

CS3354 Software Engineering

Final Project Deliverable 2

Bookshelf Software

Harshith Nanda

Brian Mathew

Jane Luo

Kevin Tian

Eyal Grinberg

Pranay Mantramurti

Shreeya Patel

1. Delegated Tasks:

- Harshith Nanda: Presentation planning and design
- Brian Mathew: Create Diagrams for deliverables.
- Jane Luo: Make another commit including a pdf/txt/doc file named “project_scope” (1.5) and create a test plan. Write conclusion and cite sources.
- Kevin Tian: Add all team members and the TA as collaborators (1.3) and evaluation of work. Estimation of cost, functional and nonfunctional requirements, and JUnit testing.
- Eyal Grinberg: Create GitHub repository named 3345-group9 and project scheduling, cost, effort, and pricing estimation.
- Pranay Mantramurti: Make the first commit to the repository; a Readme file (1.4) and work on presentation.
- Shreeya Patel: Comparing our project to other similar designs

2. Project 1 Deliverable Content

1. Project Feedback

Group Number: 9

Group Members: Harshith Nanda, Brian Mathew, Jane Luo, Kevin Tian, Eyal Grinberg, and Pranay Mantramurti, Shreeya Patel

Project Title: Bookshelf software

Project Purpose/Motivation: The purpose of this project is to deliver to our customers an e-reading software that will provide a center where readers can access, read, and store their books. We want to enhance people’s reading experience by ensuring they have a seamless experience with their e-reader. Currently, there is no good e-reading software that will store all the pdfs and e-books in a clear interface. This will be used by businesses working with multiple documents and also casual readers who just want to see all the books they own.

Delegated Tasks:

- Harshith Nanda: Presentation planning and design
- Brian Mathew: Create Diagrams for deliverables.
- Jane Luo: Make another commit including a pdf/txt/doc file named “project_scope” (1.5) and create a test plan.
- Kevin Tian: Add all team members and the TA as collaborators (1.3) and evaluation of work.
- Eyal Grinberg: Create GitHub repository named 3345-group9 and project scheduling, cost, effort, and pricing estimation.
- Pranay Mantramurti: Make the first commit to the repository; a Readme file (1.4) and make the deliverable 1 slides for presentation.
- Shreeya Patel: Comparing our project to other similar designs

Scholar Paper: No

Designated Submitter: Harshith Nanda

Library App:

- 4 Book Shelf Software
 - 4.1 Book management
 - 4.1.1 Load books from Download folder, provide support to .txt and .pdf

- 4.1.2 Delete books
- 4.1.3 Add category of books
- 4.1.4 Manage categories (add/remove books)
- 4.1.5 Search books by text query
- 4.2 Book reading
 - 4.2.1 Swipe to go to the next/previous page
 - 4.2.2 Bookmark a page and go to the bookmark page (optional for .pdf)
 - 4.2.3 Day & night mode (optional for .pdf)
 - 4.2.4 Search for word and go to the word (optional for .pdf)
 - 4.2.5 Change font and size of the text in the book (optional)
 - 4.2.6 Extract chapters and directly go to certain chapters (optional)
- 4.3 Book notations (optional)
 - 4.3.1 Add notation to certain page
 - 4.3.2 View notations on page with notations
 - 4.3.3 Edit notations
 - 4.3.4 View all notations for a book
 - 4.3.5 Delete notations
- 4.4 Other
 - 4.4.1 Zoom in/out and scroll whenever necessary

Final Project Proposal

Good choice for a topic! Reading is always cool and it is great to take advantage of technology so that readers can share their experiences/suggestions on books with each other.

It is great to see a detailed break down of the tasks you have worked on already. Good job.

In the final report, please make sure to include comparison with similar applications -if any-, make sure that you differentiate your design from those, and explicitly specify how.

Fair delegation of tasks.

Please share this feedback with your group members.

You are good to go. Have fun with the project and hope everyone enjoys the collaboration.

The feedback was fairly approving and for the most part, only offered suggestions about what to include in our final project. We will continue with our original plan and we will also be sure to include a comparison to similar software in our final report.

2. Repository URL: <https://github.com/kxt190007/3354-Group9>

3. Delegation of Tasks

1.2. Create a GitHub repository named 3354-Group9 - Kevin

1.3. Add all team members, and the TA - Kevin

1.4. Make the first commit to the repository - Pranay

1.5. Make another commit including a pdf named "project_scope" - Brian

Software process model - Brian
Software requirements - Kevin
Use case diagram - Jane
Sequence diagram - Harshith
Class diagram - Pranay
Architectural design - Eyal

4. We plan to use the agile software process model so that we can take advantage of the continual improvement that it allows. Because none of us are experts in running a library, we feel it is better for us to get a prototype out there as quick as possible and allow people who specialize in this subject to critique it directly. This will also allow us to take advantage of everyone on our team's individual skills because agile development allows us to consider that when assigning tasks. We plan to use scrum meetings to keep the team updated on all new developments and talk about potential upcoming issues.

5. Software requirements

- Functional requirements

The functional requirements for the bookshelf software are that

1. The user must be able to load downloaded books from a folder and the software must provide support for .txt and .pdf file types
2. The user must be able to delete books
3. The user must be able to add a category of books
4. The user must be able to add or remove books from a category
5. The user must be able to search for books through a text query
6. The user must be able to swipe to go to the next/previous page while reading.
7. The user must be able to bookmark a page and go to the bookmarked page
8. The software must contain a day and night mode that the user can toggle between
9. The user must be able to search for a word through a text query and be able to go to that word.
10. The user must be able to change the font and size of the text in the book
11. The user must be able to extract chapters and go directly to them
12. The user must be able to add a notation to a page in the book
13. The user must be able to view all the notations on a page that has notations
14. The user must be able to edit notations for a book
15. The user must be able to delete notations

16. The user must be able to zoom in and out and scroll whenever necessary
17. Each user must have a username that will help the system find them and store their books
18. Everytime the user syncs with their account, their downloaded books will be synced to the downloads folder

- Non-functional requirements

The non-functional requirements for the bookshelf software are

1. 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.

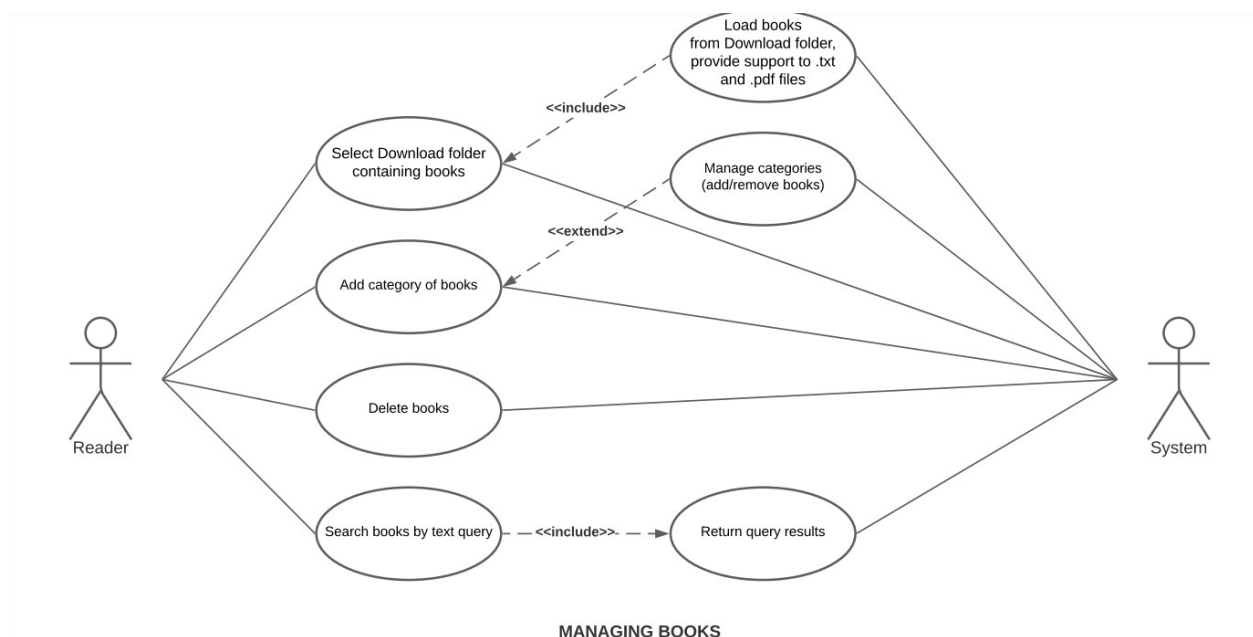
2. Organizational requirement

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

3. External requirements

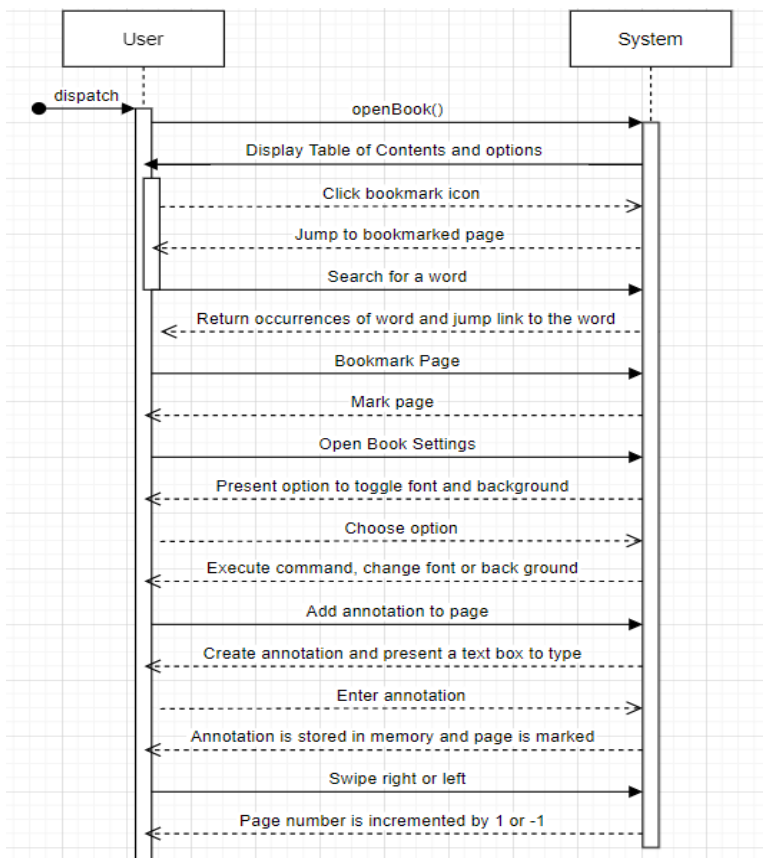
The bookshelf software will comply with all the rules stated in The Copyright Act of 1976

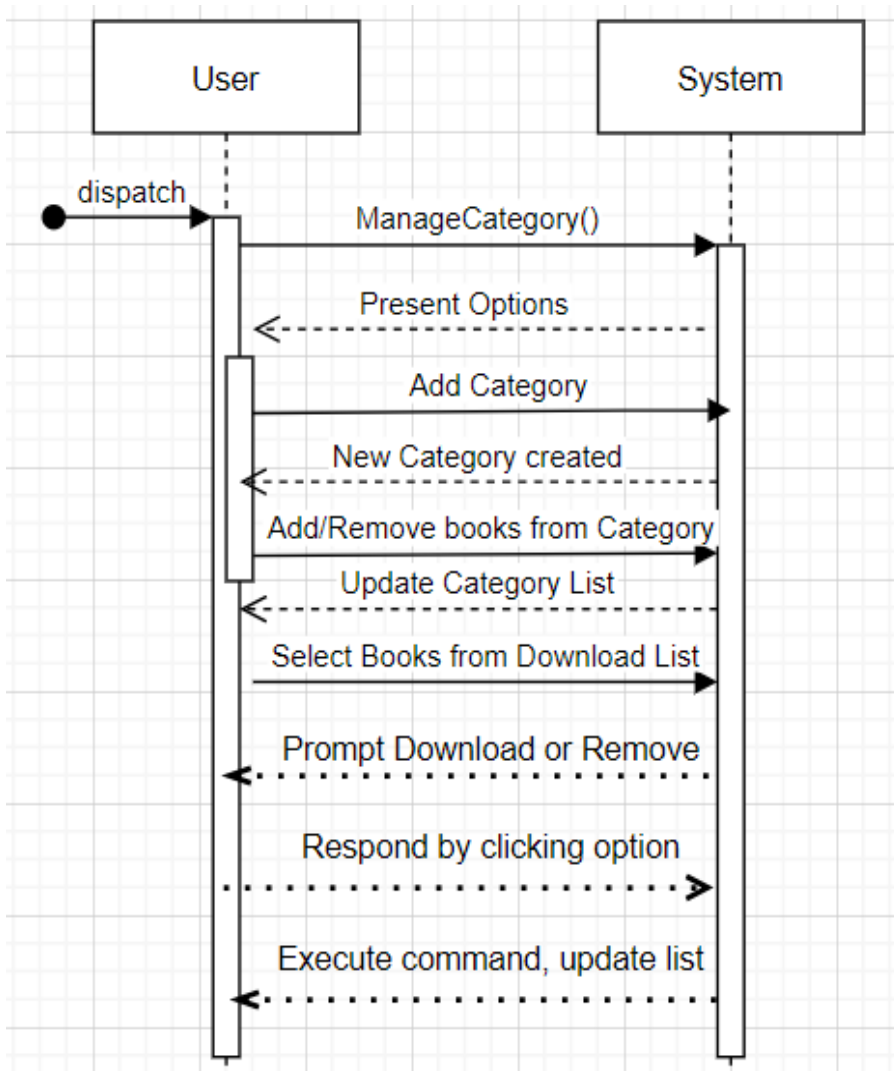
6. Use case diagram - Jane



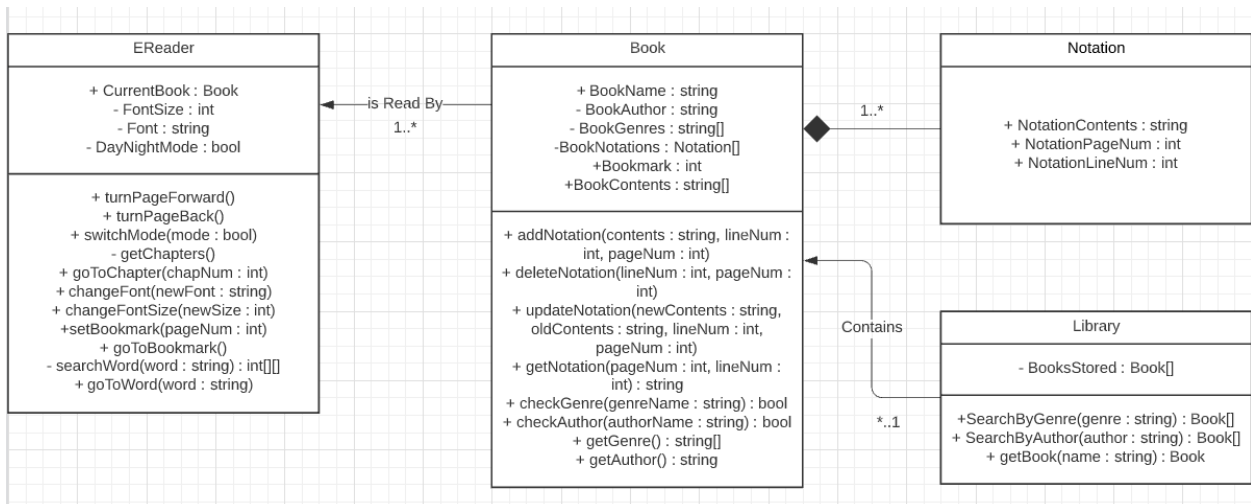


7. Sequence diagram - Harshith Nanda

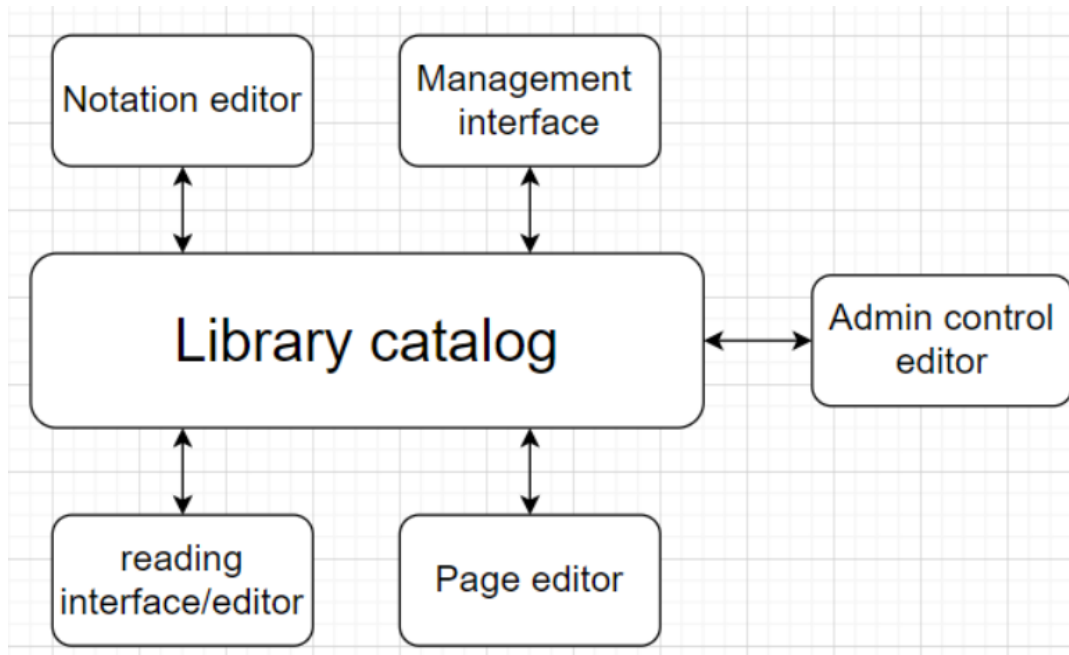




8. Class diagram - Pranay



9. Architectural design - [9.3 Repository architecture pattern] - Eyal



3. We will be choosing Option 2: Prerecord your captioned presentations.

4. Project Scheduling, Cost, Effort and Pricing Estimation, Project duration and staffing:

4.1. Schedule

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 and receiving initial client feedback	5 days	7
Dec 20 - Dec 28 (Excluding holidays)	Updating product to better meet client specifications	4 days (excluding Holidays)	7
Dec 29 - Dec 31	Receive Client Feedback	3 days	7
Jan 3- Jan 14	Update deploy final product	10 days	7

Total time for project completion is just under 8 weeks.

The number of working hours per day is 8.

We do not plan on using weekend time for this project. All work will be completed during the day.

4.2. Estimation

We decided to use **Function Point (FP)**.

a. Assumptions

- i. Assume team productivity is 60 function points per person week
- ii. Assume team of 6

b. Determine function category count.

- i. User Input :find book, find page login, logout, load books, delete books, add books, remove books from category, previous or next page, bookmark, toggle night mode, toggle font and size, extract a chapter, add notation, edit notation, delete notation, zoom = 17
- ii. User Output : return book, return page, return notation, return add book success or failure, delete book success, delete notation success, login success, book downloaded success, logout success, create account success, extract chapter success, edit success = 12
- iii. User Queries : search for notation, search for book query, search for word query = 3
- iv. Data files and relational tables : Book Library, Notation data, bookmark data, book data, account data, folder data, authentication data = 7
- v. External interfaces : Downloads Folder, Book information = 2

c. Determine complexity.

- i. Simple

d. Compute gross function point (GFP).

	Function Category	Count	Complexity	Complexity	Complexity	Count x Complexity
			Simple	Average	Complex	
1	Number of user input	17	3	4	6	68
2	Number of user output	12	4	5	7	60
3	Number of user queries	3	3	4	6	9
4	Number of data files and relational tables	7	7	10	15	105
5	Number of External interfaces	2	5	7	10	10
					GFP	252

- e. Determine processing complexity (PC)
 - i. $P1 = 5, P12 = 5, P14 = 5, PC4 = 5$ and $PC11 = 5$
 - ii. $P2\ 3\ 4\ 7 = 1$
 - iii. Remaining are average
 - iv. $PC = (5 * 5) + (4 * 1) + (5 * 3) = 44$
- f. Compute processing complexity adjustment (PCA)
 - i. $PCA = 0.65 + 0.01 (44) = 1.09$
- g. Compute function point (FP) using the formula:
 - i. $FP = GFP \times PCA = 274.68$
- h. Estimated Effort
 - i. $E = FP/productivity = 274.68/5 = 54.936 \sim 55$ person-week
- i. Project duration
 - i. $D = E / \text{team size} = 55 / 7 = 7.848$ weeks, ~ 8 weeks

4.3. Estimated cost of hardware products

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

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

Food: \$2,000 a month, \$24,000 a year

Total: to operate for a year, we would need \$70,500.

4.4. Estimated cost of software products (such as licensed software, etc.)

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

M200*	1500 GB	256 GB	64 vCPUs	\$14.59/hr
-------	---------	--------	----------	------------

\$14.59/hr [1]. Assuming 100% uptime, this will cost us \$127,808.4 per year

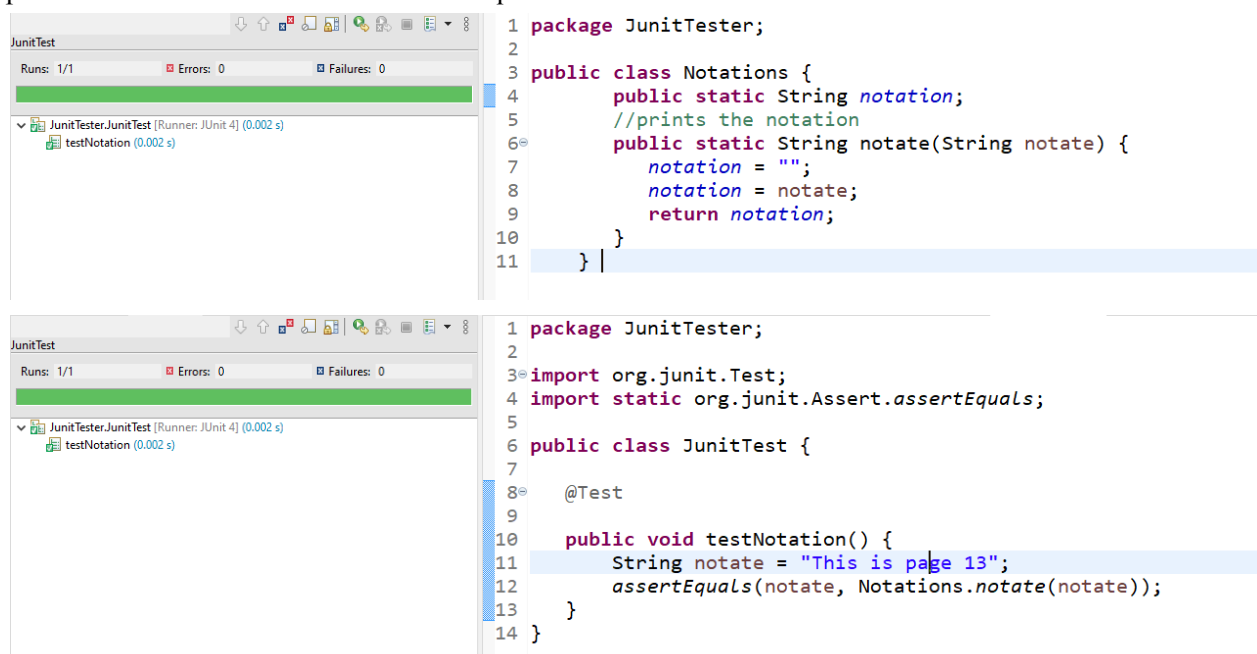
We will also use AWS to host the servers. We will use AWS Storage Gateway to be able to store resources like the books on their servers. Since we will be using around 50 TB per month, this will cost us \$6055.81 per month to run [2]. This comes out to \$72,669.72 per year.

4.5. Estimated cost of personnel (number of people to code the end product, training cost after installation)

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

5. Testing

We are testing the notation function of our software. First we define a string as notate and then we pass it into the function notate in class Notations. Then, we set the static string “notation” equal to the string we passed in. Then we check if the two are equal.



```
1 package JUnitTester;
2
3 public class Notations {
4     public static String notation;
5     //prints the notation
6     public static String notate(String notate) {
7         notation = "";
8         notation = notate;
9         return notation;
10    }
11 }

1 package JUnitTester;
2
3 import org.junit.Test;
4 import static org.junit.Assert.assertEquals;
5
6 public class JUnitTest {
7
8     @Test
9
10    public void testNotation() {
11        String notate = "This is page 13";
12        assertEquals(notate, Notations.notate(notate));
13    }
14 }
```

6. Comparison

Our bookshelf software goes under the category of e-reading software applications. Our software’s objective is to create a place where readers can easily access, read, and store their books. An application that has similar features to ours is the Kindle. Some main features that both of the applications have in common is that they both include automatic bookmarks, adding notations, searching for words within books, ability to delete and categorize books, page-turning function, and adjustable text size. They both aim to make reading online easier and more accessible to their readers. The big difference, however, is our bookshelf software aims to store all the pdfs and e-books in a clear interface, while the kindle is not

able to provide that. This allows businesses with multiple documents and casual readers to see and keep track of all their books.

7. 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.

8. References

- [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].

9. Presentation slides.

https://docs.google.com/presentation/d/15IanXz5vfVGBFNOEmeX7UDft2AvkuXx_kuvz6oKKmyl/edit?usp=sharing

Link to presentation:

https://cometmail-my.sharepoint.com/:v/g/personal/kxt190007_utdallas_edu/EcuI4QDQeLNKi9zK5c3q_G8BX6kq1F_InBq_pwzTdNZvkA?e=ge8sQg