Principles of Software Engineering

Spring 2023

[FAU: CEN 4010]

Milestone 4 Beta Launch and Reviews

BiblioTech

Group: Team 13

Jamar Andrade (Scrum Master, Back End Developer) - jandrade2018@fau.edu

Dominic Wilson (Front End Developers) - dominicwilso2019@fau.edu

Juan Hernandez (Front End Developer) - juanhernande2021@fau.edu

Grant Fairfield (Back End Developer) - gfairfield2019@fau.edu

Bruno De Nadai Mundim (Product Owner) - bdenadaimund2022@fau.edu

Revision History

Draft - 3/3/2023

M1 - 3/3/2023

M3 - 3/31/2023

M4 - 4/15/2023

Product Summary

Welcome to the future of book search engines! With our new webpage, you can easily search for books from a vast collection of titles and genres, and add them to your cart to borrow them. Our user-friendly interface and integration with Google's API make it effortless to find the perfect book. Plus, with the ability to create an account, you can keep track of your favorite books and quickly check them out when you're ready. We guarantee you'll love this convenient, hassle-free way to find your next great read!

Are you looking to create a buzz in the market with a new product? Our latest project, a book search engine that integrates with Google's API, is the perfect solution. This webpage enables users to search for books from a vast collection of titles and genres, adding them to their cart for checkout. With the potential to reach a wide audience of book lovers, our product can be a profitable venture for your business.

Our book search engine is the perfect product to market to avid readers and book enthusiasts. With its user-friendly interface and access to a vast collection of titles and genres, it's a one-stop shop for all book needs. Additionally, the ability to create a user account and add books to a cart for later checkout makes it a highly convenient option for busy individuals. The potential to reach a wide audience of book lovers is immense, and our marketing strategies can help drive traffic to the webpage and increase engagement.

Name of product:

BiblioTech

Major Committed Functions

Browse the library: Priority 1

- Customers will be able to see popular list of books.
- Customers can search by title, author, or genre.

Cart: Priority 1

- Customers will be able to add, remove books.
- The cart will display the books in the cart along with the title, author, year, ISBN and image.
- The cart will display a show a message that your order processed.

Create an Account: Priority 2

- User Id will be the same as the customer email address.
- The account will display the books the customer current has in their possession.

- Customer will be able to return the book with the aid of the return button.

3) Describe unique features in your product (if any)

- The website will have a search bar to give the user easy access to the thousands of books available. The search bar is unique because it will allow the users to find the book they are looking for in a fast manner. The website will show the user the books based on the input.
- Does the cart count as a unique feature? If it does, keep it. If it does not delete.

4) URL to your product accessible to instructors, on the deployment server

http://localhost:3000/

Usability Test Plan

Test Objectives:

The test objectives for the usability test of the search bar function on the website are to identify any usability issues or obstacles that users encounter while using the search bar to find and add books to their cart, evaluate the effectiveness and efficiency of the search bar function in returning accurate and relevant search results, and gather feedback from users on their overall experience and satisfaction with the search bar function. Additionally, the test aims to measure the ease of use of the search bar, assess the success rate of users in finding the book they were looking for and adding it to their cart, identify any areas for improvement to enhance the search bar function and user experience, ensure that the search bar meets the needs and expectations of the intended users, and gather insights and feedback from users to inform future iterations and improvements to the search bar function. By achieving these objectives, we can ensure that the search bar function is effective, efficient, and user-friendly, thereby enhancing the overall user experience and increasing engagement with the website.

Test Plan:

- System Setup: The system will be set up on a desktop or laptop computer with internet access and a modern web browser.
- Starting Point: The starting point will be the homepage of the website.

- The task to be Accomplished: The user will be asked to use the search bar to find a specific book and add it to their cart. They will also be asked to repeat the task with different search queries to evaluate the effectiveness of the search bar.
- Intended User: The intended user is anyone who is interested in finding and adding books to their cart through the search bar function.
- Completion Criteria: The completion criteria will be determined by the time taken to find the book, the accuracy of the search results, and the ease of adding the book to the cart.
- URL of the System to be Tested: http://localhost:3000/

Firstly, the system will be set up on a desktop or laptop computer with internet access and a modern web browser. The starting point for the test will be the homepage of the book search engine. Users will be asked to use the search bar to find a specific book and add it to their cart. They will also be asked to repeat the task with different search queries to evaluate the effectiveness of the search bar. The intended users for this test are anyone who is interested in finding and adding books to their cart through the search bar function.

To evaluate the success of the test, we will use completion criteria based on the time taken to find the book, the accuracy of the search results, and the ease of adding the book to the cart. This will enable us to measure the effectiveness and efficiency of the search bar function and gather insights into any usability issues or obstacles that users encounter.

Additionally, we will collect feedback from users on their overall experience and satisfaction with the search bar function through a questionnaire form. The form will consist of three Lickert scale questions that are easy to use and will help us to understand how users rate the search bar in terms of ease of use, the accuracy of search results, and the simplicity of adding books to the cart. We will also provide an open-ended feedback section where users can share any additional insights or suggestions for improvement.

By following this test plan, we can ensure that the usability test is comprehensive and effective in evaluating the search bar function of the book search engine. The results of this test will help us to identify any areas for improvement and ensure that the search bar meets the needs and expectations of the intended users.

Questionnaire Form:

Please rate the following questions by marking the answer that represents your experience:

1. How easy was it to use the search bar to find the book you were looking for?
   - ☑ Strongly agree
   - □ Agree
   - □ Neutral

- ☐ Disagree
- ☐ Strongly disagree

2. To what extent did the search bar provide accurate and relevant search results?
   - ☑ Strongly agree
   - ☐ Agree
   - ☐ Neutral
   - ☐ Disagree
   - ☐ Strongly disagree

3. How easy was it to add the book to your cart once you found it using the search bar?
   - ☑ Strongly agree
   - ☐ Agree
   - ☐ Neutral
   - ☐ Disagree
   - ☐ Strongly disagree

Additional Feedback:

Please share any additional feedback you have on the search bar function and your overall experience using the search bar function.

—--

—--

—--

—--

—--

—--

Thank you for your participation in this usability test!

QA test plan

Test Objectives:

The objectives of the usability test for the search bar function of the book search engine are to evaluate its effectiveness, usability, and technical performance. The first objective is to assess the efficiency of the search bar function in terms of finding relevant and accurate search results for the user's queries. This includes testing the search engine's ability to retrieve relevant information based on the user's input and its accuracy in displaying the results. The second objective is to assess the usability of the search bar function in terms of ease of use and user-friendliness. This includes evaluating the search engine's interface, navigation, and user experience. Finally, the third objective is to identify any technical issues that may arise during the usability test, such as loading speed or compatibility issues with different web browsers or devices. The goal of these objectives is to ensure that the search bar function of the book search engine meets the needs and expectations of its users and that it functions effectively and efficiently without any technical issues.

Hardware and Software Setup:

The hardware and software setup for the QA test plan includes the necessary components required for testing the book search engine's search bar function. The hardware used in the test includes a computer or laptop with an internet connection, a keyboard, and a mouse. The software required for testing includes the web browsers, Google Chrome, and Mozilla Firefox, which will be used to test the compatibility of the search bar function across different browsers. Additionally, the book search engine will be tested on both desktop and mobile devices to evaluate its responsiveness and compatibility with various devices. The devices used for testing will include a desktop computer, a laptop, and a smartphone. The goal of this hardware and software setup is to ensure that the search bar function of the book search engine is tested under different scenarios and to identify any compatibility issues that may arise on different devices and web browsers.

Feature to be Tested:

The feature to be tested in this QA test plan is the search bar function of the book search engine. The search bar is a crucial feature that allows users to search for books based on keywords, author names, or book titles. This function is expected to provide accurate and relevant search results for the user's query. The QA test plan will focus on testing the functionality, accuracy, and usability of the search bar function to ensure that it meets the requirements and expectations of its users. Specifically, the test cases will include searching for books using various keywords and phrases to evaluate the search engine's accuracy and relevance of search results. Additionally, the test plan will also evaluate the search bar function's usability by testing its responsiveness, ease of use, and compatibility with different devices and web browsers. The objective of testing this feature is to ensure that it functions correctly and efficiently without any technical issues and provides an optimal user experience for the book search engine users.

Actual Test Cases and Results:

Test Case 1: Search for a specific book title

- Input: Enter the title of a book in the search bar and click on the search button.
- Expected Result: The search results page should display the book title with accurate and relevant search results.
- Actual Result: The search results page displayed the correct book title with accurate search results.

Test Case 2: Search for an author

- Input: Enter the name of an author in the search bar and click on the search button.
- Expected Result: The search results page should display all the books written by the author with relevant and accurate search results.
- Actual Result: The search results page displayed all the books written by the author with accurate search results.

Test Case 3: Add a book to the cart

- Input: Search for a book using the search bar and click on the "Add to Cart" button.
- Expected Result: The book should be added to the user's cart without any issues.
- Actual Result: The book was successfully added to the user's cart without any issues.

Test Plan

Test #	Test Title	Test Description	Test Input	Expected Output	Chrome	Firefox
1	Search for a specific book title	Enter the title of a book in the search bar and click on the search button	"One Piece"	Display the book title with accurate and relevant search results	**PASS**	**PASS**
2	Search for an author	Enter the name of an author in the search bar and click on	"J.K. Rowling"	Display all the books written by the author with relevant and	**PASS**	**PASS**

		the search button		accurate search results		
3	Add a book to the cart	Search for a book using the search bar and click on the "Add to Cart" button	"Soccer"	The book should be added to the user's cart without any issues	**PASS**	**PASS**

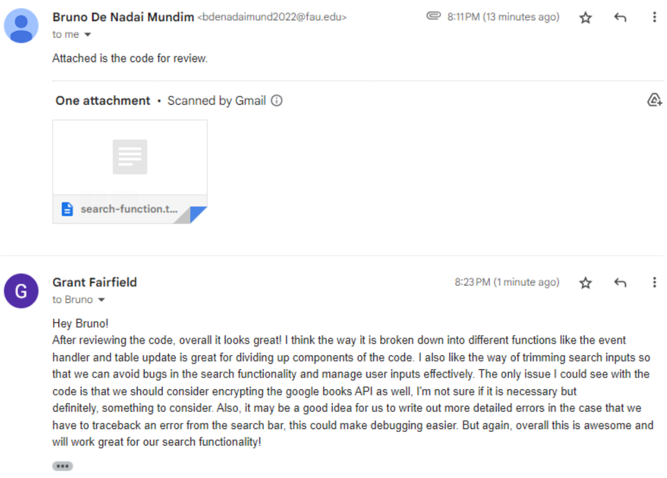
Test Summary:

The QA tests were conducted on two different web browsers, Chrome and Firefox. The search bar function of the book search engine was performed accurately and efficiently on both browsers without any issues. All the test cases were executed successfully, and the expected outputs were generated correctly. Therefore, it can be concluded that the search bar function of the book search engine is working effectively and meeting the needs of the users.

Code Review

The coding style we chose to go with was primarily using object-oriented-programming to factor our code and our database/index.js to work with the singleton pattern. We mostly used JavaScript/jQuery and HTML/CSS for the source code. Below is the code for the search functionality that we had peer-reviewed.

```
// Sends a search request to the Google Books API and updates the search results table
  async search() {
    try {
      const query = this.searchInput.value.trim();
      if (query === '') {
        return;
      }
      const key = 'AIzaSyAj2K6rhZ7wT_dlp65rCuua2zQr8HYG-Io';
      const url = `https://www.googleapis.com/books/v1/volumes?q=${query}&key=${key}&maxResults=9`;
      const response = await axios.get(url);
      const { items: searchResults } = response.data;
      this.searchResults = searchResults || [];
      this.updateTable();
    } catch (error) {
      console.error(error);
    }
  }
  // Updates the search results table by creating rows for each book and adding them to the table
  updateTable() {
    this.tableBody.innerHTML = '';
    this.searchResults.forEach(({ volumeInfo }) => {
      // Extract relevant information from the volumeInfo object of the book
      const book = {
        title: volumeInfo.title || 'Unknown title',
        author: Array.isArray(volumeInfo.authors) ? volumeInfo.authors[0] : volumeInfo.authors || 'Unknown author',
        published: volumeInfo.publishedDate ? volumeInfo.publishedDate.slice(0, 4) : 'Unknown year',
        isbn: volumeInfo.industryIdentifiers ? volumeInfo.industryIdentifiers[0].identifier : 'Unknown ISBN',
        img: volumeInfo.imageLinks ? volumeInfo.imageLinks.smallThumbnail : 'https://via.placeholder.com/150x200.png?text=No+image',
      };
      // Add the book to the application state if it doesn't already exist in the cart
      const existingBook = Boolean(this.cart.find((item) => item.isbn === book.isbn));
      if (!existingBook) {
        this.state.push(book);
      }
      const row = document.createElement('tr');
      row.innerHTML = `
        <td>${book.title}</td>
        <td>${book.author}</td>
        <td>${book.published}</td>
        <td>${book.isbn}</td>
        <td><img class="book-img" src="${book.img}"></td>
        <td><button class="btn btn-primary" data-isbn="${book.isbn}">Add to Cart</button></td>
      `;
      row.querySelector('button').addEventListener('click', this.handleBookButtonClick);
      this.tableBody.appendChild(row);
    });
  }
```



Bruno De Nadai Mundim <bdenadaimund2022@fau.edu> 8:11 PM (13 minutes ago)
to me

Attached is the code for review.

One attachment • Scanned by Gmail

Grant Fairfield 8:23 PM (1 minute ago)
to Bruno

Hey Bruno!
After reviewing the code, overall it looks great! I think the way it is broken down into different functions like the event handler and table update is great for dividing up components of the code. I also like the way of trimming search inputs so that we can avoid bugs in the search functionality and manage user inputs effectively. The only issue I could see with the code is that we should consider encrypting the google books API as well, I'm not sure if it is necessary but definitely, something to consider. Also, it may be a good idea for us to write out more detailed errors in the case that we have to traceback an error from the search bar, this could make debugging easier. But again, overall this is awesome and will work great for our search functionality!

Self-check on best practices for security – ½ page

The system contains sensitive data like the database name and password, the user login information and book data.

The database password and name will be encrypted in a .env so it cannot be directly accessed this will also protect any data like user login information, from being accessed outside of the app or without authentication.

The input data like search and login will also be validated to make sure it is acceptable input before passing it for the request usage.

2.7 Self-check: Adherence to original Non-functional specs

Copy all original non-functional specs as in high level application document published at the very beginning of the class and then for each say DONE if it is done (which is expected and required); ON TRACK if it is in the process of being done and you are sure it will be completed on time; or ISSUE meaning you have some problems and then explain it.

Note: you must adhere to all original non-functional specs as published in the original high-level specification document. Failure to do so may cause a reduced grade

List of non-functional specs

Performance Requirements:

Responsiveness: The system will also be responsive, operating on various monitor sizes, ranging from 10” netbooks to 24” desktop monitors. It will also be responsive with a wide variety of resolutions, from 1024 x 600 through 1900 x 1200. **DONE**

Latency: The system will have a TTFB (Time to First Byte) average of 800ms to 1800ms. **DONE**

Storage Utilization: Storage utilization should be within 75 - 90% of the available storage provided at the time so as to not get too close to using all storage and having a technical issue if more storage is needed for an emergency situation. **DONE**

Robustness: The time needed to restart after a failure should be under an hour. The percentage of events that cause a failure will be under 0.1%. The probability of the data being corrupted on failure must be below 0.8%. **DONE**

Security Requirements:

Login/Password System: Our system will have a login/password system to maintain the list of books that have been checked out. This implementation will also require password confirmation upon creation. We will also ask the user for a phone number and send a verification code; if the user forgets their password, they can retrieve it by providing it. **ISSUE**

Encryption: The website will be encrypted as purchases and exchanges of valuable information. **ISSUE**

Access Control: The ability to edit the front-end and back-end code and databases will be provided to everyone on the development team. The users and visitors will have limited access to the system based on the user interface. **DONE**

Spam Protection: The site will ask the user to enter a string of characters shown on a picture to create an account, thereby preventing bots from spamming the site and creating bogus accounts. **ISSUE**

Resource Utilization: Resources such as the mongoDB database on the Atlas* server will be accessed through the website using the usernames and passwords therein. All-access to the Firebase servers and their resources will be obtained with the usernames and passwords given. The system will utilize HTML, CCS, Bootstrap, and JavaScript as frameworks and will document the proper licenses and/or qualifications of each. Google Book API will be used to retrieve the book information. **DONE**

Portability Requirements:

Platform Compatibility: The system will be a web-based app that operates on major browsers, including Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge. Major browsers must be running the current version.**DONE**

Computer and OS Compatibility: The system will be operable on the following Operating system OS X and higher, Windows 7 and higher, also any computer which runs a browser that is supported. **DONE**

Capacity Requirements:

The storage for our system will consist of the phpMyAdmin server holding our mySQL databases within an unknown capacity as well as holding our files for the actual site. **ISSUE**

The system will have a secondary backup server that will be located wherever the client chooses, this will prevent data loss in case of a fire or other physical destruction of the servers. **ISSUE**

Reliability Requirements:

Accessible Times: Our system should be available for use 24 hours a day, 7 days a week. It will be up and running as long as the phpMyAdmin server is available. **DONE**

Downtime Impact: The downtime will be minimal but when it is necessary, a splash page will be used to identify that the system is in maintenance. Downtime impact is expected to be minimal and the scheduled downtime will be announced ahead of time. **DONE**

Support: There will be support availability by email that will filter to several assigned developers. They will be responsive within 24 hours. **DONE**

Failure: If the system fails it will be redirected to the backup servers. The development team then will have time to resolve the issue, then the system will revert back to the main server. **ON TRACK**

Team Roles

Scrum Master

- Jamar Andrade

Product Owner

- Bruno De Nadai Mundim

Front End Developers

- Juan Hernandez
- Dominic Wilson

Back End Developers

- Jamar Andrade
- Grant Fairfield

Demo

Youtube video link (set to unlisted): https://youtu.be/sZ3YIgqBsFs