**University of Maryland Global Campus**

**Library Management System (LMS)**

**Project Design**

**CMSC 495 7380**

**11 April 2023**

**Version 1.6**

**Group 3: Brandon Durham, Benjamin Kus, Benjamin Ramos**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **REVISION** | **DATE** | **DESCRIPTION** | **Author** |
| 1.0 | 4/4/2023 | Initial | Ben Ramos |
| 1.1 | 4/5/2023 | Addition of Pseudocode | Brandon Durham |
| 1.2 | 4/5/2023 | Document modification | Ben Ramos |
| 1.3 | 4/7/2023 | Inclusion of Event Scenarios | Ben Kus |
| 1.4 | 4/8/2023 | Modification of Pseudocode  Addition of Risk Mitigation | Brandon Durham  Ben Kus |
| 1.5 | 4/11/2023 | Risk Management | Brandon Durham |
| 1.6 | 4/11/2023 | Addition of scenario descriptions  Modification to pseudocode  Document modifications | Ben Kus  Brandon Durham  Ben Ramos |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

TABLE OF CONTENTS

1. Event Trace Scenarios
   1. Start Up Scenario
   2. Scenario 1
   3. Scenario 2
   4. Scenario 3
   5. Scenario 4
   6. Scenario 5
   7. Shut Down Scenario
   8. Error Handling Scenario
2. Class Diagram
3. Risk Mitigation
4. **Event Trace Scenario**

*Start Up Scenario*

**Diagram

Description automatically generated**

Description: The system shall provide administrator with a login screen

Precondition: Administrator met software/hardware requirements and has installed application. The administrator has the application open.

Post-condition: The administrator is at the “Admin Menu.”

*Scenario 1*

**Diagram

Description automatically generated**

Description: The system shall provide an administrator specific display to conduct admin role

Precondition: The administrator has successfully logged into the application

Post-condition: The administrator will have number of menu options to conduct roles

*Scenario 2*

**Table

Description automatically generated**

Description: The system shall provide an administrator specific display to check out books

Precondition: The administrator has successfully logged into the application and is displaying the menu options

Post-condition: The administrator will have issued a book. The admin will have connected to the database and updates were made. The connection to the database will close and the admin menu will be displayed.

*Scenario 3*

Diagram

Description automatically generated

Description: The system shall allow the admins to include library borrower’s information during issuing

Precondition: The administrator is at the “issue book display”.

Post-condition: The administrator will have connected to the database and have updated it wit the borrower’s information (First Name, Last Name, Current Address). The connection will have been closed and the database updated and the admin menu will be displayed

*Scenario 4*

**Diagram, table

Description automatically generated with medium confidence**

Description: The system shall allow the administrator to search for books based on title, author, etc.

Precondition: The administrator is at the “book search menu”.

Post-condition: The administrator will have entered the search criteria and a result will be displayed on “search results”. The menu will display current information related to the search result.

*Scenario 5*

**Diagram

Description automatically generated**

Description: The system shall allow administrators to add and remove books

Precondition: The administrator is at the “Add/Remove book menu”.

Post-condition: If the administrator is adding a book, they will have inputted the required information and have updated the database. The menu will appear with the book information that was entered. If a book is removed a, the administrator will search for the book and remove it. The database will have updated in each scenario to reflected the addition or removal.

**Shut Down Scenario**

**Diagram

Description automatically generated**

Description: The system shall provide ability for admins to log out and end a session

Precondition: The application is open, and the administrator has logged in.

Post-condition: The application will have closed. The administrator will have either closed the application which will result in it signing out or they can with the sign out which will bring the administrator back to the log in prompt.

**Error Handling Scenarios**

**Diagram

Description automatically generated**

Description: The system will identify when books are trying to be added which already exist or being searched for that don’t exist.

Precondition: The administrator has successfully logged into the application and is displaying the search menu options

Post-condition: The application will display an error that the book was previously added to the inventory and cannot be duplicated. If the book doesn’t exist and error will be displayed that the book does not exist.

1. **Class Design**

Below is the pseudocode with the various classes and subsystems.

Database tables:

Books table (Author, Title, BookID, Genre)

IssuedBooks table (Borrower info, book info)

Other:

Subsystem admin login:

Login extends JFrame{  
 admin login frame;

jPanel contentPane;

JTextFields pswdField, userNField;

Public static void main (String[] args) {  
 //launch the app

EventQueue(Later) new Runnable

Public void run() {

try{

frame is new admin login();

frame visible is true

} catch

Print stack

}

Public admin login(){

Create GUI for login screen

Set username to admin

Set password “pswd” to admin123

If both username and pswd match, allow logon

Else don’t allow logon and display error message

}

}

Subsystems addbook:

Public class addBooks{

Create addbooks btn;

Public static int save (Stringe bookID, String Title, String Author, String genre, int quantity){

Try{  
 connect to database;

Prepared statement to insert book information above into table;

Set values;

Update database;

Close connection;

} catch exception{}

Return status;

}

Return to menu;

}

Subsystem remove books:

Public class removeBooks{

Create removebooks btn;

Public static int save (String bookID, String Title, String Author, String genre, int quantity){

Try{  
 connect to database;

Prepared statement to remove or delete a book using a primary key;

Set values;

Update database;

Close connection;

} catch exception{}

Return status;

}

Ret

Subsystem issue book:

Public class issueBook(){

Create issuebooks btn;

Public static Boolean checkBook(String bookID){

Boolean is false;

Try{

connect to database;

Prepared statement to select information from books table;

Execute query;

Update status;

Close connection;

} catch exception e{}

Return status;

}

Public static int save(int bookID, int studentID, String studentName, String student contact){

Status is 0;

Try{

connect to database;

Prepared statement to insert information into issueBook table;

Execute query;

Update status;

Close connection;

} catch exception e{}

Return status;

Return to menu;

}

Update the quantity of books;

Subsystem View books

Public class viewBook(){

GUI for book display/ create view books btn;

Public static void main(String[]args)

Run this component

Try{

connect to database;

Prepared statement to select information from books table;

Execute query;

Display information in table format;

Update status;

Close connection;

} catch exception e{}

Return status;

return to menu

}

Subsystem View books

Public class viewIssuedBook(){

GUI for issued book display/ create view issued books button;

Public static void main(String[]args)

Run this component;

Try{

connect to database;

Prepared statement to select information from issuedBooks table;

Execute query;

Display information in table format;

Update status;

Close connection;

} catch exception e{}

Return status;

return to menu;

}

Subsystem return Book

Public class returnBook(){  
EventQueue for runnable component;

Public void run()

Try{

Run;

}catch( exception e){  
 }

Public returnBook(){  
 setup Gui configuration for this component

jLabel BookID;

jLbadel Student ID;

set up return book button;

try{

connect to database;

prepared statement to get quantity from books table;

update quantity from books/ issued books table;

set status to update;

close the connection;

return to menu;

}catch(exception e){}

}

Subsystem search books:

Public class searchBooks(){  
 setup Gui configuration for this component

jLabel BookID;

jLbadel other book information;

set up search book button;

try{

connect to database;

prepared statement to get some information from books table dependent on author, genre, title, etc;

display books with specified criteria;

set status to update;

close the connection;

}catch(exception e){}

1. **Risk Mitigation**

|  |  |
| --- | --- |
| **Identify risks** | **Risk mitigation** |
| Admin enters login password incorrectly | Admin will be displayed a warning message of possible locked account |
| Invalid inputs such as special characters, letter grades, negative scores or grades higher than one hundred. | System will verify usage of invalid characters during data entry |
| Password complexity requirement | Require admins to meet password complexity requirements |
| Chance of data loss/modification by admin | Display a warning message prior to deleting/saving over a record |