Website Report

Harry Bridgen

University of Nottingham

COMP1004 Databases and Interfaces

22nd March 2023

Contents

[1 Introduction 1](#_Toc130672865)

[2 Design 1](#_Toc130672866)

[3 Development 3](#_Toc130672867)

[3.1 HTML 3](#_Toc130672868)

[3.2 CSS 5](#_Toc130672869)

[3.3 JavaScript 8](#_Toc130672870)

[3.4 PHP 11](#_Toc130672871)

# 1 Introduction

The purpose of this project was to design and develop an interactive website that facilitates front-end communication with a back-end database. The project comprises of two parts: Coursework 1, which involved developing a one-to-many MySQL database with actor and movie tables and Coursework 2, which involved building a web interface to interact with the database, allowing users to view, add and remove actors and movies. This report will explain the features used in HTML, CSS, JavaScript, PHP and MySQL to build the website.

# 2 Design

The design process was started by creating two simple wireframes that show the functionality and look of the website. Four primary features were incorporated into the designs: 1) A suggestion box that shows movies or actors suggestions based on user input, with the ability to click on one of the suggestions to instantly input it into the search box. 2) An information box that displays information about actors or movies. 3) An upload button that allows the user to upload images of actors and movies. 4) The ability to add and remove movies and actors through forms.

Diagram

Description automatically generated

Figure 1 – “Movies” Wireframe

A picture containing graphical user interface

Description automatically generated Figure 2 - "Edit Movies" Wireframe

# 3 Development

## 3.1 HTML

I used a variety of HTML features in my website including:

<!DOCTYPE html> A declaration that defines the document type.

<html> The root element of a HTML page.

<head> The head element, which contains metadata about the page, such as title, scrips, and styles.

<title> The title element defines the title shown in the browsers tab.

<script> The script element which is used to include JavaScript code.

<link> The linked element which is used to include stylesheets such as CSS and favicons. It is also used for setting the tab icons in the browser.

 Figure 3 - "Movies" HTML Head

<body> The body element, which is used to define the main content of the HTML page.

<div> The div element, which is used to define a container within the HTML page.

<ul> The unordered list element, which represents an unordered list of items.

<li> The list item element, which is used to define an element within a list.

<a> The anchor element, which is used to create hyperlinks to other pages and or resources.

<span> The span element is a generic container which does not inherently represent anything. In this case, it represents the navigation bar favicons.

<header> The header element represents introductory content such as titles for pages.

<h1> The h1 element which is used for styling and defining headers.

<section> The section element is used to group related content.

<form> The form element is used to create a form which allows the user to input data, then submit the form. Forms can have various attributes, such as “onsubmit” which defines what happens when the user submits the form.

<label> The label element is used to make a label for another element.

<input> The input element is used to create various input controls, such as text fields, checkboxes, buttons, and dates. These input elements are used within forms. Input elements can have attributes, such as “onkeyup” which defines what happens when the user after the user has typed something or “onblur” which defines what happens when the user clicks away from the element.

<br> The break element is used to insert a blank line in the HTML page.

 Figure 4 - "Movies" HTML Body

## 3.2 CSS

To style the HTML, I used the following CSS features:

‘@import’: Used to import CSS files or fonts. In this case, importing the Roboto font.

 Figure 5 - Import Roboto

Selectors: For targeting certain elements. For example:

 Figure 6 - Selector Examples

Fonts: Setting the font of an element. For example, setting the HTML element ‘body’ to ‘Roboto’, which is an imported font. The fonts after Roboto are in case Roboto fails to load.

 Figure 7 - Font Family Example

Colours: Setting the colour of text, elements and background colours using HEX codes. For example, setting the background colour of ‘body’ to slightly grey (#cfcfcf).

 Figure 8 - Background Colour Example

Width and Height: Setting the width and hight of elements.

 Figure 9 - Max Width Example

Margins and Padding: Setting margins and padding controls the spacing around and in elements. This gives the HTML structure. For example, using the margin to auto will centre the container and the padding to 1rm will make a 16-pixel spacing around the edge of the container, to ensure anything within the container isn’t too close to the edge.

 Figure 10 - Margin and Padding Example

Borders: Applying borders to elements with sizes, border types and colours.

 Figure 11 - Border Example

Text Properties: Styling text, such as ‘font-size’ and ‘text-align’.

 Figure 12 - Font Size Example

Pseudo-classes: Applying styles on user interactions. For example, ‘:hover’ on an attribute selector, which colours the background a certain colour.

 Figure 13- Hover Example

Box Properties: Setting ‘box-shadow’ to add a shadow effect around boxes.

 Figure 14 - Box Shadow Example

Positioning: Positioning elements using the ‘position’ property with values such as ‘relative’ and ‘absolute’.

 Figure 15 - Positioning Example

Transitions: Adding transitions between style changes with the ‘transition’ property. For example, when hovering over a submit button, transition the colour to a darker shade of blue.

 Figure 16 - Background Colour Transition Example

## 3.3 JavaScript

To make the website interactive and communicate with the backend database, the following features were used:

Functions: Reusable blocks of code that can be called using parameters to perform a specific task. Functions are defined using the ‘function’ followed by open and closed brackets that contain the parameters.

 Figure 17 - Function Declaration Example

Document Object Model (DOM): Used for accessing and modifying HTML elements and properties. The use of DOM is appropriate for the toggle\_form.js script since ‘DOMContentLoaded’ calls when the HTML document has been completely loaded. This prevents potential issues that could happens if the script tries access elements before they are loaded to the page.

 Figure 18 - 'DOMContentLoaded' Example

EventListeners: Event listeners can be used for overriding the default behaviour of HTML form submissions, or for managing events such as ‘click’ or ‘DOMContentLoaded’.

 Figure 19 – Click and DOM Event Listener Examples

XMLHttpRequest: Asynchronous HTML requests using the AJAX object ‘XMLHttpRequest’. AJAX allows JavaScript to send a database request to the server, which will read the PHP and send the data back to the frontend, as opposed to a synchronous request, in which the frontend directly communicates with the database with PHP. It is appropriate to use AJAX because the data should be visible on the HTML page without having to reloading the page.

 Figure 20 - XMLHttpRequest Example

Timers: Timers delay the execution of code, using the ‘setTimeout’ function. This is used on the website for making a popup window that disappears after a certain amount of time (1500 milliseconds or 1.5 seconds).

 Figure 21 - setTimeout Example

FormData: Managing data from HTML form fields using the ‘FormData’ object. This allows for easy extraction of data and sending of data through AJAX.

 Figure 22 - FormData Example

PHP Query Parameters: Sending query parameters when calling PHP scripts using GET requests. This allows for the passing of data into PHP from JavaScript.

 Figure 23 - PHP GET Parameters Example

## 3.4 PHP

MySQLi: This allows for the connection and interaction with MySQL databases.

 Figure 24 - MySQLi Connection Example

SQL Statements: SQL Statements are used to access, add and delete specific data within a MySQL database. Naturally, the SQL statements used throughout the PHP files are assigned to the variable to ‘$sql’, for use in PHP scripts.

 Figure 25 - SQL Statement Example

Prepared Statements: Prepared statements are used throughout the website as it is used to prevent MySQL injection attacks. This increases the security and professionalism of the website.



Figure 26 - SQL Prepared Statement and Execution Example

‘$\_POST’ and ‘$\_GET’: HTTP request methods that are used to retrieve or send data to the server. In this case, they are used to pass values into the PHP script.

 Figure 27 - '$\_GET' Example

File Uploads: File uploads are used to add pictures of actors or movies. This is achieved by first checking if the movie or actor already has a in the respective image column in the database. If they don’t, then the HTML shows an upload button. The upload button is a form that allows a user to choose a file to represent the movie or actor. Then, the PHP file ‘upload\_image.php’ checks if the file is under 2MB, and is a JPG, JPEG or PNG file type. The name of the movie or actor has it’s spaces replaced with underscores, and becomes the new name for the file. For example, Brad Pitt becomes Brad\_Pitt.jpg. If the file is successfully moved to the appropriate file location on the server, the file name will be added to the respective image column in the database. This identifies an image with a movie or actor.

 Figure 28 - Image Upload Example