AL Local Library Management System

Andie Leonhard

System Requirements Specification

Updated 11/7/2024

Table of Contents

Customer Problem Statements & System Requirements:

Problem statement: Even for a smaller locale, managing a public library involves the management of a large number of media and users reliably. This means keeping track of all inventory of various types (DVDs, CDs, books)[1] including adding or removing items from inventory, managing users checking out or returning items, managing holds and waitlists for more popular items, and finding locations of items. Without proper management of inventory and space, problems will arise such as lost or misplaced items and cause major issues in the library's ability to continue to provide service to the community.[2]

Objectives: The system sets out to make resource tracking easy for library staff members by keeping real-time records of the location and status of resources and to allow visitors a way to search available resources and manage their checkouts, reservations, and fees from home. The system should be scalable as the library's inventory and visitor counts may vary.

System requirements: (bolded items have already been successfully implemented)

- Staff should be able to:
    - Add new user/library card to system
    - Log in/out/create account
    - Add items of various types (DVDs, VHS, CDs, books, magazines) to inventory
    - **Search inventory items**
    - View item details (including title, author, ISBN, checkout status, location/shelf, waitlist, checkout history, etc.)
    - Edit item details
    - Remove items from inventory
    - **Check out items from inventory on a specified visitor's library card/account**
    - **Return items to inventory**
    - Look up visitor information by library card number (Checkouts, holds, reservations, fees)
    - Process fee payments for a specified visitor's library card/account
- Visitors should be able to:
    - Log in/out/create account from library card
    - **Search catalogue**

  - View item details
  - Place hold/join waitlist for an item
  - **View checkout history and holds**
  - View fees/pay fees

Typical customers: The software would be purchased by library management, so likely city officials, but users would be librarians and library visitors, so these will be considered most when developing system.

[1]Removed a couple types of media in order to minimize scope.

[2]Removed the system functions related to reserving rooms/spaces in order to minimize scope and focus more on inventory functions.

Functional Requirements Specification:

Functionality: The system will consist of a database to manage library resources, a web app for visitors to view available inventory and spaces, as well as to manage their own checkouts, holds, reservations, and fees, and a traditional application to allow for staff management of resources onsite, including adding and editing inventory, and processing checkouts and returns. Both the staff application and the web app will have log ins allowing users to access information from the database in real time (with varying permissions), allowing for visitors and staff to view relevant information in real time.

Requirements: The traditional application must be able to support basic library functions performed by staff members, meaning it must be able to access the database. The web application needs to perform simple functions for library users remotely, so must be able to have limited access to the database.

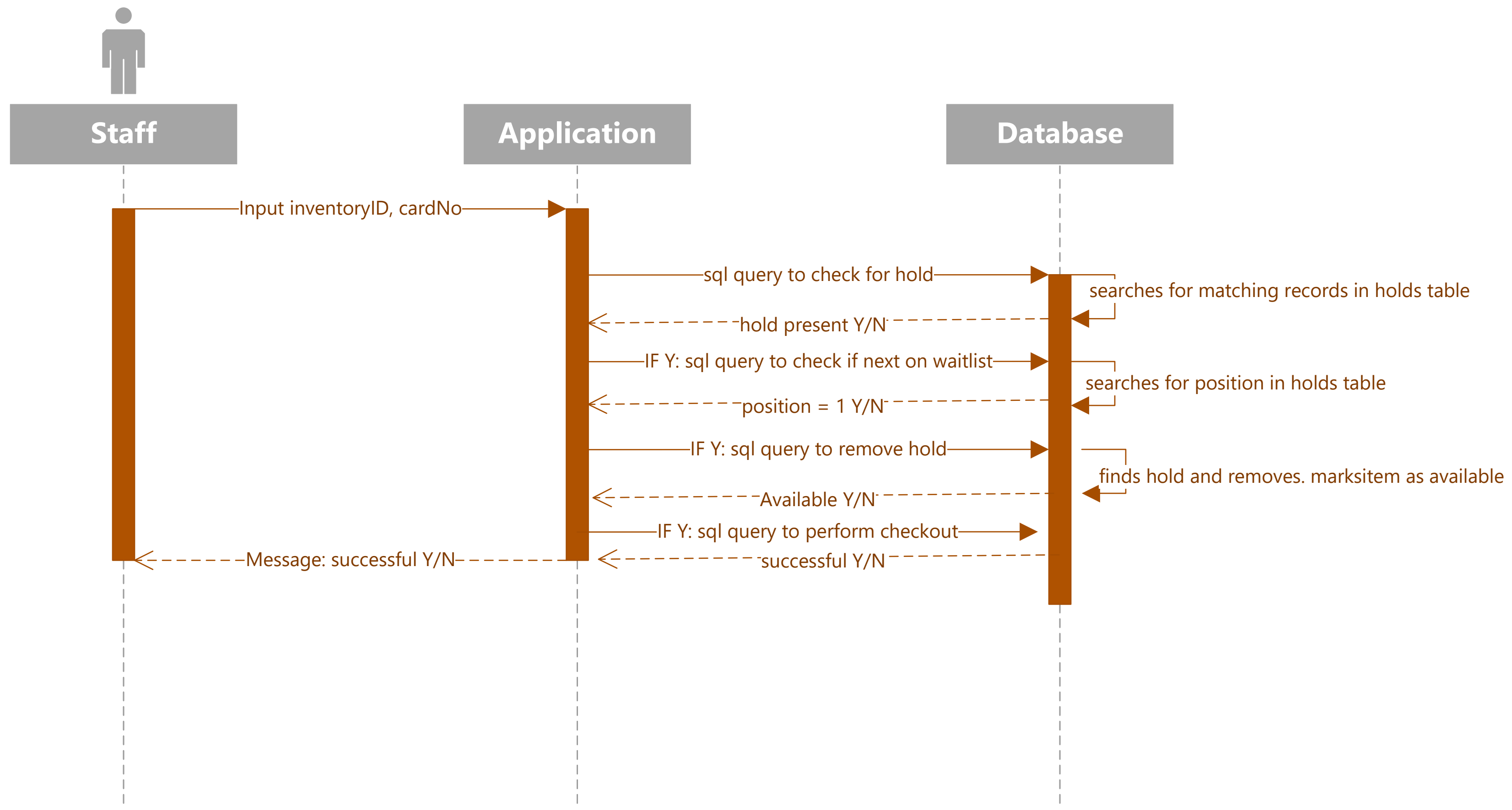
Required methods/use cases:

- Traditional Application:
    - Add user
    - Add items to inventory
    - **Search inventory items**
    - View item details
    - Edit item details
    - Remove item from inventory
    - **Check out items**
    - **Return items to inventory**
    - Look up visitor information
    - Process fee payments
- Web application:
    - Log in/out/create account
    - **Search catalogue**
    - View item details
    - Place hold
    - Remove hold
    - **View checkout history and holds**
    - View fees/pay fees

Database Requirements: The database must store comprehensive information about inventory as possible, within reasonable time and resource limits. This includes information about the type of media, title, author/artist/director or studio depending on the item type, genre, location within the library, and any other basic identifying information that users might find helpful. In addition, the database must track check out, holds and waitlists, and returns, as well as user information and existing fine balances. This requires a well-built, normalized relational database, which must be compatible with the traditional and web applications that need to access this data.

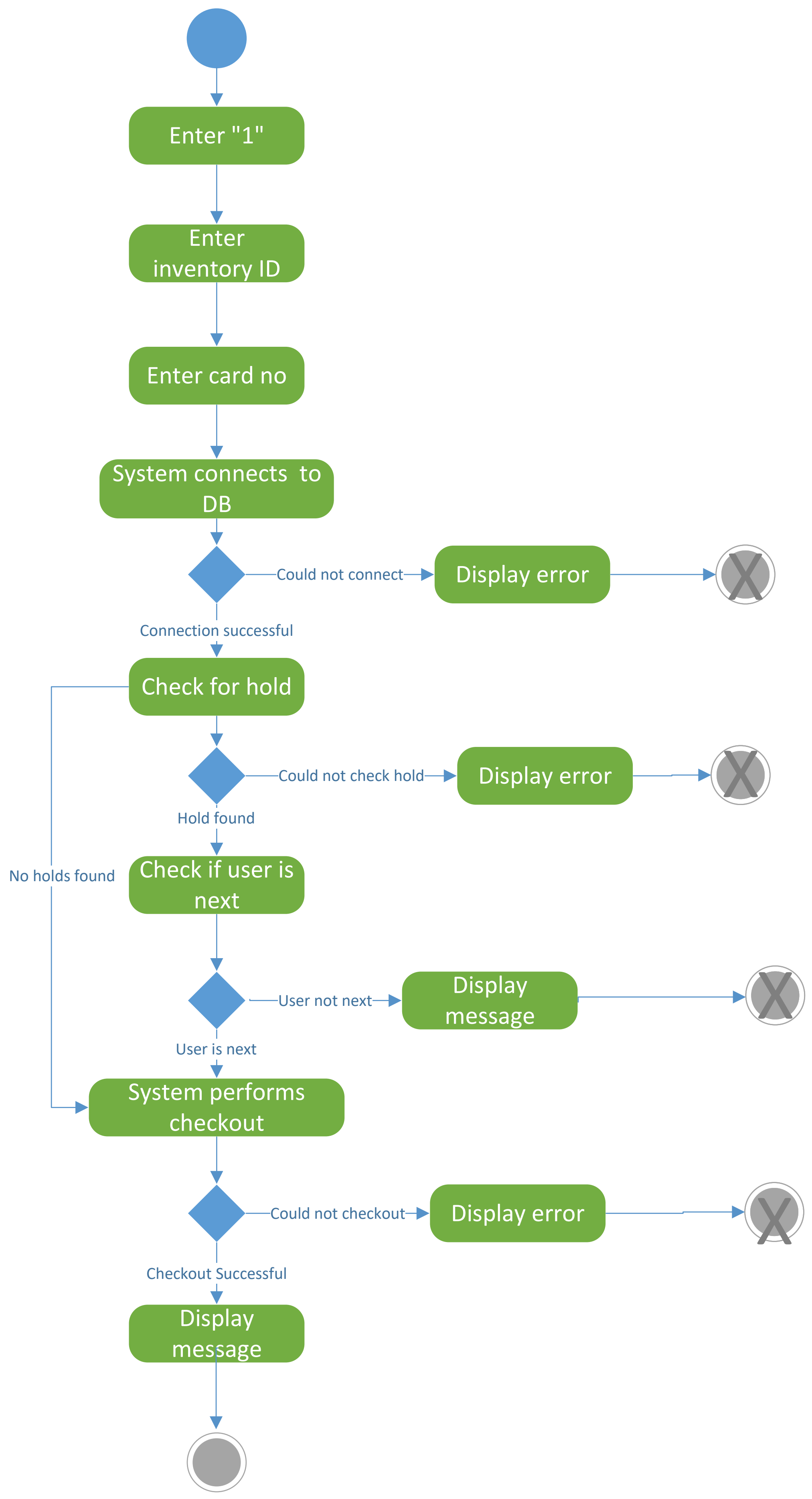
System Sequence Diagram:

Sequence Diagram: Check Out Item

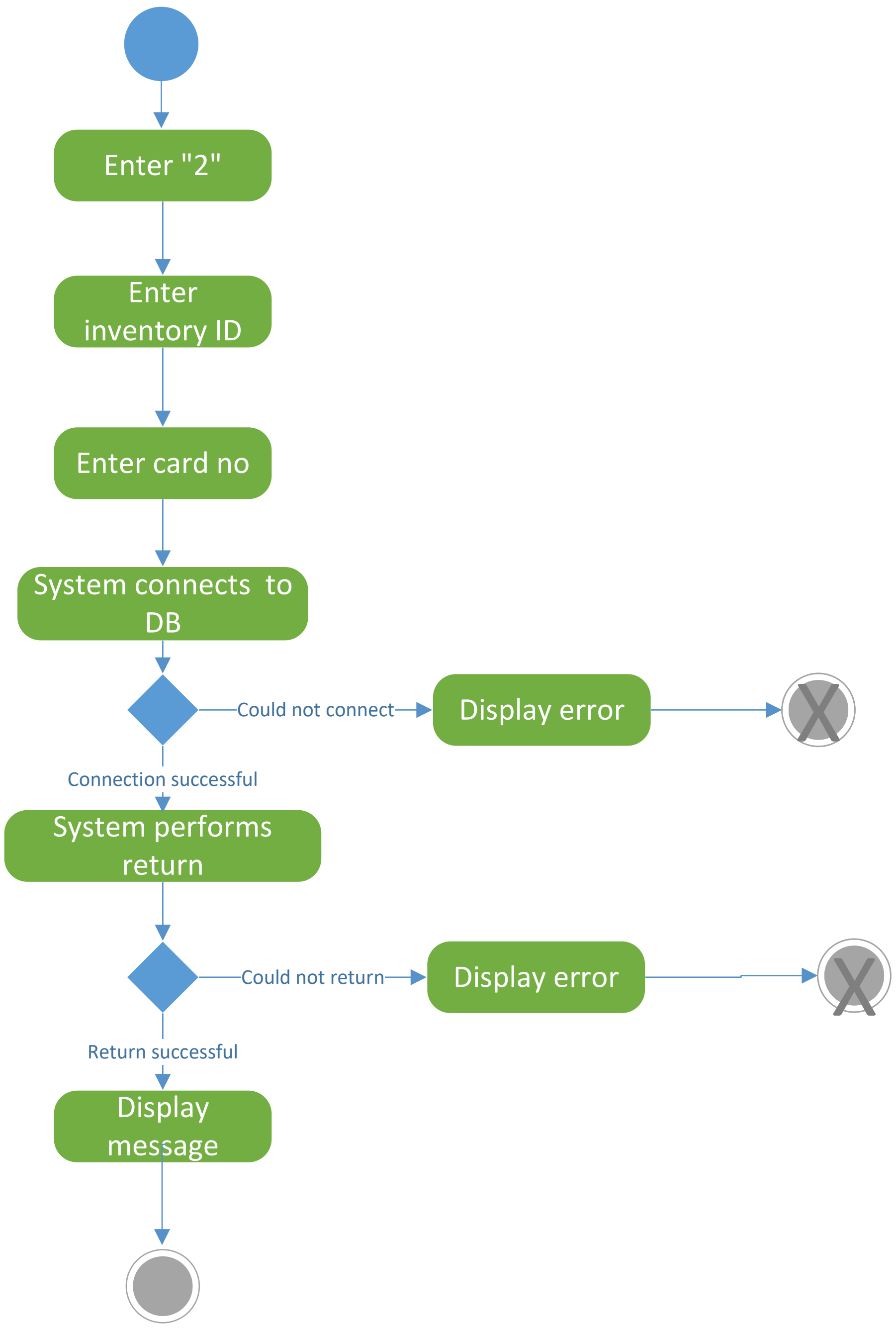

Staff
Application
Database
Input inventoryID, cardNo
sql query to check for hold
searches for matching records in holds table
hold present Y/N
IF Y: sql query to check if next on waitlist
searches for position in holds table
position = 1 Y/N
IF Y: sql query to remove hold
finds hold and removes. marksitem as available
Available Y/N
IF Y: sql query to perform checkout
Message: successful Y/N
successful Y/N

Activity Diagrams:

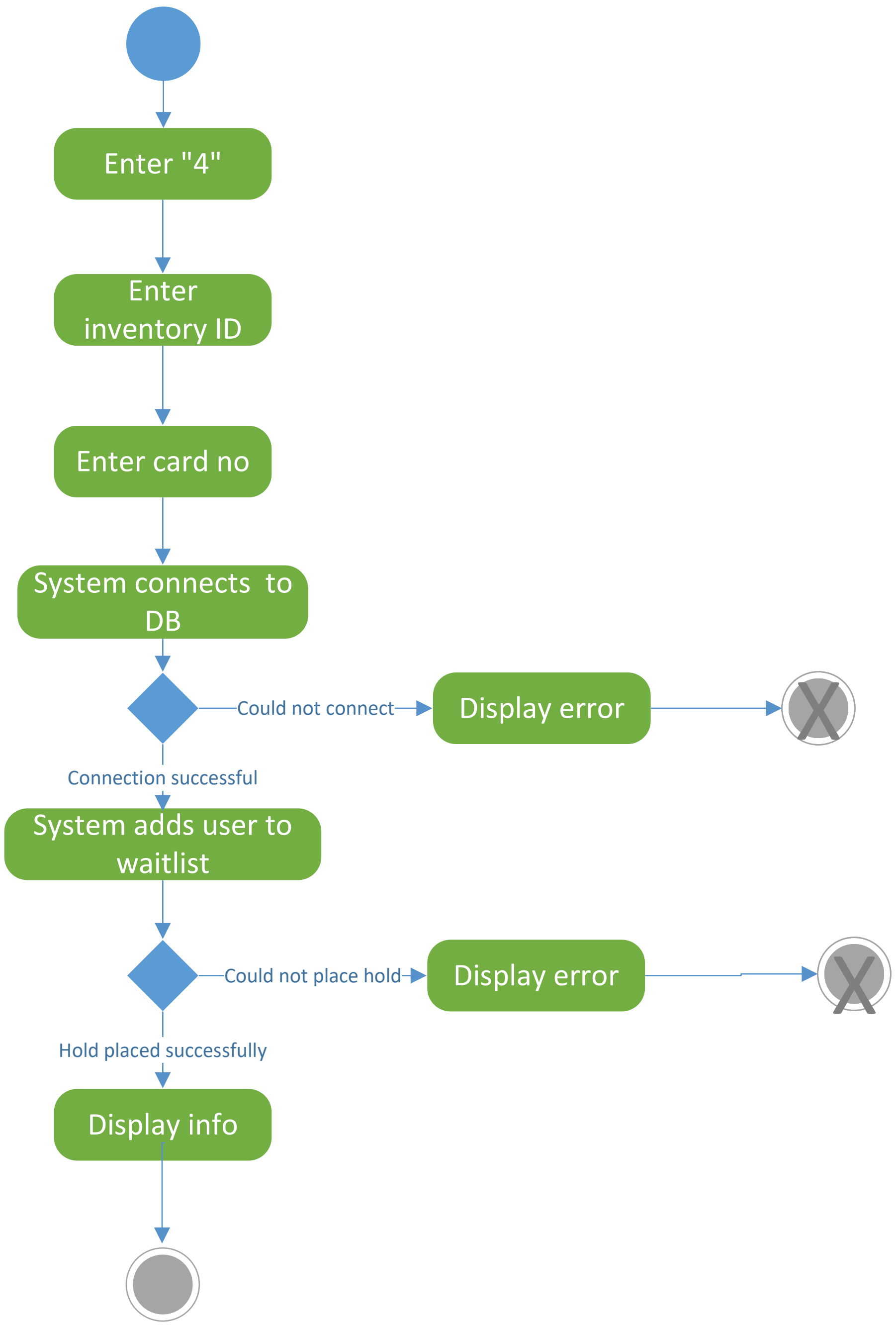
Use Case: Check Out Item



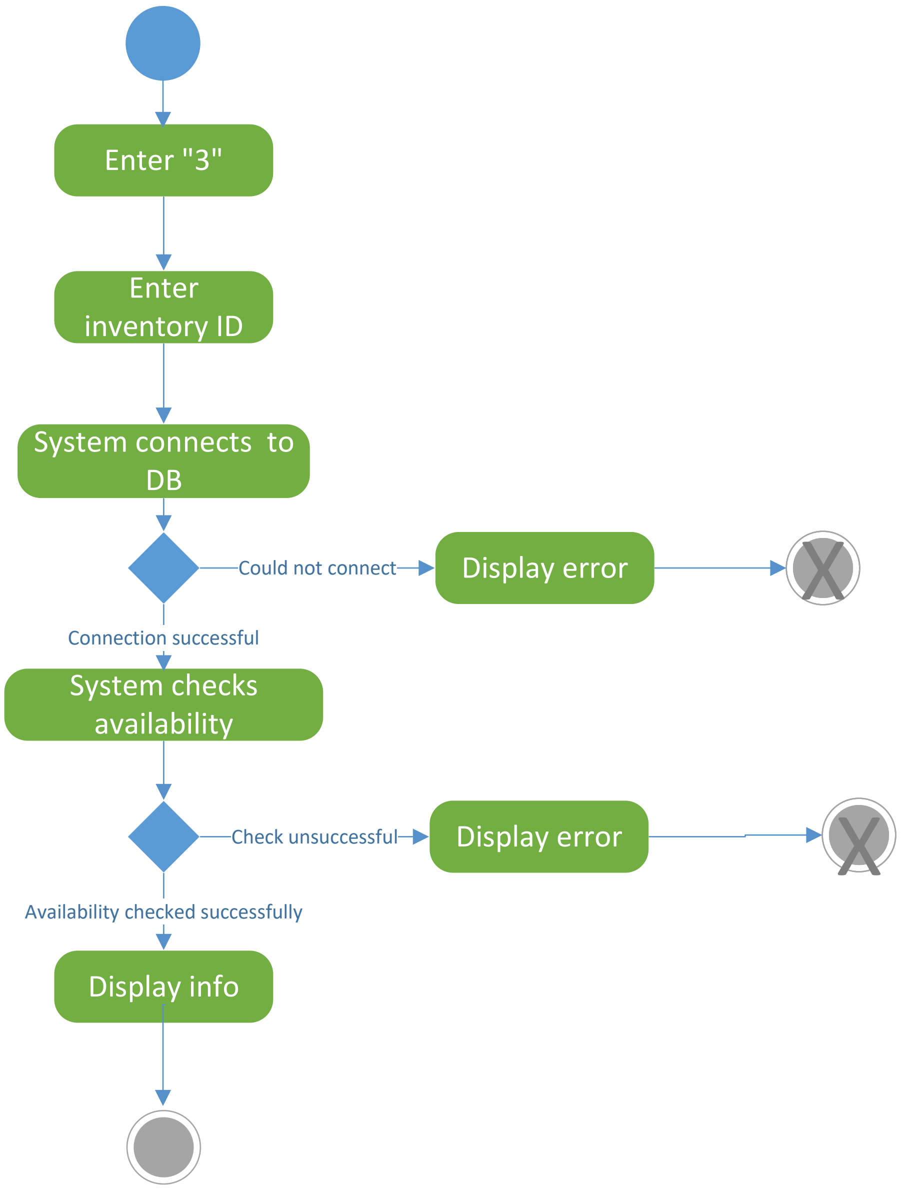
Use Case: Return Item



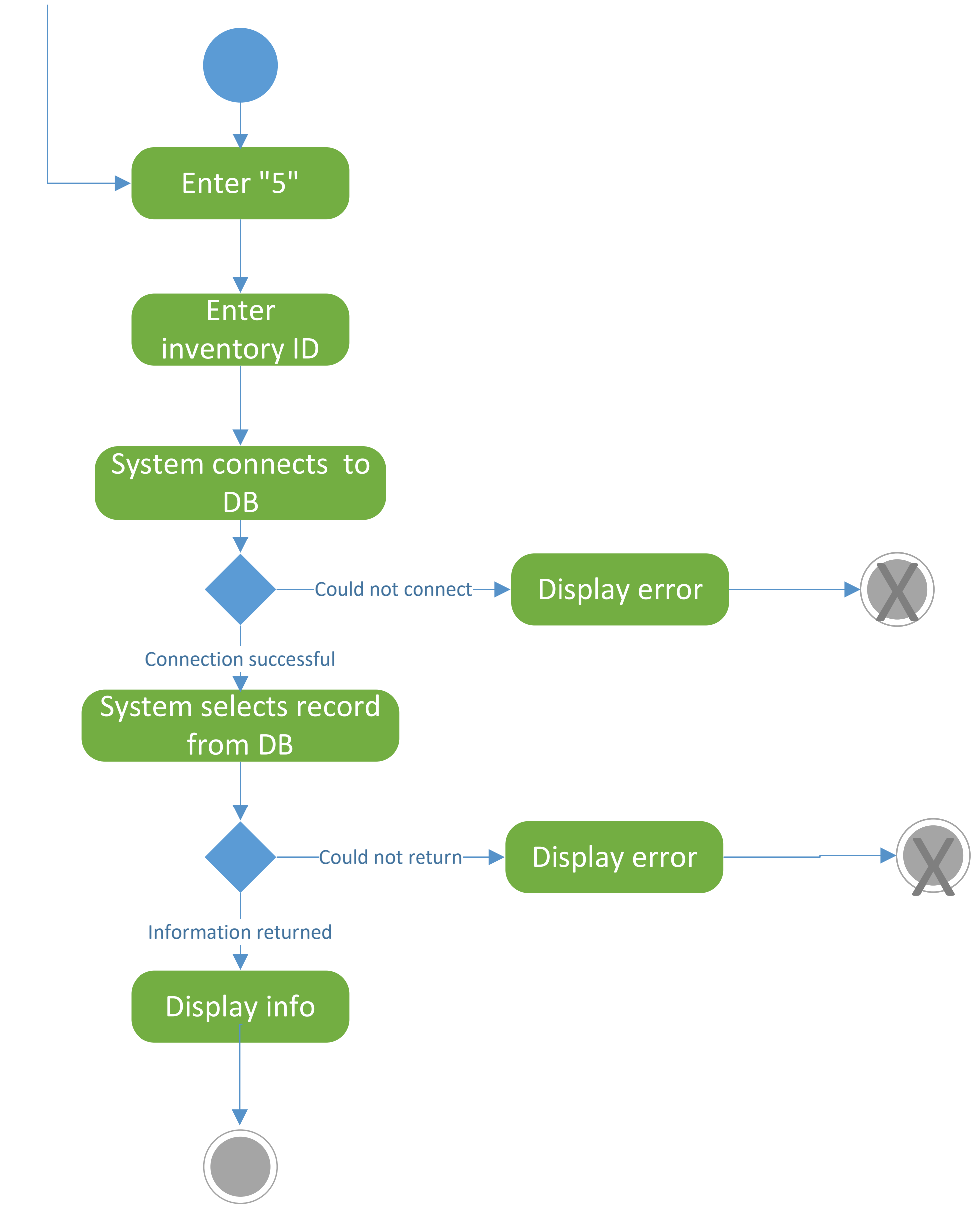
Use Case: Place Hold



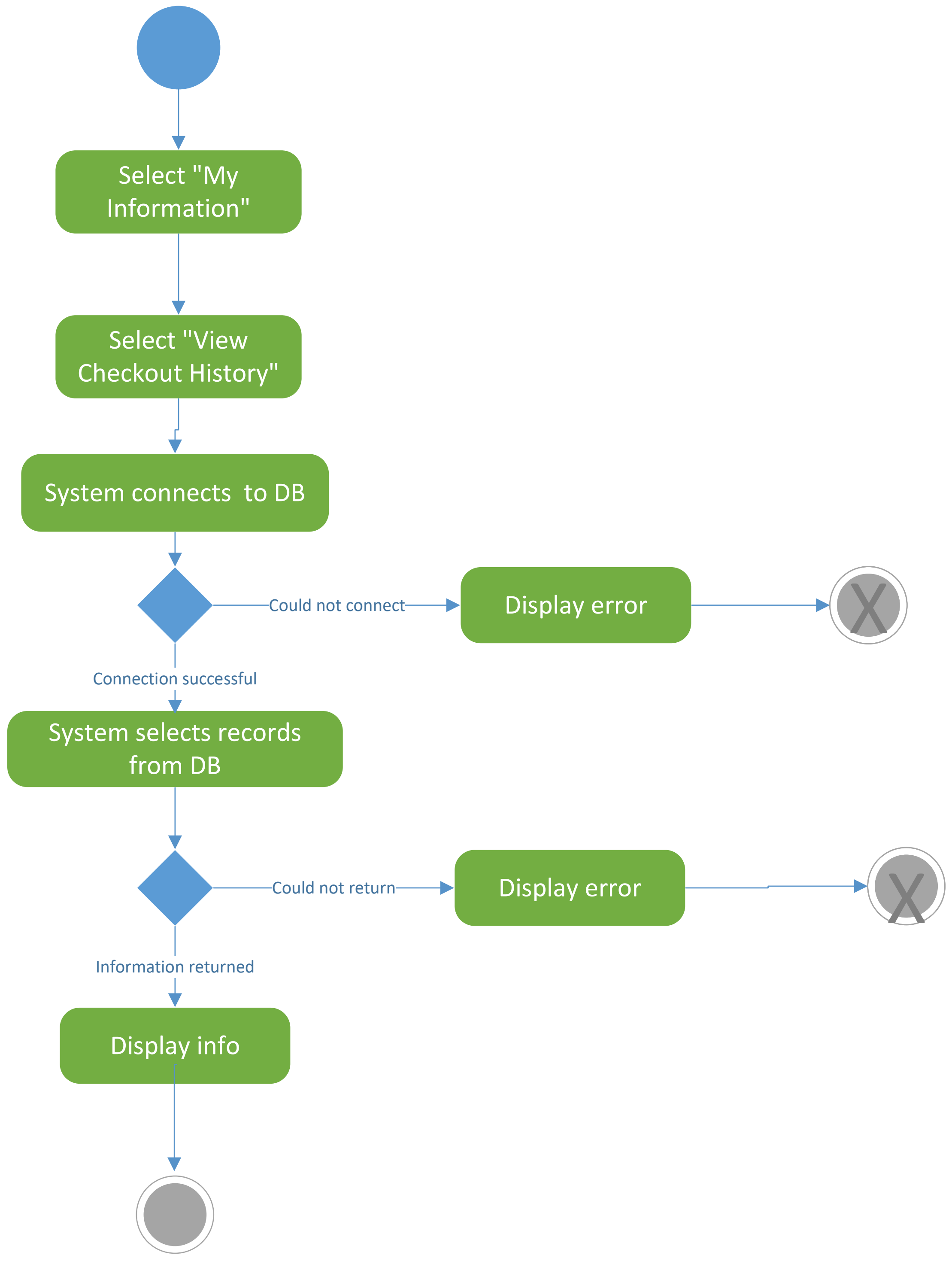
Use Case: Check Availability



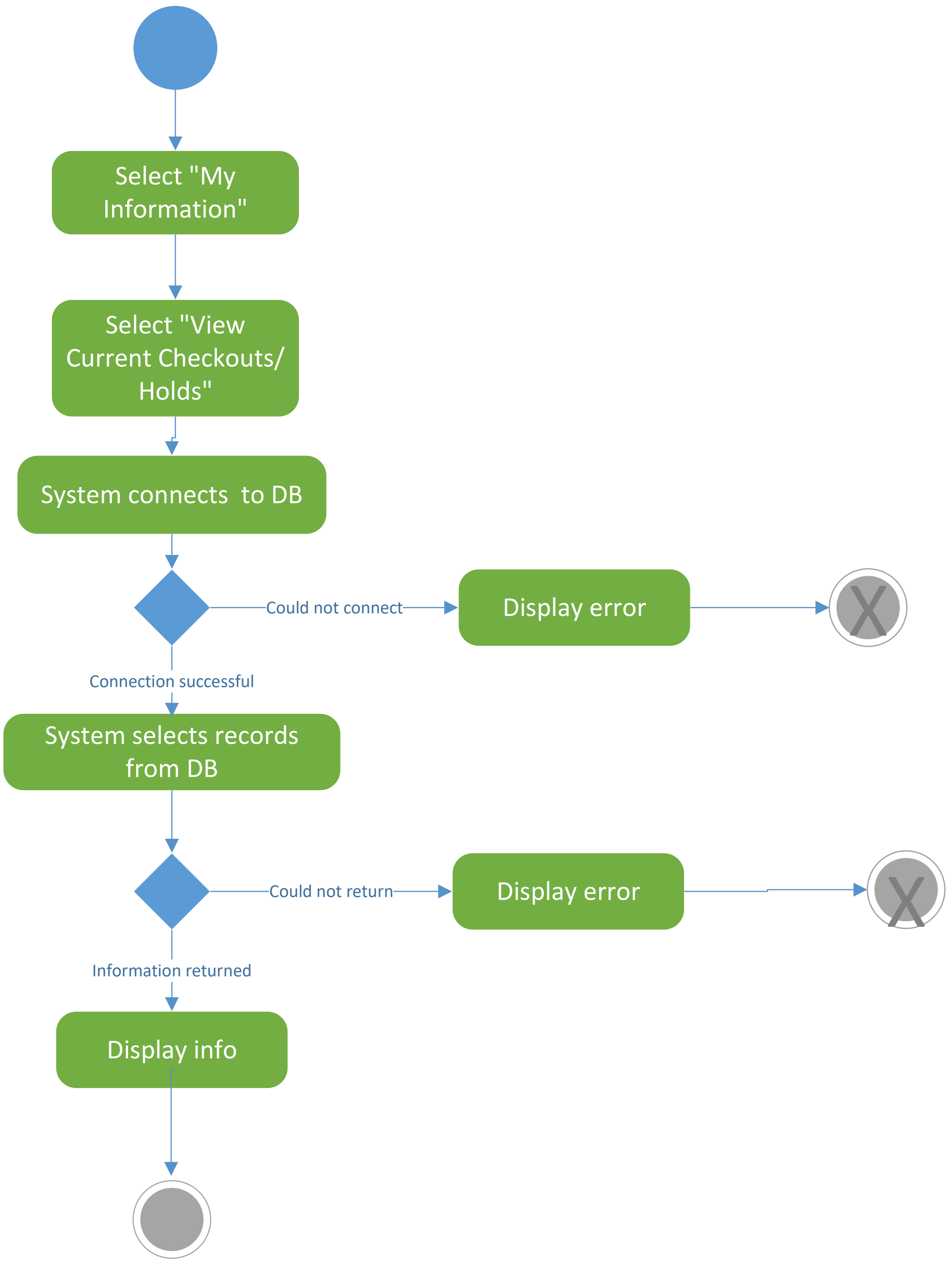
Use Case: Invnetory Search



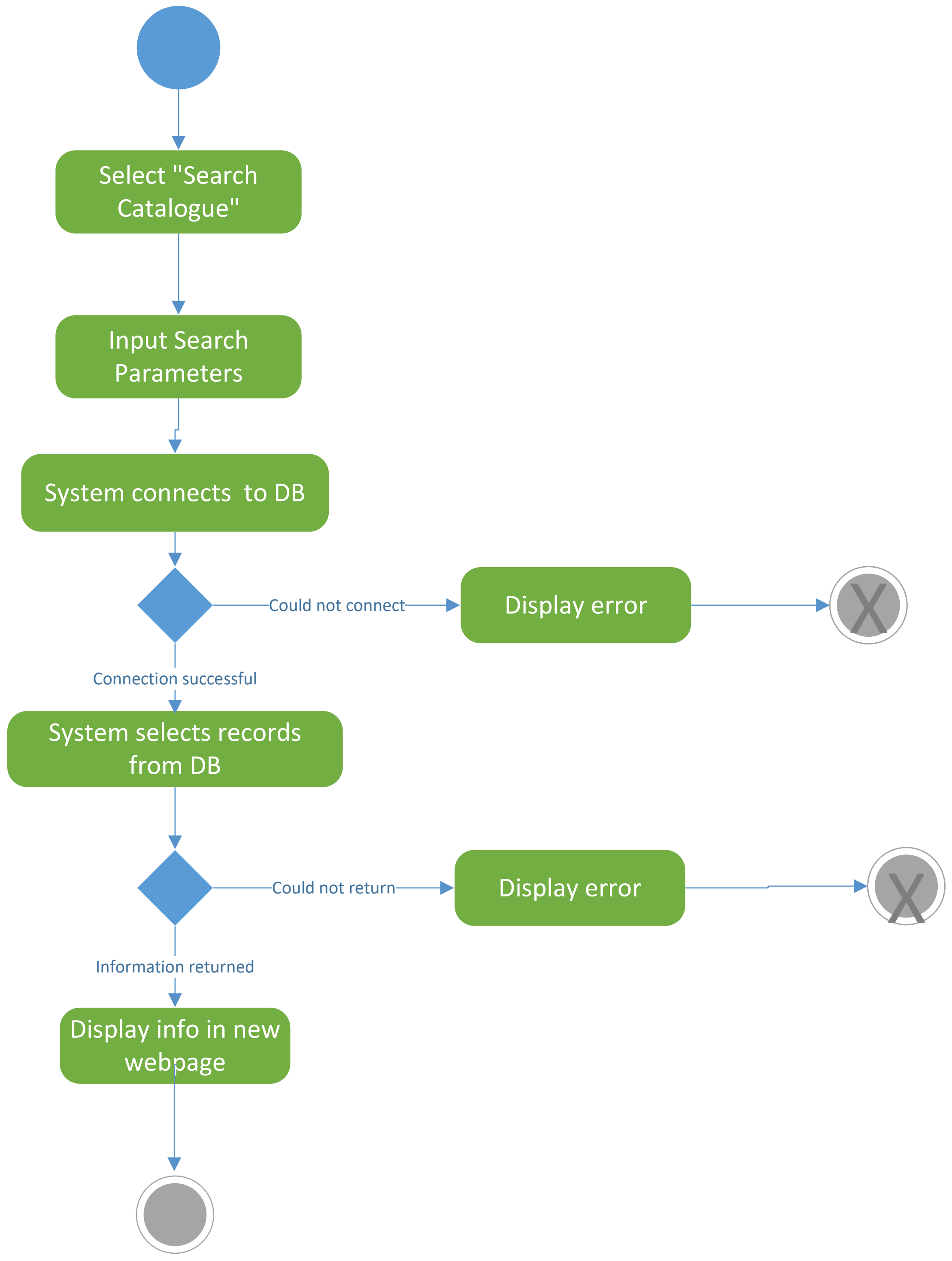
Use Case: View Checkout History



Use Case: View Current Holds and Checkouts



Use Case: Search Inventory



User Interface Specification:

Stakeholders: Library management and staff, community library patrons

Actors and Goals:

- Library Management: Manage inventory, users, and balances successfully to minimize errors and unnecessary complications and overhead
- Library Staff: To perform basic library functions conveniently and accurately
- Library Users: Access library resources and manage account easily and remotely

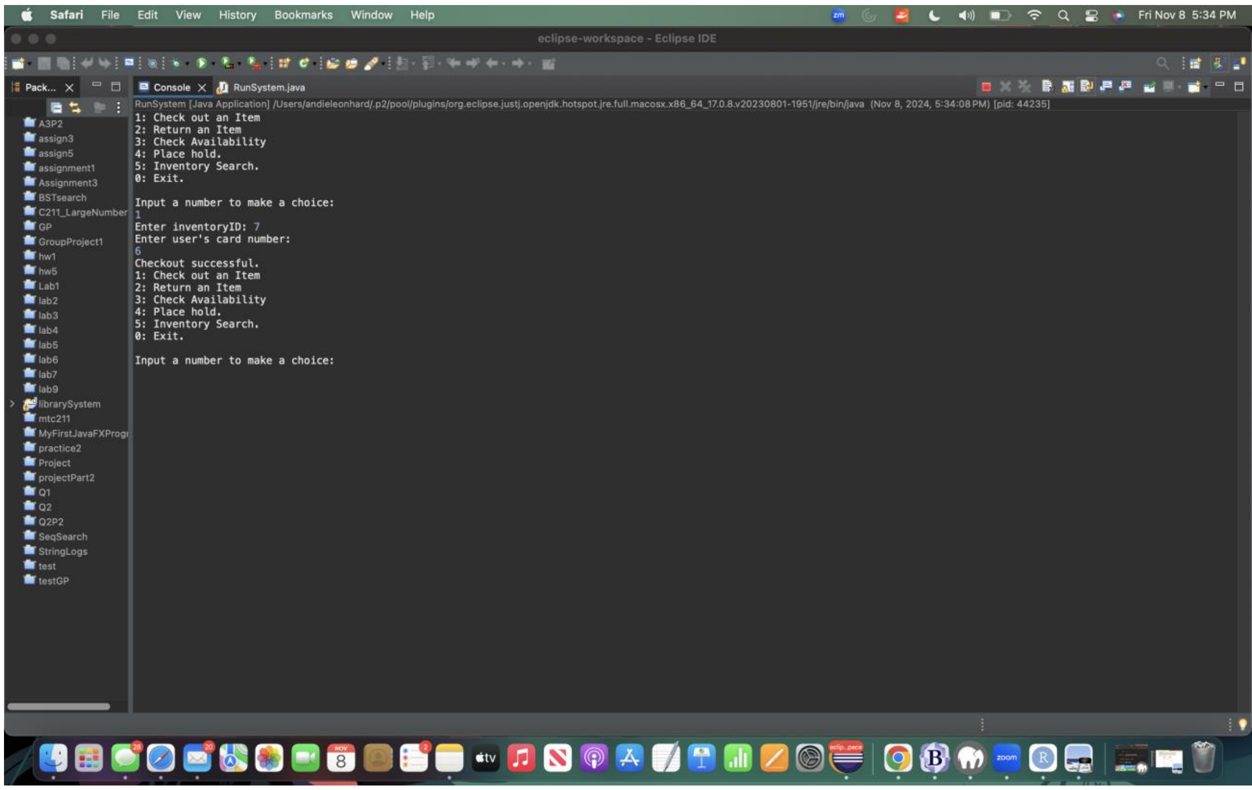
Use Cases:

Staff: Checkout item[3]

- Starting at menu, enter corresponding number to check out item
- Enter (or, practically, scan in) inventoryID of item to be checked out
- Enter (or, practically, scan in) cardNo of user checking out item

Navigation Path: Menu > Input Inventory ID > Input Card No > Success

Number of clicks: 0 (maybe 1 to set cursor)



```
RunSystem [Java Application] /Users/andieleonhard/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_17.0.8.v20230801-1951/jre/bin/java (Nov 8, 2024, 5:34:08 PM) [pid: 44235]
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
1
Enter inventoryID: 7
Enter user's card number:
6
Checkout successful.
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
```

Number of Keystrokes: 3

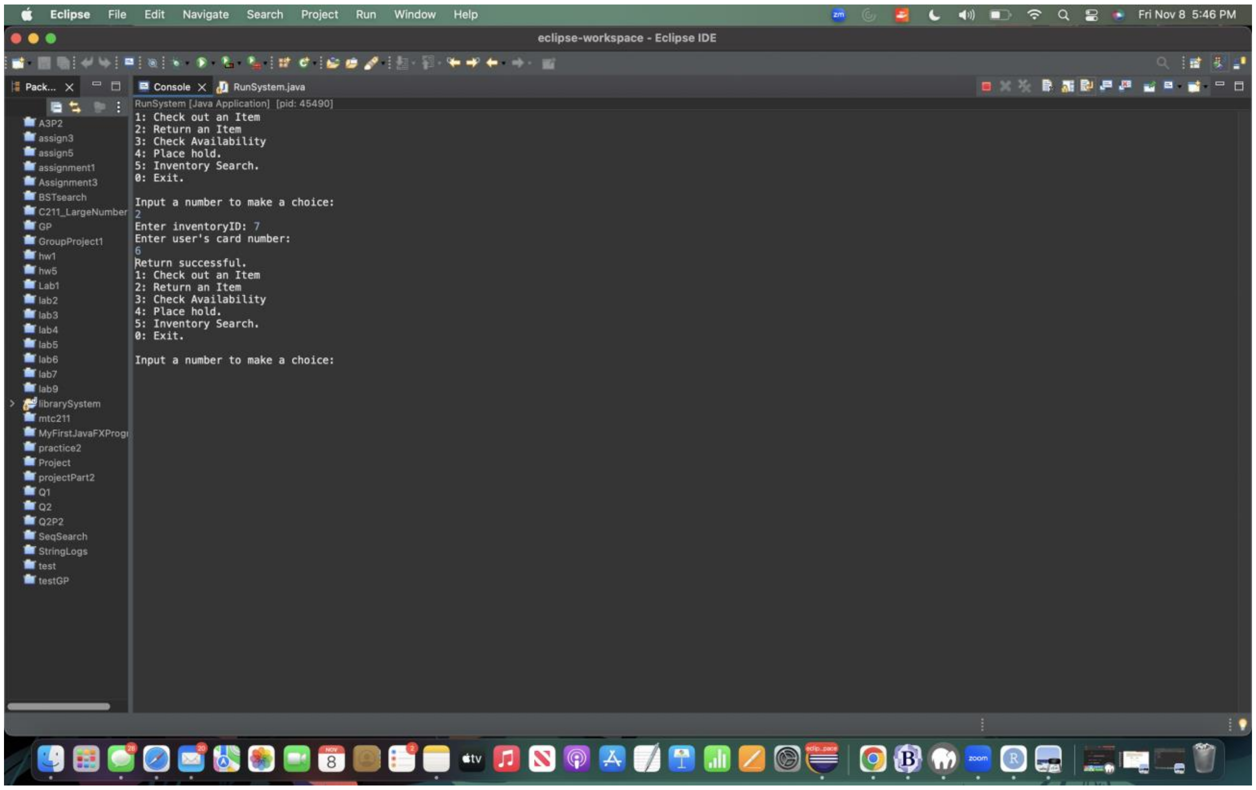
Staff: Return item

- Starting at menu, enter corresponding number to return item
- Enter (or, practically, scan in) inventoryID of item to be returned
- Enter (or, practically, scan in) cardNo of user returning item

Navigation Path: Menu > Input Inventory ID > Input Card No > Success

Number of clicks: 0 (maybe 1 to set cursor)

Number of Keystrokes: 3



```
RunSystem [Java Application]  [pid: 45490]
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
2
Enter inventoryID: 7
Enter user's card number:
6
Return successful.
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
```

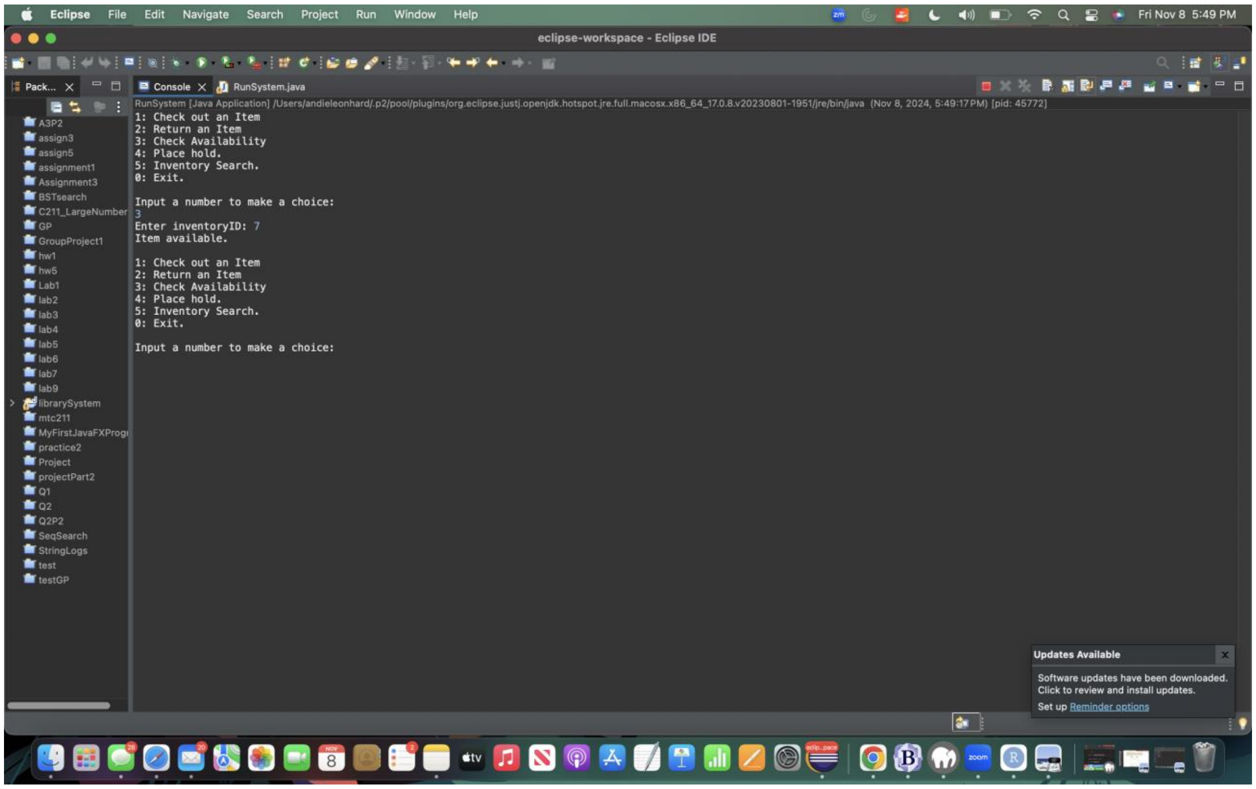
Staff: Check Availability

- Starting at menu, enter corresponding number to check availability
- Enter (or, practically, scan in) inventoryID of item to check availability

Navigation Path: Menu > Input Inventory ID > Success

Number of clicks: 0 (maybe 1 to set cursor)

Number of Keystrokes: 2



```
RunSystem [Java Application] /Users/andieleonhard/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_17.0.8.v20230801-1951/jre/bin/java  (Nov 8, 2024, 5:49:17 PM) [pid: 45772]
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
3
Enter inventoryID: 7
Item available.

1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
```

Updates Available

Software updates have been downloaded. Click to review and install updates.

Set up Reminder options

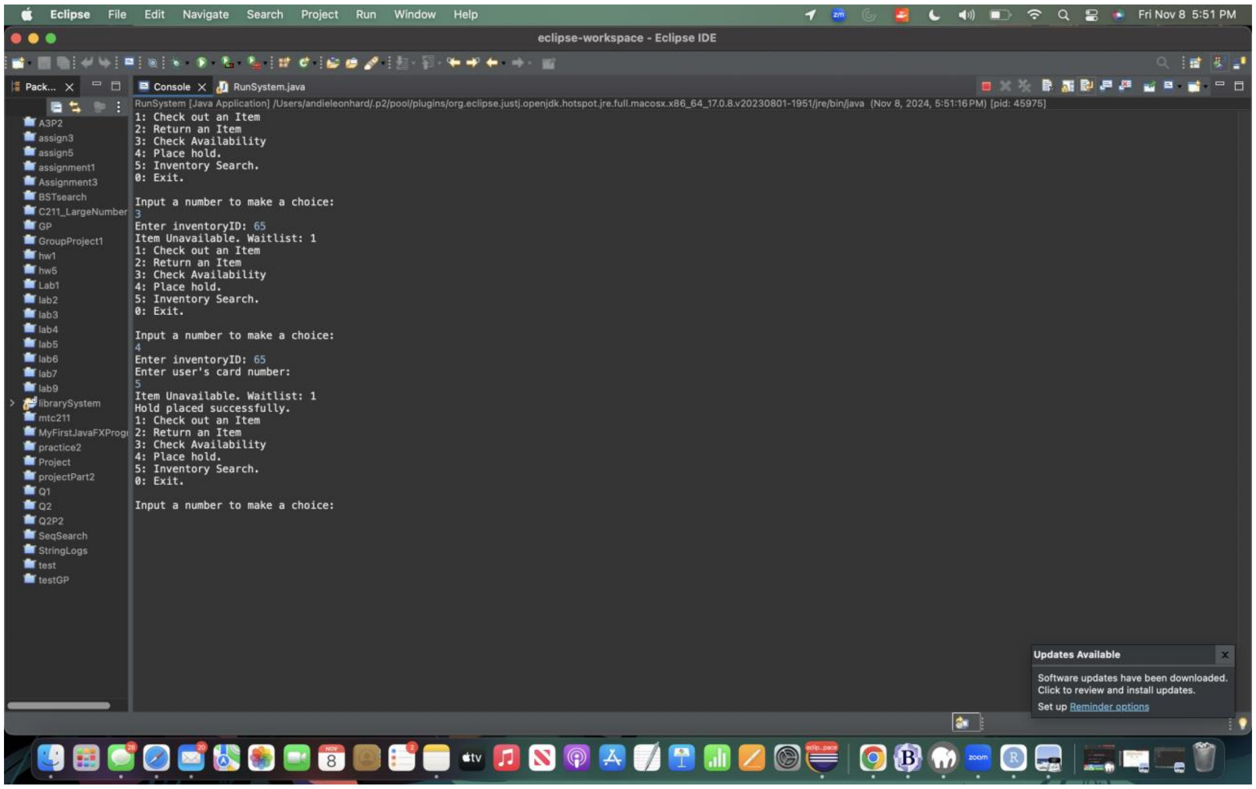
Staff: Place Hold

- Starting at menu, enter corresponding number to place hold
- Enter (or, practically, scan in) inventoryID of item to be placed on hold
- Enter (or, practically, scan in) cardNo of user placing the hold

Navigation Path: Menu > Input Inventory ID > Input Card No > Success

Number of clicks: 0 (maybe 1 to set cursor)

Number of Keystrokes: 3



```
RunSystem [Java Application] /Users/andieleonhard/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_17.0.8.v20230801-1951/jre/bin/java (Nov 8, 2024, 5:51:16 PM) [pid: 45975]
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
3
Enter inventoryID: 65
Item Unavailable. Waitlist: 1
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
4
Enter inventoryID: 65
Enter user's card number:
5
Item Unavailable. Waitlist: 1
Hold placed successfully.
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
```

(showing check availability step as well, which is not part of this use case, but just to show that this item does actually have a waitlist)

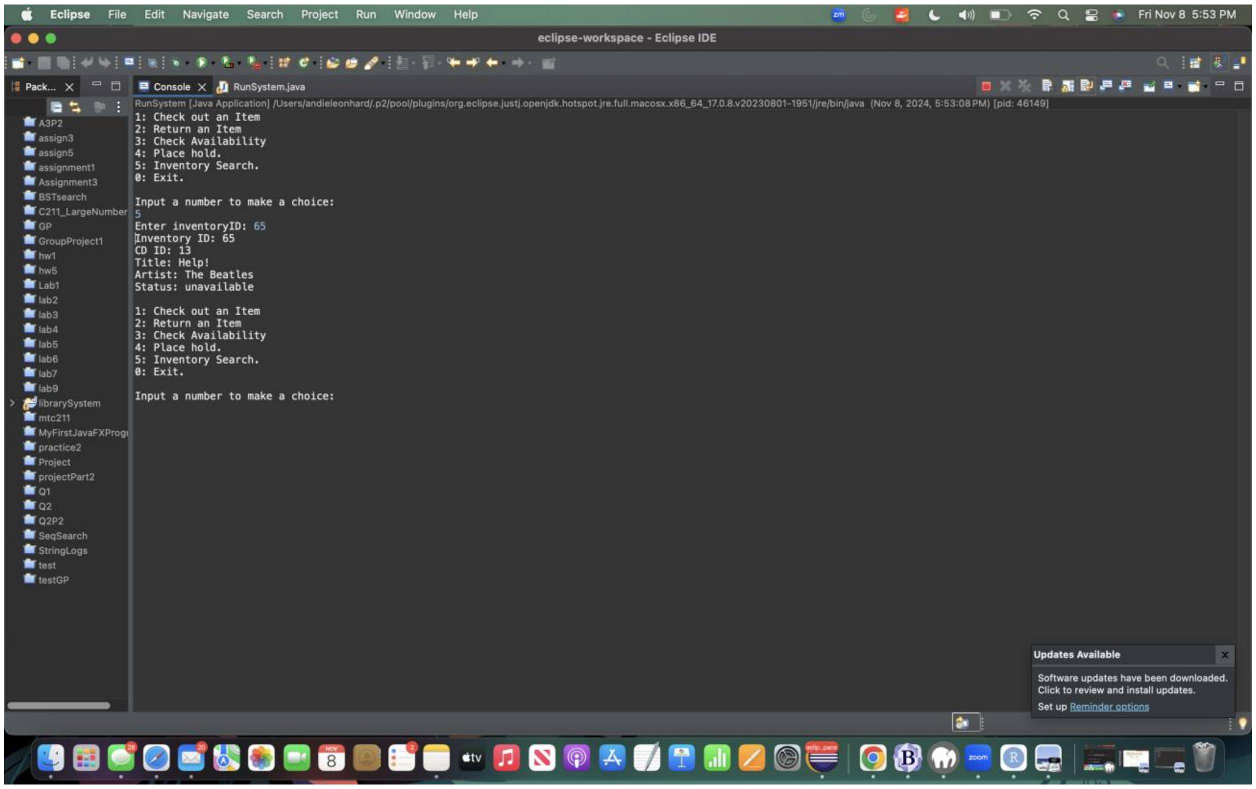
Staff: Inventory Search

- Starting at menu, enter corresponding number to inventory search
- Enter inventory ID

  Navigation Path: Menu > Input Inventory ID > Success

  Number of clicks: 0 (maybe 1 to set cursor)

  Number of Keystrokes: 2



```
RunSystem [Java Application] /Users/andieleonhard/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86_64_17.0.8.v20230801-1951/jre/bin/java (Nov 8, 2024, 5:53:08 PM) [pid: 46149]
1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
5
Enter inventoryID: 65
Inventory ID: 65
CD ID: 13
Title: Help!
Artist: The Beatles
Status: unavailable

1: Check out an Item
2: Return an Item
3: Check Availability
4: Place hold.
5: Inventory Search.
0: Exit.

Input a number to make a choice:
```

Updates Available
Software updates have been downloaded. Click to review and install updates.
Set up Reminder options

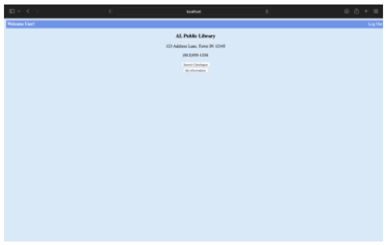
Guest: View Current Checkouts and Holds

- Start at Home. Select button for "My Information"
- On the new webpage, select the button for viewing current checkouts and holds

  Navigation Path: Home > My Information > Current Checkouts > Success

  Number of clicks: 2

  Number of Keystrokes: 0

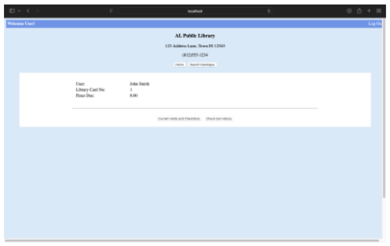


Welcome User! Log Out

AL Public Library

123 Address Lane, Town IN 12345

(812)555-1234



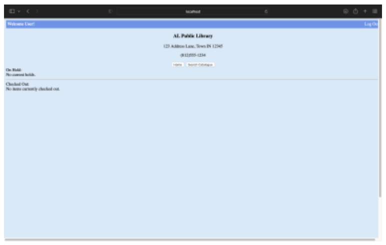
Welcome User! Log Out

AL Public Library

123 Address Lane, Town IN 12345

(812)555-1234

User: John Smith
Library Card No: 1
Fines Due: 0.00



Welcome User! Log Out

AL Public Library

123 Address Lane, Town IN 12345

(812)555-1234

On Hold:
No current holds.

Checked Out:
No items currently checked out.

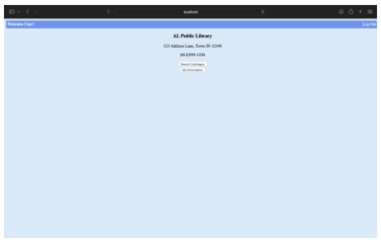
Guest: View Check Out History

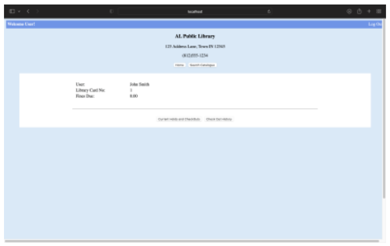
- Start at Home. Select button for "My Information"
- On the new webpage, select the button for viewing checkout history

  Navigation Path: Home > My Information > Checkout History > Success

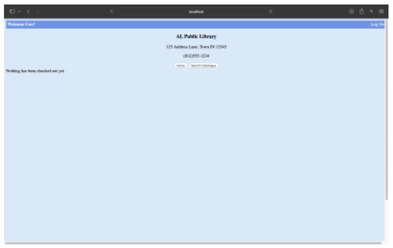
Number of clicks: 2

Number of Keystrokes: 0





User: John Smith
Library Card No: 1
Fines Due: 0.00



Nothing has been checked out yet.

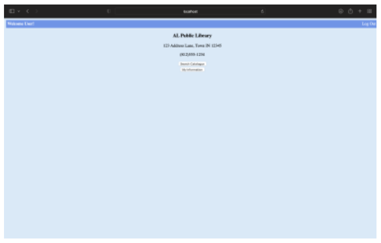
Guest: Inventory Search

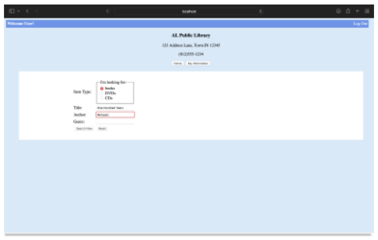
- Start at Home. Select button for "Search Catalogue"
- On the new webpage, select the radio button for the desired item type
- Input information in the search fields.
- Select the button for "Search"
- View results on new webpage

Navigation Path: Home > Catalogue Search > Enter Information > Success

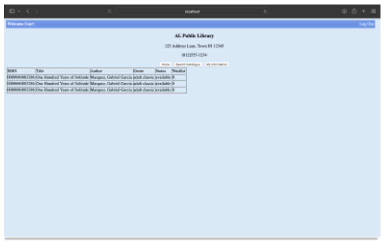
Number of clicks: 3+ (may be more to enter cursor into different input fields)

Number of Keystrokes: 1+ (depends on input search criteria





Item Type: books
Title: One Hundred Years
Author: Marquez
Genre:



ISBN	Title	Author	Genre	Status	Waitlist
[illegible]	One Hundred Years of Solitude	Marquez, Gabriel Garcia	adult classic	available	0
[illegible]	One Hundred Years of Solitude	Marquez, Gabriel Garcia	adult classic	available	0
[illegible]	One Hundred Years of Solitude	Marquez, Gabriel Garcia	adult classic	available	0

[3]This interface will be updated with a GUI that I am currently working on, but as of now, the application runs in console. This section will be updated with new images once interface design is complete.

Project Plan:

Project planning: For the purpose of this project, the system will consist of a web app, a traditional application, as well as a database, all of which will be basic enough to be hosted on my local machine, so everything should be able to run from one machine, but the app will have to be mac/pc compatible while the web app and database will be run from the web. In the case of a business, a larger system would be more practical, so scalability is important.

Development approach: As of now, the plan is to create a relational database in phpMyAdmin, the web application will be created using html/javascript with php, and the web application will primarily be programmed with java. Users will use the web app and java app interfaces to access the database indirectly to provide more security and access controls as well as a better and more usable graphical interface.

Development plan:[4]

- Complete additional modeling/planning activities for system (ongoing)
  - Progress: Most of the planning is done, but some modeling for additional use cases is still required. In addition, requirements and plans are updated as necessary as the project progresses.
- Develop basic database framework and normalize (~1-2 week)
  - Progress: Database is in working order and normalized, but additional views, triggers, or procedures may be added if required as project progresses.
- Create basic functioning application, integrating access to database (~3-4 weeks, ongoing)
  - Progress: While several basic functions have been implemented in the app, several more are still required. In addition, the GUI needs implemented as well. Integration with the database has been successful so far.
- Create web application, integrating access to database (~3-4 weeks, ongoing)
  - Progress: The web application functions at the most basic level and integration with the database has been successful so far, but additional features will be implemented as well as work on the visual appearance.
- Test functionality and usability (~1 week)
- Adapt/make changes/finalize (~1 week)

[4]Project schedule was adapted due to new due date not leaviing time for planning stages to occur prior to development beginning. In addition, original schedule was incredibly optimistic about my coding skills and time management, so development will continue concurrently with other steps.