

**CEN 4010 Principles of Software Engineering**

Summer 2020

**Group 2 “The Fam” Evaluation:**

-Oscar Aquino (25%)

-Ryan Bates (25%)

-Elizabeth Garcia (25%)

-Jesse Kelly (25%)



Revision 1 7/25/2020



**CEN 4010 Principles of Software Engineering**

Summer 2020

**Milestone 4 Beta Launch and Reviews**

**“What 2 Watch” -CURRENT SITE DEMO:**

<https://www.youtube.com/watch?v=r7AUrbsz_Ws>

**Team #2**

Front End Team Leader/Scrum Master: Ryan Bates  
 Back End Team Leader: Oscar Aquino  
 Github/Trello Master: Elizabeth Garcia  
 Product Owner: All Software Developed and owned by Group 2  
 Software Developers:

Oscar Aquino - [oaquino2017@fau.edu](mailto:oaquino2017@fau.edu)

Elizabeth Garcia - [elizabethgar2017@fau.edu](mailto:elizabethgar2017@fau.edu)

Ryan Bates – [batesr2013@fau.edu](mailto:batesr2013@fau.edu)

Jesse Kelly – [jkelly2019@fau.edu](mailto:jkelly2019@fau.edu)

Document date: 7/27/2020

Revised history:



A website dedicated to the establishment of quality choices in what to watch established by the consumers, and how to find them.

**PRODUCT SUMMARY**

“What 2 Watch”

Priority 1 Functions include a search function, a user sign-up function, and a user discussion board. With these functions as our top priority we believe that our software has the potential for an easy and generalized platform in terms of usability and satisfaction. For our first launch, we will have our priority 1 functions squared away in hopes of bringing the user a full functioning experience with minimal bugs. The thinking behind the choice for each priority 1 function is as follows.

**User sign-up function** is the most important to us because on our first launch, we want the user to feel as if they’re joining us on this journey and as if they’re a part of the team and what we’re doing. Having them sign up allows for us to keep them updated as newer versions are released.

**Search function** is one that most software takes advantage of and is one that fits nicely with what we’re trying to accomplish on “What 2 Watch”. This function is for the user’s convenience and versatility when deciding what to watch with any specific ideas in mind.

**User Discussion Board** is a way for the users to post personalized feedback that our other user’s might be able to relate to. This social aspect of our software is especially important to help build our community as well as add versatility in the sense that our users will be able to evaluate their decisions from an informational perspective as well as being able to read into others’ perspective that may have already been exposed to certain content.

**USABILITY TEST PLAN - (Movie Search Functionality)**

**Test Objectives –**

In an effort to isolate one aspect of functionality for a usability test, we will find it worthwhile to get a grasp on how our search functionality is performing primarily in response to how helpful it is in the context of our website, and how it performs against other methods of implementation. It will be in our best interest to gather an unbiased perspective from clients that haven’t yet been exposed to our platform.

The most basic necessities at this increment of our development include: being able to type in a movie title and press the search icon, refreshing to the correct page, and being able to search as many titles as you prefer in a row.

Some bugs that we’re aware of include: On the home page, the search icon placement is overlapping the user login button, the enter button is not yet synonymous with the search button on the site.

**Test Plan –**

* Follow the link to the home page of our site, “What 2 Watch”
* As an outside user without an account yet, attempt to search for some of the titles that are suggested on the home page, and see if they are displayed on their own page.
* From the page the user has arrived at after the initial search, continue to search at least 3 more titles in an effort to see if the search function will perform from the last page you were on. (The goal is to have search functioning at each page on the site)

**LINK TO HOMEPAGE OF WHAT2WATCH:**

<https://lamp.cse.fau.edu/~cen4010s2020_g02/Milestone_4/BOOTSTRAP/www/index.html>

**Likert Scale Questions –**

\*(this survey was filled out by an outside user with little exposure to computer science)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Strongly Disagree | Disagree | Neither Disagree or Agree | Agree | Strongly Agree |
| The Search Bar loads the exact movie typed in and was not case sensitive. |  |  |  |  |  |
| The Search Bar performed from the home page as well as the movie page allowing for multiple searches in a row. |  |  |  |  |  |
| The Search Bar works similarly to other search engines you have used in the past. |  |  |  |  |  |

**QA TEST PLAN**

**Test Objective**:

The objective for this test plan is to define our software setup and cover some of the basic specifications in regard to what we are including. The main function that this test plan is focusing on is one of our priority 1 functions which is the search function. Starting with some basic information, we will assess the foundation of certain specifications that have went into the platform that we currently have. This will allow for a full circle understanding of what we’re doing and trying to accomplish before presenting the test cases for the search function. We will also include test results at the end of this section to give an idea of where the software currently stands.

**Hardware/Software Setup/Basic Information**

Product Name: “What 2 Watch”

Developer: Team 2

Computing Environment: Web-based (computers, tablets, phones)

Type of Software: User Interface

Type of Hardware: Computer, Laptop, Mobile Phone, Tablet

Users: Within the system users will have an account

Purpose of Testing: Priority 1 functionality for Search function

Type of Testing: Functionality Tests

Priority 1 feature to be tested: SEARCH FUNCTIONALITY

TEST CASES:

1. Search box and Search Button (submit script)
2. API concatenation with local storage string title
3. API JSON fetch and parsing to HTML

TESTING TABLE:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Test #** | **Test Title** | **Description** | **Input** | **Expected Output** | **Results** |
| **Browser 1 (Chrome)** | **1** | Search box and Search Button | Testing the submit script functionality using HTML form format | Movie title as a string | String is saved into local storage | PASS |
|  | **2** | API concatenation with local storage string title | Testing the fluidity of the search functionality string concatenation for the API | The saved local string | String is concatenated onto the API function call | PASS |
|  | **3** | API JSON fetch and parsing to HTML | Testing the actual selection of the correct movie title | Concatenated String | Matching JSON object return | PASS |
| **Browser 2**  **(Edge)** | **1** | Search box and Search Button | “ | “ | “ | PASS |
|  | **2** | API concatenation with local storage string title | “ | “ | “ | PASS |
|  | **3** | API JSON fetch and parsing to HTML | “ | “ | “ | PASS |

**CODE REVIEW**

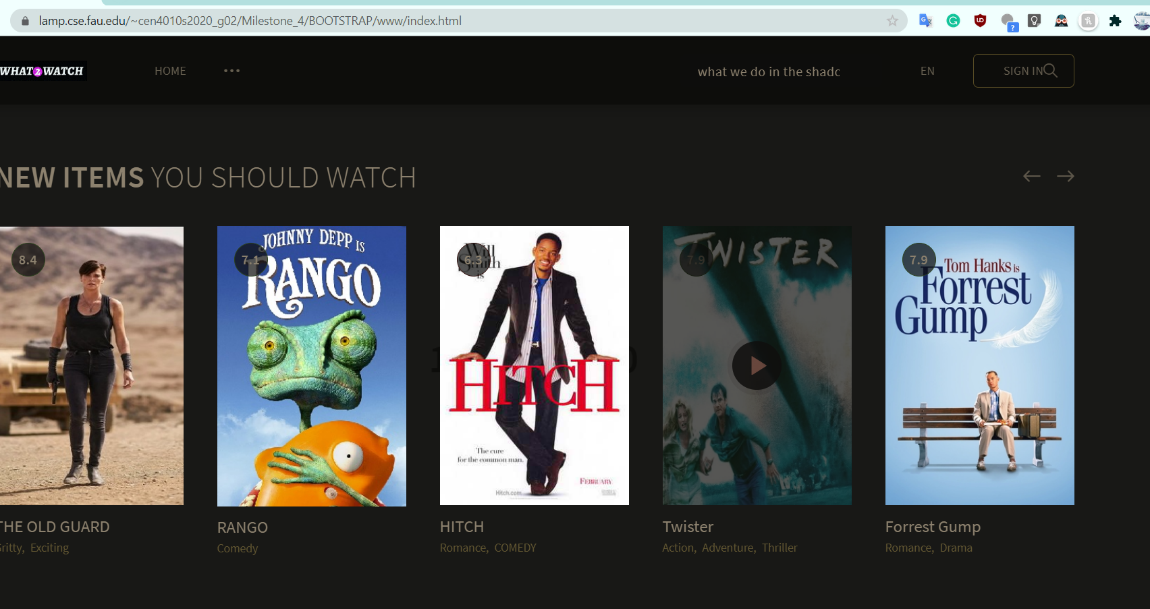
Code was written in Visual Studio and Brackets and merged together.

SEARCH FUNCTION BREAKDOWN:

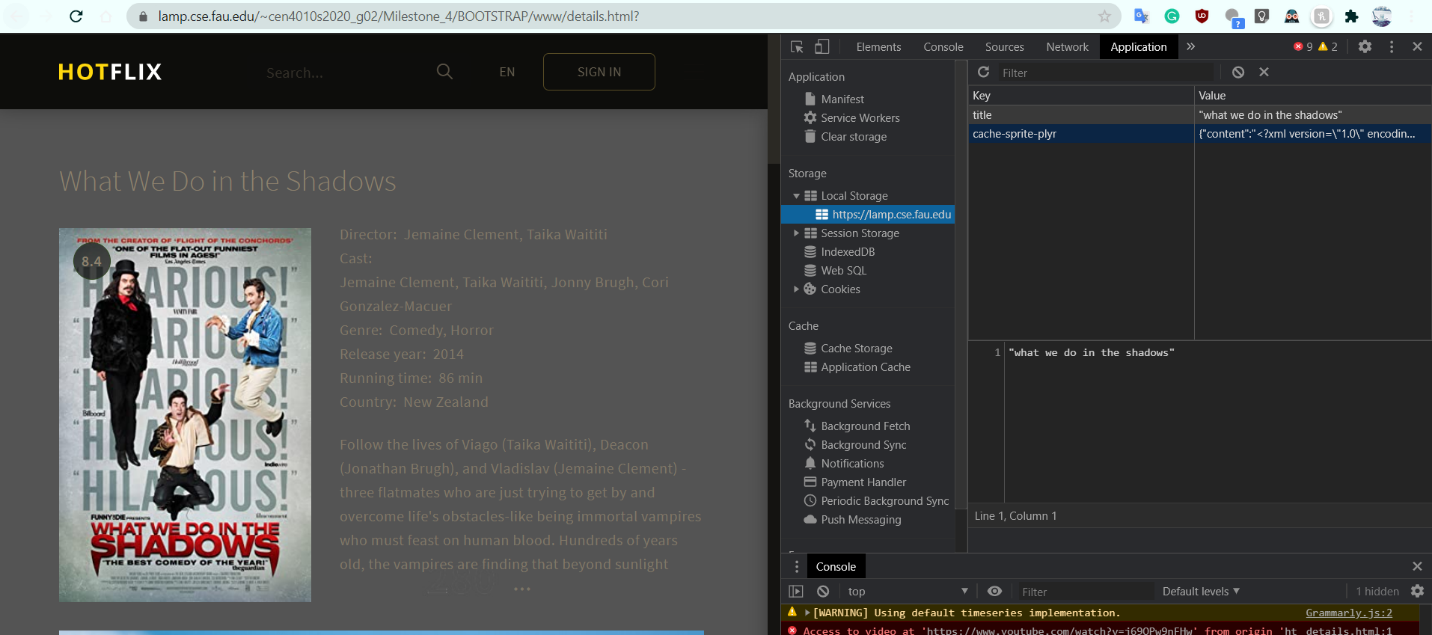
API and search breakdown:

User types a string into the search box.

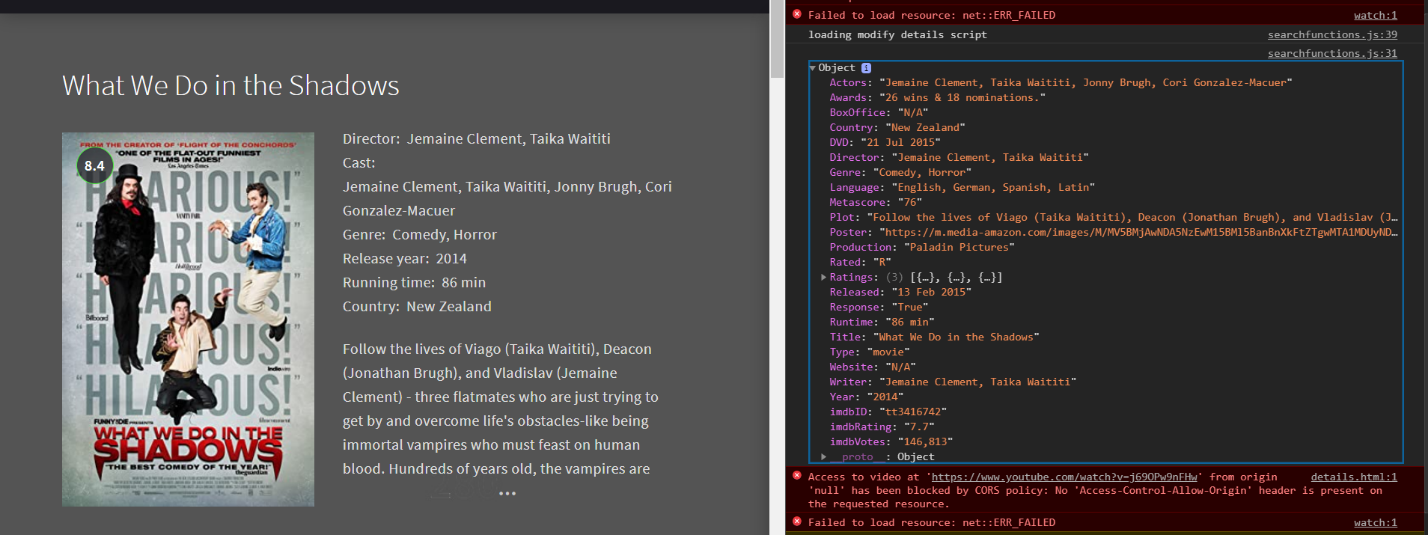
1. The search box and search button are structured using a HTML form format.
2. When the user clicks the button then the form is submitted, activating the JS file, “search functions”.
3. The string is automatically saved into local storage and sent to a function that concatenates the string into the API URL. Using fetch, the API (omdb) returns a json. The json is then fetched and aggregated into the HTML DOM using query selector.



Above when the user types in a string, then clicks search, the form will submit, and redirect the user.



Once it submits, you are taken to the details page and the string is saved to local storage (right).



The object is fetched ( console log on the right shows the response) using the string title saved to local storage and then concatenated onto the API url call. Then we parse through the json object and modify the DOM using query selector.

Using the API functions, we can also auto select the closest match. Typing: “What we do in the shad” will bring up the movie “What We Do in the Shadows”, a longer plot is also used in our application to fill the entire plot div.

1. **const** search\_movie = **function**(){
2. document.getElementById('search\_button').addEventListener('click', sendMovie);
3. }
5. **const** sendMovie = **function**(event){
6. event.preventDefault();
7. **const** user\_title = document.getElementById('title').value;
8. localStorage.setItem('title', JSON.stringify(user\_title));
9. document.getElementById('form-search').submit();
10. }
12. **function** consoleTitle(){
13. console.log('Hello, Console!')
14. }

17. async **function** apiCall(){
18. let movie\_name\_storage = localStorage.getItem('title');
20. **const** response = await fetch('https://www.omdbapi.com/?t=' + movie\_name\_storage + encodeURI('&apikey=7b3941a8')+ "&plot=full");
21. **const** api\_response\_print = await response.json();
22. document.querySelector('#details\_title').innerHTML = api\_response\_print.Title;
23. document.querySelector('#director\_name').innerHTML = api\_response\_print.Director;
24. document.querySelector('#cast\_names').innerHTML = api\_response\_print.Actors;
25. document.querySelector('#genre\_details').innerHTML = api\_response\_print.Genre;
26. document.querySelector('#release\_year').innerHTML = api\_response\_print.Year;
27. document.querySelector('#runtime\_details').innerHTML = api\_response\_print.Runtime;
28. document.querySelector('#country\_details').innerHTML = api\_response\_print.Country;
29. document.querySelector('#plot\_details').innerHTML = api\_response\_print.Plot;
30. document.querySelector('#poster\_details').src = api\_response\_print.Poster;
31. console.log(api\_response\_print);
32. }


36. document.addEventListener('DOMContentLoaded', search\_movie);
38. **function** modify\_details\_page(){
39. console.log("loading modify details script");
40. apiCall();
41. }

**SELF-CHECK: BEST PRACTICE (SECURITY)**

The most important assets that we are protecting in terms of security are the users’ information and passwords.

Our user login section is in the process of being built, but authentication is required and passwords will be encrypted in order to protect each individual user.

Once the password is encrypted it will be stored in the Database.

**SELF-CHECK: ADHERENCE TO ORIGINAL NON-FUNCTIONAL SPECS**

DONE Performance

ISSUE Capacity - non-functional spec was removed on our Milestone 3, therefore we are not making this a priority

ON TRACK Reliability

ON TRACK Maintainability

ON TRACK Serviceability

DONE Security

DONE Usability