Bookshelf Software

● ● ●

Harshith Nanda
Brian Mathew
Jane Luo
Kevin Tian
Eyal Grinberg
Pranay Mantramurti
Shreeya Patel

Objective

- Deliver to customers e-reading software that provides a center where readers can access, read, and store books.
- Enhance reading experience by ensuring seamless experience with their e-reader
- Can be used by businesses working with multiple documents and also casual readers who just want to see all the books they own

Cost estimation

In order to give the employees a good environment where they can work efficiently, we will need good computers for them, office space for them to work in, and provide food daily.

Computers for the engineers: $1,500 per person. $10,500 total.

Office space: $3,000 a month. $36,000 a year

Food: $2000 a month. $24,000 a year

Cost continued

We will use MongoDB for storing user data. This will just store the username, password, authentication, and the list of books each person has.

We will need 1500 GB of storage for this. This will cost $14.59/hr [1]. Assuming 100% uptime, this will cost us $127,808.4 per year

We will also use AWS to host the servers. We will use AWS Storage Gateway to be able to store resources like the books on their servers. Since we will be using around 50 TB per month, this will cost us $6055.81 per month to run [2]. This comes out to $72,669.72 per year.

| M200* | 1500 GB | 256 GB | 64 vCPUs | $14.59/hr |

Cost continued

We will need 7 people. The average salary per person is $83,000 per year [3]. This comes out to $581,000 per year.

Including the cost of all of our expenses, it would cost us around $851,978.12 dollars every year.

Project Timeline

Dates	Title	Time Needed	Team Members Needed
Nov 11-12	Project Planning and gathering requirements	2 days	7
Nov 15-Nov 19	Mock up the front end interface	5 days	3
Nov 15-Nov 19	Design the backend architecture	5 days	4
Nov 22-Dec 3	Implement the front end	10 days	3
Nov 22-Dec 3	Implement the back end	10 days	4
Dec 6 - Dec 10	Testing	5 days	7
Dec 13 - Dec 17	Deployment test program and receiving initial user feedback	5 days	7
Dec 20 - Dec 28 (Excluding holidays)	Updating product to better meet user guidance	4 days (excluding Holidays)	7
Dec 29 - Dec 31	Receive user Feedback	3 days	7
Jan 3- Jan 14	Update and deploy final product	10 days	7

Functional Requirements

- The user must be able to edit library
- The user must be able to search for books through a text query
- The user must be able to swipe to go to the next/previous page while reading.
- The user must be able to bookmark a page and go to the bookmarked page
- The user must be able to format book and ereader
- The user must be able to search for a word through a text query and be able to go to that word.
- The user must be able to extract chapters and go directly to them
- The user must be able to make, view, edit, and delete notations for a book

Non-Functional Requirements

Product requirement

The bookshelf software will be available to users whether they are offline or online. The software will be fast and will pre-load pages of the book and store it in memory. In case of system update, the update shall be done while the user is not on the app so reading will not be disrupted.
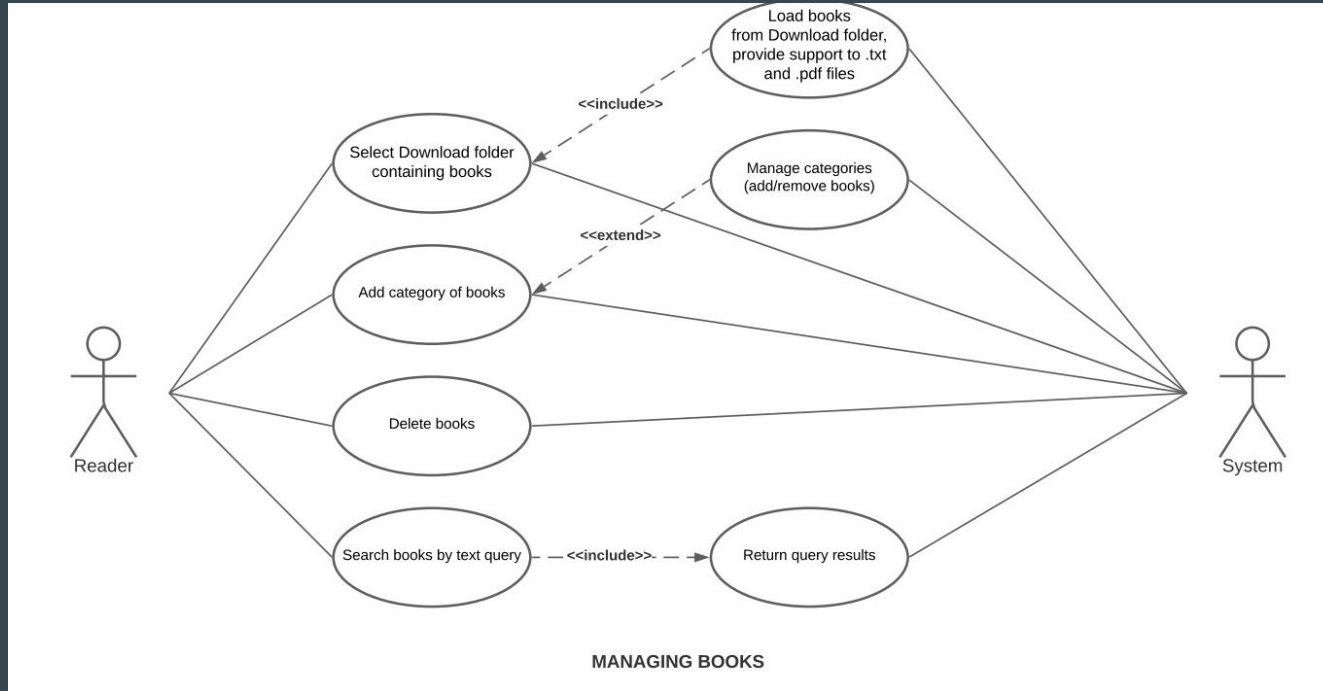
Organizational requirement

The user will be able to log into their account with a username and password, which will have all of their books saved.

External requirements

The bookshelf software will comply with all the rules stated in The Copyright Act of 1976

Use Case Diagram 1

Load books from Download folder, provide support to .txt and .pdf files

<<include>>

Select Download folder containing books

Manage categories (add/remove books)

<<extend>>

Add category of books

Delete books

Reader

System

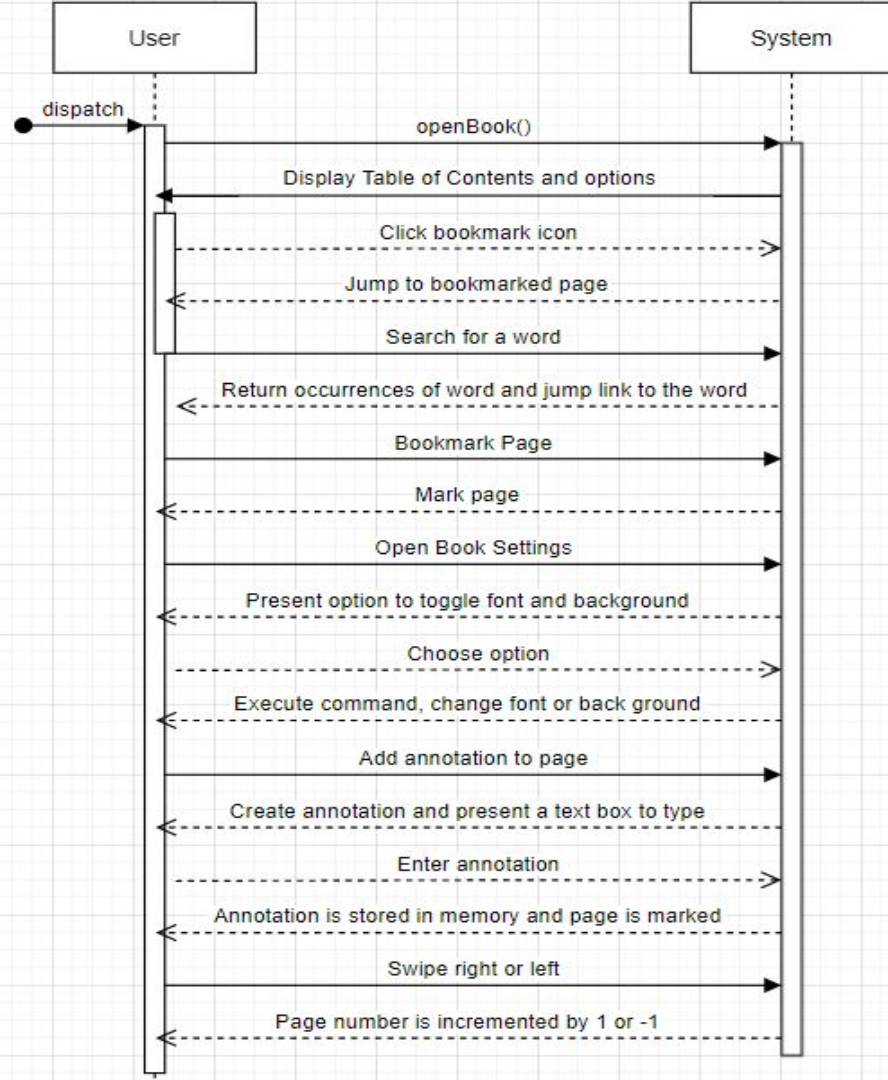Search books by text query

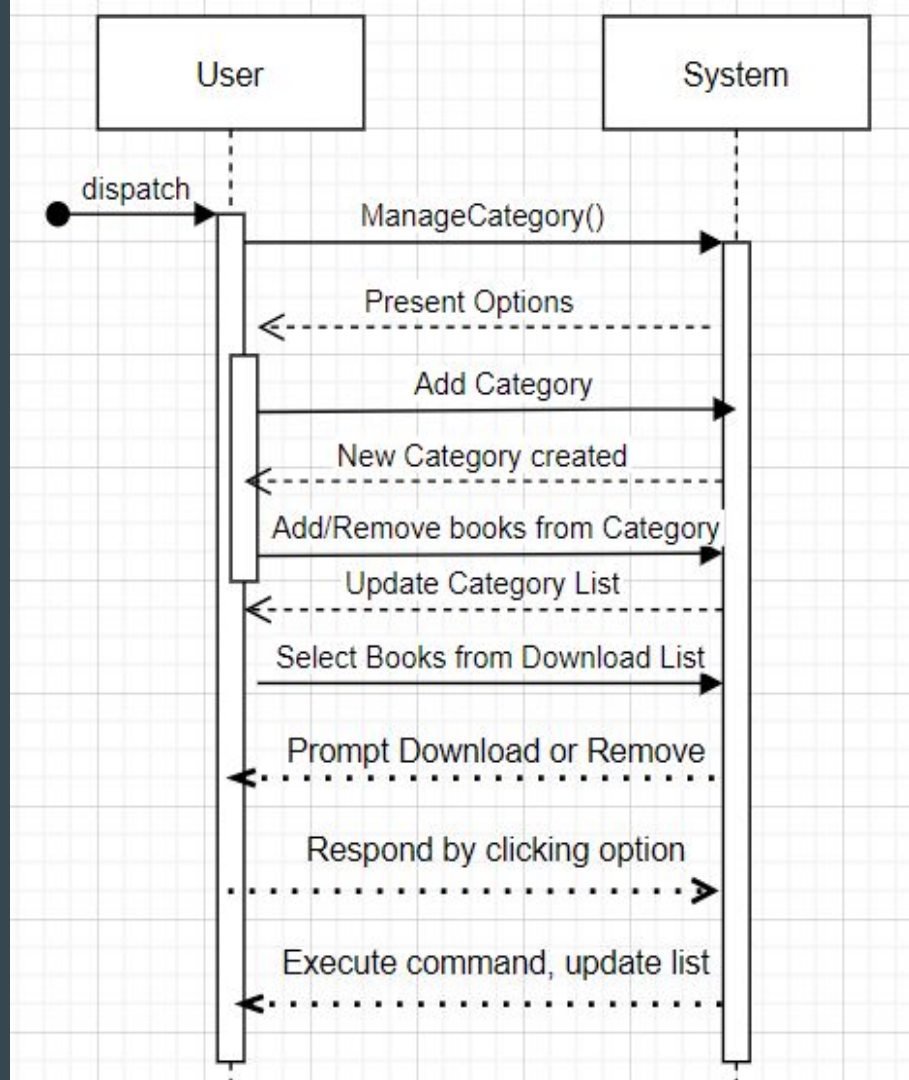<<include>>
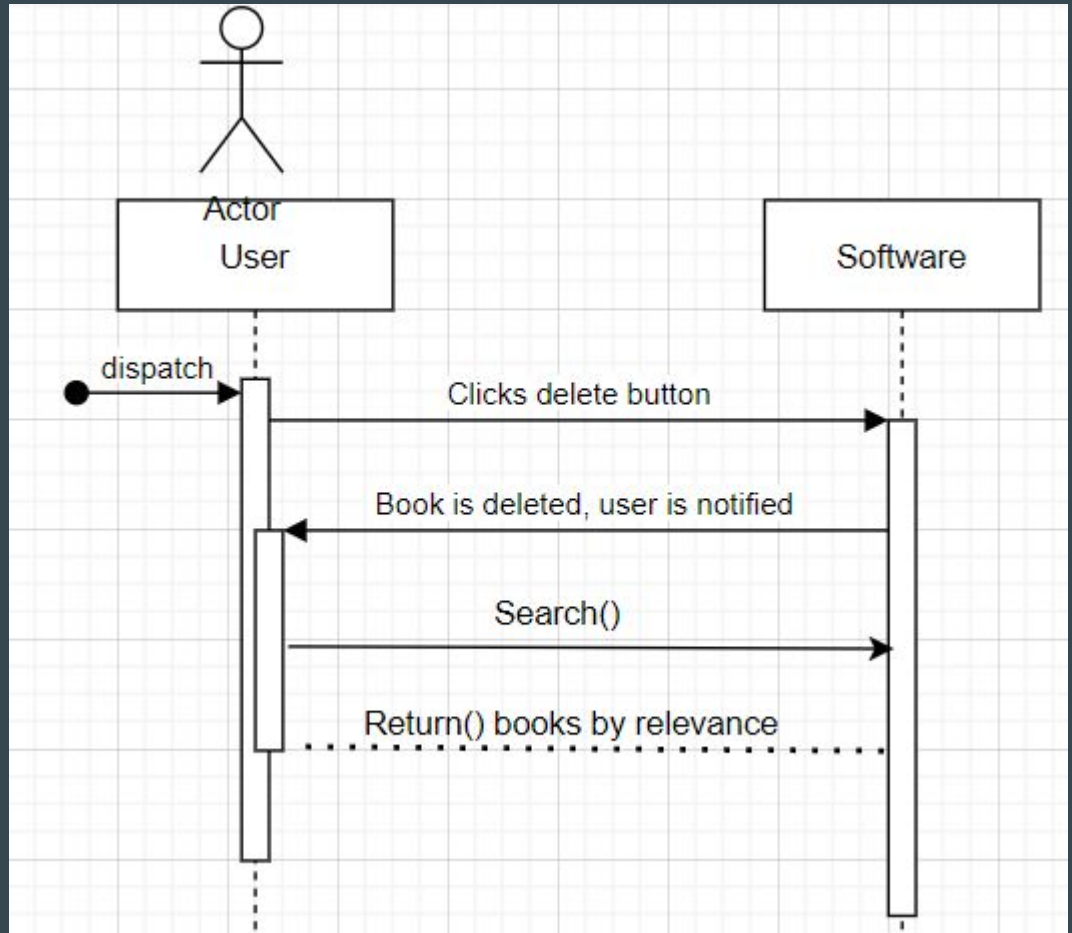
Return query results

MANAGING BOOKS

Use Case Diagram 2
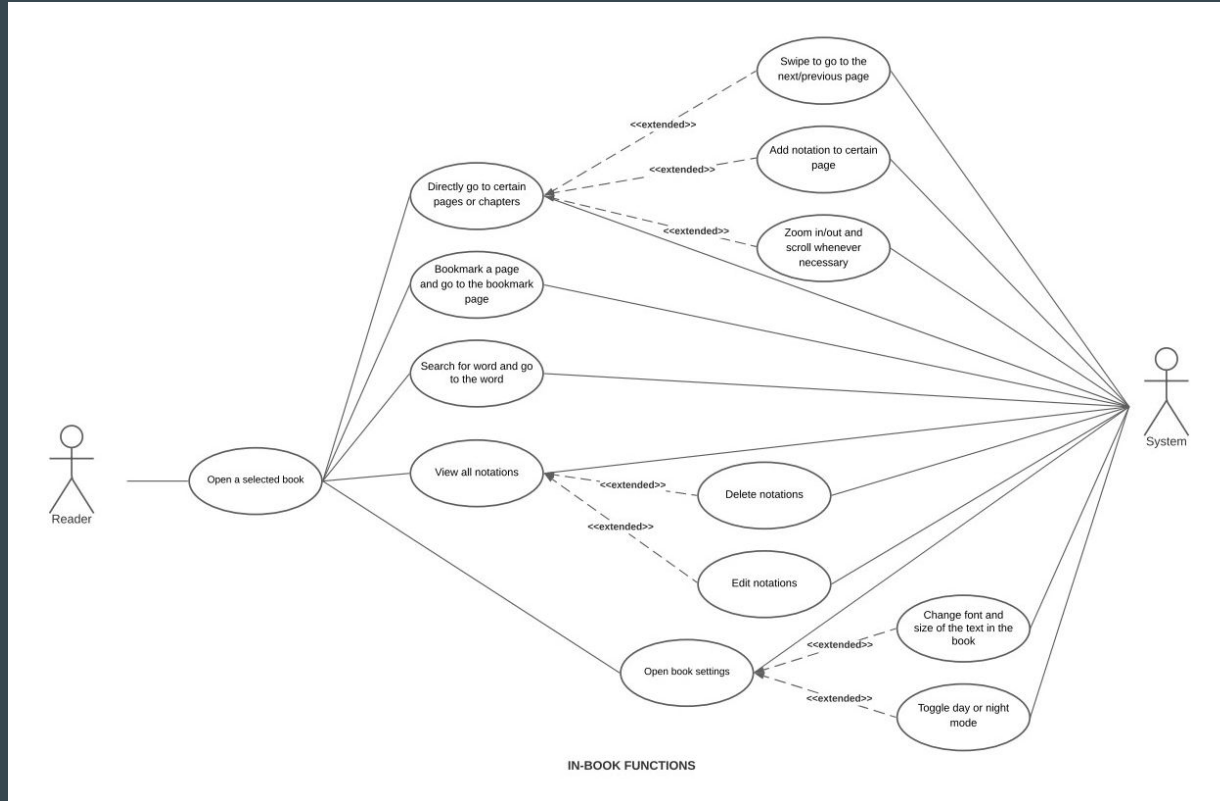
Sequence Diagram:
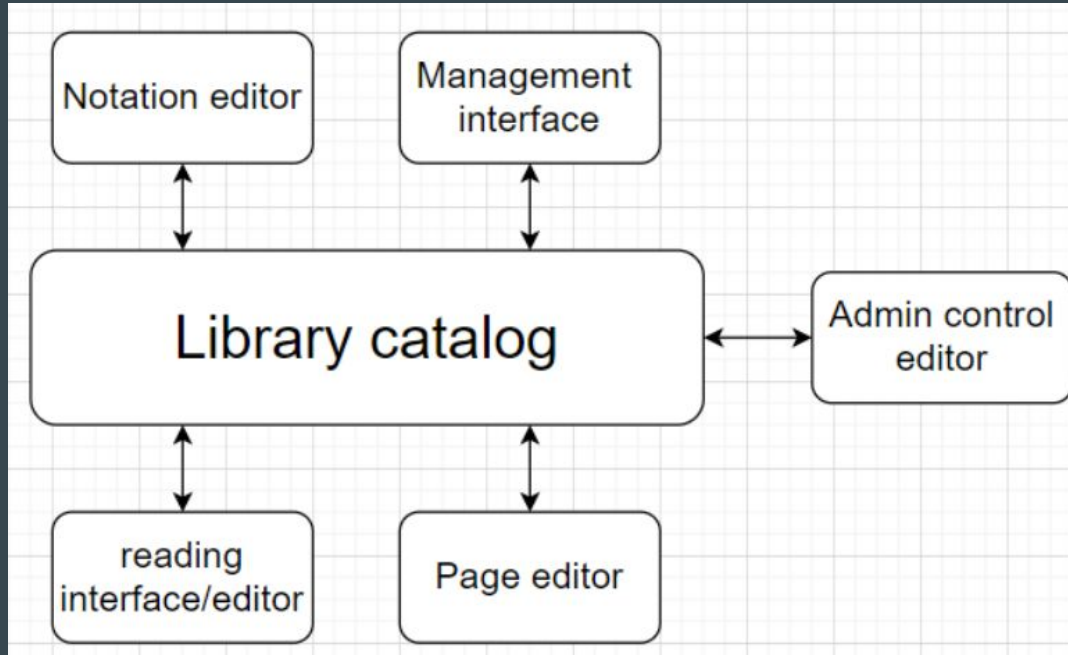Book Navigation

Sequence Diagram: Category Management

Sequence Diagram: Search & Delete

Class Diagram

EReader

+ CurrentBook : Book
- FontSize : int
- Font : string
- DayNightMode : bool

+ turnPageForward()
+ turnPageBack()
+ switchMode(mode : bool)
- getChapters()
+ goToChapter(chapNum : int)
+ changeFont(newFont : string)
+ changeFontSize(newSize : int)
+ setBookmark(pageNum : int)
+ goToBookmark()
- searchWord(word : string) : int[][]
+ goToWord(word : string)

is Read By

1..*

Book

+ BookName : string
- BookAuthor : string
- BookGenres : string[]
- BookNotations : Notation[]
+ Bookmark : int
+ BookContents : string[]

+ addNotation(contents : string, lineNum : int, pageNum : int)
+ deleteNotation(lineNum : int, pageNum : int)
+ updateNotation(newContents : string, oldContents : string, lineNum : int, pageNum : int)
+ getNotation(pageNum : int, lineNum : int) : string
+ checkGenre(genreName : string) : bool
+ checkAuthor(authorName : string) : bool
+ getGenre() : string[]
+ getAuthor() : string

1..*

Notation

+ NotationContents : string
+ NotationPageNum : int
+ NotationLineNum : int

Contains

*..1

Library

- BooksStored : Book[]

+ SearchByGenre(genre : string) : Book[]
+ SearchByAuthor(author : string) : Book[]
+ getBook(name : string) : Book

Architectural Design - Repository architecture

Conclusion

Overall, our work proceeded fluidly and according to our plan. A small change that we had to make was to our cost modeling technique, which was the function point method. We realized that we overestimated the productivity of a typical team and decided the original calculation of the project duration was unrealistic. After recalculating, we were able to come up with a reasonable project duration and schedule.

Citations

[1]   "MongoDB Pricing," *MongoDB*. [Online]. Available: https://www.mongodb.com/pricing. [Accessed Nov. 10, 2021].

[2]   "AWS Pricing," *AWS*. [Online]. Available: https://aws.amazon.com/pricing/. [Accessed Nov. 10, 2021].

[3]   "Average Software Engineer Salary," *Payscale*. [Online]. Available: https://www.payscale.com/research/US/Job=Software_Engineer/Salary. [Accessed Nov. 10, 2021].