# LibTrack:

# the personal reading tracking platform

# Table of Contents

Individual Contributions Breakdown

Project Roles

Project Description and Use Cases

Risk Assessment

Class Diagrams

User Interface Design & Implementation

Design of Tests

Project Timeline

## Individual Contributions Breakdown

All members contributed to the project in a significant way.

## Project Roles

**Project Manager: Anton Perederii**

“As a Project Manager, my primary tasks were bridging together the business and the developer parts of the team and leading them in the process of developing the application. I organized, prepared the agenda, and led weekly meetings, distributed and supervised the completion of tasks and handled the situations when people faced obstacles with their tasks. I was also the main writer of this project report and I have set up all communication within the team and conducted basic HR management.“

**Product Owner: Zdravko Rashkov**

“As a product owner and business analyst, my primary tasks were to come up with the entire scope of the project for the proposal, what it would include, mostly on the business part, and also assess potential risks and problems that could arise and how to solve them. We interconnected the whole business part with the IT team so that everything could happen accordingly and at the same time. We also did competitor research in the business department to gain valuable information about the situation in the market and come up with ideas to make our product unique and differentiative.”

**Business Analyst: Iren Dimitrova**

“As a Business Analyst, UX/UI Designer, and Brand Manager on our team, I undertook a multi-layered role. My tasks included various aspects of the project, from conceptualization to execution and refinement. Most recently, I created the template for our project presentation and outlined its structure, ensuring coherence and clarity in our communication with the rest of the class. In the design phase, I conceptualized and created mock-ups and sketches to visualize the application’s layout and functionality Following feedback from the team, I refined the design and user interface, implementing polishing touches to enhance usability and address any identified issues. In addition to design tasks, together with my colleagues in the Business department of our project, I conducted Competitor Research to gain insights into the market landscape and identify opportunities for differentiation. This then helped immensely when I created the List of Requirements, which guided the development team in implementing features and functionalities essential for surpassing our competitors and presenting higher value to our users.”

**Developer: Boris Slavchev**

“As a developer on this project, my responsibilities encompassed various crucial aspects. In addition to designing and implementing the graphical user interface (GUI) using Java Swing components, I fostered effective collaboration and communication with team members, and the project manager, ensuring alignment with project goals and requirements. We utilized collaboration tools such as Git to manage code changes, track issues, and facilitate communication and coordination among team members. “

**Developer: Fady Abdalla**

“As a developer, I mainly worked on the backend part, especially making sure the pin feature was efficient with the help of our QA. In discussions about the project, I shared ideas about features, how to implement them, and how to solve any problems that came up. Plus, I made a UML diagram to show how everything in the system fits together visually.”

**Developer: Petar Zlatkovski**

“As a developer, in addition to gathering and organizing the bookstore's book data into an Excel sheet, I collaborated closely with Fady and Boris to implement crucial features within the software. My contributions extended beyond data management to actively participating in coding tasks, particularly focusing on refining the functionality related to the Pin and enhancing the graphical user interface (GUI). This involved iterative development and troubleshooting to ensure seamless integration of these components into the software system. Overall, my involvement in the project spanned from data organization to hands-on coding, demonstrating a multifaceted approach to problem-solving and project execution within the software engineering domain."

**Quality Control: Artem Kalmakov**

"As a QA, my responsibilities in this project included testing of functions written by developers, seeing UI, UX from the end-user perspective, and evaluating that to give feedback to developers and request changes. I did code reviewing and ensured the quality of the code written throughout the whole app, and participated in discussions related to project development, such as discussions on features that we needed to implement and possible hardships that we could be facing. I did directly assist developers in a few tasks, one of them being the connection of pin code functionality with the main component of the app, view of library and bookshelf."

**Cybersecurity: Martin Georgiev**

"As cybersecurity, I enhanced the security of the PIN code class by using AES encryption on the PIN code itself to prevent it from being accessed through the text file. I also created the BookPanel pop-up window along with its functionalities for displaying book information, finding and displaying their cover image, updating page progress, and moving between the library and bookshelf. I also assisted with finding descriptions for the books and some optimization for the book files so they are more uniform and readable by the program. I also created the .jar file for the program."

## Project Description and Use Cases

Our project is a personal reading tracking platform that utilizes Java. The purpose of the LibTrack application is to bring comfort of regular readers who have to switch from book to book constantly, be these textbooks for studying, self-education and non-fiction works, or fiction stories. With LibTrack, one can consider which books to read, add them to the shelf to mark that they own the book, rate them for personal purposes, and keep track of which page they stopped the last time they were reading the book.

Adopting a structured project management approach, we adhered to a collectively designated timeline to ensure timely delivery and effective collaboration among team members, who were split into business and development subgroups. The development process was guided by Agile methodologies, allowing us to adapt to changing requirements, implement new ideas throughout the development process, and prioritize user (tester) feedback throughout the project lifecycle.

Many frequent readers may find LibTrack a useful addition to their daily life, so the use cases of LibTrack include:

- **Tracking and updating the list of books read**
  For people, who want to know how many and which books they have read, they can do this in Libtrack.
- **Rating the books**
  The application allows users to rate the books they have, thus allowing them to have their own rated collection of books. This eliminates the need for any supplementary websites created for this purpose.
- **Tracking the book reading progress**
  Users will be able to track their progress on each of the books by simply setting the page on which they stopped. This eliminates the need for users to use bookmarks which can get damaged or lost.
- **Discovering new books**
  Using the “library” feature of LibTrack, users will be able to consider which new books to read and add them to their “shelves” directly in the app. This makes the user experience more comfortable and gives them one more option to discover new books.

## Security & Risks Analysis

1) **Protection of data and privacy:**

Risk: Unauthorized access to personal information of users from people, including names, passwords, and reading habits.

Solution: Implement robust authentication mechanisms and encrypt sensitive data both in transit and at rest.

### 2) Authentication and Authorization

Risk: Weak authentication and protection allowing unauthorized access to user accounts that do not belong to them.

Solution: Use strong, multi-factor authentication methods and ensure proper session management.

### 3) Application and Data Integrity

Risk: Data corruption or manipulation through SQL injection, Cross-Site Scripting (XSS), or other attack vectors.

Solution: Employ input validation, prepared statements for database access, and content security policies.

### 4) Service Availability

Risk: Denial-of-Service (DoS) attacks can render the app unavailable to users.

Solution: Implement rate limiting, use a Content Delivery Network (CDN), and have DDoS protection measures in place.

### 5) Data Backup and Recovery

Risk: Loss of data due to hardware failure, software bugs, or ransomware attacks.

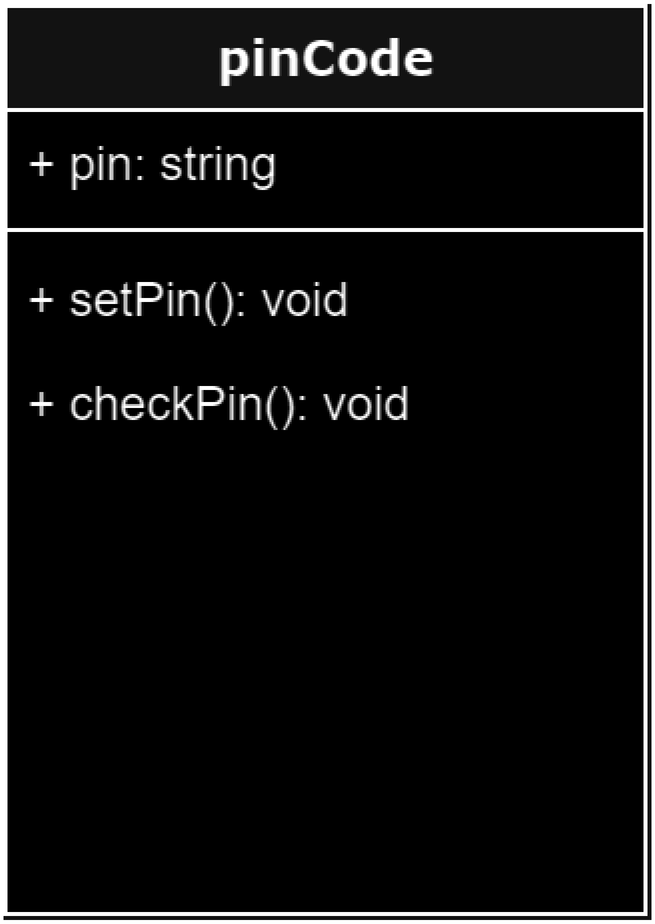
Solution: Regularly back up data using reliable storage solutions and have a disaster recovery plan.

### 6) Third-party Services and Libraries

Risk: Vulnerabilities in third-party services and libraries could compromise app security.

Solution: Keep all dependencies up to date and conduct regular security audits on third-party components.

# Class Diagrams

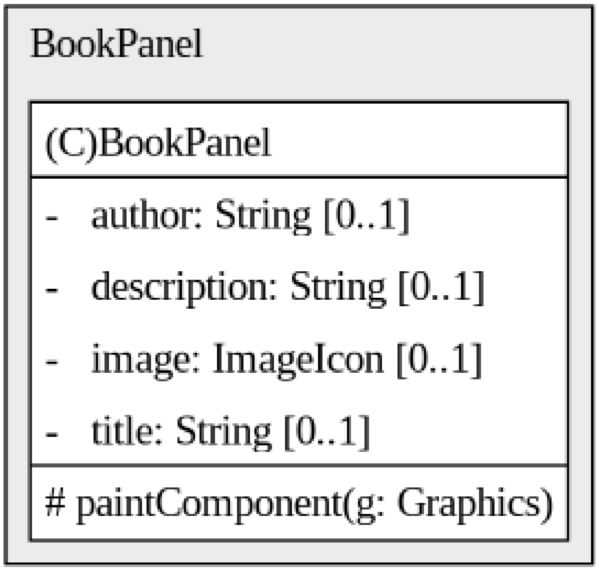


**pinCode**

+ pin: string

+ setPin(): void

+ checkPin(): void

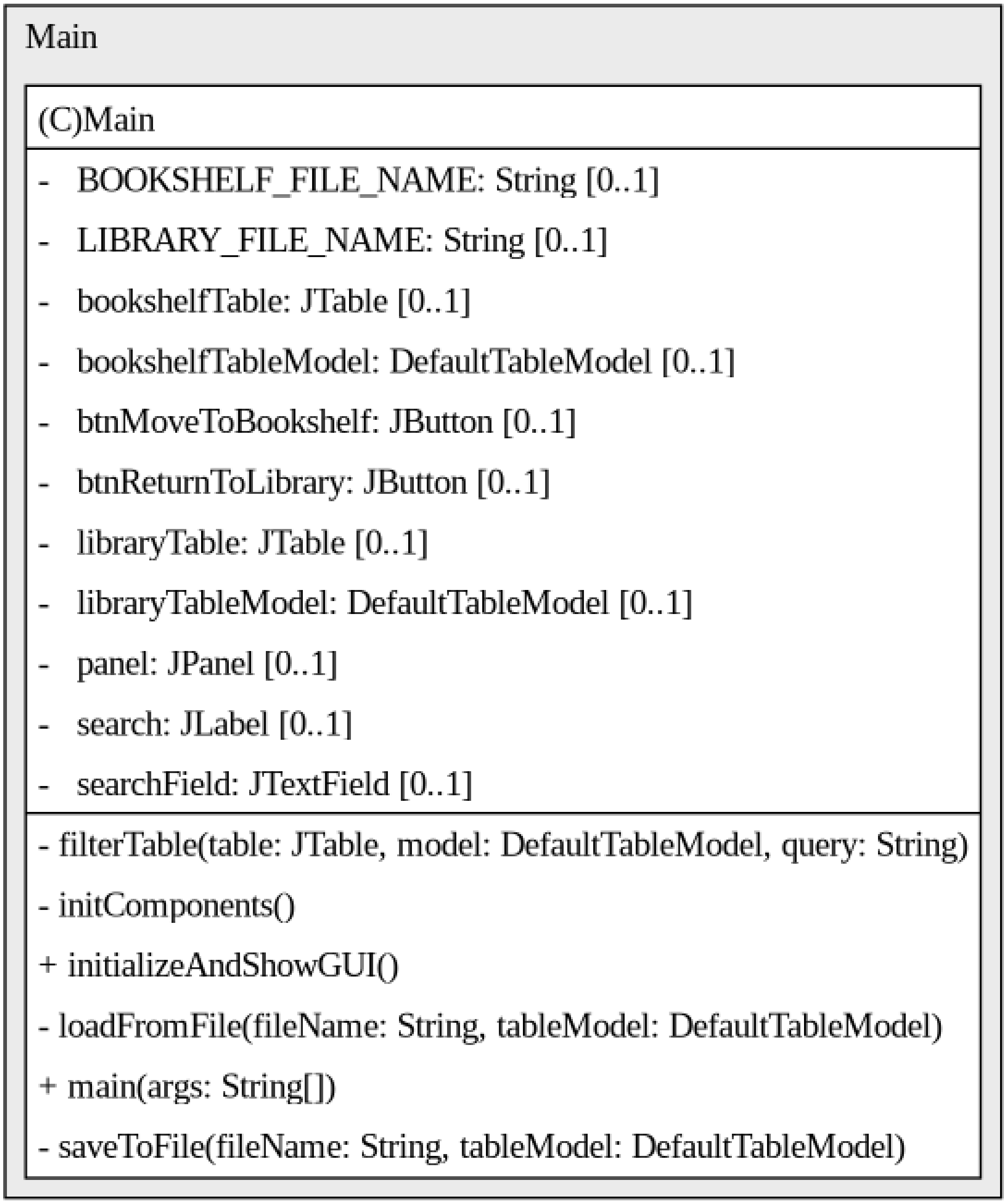


BookPanel

(C)BookPanel

- author: String [0..1]
- description: String [0..1]
- image: ImageIcon [0..1]
- title: String [0..1]

# paintComponent(g: Graphics)



Main

(C)Main

- BOOKSHELF_FILE_NAME: String [0..1]
- LIBRARY_FILE_NAME: String [0..1]
- bookshelfTable: JTable [0..1]
- bookshelfTableModel: DefaultTableModel [0..1]
- btnMoveToBookshelf: JButton [0..1]
- btnReturnToLibrary: JButton [0..1]
- libraryTable: JTable [0..1]
- libraryTableModel: DefaultTableModel [0..1]
- panel: JPanel [0..1]
- search: JLabel [0..1]
- searchField: JTextField [0..1]

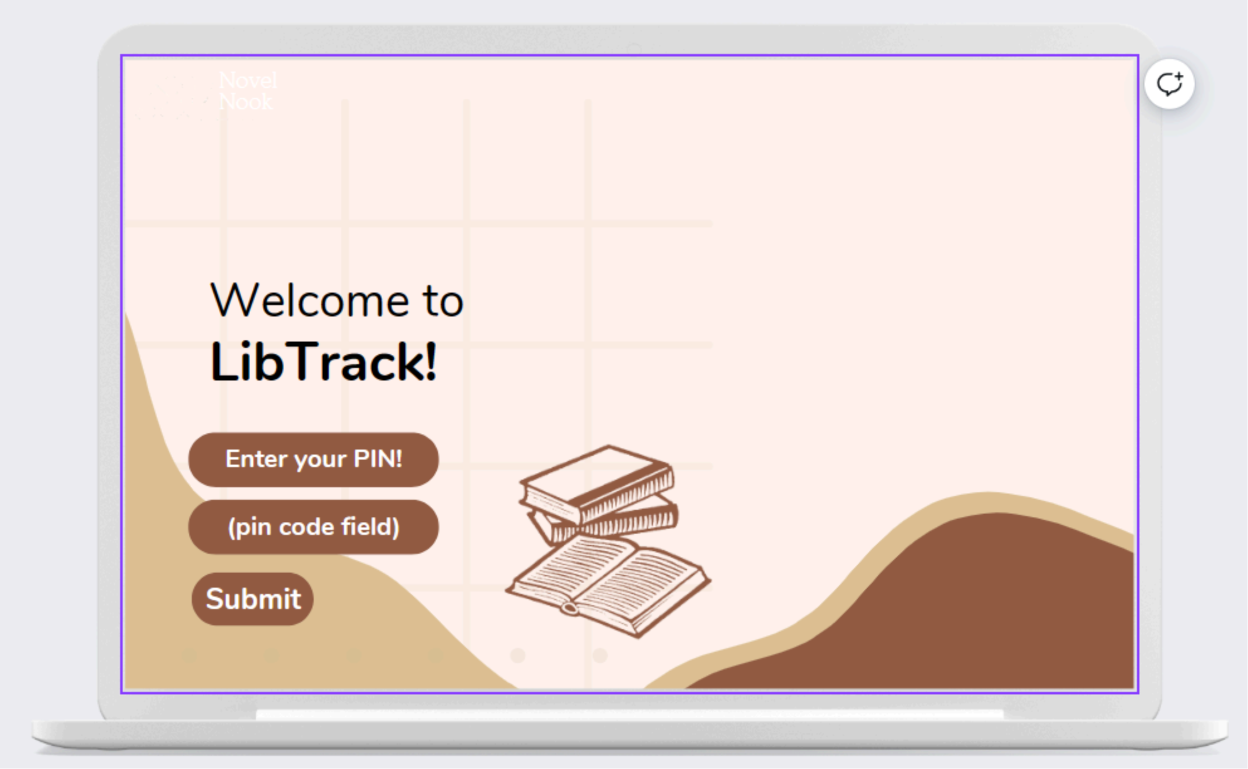
```
- filterTable(table: JTable, model: DefaultTableModel, query: String)
- initComponents()
+ initializeAndShowGUI()
- loadFromFile(fileName: String, tableModel: DefaultTableModel)
+ main(args: String[])
- saveToFile(fileName: String, tableModel: DefaultTableModel)
```

As you can see, the class structure is effective, efficient, and does not have any redundant components.

## User Interface Design & Implementation

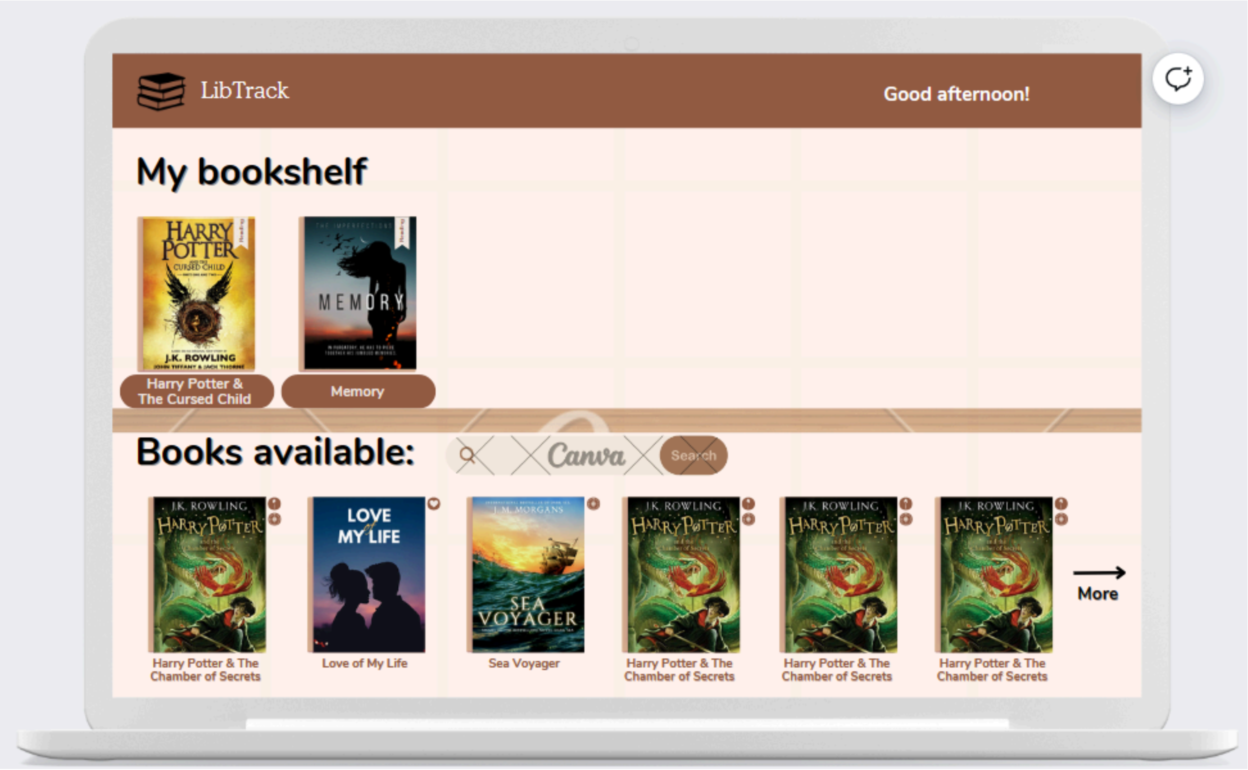
The user interface was developed with a vision to look cozy, user-friendly, and simple. Therefore, it features somewhat minimalistic design focused on warmer colors, and a minimal amount of visual effects. Therefore, the Business Analyst, who had also undertaken the role of the designer for the application and has sketched the following design of the application:

Welcome screen

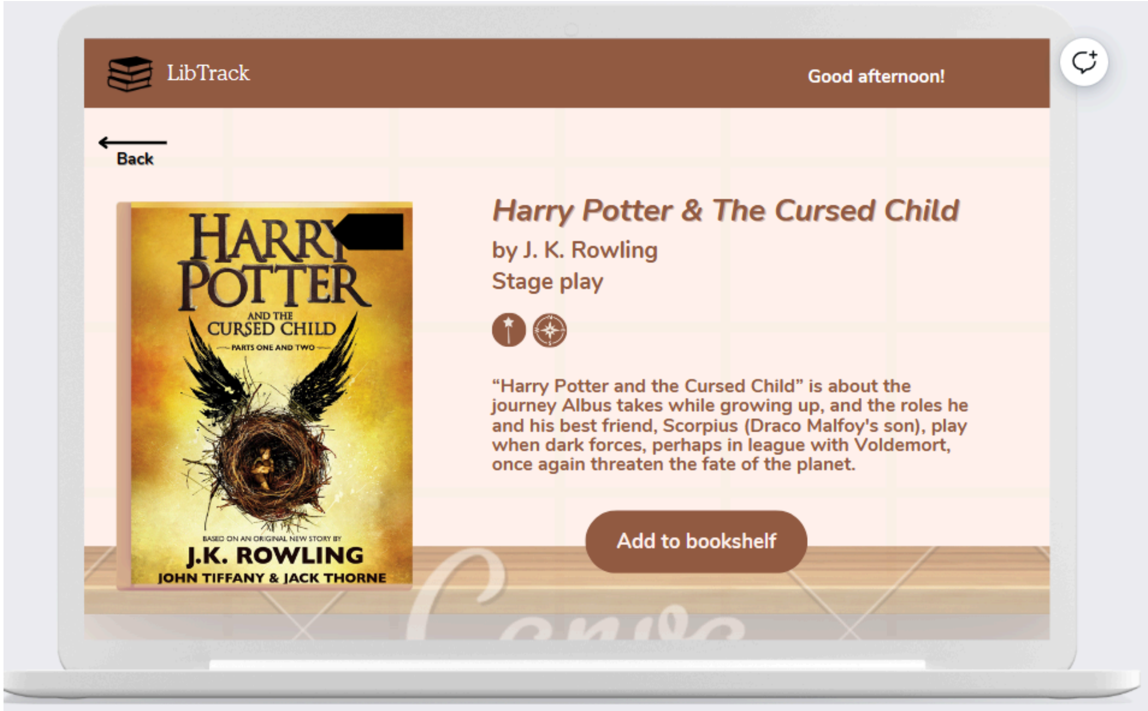


Welcome to
LibTrack!

Main screen

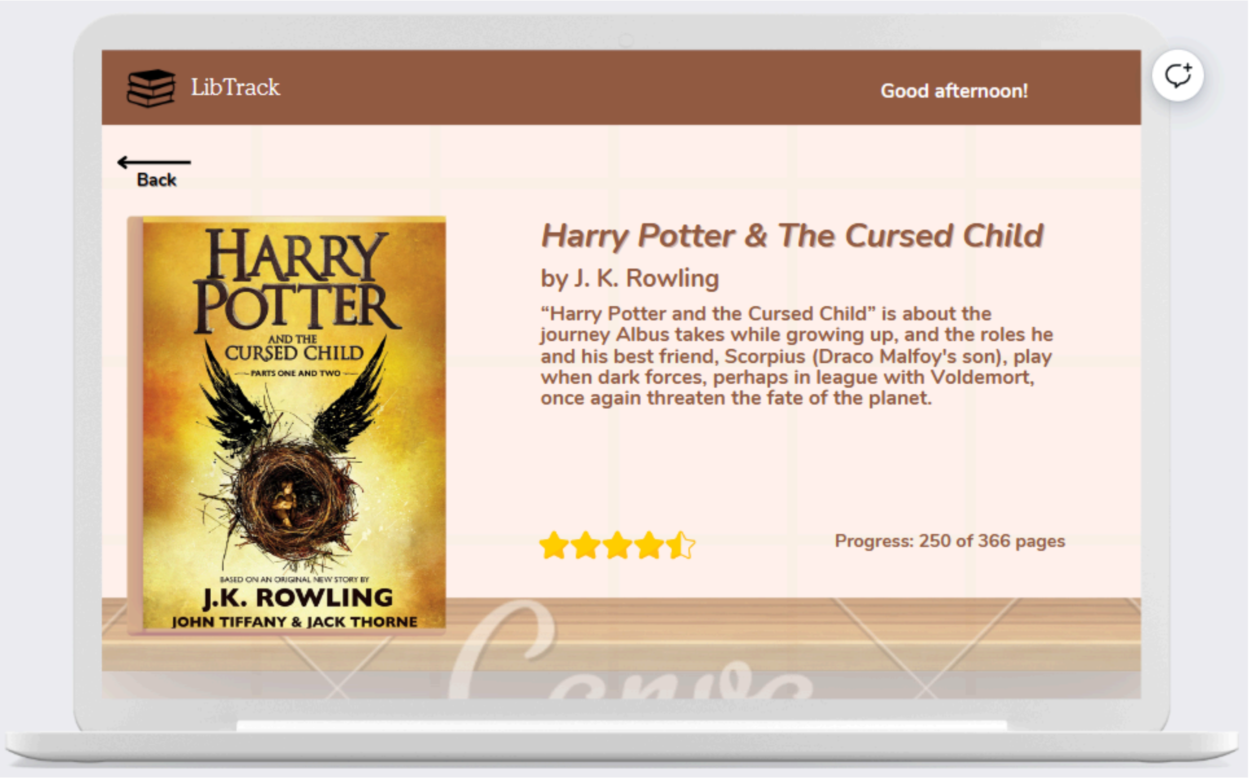

LibTrack
Good afternoon!
My bookshelf
Harry Potter & The Cursed Child
Memory
Books available:
Canva
Search
Harry Potter & The Chamber of Secrets
Love of My Life
Sea Voyager
Harry Potter & The Chamber of Secrets
Harry Potter & The Chamber of Secrets
Harry Potter & The Chamber of Secrets
More

Book-adding window



“Harry Potter and the Cursed Child” is about the journey Albus takes while growing up, and the roles he and his best friend, Scorpius (Draco Malfoy's son), play when dark forces, perhaps in league with Voldemort, once again threaten the fate of the planet.

Book on the bookshelf window


LibTrack
Good afternoon!
Back
HARRY POTTER AND THE CURSED CHILD
PARTS ONE AND TWO
BASED ON AN ORIGINAL NEW STORY BY
J.K. ROWLING
JOHN TIFFANY & JACK THORNE

*Harry Potter & The Cursed Child*

by J. K. Rowling

“Harry Potter and the Cursed Child” is about the journey Albus takes while growing up, and the roles he and his best friend, Scorpius (Draco Malfoy's son), play when dark forces, perhaps in league with Voldemort, once again threaten the fate of the planet.

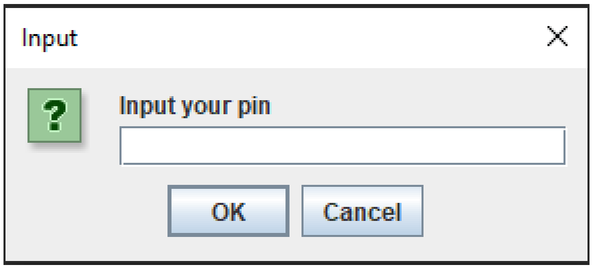
Progress: 250 of 366 pages

As one can see, the main screen was functionally split into what the user already has on the bookshelf and the “library”, where users can check out suggestions for what to read.

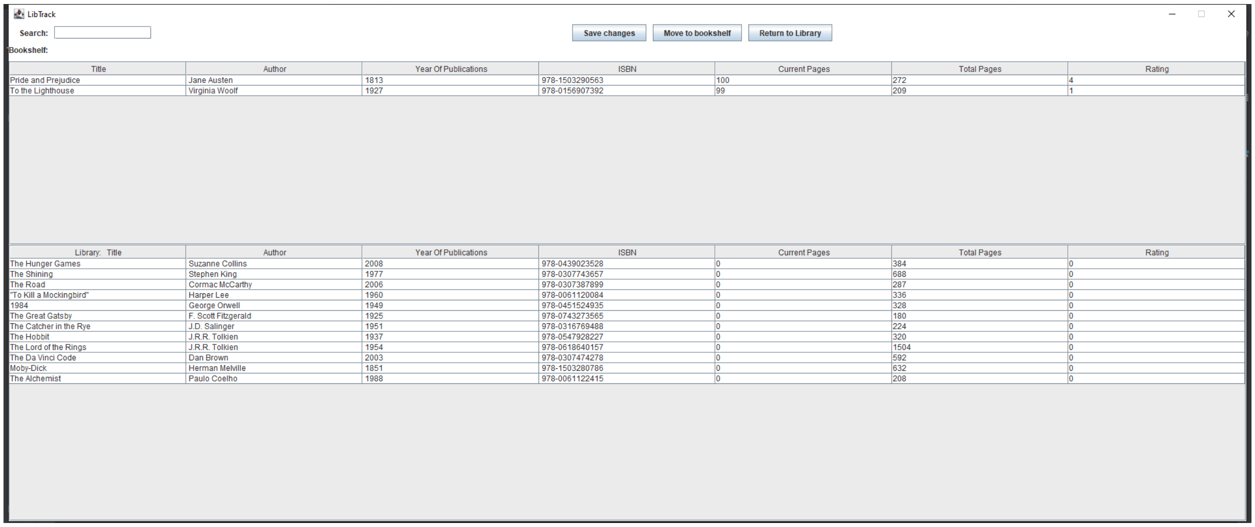
Unfortunately, as we progressed with developing the application, we had found out how troublesome it is to implement such level of UI into Java. We have tried raw Java, and we have tried using add-ons and frameworks like Maven, but nothing worked. After numerous hours of the development team were spent addressing this issue, a collective decision was undertaken to simplify the UI of the program while retaining its key features. Therefore, our team has concluded that alternative approaches are not possible and that we will stick to the current, simple, user-friendly, understandable and minimalistic UI.

Therefore, by the moment of finalizing this project, these are the screenshots of current version of the project. The older design is still a goal to be achieved, however, with the resources the team had at hand, we had to stick to this design for the time being.

## Pincode setting window



## Main screen

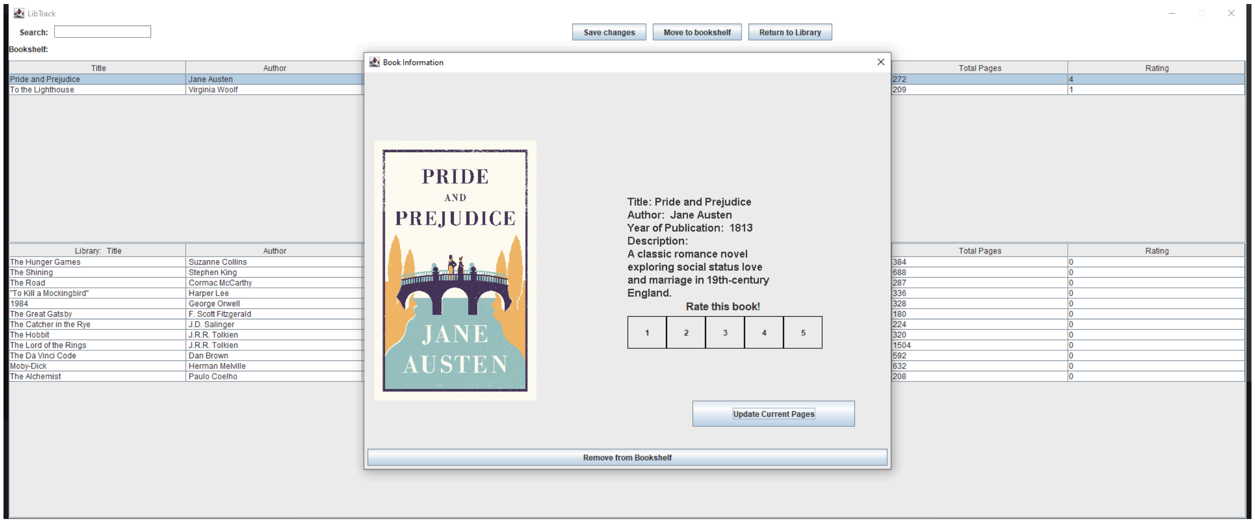


Bookshelf:

| Title | Author | Year Of Publications | ISBN | Current Pages | Total Pages | Rating |
|---|---|---|---|---|---|---|
| Pride and Prejudice | Jane Austen | 1813 | 978-1503290563 | 100 | 272 | 4 |
| To the Lighthouse | Virginia Woolf | 1927 | 978-0156907392 | 99 | 209 | 1 |

| Library: Title | Author | Year Of Publications | ISBN | Current Pages | Total Pages | Rating |
|---|---|---|---|---|---|---|
| The Hunger Games | Suzanne Collins | 2008 | 978-0439023528 | 0 | 384 | 0 |
| The Shining | Stephen King | 1977 | 978-0307743657 | 0 | 688 | 0 |
| The Road | Cormac McCarthy | 2006 | 978-0307387899 | 0 | 287 | 0 |
| "To Kill a Mockingbird" | Harper Lee | 1960 | 978-0061120084 | 0 | 336 | 0 |
| 1984 | George Orwell | 1949 | 978-0451524935 | 0 | 328 | 0 |
| The Great Gatsby | F. Scott Fitzgerald | 1925 | 978-0743273565 | 0 | 180 | 0 |
| The Catcher in the Rye | J.D. Salinger | 1951 | 978-0316769488 | 0 | 224 | 0 |
| The Hobbit | J.R.R. Tolkien | 1937 | 978-0547928227 | 0 | 320 | 0 |
| The Lord of the Rings | J.R.R. Tolkien | 1954 | 978-0618640157 | 0 | 1504 | 0 |
| The Da Vinci Code | Dan Brown | 2003 | 978-0307474278 | 0 | 592 | 0 |
| Moby-Dick | Herman Melville | 1851 | 978-1503280786 | 0 | 632 | 0 |
| The Alchemist | Paulo Coelho | 1988 | 978-0061122415 | 0 | 208 | 0 |

## Added book screen



Bookshelf:

| Title | Author | Total Pages | Rating |
|---|---|---|---|
| Pride and Prejudice | Jane Austen | 272 | 4 |
| To the Lighthouse | Virginia Woolf | 209 | 1 |

| Library: Title | Author | Total Pages | Rating |
|---|---|---|---|
| The Hunger Games | Suzanne Collins | 384 | 0 |
| The Shining | Stephen King | 688 | 0 |
| The Road | Cormac McCarthy | 287 | 0 |
| "To Kill a Mockingbird" | Harper Lee | 336 | 0 |
| 1984 | George Orwell | 328 | 0 |
| The Great Gatsby | F. Scott Fitzgerald | 180 | 0 |
| The Catcher in the Rye | J.D. Salinger | 224 | 0 |
| The Hobbit | J.R.R. Tolkien | 320 | 0 |
| The Lord of the Rings | J.R.R. Tolkien | 1504 | 0 |
| The Da Vinci Code | Dan Brown | 592 | 0 |
| Moby-Dick | Herman Melville | 632 | 0 |
| The Alchemist | Paulo Coelho | 208 | 0 |

Title: Pride and Prejudice
Author: Jane Austen
Year of Publication: 1813
Description:
A classic romance novel exploring social status love and marriage in 19th-century England.

# Design of Tests (Test Plan)

**Objective**: To ensure the functionality, usability, and security of the Book Tracker app developed in Java.

**Scope**: This test plan covers the core features of the app, including adding books, tracking reading progress, and pin code protection.

**Testing Methodology**: This plan will utilize a combination of black-box and white-box testing techniques.

**Testing Tools**: JUnit for unit testing (pin code functionality), user interactions for UI testing.

## Test Cases:

**1. Add Book To The Bookshelf Functionality:**

Test Case 1.1: Verify that a user can successfully add a book to the bookshelf.

Test Case 1.2: Verify that the app handles missing information appropriately (e.g., displays error messages or null for missing required fields).

Test Case 1.3: Verify that the added book appears correctly in "Your Bookshelf" with the provided information and image.

**2. Track Reading progress:**

Test Case 2.1: Verify that a user can input the number of pages read for a specific book.

Test Case 2.3: Verify that the reading progress is saved and persists even after closing and reopening the app.

Test Case 2.4: Verify that the user can mark a book as "finished" when reaching the total page count.

**[TO BE IMPLEMENTED IN THE FUTURE]**

Test Case 2.5: Verify that the user cannot set more pages than there is in the book.

**3. Pin Code Protection:**

Test Case 3.1: Verify that the app prompts the user to set a pin code on the first launch.

Test Case 3.2: Verify that the app requires the correct pin code to access the bookshelf.

Test Case 3.3: Verify that the app handles incorrect pin entries appropriately (e.g., displays an error message and limits attempts).

**4. Usability Testing:**

Test Case 4.1: Evaluate the overall user interface for clarity, intuitiveness, and ease of navigation.

Test Case 4.2: Assess the responsiveness of the app and ensure smooth transitions between screens.

Test Case 4.3: Verify that the app is visually appealing and uses appropriate fonts, colors, and image sizes.

**5. Performance Testing (Optional):**

Test Case 5.1: Measure the app's launch time and responsiveness under normal conditions.

Test Case 5.2: Evaluate the app's performance when handling a large number of books on the bookshelf.

**Pass/Fail Criteria:**

Most of the test cases have a defined expected result.

A test case passes if the actual result matches the expected result.

A test case fails if the actual result deviates from the expected result.

A test case is controversial if the result is not expected, but end user won’t have difficulties with this part or the code is ready for deployment, though some refactor in the future is needed.

Reporting:

As we can see, the overwhelming majority of the tests show that the actual result is consistent with the expected results. After this thorough testing, we see that, generally, the app is ready for the user. It does not crash when something unexpected is being done. Data is saved through a button, and unchangeable fields are protected from the user's inputs. The code is coherent.

## Project Timeline

Since the beginning of February, our team has been meeting regularly each week to strategize the project development. Initially, we outlined both functional and non-functional requirements, along with potential use cases and user narratives. After utilizing them, we were able to solidify our understanding of the application's anticipated functionality and visual layout, as well as the scope of the project in general. This led us into the initial development phases. We have debated on choosing the right methodology for development, language, platform, and method of tracking the task record. As a result, we stuck with the Agile approach, as we tried to implement gradual improvements into our application cycle after cycle.

Our team has decided to split into two groups: one oriented more towards programming-related tasks, and the other oriented towards the business and planning side of the project. Thus, we have started out by creating a GitHub repository, and creating outlines for the class structure of the program on the developer group side and creating project scope and risk assessment plans of the application on the business group side.

Our work was organized into 1-week sprints, with weekly Sunday meetings online on which we were taking the most important collective decisions (e.g. the general direction of the application, its name, etc), reported our progress on current tasks and distributed new ones. This helped us to keep a close track of the timely development of the project but also allowed us to carefully control the process of development of the application and react to unexpected situations, giving everyone involved the opportunity to approve or object to a decision taken by someone or a part of the system which has taken place during the sprint.

With the first half of the project being ready by mid-late March and the midterm season coming up, our team took a 3-week break after which we have conducted a recap and audit on the tasks that we have completed so far and then we carried on with the development. By the end of the semester, the biggest challenge that remained was to implement a good visual interface, which was not an easy task with Java and that has consumed countless hours of the development team. Therefore, we had no choice but to adapt and change our approach to the visual interface of the program, making it more simple while being understandable and user-friendly. This was the only change to the project scope we have committed to throughout the period of development. Other than that, the initial project scope turned out to be very strong, fitting, and accurate and served us well.

In terms of version control, we made extra effort to make sure that it was being handled in the best possible manner. For each significant task, we had a separate branch in GitHub, which was then merged together once ready and reviewed. The Quality Assurance Specialist in our team was overseeing the merges and assuring that the code in each branch was efficient and effective.

To finish the project on time, one of the additional measures undertaken was to introduce work-together sessions, where team members would sit together and work on their respective tasks and communicate immediately if the need arose. This significantly helped us accelerate the pace of development, and by April 28th the project was finally finished and successfully delivered. As a result, we had a convenient and user-friendly personal reading tracking platform ready for the first users to try it out. The project's particular strengths are a simple but user-friendly user interface, very low weight of the project itself, the implementation of security for users, and the possibility to track page records and rate user's favorite books.