Principles of Software Engineering

2023

*[FAU: CEN 4010]*

Milestone 1 Project Proposal and High-level description

BiblioTech

**Group: Team 13**

**Team Lead**

Jamar Andrade - jandrade2018@fau.edu

**Team members**

Dominic Wilson - dominicwilso2019@fau.edu

Juan Hernandez - juanhernande2021@fau.edu

Grant Fairfield - gfairfield2019@fau.edu

Bruno De Nadai Mundim - bdenadaimund2022@fau.edu

M1 3/3/2023

Revision History

Draft - 3/3/2023

Table of Contents

[**Executive Summary 3**](#_heading=h.ta7xv5mpvn5c)

[**Use Cases 5**](#_heading=h.7izka4j29e9k)

[Use Case 1: Book Search and Availability Check 5](#_heading=h.y934v2arsid1)

[Use Case 2: Book Checkout 6](#_heading=h.u1odz8sfncjb)

[Use Case 3: Book Return 7](#_heading=h.na43bhkpg7y3)

[Use Case 4: Cart Creation 8](#_heading=h.zbllie5t01o5)

[Use Case 5: Cart Modification 9](#_heading=h.5n3vn6nk4atc)

[**Data Definition 10**](#_heading=h.hzv0opjepg25)

[**Initial list of functional specs 13**](#_heading=h.en2qx4nglh5e)

[Non-Member expectation 13](#_heading=h.nzquanfyar4m)

[1. Creating Account 13](#_heading=h.uv466xwhgjvy)

[2. Browse the Library 13](#_heading=h.1c4iowvm3g8j)

[3. Cart 14](#_heading=h.8x9hnevnh6or)

[Members expectations 15](#_heading=h.o1xvusx6aupr)

[4. Edit Profile 15](#_heading=h.km7ruysta9gl)

[**5. Review 15**](#_heading=h.a01w6swhixx3)

[**6. Contact Customer Service 16**](#_heading=h.ohrd3n8g58ci)

[**List of non-functional specs 17**](#_heading=h.uhexip4z1wyr)

[Performance Requirements: 17](#_heading=h.bgxyyv14p5nu)

[Security Requirements: 17](#_heading=h.8feek26iavg8)

[Portability Requirements: 18](#_heading=h.yn4bqtsieetg)

[**Capacity Requirements: 18**](#_heading=)

[Reliability Requirements: 18](#_heading=h.grq0o8j0xlmy)

[**Competitive analysis 19**](#_heading=h.294cmreybs9s)

[Planned advantages: 20](#_heading=h.m24sesg9czc3)

[**High-level system architecture 21**](#_heading=h.2xiziaqqwk4p)

[**Team Roles 23**](#_heading=h.8tr44j8qutu8)

[Scrum Master: 23](#_heading=h.o696295bzrlv)

[Front End Developers: 23](#_heading=h.1m904wwbkbxy)

[Back End Developers 23](#_heading=h.kdbummao2obp)

# 

# 

# 

# 

# Executive Summary

For many years, libraries relied on manual work to complete tasks such as keeping an inventory, checking if a book is being rented with cards on a circulation desk, collecting fines for books that haven’t been returned. Library Management Systems provides libraries with tools to manage their resources and streamline their operations. They have revolutionized the way libraries operate, from borrowing and lending to inventory management.

BiblioTech is a comprehensive system that includes features for managing library resources, users, and circulation. One of the benefits of the system is its ability to manage library resources efficiently, helping librarians catalog their resources in a centralized database and making it easier for them to manage their inventory.

One of the advantages of BiblioTech is that it is highly customizable. It can be configured to meet the individual needs of different types of libraries, such as academic, and public. The system can also be customized to include the library’s own branding, logos, and colors, giving it a more personalized touch. This strengthens the library's identity in the community.

BiblioTech can be accessed from anywhere with an internet connection. Users can access it from their homes, offices, or any other location. This provides flexibility to the user and allows them to search for materials without having to physically go to the library, saving them time and effort. The system is also designed to be user-friendly, with intuitive search and navigation tools that make it easier for users to find the materials they need and also to reserve materials, renew loans, and receive notices about overdue materials.

**BiblioTech - Bringing the library to your fingertips.**

# Use Cases

## Use Case 1: Book Search and Availability Check

Actors:

* User
* Application

Preconditions:

* User must be logged in to their account.

Primary Flow of Events:

1. User navigates to the search page of the application.
2. User enters the title, author, or keyword of the book they are looking for.
3. Application displays all available books that match the search criteria.
4. User selects a book to view its details.
5. Application displays the book's details, including author, publication date, and ISBN.
6. Application displays the book's availability status.
7. User repeats steps 2-6 as needed to find the desired book.

Alternative Flows (Errors):

1. User is not logged in to their account and is prompted to log in before proceeding.
2. No books match the search criteria and the application displays a message indicating no results were found.
3. User selects the wrong book and needs to go back to the search results.

## Use Case 2: Book Checkout

Actors:

* User
* Application

Preconditions:

* User must have a valid library card registered with the online library.
* User must be logged in to their account.
* User must have a book in their cart to checkout.

Primary Flow of Events:

1. User navigates to their cart and clicks the "checkout" button.
2. Application prompts the user to enter their email address to receive a loan confirmation.
3. Application verifies the user's library card and confirms the availability of the book.
4. Application sends a loan confirmation to the user via email.
5. User receives the loan confirmation and checks out the book online.

Alternative Flows (Errors):

1. User does not have a valid library card and is prompted to register for one.
2. User does not have any books in their cart and is prompted to add books to their cart before proceeding.
3. User's library card is invalid and the availability of the book cannot be verified.
4. Application encounters a technical issue and the loan confirmation is not sent to the user.
5. User encounters a technical issue with checking out the book online.

## Use Case 3: Book Return

Actors:

* User
* Application

Preconditions:

* User must have a book checked out from the online library.
* User must be logged in to their account.

Primary Flow of Events:

1. User navigates to their loan history page and selects the book they want to return.
2. User clicks the "return" button.
3. Application prompts the user to enter their email address to confirm the return request.
4. Application updates the book's availability status to "available."

Alternative Flows (Errors):

1. User does not have any books checked out and is prompted to borrow a book before proceeding.
2. User encounters a technical issue with the return button and needs to refresh the page.

## Use Case 4: Cart Creation

Actors:

* User
* Application

Preconditions:

* User must be logged in to their account.

Primary Flow of Events:

1. User navigates to the book search page and selects a book they want to borrow.
2. User clicks the "add to cart" button.
3. Application adds the selected book to the user's cart.
4. User repeats steps 1-3 as needed to add more books to the cart.

Alternative Flows (Errors):

1. User encounters a technical issue with the add to cart button and needs to refresh the page.

## 

## Use Case 5: Cart Modification

Actors:

* User
* Application

Preconditions:

* User must be logged in to their account. User must have books in their cart. Primary Flow of Events:
* User navigates to their cart page and selects a book they want to remove. User clicks the "remove from cart" button. Application removes the selected book from the user's cart. User repeats steps 1-3 as needed to remove more books from the cart or add books back to the cart as described in Use Case 4.

Alternative Flows (Errors):

1. User does not have any books in their cart and is prompted to add books to their cart before proceeding.
2. User encounters a technical issue with the remove from cart button and needs to refresh the page.

# Data Definition

| **Name** | **Meaning** | **Usage** | **Comment** |
| --- | --- | --- | --- |
| User | actor | Use Case scenarios | Adds and access data from database |
| Resource | data | Use Case Scenarios | What users can access on the website |
| Borrowing | data | Use Case Scenarios | Let’s user know material is borrowed already |
| Review | service | Site user service | Allows user to rate the book/article |
| Search | service | Site user service | Allows user to search for specific books/articles |
| System | platform hardware and services | Use Case scenarios | Front end design, back end supporting services, and database |

**Data Definition**

| **Name** | **Meaning** | **Usage** | **Comment** |
| --- | --- | --- | --- |
| Newset | data | Use Case scenarios | Store the newest adding activities |
| InvalidUser | actor | Use Case Scenarios | A user who is not registered with the system |
| Login | service | Site user service | Allow users to have the ability to borrow books |
| Library | service | Site user service | Online library application |
| Search | service | Site user service | Allow user find location/activities |
| Catalog | service | Site user service | Organizes and presents resources to the user |
| Filter | service | Site user service | help user search more clearly by giving specific options |
| BiblioTech.com | domain Name | Site user service | The BiblioTech data type is a unique identifier for a specific online library application |
| Server | production server | Use Case scenarios | Online library application infrastructure |
| HomePage | user Interface | User interface | The first page that a user goes to |

**Data Definition**

| **Name** | **Meaning** | **Usage** | **Comment** |
| --- | --- | --- | --- |
| Website | User Interface | User interface | Front end display for user interaction |
| API (Application Programming Interface) | service | Site user service | Read a book database |
| Cart | service | site user service | Stores the user’s books for checkout |
| LibraryCard | service | Site user service | The online library application's security and access control features |

# Initial list of functional specs

## Non-Member expectation

### Creating Account

* + **1.1** The system shall allow the user to create an account by storing UserID, Password, Date of Birth, First Name, Last name and Phone number. The system shall not allow the User to Create an account if the UserID chosen by the user already exists in the System's Database. Also, the system shall prevent the user from creating an account if the user's chosen password does not match the re-enter password field. The following areas must be filled in for the user to create an account successfully: First Name, Last Name, UserID, Password, Re-enter Password, Phone number, and Date of Birth**.**
  + 1.2 Stimulus/Response Sequence
    1. A user enters a UserID (same as email)
    2. A user enters a password.
    3. The user re-enters the password for confirmation.
    4. User shall enter their First and Last Name.
    5. User shall enter their date of birth.
    6. The user shall provide their phone number.
    7. The system shall check if UserID is available.
    8. The system shall validate the password.
    9. The system shall store the user's name, date of birth, and phone number.
    10. The system will have a button redirecting the user to the home page.
  + 1.3 Function requirement label
    1. REQ 1.1 Creating Account

### Browse the Library

* + **2.1** Users can browse the online Library by using the search bar tab on the homepage or picking from a popular list
  + 2.2 Stimulus/Response Sequence - Popular Picks
    1. User scrolls to the Popular Picks
    2. System shall have a filtered list by popularity for the User
  + **2.3** User will browse the system by entering a search by the title of the book, the author of the book or the genre.The system shall prevent the user from deleting books from the list that the user filters out. The user has to hit new search.
  + 2.4 Stimulus/Response Sequence - Search
    1. The user enters search criteria(title of book, author or genre) into the search
    2. The system shall supply the user with a list of books by title, author or genre
    3. The system shall have a button that will allow the user to return to the home page
  + 2.5 Function requirement label
    1. REQ 2.1 Browse by Popular Picks
    2. REQ 2.3 Browse by Search

### Cart

* + 3.1.
    1. The system has provided a checkout cart. The user can view the books and the author name, remove the selected books and view the price and the sub-total. The system shall allow the user to enter a debit/credit number, CVV, expiration date, name, and billing address
  + 3.2 Stimulus/Response Sequence
    1. The user clicks on the checkout button.
    2. The system shall provide the tax and total.
    3. The system shall have a button to allow the user to return to the book catalog.
    4. The system shall have an additional card button, where the user can add card information and billing address.
    5. The system will provide a return date for the book.
  + 3.3 Function requirement label
    1. REQ 3.1 views the selected item in the cart.

## Members expectations

### Edit Profile

* + 4.1
    1. Users shall be able to edit their profile by providing a name, date of birth, and phone. The system shall store the name, date of birth, and phone when the user clicks the save button. The system shall prevent any changes to the user's profile if any of the fields are left blank. The user must type the information in a valid format for the system to store the information.
  + 4.2 Stimulus/Responsive Sequence
    1. The user will navigate to "my account page".
    2. The user will click on the edit profile button.
    3. User will input their name and date of birth.
    4. The user shall click save
    5. The system shall store their name, date of birth, and phone.
    6. The system shall refresh the user profile with updated information as a confirmation.
    7. The system will have a button to redirect the user back to the home page at will.
  + 4.3 Function requirement label
    1. REQ 4.1 Edit Profile.

### Review

* + **5.1** the user will navigate to the review section and be able to write a review on specific books. The system shall prevent the user a text box.
  + 5.2 Stimulus/Responsive Sequence
    1. Users will use the browse function to navigate the review section.
    2. The user will scroll to the review section and write a 150 character review.
    3. The system shall store the user review.
    4. The system shall display the latest review stored.
    5. The system will have a more review button below the last review
  + 5.3 Function requirement label
    1. REQ 5.1 review

### Contact Customer Service

* + **6.1** Users will be allowed to contact via email. User shall type in their name, email, and their query. The system shall store these fields and submit them once the user clicks the send button. The system shall prevent the user from contacting developers directly. The user must submit a ticket, which shall be redirected to the correct personnel.
  + 6.2 Stimulus/Responsive Sequence
    1. The user shall navigate to Contact Us in the navigation bar.
    2. The user shall fill in their Name, Email and type up their query.
    3. The user shall then click the "Send" button underneath the Query box.
    4. The system shall store the information and submit it.
    5. The system shall state that the information was sent and thank the user.
    6. The system shall include a button redirecting the user to the Home page
  + 6.3 Function requirement label
    1. REQ 5.1 Contact Us

# 

# 

# List of non-functional specs

## Performance Requirements:

1. *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.
2. *Latency:*  The system will have a TTFB (Time to First Byte)an average of 800ms to 1800ms.
3. *Storage Utilization*: Storage utilization should be within 75 - 90% of the available storage provided at the time as to not get too close to using all storage and having a technical issue if more storage is needed for an emergency situation.
4. *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%.

## Security Requirements:

1. *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.
2. *Encryption*: The website will be encrypted as purchases and exchanges of valuable information, such as credit card numbers, will occur.
3. *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.
4. *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.
5. *Resource Utilization*: Resources such as the MySQL database on the phpMyAdmin server will be accessed through the PHP code using the usernames and passwords therein. All-access to the phpMyAdmin servers and their resources will be obtained with the usernames and passwords given. The system will utilize HTML and CCS3 as frameworks and will document the proper licenses and/or qualifications of each.

## Portability Requirements:

1. *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.
2. *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.

## Capacity Requirements:

1. The storage for our system will consist of the phpMyAdmin server holding our mySQL databases within an unknown capacity as well holding our files for the actual site.
2. 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.

## Reliability Requirements:

1. *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.
2. *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.
3. *Support*: There will be support availability by email that will filter to several assigned developers. They will be responsive within 24 hours.
4. 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.

# Competitive analysis

The analysis of competitors' websites will focus on six main features:

(homepage, design, navigation, search, content, usability) and three additional features (download, trending and add item). The competitive analysis will utilize a numerical scale (1=bad, 2=poor, 3=fair, 4=good, 5=outstanding) and consists of four web sites chosen for their focus on the open source model, books, and accessibility.

|  | **BiblioTech** | **Open Library** | **Project Gutenberg** | **Internet Archive** | **Free Library** |
| --- | --- | --- | --- | --- | --- |
| **Homepage:** | 5 | 4 | 2 | 2 | 4 |
| **Design:** | 4 | 4 | 1 | 3 | 4 |
| **Navigation:** | 5 | 4 | 2 | 4 | 3 |
| **Search:** | 4 | 4 | 4 | 3 | 3 |
| **Content:** | 4 | 5 | 4 | 5 | 1 |
| **Usability:** | 5 | 3 | 4 | 3 | 3 |
| **Download:** | 5 | 0 | 2 | 4 | 0 |
| **Trending/ Popular Content:** | 5 | 4 | 2 | 0 | 2 |
| **Add Item:** | 4 | 4 | 3 | 4 | 0 |
| **Mean:** | **4.5** | **3.6** | **2.7** | **3.1** | **2.2** |

**BiblioTech (4.5)**

BiblioTech's homepage is designed with simplicity and visual appeal, featuring striking photos and a prominent search bar that quickly directs users to the desired resources. The site's navigation is consistent and includes a helpful breadcrumb feature that enhances the user experience. The search function is both efficient and user-friendly, with autocomplete functionality that speeds up the search process. The site's content emphasizes book availability and user security, ensuring that users can access and use the resources with peace of mind. Additionally, BiblioTech offers a range of other useful features, including testimonials, a cart, and a search bar, all of which are seamlessly integrated into the site's design to enhance the user experience. Overall, BiblioTech is a well-designed, user-friendly platform that provides a variety of valuable features to users.

**Open Library (3.6)** [**https://openlibrary.org**](https://openlibrary.org)

Open Library features a content-filled homepage with many options for interested readers varying from trending books to sorted books by assorted categories. The search feature and the browsing option offer a lot and are quick to find the best books for a reader. Where the site lacks is on the design side, it has a very monotonous design and color scheme. The great content features are implemented in a very uninteresting way. Even with this, the site is quite user-friendly and offers most of the additional features besides the download option.

**Project Gutenberg (3.6)** [**https://www.gutenberg.org**](https://www.gutenberg.org)

The Project Gutenberg homepage and design is extremely lackluster, it offers a somewhat popping color scheme but overall design and homepage are extremely outdated. It is clear that the page has not been updated in a long time. The page does offer a good pool of content and search/usability features due to the simplicity of its implementation. It also offers all of the additional features, however, they are difficult to use or are not fully implemented like the download feature.

**Internet Archive (3.1)** [**https://archive.org**](https://archive.org)

The Internet Archive has many of the same features and issues as the Open Library. IT also has a lacking design with much to desire from the overall look and homepage but it is filled with content. The Internet Archive has a wide variety of content even compared to all of its competitors. At times it can be difficult to use or find features, but it does implement most of the additional features besides being able to see popular/trending titles.

**Free Library (2.2)** <https://www.freelibrary.org>

The Free Library homepage is very well put together and appealing with color. It also offers great usability and the ability to search for content and navigate through the site. It is clear though that this site is geared specifically towards education and so it only offers very limited content compared to other options. Due to this site not being as open source as the others, it does not implement the ability to contribute or download content, just the popular feature.

### **Planned advantages:**

BiblioTech is an exceptional platform that encompasses all the desirable features of its competitors. Its vast pool of content, vibrant design, and intuitive navigation make it a standout in the industry. BiblioTech's mission is to identify and address the shortcomings of other online libraries to provide a comprehensive platform that caters to all needs. It goes above and beyond by offering unique features such as content downloading, trending and popular title browsing, and user-generated content with moderation. The platform is meticulously designed to provide an exceptional user experience with a modern front-end and intuitive features.

# High-level system architecture

1. **FAU Lamp Server**: FAU Lamp Server, shall be hosting our software engineering development project.
2. **Discord**: Discord application is the means that the group shall be communicating with each other for the development of the project during the semester. All communication has to be through the Discord server created for the project.
3. **Bugzilla:** Bugzilla is a bug-tracking system used to track project features, issues, and bugs, the group will use this tool during the semester to track the development process for the final project
4. **Jira Software**: Jira software will be used to track the progress of the whole team. Individual and group tasks will be assigned along with a due date to keep track of the work.
5. **PhpMyAdmin**: PhpMyAdmin will be used to interact with MySQL database management. Users will be locating books from the database via the website input function and developers will manage the data (by either deleting) or adding items from and to the database.
6. **mySQL Database**: MySQL database is the database that is being used for the data that will be handled for the project.
7. **Visual Studio Code (VSC)**: Visual studio code is the IDE that the developers will use to create the code for the website. Languages to be used for the development of the website will be the following
   1. HyperText Mark-up Language (HTML) - will be the language that will allow the browser to display the website
   2. Cascading Style Sheets (CSS) - will be the language used to decorate the web pages
   3. Personal Home Page (PHP) - will be the language used for server-side functionality for the database and real-time edits in the tables.
   4. Javascript - will be the language used for client-side functionality that will be handled for User Interface(UI) needs to make the user experience enjoyable.
   5. Bootstrap(Optional) - We still need to decide if we will be using bootstrap or only CSS for decorating the website. Bootstrap will be the framework used for code construction for web pages within the group's project.
8. **Github**: Github is a tool that the developers will be using to work on the code. There will be a main branch managed by the project owner and the scrum master where all the developers will merge their updated code. The updated versions of the code will be merged along with comments to keep it organized.
9. **Browser Compatibility**: The system will be a web-based web app that operates on at least two of all of the major browsers, including Google Chrome, Mozilla Firefox, Safari, Opera, and Internet Explorer. It will have functionality in it that will provide alternatives if the browser does not have JavaScript installed on it.
10. **API(Google Books API or Open Library API ):**
    1. Google Books API: This API provides access to the Google Books database, which includes information on millions of books. With this API, developers can build a website that can search for books by title, author, ISBN, and other criteria, and retrieve information such as book descriptions, reviews, and ratings.
    2. Open Library API: This API provides access to the Open Library database, which contains information on over 20 million books. With this API, developers can build a website that searches for books by title, author, ISBN, and other criteria, and retrieves information such as book descriptions, cover images, and edition information.
11. **Express:** Express is a framework in node JS. Express is a popular web framework for Node.js that is used to build web applications and APIs. Developers will use its set of features and tools to help create the website.

# 

# 

# 

# 

# 

# 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