**Ammer Saeed**

**CMS ID: 466445**

**Department of Computing**

**CS 212: Object Oriented Programming**

**Class: BSSE-14B**

**DOCUMENTATION**

I've applied Object-Oriented Programming (OOP) principles like Encapsulation, which ensures data hiding and abstraction. To manage data access, I've employed Getter and Setter methods. Additionally, for enhanced security, I've encapsulated the Setter methods within the parameterized constructor.

For data storage, I've utilized file handling techniques leveraging technologies such as FileWriter and RandomAccessFile. These enable efficient management and retrieval of data from files.

To ensure robustness, I've implemented comprehensive error handling mechanisms. This includes techniques like Try-Catch blocks and type casting to manage and respond to potential errors effectively.

GUI also used in this Application to enhanced the user experience

**FRONT END CODE IS UPLOADED IN THE GITHUB REPOSITRY**

**(SCREENSHOTS ATTACHED IN THIS FILE)**

**USER CLASS**

public class user {

*// cant make the private because of extensive file handling*

    int userID;

    String name;

    String ContactInfo;

    int borrowedBooks=0;

    public user(int userId\_,String name\_,String ContactInfo\_,int borrowedBooks\_)

    {

       this.setBook(userId\_, name\_, ContactInfo\_, borrowedBooks\_);

    }

    public user()

    {

    }

*/////////SETTER & GETTER /////////////////*

    private void setBook(int userId\_,String name\_,String ContactInfo\_,int borrowedBooks\_)

    {

        this.userID = userId\_;

        this.name = name\_;

        this.ContactInfo = ContactInfo\_;

        this.borrowedBooks = borrowedBooks\_;

    }

}

**BOOK CLASS**

public class book {

int bookID;

String title;

String author;

String genre;

Boolean avalibilty\_status;

public book(int bookID ,String title\_ , String author\_ , String genre\_, Boolean avalibilty\_status\_ )

{

    this.setBook(bookID, title\_, author\_, genre\_, avalibilty\_status\_);

}

public book()

{

}

*/////////SETTER & GETTER /////////////////*

private void setBook(int bookID ,String title\_ , String author\_ , String genre\_, Boolean avalibilty\_status\_ )

{

    this.bookID = bookID;

    this.title = title\_;

    this.author = author\_;

    this.genre = genre\_;

    this.avalibilty\_status = avalibilty\_status\_;

}

}

**LIBRARY CLASS**

import java.util.ArrayList;

import java.util.Scanner;

import java.io.File;

import java.io.FileWriter;

import java.io.RandomAccessFile;

public class Library {

*//Encapsulation*

private static boolean checker = true;

private static boolean userchecker = true;

*//////////////////////////////////////////////*

*///////////////////METHODS////////////////////*

*//////////////////////////////////////////////*

public static void AddBook(book obj,ArrayList<book> arr)

{

 try

 {

    RandomAccessFile file = new RandomAccessFile("book\_data.txt", "rw");

    if(file.length()==0)

    {

        file.seek(0);

    }

    else{

        file.seek(file.length());

    }

    file.writeBytes(obj.bookID + "," +  obj.title + "," + obj.author + ","+obj.genre+","+obj.avalibilty\_status+"\n");

    System.out.println("book saved successfully");

    file.close();

 }

 catch(Exception e) {

    System.out.println("you faced an issue while adding book " + e);

 }

 System.out.println("------------------------------------------------------------");

System.out.println("BOOK DETAIL :\n"+ "Book id: " +obj.bookID+"\n"+ "Book Title: "+obj.title  + "\n"+"Book Author: "+obj.author+"\n"+"Book genre: "+obj.genre+"\n"+"Book Avability status: "+obj.avalibilty\_status);

System.out.println("-------------------------------------------------------------");

}

public static void AddUser(user obj,ArrayList<user> arr)

{

*// arr.add(obj);*

    try {

        RandomAccessFile userfile = new RandomAccessFile("user\_data.txt", "rw");

*//userfile.seek(0);*

        if(userfile.length()==0)

    {

        userfile.seek(0);

    }

    else{

        userfile.seek(userfile.length());

    }

        userfile.writeBytes(obj.userID + "," + obj.name + "," + obj.ContactInfo + "," + obj.borrowedBooks+"\n");

        userfile.close();

*//arr.add(obj);*

    } catch (Exception e) {

        System.out.println("you faced an issue while adding user " + e);

    }

}

public static void CheckingOut(String title ,String user,ArrayList<book> bookarr,ArrayList<user> userarr)

{

    boolean flag4=true,flag5=true;

    for(book Bookvalue : bookarr)

            {

                if(Bookvalue.title.equalsIgnoreCase(title) && Bookvalue.avalibilty\_status.equals(true))

                for(user userValue : userarr)

                {   flag4=false;

                {

                    if(userValue.name.equalsIgnoreCase(user) )

                    {   userValue.borrowedBooks++;

                        flag5=false;

                        System.out.println(user + " checked out " + title);

                        Bookvalue.avalibilty\_status=false;

                        clearBookFile("book\_data.txt");

                        UpdateBookFile(bookarr,"book\_data.txt");

                        clearUserFile("user\_data.txt");

                        UpdateUserFile(userarr,"user\_data.txt");

                        System.out.println("New status: " + Bookvalue.avalibilty\_status);

                    }

                }

                }

            }

            if(flag4 == true)

            {

                System.out.println("User doesnt found");

                System.out.println("--------------------------------------");

            }

            else if(flag5 == true)

            {

                System.out.println("Book doesnt found");

                System.out.println("--------------------------------------");

            }

}

public static void ReturningBook (String book, String  user,ArrayList<book> bookarr, ArrayList<user> userarr)

{

    boolean flag6 = true , flag7 = true;

    for(user value2 : userarr)

    {

        if(value2.name.equalsIgnoreCase(user))

        {   value2.borrowedBooks--;

            flag6 = false;

            for(book value1:bookarr)

            {

                if(value1.avalibilty\_status.equals(false))

                {   flag7 = false;

                    System.out.println(user + " returned out " + book);

                    value1.avalibilty\_status=true;

                    System.out.println("New status: " + value1.avalibilty\_status);

                    clearBookFile("book\_data.txt");

                    UpdateBookFile(bookarr, "book\_data.txt");

                    clearUserFile("user\_data.txt");

                    UpdateUserFile(userarr,"user\_data.txt");

                }

            }

        }

    }

    if(flag6==true)

    {

        System.out.println("User doesnt found");

        System.out.println("--------------------------------------");

    }

    else if(flag7==true)

    {

        System.out.println("book doesnt found");

        System.out.println("--------------------------------------");

    }

}

public static void SearchBook(String Book\_Or\_Author,ArrayList<book> bookarr)

{   boolean flag8=true;

    for(book bookvalue : bookarr)

    {

        if(bookvalue.title.equalsIgnoreCase(Book\_Or\_Author) || bookvalue.author.equalsIgnoreCase(Book\_Or\_Author) )

        {   flag8 = false;

            System.out.println("Book details: \n"+"BOOK ID:" + bookvalue.bookID+"\n"+"BOOK TITLE: "+bookvalue.title+"\n"+"BOOK VALUE: "+ bookvalue.author+"\n"+"BOOK GENRE: "+bookvalue.genre+"\n"+"BOOK AVABILITY STATUS: "+bookvalue.avalibilty\_status);

            System.out.println("--------------------------------------------------");

        }

    }

    if(flag8==true)

    {

        System.out.println("Book doesnt found");

        System.out.println("--------------------------------------");

    }

}

*//////////////////LIBRARY HANDLING METHODS///////////////////////*

public static void UserRestoringDatainArray(ArrayList<user> arr)

{

    try (Scanner scanner = new Scanner(new File("user\_data.txt"))){

        while(scanner.hasNextLine())

        {

            String line = scanner.nextLine();

            String[] parts = line.split(",");

            user obj  = new user();

            obj.userID = Integer.parseInt(parts[0]);

            obj.name = parts[1];

            obj.ContactInfo = parts[2];

            obj.borrowedBooks = Integer.parseInt(parts[3]);

            arr.add(obj);

        }

    }

    catch(Exception e) {

*// System.out.println("you faced an issue while restoring users " + e);*

    }

}

public static void BookRestoringDatainArray(ArrayList<book> arr)

{

    int bookID;

    String title,author,genre;

    boolean availability = false;

    try (Scanner scanner = new Scanner(new File("book\_data.txt"))){

        while(scanner.hasNextLine())

        {

            String line = scanner.nextLine();

            String[] parts = line.split(",");

            bookID = Integer.parseInt(parts[0]);

            title = parts[1];

            author = parts[2];

            genre = parts[3];

            availability = Boolean.parseBoolean(parts[4]);

            book obj  = new book(bookID,title,author,genre,availability);

            arr.add(obj);

        }

    }

    catch(Exception e) {

    }

}

public static void UpdateBookFile(ArrayList<book> bookarr, String filename) {

    try {

        RandomAccessFile file = new RandomAccessFile("book\_data.txt", "rw");

*//userfile.seek(0);*

        file.seek(0);

        for (book valueBook : bookarr) {

            file.writeBytes(valueBook.bookID + "," + valueBook.title + "," +

                                  valueBook.author + "," + valueBook.genre + "," +

                                  valueBook.avalibilty\_status + "\n");

        }

        file.close();

*//arr.add(obj);*

    } catch (Exception e) {

        System.out.println("you faced an issue while adding user " + e);

    }

}

public static void clearBookFile(String fileName) {

    try {

        File file = new File(fileName);

        if (file.exists()) {

*// file.delete();*

*// System.out.println("File deleted successfully.");*

            if(file.delete())

            {

                if(file.createNewFile())

                {

                    System.out.println("new file created");

                    FileWriter writer = new FileWriter(fileName);

                    writer.write(0+","+"null"+","+"null"+","+"null" + false);

                }

            }

        } else {

            System.out.println("File does not exist.");

        }

    } catch (Exception e) {

        System.out.println("Error deleting file: " + e.getMessage());

    }

}

public static void clearUserFile(String fileName) {

    try {

        File file = new File(fileName);

        if (file.exists()) {

*// file.delete();*

*// System.out.println("File deleted successfully.");*

            if(file.delete())

            {

                if(file.createNewFile())

                {

                    System.out.println("new file created");

                    FileWriter writer = new FileWriter(fileName);

                    writer.write(0+","+"null"+","+"null"+","+0);

                }

            }

        } else {

            System.out.println("File does not exist.");

        }

    } catch (Exception e) {

        System.out.println("Error deleting file: " + e.getMessage());

    }

}

public static void UpdateUserFile(ArrayList<user> userarr, String filename) {

try {

    RandomAccessFile userfile = new RandomAccessFile("user\_data.txt", "rw");

    userfile.seek(0);

    for (user valueUser : userarr) {

        userfile.writeBytes(valueUser.userID + "," + valueUser.name + "," +

        valueUser.ContactInfo + "," + valueUser.borrowedBooks + "\n");

        }

        userfile.close();

}

catch (Exception e) {

    System.out.println("you faced an issue while adding user " + e);

    }

}

 public static void displayLibrary(ArrayList<book> Book\_array,ArrayList<user> User\_array)

 {

*///lOCAL VARIABLES///*

    String title , author, genre,username,contactInfo\_,input="0";

    String book\_id,user\_id;

    Scanner sc = new Scanner(System.in);

    boolean checker = true;

    while(checker)

    {

    System.out.println("|1.Add the Book");

    System.out.println("|2.Add the User");

    System.out.println("|3.Checking out");

    System.out.println("|4.Returning book");

    System.out.println("|5.Search book details");

    System.out.println("|6.Exit");

    System.out.print("=> ");

    input = sc.nextLine();

    try {

        int temp;

        temp = Integer.parseInt(input);

        if(temp<7&&temp>0)

        {

            checker =true;

        switch (Integer.parseInt(input)) {

        case 1:

        System.out.println("Enter your Book ID:");

        book\_id = sc.nextLine();

        try {

            int flag2;

            flag2 = Integer.parseInt(book\_id);

        } catch (Exception e) {

            System.out.println("Invalid book id");

            System.out.println("--------------------------------------");

        }

        System.out.println("Enter your Book title:");

        title = sc.nextLine();

        System.out.println("Enter book author name:");

        author = sc.nextLine();

        System.out.println("Enter book genre:");

        genre = sc.nextLine();

            if(!genre.matches("[a-zA-Z]+"))

            {

                System.out.println("invalid input!");

                System.out.println("--------------------------------------");

            }

            else {

                book bk = new book(Integer.parseInt(book\_id),title,author,genre,true);

                Library.AddBook(bk,Book\_array);

            }

            break;

        case 2:

        System.out.println("Enter User ID:");

        user\_id = sc.nextLine();

        try {

            int flag3;

            flag3 = Integer.parseInt(user\_id);

        } catch (Exception e) {

            System.out.println("Invalid input");

        }

        System.out.println("Enter User name:");

        username = sc.nextLine();

        System.out.println("Enter User Contact info:");

        contactInfo\_ = sc.nextLine(); *//Contact Info*

        user us = new user(Integer.parseInt(user\_id),username,contactInfo\_,0);

        Library.AddUser(us, User\_array);

        break;

        case 3:

        Library.BookRestoringDatainArray(Book\_array);

        Library.UserRestoringDatainArray(User\_array);

        System.out.println("Enter your Book title:");

        title = sc.nextLine();

        System.out.println("Enter User name:");

        username = sc.nextLine();

        Library.CheckingOut(title, username, Book\_array, User\_array);

        break;

        case 4:

        Library.BookRestoringDatainArray(Book\_array);

        Library.UserRestoringDatainArray(User\_array);

        System.out.println("Enter your Book title:");

        title = sc.nextLine();

        System.out.println("Enter User name:");

        username = sc.nextLine();

        Library.ReturningBook(title, username, Book\_array, User\_array);

        break;

        case 5:

        Library.BookRestoringDatainArray(Book\_array);

        Library.UserRestoringDatainArray(User\_array);

        System.out.println("Enter Book name: ");

        title = sc.nextLine();

        Library.SearchBook(title,Book\_array);

        break;

         case 6:

         System.exit(0);

    }

    }

    else {

        System.out.println("invalid input");

        System.out.println("--------------------------------------");

    }

    }

    catch (Exception e) {

        System.out.println("Invalid Input");

        System.out.println("--------------------------------------");

    }

    }

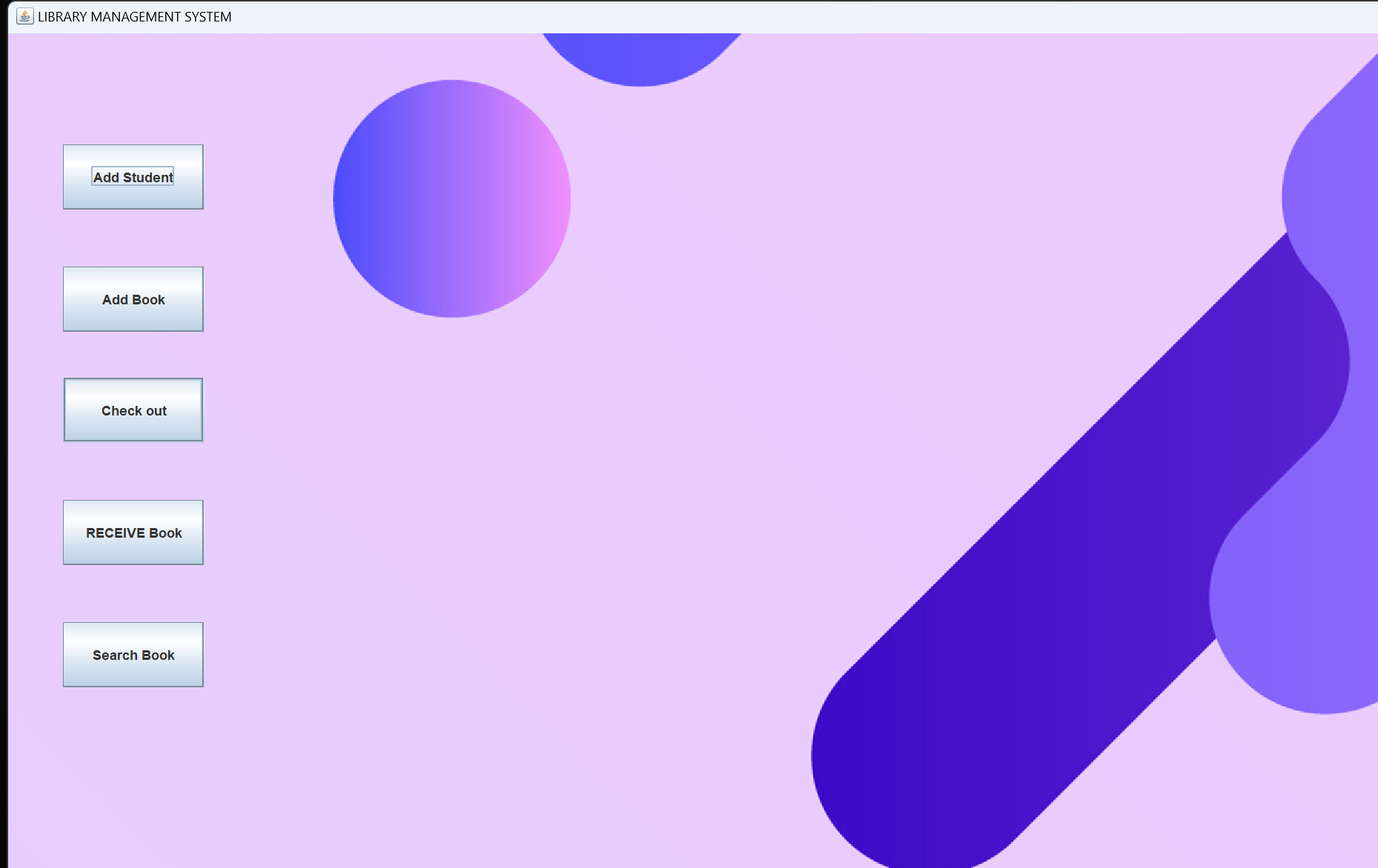
 }

}

**MAIN CLASS CODE**

**OUTPUT**

**HOME PAGE:**

****

**ADD USER:**

**A screenshot of a computer

Description automatically generated**

**ADD BOOK:**

**A screenshot of a computer

Description automatically generated**

**CHECKOUT:**

**A screenshot of a computer

Description automatically generated**

**RETURN BOOK:**

**A screenshot of a computer

Description automatically generated**

**SEARCH BOOK: A screenshot of a computer

Description automatically generated**

**FILE HANDLING :**

**BEFORE:**

**A screenshot of a computer

Description automatically generated**

**AFTER:**

**A screenshot of a computer

Description automatically generated**

**BEFORE:**

A screenshot of a computer

Description automatically generated

**AFTER:**

**A screenshot of a computer

Description automatically generated**

**ERROR HANDLING**

