**Use Case UC1: Add Book**

**Scope:** BookKeeper Application

**Level:** user goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** Book is added and is accurately displayed on all screens that should display it (“All Books” and any other Category specified).

**Main Success Scenario (or Basic Flow):**

1. A User navigates to a page displaying the categories they have created.
2. User selects an icon indicating adding a book (“+”) and a form is opened.
3. User types in the Name of the book, the author, and total number of pages.
4. User selects from a drop-down in which category the book should be listed.
5. User clicks “Submit”, and the book is saved.
6. User is redirected to the page of the category they specified (or “All Books” if no category was specified) to view the newly added book.

**Extensions (or Alternative Flows):**

1. At any time, System fails:

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.
2. System reconstructs prior state from before the transaction that was taking place when the system failed.
3. User attempts the transaction again.

3a. User forgets to enter the name of book, author, or total number of pages.

1. User presses “Submit” to enter incomplete data.

2. System checks to ensure that all values are entered and discovers that not all parts of the form are filled out.

3. System displays a message telling the User to “Please enter values for all required fields”.

4. User enter values for any empty fields, presses the “Submit” button, and the book is added.

5. User is redirected to the page of the category they specified to view the newly added book.

3b. User enters the wrong name of book, author, or total number of pages

1. User presses “Submit” to enter the incorrect data.

2. User is redirected to the page of the category they specified (or “All Books” if no category was specified) to

view the newly added book with the incorrect data.

3. User navigates to the category page for the category that contains the newly added book.

4. User selects the newly added book from the listing and changes the book’s data to the correct data using the

Edit Book use case.

4a. User forgets to select a category from the drop-down menu

1. Book is by default added to the “All” category

2. User clicks “Submit”, and the book is saved.

3. User is redirected to the page listing “All Books” to view the newly added book.

4. User adds the book to another category using the Add Book to Category use case

4b. User selects the wrong category from the drop-down menu

1. Book is added to the category specified, even though it is not what the User intended

2. User clicks “Submit”, and the book is saved.

3. User is redirected to the page of the category in which they just added the book on accident.

4. User changes the category the book is contained in by using the Add Book to Category use case

5a. User decides to cancel the creation of the new book instead of pressing “Submit”

1. User is redirected back to the page listing all of the categories.

2. No data is saved, and the listing of all books in all categories is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should forms for creating new items or editing previous items be on separate pages, or should they also be pop-up modals?

**Use Case UC2: Create Category**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** New Category name is displayed on the page listing all other categories.

**Main Success Scenario (or Basic Flow:**

1. A User navigates to a page displaying the categories they have created.
2. The User selects an icon indicating adding a category (“+”) and a form is opened.
3. User specifies the name of the category.
4. User presses “Submit” to create the new category.
5. User is redirected to page displaying all categories to view the newly created category.

**Extensions (or Alternative Flows):**

a. At any time, System fails:

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.
2. System reconstructs prior state from before the transaction that was taking place when the system failed.
3. User attempts the transaction again.

3a. User does not specify a name.

1. User presses “Submit” to create the new category.

2. System checks to ensure that the form is filled out and discovers that the place in the form for the name is

Empty

3. System notifies Users and asks User to “Please enter a name”

4. User enters a name and presses “Submit” to create the new category.

5. User is redirected to page displaying all categories to view the newly created category.

3b. User enters the wrong name.

1. Users presses “Submit” and submits an incorrect name.

2. User is redirected to page displaying all categories to view the newly created category.

3. User edits the category name using the Edit Category case.

4a. User decides to select “Cancel” to cancel the category creation instead of “Submit”.

1. User is redirected back to the page listing all of the categories.

2. No data is saved, and the listing of all the categories is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for creating a category be in a pop-up modal or a separate page?

**Use Case UC3: Add Personal Rating**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The book’s information page displays the user’s rating out of 10 instead of displaying an option to add one labeled “How would you rate this book?”.

**Main Success Scenario (or Basic Flow):**

1. A User navigates to a page displaying the categories they have created.
2. User selects any given category and is taken to a listing of all books in that category.
3. User selects a book and the book’s information is displayed on a second page.
4. If the book is classified as “Read”, a button is available to add a rating labeled “How would you rate this book?” near the top of the page, beneath the book’s title.
5. User clicks button to add rating, causing a form to appear.
6. User selects a score out of 10.
7. User presses submit to add the rating.
8. Rating is now displayed beneath the book’s title; can be clicked on to change.

**Extensions (or Alternative Flows):**

* 1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

4a. If the book is not classified as “Read” and is instead classified as “Currently-Reading” or “To-Read”, the option to add

a rating is not available.

1. User is not able to add a rating until finishing the book but can log progress instead if the book is classified as “Currently Reading”

6a. User does not select a score before pressing “Submit”

1. User presses “Submit” even though no score is selected.

2. System checks that there is new data to save and discovers that no rating has been specified.

3. System informs User that they have not specified a rating and returns them back to the form.

4. User is unable to leave the form until they select “Cancel” instead.

5. User selects “Cancel”.

6. The rating is unchanged and is still blank on the book’s information page.

6b. User enters a score they did not intend to submit.

1. User is redirected back to the page displaying the book’s information.

2. User selects the rating and changes the rating using the Edit Personal Rating case.

7. User enters a score but then selects “Cancel”.

1. User is redirected back to the page displaying the book’s information.

2. No data is saved, and the book’s information is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for adding a rating be in a pop-up modal or a separate page?

**Use Case UC4: Add Review**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The book’s information page displays first few words of the user’s review at the bottom of the page with the option to click on it and read the entire review, instead of displaying the option to “Add Your Personal Review”.

**Main Success Scenario (or Basic Flow):**

1. A User navigates to a page displaying the categories they have created.
2. User selects any given category and is taken to a listing of all books in that category.
3. User selects a book and the book’s information is displayed on a second page.
4. If the book is classified as “Read”, a button is available at the bottom of the page to add a review, labeled “Add Your Personal Review”.
5. User clicks button to add review.
6. User types personal review in a text box.
7. User presses “Submit” to add the review.
8. The first few words of the review are displayed at the bottom of the information page; when clicked on, the entire review is displayed in a modal box.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

4a. If the book is not classified as “Read” and is instead classified as “Currently-Reading” or “To-Read”, the option to add

a review is not available.

1. User is not able to add a review until finishing the book but can log progress instead if the book is classified as “Currently Reading”

6a. User does not enter any text constituting a review before pressing “Submit”

1. User presses “Submit” even though no text is entered in the box for the review.

2. System checks that there is new data to save and discovers that no review has been added.

3. System informs User that they have not entered any text and returns them back to the form.

4. User is unable to leave the form until they select “Cancel” instead.

5. User selects “Cancel”.

6. The review is unchanged and is still blank on the book’s information page.

6b. User enters a review they find incorrect or incomplete in some way.

1. User is redirected back to the page displaying the book’s information.

2. User selects the review and changes the rating using the Edit Personal Review case.

7. User enters a review but then selects “Cancel”.

1. User is redirected back to the page displaying the book’s information.

2. No data is saved, and the book’s information is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for creating a review be in a pop-up modal or a separate page?

**Use Case UC5: Log Progress**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The book’s information page displays a progress bar indicating the user’s progress along with a percentage indicating how much of the book they have read. This is displayed beneath the book’s title instead of an option to “Log Your Progress”.

**Main Success Scenario (or Basic Flow):**

1. A User navigates to a page displaying the categories they have created.
2. User selects any given category and is taken to a listing of all books in that category.
3. User selects a book and the book’s information is displayed on a second page.
4. If the book is classified as “Currently-Reading”, a button is available close to the book’s title to “Log Your Progress”.
5. User clicks button to log progress.
6. In a form, the user enters a page number to indicate how far they are in reading the book.
7. User presses “Submit” to add the progress.
8. Beneath the title on the book’s information page, the user’s progress is displayed as a percentage and in a progress bar, which can be clicked at any time to modify.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

4a. If the book is not classified as “Currently-Reading” and is instead classified as “Read” or “To-Read”, the option to log progress is not available.

1. User is not able to log progress before or after actually reading the book but add a review or rating instead if the User has already finished the book and it is classified as “Read”

6a. User does not enter a page number before pressing “Submit”

1. User presses “Submit” even though no page number is entered.

2. System checks that there is new data to save and discovers that no page number has been entered.

3. System informs User that they have not specified a page number and returns them back to the form.

4. User is unable to leave the form until they select “Cancel” instead.

5. User selects “Cancel”.

6. User is redirected back to the book’s information page.

7. The progress is unchanged on the book’s information page.

6b. User enters a page number that is incorrect or that they did not intend to submit.

1. User is redirected back to the page displaying the book’s information.

2. User selects the progress and changes the progress repeating the steps above in the Log Progress case.

6c. User enters a page number greater than the book’s total number of pages.

1. The System validates the data entered and discovers that the current page number exceeds the total number

of pages.

2. The System informs the User that the page number they entered is invalid and returns them back to the form.

3. User is unable to leave the form until they enter a valid page number of select “Cancel” instead.

3a. User enters valid page number.

1. User presses “Submit” to add the progress.
2. Beneath the title on the book’s information page, the user’s progress is displayed as a percentage and in a progress bar, which can be clicked at any time to modify.

3b. User selects “Cancel”.

1. User is redirected back to the book’s information page.

2. The progress is unchanged on the book’s information page.

7. User enters a page number but then selects “Cancel”.

1. User is redirected back to the page displaying the book’s information.

2. No data is saved, and the book’s information is unchanged (either blank or remains the page number it was

before).

**Special Requirements:**

- All messages from the system indicating invalid input should be displayed in pop-up modals so that the user cannot move forward in the application without dealing with the situation at hand and entering an answer into the message box.

- Progress should be displayed in a progress bar on the screen along with the percentage.

- Progress should be calculated by dividing the current page number by the total number of pages and multiplying by 100.

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for logging progress be in a pop-up modal or a separate page?

**Use Case UC6: Delete Book**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** On the Category page in which the deleted book was once contained, the book the user deleted is no long displayed with the other books in that category or in any category that once contained it.

**Main Success Scenario (or Basic Flow):**

1. A user navigates to a page displaying all the categories they have created.
2. User clicks on a category, redirecting them to a page displaying a list and icon of all books added.
3. User then clicks on the book listing of the book they want to delete, taking them to the book’s information page.
4. The user clicks on an option to “Edit Book” with a pencil icon.
5. A form appears on a new page, in which the user can edit the book’s information.
6. At the bottom of the page, user clicks a red button labeled “Delete Book”.
7. System displays a prompt asking if the user would like to confirm this action (“Are you sure you want to delete this book?”)
8. The user clicks “Yes” to confirm, and the system deletes the book’s entry from the database.
9. The page reloads and the category page is shown displaying all remaining books, with the deleted book now missing.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

7a. User decides not to delete the book.

1. User clicks “No” when prompted to confirm the deletion.

2. User is redirected back to the page where the User can edit the book.

3. Book is not deleted and is displayed in all categories that contain it.

8a. User deletes a book they did not mean to delete.

1. User clicks “Yes” to confirm the deletion.

2. User is redirected back to the page displaying a listing of all books in the category that the deleted book was

once contained in.

3. User must add a new book with the same information using the Add Book case.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for editing/deleting a book be in a pop-up modal or a separate page?

**Use Case UC7: Edit Personal Rating**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The user’s new rating is displayed beneath the book’s title instead of the former rating.

**Main Success Scenario (or Basic Flow):**

1. A user navigates to a page displaying all the categories they have created.
2. User clicks on a category, redirecting them to a page displaying a list and icon of all books added.
3. User then clicks on the book listing of the book they want to edit, taking them to the book’s information page.
4. If the book is classified as “Read”, a rating is displayed near the top of the page, beneath the book’s title.
5. User clicks the current rating to change it, causing a form to appear.
6. User selects a score out of 10 and presses submit.
7. New rating is now displayed beneath the book’s title; can be clicked on to change.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

5a. User decides not to change their rating.

1. User selects “Cancel” instead of submitting a new rating.

2. User is redirected back to book information page.

3. Rating is unchanged and indicates the same rating the User submitted last.

6a. User does not select a score before pressing “Submit”

1. User presses “Submit” even though no score is selected.

2. System checks that there is new data to save and discovers that no rating has been specified.

3. System informs User that they have not specified a rating and returns them back to the form.

4. User is unable to leave the form until they select “Cancel” instead.

5. User selects “Cancel”.

6. The rating is unchanged and is still blank on the book’s information page.

6b. User enters a score they did not intend to submit.

1. User is redirected back to the page displaying the book’s information.

2. User selects the rating and changes the rating using the Edit Personal Rating case.

7. User enters a score but then selects “Cancel”.

1. User is redirected back to the page displaying the book’s information.

2. No data is saved, and the book’s information is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for editing a rating be in a pop-up modal or a separate page?

**Use Case UC8: Delete Category**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The category is no longer displayed on the page listing all categories, but the books are retained and can be found in “All Books” and any other category of which they are a part.

**Main Success Scenario (or Basic Flow):**

1. A user navigates to a page displaying all the categories they have created.
2. User clicks on a category, redirecting them to a page displaying a listing all books contained in the category.
3. User then clicks on an option at the top of the screen below the Category Title labelled “Edit Category” with a pencil icon, denoting that the Category information can be edited.
4. User is taken to another form in which the User can edit the Category Title or delete the category, denoted by a red button at the bottom of the screen labelled “Delete Category”.
5. User clicks the “Delete Category” button
6. The system prompts the User to confirm this action with a modal box labeled “Are you sure you want to remove this category?” and explaining that the category will need to be remade entirely if the user wants to use it again, with the options “Yes” and “Cancel”.
7. The user clicks the “Yes” button.
8. System removes the category but retains all books in the “All Book” category and any other category in which they are included.
9. User is redirected to the page displaying all other categories, and the deleted category is no longer displayed.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

7a. User decides not to delete the category.

1. User clicks “No” when prompted to confirm the deletion.

2. User is redirected back to the page where the User can edit the category.

3. Category is not deleted and is displayed in listing of categories

8a. User deletes a category they did not mean to delete.

1. User clicks “Yes” to confirm the deletion.

2. User is redirected back to the page displaying a listing of categories

3. User must add a new category with the same information using the Add Category case.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for editing/deleting a category be in a pop-up modal or a separate page?

**Use Case UC9: Edit Category**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The Category’s name is changed from the former name to the new one specified by the user both on the Category’s information page and in the listing of all the user’s categories.

**Main Success Scenario (or Basic Flow):**

1. A user navigates to a page displaying all the categories they have created.
2. User clicks on a category, redirecting them to a page displaying a listing all books contained in the category.
3. User then clicks on an option at the top of the screen below the Category Title labelled “Edit Category” with a pencil icon, denoting that the Category information can be edited.
4. User is taken to another form in which the User can edit the Category Title or delete the category, denoted by a red button at the bottom of the screen labelled “Delete Category”.
5. User clicks on the title and enters a new title to replace it.
6. User presses “Submit” to edit the category.
7. The system prompts the User to confirm this action with a modal box labeled “Are you sure you want to rename this category?” and explaining that this action cannot be undone, with the options “Yes” and “Cancel”.
8. The user clicks the “Yes” button.
9. System replaces the former title with the newly specified Category name
10. User is redirected to the Category’s page listing all books contained within it, and the newly specified Category name is displayed as the Title instead.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

5a. User does not specify a name.

1. User presses “Submit” to create the new category.

2. System checks to ensure that the form is filled out and discovers that the place in the form for the name is

Empty

3. System notifies Users and asks User to “Please enter a name”

4. User enters a name and presses “Submit” to create the new category.

5. User is redirected to page displaying all categories to view the newly created category.

5b. User enters the wrong name.

1. Users presses “Submit” and submits an incorrect name.

2. User is redirected to page displaying all categories to view the newly created category.

3. User edits the category name using the Edit Category case.

6a. User decides to select “Cancel” to cancel the category creation instead of “Submit”.

1. User is redirected back to the page listing all of the categories.

2. No data is saved, and the listing of all the categories is unchanged.

8a. User decides not to change the category and clicks “No” instead.

1. User is redirected to page displaying the books in the category.

2. Category data is unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for editing/deleting a category be in a pop-up modal or a separate page?

**Use Case UC10: Add Book to Category**

**Scope:** BookKeeper Application

**Level:** User goal

**Primary Actor:** User

**Stakeholders and Interests:**

* User: Wants a simple, intuitive interface for logging books that they can easily update and where they can easily view user and book data. Wants a robust application that accurately records and displays all changes. Wants an application that can handle small changes every few days or a succession of several changes and updates in a short timeframe.

**Preconditions:** User has successfully created an account or logged into a previously created account with proper authentication.

**Success Guarantee (or Postconditions):** The book now has additional categories listed on its information page reflecting the categories to which the User added the book. The book can also be found listed in the book listing page for any of these categories.

**Main Success Scenario (or Basic Flow):**

1. A user navigates to a page displaying all the categories they have created.
2. User clicks on a category, redirecting them to a page displaying a listing of all books contained in the category.
3. User then clicks on an already created Book
4. User is redirected to the Book’s information page.
5. User selects option to “Edit Categories”.
6. User is redirected to a new page with a form where the user can add or remove categories in which the book is included.
7. To add the book to another category, the User clicks on a drop-down menu displaying all created categories; there is no limit to the number of categories the book can be part of.
8. The User clicks “Submit Changes”.
9. The System prompts the User to confirm the changes, listing which categories the book will now be part of, and with options “Yes” or “No”.
10. User selects “Yes” and System updates categories.
11. User is redirected to the book’s information page, where the book now has more categories listed under its Title.

**Extensions (or Alternative Flows):**

1. At any time, System fails

Ensure all incomplete transactions are cancelled so that data is not inconsistent.

1. User restarts System and logs in.

2. System reconstructs prior state from before the transaction that was taking place when the system failed.

3. User attempts the transaction again.

7a. User does not add book to any more categories.

1a. User clicks “Submit changes”

a. User is redirected back to book information page.

b. The book’s categories are unchanged.

1b. User clicks “Cancel”

a. User is redirected back to book information page.

b. The book’s categories are unchanged.

8a. User wants undo changes.

1. When prompted to confirm the changes, user selects “No”.

2. User is redirected back to book information page.

3. The book’s categories are unchanged.

**Special Requirements**

- All pop-up messages should be modals so that the user cannot access other parts of the page until they submit an answer to the pop-up message

**Frequency of Occurrence:** Could be nearly continuous.

**Open Issues:**

- Should the form for editing a book’s categories be in a pop-up modal or a separate page?