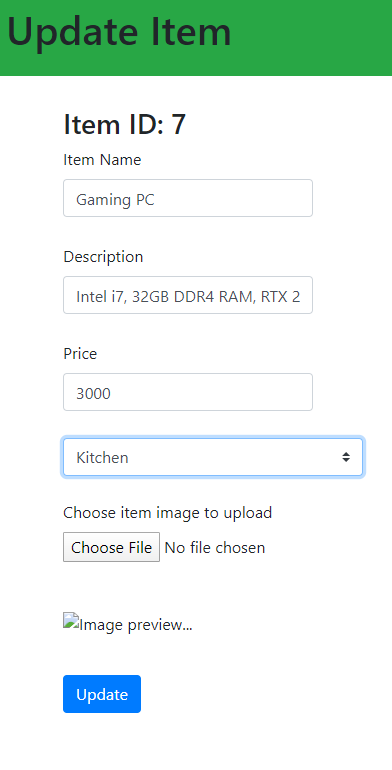


Adding an item is the same as a subcategory except a user must upload an image this time and select a subcategory instead of category

Once an item has been submitted, it’s displayed in the shop page (p19) and the update/delete admin control (p15), with its corresponding subcategory and parent category.

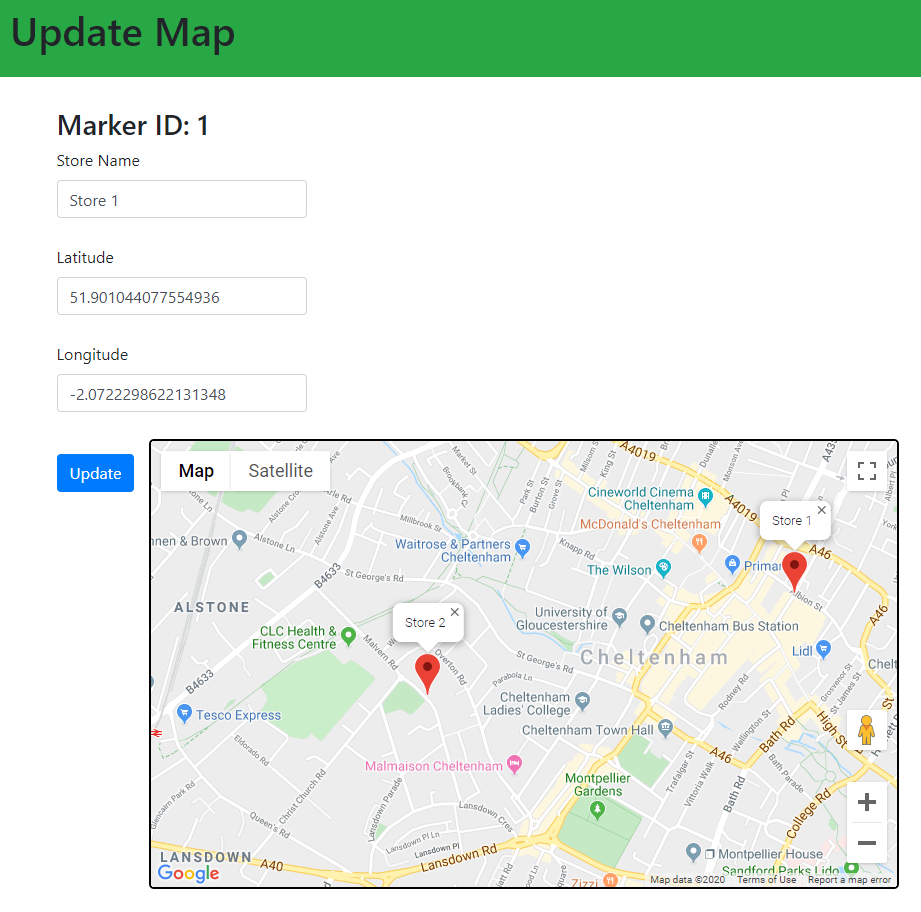
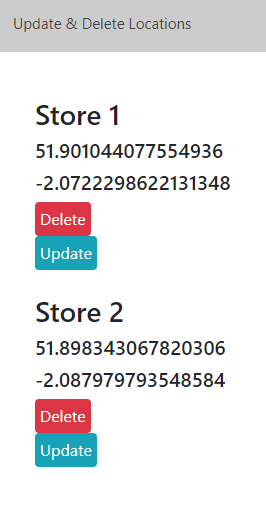
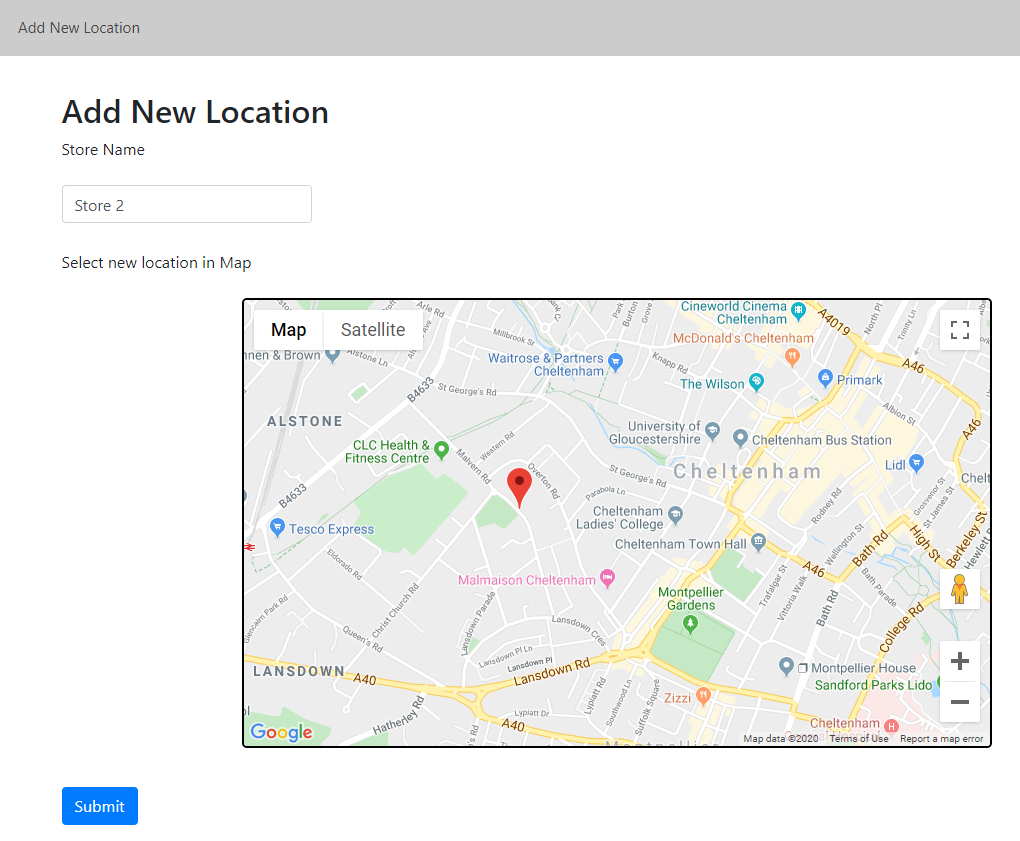


Admins can view items indented under subcategories, under categories, all with buttons to delete and update



Deleting a category will display this message and will delete everything underneath it

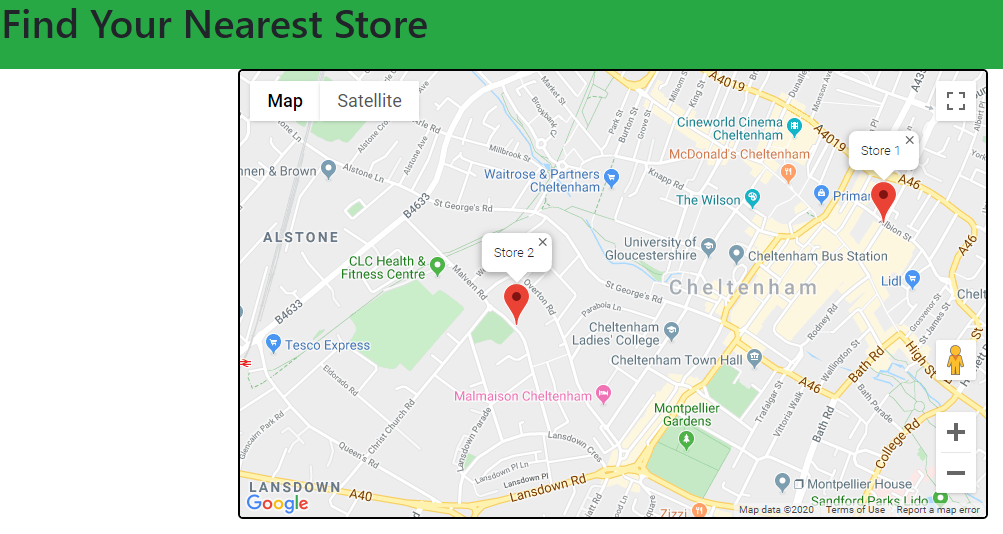
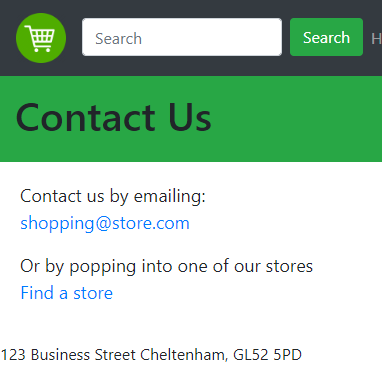
Update buttons takes the user to the update page for that type of element, with all the fields filled out and ready to edit



Users can add a location by dragging the marker, entering a name and submitting

Unfortunately, the users can’t drag the marker to update a location, they must manually change the coordinates

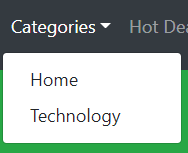
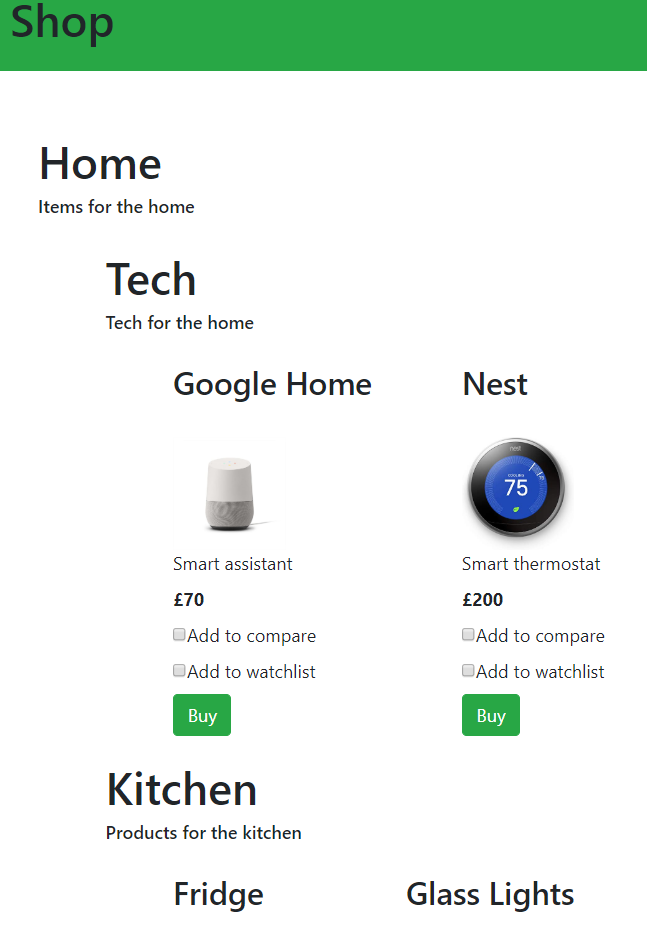
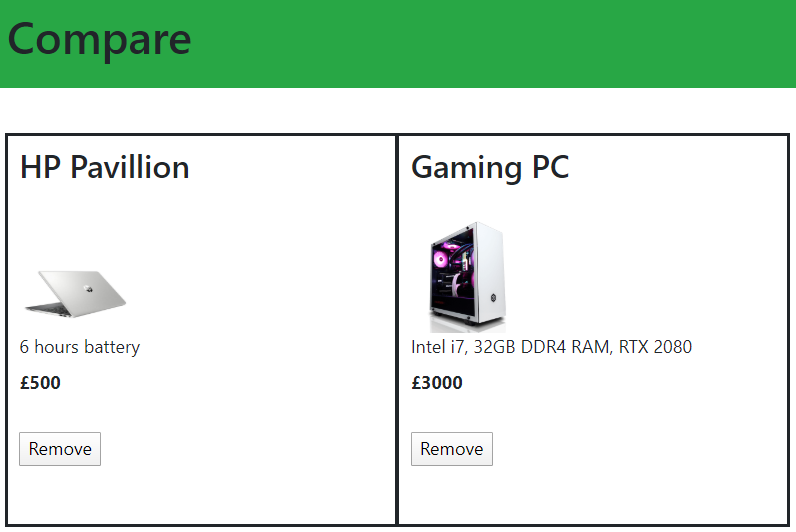
Locations are displayed in the update/delete admin control where they can be deleted and updated



The Locations added in the admin page appear here

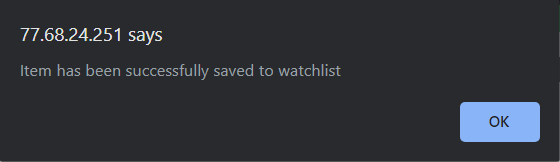
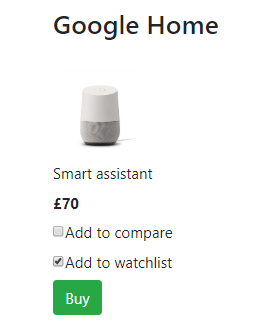
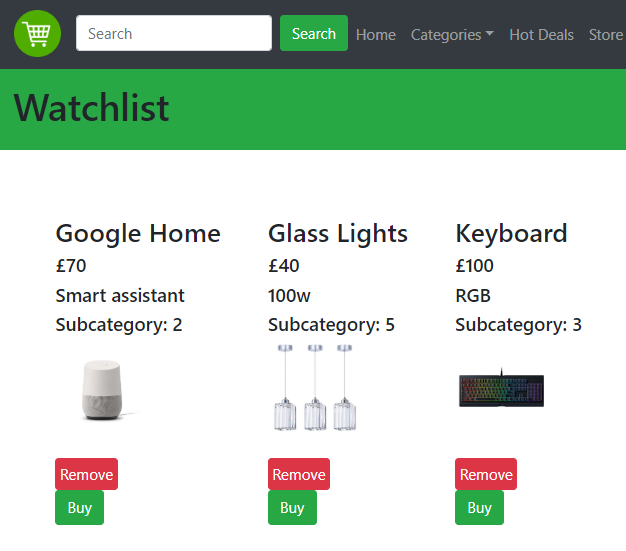
Opens link to user’s email client with the subject: Enquiry

Link to store locator



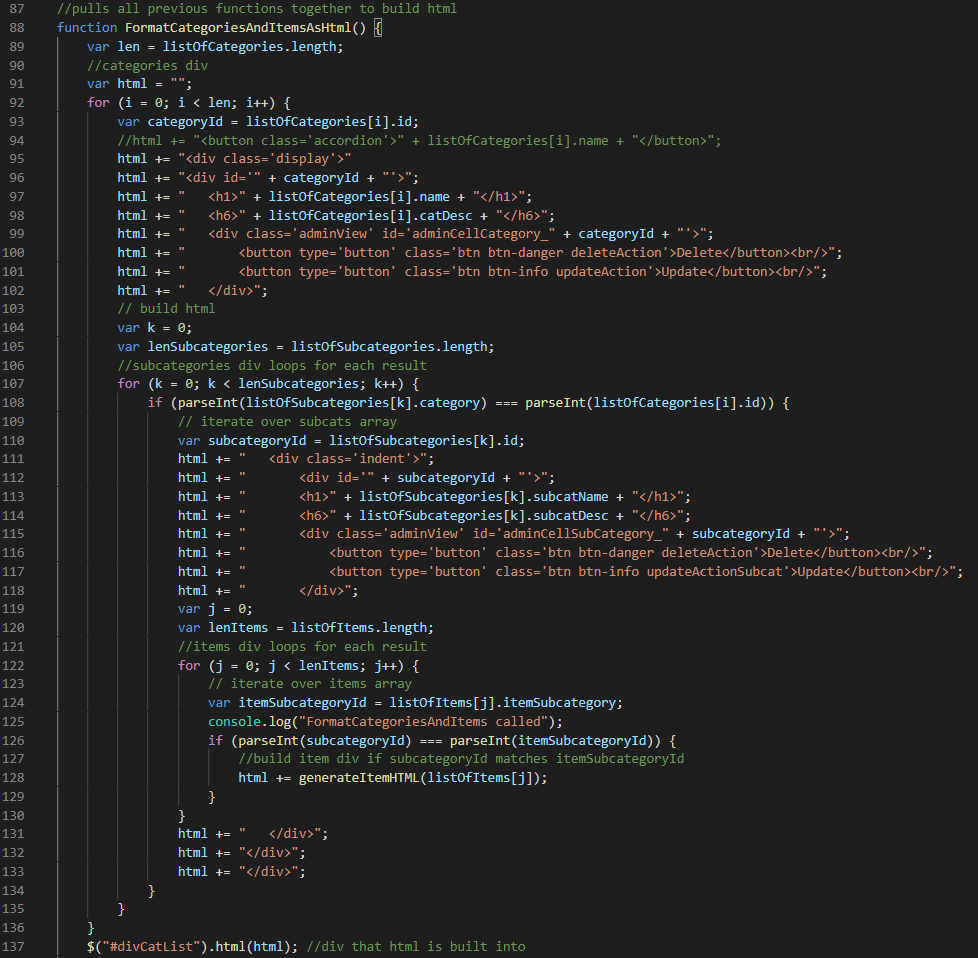
Items have options to add to compare and watchlist and to buy (button takes user to “under construction” page). Adding to compare will display the item in the compare section that appears beneath the shop. A maximum 2 items can be compared. Removing an item will uncheck its checkbox.

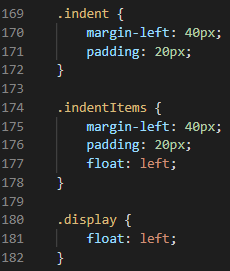
Categories added will appear in the navigation dropdown, user is taken to where that category is on the shop page



Adding an item to watchlist will alert the user and save it to the database. To access the watchlist a user must use the public login (p11). From there they can buy or remove an item. Buying will redirect to the “under construction” page. Removing an item will remove it from the database, but not the shop.

# Interesting Use of HTML/CSS/JavaScript



The interesting use of HTML/CSS/JavaScript in this site all comes from the ‘listCat.js’ JavaScript file. This is the code responsible for assembling the items within the subcategories within both the categories on the shop page and admin page. It decides which buttons or information to display depending on the page URL; the admin page will hide the checkboxes and buy button and the shop page show them and hide the admin update/delete. This code is interesting because it creates divs using JavaScript meaning everything is in one file and you can use your variables directly within the HTML.

The values from the DB are all put into an array first because it works better for obtaining the ID when implementing it in the HTML. Earlier in the code, there is a function to delete objects that cascades to delete all child objects with corresponding ID’s. It does this by call the next function to delete which triggers the one after then and then deletes the ID for its own function.

This makes the page dynamic as everything is built using values from the DB so it’ll only ever display what’s there.

# Shortcomings

While this site is functional, some things could be improved.

1. Images have to use full path to locate them on client-side website. An upload function could be added to add files to a main server to be all in one place.
2. The site uses lots of files making it slower for browser to load. A way to negate this would be to use a single page application where all the HTML is in a main file and uses JavaScript to hide and show elements, giving the illusion of a multi-paged site whilst increasing load times
3. A problem with client-side is you can’t guarantee you have permission to run IndexedDB or JavaScript so it might not work altogether. SQL or other databases are more streamlined and easier to use than IndexedDB
4. Logins can’t be easily encrypted on client-side. You would normally be comparing hash values rather than storing values in plaintext like this site. Login details can currently be accessed through developer tools looking at the DB
5. Hot deals page does nothing – in future the site could add a checkbox in the add item form to add to the hot deals page
6. Carousel isn’t linked to IndexedDB, images come from their file path – in future it would be linked to IndexedDB to show items with some information
7. There are currently no cookies on the site, so the logins only serve to redirect to a specific page. Future implementations would have cookies for users to properly log in and out

# References

**Advice**

UKHO and Clarke, J. (2019) ‘Consultations’.

**IndexedDB wrapper**

CT4009, 2019. *IndexedDB Wrapper API.*[Online]  
Available at: moodle.glos.ac.uk

[Accessed 2019 November 18].

**Images**

*Customize Gamer Dragon Gaming PC* (no date). Available at: https://www.cyberpowerpc.com/system/Gamer-Dragon (Accessed: 6 January 2020).

*Refrigerators in Calgary | Atlas Appliances* (no date). Available at: https://www.atlasappliances.com/en/catalog/product/247656-Samsung-RF26J7500SR (Accessed: 6 January 2020).

*Google Home Speaker | Walmart Canada* (no date). Available at: https://www.walmart.ca/en/ip/google-home-speaker/6000197114649 (Accessed: 6 January 2020).

*Google, T3007ES, Nest Learning Thermostat, 3rd Gen, Smart Thermostat, Stainless Steel, Works With Alexa - - Amazon.com* (no date). Available at: https://www.amazon.com/Nest-T3007ES-Thermostat-Temperature-Generation/dp/B0131RG6VK (Accessed: 6 January 2020).

*Amazon.com: Razer Cynosa Chroma Gaming Keyboard: Customizable Chroma RGB Lighting - Individually Backlit Keys - Spill-Resistant Design - Programmable Macro Functionality: Computers & Accessories* (no date). Available at: https://www.amazon.com/Razer-Cynosa-Chroma-Gaming-Keyboard/dp/B075KMZ4MX?ref\_=fsclp\_pl\_dp\_2 (Accessed: 6 January 2020).

*HP 17.3" Intel Pentium 8GB RAM, 1TB HDD Touchscreen Laptop Bundle - 9187738 | HSN* (no date). Available at: https://www.hsn.com/products/hp-173-intel-pentium-8gb-ram-1tb-hdd-touchscreen-laptop/9187738 (Accessed: 6 January 2020).

*SHENGQINGTOP Modern Crystal Pendant Light for Kitchen Island, Bedroom, Living Room (Cuboid Pendant Light) - - Amazon.com* (no date). Available at: https://www.amazon.com/ShengQing-Crystal-Pendant-1-Light-Lighting/dp/B07CNJL9ZP?th=1 (Accessed: 6 January 2020).

*Buy, cart, circle, ecommerce, green, shopping, trolley icon* (no date). Available at: https://www.iconfinder.com/icons/536262/buy\_cart\_circle\_ecommerce\_green\_shopping\_trolley\_icon (Accessed: 8 January 2020).

*Searching for array elements in IndexedDB* (no date). Available at: https://www.raymondcamden.com/2012/08/10/Searching-for-array-elements-in-IndexedDB (Accessed: 8 January 2020).

*Bootstrap Buttons* (no date). Available at: https://www.w3schools.com/bootstrap/bootstrap\_buttons.asp (Accessed: 8 January 2020).*Navbar · Bootstrap* (no date). Available at: https://getbootstrap.com/docs/4.0/components/navbar/ (Accessed: 8 January 2020).

*HTML DOM Input Checkbox Object* (no date). Available at: https://www.w3schools.com/jsref/dom\_obj\_checkbox.asp (Accessed: 8 January 2020).*Tryit Editor v3.6* (no date). Available at: https://www.w3schools.com/tags/tryit.asp?filename=tryhtml\_link\_mailto (Accessed: 8 January 2020).

**Accordion**

*How To Create an Accordion* (no date). Available at: https://www.w3schools.com/howto/howto\_js\_accordion.asp (Accessed: 8 January 2020).

**IndexedDB**

*IndexedDB fundamentals - Plus a indexeddb example - tutorial* (no date). Available at: https://www.vanamco.com/2014/11/14/indexeddb-fundamentals-plus-a-indexeddb-example-tutorial/ (Accessed: 8 January 2020).

*Build a basic web app with IndexedDB - DEV Community 👩‍💻👨‍💻* (no date). Available at: https://dev.to/andyhaskell/build-a-basic-web-app-with-indexeddb-38ef (Accessed: 8 January 2020).

*Using IndexedDB - Web APIs | MDN* (no date). Available at: https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB\_API/Using\_IndexedDB (Accessed: 8 January 2020).

**Carousel**

*Tryit Editor v3.6* (no date). Available at: https://www.w3schools.com/bootstrap4/tryit.asp?filename=trybs\_ref\_js\_carousel2 (Accessed: 8 January 2020).

**Navbar**

*Tryit Editor v3.6* (no date). Available at: https://www.w3schools.com/bootstrap4/tryit.asp?filename=trybs\_navbar\_form&stacked=h (Accessed: 8 January 2020).

*Bootstrap 4 Navigation Bar* (no date). Available at: https://www.w3schools.com/bootstrap4/bootstrap\_navbar.asp (Accessed: 8 January 2020).

*Tryit Editor v3.6* (no date). Available at: https://www.w3schools.com/bootstrap4/tryit.asp?filename=trybs\_navbar\_dropdown&stacked=h (Accessed: 8 January 2020).

**Login Form**

*Tryit Editor v3.6* (no date). Available at: https://www.w3schools.com/bootstrap4/tryit.asp?filename=trybs\_form\_basic&stacked=h (Accessed: 8 January 2020).

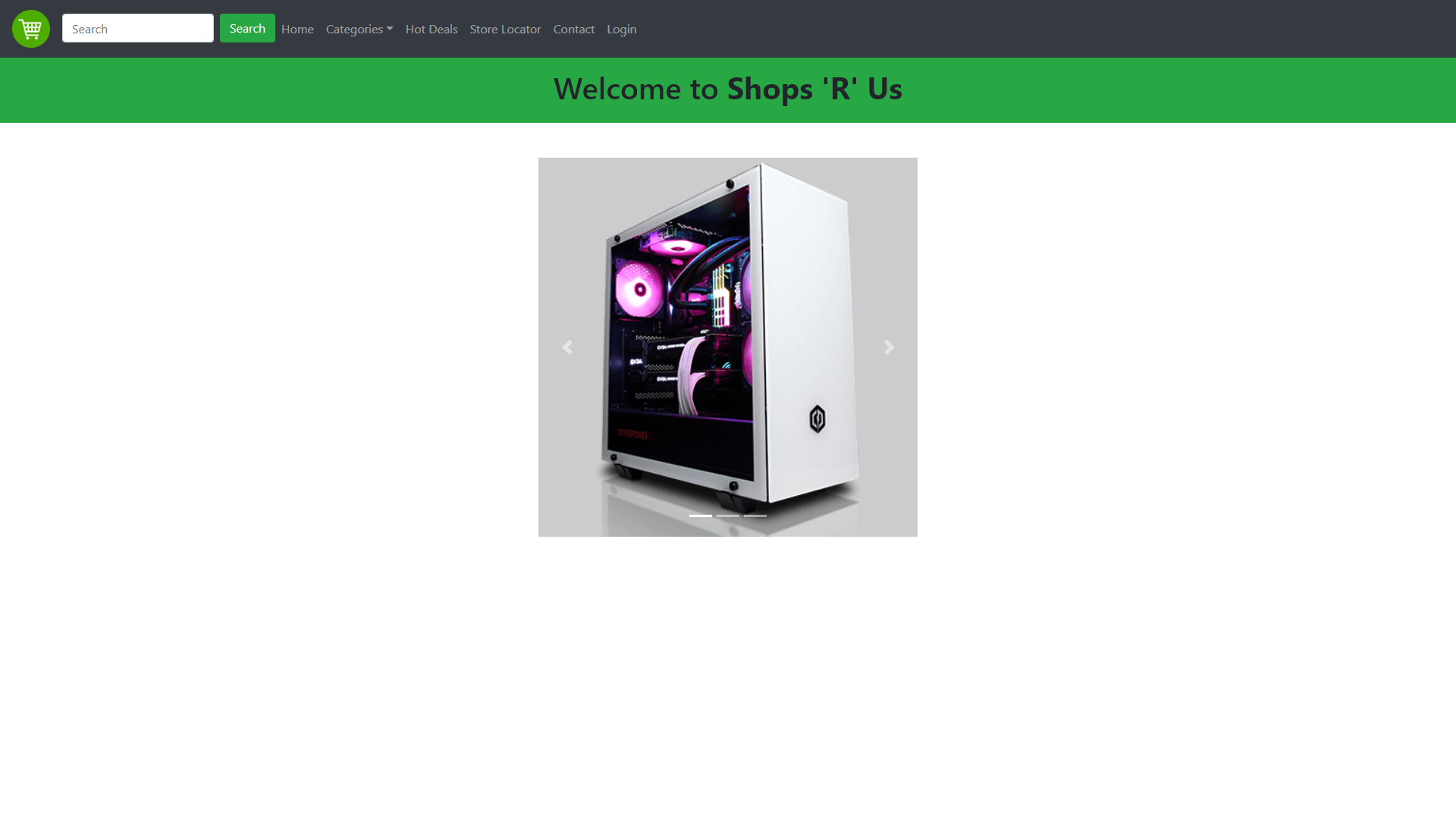
**Checkbox**

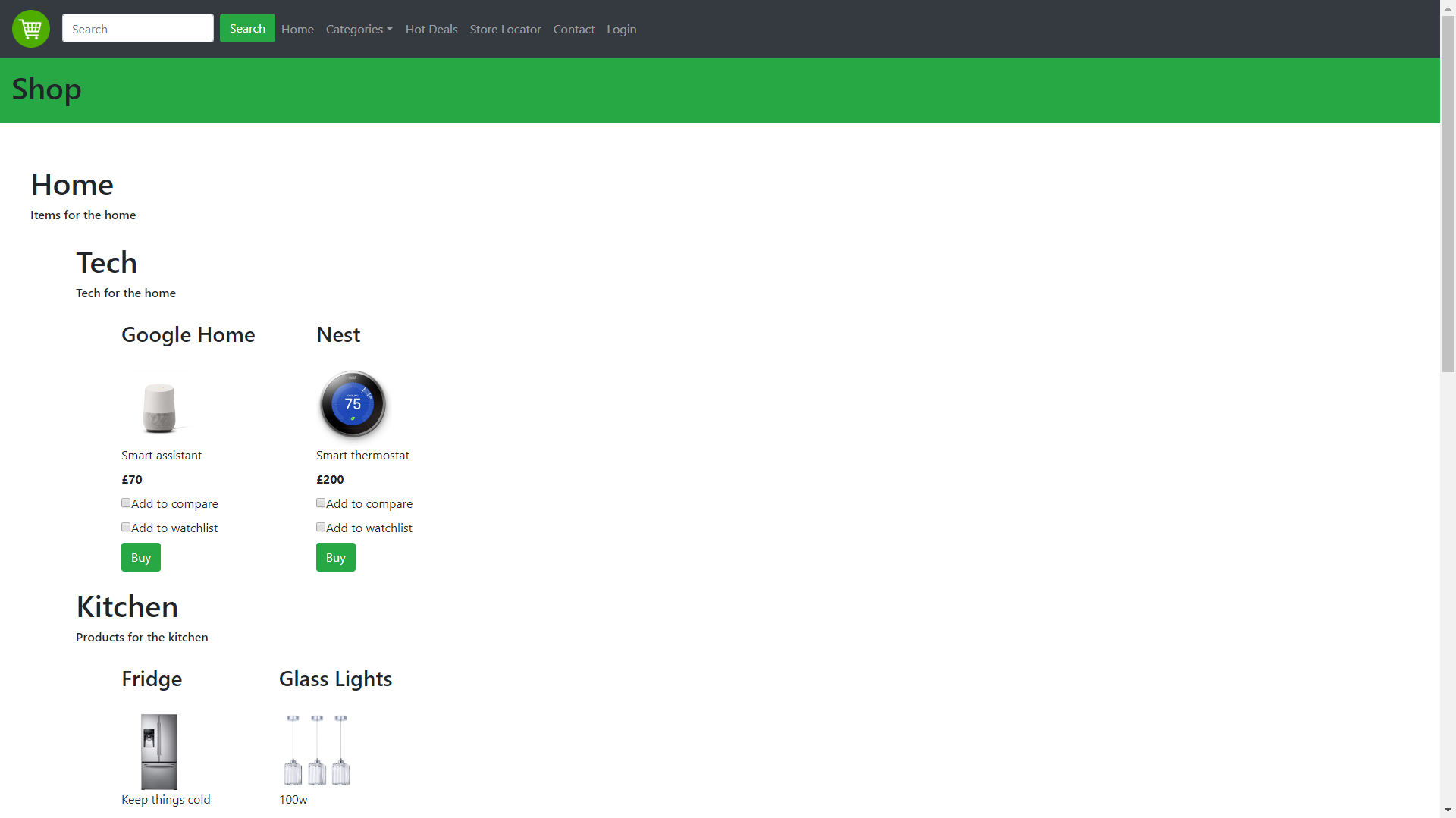
*HTML input type="checkbox"* (no date). Available at: https://www.w3schools.com/tags/att\_input\_type\_checkbox.asp (Accessed: 8 January 2020).

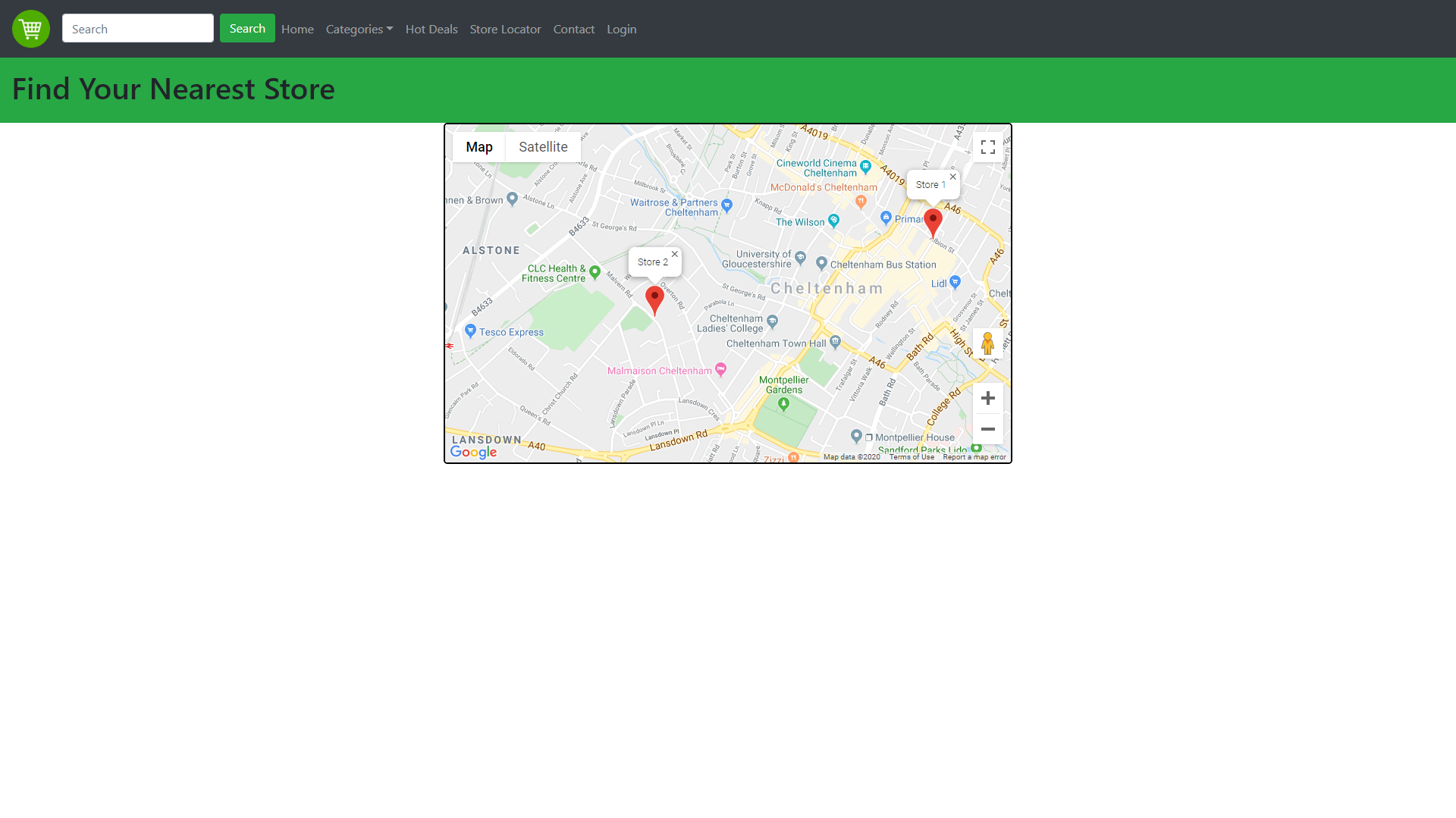
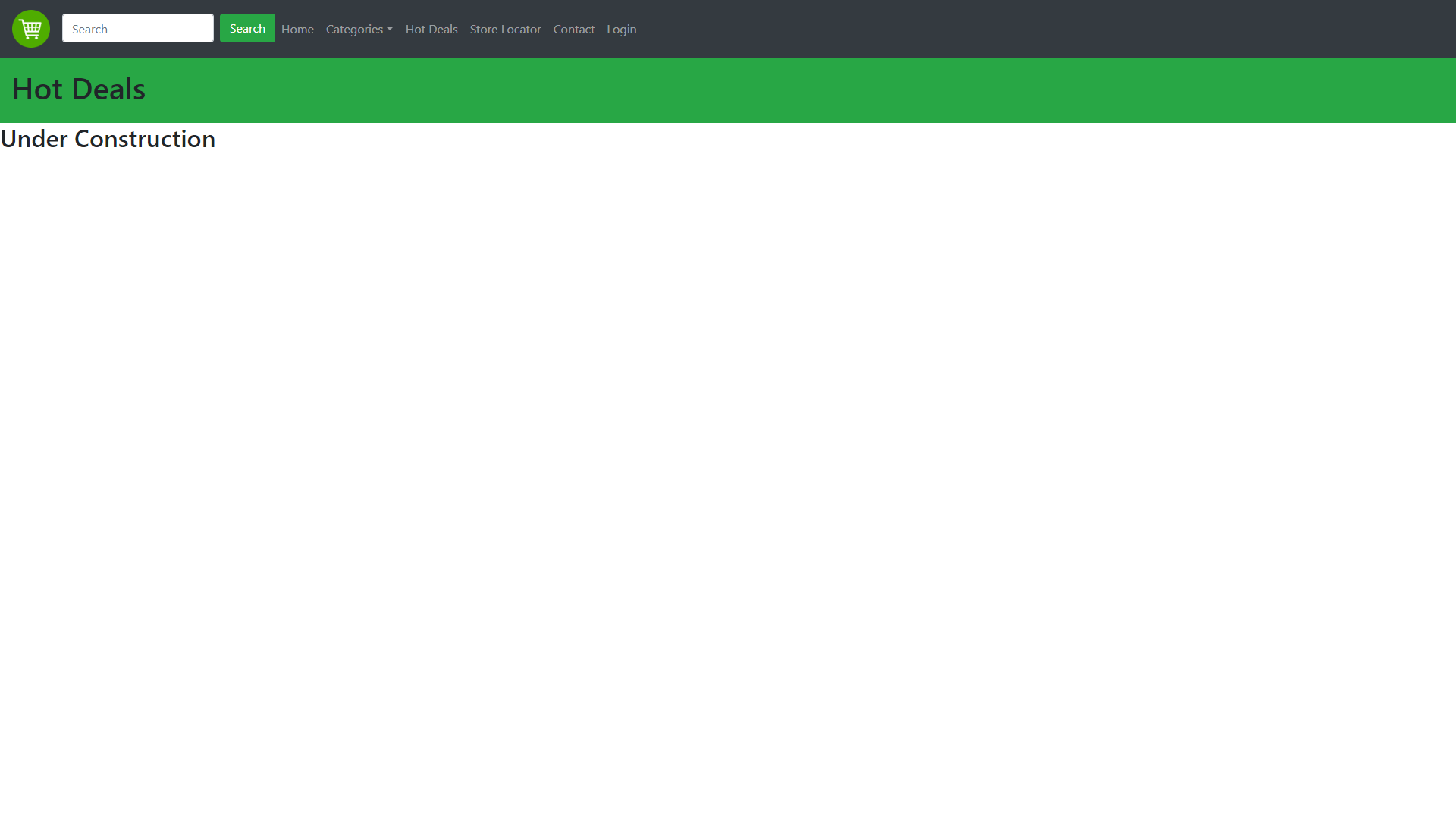
**== vs. ===**

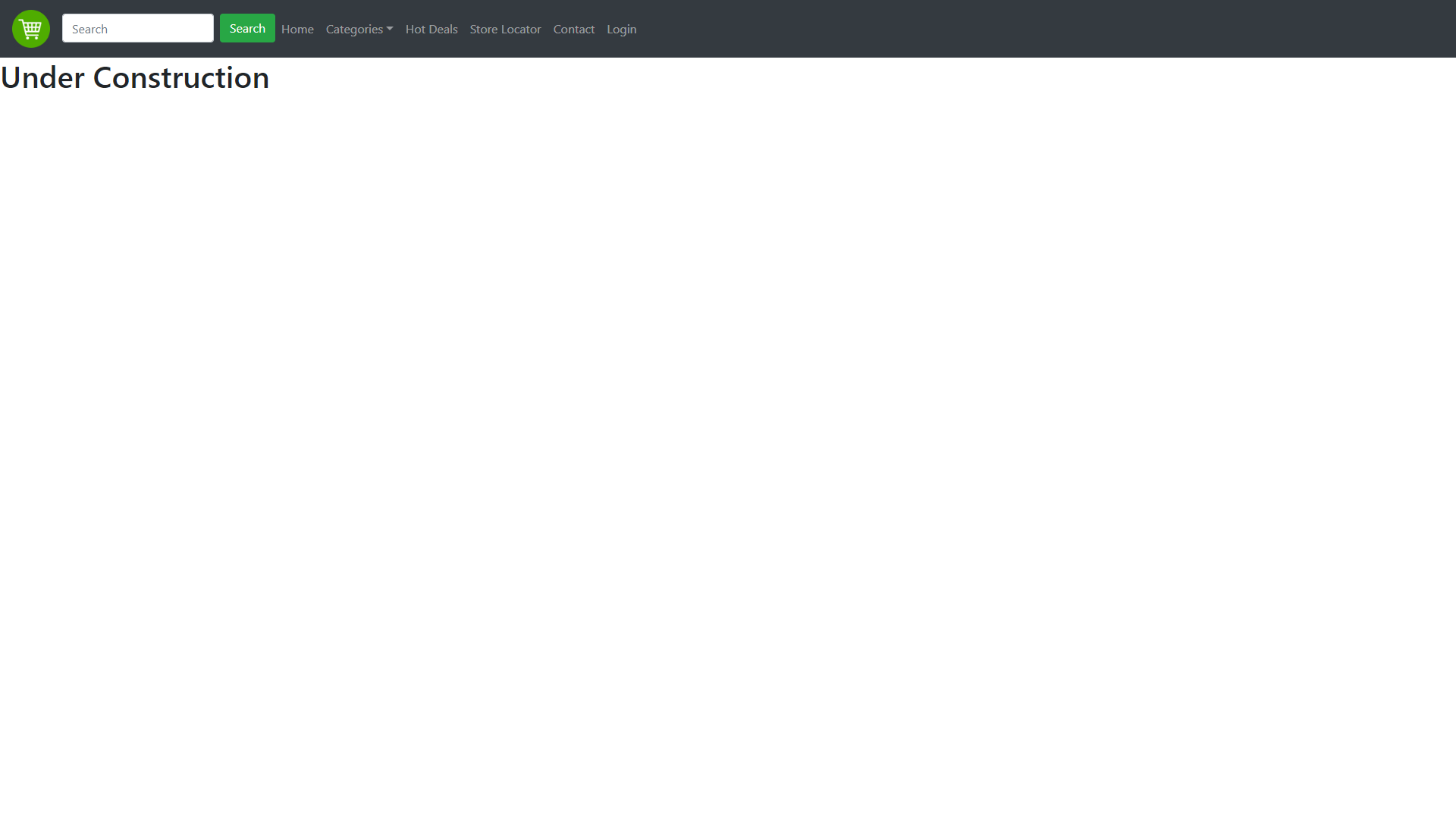
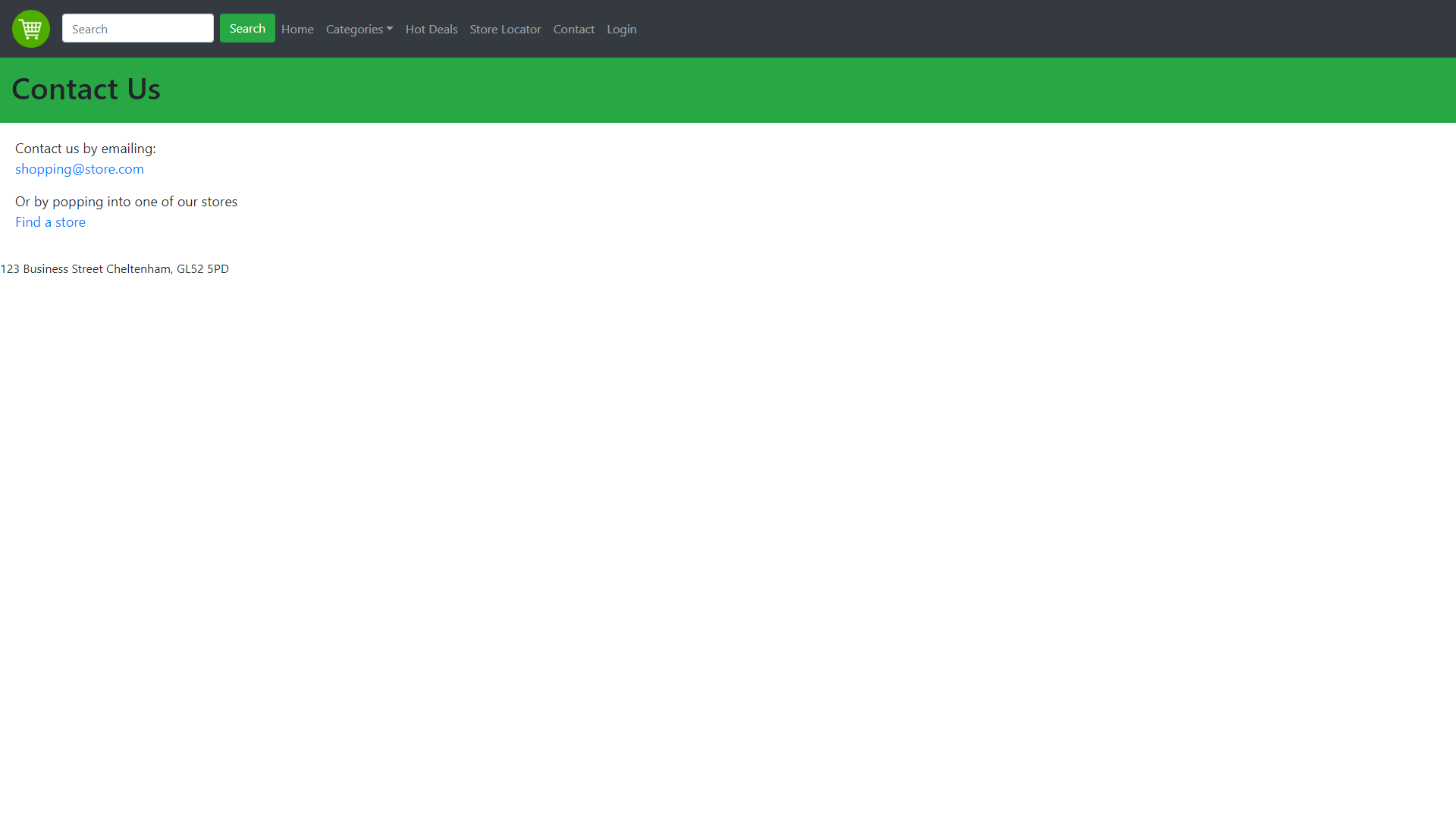
*JavaScript — Double Equals vs. Triple Equals - codeburst* (no date). Available at: https://codeburst.io/javascript-double-equals-vs-triple-equals-61d4ce5a121a (Accessed: 8 January 2020).

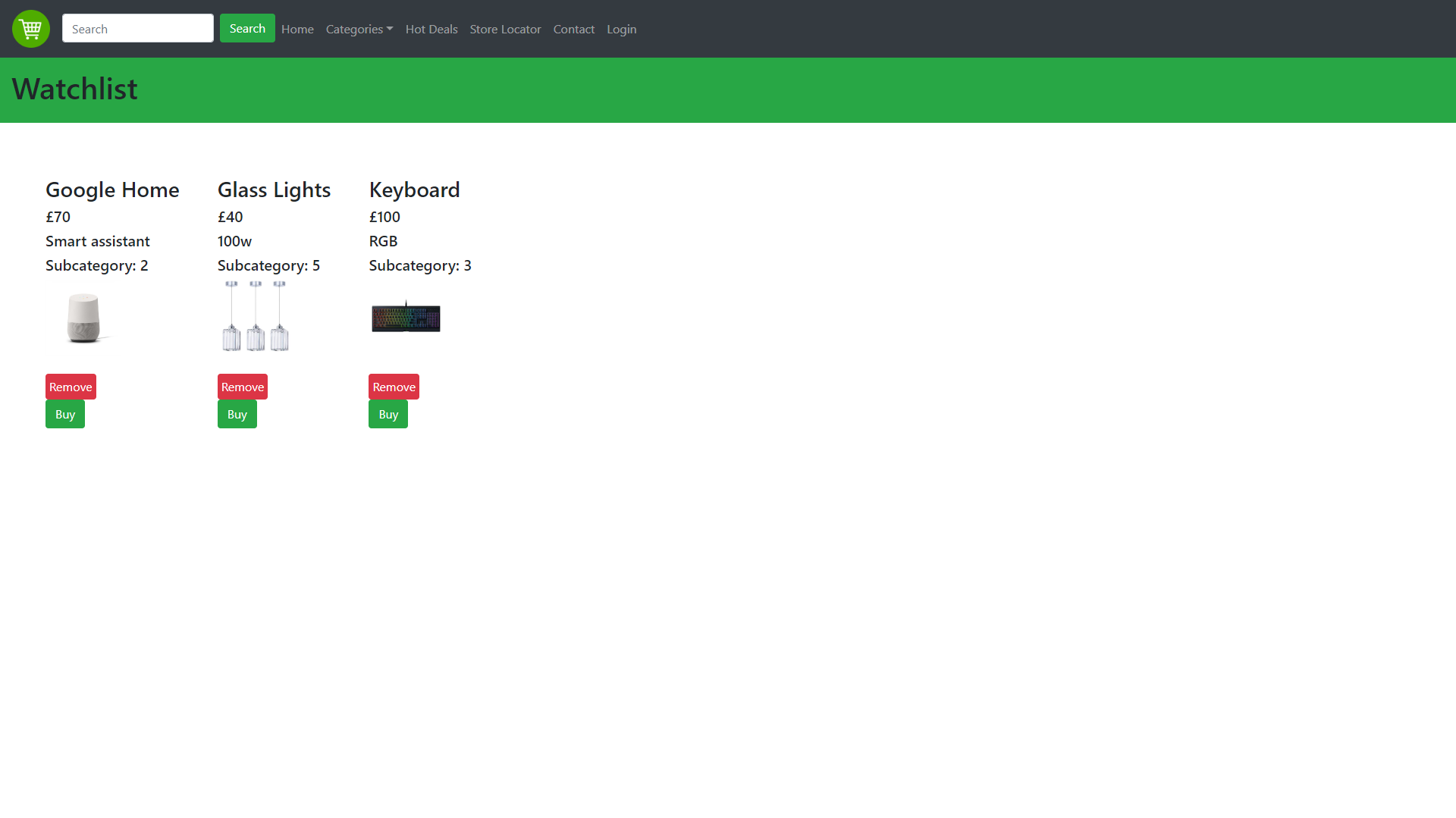
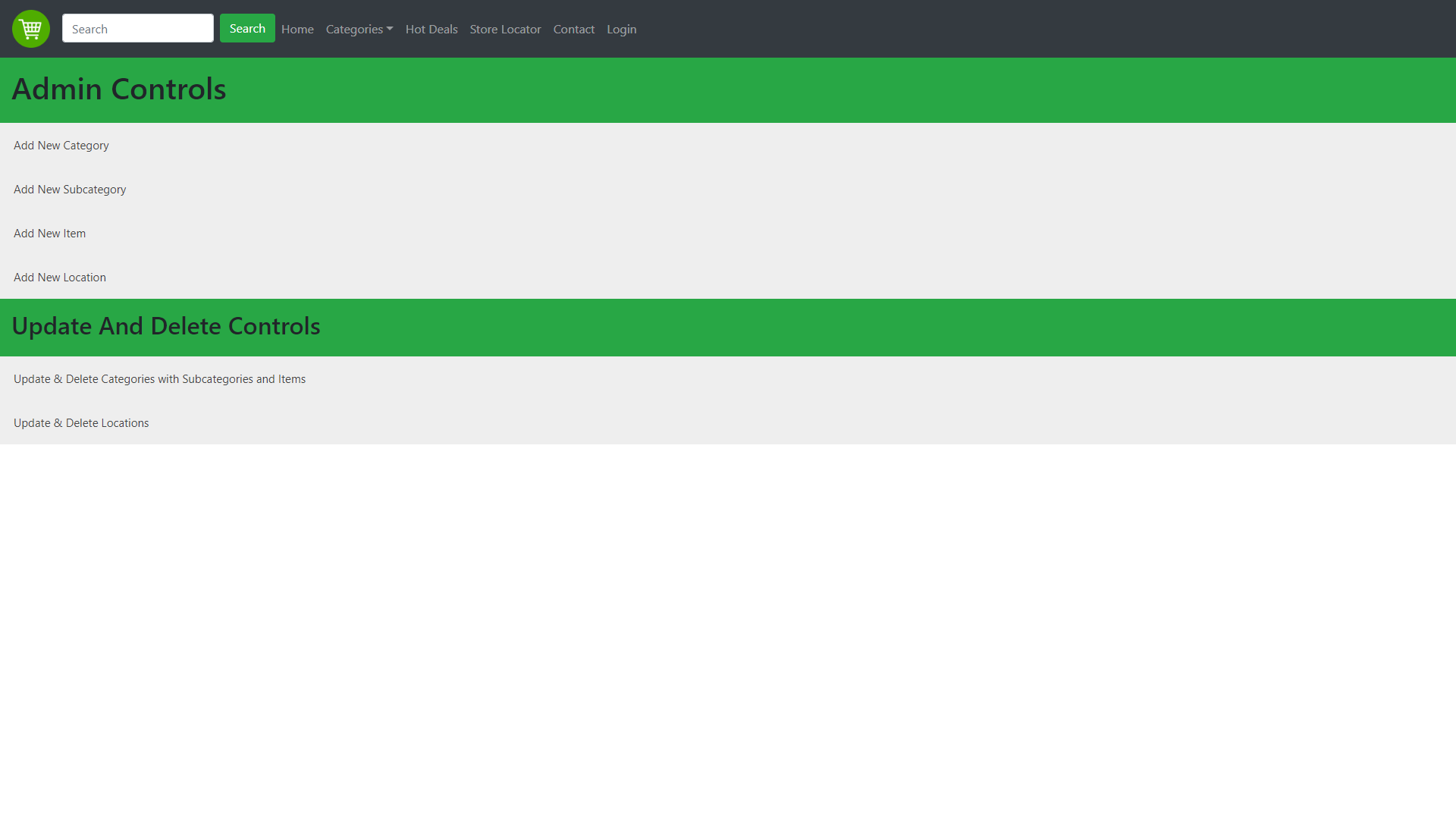
# Appendix 1









# Appendix 2

## styles.css

1. .div\_block\_right {
2. float: right;
3. clear: both;
4. margin: 10px;
5. top: 100px;
6. z-index: 1000;
7. }
8. #contact {
9. margin: 10px;
10. padding: 10px;
11. font-size: 18px;
12. }
13. .title {
14. padding: 15px;
15. background-color: #28a745;
16. width: 100%;
17. }
18. #homeTitle {
19. float: center;
20. text-align: center;
21. }
22. .nav-item.active{
23. background-color: transparent;
24. }
25. .carousel-inner img {
26. width: 500px;
27. height: 500px;
28. }
29. #Carousel {
30. margin: auto;
31. width: 500px;
32. height: 500px;
33. }
34. .form {
35. float: right;
36. margin-right: 20px;
37. }
38. #form-inline {
39. width: 200px;
40. }
41. .container {
42. padding: 30px;
43. border: 2px;
44. }
45. .formContainer {
46. padding: 15px;
47. }
48. #login.formContainer {
49. top: 76px;
50. left: 450px;
51. position: absolute;
52. z-index: 9999;
53. height: fit-content;
54. width: 400px;
55. clear: both;
56. float: left;
57. background-color: rgb(61, 61, 61);
58. opacity: 0.8;
59. color: white;
60. border-radius: 7px;
61. }
62. .btn-primary {
63. float: right;
64. }
65. label.pwd {
66. text-align: left;
67. }
68. #map {
69. left: 400px;
70. top: 100px;
71. }
72. li .indent {
73. padding-left: 3em;
74. }
75. a:hover {
76. background-color: #1d2630;
77. border-radius: 5px;
78. }
79. a:hover.dropdown-item {
80. background-color: #ccc;
81. border-radius: 3px;
82. }
83. .accordion {
84. background-color: #eee;
85. color: #444;
86. cursor: pointer;
87. padding: 18px;
88. width: 100%;
89. border: none;
90. text-align: left;
91. outline: none;
92. font-size: 15px;
93. transition: 0.4s;
94. }
95. .active,
96. .accordion:hover {
97. background-color: #ccc;
98. }
99. .accordionUpdate {
100. background-color: #28a745;
101. }
102. .panel {
103. margin: 5px;
104. padding: 0 10px;
105. display: none;
106. background-color: white;
107. overflow: hidden;
108. }
109. #mapInput, #mapView {
110. width: auto;
111. height: 450px;
112. border-bottom-left-radius: 5px;
113. border-bottom-right-radius: 5px;
114. border-top-left-radius: 5px;
115. border-top-right-radius: 5px;
116. margin: auto;
117. border: 2px solid;
118. border-color: rgb(0, 0, 0);
119. }
120. #markerList {
121. float: left;
122. margin: 10px;
123. }
124. .leftCell {
125. max-width: 200px;
126. }
127. .leftCellCompare {
128. height: 350px;
129. width: 350px;
130. float: left;
131. border: 3px solid;
132. margin-top: 10px;
133. margin-left: -2px;
134. margin-bottom: -2px;
135. padding: 10px;
136. }
137. .categoryDisplayText {
138. border: 1px solid silver;
139. height: 300px;
140. min-width: 500px;
141. }
142. .indent {
143. margin-left: 40px;
144. padding: 20px;
145. }
146. .indentItems {
147. margin-left: 40px;
148. padding: 20px;
149. float: left;
150. }
151. .display {
152. float: left;
153. }
154. .btn-primary {
155. float: left;
156. margin: 15px;
157. margin-left: 0px;
158. }
159. .form-control.text {
160. margin-bottom: 3px;
161. width: 250px;
162. }
163. .browser-default.custom-select {
164. width: 300px;
165. }
166. #divCatList.container {
167. margin: 25px;
168. float: left;
169. }
170. .btn.btn-danger,.btn.btn-info {
171. padding: 4px;
172. }
173. #divCompare,#divCompareList.container {
174. display: inline-block;
175. width: 100%;
176. }
177. #watchItem {
178. float: left;
179. margin: 5px;
180. padding: 20px;
181. }

## HomePage.html

1. <!DOCTYPE html>
2. <html lang="en-gb">
3. <head>
4. <!--helps search engines define what the purpose of the site is-->
5. <meta name="description" content="catalogue and shopping website">
6. <meta name="keywords" content="shopping, search, catalogue">
7. <!--defines correct character set-->
8. <meta charset="utf-8">
9. <!--ensures site will render correctly on all platforms, primarily mobile-->
10. <meta name="viewport" content="width=device-width, initial-scale=1">
11. <link rel="stylesheet" href="../css/styles.css" />
12. <!--links to the Boostrap CDN-->
13. <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
14. <!-- jquery link -->
15. <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
16. <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
17. <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
18. <!-- indexeddb wrapper and main javascript file links -->
19. <script src="../js/main.js"></script>
20. <script src="../js/lib/indexeddb.js"></script>
21. <!--title of site-->
22. <title>Catalogue Website Home</title>
23. </head>
24. <body>
25. <!--defines type of navbar and makes navbar dark and controls spacing of items-->
26. <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
27. <!-- logo link to home page -->
28. <a class="navbar-brand" href="../home/HomePage.html">
29. <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">
30. </a>
31. <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"
32. aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
33. <span class="navbar-toggler-icon"></span>
34. </button>
35. <div class="collapse navbar-collapse" id="navbarNavDropdown">
36. <ul class="navbar-nav">
37. <!--creates search bar form-->
38. <li class="nav-item active">
39. <div id="search">
40. <form class="form-inline" action="../Search/results.html">
41. <input class="form-control mr-sm-2" type="text" placeholder="Search">
42. <button class="btn btn-success" type="submit">Search</button>
43. </form>
44. </div>
45. </li>
46. <li class="nav-item">
47. <!--link to home page-->
48. <a class="nav-link" href="../home/HomePage.html">Home</a>
49. </li>
50. <li class="nav-item dropdown">
51. <!--creates dropdown-->
52. <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>
53. <!--name and type of dropdown-->
54. <div id="dropdown" class="dropdown-menu"></div>
55. </li>
56. <li class="nav-item">
57. <!--link to hot deals page-->
58. <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>
59. </li>
60. <li class="nav-item">
61. <!--link to store locator page-->
62. <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>
63. </li>
64. <li class="nav-item">
65. <!--link to contact page-->
66. <a class="nav-link" href="../Contact/contact.html">Contact</a>
67. </li>
68. <li class="nav-item">
69. <a class="nav-link" id="loginToggle">Login</a>
70. </li>
71. </ul>
72. </div>
73. </nav>
74. <!--creates login form-->
75. <div id="login" class="formContainer">
76. <div class="form-group">
77. <label for="email">Email</label>
78. <!--defines type of input, id, placeholder and requires user to make input-->
79. <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>
80. </div>
81. <div class="form-group">
82. <label for="key">Password</label>
83. <!--defines type of input, id, placeholder and requires user to make input-->
84. <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" encrypt="bcrypt" salt="abc..." rounds="4" required>
85. </div>
86. <button type="button" id="btnSubmitLogin" class="btn btn-primary">Submit</button>
87. </div>
88. <div class="title" id="homeTitle">
89. <h1>Welcome to <b>Shops 'R' Us</b></h1>
90. </div>
91. <!--creates carousel-->
92. <div class="container mt-3">
93. <div id="Carousel" class="carousel slide" data-ride="carousel">
94. <!-- Indicators -->
95. <ul class="carousel-indicators">
96. <li data-target="#Carousel" data-slide-to="0" class="active"></li>
97. <li data-target="#Carousel" data-slide-to="1"></li>
98. <li data-target="#Carousel" data-slide-to="2"></li>
99. </ul>
100. <!-- The slideshow -->
101. <div class="carousel-inner">
102. <div class="carousel-item active">
103. <!--location of specific image with description for accessibility purposes and sizes-->
104. <img src="../images/pc.jpeg" alt="pc" width="200" height="400">
105. </div>
106. <div class="carousel-item">
107. <img src="../images/laptop.jpg" alt="laptop" width="200" height="400">
108. </div>
109. <div class="carousel-item">
110. <img src="../images/ghome.jpeg" alt="google home" width="200" height="400">
111. </div>
112. </div>
113. <!-- Left and right controls -->
114. <a class="carousel-control-prev" href="#Carousel" data-slide="prev">
115. <span class="carousel-control-prev-icon"></span>
116. </a>
117. <a class="carousel-control-next" href="#Carousel" data-slide="next">
118. <span class="carousel-control-next-icon"></span>
119. </a>
120. </div>
121. </div>
122. </body>
123. </html>

## catTemplate.html

1. <!DOCTYPE html>
2. <html lang="en-gb">
3. <head>
4. <!--helps search engines define what the purpose of the site is-->
5. <meta name="description" content="catalogue and shopping website">
6. <meta name="keywords" content="shopping, search, catalogue">
7. <!--defines correct character set-->
8. <meta charset="utf-8">
9. <!--ensures site will render correctly on all platforms, primarily mobile-->
10. <meta name="viewport" content="width=device-width, initial-scale=1">
11. <link rel="stylesheet" href="../css/styles.css" />
12. <!--links to the Boostrap CDN-->
13. <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
14. <!-- jquery link -->
15. <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
16. <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
17. <!-- bootstrap js link -->
18. <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
19. <!-- indexeddb wrapper and main javascript file links -->
20. <script src="../js/main.js"></script>
21. <script src="../js/lib/indexeddb.js"></script>
22. <!--title of site-->
23. <title>Catalogue Website Shop</title>
24. </head>
25. <body>
26. <!--defines type of navbar and makes navbar dark and controls spacing of items-->
27. <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
28. <!-- logo link to home page -->
29. <a class="navbar-brand" href="../home/HomePage.html">
30. <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">
31. </a>
32. <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"
33. aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
34. <span class="navbar-toggler-icon"></span>
35. </button>
36. <div class="collapse navbar-collapse" id="navbarNavDropdown">
37. <ul class="navbar-nav">
38. <!--creates search bar form-->
39. <li class="nav-item active">
40. <div id="search">
41. <!-- search form goes to search results page -->
42. <form class="form-inline" action="../Search/results.html">
43. <input class="form-control mr-sm-2" type="text" placeholder="Search">
44. <button class="btn btn-success" type="submit">Search</button>
45. </form>
46. </div>
47. </li>
48. <li class="nav-item">
49. <!--link to home page-->
50. <a class="nav-link" href="../home/HomePage.html">Home</a>
51. </li>
52. <li class="nav-item dropdown">
53. <!--creates dropdown-->
54. <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>
55. <!--name and type of dropdown-->
56. <div id="dropdown" class="dropdown-menu"></div>
57. </li>
58. <li class="nav-item">
59. <!--link to hot deals page-->
60. <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>
61. </li>
62. <li class="nav-item">
63. <!--link to store locator page-->
64. <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>
65. </li>
66. <li class="nav-item">
67. <!--link to contact page-->
68. <a class="nav-link" href="../Contact/contact.html">Contact</a>
69. </li>
70. <li class="nav-item">
71. <a class="nav-link" id="loginToggle">Login</a>
72. </li>
73. </ul>
74. </div>
75. </nav>
76. <!--creates login form-->
77. <div id="login" class="formContainer">
78. <div class="form-group">
79. <label for="email">Email</label>
80. <!--defines type of input, id, placeholder and requires user to make input-->
81. <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>
82. </div>
83. <div class="form-group">
84. <label for="key">Password</label>
85. <!--defines type of input, id, placeholder and requires user to make input-->
86. <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>
87. </div>
88. <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>
89. </form>
90. </div>
91. <div class="title">
92. <h1>Shop</h1>
93. </div>
94. <!-- div for all categories, subcats and items -->
95. <div id="divCatList" class="container">
96. </div>
97. <!-- comparison div -->
98. <div id="divCompare" class="title">
99. <h1>Compare</h1>
100. </div>
101. <div id="divCompareList" class="container">
102. </div>
103. <script src="../js/ListCat/listCat.js"></script>
104. <script src="../js/ListCat/initCompare.js"></script>
105. <script>
107. </script>
108. </body>
109. </html>

## contact.html

1. <!DOCTYPE html>
2. <html lang="en-gb">
3. <head>
4. <!--helps search engines define what the purpose of the site is-->
5. <meta name="description" content="catalogue and shopping website">
6. <meta name="keywords" content="shopping, search, catalogue">
7. <!--defines correct character set-->
8. <meta charset="utf-8">
9. <!--ensures site will render correctly on all platforms, primarily mobile-->
10. <meta name="viewport" content="width=device-width, initial-scale=1">
11. <link rel="stylesheet" href="../css/styles.css" />
12. <!--links to the Boostrap CDN-->
13. <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
14. <!-- jquery link -->
15. <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
16. <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
17. <!-- bootstrap js link -->
18. <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
19. <!-- indexeddb wrapper and main javascript file links -->
20. <script src="../js/main.js"></script>
21. <script src="../js/lib/indexeddb.js"></script>
22. <!--title of site-->
23. <title>Catalogue Website Contact Us</title>
24. </head>
25. <body>
26. <!--defines type of navbar and makes navbar dark and controls spacing of items-->
27. <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
28. <!-- logo link to home page -->
29. <a class="navbar-brand" href="../home/HomePage.html">
30. <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">
31. </a>
32. <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"
33. aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
34. <span class="navbar-toggler-icon"></span>
35. </button>
36. <div class="collapse navbar-collapse" id="navbarNavDropdown">
37. <ul class="navbar-nav">
38. <!--creates search bar form-->
39. <li class="nav-item active">
40. <div id="search">
41. <!-- search form goes to search results page -->
42. <form class="form-inline" action="../Search/results.html">
43. <input class="form-control mr-sm-2" type="text" placeholder="Search">
44. <button class="btn btn-success" type="submit">Search</button>
45. </form>
46. </div>
47. </li>
48. <li class="nav-item">
49. <!--link to home page-->
50. <a class="nav-link" href="../home/HomePage.html">Home</a>
51. </li>
52. <li class="nav-item dropdown">
53. <!--creates dropdown-->
54. <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>
55. <!--name and type of dropdown-->
56. <div id="dropdown" class="dropdown-menu"></div>
57. </li>
58. <li class="nav-item">
59. <!--link to hot deals page-->
60. <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>
61. </li>
62. <li class="nav-item">
63. <!--link to store locator page-->
64. <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>
65. </li>
66. <li class="nav-item">
67. <!--link to contact page-->
68. <a class="nav-link" href="../Contact/contact.html">Contact</a>
69. </li>
70. <li class="nav-item">
71. <a class="nav-link" id="loginToggle">Login</a>
72. </li>
73. </ul>
74. </div>
75. </nav>
76. <!--creates login form-->
77. <div id="login" class="formContainer">
78. <div class="form-group">
79. <label for="email">Email</label>
80. <!--defines type of input, id, placeholder and requires user to make input-->
81. <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>
82. </div>
83. <div class="form-group">
84. <label for="key">Password</label>
85. <!--defines type of input, id, placeholder and requires user to make input-->
86. <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>
87. </div>
88. <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>
89. </form>
90. </div>
91. <div class="title"><h1>Contact Us</h1></div>
92. <div id="contact">
93. <p>Contact us by emailing:<br/>
94. <!-- opens email draft in user's email client with subject -->
95. <a href="mailto:shopping@store.com?Subject=Enquiry" target="\_top">shopping@store.com</a>
96. </p>
97. <p>Or by popping into one of our stores<br/>
98. <a href="../Map/EventList/EventList.html">Find a store</a> <!-- link to store locator page -->
99. </p>
100. </div>
101. <!--defines footer-->
102. <div id="footer">
103. <footer>This is the footer</footer>
104. </div>
105. </body>
106. </html>

## deals.html

1. <!DOCTYPE html>
2. <html lang="en-gb">
3. <head>
4. <!--helps search engines define what the purpose of the site is-->
5. <meta name="description" content="catalogue and shopping website">
6. <meta name="keywords" content="shopping, search, catalogue">
7. <!--defines correct character set-->
8. <meta charset="utf-8">
9. <!--ensures site will render correctly on all platforms, primarily mobile-->
10. <meta name="viewport" content="width=device-width, initial-scale=1">
11. <link rel="stylesheet" href="../css/styles.css" />
12. <!--links to the Boostrap CDN-->
13. <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
14. <!-- jquery link -->
15. <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
16. <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>
17. <!-- bootstrap js link -->
18. <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>
19. <!-- indexeddb wrapper and main javascript file links -->
20. <script src="../js/main.js"></script>
21. <script src="../js/lib/indexeddb.js"></script>
22. <!--title of site-->
23. <title>Catalogue Website Shop</title>
24. </head>
25. <body>
26. <!--defines type of navbar and makes navbar dark and controls spacing of items-->
27. <nav class="navbar navbar-expand-sm bg-dark navbar-dark">
28. <!-- logo link to home page -->
29. <a class="navbar-brand" href="../home/HomePage.html">
30. <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">
31. </a>
32. <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"
33. aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
34. <span class="navbar-toggler-icon"></span>
35. </button>
36. <div class="collapse navbar-collapse" id="navbarNavDropdown">
37. <ul class="navbar-nav">
38. <!--creates search bar form-->
39. <li class="nav-item active">
40. <div id="search">
41. <!-- search form goes to search results page -->
42. <form class="form-inline" action="../Search/results.html">
43. <input class="form-control mr-sm-2" type="text" placeholder="Search">
44. <button class="btn btn-success" type="submit">Search</button>
45. </form>
46. </div>
47. </li>
48. <li class="nav-item">
49. <!--link to home page-->
50. <a class="nav-link" href="../home/HomePage.html">Home</a>
51. </li>
52. <li class="nav-item dropdown">
53. <!--creates dropdown-->
54. <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>
55. <!--name and type of dropdown-->
56. <div id="dropdown" class="dropdown-menu"></div>
57. </li>
58. <li class="nav-item">
59. <!--link to hot deals page-->
60. <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>
61. </li>
62. <li class="nav-item">
63. <!--link to store locator page-->
64. <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>
65. </li>
66. <li class="nav-item">
67. <!--link to contact page-->
68. <a class="nav-link" href="../Contact/contact.html">Contact</a>
69. </li>
70. <li class="nav-item">
71. <a class="nav-link" id="loginToggle">Login</a>
72. </li>
73. </ul>
74. </div>
75. </nav>
76. <!--creates login form-->
77. <div id="login" class="formContainer">
78. <div class="form-group">
79. <label for="email">Email</label>
80. <!--defines type of input, id, placeholder and requires user to make input-->
81. <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>
82. </div>
83. <div class="form-group">
84. <label for="key">Password</label>
85. <!--defines type of input, id, placeholder and requires user to make input-->
86. <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>
87. </div>
88. <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>
89. </form>
90. </div>
91. <div class="title"><h1>Hot Deals</h1></div>
93. <h2>Under Construction</h2>
94. </body>
95. </html>

## 2.6 crud.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- google map api and jquery links -->

    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB6QCGqF325m9QTLFst-qHTbE1lUaePKHY"></script>

    <script src="https://code.jquery.com/jquery-3.2.1.js"

        integrity="sha256-DZAnKJ/6XZ9si04Hgrsxu/8s717jcIzLy3oi35EouyE=" crossorigin="anonymous">

        </script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Admin Controls</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <div id="logo">

            <a class="navbar-brand" href="../home/HomePage.html">

                <img src="../images/Misc/trolley.png" id="logo" alt="Trolley Logo" width="50" height="50">

            </a>

        </div>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Admin Controls</h1>

    </div>

    <!-- all functions are in separate accordion divs to look cleaner -->

    <!-- form for add category func, links to AddCategory.js -->

    <button class="accordion">Add New Category</button>

    <div class="panel">

        <div id="AddCatFunc" class="container">

            <h2> Add Category </h2>

            <form id="formAddCat">

                <label> Category Name </label> <br />

                <input type="text" class="form-control text" id="txtCatName" name="catName" placeholder="Enter Category"

                    required="required"> <br />

                <label> Category Description </label> <br />

                <input type="text" class="form-control text" id="txtCatDesc" name="catDesc" placeholder="Description"

                    required="required">

                <input type="submit" class="btn btn-primary" id="btnAddCat" value="Add">

            </form>

        </div>

    </div>

    <!-- form for add subcategory func, links to AddSubcategory.js -->

    <button class="accordion">Add New Subcategory</button>

    <div class="panel">

        <div id="AddSubcatFunc" class="container">

            <h2> Add Subcategory </h2>

            <form id="formAddSubcat">

                <div>

                    <!--creates dropdown fed with categories from db-->

                    <select id="AdminDrop1" class="browser-default custom-select" required="required"></select>

                    <br />&nbsp;

                    <br />

                </div>

                <label> Subcategory Name </label> <br />

                <input type="text" class="form-control text" id="txtSubcatName" name="SubcatName"

                    placeholder="Enter Subcategory" required="required">

                <br />

                <label> Subcategory Description </label> <br />

                <input type="text" class="form-control text" id="txtSubcatDesc" name="SubcatDesc"

                    placeholder="Description" required="required">

                <input type="submit" class="btn btn-primary" id="btnAddSubcat" value="Add">

            </form>

        </div>

    </div>

    <!-- add new item, links to AddItem/DAO.js -->

    <button class="accordion">Add New Item</button>

    <div class="panel">

        <!-- Add item function -->

        <div id='FuncAdd' class="container">

            <h2> Add item </h2>

            <form id="formSellItem">

                <div>

                    <label> Item Name </label> <br />

                    <input type="text" class="form-control text" id="txtItemName" name="itemName"

                        placeholder="Enter Item" required="required"> <br />

                </div>

                <div>

                    <label> Description </label> <br />

                    <input type="text" class="form-control text" id="txtDesc" name="Desc" placeholder="Description"

                        required="required"> <br />

                </div>

                <div>

                    <label> Price </label> <br />

                    <input type="int" class="form-control text" id="intPrice" name="Price" placeholder="Price"

                        required="required"> <br />

                </div>

                <div>

                    <!--creates dropdown fed with categories from db-->

                    <select id="AdminDrop" class="browser-default custom-select"></select>

                    <br />&nbsp;

                    <br />

                </div>

                <!-- image upload -->

                <div>

                    <label for="file">Choose item image to upload</label>

                    <input type="file" class="form-control-file" id="fileItemImage" name="file"> <br />

                    <img src="" alt="Image preview..." id="bannerImg"> <br />

                </div>

                <input type="submit" class="btn btn-primary" id="btnSaveItem" value="Save">

            </form>

        </div>

    </div>

    <!-- input for new location, links to InsertEvent/DAO.js

    uses google map api -->

    <button class="accordion">Add New Location</button>

    <div class="panel">

        <div id="addLoc" class="container">

            <h2>Add New Location</h2>

            <form id="formInsertEvent">

                Store Name <br/> <br/>

                <input type="text" class="form-control text" id="txtEvent" placeholder="Enter Store"

                    required="required"> <br />

                Select new location in Map<br /> <br />

                <!--placeholder for Google Map, javascript loads map-->

                <div id="mapInput" style="width: 750px; height: 450px;"></div><br />

                <input type="submit" class="btn btn-primary" id="btnSubmit">

            </form>

        </div>

    </div>

    <div class="title">

        <h2>Update And Delete Controls</h2>

    </div>

    <!-- displays category list to delete/update -->

    <button class="accordion">Update & Delete Categories with Subcategories and Items</button>

    <div class="panel">

        <div id="divCatList" class="container">

            <h2>Category List</h2>

        </div>

    </div>

    <button class="accordion">Update & Delete Locations</button>

    <div class="panel">

        <!-- displays map marker list to delete/update -->

        <div id="markerList" class="container">

            <h2>Store Markers List</h2>

        </div>

        <script src="../Map/listLocations.js"></script>

    </div>

    <script src="../js/accordion.js"></script>

    <script src="../js/AddCat/AddCategory.js"></script>

    <script src="../js/AddSubcat/AddSubcategory.js"></script>

    <script src="../js/AddFunc/AddItem.js"></script>

    <script src="../js/AddFunc/AddItemDAO.js"></script>

    <script src="../js/ListCat/listCat.js"></script>

    <script src="../js/ListCat/initCompare.js"></script>

    <script src="../Map/InsertEvent.js"></script>

    <script src="../Map/InsertEventDAO.js"></script>

    <script>

        $("#AdminDrop").change(function (event) {

            //event.preventDefault();

            var value = $(this).val();

            console.log("selCat was clicked value:" + value)

            //return false;

        });

    </script>

</body>

</html>

## 2.7 watch.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Watchlist</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Watchlist</h1>

    </div>

    <div id="divEmpty">

        <h5>Your watchlist is empty</h5>

    </div>

    <div id="divItemList" class="container">

    </div>

    <script src="../js/ListFunc/ListItems.js"></script>

    <script src="../js/ListFunc/ListItemsDAO.js"></script>

</body>

</html>

## 2.8 EventList.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- google map api and jquery links -->

    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB6QCGqF325m9QTLFst-qHTbE1lUaePKHY"></script>

    <script src="https://code.jquery.com/jquery-3.2.1.js" integrity="sha256-DZAnKJ/6XZ9si04Hgrsxu/8s717jcIzLy3oi35EouyE=" crossorigin="anonymous"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../../js/main.js"></script>

    <script src="../../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Store Locator</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../../home/HomePage.html">

            <img src="../../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <!--submit button-->

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title"><h1>Find Your Nearest Store</h1></div>

    <!--placeholder for Google Map, javascript loads map-->

    <div id="mapView" style="width: 750px; height: 450px;"></div>

    <script src="./EventList.js"></script>

    <script src="./EventListDAO.js"></script>

</body>

</html>

## 2.9 results.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Under Construction</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <h1>Under Construction</h1>

</body>

</html>

## 2.10 AddCategory.js

$('#formAddCat').submit(function(event) { //function fires when form is submitted

    event.preventDefault();

    console.log("formAddCat submit:");

    //sets correct db and store of categories

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('categories');

    startDB(function() {

        saveCatData();

        alert("Category has been saved successfully");

        location.reload();

    });

});

var newCategoryID = 0;

//saves form data into categories table

function saveCatData() {

    var name = $('#txtCatName').val();

    var catDesc = $('#txtCatDesc').val();

    //format of table

    var data = {

        'name': name,

        'catDesc': catDesc

    };

    //saves new category in db

    insertOne(data, function(lastID) {

        event.preventDefault();

        console.log("saveCatData lastID:" + lastID);

        return false;

    });

}

## 2.11 AddItem.js

$(document).on('change', '#fileItemImage', function(event) {

    previewFile();

});

//func to view preview of image before submitting

function previewFile(){

    var preview = document.querySelector('img#bannerImg');

    var file = document.querySelector('input[type=file]').files[0];

    var reader = new FileReader();

    reader.onloadend = function () { //source of img preview

        preview.src = reader.result;

    }

    if (file) {

        reader.readAsDataURL(file); //show preview

    } else {

        preview.src = ""; //preview empty

    }

}

//data saved to db on form submit

$('#formSellItem').submit(function(event)  {

    console.log("AdminDrop: " + $('#AdminDrop').val());

    event.preventDefault();

    if ($('#AdminDrop').val() === '0'){ //guard against empty input

        alert("Please Select a Category");

        return false;

    }

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'events', 'watchlist']);

    setCurrObjectStoreName('items');

        startDB(function () {

        saveItemData();

        alert("Item has been saved successfully");

        location.reload();

        return true;

    });

});

## 2.12 AddItemDAO.js

//save value of form input into db

function saveItemData() {

    var itemName = $('#txtItemName').val();

    var itemDesc = $('#txtDesc').val();

    var itemPrice = $('#intPrice').val();

    var itemSubcategory = $('#AdminDrop').val();

    console.log("saveItemData itemSubcategory: "+ itemSubcategory);

    //format of db table

    var data = {

        'itemName': itemName,

        'itemDesc': itemDesc,

        'itemPrice': itemPrice,

        'itemSubcategory': itemSubcategory,

    };

    //image upload input

    var fileInput = $('#fileItemImage');

    var selectedFile = fileInput.get(0).files[0];

    if (typeof selectedFile != 'undefined')

        data.itemImage = selectedFile; //image stored in db

    //insert into db

    insertOne(data, function(lastID) {

        event.preventDefault();

        return false;

    });

}

## 2.13 AddSubcategory.js

$('#formAddSubcat').submit(function(event) { //function fires when form is submitted

    console.log("AdminDrop1: " + $('#AdminDrop1').val());

    event.preventDefault();

    if ($('#AdminDrop1').val() === '0'){ //requires input in dropdown

        alert("Please Select a Category");

        return false;

    }

    event.preventDefault();

    console.log("formAddSubat submit: ");

    //sets correct db and store of categories

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events']);

    setCurrObjectStoreName('subcategories');

    startDB(function() {

        saveSubcatData();

        alert("Subcategory has been saved successfully");

        location.reload();

    });

});

var newSubcategoryID = 0;

//saves form data into categories table

function saveSubcatData() {

    var subcatName = $('#txtSubcatName').val();

    var subcatDesc = $('#txtSubcatDesc').val();

    var category = $('#AdminDrop1').val();

    //format of table

    var data = {

        'subcatName': subcatName,

        'subcatDesc': subcatDesc,

        'category': category

    };

    //saves new category in db

    insertOne(data, function(lastID) {

        event.preventDefault();

        console.log("saveSubcatData lastID:" + lastID);

        return false;

    });

}

## 2.14 indexeddb.js

    //This is an indexeddb wrapper javascript library

    //Global variables

    var db, indexedDB, IDBTransaction, currObjectStoreName, databaseName, objectStores;

    //startDB creates connection with the databaseName

    //and create database and object stores

    function startDB(successCallback, failureCallback) {

        try {

            indexedDB = window.indexedDB || window.webkitIndexedDB || window.mozIndexedDB || window.msIndexedDB;

            IDBTransaction = window.IDBTransaction || window.webkitIDBTransaction || window.msIDBTransaction;

            IDBKeyRange = window.IDBKeyRange || window.webkitIDBKeyRange || window.msIDBKeyRange;

        }

        catch(e) {

            console.log('Initial IndexedDB error: ' + e);

            failureCallback();

        }

        if(!window.indexedDB) {

            failureCallback();

            return;

        }

        var request = indexedDB.open(databaseName, 2);

        //The onupgradeneeded property is triggered when a database

        //of a bigger version number than the existing stored database is loaded.

        request.onupgradeneeded = function(event) {

            console.log('onupgradeneeded method is invoked');

            db = event.target.result;

            for(i=0; i < objectStores.length; i++) {

                db.createObjectStore(objectStores[i], { keyPath: 'id', autoIncrement: true });

            }

        };

        request.onsuccess = function(event) {

            db = event.target.result;

            successCallback && successCallback();

        };

        request.onerror = function(event) {

            console.log('User don\'t allow IndexedDB to use offline storage.');

            failureCallback();

        };

    }

    //Just print any indexeddb related error message in console window

    function indexedDBError(event) {

        console.log('An error occurred in IndexedDB', event);

    }

    //setDatabaseName sets the Database name and Object Stores required for a website

    function setDatabaseName(dbName, objStores) {

        databaseName = dbName;

        objectStores = objStores;

        console.log('Database : ', dbName);

    }

    //setCurrObjectStoreName set the current object store to store or retrieve data

    function setCurrObjectStoreName(objStoreName) {

        currObjectStoreName = objStoreName;

        console.log('Current Object Store Name : ', currObjectStoreName);

    }

    //selectAll retrieves all data from the current object store

    function selectAll(successCallback) {

        var transaction = db.transaction([currObjectStoreName], IDBTransaction.READ\_ONLY || 'readonly'),

            objectStore, request, results = [];

        transaction.onerror = indexedDBError;

        objectStore = transaction.objectStore(currObjectStoreName);

        request = objectStore.openCursor();

        request.onerror = indexedDBError;

        request.onsuccess = function(event) {

            // event.target means request

            var cursor = event.target.result;

            if(!cursor) {

                if(successCallback) {

                    successCallback(results);

                }

                return;

            }

            results.push(cursor.value);

            cursor.continue();

        };

    }

    //insertOne inserts data into the current object store

    //This function also creates unique id for each data

    function insertOne(data, successCallback) {

        var transaction = db.transaction([currObjectStoreName], IDBTransaction.READ\_WRITE || 'readwrite'),

            objectStore, request, lastID;

        objectStore = transaction.objectStore(currObjectStoreName);

        request = objectStore.add(data);

        request.onsuccess = function(event) {

            lastID = event.target.result;

        }

        request.onerror = indexedDBError;

        transaction.onerror = indexedDBError;

        transaction.oncomplete = function(event) {

            console.log('Data was inserted succesfully');

            if(successCallback) {

                successCallback(lastID);

            }

        };

    }

    //deleteOne inserts data into the current object store

    function deleteOne(id, successCallback) {

        var transaction = db.transaction([currObjectStoreName], IDBTransaction.READ\_WRITE || 'readwrite'),

            objectStore, request;

        objectStore = transaction.objectStore(currObjectStoreName);

        request = objectStore.delete(id);

        request.onerror = indexedDBError;

        request.onsuccess = function(event) {

            var result = event.target.result;

        };

        transaction.onerror = indexedDBError;

        transaction.oncomplete = function() {

            console.log('Data with ' + id + ' was deleted successfully');

            if(successCallback) {

                successCallback();

            }

        };

    }

    //updateOne updates a specific data in the current object store

    function updateOne(data, successCallback) {

        var transaction = db.transaction([currObjectStoreName], IDBTransaction.READ\_WRITE || 'readwrite'),

            objectStore, request, lastID;

        objectStore = transaction.objectStore(currObjectStoreName);

        request = objectStore.put(data);

        request.onsuccess = function(event) {

            lastID = event.target.result;

        }

        request.onerror = indexedDBError;

        transaction.onerror = indexedDBError;

        transaction.oncomplete = function(event) {

            console.log('Data with ' + lastID + ' was updated successfully');

            if(successCallback) {

                successCallback(lastID);

            }

        };

    }

    //selectOne select just one data

    function selectOne(id, successCallback) {

        var transaction = db.transaction([currObjectStoreName], IDBTransaction.READ\_ONLY || 'readonly'),

            objectStore, request;

        transaction.onerror = indexedDBError;

        objectStore = transaction.objectStore(currObjectStoreName);

        request = objectStore.get(parseInt(id));

        request.onerror = indexedDBError;

        request.onsuccess = function(event) {

            // event.target means request

            var record = request.result;

            if(record) {

                if(successCallback) {

                    successCallback(record);

                }

                return;

            }

        };

    }

## 2.15 initCompare.js

//script to add max two items to compare list at bottom of page

var compareItemsList = [];

var maxItemsToCompare = 2;

$(document).ready(function () {

    initialisePage();

    $(document).on("change", "input[class='cellChkbox']", function () {

        //guard for if checkbox value is empty

        if ($(this).is(":checked")) {

            var chkBoxId = this.id;

            if (chkBoxId === undefined && chkBoxId === "") {

                alert("please select an item to compare");

                return false;

            }

            var idArray = chkBoxId.split("\_");

            var itemId = idArray[1];

            AddItemToCompare(itemId);

        } else {

            HideCompareDiv(); //hide compare div when not in use

        }

    });

    function AddItemToCompare(itemId) {

        var numberOfExistingItems = compareItemsList.length;

        // check if wanted item exists

        if (numberOfExistingItems > 0 && (compareItemsList.includes("itemId"))) {

            alert("This item is already being compared.");

            return;

        }

        if (numberOfExistingItems < maxItemsToCompare) {

            // add the item if items in compare are less than 2

            compareItemsList.push(itemId);

            //creates div for items to go in side by side

            var itemDivhtml = "<div class='leftCellCompare' id='cellCompare\_" + itemId + "'>" + $("#cellItem\_" + itemId).html();

            itemDivhtml += "<br />";

            //option to remove item

            itemDivhtml += "<input type='button' id='removeCompareItem\_" + itemId + "' class='removeCompareItem' value='Remove' /> "

            itemDivhtml += "</div>";

            $("#divCompareList").append(itemDivhtml); //defines which div to put compare items in

            ShowCompareDiv(); //show compare function now values are added

        }

        else {

            alert("Only " + maxItemsToCompare + " items can be compared."); //returns if max items are already in div

        }

    }

    //event to called remove func

    $(document).on("click", "input[class='removeCompareItem']", function () {

        var classId = this.id;

        console.log("removeCompareItem click classId: " + classId);

        var idArray = classId.split("\_");

        var itemId = idArray[1];

        RemoveItemToCompare(itemId);

    });

    //function to remove item from compare

    function RemoveItemToCompare(itemId) {

        compareItemsList.pop(itemId);

        $("#compareCheckBox\_" + itemId).prop("checked", false); //unchecks checkbox

        $("#cellCompare\_" + itemId).remove(); //removes item with matching id

        HideCompareDiv(); //hide compare div now not in use again

    }

});

//runs on page load

function initialisePage() {

    console.log("initialisePage called")

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('categories');

    startDB(function () { // async func

        getAllCategories(FormatCategoriesAndItemsAsHtml); //calls function from listCat.js to populate cat/subcat/item divs

    });

    HideCompareDiv();

}

//hide compare div when empty

function HideCompareDiv() {

    if (compareItemsList.length === 0) {

        $("#divCompare").hide();

    }

}

//show compare div when values added

function ShowCompareDiv() {

    $("#divCompare").show();

    $("#divCompareList").show();

}

## 2.16 listCat.js

//global vars to hold the categories, subcategories and items so that the html can be built once they have been retrieved from the db

var listOfItems = [];

var listOfCategories = [];

var listOfSubcategories = [];

//retrieves category list from categories table in db

function getAllCategories(callBack) {

    selectAll(function (results) {

        var len = results.length;

        var i;

        //loop continues for every value in db table

        for (i = 0; i < len; i++) {

            listOfCategories[i] = results[i];

        }

        setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

        setCurrObjectStoreName('subcategories');

        // need to get all the subcats and items before building the html

        startDB(function () {

            getAllSubcategories(callBack);

        }); // async func

    });

}

//retrieve all values from subcategories table in db

//items need to be obtained before html can be built

function getAllSubcategories(callBack) {

    selectAll(function (results) {

        var len = results.length;

        var i;

        //loop for each result

        for (i = 0; i < len; i++) {

            listOfSubcategories[i] = results[i];

        }

        console.log("getAllSubcategories.length:" + listOfSubcategories.length);

        setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

        setCurrObjectStoreName('items');

        startDB(function () {

            getAllItems(callBack);

        }); // async func

    });

}

//retrieves all values from items table in db

//html can't be built until values are obtained

function getAllItems(callBack) {

    selectAll(function (results) {

        var len = results.length;

        var i;

        for (i = 0; i < len; i++) {

            listOfItems[i] = results[i];

        }

        console.log("getAllItems listOfItems.length:" + listOfItems.length);

        //optional callBack - to build the html at the end of the call stack, if it is needed

        if (callBack !== undefined && callBack !== "") {

            callBack();

        }

    });

}

function populateAllCategoriesArray() { //func to get category values without invoking callbacks

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('categories');

    startDB(function () {

        getAllCategories();

    }); // async func

}

function populateAllSubcatsArray() { //get subcat values without callbacks

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('subcategories');

    startDB(function () {

        getAllSubcategories();

    }); // async func

}

function populateAllItemsArray() { //get item values without callbacks

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('items');

    startDB(function () {

        getAllItems();

    }); // async func

}

//pulls all previous functions together to build html

function FormatCategoriesAndItemsAsHtml() {

    var len = listOfCategories.length;

    //categories div

    var html = "";

    for (i = 0; i < len; i++) {

        var categoryId = listOfCategories[i].id;

        //html += "<button class='accordion'>" + listOfCategories[i].name + "</button>";

        html += "<div class='display'>"

        html += "<div id='" + categoryId + "'>";

        html += "   <h1>" + listOfCategories[i].name + "</h1>";

        html += "   <h6>" + listOfCategories[i].catDesc + "</h6>";

        html += "   <div class='adminView' id='adminCellCategory\_" + categoryId + "'>";

        html += "       <button type='button' class='btn btn-danger deleteAction'>Delete</button><br/>";

        html += "       <button type='button' class='btn btn-info updateAction'>Update</button><br/>";

        html += "   </div>";

        // build html

        var k = 0;

        var lenSubcategories = listOfSubcategories.length;

        //subcategories div loops for each result

        for (k = 0; k < lenSubcategories; k++) {

            if (parseInt(listOfSubcategories[k].category) === parseInt(listOfCategories[i].id)) {

                // iterate over subcats array

                var subcategoryId = listOfSubcategories[k].id;

                html += "   <div class='indent'>";

                html += "       <div id='" + subcategoryId + "'>";

                html += "       <h1>" + listOfSubcategories[k].subcatName + "</h1>";

                html += "       <h6>" + listOfSubcategories[k].subcatDesc + "</h6>";

                html += "       <div class='adminView' id='adminCellSubCategory\_" + subcategoryId + "'>";

                html += "           <button type='button' class='btn btn-danger deleteAction'>Delete</button><br/>";

                html += "           <button type='button' class='btn btn-info updateActionSubcat'>Update</button><br/>";

                html += "       </div>";

                var j = 0;

                var lenItems = listOfItems.length;

                //items div loops for each result

                for (j = 0; j < lenItems; j++) {

                    // iterate over items array

                    var itemSubcategoryId = listOfItems[j].itemSubcategory;

                    console.log("FormatCategoriesAndItems called");

                    if (parseInt(subcategoryId) === parseInt(itemSubcategoryId)) {

                        //build item div if subcategoryId matches itemSubcategoryId

                        html += generateItemHTML(listOfItems[j]);

                    }

                }

                html += "   </div>";

                html += "</div>";

                html += "</div>";

            }

        }

    }

    $("#divCatList").html(html); //div that html is built into

    $('.adminView').hide(); //hides admin functions while on shop page

    //if url includes name of admin page then admin functions appear

    //checkboxes and buy button hidden as not needed

    var url = window.location.href;

    var adminCheck = url.includes("crud");

    if (adminCheck === true) {

        $('.adminView').show();

        $('.checkbox').hide();

        $('.buy').hide();

    }

    else {

        return;

    }

    //func to delete all child elements of category

    $('.deleteAction').click(function () {

        getAllSubcategories();// cascade build all 3 arrays

        //gets array of categories

        var idArray = $(this).parent().attr('id').split("\_");

        var parentId = idArray[0];

        var id = parseInt(idArray[1]);

        var storename = "";

        if (parentId.indexOf("adminCellCategory") !== -1) { //deletes category and cascades to invoke subcat and item deletes

            storename = "categories";

            if (!confirm("Are you sure you want to delete this category?\nWARNING: ALL related subcategories AND their related items will also be deleted?")) {

                return;

            };

            // get all subcategories for this category and items and delete

            for (i = 0; i < listOfSubcategories.length; i++) {

                var subCatId = listOfSubcategories[i];

                if (subCatId.categoryid === id) {

                    deleteItemsBySubCategory(subCatId.id);

                    deleteById("subcategories", subCatId.id);

                }

            }

            deleteById(storename, id);

        }

        if (parentId.indexOf("adminCellSubCategory") !== -1) {  //check to delete subcategory

            storename = "subcategories";

            if (!confirm("Are you sure you want to delete this subcategory\nWARNING: ALL related items will also be deleted?")) {

                return;

            };

            deleteItemsBySubCategory(id);

            deleteById(storename, id);

        }

        if (parentId.indexOf("adminCellItem") !== -1) { //check to see which items are in subcategory to be deleted

            storename = "items";

            if (!confirm("Are you sure you want to delete this item?")) {

                return;

            };

            deleteById(storename, id);

        }

        return false;

    });

    function deleteItemsBySubCategory(subcategoryId) { //deletes subcategory along with child items

        for (i = 0; i < listOfItems.length; i++) {

            var item = listOfItems[i];

            if (item.subcategoryid === subcategoryId) {

                deleteById("items", item.id);

            }

        }

    }

    //func to delete by store name rather than setting db store each time

    function deleteById(storename, id) {

        console.log("deleteAction: storename: " + storename);

        setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

        setCurrObjectStoreName(storename);

        startDB(function () {

            deleteOne(id, function () {

                alert("deleted!");

                location.reload();

            });

        }); // async func

    }

    $('.updateAction').click(function () { //redirects to category update page with id

        populateAllCategoriesArray();

        var idArray = $(this).parent().attr('id').split("\_");

        var parentId = idArray[0];

        var updateCatID = getIdFromSplit($(this).parent().attr('id'));

        window.open("../Update/updateCat.html?catID=" + updateCatID, "\_self");

        return false;

    });

    $('.updateActionSubcat').click(function () { //redirects to subcat update page with id

        populateAllSubcatsArray();

        var idArray = $(this).parent().attr('id').split("\_");

        var parentId = idArray[0];

        var updateSubcatID = getIdFromSplit($(this).parent().attr('id'));

        window.open("../Update/updateSubcat.html?subcatID=" + updateSubcatID, "\_self");

        return false;

    });

    $('.updateActionItem').click(function () { //redirects to item update page with id

        populateAllItemsArray();

        var idArray = $(this).parent().attr('id').split("\_");

        var parentId = idArray[0];

        var updateItemID = getIdFromSplit($(this).parent().attr('id'));

        window.open("../Update/Update.html?itemID=" + updateItemID, "\_self");

        return false;

    });

    $(".buy").click(function () { //redirects to under construction page when buy button clicked

        window.open('../Search/results.html');

        return false;

    })

}

//builds div for item to be placed in under subcat

function generateItemHTML(item) {

    var itemId = item.id;

    var html = "<div class='indentItems'>";

    html += "       <div class='leftCell' id='cellItem\_" + itemId + "'>";

    html += "           <h3>" + item.itemName + "</h3><br/>";

    html += "           <img src='../images/" + item.itemImage.name + "' height='100' width='100'/><br/>";

    html += "           <label>" + item.itemDesc + "</label><br/>";

    html += "           <label><b>&pound;" + item.itemPrice + "</b></label><br/>";

    html += "       </div>";

    //admin controls, only available on crud page

    html += "       <div class='adminView' id='adminCellItem\_" + itemId + "'>";

    html += "           <button type='button' class='btn btn-danger deleteAction'>Delete</button><br/>";

    html += "           <button type='button' class='btn btn-info updateActionItem'>Update</button><br/>";

    html += "       </div>";

    //checkbox to add to compare div

    html += "       <div class='checkbox'><input class='cellChkbox' type='checkbox' name='compare' value='Add to Compare' id='compareCheckBox\_" + itemId + "'/><label for='compareCheckBox\_" + itemId + "'> Add to compare</label></div>";

    //checkbox to add to watchlist

    html += "       <div id='watchItem\_" + itemId + "' class='checkbox'><input class='watchChkbox' type='checkbox' name='watch' value='Add to Watchlist' id='watchCheckBox\_" + itemId + "'/><label for='watchCheckBox\_" + itemId + "'> Add to watchlist</label></div>";

    html += "       <div class='buyItem'><button id='buy' class='btn btn-success buy' value='Buy'>Buy</button></div>";

    html += "   </div>";

    return html;

}

//func to separate number and string values in id

function getIdFromSplit(element) {

    var idArray = element.split("\_");

    var first = idArray[0];

    var id = parseInt(idArray[1]);

    return id;

}

//watchlist

$(document).on("change", "input[class='watchChkbox']", function () {

    populateAllItemsArray();//grabs items array

    var idArray = $(this).parent().attr('id').split("\_");

    var parentId = idArray[0];

    var watchItemID = getIdFromSplit($(this).parent().attr('id'));

    if ($(this).prop("checked")) {

        if (parentId.indexOf("watchItem") !== -1) {

            console.log("on change: watchItemID: " + $('#watchItem\_'));

            console.log("AddtoWatchlist fired")

            AddtoWatchlist(watchItemID);

        }

    }

    return;

});

//add to watchlist function

function AddtoWatchlist(watchItemID) {

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('items');

    startDB(function () {

        selectOne(watchItemID, function (result) { //selects item with matching id

            console.log("AddtoWatchlist result: " + result.itemName);

            saveWatchlistData(result); //saves item data to entry in watchlist table

            alert("Item has been successfully saved to watchlist");

            location.reload();

        });

    });

}

var newWatchItemID = 0;

//saves id's data into watchlist table

function saveWatchlistData(itemData) {

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('watchlist');

    startDB(function () {

        //format of table

        var data = {

            'itemName': itemData.itemName,

            'itemDesc': itemData.itemDesc,

            'itemPrice': itemData.itemPrice,

            'itemSubcategory': itemData.itemSubcategory,

            'itemImage': itemData.itemImage

        };

        //saves new item in watchlist table

        insertOne(data, function (lastID) {

            event.preventDefault();

            console.log("saveWatchlistData lastID:" + lastID);

            return false;

        });

    });

}

## 2.17 ListItems.js

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

setCurrObjectStoreName('watchlist'); //pulls data from watchlist store

startDB(function() {

    showAllItems(); //calls func in ListItemsDAO.js

});

## 2.18 ListItemsDAO.js

function showAllItems() {

    selectAll(function (results) {

        var len = results.length;

        var html = '',

            i;

        for (i = 0; i < len; i++) { //loops for all values in store

            $('#divEmpty').hide();

            var item\_id = results[i].id; //gets item id

            //builds div and populates with corresponding data

            html += '<div id="watchItem">';

            html += '   <div id="' + item\_id + '">';

            html += '   <h3> Name: ' + results[i].itemName + '</h3>';

            var img\_id = "image-" + item\_id;

            html += '   <h5> Price: £' + results[i].itemPrice + '</h5>';

            html += '   <h5> Description: ' + results[i].itemDesc + '</h5>';

            html += '   <h5> Subcategory: ' + results[i].itemSubcategory + '</h5>';

            html += '   <img src="../images' + results[i].itemImage.name + '" id=' + img\_id + ' height="100" width="100"/><br/><br/>';

            html += "   <button type='button' class='btn btn-danger actionDelete'>Remove</button><br/>"; //option to remove from watchlist

            html += "   <div class='buyItem'><button id='buy' class='btn btn-success buy' value='Buy'>Buy</button></div>";

            html += '   </div>';

            html += '</div>';

        }

        $('#divItemList').html(html); //div js populates

        for (i = 0; i < len; i++) {

            var obj\_url = window.URL.createObjectURL(results[i].itemImage);

            var item\_id = results[i].id;

            var img\_id = "image-" + item\_id;

            $('#' + img\_id).attr('src', obj\_url);

        }

        //func to delete item with id matching parent of the button click

        $('.actionDelete').click(function () {

            var itemID = parseInt($(this).parent().attr('id'));

            deleteOne(itemID, function () {

                alert("Item " + itemID + " was deleted from your watchlist");

                location.reload();

            })

            return false;

        });

        $(".buy").click(function () { //redirects to under construction page when buy button clicked

            window.open('../Search/results.html');

        })

    });

}

## 2.19 updateCat.js

var urlParams = new URLSearchParams(window.location.search); //retrieves id in the window to select which entry to display

var catID = urlParams.get('catID');

$('#catID').html("Category ID: " + catID);

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

setCurrObjectStoreName('categories');

var data;

startDB(function () {

    selectOne(catID, function(result) { //selects entry with id matching url to display

        $('#txtCatName').val(result.name);

        $('#txtCatDesc').val(result.catDesc);

        data = result;

    })

});

//new values replace old when form submitted

$('#formUpdateCategory').submit(function(event)  {

    event.preventDefault();

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('categories');

    startDB(function () {

        updateCatData(data); //calls from DAO

        alert("Category has been updated successfully");

    });

});

## 2.20 updateCatDAO.js

function updateCatData(data) { //func gets input values and replaces old ones with them when called

    var catName = $('#txtCatName').val();

    var catDesc = $('#txtCatDesc').val();

    data.name = catName;

    data.catDesc = catDesc;

    updateOne(data, function(lastID) { //uses updateOne from indexeddb wrapper

        event.preventDefault();

        return false;

    });

}

## 2.21 UpdateItem.js

var urlParams = new URLSearchParams(window.location.search); //retrieves id in the window to select which entry to display

var itemID = urlParams.get('itemID');

$('#itemID').html("Item ID: " + itemID);

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

setCurrObjectStoreName('items');

var data;

startDB(function () { //selects entry with id matching url to display

    selectOne(itemID, function(result) {

        $('#txtItemName').val(result.itemName);

        $('#txtDesc').val(result.itemDesc);

        $('#intPrice').val(result.itemPrice);

        $('#AdminDrop').val(result.itemSubcategory);

        data = result;

    })

})

$(document).on('change', '#fileItemImage', function(event) { //once new image uploaded, displayed in img preview

    previewFile();

});

//func to display img in preview div

function previewFile(){

    var preview = document.querySelector('img#bannerImg');

    var file = document.querySelector('input[type=file]').files[0];

    var reader = new FileReader();

    reader.onloadend = function () {

        preview.src = reader.result;

    }

    if (file) {

        reader.readAsDataURL(file);

    } else {

        preview.src = "";

    }

}

//new values replace old when form submitted

$('#formUpdateItem').submit(function(event)  {

    event.preventDefault();

    if ($('#AdminDrop').val() === '0'){ //guard against empty input

        alert("Please Select a Category");

        return false;

    }

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('items');

    startDB(function () {

        updateItemData(data); //calls from DAO

        alert("Item has been updated successfully!");

    });

});

## 2.22 UpdateItemDAO.js

function updateItemData(data) { //func gets input values and replaces old ones with them when called

    var itemName = $('#txtItemName').val();

    var itemDesc = $('#txtDesc').val();

    var itemPrice = $('#intPrice').val();

    var itemSubcategory = $('#AdminDrop').val();

    //add new values in table

    data.itemName = itemName;

    data.itemDesc = itemDesc;

    data.itemPrice = itemPrice;

    data.itemSubcategory = itemSubcategory;

    var fileInput = $('#fileItemImage'); //file upload input

    var selectedFile = fileInput.get(0).files[0];

    if (typeof selectedFile != 'undefined')

        data.itemImage = selectedFile;

    updateOne(data, function(lastID) { //uses updateOne from indexeddb wrapper

        event.preventDefault();

        return false;

    });

}

## 2.23 updateMap.js

var urlParams = new URLSearchParams(window.location.search); //retrieves id in the window to select which entry to display

var markerID = urlParams.get('markerID');

$('#markerID').html("Marker ID: " + markerID);

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

setCurrObjectStoreName('events');

var data;

startDB(function () {

    selectOne(markerID, function(result) { //fills in form with current values

        $('#txtMarkerName').val(result.eventName);

        $('#txtLatitude').val(result.eventLat);

        $('#txtLongitude').val(result.eventLng);

        data = result;

    })

})

//new values replace old when form submitted

$('#formUpdateMarker').submit(function(event)  {

    event.preventDefault();

    //sets correct obj store

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('events');

    startDB(function () {

        updateMarkerData(data); //call func from DAO

        alert("Map has been updated successfully");

    });

});

## 2.24 updateMapDAO.js

function updateMarkerData(data) { //func gets input values and replaces old ones with them when called

    var markerName = $('#txtMarkerName').val();

    var markerLat = $('#txtLatitude').val();

    var markerLong = $('#txtLongitude').val();

    data.eventName = markerName;

    data.eventLat = markerLat;

    data.eventLng = markerLong;

    updateOne(data, function(lastID) { //uses updateOne from indexeddb wrapper

        event.preventDefault();

        return false;

    });

}

## 2.25 updateSubcat.js

var urlParams = new URLSearchParams(window.location.search); //retrieves id in the window to select which entry to display

var subcatID = urlParams.get('subcatID');

$('#subcatID').html("Subcategory ID: " + subcatID);

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

setCurrObjectStoreName('subcategories');

var data;

startDB(function () {

    selectOne(subcatID, function(result) { //fills in form with current values

        $('#txtSubcatName').val(result.subcatName);

        $('#txtSubcatDesc').val(result.subcatDesc);

        data = result;

    })

})

//new values replace old when form submitted

$('#formUpdateSubcategory').submit(function(event)  {

    event.preventDefault();

    //sets correct obj store

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('subcategories');

    startDB(function () {

        updateSubcatData(data); //call func from DAO

        alert("Subcategory has been updated successfully");

    });

});

## 2.26 updateSubcatDAO.js

function updateSubcatData(data) { //func gets input values and replaces old ones with them when called

    var subcatName = $('#txtSubcatName').val();

    var subcatDesc = $('#txtSubcatDesc').val();

    data.subcatName = subcatName;

    data.subcatDesc = subcatDesc;

    updateOne(data, function(lastID) { //uses updateOne from indexeddb wrapper

        event.preventDefault();

        return false;

    });

}

## 2.27 accordion.js

//accordion js to make divs open and close on click

var acc = document.getElementsByClassName("accordion");

    var i;

    for (i = 0; i < acc.length; i++) {

        acc[i].addEventListener("click", function () {

            this.classList.toggle("active");

            var panel = this.nextElementSibling;

            if (panel.style.display === "block") {

                panel.style.display = "none";

            } else {

                panel.style.display = "block";

            }

        });

    }

## 2.28 main.js

//Global variables

const loginAdminCookieName = "elephant";

const loginCookieDays = 1;

//build db tables

const events = [

];

const items = [

];

const users = [{

    username: "public@test.com",

    key: "321password",

    role: "public",

    isLoggedIn: false

},

{

    username: "admin@test.com",

    key: "password123",

    role: "administrator",

    isLoggedIn: false

}

];

const categories = [

];

const subcategories = [

]

const watchlist = [

]

var db, indexedDB, IDBTransaction, currObjectStoreName, databaseName, objectStores;

$(document).ready(function () {

    // once the document has finished loading run this lot

    // initialise the UI

    init();

    //displays page url in console

    var url = window.location.href;

    console.log("you are on the " + url + "page");

    listCategories(displayCategoriesMenu);

    if (url.indexOf("catTemplate") !== -1) {

        console.log("\*\*\*\*\*\*\*\*\*\*url: " + url);

        listCategories(displayCategories);

    }

    //functions to show categories from db in selections

    listDepartments(displayDepartments);

    listSubDepts(displaySubDepts);

    //login field drops down when clicked, off by default

    $("#loginToggle").click(function () {

        $("#login").toggle();

    });

    //login func fired when form submitted

    $("#btnSubmitLogin").click(function (event) {

        setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

        setCurrObjectStoreName('users');

        startDB(function () {

            getAllUsers();

        });

    });

    //func to validate login credentials

    function getAllUsers() {

        var isValidUser = false;

        var validUser;

        selectAll(function (results) {

            var len = results.length;

            var i;

            for (i = 0; i < len; i++) { // loop over the users

                let dbUser = results[i];

                isValidUser = ValidateUser(dbUser);

                if (isValidUser) {

                    validUser = dbUser;

                    break;

                }

            }

            if (!isValidUser) {

                alert("Please login with valid credentials.");

                return;

            }

            if (validUser.role === "administrator") {

                //redirect to admin page where add/edit/delete functions are found

                url = "../Login/crud.html"

            } else if (validUser.role === "public") {

                //redirect to public page where watchlist and account functions are found

                url = "../WatchList/watch.html"

            }

            window.location.href = url;

        });

    }

    //logic to check credentials are correct

    function ValidateUser(dbUser) {

        // get form data

        var redirectUrl = "";

        let formUsername = $("#email").val();

        let formKey = $("#key").val();

        if (formUsername === undefined || formUsername === "") {

            alert("Invalid credentials please try again un");

            return false;

        }

        if (formKey === undefined || formKey === "") {

            alert("Invalid credentials please try again pw");

            return false;

        }

        if (dbUser) {

            let username = dbUser.username;

            let key = dbUser.key; // this will be the hashed password

            //compare credentials

            if (formUsername === username && formKey === key) {

                return true;

            } else {

                console.log("no user match found.");

                return false;

            }

        } else {

            console.log("no user found");

            return false;

        }

    }

    //displays categories in nav dropdown when clicked

    $(".liCat a").click(function (e) {

        e.preventDefault();

        var txt = $(this).text();

        console.log("click event: " + txt);

        DisplayCategoryInDiv($(this).id);

    });

});

//gets categories from db to display in div

function DisplayCategoryInDiv(catId) {

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('categories');

    var data;

    startDB(function () {

        selectOne(catId, function (result) {

            data = result;

            $("#categoryDiv").html("category: " + data.name);

        })

    })

}

function DisplayError() {

    console.error("There was an error");

}

function init() {

    initDB();

    //hides login form on page load

    $("#login").hide();

}

function initDB() {

    //open database

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories', 'events', 'watchlist']);

    //checks user's browser for indexeddb support

    window.indexedDB = window.indexedDB || window.mozIndexedDB || window.webkitIndexedDB || window.msIndexedDB;

    //This line should only be needed if it is needed to support the object's constants for older browsers

    window.IDBTransaction = window.IDBTransaction || window.webkitIDBTransaction || window.msIDBTransaction || { READ\_WRITE: "readwrite" };

    window.IDBKeyRange = window.IDBKeyRange || window.webkitIDBKeyRange || window.msIDBKeyRange;

    //if browser has no support message will display in console

    if (!window.indexedDB) {

        console.log("Your browser doesn't support a stable version of IndexedDB. Several features will not be available.");

    }

    //connect to the db

    let request = window.indexedDB.open(databaseName, 2);

    // handle any connection error

    request.onerror = function (event) {

        console.log("request error: unknown 84");

    };

    //handle connection success

    request.onsuccess = function (event) {

        db = event.target.result;

        console.log("success: " + db);

        console.log('Database : ', databaseName);

    };

    // this only happens once to create the db is a new version update is required or the db is deleted manually

    request.onupgradeneeded = function (event) {

        db = event.target.result;

        console.log("onupgradeneeded: " + db);

        //each table is defined here, ready for values to be added

        let objStoreUsers = db.createObjectStore("users", { keyPath: "id", autoIncrement: true });

        let objStoreCategories = db.createObjectStore("categories", { keyPath: "id", autoIncrement: true });

        let objStoreSubcategories = db.createObjectStore("subcategories", { keyPath: "id", autoIncrement: true });

        let objStoreItems = db.createObjectStore("items", { keyPath: "id", autoIncrement: true });

        let objStoreEvents = db.createObjectStore("events", { keyPath: "id", autoIncrement: true });

        let objStoreWatchlist = db.createObjectStore("watchlist", { keyPath: "id", autoIncrement: true });

        //key values generated automatically.

        users.forEach(function (user) {

            objStoreUsers.add(user);

        });

        categories.forEach(function (category) {

            objStoreCategories.add(category);

        });

        subcategories.forEach(function (subcategory) {

            objStoreSubcategories.add(subcategory);

        });

        items.forEach(function (item) {

            objStoreItems.add(item);

        });

        events.forEach(function (event) {

            objStoreEvents.add(event);

        });

        watchlist.forEach(function (item) {

            objStoreWatchlist.add(item);

        });

        db.onerror = function (event) {

            // Generic error handler for all errors targeted at db's requests

            console.error("Database error: " + event.target.error + ", " + event.target.errorCode);

        };

    };

}

function listItems(callBack) {

    // new call from the page so need to get a connection to the DB

    var request = window.indexedDB.open("dbCat", 2);

    request.onerror = function (event) {

        alert("Unable to retrieve data from the database at this time, please try later.");

    };

    // connection was successful

    request.onsuccess = function (event) {

        let db1 = event.target.result;

        let storeName = "items";

        $("#items").text("");

        let tx = db1.transaction(storeName).objectStore(storeName);

        tx.openCursor().onsuccess = function (event) {

            var cursor = event.target.result;

            if (cursor) {

                //console.log(storeName + ":key id: " + cursor.key + " name:" + cursor.value.name + ", price:" + cursor.value.price);

                callBack(cursor.key, cursor.value.name, cursor.value.desc, cursor.value.price, cursor.value.title, cursor.value.image, cursor.value.category);

                cursor.continue();

            } else {

                // console.log("No more entries!");

            }

        };

    };

}

//callback to func to display cats in div

function listCategories(callBack) {

    // new call from the page so need to get a connection to db

    var request = window.indexedDB.open("dbCat", 2);

    request.onerror = function (event) {

        alert("Unable to retrieve data from the database at this time, please try later. 261");

        console.log("error 262")

    };

    //connection was successful

    request.onsuccess = function (event) {

        let db1 = event.target.result;

        console.log("success")

        let storeName = "categories";

        let tx = db1.transaction(storeName).objectStore(storeName);

        $("#categories").text("");

        tx.openCursor().onsuccess = function (event) {

            var cursor = event.target.result;

            if (cursor) {

                callBack(cursor.key, cursor.value.name, cursor.value.parentcategory);

                cursor.continue();

            } else {

                //console.log("No more entries");

            }

        };

    };

}

//callback to func to display cats in selection

function listDepartments(callBack) {

    // new call from the page so need to get a connection to the DB

    var request = window.indexedDB.open("dbCat", 2);

    request.onerror = function (event) {

        alert("Unable to retrieve data from the database at this time, please try later. 309");

        console.log("error 294")

    };

    // connection was successful

    request.onsuccess = function (event) {

        let db1 = event.target.result;

        console.log("success")

        let storeName = "subcategories";

        let tx = db1.transaction(storeName).objectStore(storeName);

        // initialise the depts dropdown

        $("#subcategories").text("");

        var option = "<option value='0'>Select Department Subcategory</option>";

        $("#AdminDrop").append(option);

        tx.openCursor().onsuccess = function (event) {

            var cursor = event.target.result;

            if (cursor) {

                callBack(cursor.key, cursor.value.subcatName, cursor.value.parentcategory);

                cursor.continue();

            } else {

                //console.log("No more entries");

            }

        };

    };

}

//callback to func to display subcats in selection

function listSubDepts(callBack) {

    // new call from the page so need to get a connection to the DB

    var request = window.indexedDB.open("dbCat", 2);

    request.onerror = function (event) {

        alert("Unable to retrieve data from the database at this time, please try later. 309");

        console.log("error 294")

    };

    // connection was successful

    request.onsuccess = function (event) {

        let db1 = event.target.result;

        console.log("success")

        let storeName = "categories";

        let tx = db1.transaction(storeName).objectStore(storeName);

        // initialise the depts dropdown

        $("#categories").text("");

        var option = "<option value='0'>Select Department Category</option>";

        $("#AdminDrop1").append(option);

        tx.openCursor().onsuccess = function (event) {

            var cursor = event.target.result;

            if (cursor) {

                callBack(cursor.key, cursor.value.name, cursor.value.parentcategory);

                cursor.continue();

            } else {

                //console.log("No more entries");

            }

        };

    };

}

//adds cat into dropdown div with link to specific cat

function displayCategoriesMenu(id, name, parentcategory) {

    var css = "";

    if (parentcategory > 0) {

        css = "indent";

    }

    var ahref = "<a class='dropdown-item " + css + "' href='../CatTemplate/catTemplate.html#" + id + "'>" + name + "</a>";

    var link = "<li>" + ahref + "</li>";

    $("#dropdown").append(link);

}

//displays cat in dropdown div

function displayCategories(id, name, parentcategory) {

    var css = "";

    if (parentcategory > 0) {

        css = "indent";

    }

    var ahref = "<a class='liCat dropdown-item " + css + "' href='#'>" + name + "</a>";

    var link = "<li>" + ahref + "</li>";

    $("#categoryPageCategoriesList").append(link);

}

//adds cat to option selection

function displayDepartments(id, name, parentcategory) {

    var css = "";

    if (parentcategory > 0) {

        css = "indent";

    }

    var option = "<option class='" + css + "' value='" + id + "'>" + name + "</option>";

    $("#AdminDrop").append(option);

}

//adds subcat to subcat option selection

function displaySubDepts(id, name, parentcategory) {

    var css = "";

    if (parentcategory > 0) {

        css = "indent";

    }

    var option = "<option class='" + css + "' value='" + id + "'>" + name + "</option>";

    $("#AdminDrop1").append(option);

}

//cookie function - not used

function IsAdminLoggedIn() {

    //check cookie and return t/f

    let adminCookie = true;

    if (adminCookie === true) {

        $("#btnLogin").val("Log out");

    }

    return adminCookie; // test default to true

}

## 2.29 EventList.js

//mapCenter located in Cheltenham

var mapCenter = new google.maps.LatLng(51.8979988098144, -2.0838599205017);

//geocoder converts coords to readable address

var geocoder = new google.maps.Geocoder();

//InfoWindow displays content in a popup above map

var InfoWindow = new google.maps.InfoWindow();

var InfoWindowUpdate = new google.maps.InfoWindow();

var myMap;

var myMapUpdate;

function initialise() {

    //initial map properties

    var mapOptions = {

        zoom: 15,

        center: mapCenter

    };

    //creates map in div placeholder

    myMap = new google.maps.Map(document.getElementById("mapView"), mapOptions);

    console.log("initialise called");

    //myMapUpdate = new google.maps.Map(document.getElementById("mapUpdate"), mapOptions);

    //retrieve saved event list from indexeddb dbCat

    //$('#formInsertEvent').submit(function () {

        //event.preventDefault();

        setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

        setCurrObjectStoreName('events');

        startDB(function () {

            //call DAO showAllEvents() function to show save event data in map

           showAllEvents();

        })

   // })

}

//addDomListener triggered when html page loaded

//executes initialise function

google.maps.event.addDomListener(window, 'load', initialise);

## 2.30 EventListDAO.js

function showAllEvents() {

    //call indexeddb function to fetch all data from events obj store

    selectAll(function(results) {

        console.log(results.length);

        console.log(results.length);

        console.log(results.length);

        //iterating all the data

        for(i=0; i<results.length; i++) {

            //name, latitude and longitude of specific event

            var eventName = results[i].eventName;

            var eventLat = results[i].eventLat;

            var eventLng = results[i].eventLng;

            //create latlng obj which will passed to create marker

            var markerLatLng = new google.maps.LatLng(eventLat, eventLng);

            //create map marker

            marker = new google.maps.Marker({

                map: myMap,

                position: markerLatLng

            });

            // markerUpdate = new google.maps.Marker({

            //     map: myMapUpdate,

            //     position: markerLatLng

            // });

            var infowindow = new google.maps.InfoWindow();

            //set event name as content of the info window

            infowindow.setContent(eventName);

            infowindow.open(myMap, marker);

            // var infowindowUpdate = new google.maps.InfoWindow();

            // //set event name as content of the info window

            // infowindowUpdate.setContent(eventName);

            // infowindowUpdate.open(myMapUpdate, markerUpdate);

        }

    });

}

## 2.31 InsertEvent.js

//mapCenter located in Cheltenham

var mapCenter = new google.maps.LatLng(51.8979988098144, -2.0838599205017);

//geocoder converts coords to readable address

var geocoder = new google.maps.Geocoder();

//InfoWindow displays content in a popup above map

var InfoWindow = new google.maps.InfoWindow();

function initialise() {

    //initial map properties

    var mapOptions = {

        zoom: 15,

        center: mapCenter

    };

    //creates map in div placeholder

    myMap = new google.maps.Map(document.getElementById('mapInput'), mapOptions);

    //create draggable marker icon in map

    marker = new google.maps.Marker({

        map: myMap,

        position: mapCenter,

        draggable: true

    });

}

//addDomListener triggered when html page loaded

//executes initialise function

google.maps.event.addDomListener(window, 'load', initialise);

//event handler for html form submit

//calls insertEvent function given in the DAO js

//stores data into indexeddb

$('#formInsertEvent').submit(function() {

    event.preventDefault();

    setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('events');

    startDB(function() {

        insertEvent(marker);

        alert("New location successfully added");

        location.reload();

    });

});

## 2.32 InsertEventDAO.js

function insertEvent(marker) {

    //get event data user input

    var eventName = $('#txtEvent').val();

    //get lat and long of current marker position

    var eventLat = marker.getPosition().lat();

    var eventLng = marker.getPosition().lng();

    //create object combining inputs

    var data = {

        'eventName' : eventName,

        'eventLat' : eventLat,

        'eventLng' : eventLng

    }

    //insert data into indexeddb dbCat

    insertOne(data, function() {

        return false;

    })

}

## 2.33 listLocations.js

setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

    setCurrObjectStoreName('events');

    startDB(function() {

        showAllMarkers();

    });

function showAllMarkers() {

    selectAll(function (results) {

        var len = results.length;

        var html = '',

            i;

        //new div is created for each item in the events table in loop

        for (i = 0; i < len; i++) {

            var marker\_id = results[i].id;

            html += "<div id='" + marker\_id + "'>";

            html += '<h3>' + results[i].eventName + "</h3>";

            html += "<h5>" + results[i].eventLat + "</h5>";

            html += "<h5>" + results[i].eventLng + "</h5>";

            html += "<button type='button' class='btn btn-danger mapActionDelete'>Delete</button><br/>"; //delete function

            html += "<button type='button' class='btn btn-info mapActionUpdate'>Update</button><br/><br/>"; //update function

            html += "</div>";

        }

        $('#markerList').html(html); //div that the newly created divs will go into

        //function to delete entry in table

        $('.mapActionDelete').click(function () {

            var markerID = parseInt($(this).parent().attr('id'));

            console.log("markerID: " + markerID)

            setDatabaseName('dbCat', ['users', 'items', 'categories', 'subcategories ', 'events', 'watchlist']);

            setCurrObjectStoreName('events');

            startDB(function() {

                deleteOne(markerID, function () {

                    alert("Store marker " + markerID + " was deleted successfully");

                    location.reload();

                })

                return false;

            });

        });

        //function to update entry

        $('.mapActionUpdate').click(function () {

            var markerID = parseInt($(this).parent().attr('id'));

            window.open("../Update/updateMap.html?markerID=" + markerID, "\_self");

            return false;

        });

    });

}

## 2.34 Update.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Update Item</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" id="logo" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Update Item </h1>

    </div>

    <!-- update form -->

    <div id="UpdateFunc" class="container">

        <form id="formUpdateItem">

            <h3 id="itemID">Item ID</h3>

            <label> Item Name </label> <br>

            <input type="text" class="form-control text" id="txtItemName" name="itemName" placeholder="Enter Item" required="required"> <br/>

            <label> Description </label> <br>

            <input type="text" class="form-control text" id="txtDesc" name="Desc" placeholder="Description" required="required"> <br/>

            <label> Price </label> <br>

            <input type="int" class="form-control text" id="intPrice" name="Price" placeholder="Price" required="required"> <br/>

            <select id="AdminDrop" class="browser-default custom-select" required="required"></select>

            <br />&nbsp;

            <br />

            <label for="file">Choose item image to upload</label>

            <input type="file" class="form-control-file" id="fileItemImage" name="file"> <br> <br>

            <img src="" alt="Image preview..." id="bannerImg"> <br> <br>

            <input type="submit" class="btn btn-primary" id="btnUpdateItem" value="Update">

        </form>

    </div>

    <script src="../js/UpdateFunc/UpdateItem.js"></script>

    <script src="../js/UpdateFunc/UpdateItemDAO.js"></script>

</body>

</html>

## 2.35 updateCat.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Update Category</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Update Category</h1>

    </div>

    <!-- update category form -->

    <div id="UpdateCatFunc" class="container">

        <form id="formUpdateCategory">

            <h3 id="catID">Category ID</h3>

            <label> Category Name </label> <br />

            <input type="text" class="form-control text" id="txtCatName" name="catName" placeholder="Enter Category" required="required"> <br /> <br />

            <label> Description </label> <br />

            <input type="text" class="form-control text" id="txtCatDesc" name="catDesc" placeholder="Description" required="required"> <br /> <br />

            <input type="submit" class="btn btn-primary" id="btnUpdateCategory" value="Update">

        </form>

    </div>

    <script src="../js/UpdateFunc/updateCat.js"></script>

    <script src="../js/UpdateFunc/updateCatDAO.js"></script>

</body>

</html>

## 2.36 updateMap.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- google map api and jquery links -->

    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB6QCGqF325m9QTLFst-qHTbE1lUaePKHY"></script>

    <script src="https://code.jquery.com/jquery-3.2.1.js"

        integrity="sha256-DZAnKJ/6XZ9si04Hgrsxu/8s717jcIzLy3oi35EouyE=" crossorigin="anonymous">

        </script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Update Map Marker</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Update Map</h1>

    </div>

    <!-- update map form -->

    <div id="UpdateMarkerFunc" class="container">

        <form id="formUpdateMarker">

            <h3 id="markerID">Store ID</h3>

            <label> Store Name </label> <br />

            <input type="text" class="form-control text" id="txtMarkerName" name="markerName" placeholder="Enter Store" required="required"> <br/>

            <label> Latitude </label> <br />

            <input type="text" class="form-control text" id="txtLatitude" name="markerLat" placeholder="Latitude" required="required"> <br/>

            <label> Longitude </label> <br />

            <input type="text" class="form-control text" id="txtLongitude" name="markerLong" placeholder="Longitude" required="required"> <br/>

            <input type="submit" class="btn btn-primary" id="btnUpdateMarker" value="Update" style="display: inline-block;">

        </form>

        <div id="mapView" style="width: 750px; height: 450px; display: inline-block;"><br/>

        </div>

    </div>

    <script src="../Map/EventList/EventList.js"></script>

    <script src="../Map/EventList/EventListDAO.js"></script>

    <script src="../js/UpdateFunc/updateMap.js"></script>

    <script src="../js/UpdateFunc/updateMapDAO.js"></script>

</body>

</html>

## 2.37 updateSubcat.html

<!DOCTYPE html>

<html lang="en-gb">

<head>

    <!--helps search engines define what the purpose of the site is-->

    <meta name="description" content="catalogue and shopping website">

    <meta name="keywords" content="shopping, search, catalogue">

    <!--defines correct character set-->

    <meta charset="utf-8">

    <!--ensures site will render correctly on all platforms, primarily mobile-->

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet" href="../css/styles.css" />

    <!--links to the Boostrap CDN-->

    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">

    <!-- jquery link -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

    <!-- bootstrap js link -->

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

    <!-- indexeddb wrapper and main javascript file links -->

    <script src="../js/main.js"></script>

    <script src="../js/lib/indexeddb.js"></script>

    <!--title of site-->

    <title>Catalogue Website Update Category</title>

</head>

<body>

    <!--defines type of navbar and makes navbar dark and controls spacing of items-->

    <nav class="navbar navbar-expand-sm bg-dark navbar-dark">

        <!-- logo link to home page -->

        <a class="navbar-brand" href="../home/HomePage.html">

            <img src="../images/Misc/trolley.png" alt="Trolley Logo" width="50" height="50">

        </a>

        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown"

            aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">

            <span class="navbar-toggler-icon"></span>

        </button>

        <div class="collapse navbar-collapse" id="navbarNavDropdown">

            <ul class="navbar-nav">

                <!--creates search bar form-->

                <li class="nav-item active">

                    <div id="search">

                        <!-- search form goes to search results page -->

                        <form class="form-inline" action="../Search/results.html">

                            <input class="form-control mr-sm-2" type="text" placeholder="Search">

                            <button class="btn btn-success" type="submit">Search</button>

                        </form>

                    </div>

                </li>

                <li class="nav-item">

                    <!--link to home page-->

                    <a class="nav-link" href="../home/HomePage.html">Home</a>

                </li>

                <li class="nav-item dropdown">

                    <!--creates dropdown-->

                    <a class="nav-link dropdown-toggle" href="#" id="navbardrop" data-toggle="dropdown">Categories</a>

                    <!--name and type of dropdown-->

                    <div id="dropdown" class="dropdown-menu"></div>

                </li>

                <li class="nav-item">

                    <!--link to hot deals page-->

                    <a class="nav-link" href="../Hot Deals/deals.html">Hot Deals</a>

                </li>

                <li class="nav-item">

                    <!--link to store locator page-->

                    <a class="nav-link" href="../Map/EventList/EventList.html">Store Locator</a>

                </li>

                <li class="nav-item">

                    <!--link to contact page-->

                    <a class="nav-link" href="../Contact/contact.html">Contact</a>

                </li>

                <li class="nav-item">

                    <a class="nav-link" id="loginToggle">Login</a>

                </li>

            </ul>

        </div>

    </nav>

    <!--creates login form-->

    <div id="login" class="formContainer">

        <div class="form-group">

            <label for="email">Email</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="email" class="form-control" id="email" placeholder="Enter email" name="email" required>

        </div>

        <div class="form-group">

            <label for="key">Password</label>

            <!--defines type of input, id, placeholder and requires user to make input-->

            <input type="password" class="form-control" id="key" placeholder="Enter password" name="key" required>

        </div>

        <button type="submit" id="btnSubmitLogin" class="btn btn-primary">Submit</button>

        </form>

    </div>

    <div class="title">

        <h1>Update Subcategory</h1>

    </div>

    <!-- update subcat form -->

    <div id="UpdateSubcatFunc" class="container">

        <form id="formUpdateSubcategory">

            <h3 id="subcatID">Subcategory ID </h3>

            <label> Subcategory Name </label> <br />

            <input type="text" class="form-control text" id="txtSubcatName" name="subcatName" placeholder="Enter Subcategory" required="required"> <br />

            <label> Description </label> <br />

            <input type="text" class="form-control text" id="txtSubcatDesc" name="subcatDesc" placeholder="Description" required="required"> <br />

            <input type="submit" class="btn btn-primary" id="btnUpdateSubcategory" value="Update">

        </form>

    </div>

    <script src="../js/UpdateFunc/updateSubcat.js"></script>

    <script src="../js/UpdateFunc/updateSubcatDAO.js"></script>

</body>

</html>