Timur Golovinov

12/24/2019

Movie Rental Website



Table of Contents

[I. Develop and run environment, tools will be used for design and development. Set up of server-side environment and database. 3](#_Toc7029794)

[Ambiguity in the specifications and requirements and clarify them before designing the application. 3](#_Toc7029795)

[II. Design the application 4](#_Toc7029796)

[The prototype of the user interface in a visual format (template web site). 4](#_Toc7029797)

[Website navigation hierarchy. 5](#_Toc7029798)

[Database design: database validated against the design, e.g. database structure. 5](#_Toc7029799)

[III. Developed application illustrated: 6](#_Toc7029800)

[A search form, searchable by the title, genre, rating and year. For example, Search movies by genre (Comedy or SciFi) with Rating (PG) in 1995 or search movies by title with genre, or search by title only. 6](#_Toc7029801)

[Full details of each movie found. If no movie is found, the application will display a user-friendly message. 7](#_Toc7029802)

[An option to display a chart image of top 10 most frequently searched movies. 8](#_Toc7029803)

[IV. Test documentation 9](#_Toc7029804)

[o Test the code against the PEAR code standard with an appropriate static analysis tool, e.g. PHP\_CodeSniffer 9](#_Toc7029805)

[▪ Fix the errors detected. E.g. PHP Code Beautifier utility. 10](#_Toc7029806)

[o Software risk analysis and a simple report. 10](#_Toc7029808)

[▪ Potential software risks or problems, e.g. possible invalid user inputs that cause unexpected behaviour? Is requirements incomplete? Security issues? 10](#_Toc7029809)

[o Tested and validated with sample inputs with screenshots. Test the logic of the contents of the application. Validate the consistency between the code and the software design. Include testing in various browsers, e.g. Chrome and Firefox. 11](#_Toc7029810)

[o An evidence of adjusted application base on the test. 11](#_Toc7029811)

# Develop and run environment, tools will be used for design and development. Set up of server-side environment and database.

Visual Studio Code will be used as a development environment. The program could be run on any web browser, such as Chrome, Edge, Mozilla, Safari and others.

Tools for design and development will be html – for the content and forms, css – for styling, php – for server-side communication. Moreover, additional tools may be used such as JavaScript – for the website behaviour, external graphics (e.g. SVG icons), API’s, various libraries, and frameworks (such as bootstrap).

XAMPP with Apache and MySQL modules will be used to provide server-side and database communication.

## Ambiguity in the specifications and requirements and clarify them before designing the application.

There could be too many ambiguity and interpretations around the existing requirements. Concerns such as security, design, UX, system architecture, branding (colour schemes, logos, etc.) could be addressed more specifically.

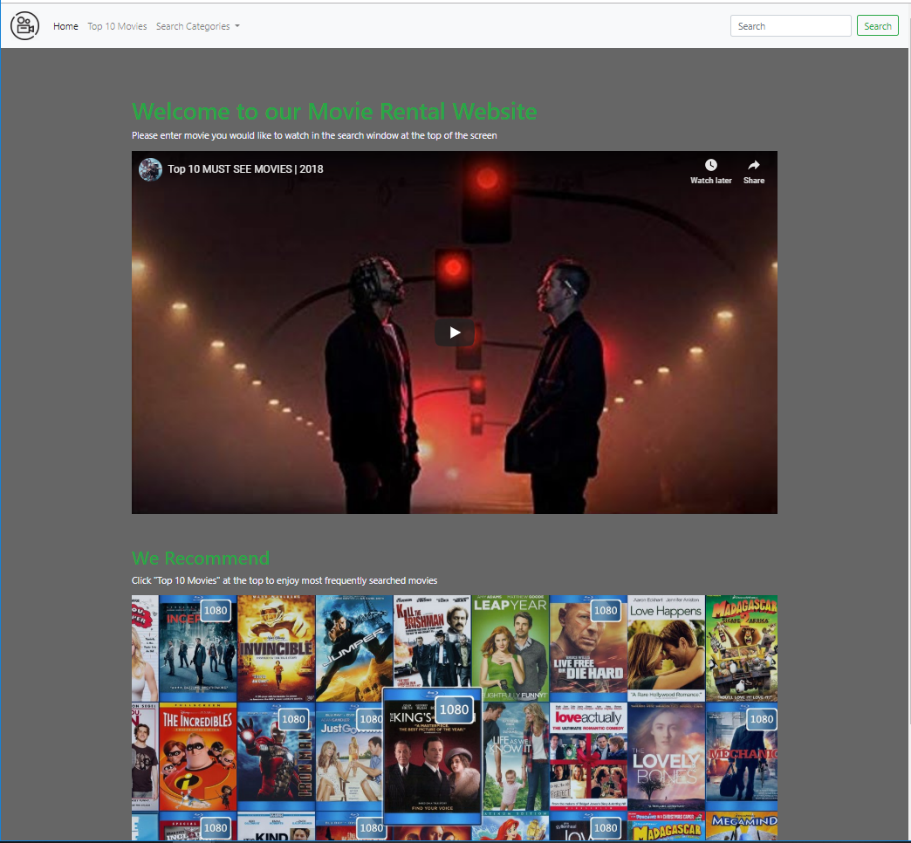
Therefore, the website should meet required criteria in all of those fields. For example:

* Security – form validation would be sufficient.
* Design/UX/Branding – style consistency (fonts, colours, shapes).
* System Architecture – not a deep navigation, simple database architecture (single table with multiple columns).

# Design the application

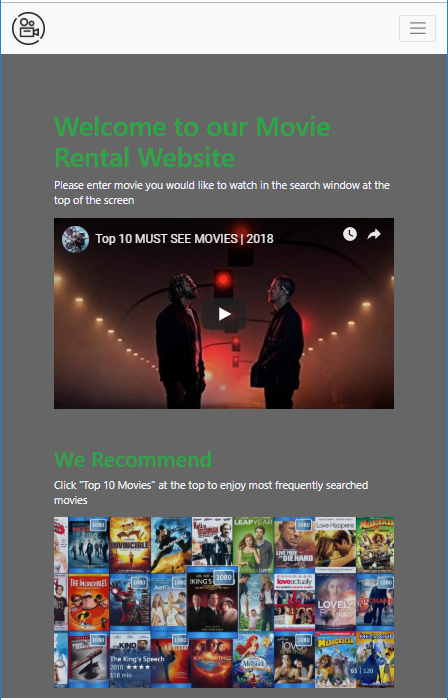
## The prototype of the user interface in a visual format (template web site).

User interface for big screens: Laptops, tables, PCs. (minimum width 792px)



Figure

Mobile version of the user interface. (Width up to 792px)



Figure

## Website navigation hierarchy.

Search

Home Page

Top 10

By Year

By Rating

By Genre

By Title

## Database design: database validated against the design, e.g. database structure.

Database “Movies” consists of one table called “movietable”. The table includes columns “ID” – Primary Key, “Tile”, “Studio”, “Status”, “Sound”, “Version”, “RecRetPrice”, “Rating”, “Year”, “Genre”, and “Aspect”.

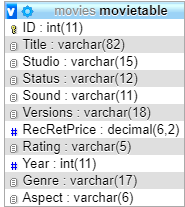


Figure 3

# Developed application illustrated:

## A search form, searchable by the title, genre, rating and year. For example, Search movies by genre (Comedy or SciFi) with Rating (PG) in 1995 or search movies by title with genre, or search by title only.

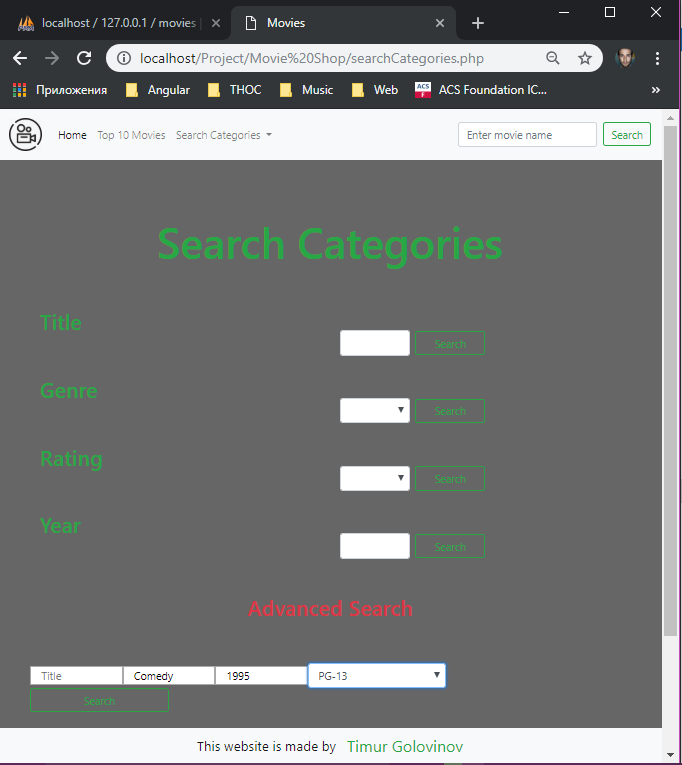


Figure 4

## Full details of each movie found. If no movie is found, the application will display a user-friendly message.

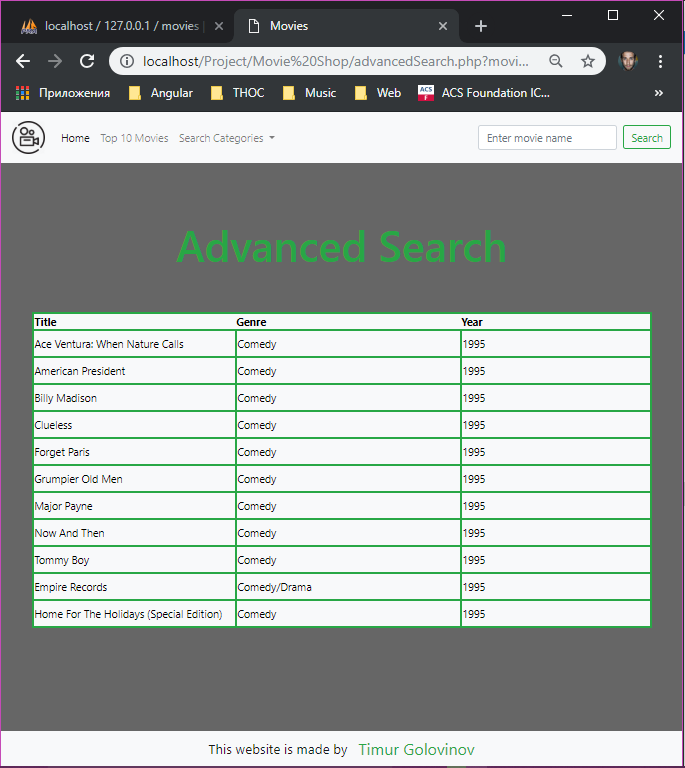


Figure 5

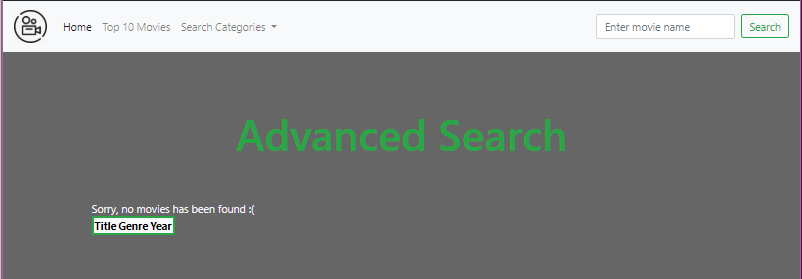


Figure 6

## An option to display a chart image of top 10 most frequently searched movies.

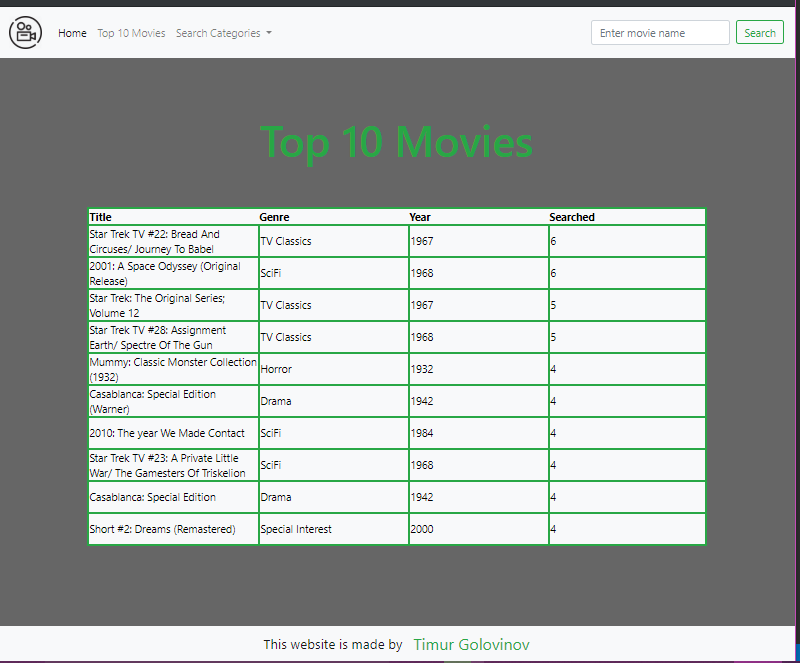


Figure 7

# Test documentation

## o Test the code against the PEAR code standard with an appropriate static analysis tool, e.g. PHP\_CodeSniffer

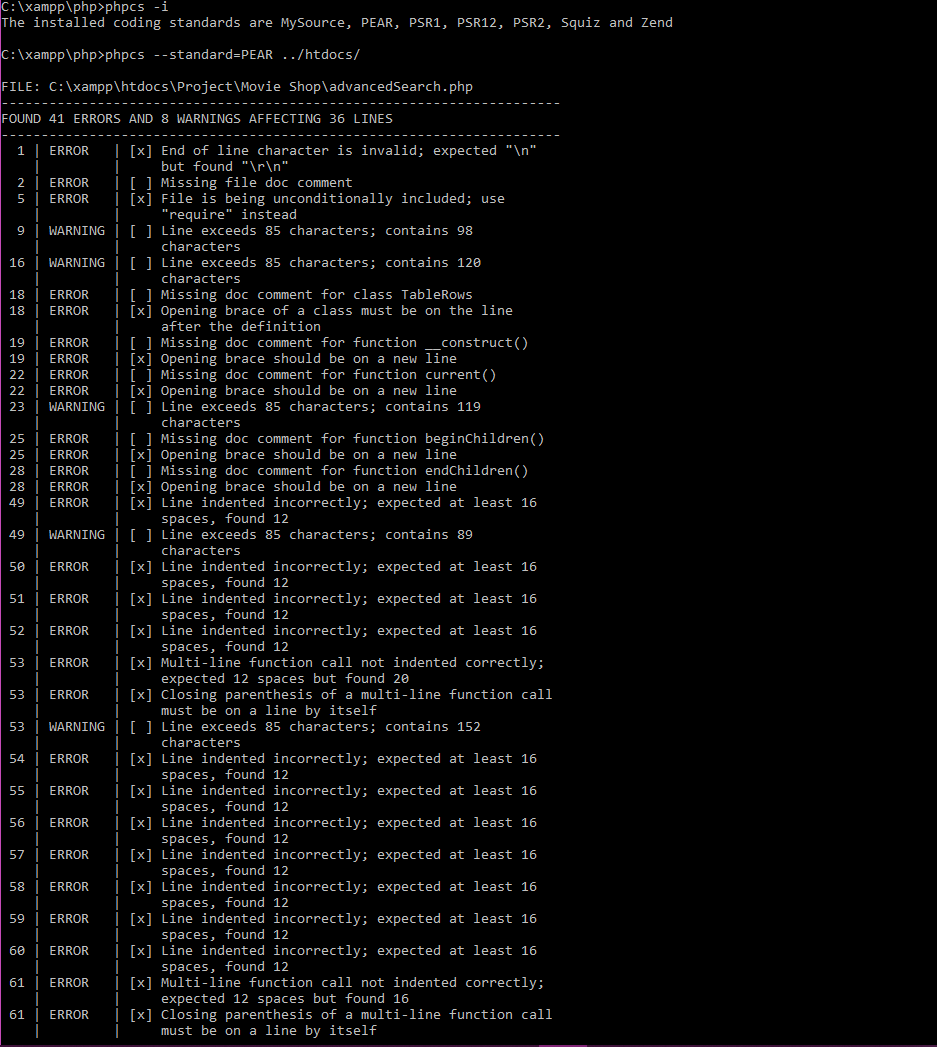


Figure 8

## ▪ Fix the errors detected. E.g. PHP Code Beautifier utility.

## 

Figure 9

## 

## o Software risk analysis and a simple report.

## ▪ Potential software risks or problems, e.g. possible invalid user inputs that cause unexpected behaviour? Is requirements incomplete? Security issues?

There is always a chance to be hacked, some of the most popular vulnerabilities are:

1. SQL injection (which we are protected of by PDO Prepared Statements).
2. Cross Site Scripting - injection of code (usually JavaScript code but it can be any client-side code) into the output of the PHP script.
3. Source Code Revelation - This one has to do with people being able to see the names and content of files they shouldn’t in the event of a breakdown in Apache’s configuration.
4. Remote File Inclusion - All you should do is go to your php.ini and check the settings on these flags.
5. Session Hijacking – Stealing Session ID. In case of user dealing with sensitive information, stolen ID can cause a serious problem.

(top-10-php-security-vulnerabilities, n.d.)

Based on above-mentioned vulnerabilities it is safe to say, that the security requirements are may be incomplete for the website that contains sensitive user information. However, in our case, PDO is meeting the security requirements.

## o Tested and validated with sample inputs with screenshots. Test the logic of the contents of the application. Validate the consistency between the code and the software design. Include testing in various browsers, e.g. Chrome and Firefox.

When input meets the requirements listed on the main page, user receives message, that member has been successfully added.

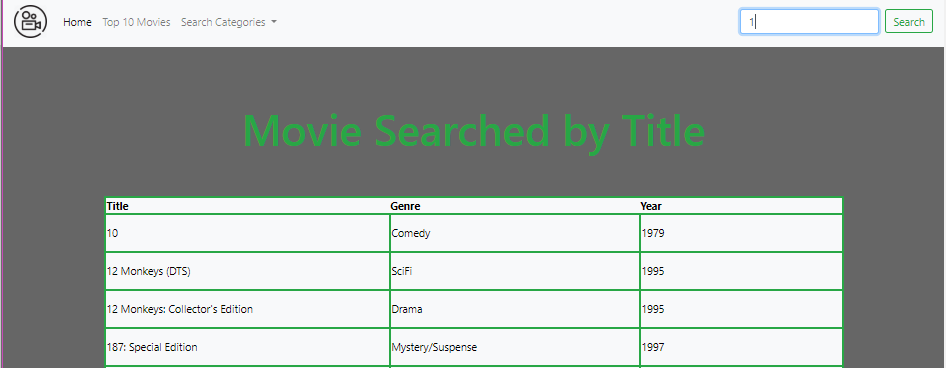


Figure 10

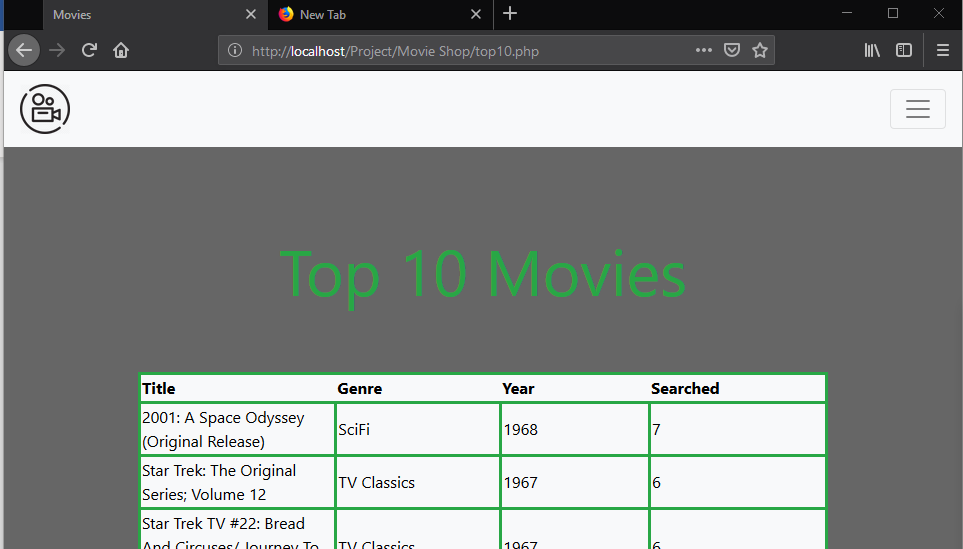


Figure 11

## o An evidence of adjusted application base on the test.

