|  |
| --- |
|  |
| Project  Web Programming |
| |  |  |  | | --- | --- | --- | | Ben Royans [P205225] | 3/11/20 | Web Programming | |

TABLE OF CONTENTS

[Functional Design and Architectural Requirements 1](#_Toc36129239)

[Requirements 1](#_Toc36129240)

[Functionality 1](#_Toc36129241)

[Design 2](#_Toc36129242)

[Development Environments 2](#_Toc36129243)

[Structure 2](#_Toc36129244)

[Development Tools 2](#_Toc36129245)

[Run-time Environments 2](#_Toc36129246)

[Server Side 2](#_Toc36129247)

[Client Side 2](#_Toc36129248)

[Database Design 3](#_Toc36129249)

[Structure 3](#_Toc36129250)

[Validation 3](#_Toc36129251)

[Design Prototype 4](#_Toc36129252)

[Genres 4](#_Toc36129253)

[Movie Search 4](#_Toc36129254)

[Popular Movie Charts 5](#_Toc36129255)

[Movie Collections 5](#_Toc36129256)

[Testing 6](#_Toc36129257)

[PEAR Code Standard Validation Screenshots 6](#_Toc36129258)

[Testing 12](#_Toc36129259)

[Validate Consistency 18](#_Toc36129260)

[Browser Tests 19](#_Toc36129261)

[Google Chrome 19](#_Toc36129262)

[Internet Explorer 19](#_Toc36129263)

[Mozilla Firefox 19](#_Toc36129264)

[References 20](#_Toc36129265)

# Functional Design and Architectural Requirements

## Requirements

The website requires a connection to a database with a front-end GUI for user access. The website has the following requirements:

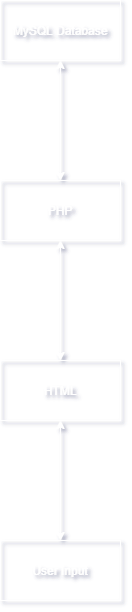
1. Connect to a MySQL database
   1. Handle roughly 2300 movie records.
   2. Generate SQL queries.
2. Search Form
   1. Title
   2. Genre
   3. Year
   4. Rating
3. Movie Details page
   1. Containing all database information on selected movie.
4. Have a chart displaying the top 10 most popular movies.
5. Use separate web modules. E.g. Header.

## Functionality

The site sources the information from the database tables, and populates the fields on the website. The database can also have records added to it through the website’s interface using MySQL.

The following basic flowchart demonstrates the flow of data in this project.

6



# Design

## Development Environments

### Structure

The basis of this webpage will be in HTML. This HTML can be generated by PHP or JavaScript scripting, which may be influenced by the results of MySQL queries. The HTML is also styled by Cascading Style Sheets (CSS).

### Development Tools

The source for these files will be developed in VS Code as it is a versatile web programming tool that accommodates these file types.

The PHP will be served locally by USBWebServer v8.0 for development and PHPMyAdmin will be used to handle database records. The movie database itself shall be imported through a provided .sql file into PHPMyAdmin.

## Run-time Environments

### Server Side

The server side environment for this website will be primarily PHP, served by USBWebServer. PHP will handle the output of HTML code and also the querying of databases and their tables in MySQL.

### Client Side

The client side of the application be served using HTML. The HTML will be exist in a structural form, and will be fleshed out with information using PHP through MySQL queries. The pages will also be augmented with JavaScript functionalities such as fetching images.

## Database Design

### Structure

The database ‘movies’ will contain two tables. The first table ‘moviesdb’ will contain the movie records containing the following items:

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Field Type** | **Length** | **Example Data** |
| ID | INT | 5 | 793 |
| Title | VARCHAR | 100 | Kill Bill: Vol. 1 |
| Studio | VARCHAR | 50 | Universal |
| Status | VARCHAR | 20 | Discontinued |
| Sound | VARCHAR | 20 | DTS 5.1 |
| Versions | VARCHAR | 50 | 4:03 |
| Recommended Retail Price | DECIMAL | 5,2 | 29.98 |
| Rating | VARCHAR | 5 | PG |
| Year | INT | 4 | 1999 |
| Genre | VARCHAR | 50 | Action |
| Aspect | VARCHAR | 50 | 2.35:1 |

The second table ‘mylist’ will contain movies selected from the previous table with the following information:

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Field Type** | **Length** | **Example Data** |
| ID | INT | 5 | 793 |
| Title | VARCHAR | 100 | Kill Bill: Vol. 1 |

### Validation

The database has been created as per the design requirements, and tested to be working correctly. To implement the popular movies requirement and add some extra functionality 3 new columns were add to the table. These new columns are:

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Field Type** | **Length** | **Example Data** |
| Search Count / Popularity | INT | 5 | 92 |
| Added to List Boolean | INT | 1 | 0 |
| A Star Rating | INT | 1 | 5 |

The import file for this database has been included.

# Design Prototype

The prototype design file, created in Figma, is included in this project. Below are the exported screenshots of the design.

### Genres



Figure . The Genres page.

### Movie Search



Figure . The search page.

### Popular Movie Charts



Figure . The Top 10 charts page.

### Movie Collections

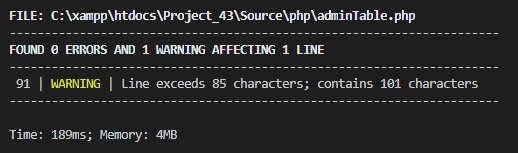


Figure . The movie collection page.

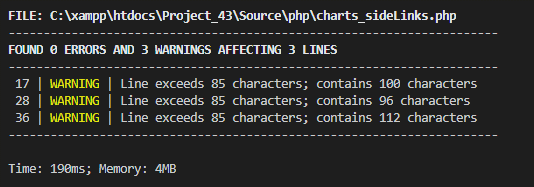
# Testing

## PEAR Code Standard Validation Screenshots

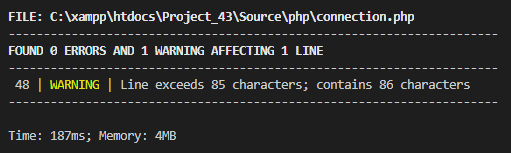
AdminTable.php



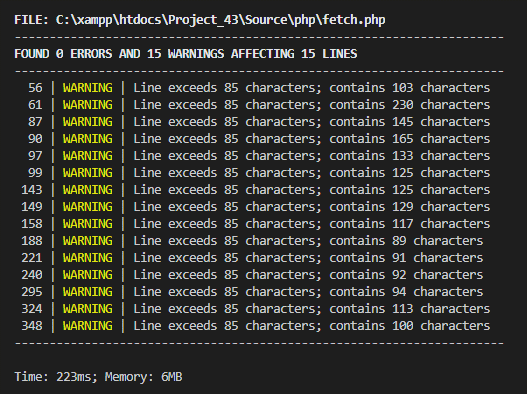
Chart\_sideLinks.php



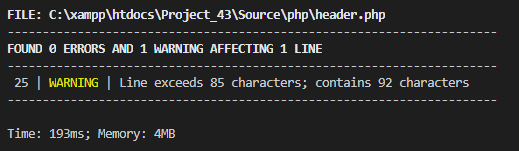
Connection.php



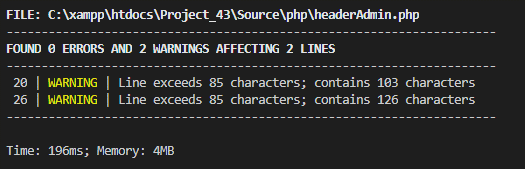
Fetch.php



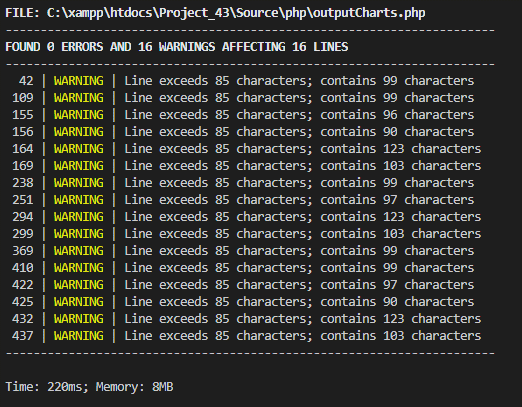
Header.php



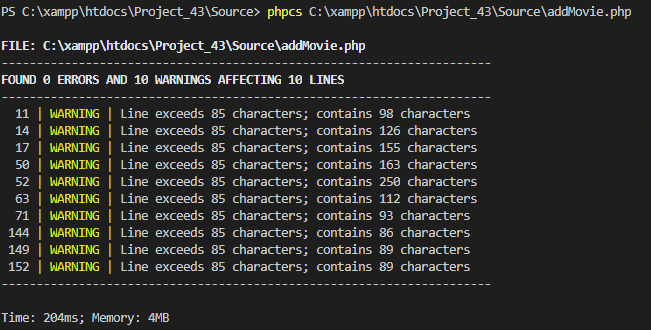
HeaderAdmin.php



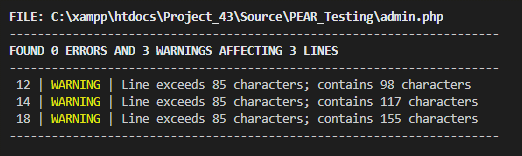
OutputCharts.php



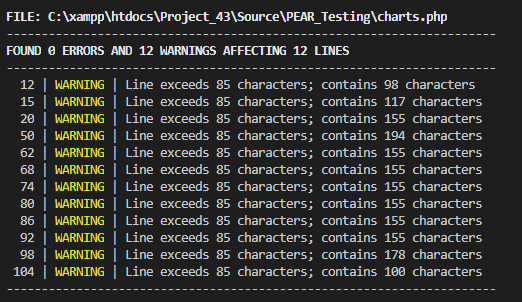
addMovie.php



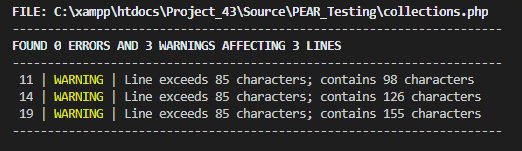
Admin.php



Charts.php



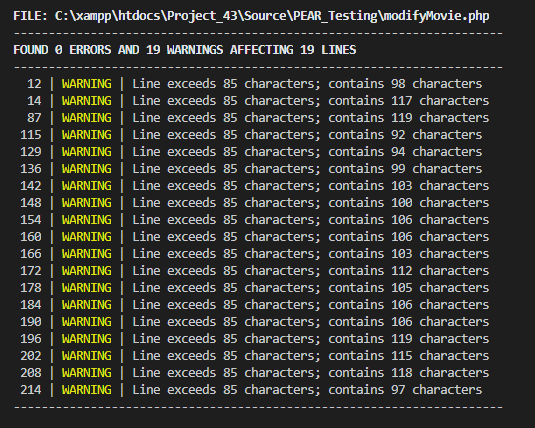
Collections.php



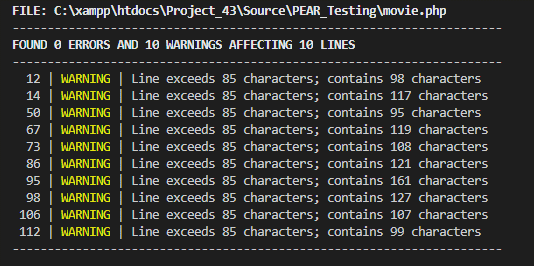
Index.php



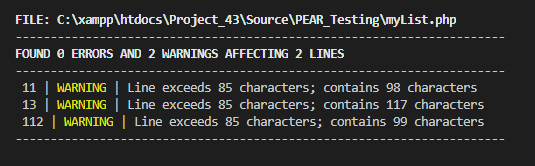
modifyMovie.php



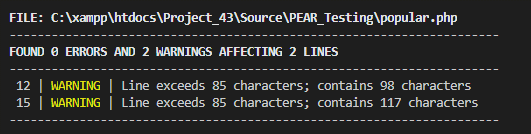
movie.php



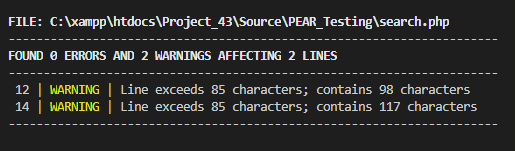
myList.php



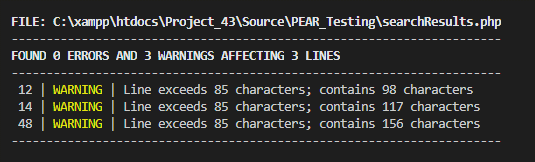
popular.php



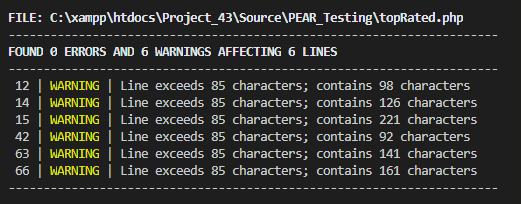
search.php



searchResults.php



topRated.php



## Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Case #** | **Functionality** | **Data used** | **Expected outcome** | **Pass / Fail** |
| 1 | Search | No data. | All movies will be resulted. | PASS |
| 2 | Search | Movie title only. | Movies containing title keyword resulted. | PASS |
| 3 | Search | Movie title and year. | Movies containing title keyword from specified year resulted. | PASS |
| 4 | Search | Movie title, year and genre. | Movies containing title keyword from specified year and genre resulted. | PASS |
| 5 | Search | Movie title, year, genre and a rating. | Movies containing title keyword from specified year, genre and rating resulted. | PASS |
| 6 | Search | A non-existent year. | No results returned. | PASS |
| 7 | Movie Details | Movie ID 1481. | Movie details displayed in table form. | PASS |
| 8 | Movie Details | Movie ID 1174. | Movie details displayed in table form. | PASS |
| 9 | Popular Charts | Top 10 80’s movies. | Shows the top 10 movies from the 1980’s. | PASS |

|  |  |
| --- | --- |
| **Case #** | **Screenshot** |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
| 7 |  |
| 8 |  |
| 9 |  |

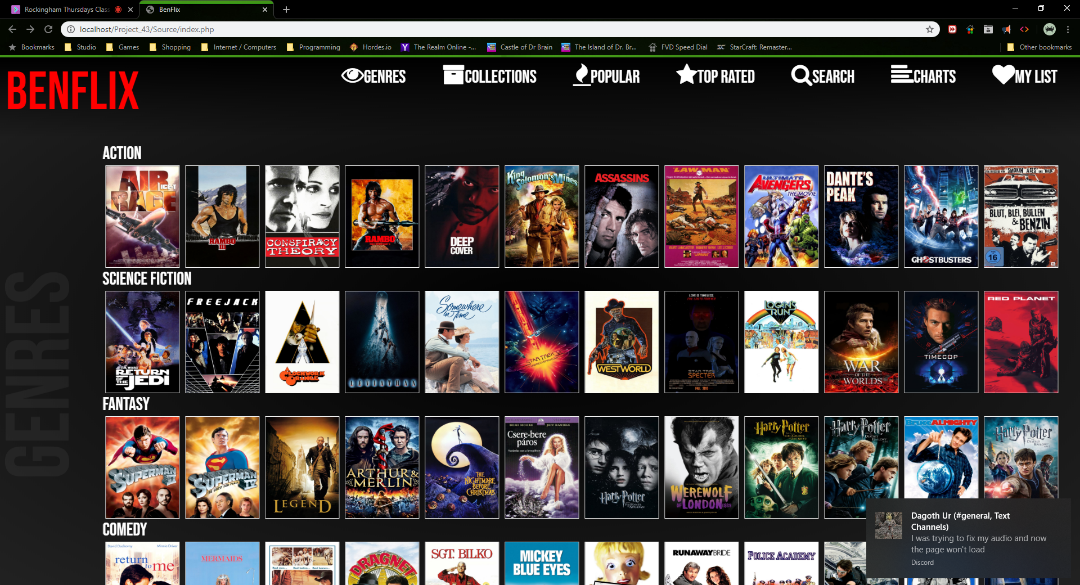
## Validate Consistency

There is a high level of consistency between the designs and the actual application. The functions such as adding a user and listing all users function correctly as intended. The physical layout also remains similar to the design created in Figma.

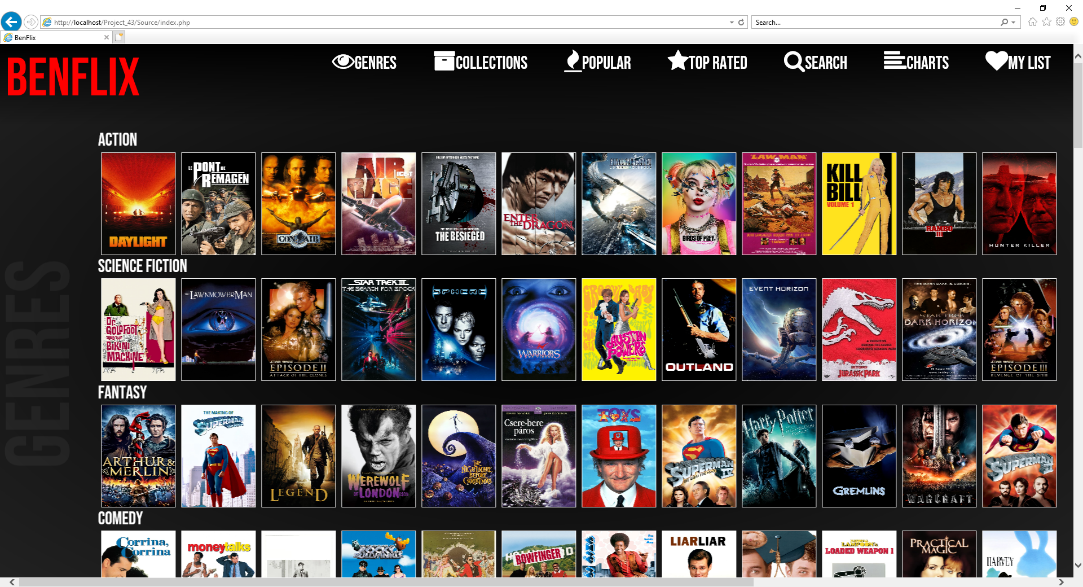
A MySQL table has been used to store the names and emails of users in the database. PHP is used in this project to provide a connection between the HTML elements of the application and the backend data in the table. This application is overall consistent with its design and requirements.

## Browser Tests

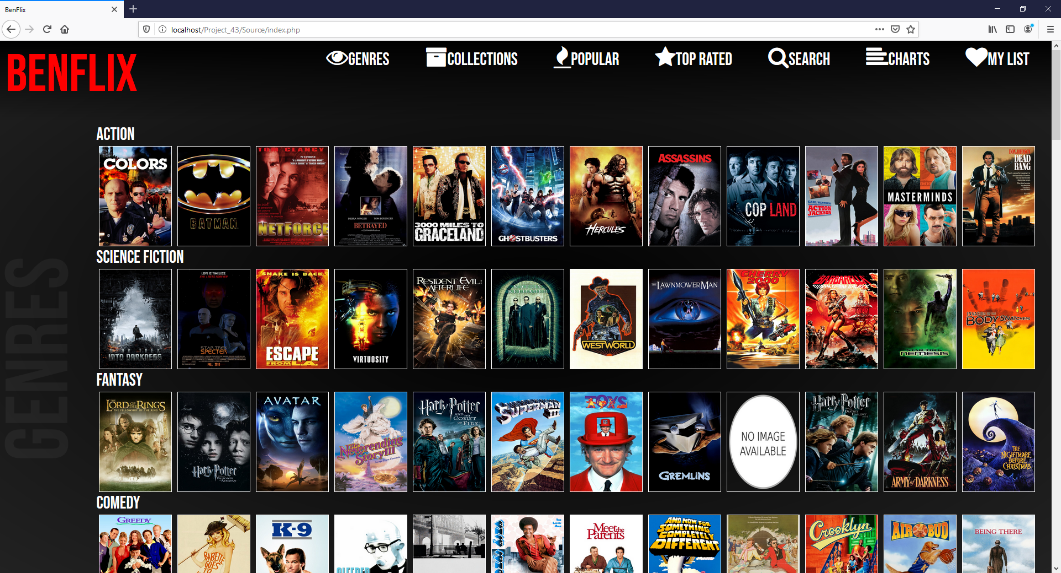
### Google Chrome



### Internet Explorer



### Mozilla Firefox



# References

Beautify Tools. (2020). *Online PHP Beautifier*. Retrieved from Beautify Tools: http://beautifytools.com/php-beautifier.php

Web Code Sniffer. (2020). *Web Code Sniffer | Interface*. Retrieved from Web Code Sniffer: https://www.webcodesniffer.net/