Fundamental in Programming

NAME – M.G.CHULI CHULANI NIKLESHA
STUDENT ID – CL/HDNET/CMU/41/10

Assignment Cover Sheet

Qualification		**Module Number and Title**
HD in Computing and Software Engineering /Network Technology and Cyber Security		CSE 4002 Fundamentals in Programming
Student Name & No.		**Assessor**
Name – M.G.Chuli Chulani Niklesha CL/HDNET/CMU/41/10		Mrs. Vindya
Hand over date		**Submission Date**
Assessment type **Coursework**	**Duration/Length of Assessment Type** Software Submission and demonstration	**Weighting of Assessment** 100%

Learner declaration
I am Chuli Chulani Niklesha(CL/HDNET/CMU/41/10) certify that the work submitted for this assignment is my own and research sources are fully acknowledged.

Marks Awarded			
First assessor			
IV marks			
Agreed grade			
Signature of the assessor		Date	

FEEDBACK FORM
INTERNATIONAL COLLEGE OF BUSINESS & TECHNOLOGY

Module: CSE 4002

Student: M.G. Chuli Chulani Niklesha

Assessor: Mrs. Vindya

Assignment: Nethra bookshop automation system

Assessor Feedback:

Marks Awarded:

Contents

Introduction

Nethra is one of the famous bookshops in Kandy. Until now the system of Nethra Bookshop was maintained manually. This assignment gives a better idea about automating that manually operated system. First, the SRS document is provided. The second is to outline the operations that take place in various unique sections of the constructed system. Lastly, a user acceptance test document and a test plan are added to verify that the system is developed correctly and that it is suitable for the intended user.

Acknowledgment

I want to express my gratitude to Ms. Vindya, our Fundamentals in Programming lecturer, for her guidance and assistance with this work. Her notes proved to be a valuable resource in finishing this task.

Task 01

1.1 System Requirement Specification Document

1.1.1 Introduction

Nethra is one of the famous bookshops in Kandy City, which provides a vast range of valuable books for schools, and educational institutes as well as to daily customers. They used manual methods to run the entire business process from the cashier to the back office. However due to some factors, they have decided to automate their entire system.

1. Rapidly increasing customer inquiries.
2. Customers place large orders and request quotes quickly, but manual quotes are often inaccurate.

These can be pointed out as reasons for automating the system. The purpose of this system is to automate the entire manual work and save time and cost. This automated system will be able to provide a better service to their customers. The SRS lists the primary functional requirements, non-functional requirements, and system limitations for the proposed system. It will be very beneficial for the development process that the SRS additionally includes pseudocode and flowcharts for every major system function.

1.1.2 Purpose

Nethra's manual system is riddled with errors. such is having trouble keeping track of orders, doing calculations wrong, copying data, being ineffective, having unhappy customers, and misplacing paperwork. They have made the financial decision to automate their manual system to address these issues.

1.1.3 Functional Requirements

As a book shop, Nethra’s automated system should have different types of functional needs. Some of the main parts are below.

1. Manage Books
2. Create Orders
3. Print receipt with discount

- Manage Books

  Nethra's automated system has two stakeholders. They are admins and users. This book manage part is done by administrators. Here, admins can insert books, search the details of inserted books, update details, delete books and view the orders.

- Create Orders

  Actually, this is done by the users. They can choose the books from our store and give the number of books that he want as the quantity. The system has given to the admin the ability to monitor the orders placed by these users through his dashboard.

- Print Receipts

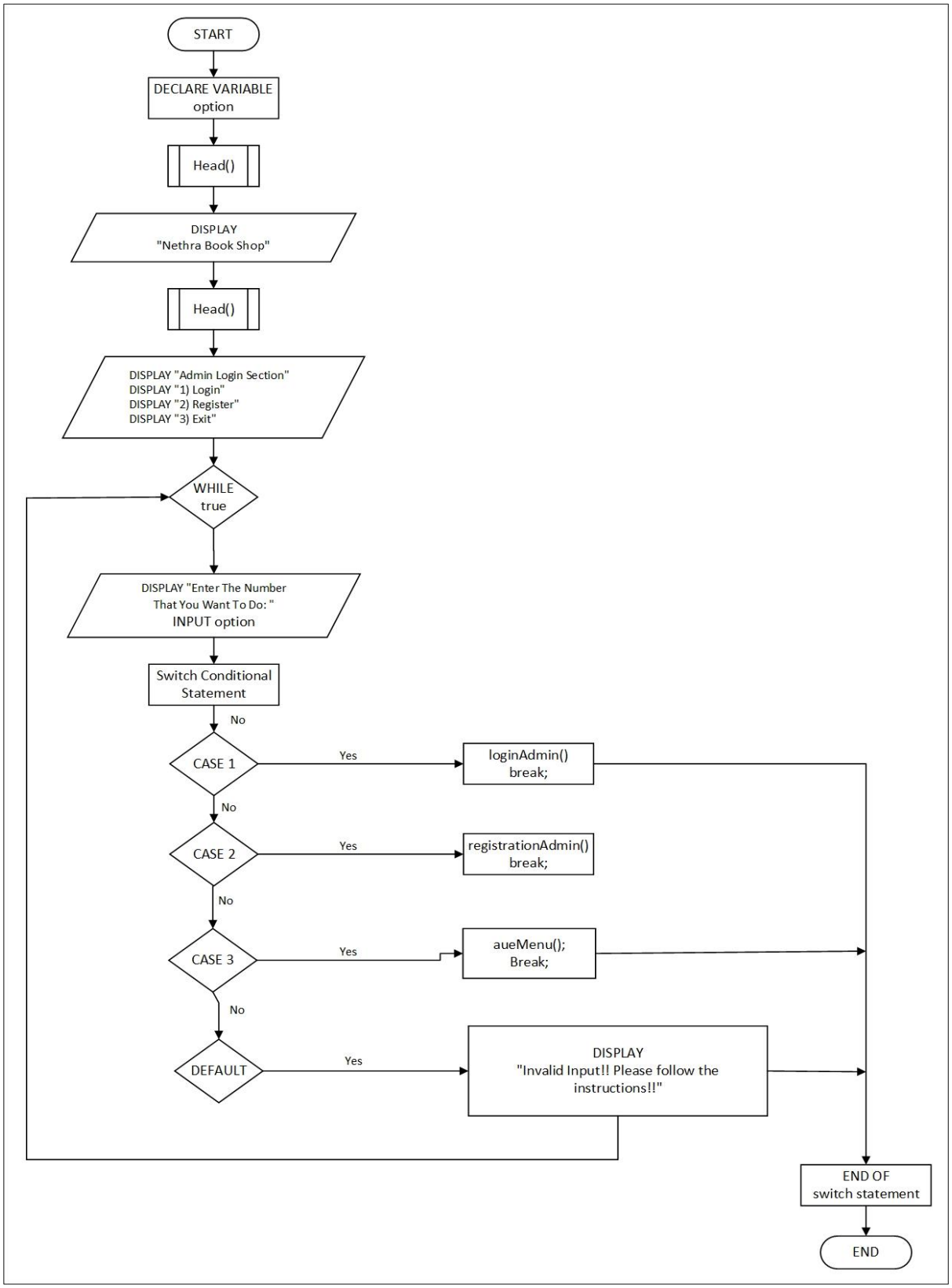
  After giving the quantity the system asked form the user that do you want to add another book. Then the user can add a book or get the receipt for his order. Here, if the user buys books worth more than one thousand and hundred rupees, he will get a 5% discount. And also, the user buy books worth than three thousand rupees, he will get a 10% discount and he buy books worth than five thousand rupees; he will get a 20% discount.

1.1.4 Non-Functional Requirements

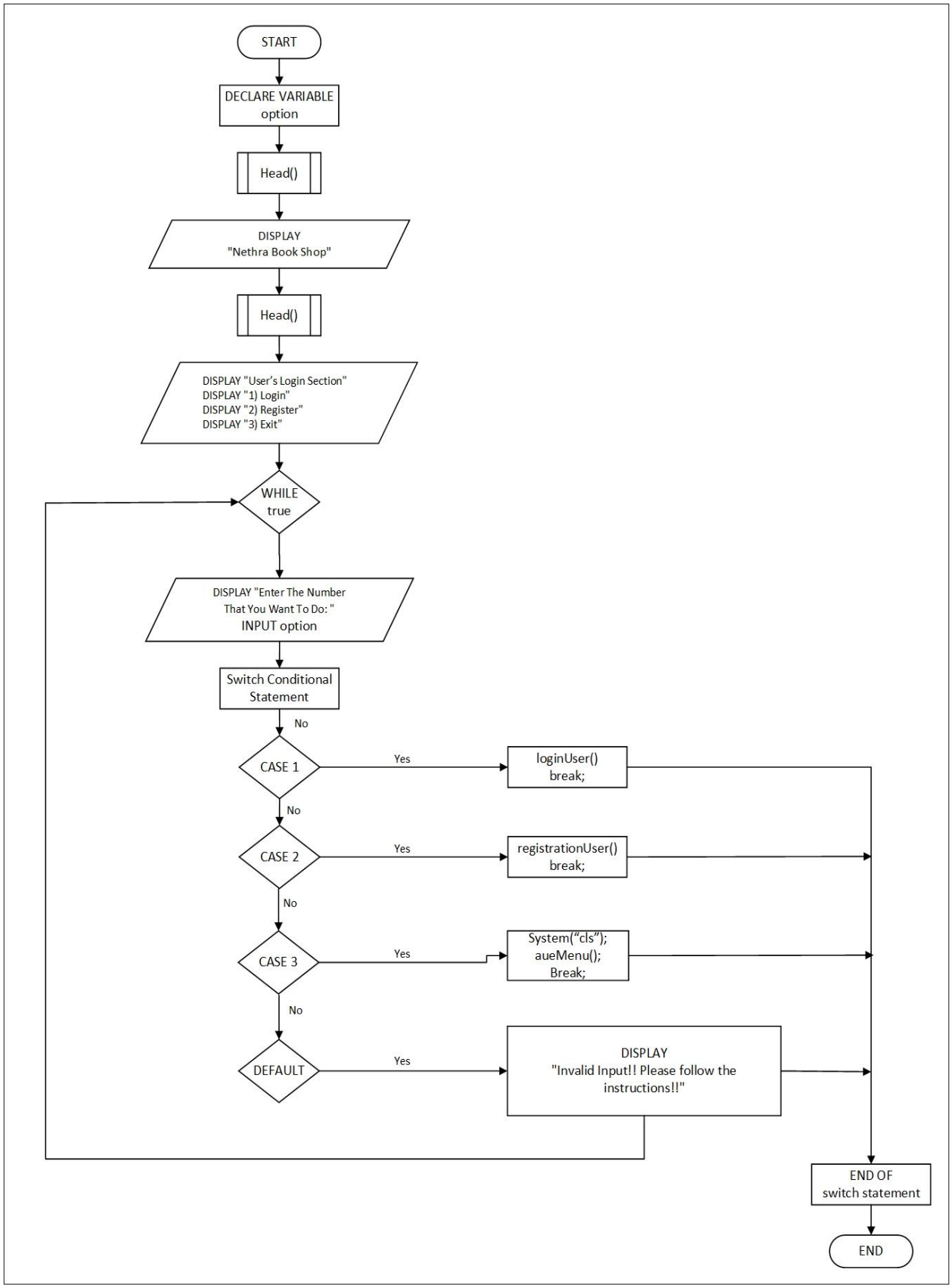
1. The system ought to respond to all user requests, including login, book addition, book updating, bill generation, and so on, in less than three seconds.
2. To promote ease of use, the user interface needs to be user-friendly.
3. The system ought to offer unambiguous error notifications and prompts for remedial measures.

1.2 Flowcharts

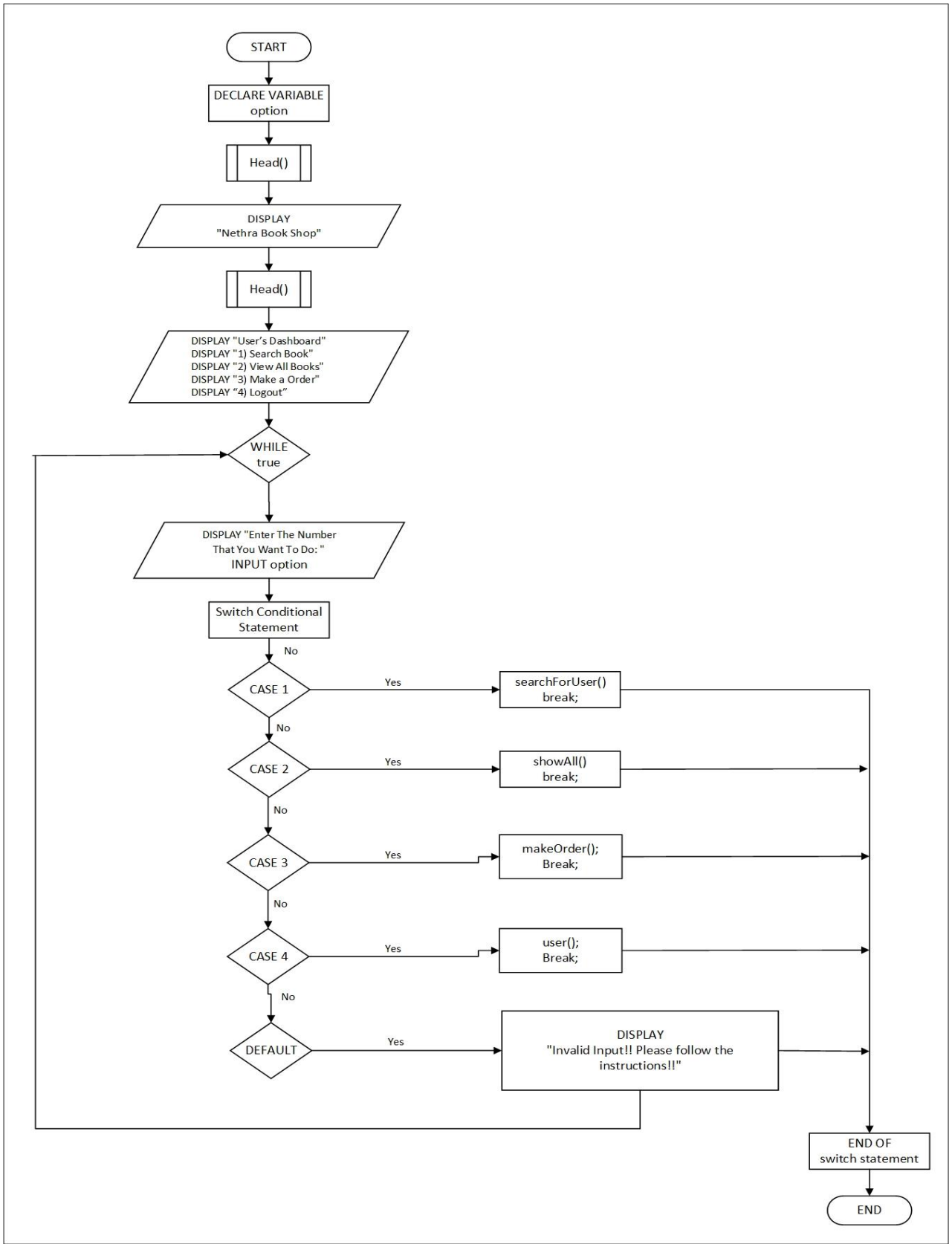
1.2.1 admin()



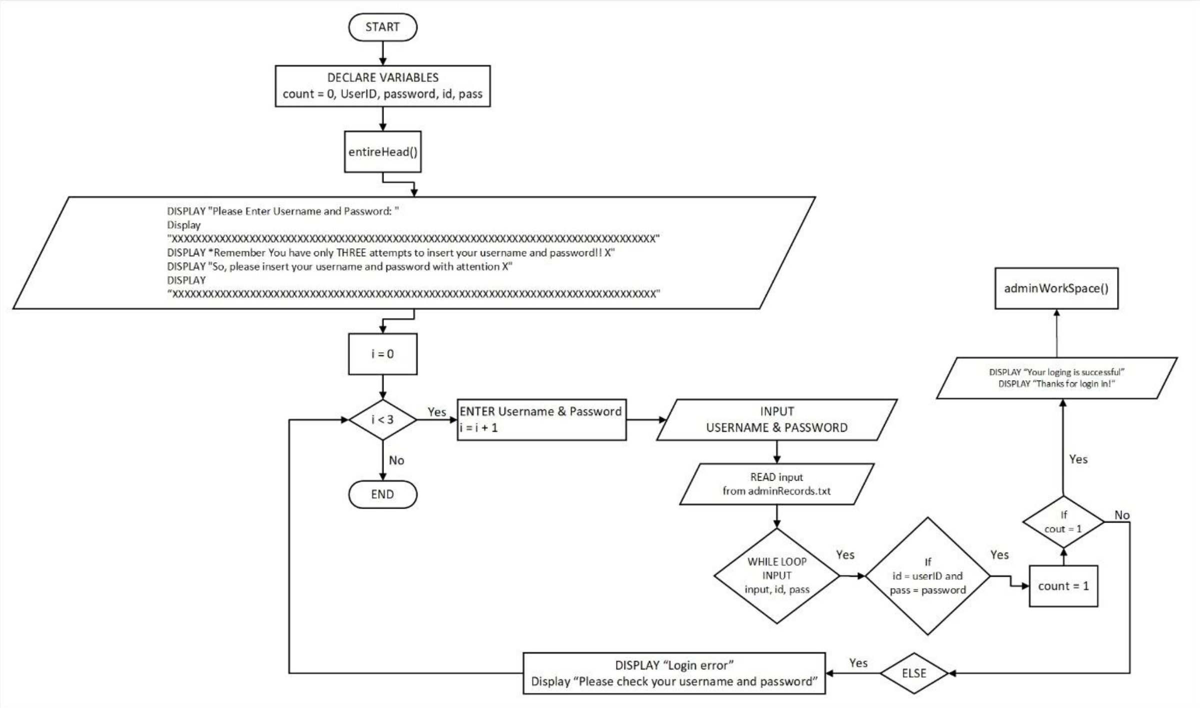
1.2.2 user()


START
DECLARE VARIABLE
option
Head()
DISPLAY
"Nethra Book Shop"
Head()
DISPLAY "User's Login Section"
DISPLAY "1) Login"
DISPLAY "2) Register"
DISPLAY "3) Exit"
WHILE
true
DISPLAY "Enter The Number
That You Want To Do: "
INPUT option
Switch Conditional
Statement
No
CASE 1
Yes
loginUser()
break;
No
CASE 2
Yes
registrationUser()
break;
No
CASE 3
Yes
System("cls");
aueMenu();
Break;
No
DEFAULT
Yes
DISPLAY
"Invalid Input!! Please follow the
instructions!!"
END OF
switch statement
END

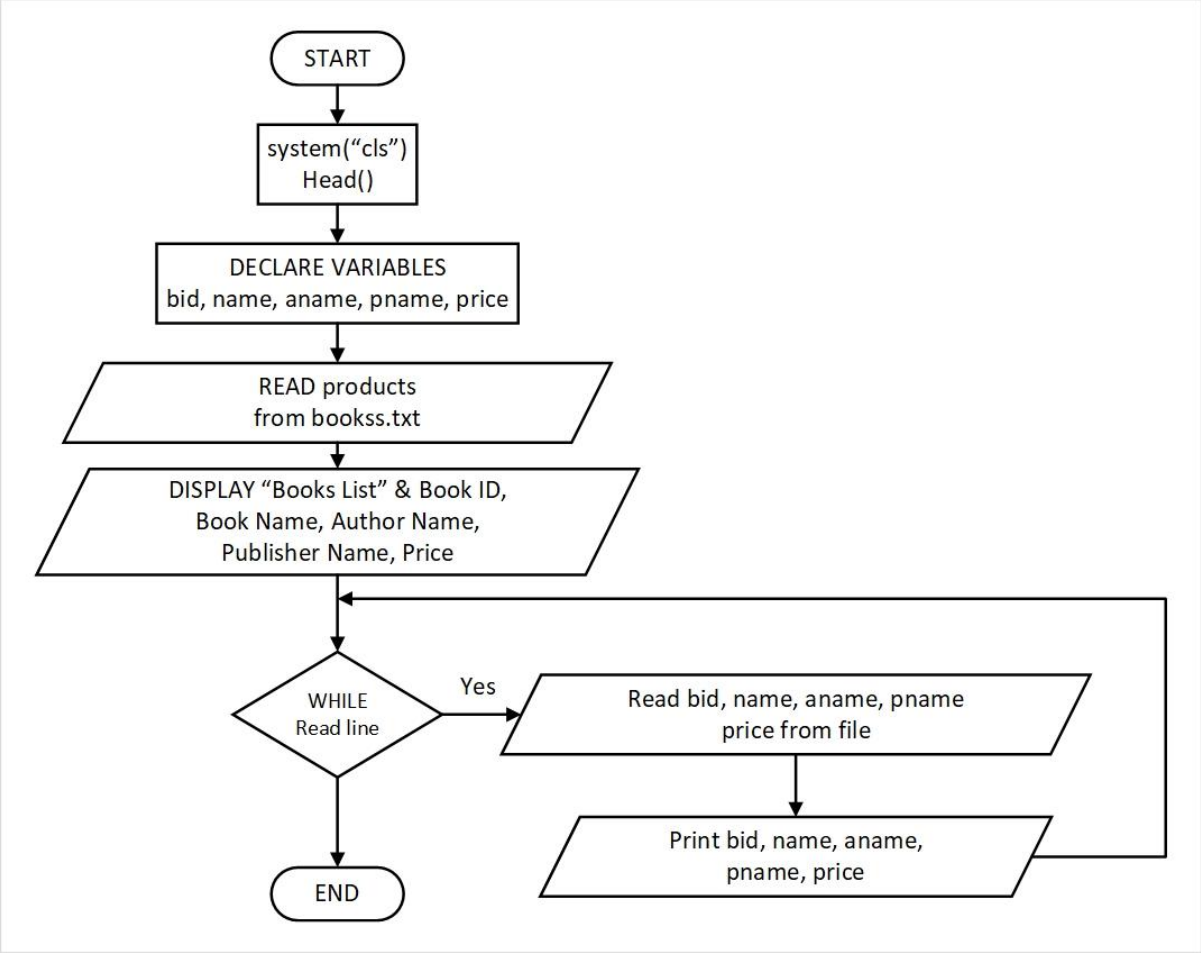
1.2.3 userWorkSpace()



1.2.4 loginAdmin()



1.2.5 table()


START
system("cls")
Head()
DECLARE VARIABLES
bid, name, aname, pname, price
READ products
from bookss.txt
DISPLAY "Books List" & Book ID,
Book Name, Author Name,
Publisher Name, Price
WHILE
Read line
Yes
Read bid, name, aname, pname
price from file
Print bid, name, aname,
pname, price
END

1.3 Pseudocodes

1.3.1 loginAdmin()

```
FUNCTION loginAdmin()
        DECLARE count = 0 , userID = "" , password = "" , id = "" , pass = ""
        DISPLAY "Please Enter Username and Password:"
        DISPLAY "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        DISPLAY "Remember you have only three attempts to insert your username and
password!"
        DISPLAY "So, please insert your username and password with attention"
        DISPLAY "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        FOR i = 1 to 3
                DISPLAY "USERNAME: "
                READ userID
                DISPLAY "PASSWORD: "
                READ password
                OPEN input file "adminRecords.txt"
                WHILE (Read id, pass from input file)
                                IF (id is equal to userID and pass is equal to password)
                                SET count to 1
                                END IF
                END WHILE
                CLOSE input file
                IF (count is equal to 1)
                                CALL Head()
                                DISPLAY "Your login is successful, " + userID
                                CALL adminWorkSpace()
                ELSE
                                DISPLAY "Login error"
                                DISPLAY "Please check your username and password"
                END IF
        END FOR
ENDFUNCTION
```

1.3.2 loginUser()

```
FUNCTION loginUser()
        DECLARE count = 0 , userID = "" , password = "" , id = "" , pass = ""
        DISPLAY "Please Enter Username and Password:"
        DISPLAY "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        DISPLAY "Remember you have only three attempts to insert your username and
password!"
        DISPLAY "So, please insert your username and password with attention"
        DISPLAY "XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
        FOR i = 1 to 3
                DISPLAY "USERNAME: "
                READ userID
                DISPLAY "PASSWORD: "
                READ password
                OPEN input file "userRecords.txt"
                WHILE (Read id, pass from input file)
                                IF (id is equal to userID and pass is equal to password)
                                SET count to 1
                                END IF
                END WHILE
                CLOSE input file
                IF (count is equal to 1)
                                CALL Head()
                                DISPLAY "Your login is successful, " + userID
                                CALL userWorkSpace()
                ELSE
                                DISPLAY "Login error"
                                DISPLAY "Please check your username and password"
                END IF
        END FOR
ENDFUNCTION
```

1.3.3 table()

```
FUNCTION table()
        CLEAR SCREEN
        CALL  Head() FUNCTION
        DECLARE VARIABLES: bid AS INT, name, aname, pname AS STRING, pprice AS
        FLOAT
        Open FILE "books.txt" IN INPUT MODE AND ASSIGN IT TO THE 'products'
        FILE STREAM
        DISPLAY "Products List" with appropriate formatting
        DISPLAY AS TABLE HEADERS: "Book ID", “Book Name", “Author Name”,
        “Publisher Name”, "Price"
                WHILE (products IS OPEN)
                        READ bid, name, aname, pname and price
                        DISPLAY name, aname, pname and price
                END WHILE
        CLOSE 'books' FILE
END FUNCTION
```

TASK 02

2.1 Implementation Details

(a) Screenshots of Sequence / Selections / Repetition

Sequence – 01

```
cout << "\n\n\t== Insert New Book ==";
cout << "\n\n ENTER BOOK ID: ";
cin >> insertBook[i].bid;
cout << "\n ENTER BOOK NAME: ";
cin >> insertBook[i].name;
cout << "\n ENTER AUTHOR NAME: ";
cin >> insertBook[i].aname;
cout << "\n ENTER PUBLISHER NAME: ";
cin >> insertBook[i].pname;
cout << "\n ENTER BOOK PRICE: Rs.";
cin >> insertBook[i].price;

ofstream bookshop("books.txt", ios::app); //Writing Books in book file
bookshop << insertBook[i].bid << " " << insertBook[i].name << " "
    << insertBook[i].aname << " " << insertBook[i].pname << " " << insertBook[i].price << endl;

ofstream showAllBooks("allbooks.txt", ios::app);
showAllBooks << endl << "Book ID: " << insertBook[i].bid << endl
    << "Book Name: " << insertBook[i].name << endl
    << "Author Name: " << insertBook[i].aname << endl
    << "Publisher Name: " << insertBook[i].pname << endl
    << "Price: Rs." << insertBook[i].price << endl;

i++;
cout << "\n\n\t " << ch << ch << " INSERT NEW BOOK SUCCESSFULLY!! " << "\n\n\n";
```

(const char [2])" "
Search Online

- In inserting a book into the system, the admin has to give the details of the relevant book. Such as book ID, name, author name, publisher name, and price, etc. After inserting these details by the user, the information is written in a file called “books.txt”. And the system gives an ability to the admin to show all books in instantly. Therefore that inserted information is also written in a file called “allbooks.txt”.

Sequence – 02

```
string ruserID, rpassword, rid, rpass;
int option;
system("cls");
Head();
cout << " NETHRA BOOK SHOP ";
Head();

cout << "\n\n\t\t Enter The username: ";
cin >> ruserID;
cout << "\t\t Enter the password: ";
cin >> rpassword;

ofstream f1("adminRecords.txt", ios::app);
f1 << ruserID << " " << rpassword << endl;

cout << "\n\t\t\t    Registration is successful! \n\n";
```

- This is a part of the registration function. Here user has to enter a username and password. And that information is written in a file called “adminRecords.txt”. And system shows a message called “Registration is successful”.

Selection – 01

```
switch (option) {
case 1:
    searchForUser();
    break;

case 2:
    showAll();
    break;

case 3:
    makeOrder();
    break;

case 4:
    user();
    break;

default:
    cout << "\t\t Invalid Input!! Please follow the instructions!! \n";
    continue;
}
```

- This is a code part of user dashboard. Actually, this is an example for a selection type of switch case. There are four cases in this switch case. The first case is searching books. The second is to show all books. The third one is making an order and the fourth one is logout (user). If the user enters a number that is out of this range, the system generates a message called “Invalid Input”.

Selection – 02

```
cout << "\n\n\t" << "Do you want to Update another Book ? (y / n) : ";
cin >> option;

if (option == "y") {
    break;
}

else if (option == "n") {
    system("cls");
    adminWorkSpace();
}

else {
    cout << "\t\t Wrong Input!! Please Follow the Instructions...!!";
    continue;
}
```

- This is a part of the update book function. Here system asks form the system that "Do you want to update another book?". Actually, this is an example of nested else if. If the user press "y", the statement will break and execute the code part of out the statement. If the user enters "n", the system calls to the adminWorkSpace function.

Repetition – 01

```
while (true) {
    cout << "\n\n\t Enter the number that you want to DO: ";
    cin >> option;

    switch (option) {
    case 1:
        loginAdmin();
        break;

    case 2:
        registrationAdmin();
        break;

    case 3:
        system("cls");
        aueMenu();
        break;

    default:
        cout << "\t\t Invalid Input!! Please follow the instructions!! \n";
        continue;
    }
}
```

- This is an example of a while loop. Actually, this can be considered an always-true loop. Here first system asks from the user that Enter the number that you want to do. After that run the switch case statement. There is a continue keyword here. Continue and break statements are available only in loops. That's why I used a infinite while loop here.

Repetition – 02

```
for (int i = 0; i < 3; i++) {
    cout << "\t\t\t USERNAME: ";
    cin >> userID;
    cout << "\t\t\t PASSWORD: ";
    cin >> password;

    ifstream input("adminRecords.txt");

    while (input >> id >> pass) {
        if (id == userID && pass == password) {
            count = 1;
        }
    }
    input.close();

    if (count == 1) {
        system("cls");
        cout << "\n " << userID << " Your login is successfull \n Thanks for login in! \n\n";
        adminWorkSpace();
    }
    else {
        cout << "\n\t Login error \n\t Please check your username and password \n\n";
    }
}
```

- This is an example of a for loop. Actually this is a part of login function. Here loop execute three times. System give three opportunities to user to give the username and password correctly. If the user can not give the correct username and password within these three attempts, System end automatically and the user have to run the program again.

b) Screenshots of Functions

Function – 01

```
void aueMenu();
void user();
void admin();
void search();
void insert();
void loginAdmin();
void registrationAdmin();
void adminWorkSpace();
void loginUser();
void registrationUser();
void userWorkSpace();
```

- These are some functions for using to modularize the entire code. Here aeuMenu is used to show first menu. And user function is used to build user's part and etc.

Function – 02

```
void registrationUser() {
    string ruserID, rpassword, rid, rpass;
    int option;
    system("cls");
    Head();
    cout << " NETHRA BOOK SHOP ";
    Head();
    cout << "\n\n\t\t Enter The username: ";
    cin >> ruserID;
    cout << "\t\t Enter the password: ";
    cin >> rpassword;
    ofstream f1("userRecords.txt", ios::app);
    f1 << ruserID << " " << rpassword << endl;
    cout << "\n\t\t\t    Registration is successful! \n\n";
    while (true) {
        cout << "\n\n\t Please Enter 1 to move to the Login Page: ";
        cin >> option;
        if (option == 1) {
            user();
        }
        else {
            cout << "\t\t Wrong Input!!! Please Insert Correct Input!!";
            continue;
        }
    }
}
```

- This is the function that is used to execute the registration process of the users. Here first user has to enter his username and password. After entering the username and password, they are written in a file called "userRecords.txt".

c) Screenshots of Arrays

Arrays – 01

```
BookShop insertBook[25];
string option;
system("cls");
Head();
cout << " NETHRA BOOK SHOP ";
Head();

while (true) {
    cout << "\n\n\t== Insert New Book ==";
    cout << "\n\n ENTER BOOK ID: ";
    cin >> insertBook[i].bid;
    cout << "\n ENTER BOOK NAME: ";
    cin >> insertBook[i].name;
    cout << "\n ENTER AUTHOR NAME: ";
    cin >> insertBook[i].aname;
    cout << "\n ENTER PUBLISHER NAME: ";
    cin >> insertBook[i].pname;
    cout << "\n ENTER BOOK PRICE: Rs.";
    cin >> insertBook[i].price;
```

- This is a part of the insert book function. There is an array called insertBook here. When the user gives the data, this array store those data temporarily.

Array – 02

```
ofstream orderss("orders.txt", ios::app);
while (true) {
    count++;
    cout << "\n\n Enter Book ID: ";
    cin >> arrbcode[c];
    bool validBooks = false;
    books.open("books.txt", ios::in);

    if (!books) {
        cout << "\n\n\t" << ch << ch << " Empty Products!!";
        break;
    }
```

- Here, There is an array called arrbcode for storing the book ID for creating bill purposes. Those relevant book IDs will be gotten from the file called books.txt. Usually, arrays are used for storing the data temporarily.

Structures

```
//struc
struct BookShop {
    int bid = 0;
    float price = 0;
    string name, aname, pname;
};

struct Order {
    int oid, quantity;
    string name, aname;
};
```

- There are two structures in my code called BookShop and Order. Structures are usually used for when variables of different data types need to be filed under a single name.

TASK 03

Test Document of the Bun Talk Billing System

3.1 Introduction

This test document aims to provide an overview of the Nethra bookshop's system's testing methods, test cases, sample questionnaires, and procedures. Should any issues arise with the software system constructed using this document, they can be resolved, enabling the system's end users to produce high-caliber software.

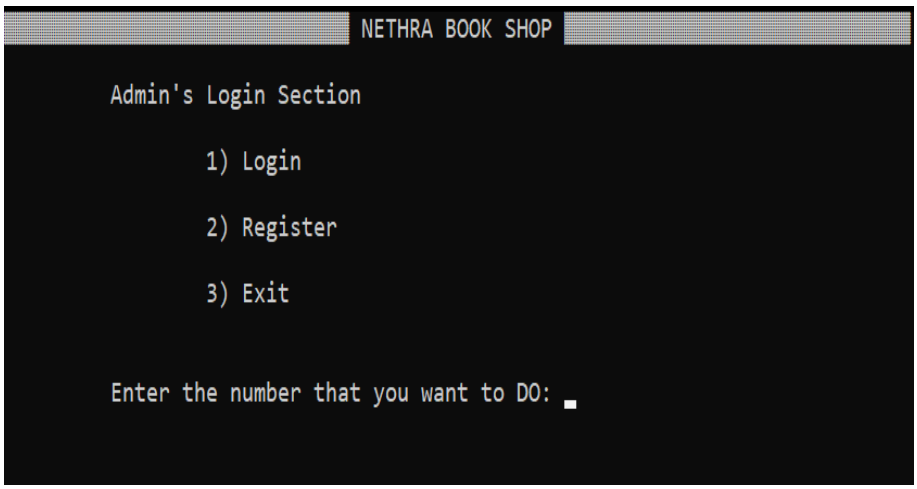
Scope - This automated system has two stakeholders. They are both users and administrators. Admins should be allowed to log in and create new accounts in this system login section. Other administrative functions, such as adding, changing, removing, and double-checking added books, can be shown here. Additionally, cashiers have access to the system entry area's administrator credentials. Other user capabilities include the ability to create orders and bills using those orders.

3.2 Test Cases

The settings selected to assess the built-in software application's functioning and performance are called test cases. Test cases are mostly used to make sure the software performs as intended.

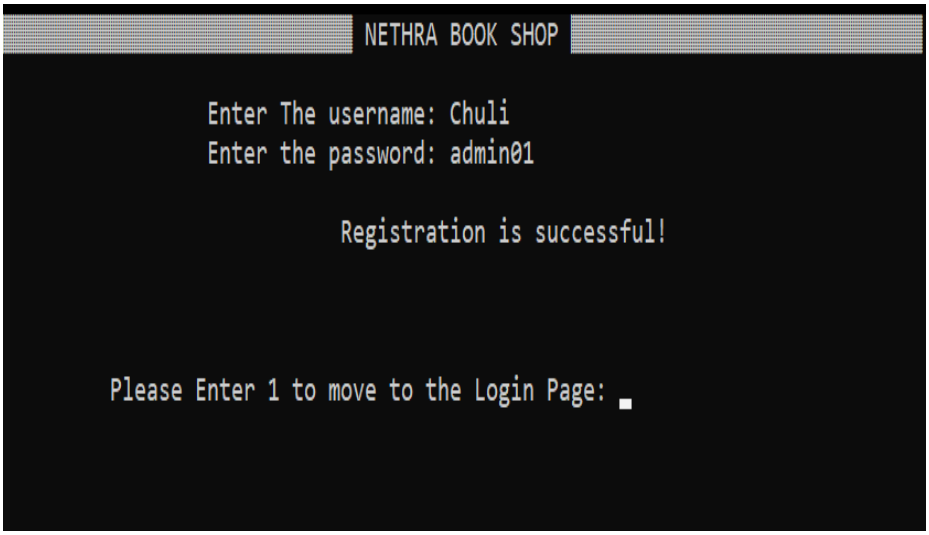
3.3 The results of each relevant test case

3.3.1 Admin's login

Test No.	Case 1
Scenario	Administrator's login page After opening the Nethra book shop automated system, select the role as "Admin". Press one for that. Then the user falls into the Administrator's login section.
Expected Result	Showing the administrator's login section page. The administrator should be able to log in to the system, register to the system, and exit from the system.
Actual Result	
Status	Pass

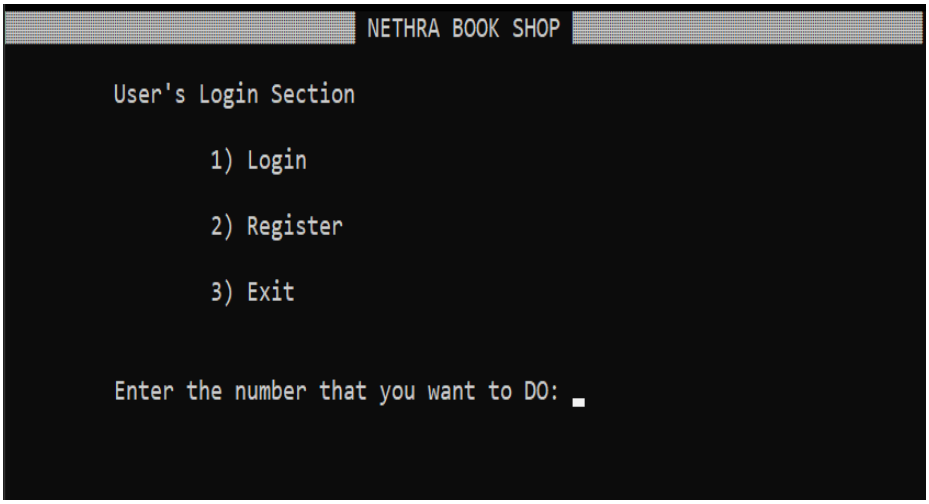
3.3.2 Admin's Registration

Test No.	Case 2
Scenario	Administrator's registration

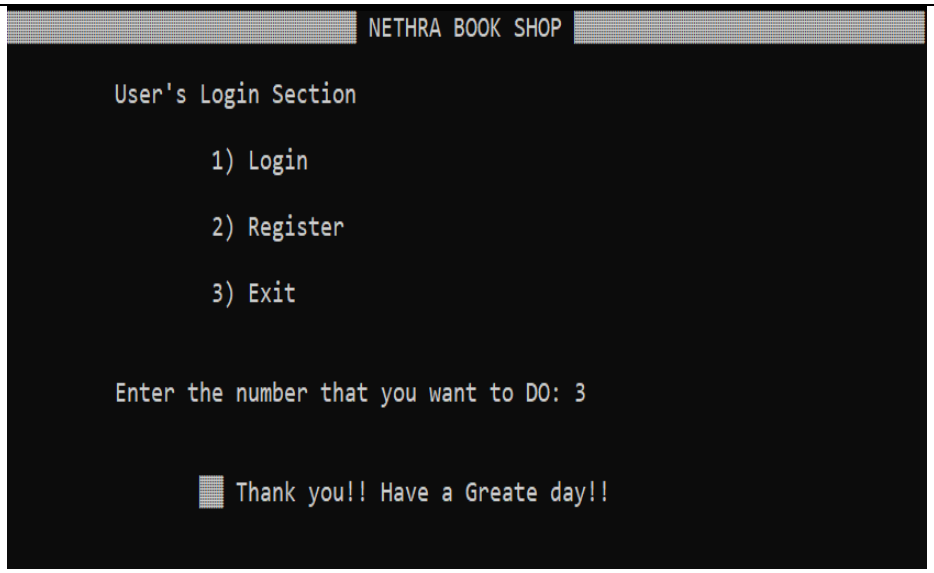
	To register for the system, access the administrator's login page and hit two. Here, the user must provide the system with their account and password.
Expected Result	After the user enters the username and password, the system shows a message that “Registration is successful!”
Actual Result	
Status	Pass

3.3.3 User’s Login

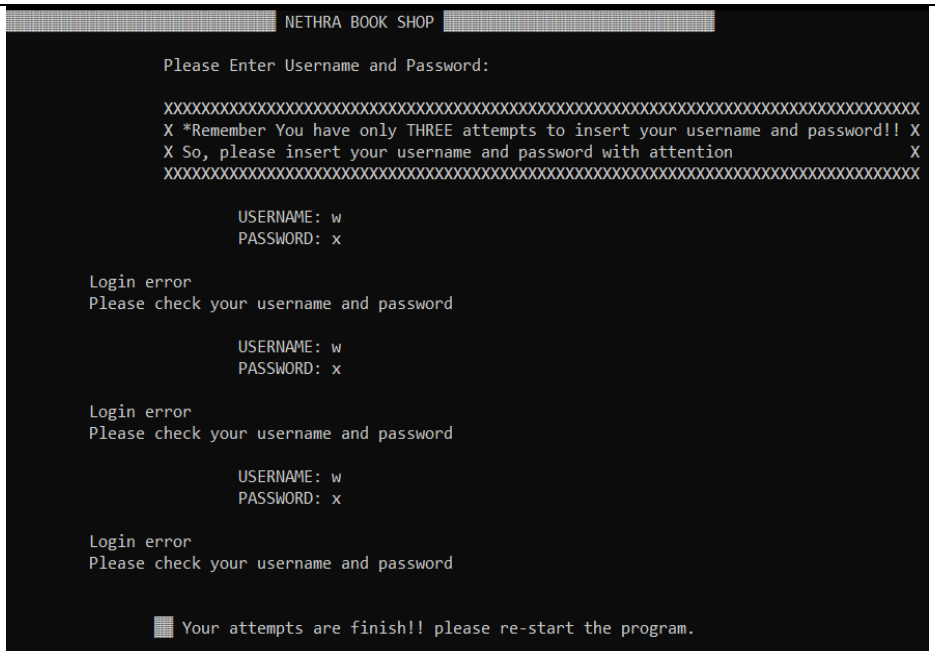
Test No.	Case 3
Scenario	User’s login section After opening the Nethra book shop automated system, select the role as “User”. Press two for that. Then the user falls into the User’s login section.
Expected Result	Showing the user’s login section page. Also the user should be able to log in to the system, register to the system, and exit from the system.

Actual Result	
Status	Pass

3.3.4 User's Exit

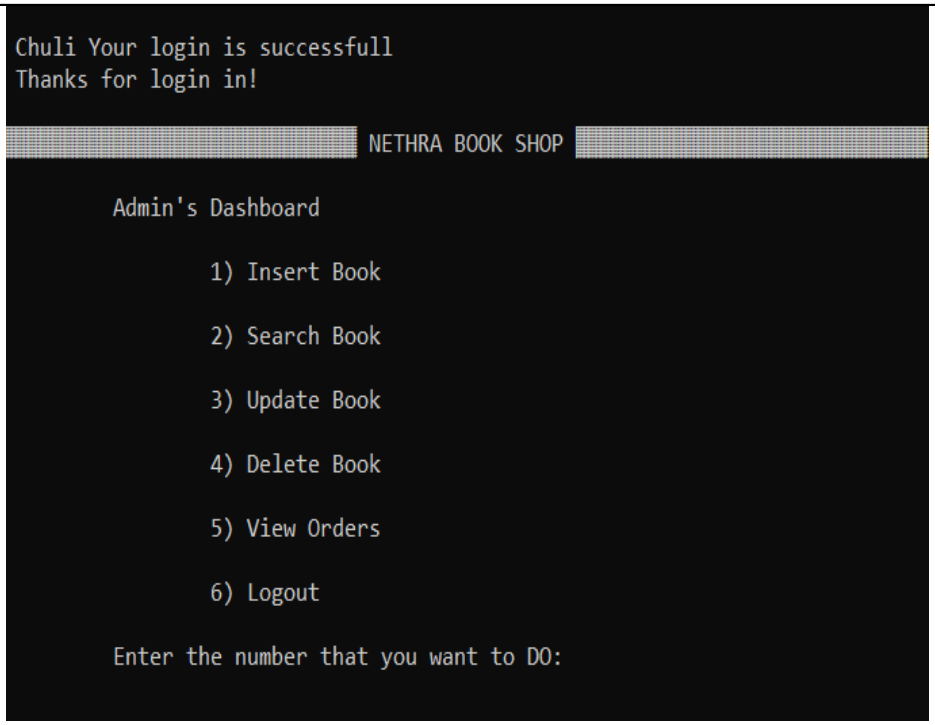
Test No.	Case 4
Scenario	<u>User's exit</u> After falling the User's login section, May he want to exit from the system. For that the user can press number three.
Expected Result	Showing a message called "Thank you!! Have a great day!! And should be finished the process of executing.
Actual Result	
Status	Pass

3.3.5 Admin's entering username and password

Test No.	Case 5
Scenario	Entering username and password After falling admin login section, the user can login to the system. For that he can press number one and give the username and password correctly.
Expected Result	If the user can't give the correct username and password within three attempts, the system will break automatically and print a message called "Your attempts are finished!! Please re-start the program".
Actual Result	
Status	Pass

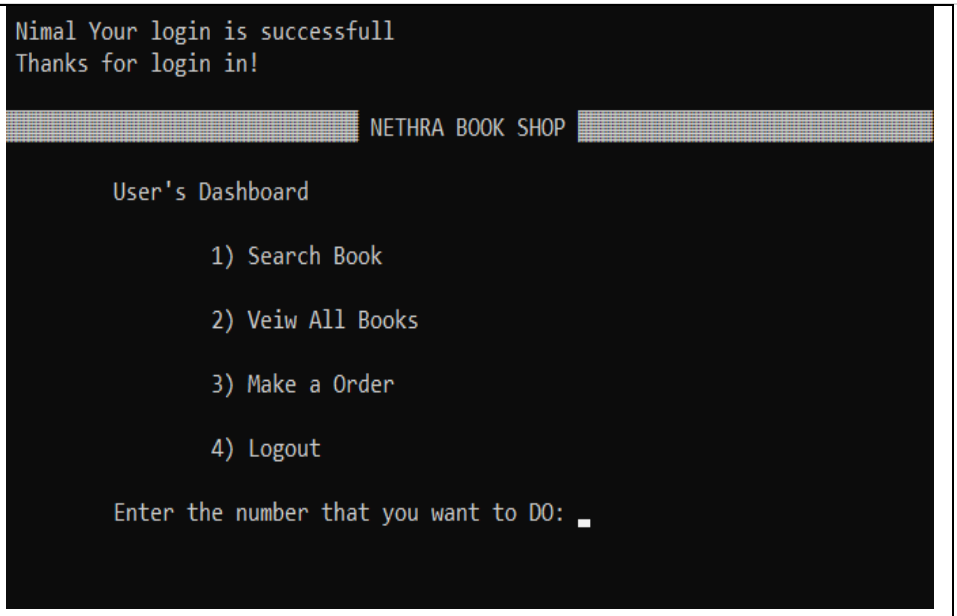
3.3.6 Admin's Dashboard

Test No.	Case 6
Scenario	Admin Dashboard After giving the username and password correctly, the admin can move to the admin dashboard.
Expected Result	After giving the correct username and password, the system shows a message called "Your login is successful" with the inserted username. And show the admin dashboard.

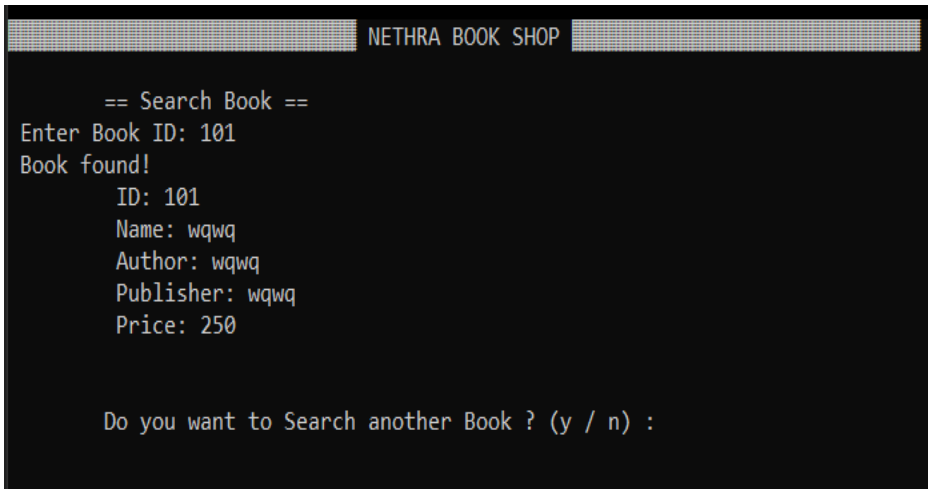
Actual Result	
Status	Pass

3.3.7 User's Dashboard

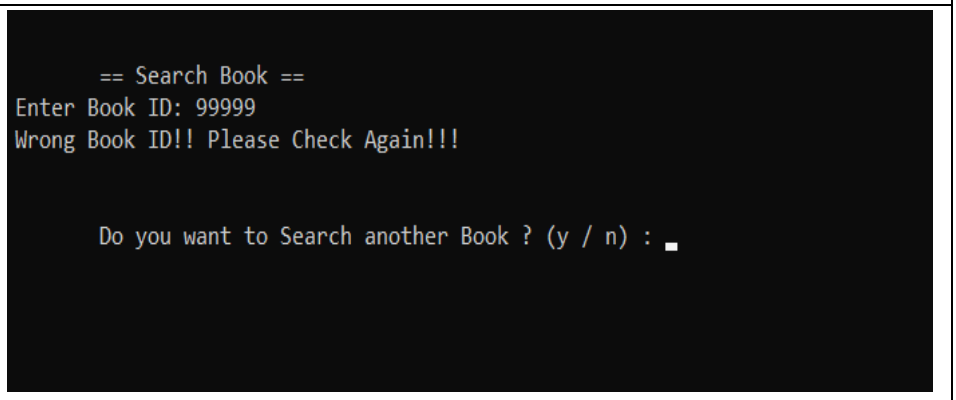
Test No.	Case 7
Scenario	User Dashboard After giving the username and password correctly, the user can move to the user dashboard.
Expected Result	After giving the correct username and password, the system shows a message called "Your login is successful" with the inserted username. And show the user dashboard.

Actual Result	
Status	Pass

3.3.8 Search Book

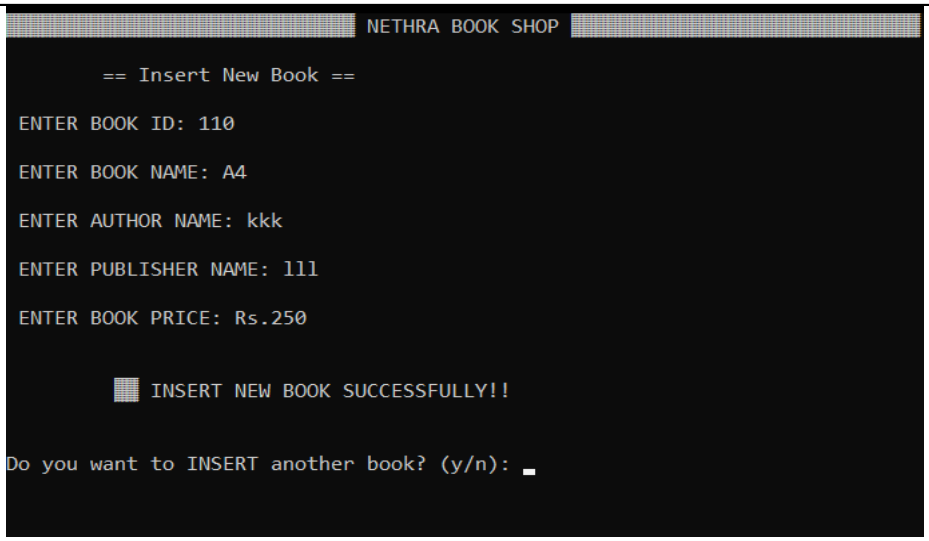
Test No.	Case 8
Scenario	<u>Search Book</u> This function has both admin and user. After selecting the search book option, first user has to insert a book ID, then he can get the details of that relevant book.
Expected Result	After giving a valid book ID, the system show the details of the book. Such as ID, name, author name, publisher name, and price, etc.
Actual Result	
Status	Pass

3.3.9 Search Book wrong ID

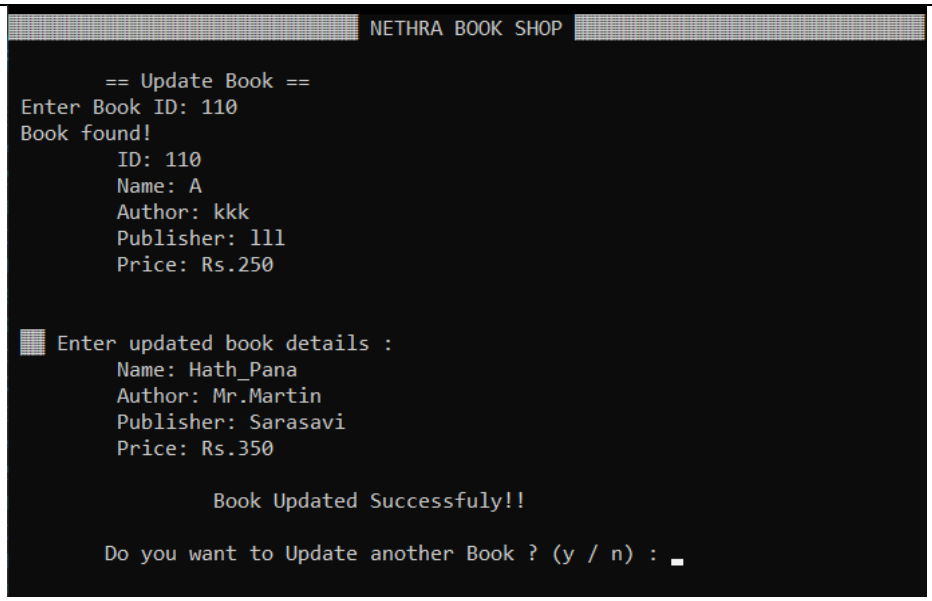
Test No.	Case 9
Scenario	Search Book wrong ID When searching a book, if the user enters an invalid book ID, the system will show an error message like “Wrong Book ID!! Please Check Again!!”
Expected Result	Error message
Actual Result	
Status	Pass

3.3.10 Insert Book

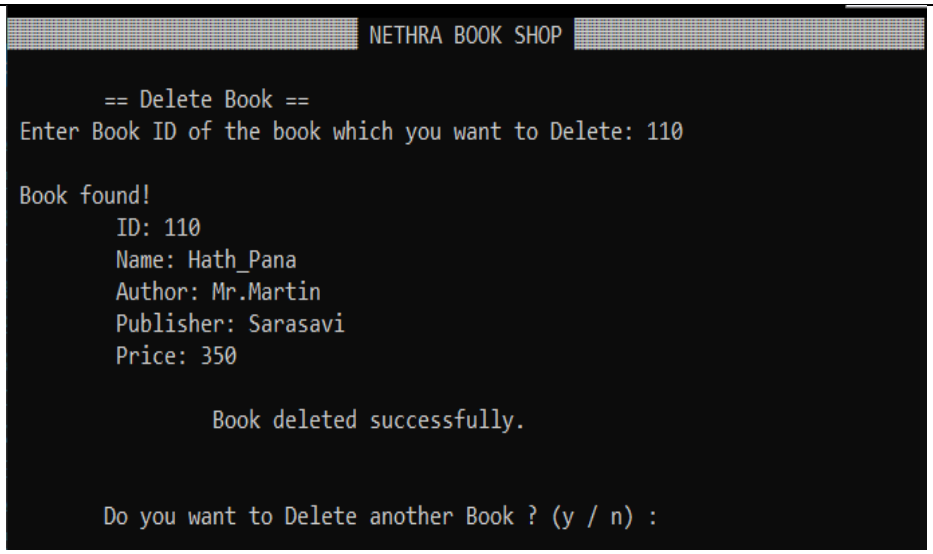
Test No.	Case 10
Scenario	Insert Book This function has only admin. After selecting the insert book option, the admin has to insert some details of the book. Such as ID, name, author name, publisher name and price.
Expected Result	After giving those details print a message like “Insert Book Successfully”

Actual Result	
Status	Pass

3.3.11 Update Book

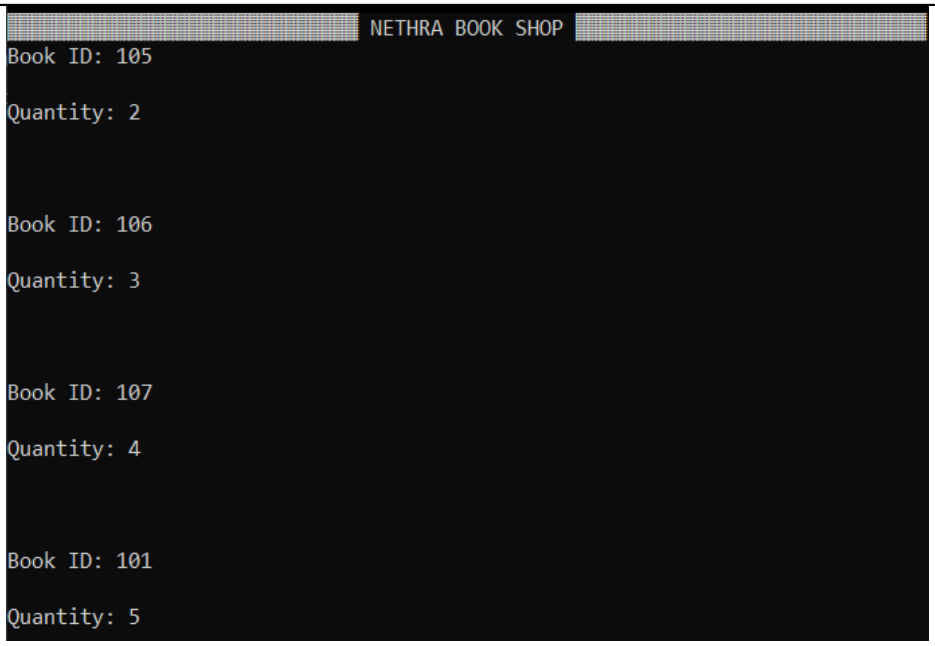
Test No.	Case 11
Scenario	Update Book This function has only admin. After selecting the update book option, the admin has to insert the book ID first. After system show the details of that book. And the system allows changing the details.
Expected Result	The system allows changing the details.
Actual Result	
Status	Pass

3.3.12 Delete Book

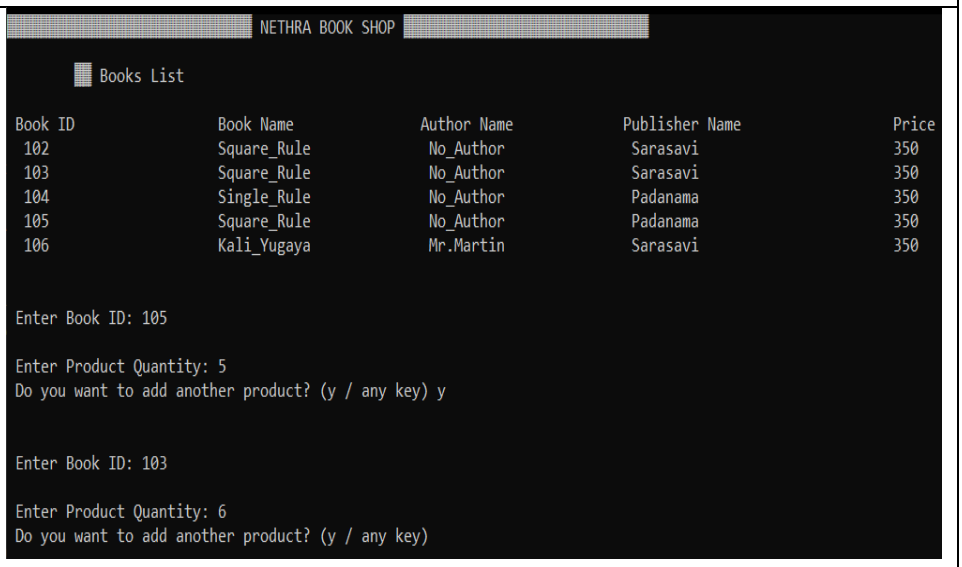
Test No.	Case 12
Scenario	Delete Book This function has only admin. After selecting the delete book option, the admin has to insert the book ID first. After the system shows the details of that book and deletes those details from the system.
Expected Result	The system allows deleting the details.
Actual Result	
Status	Pass

3.3.13 View Orders

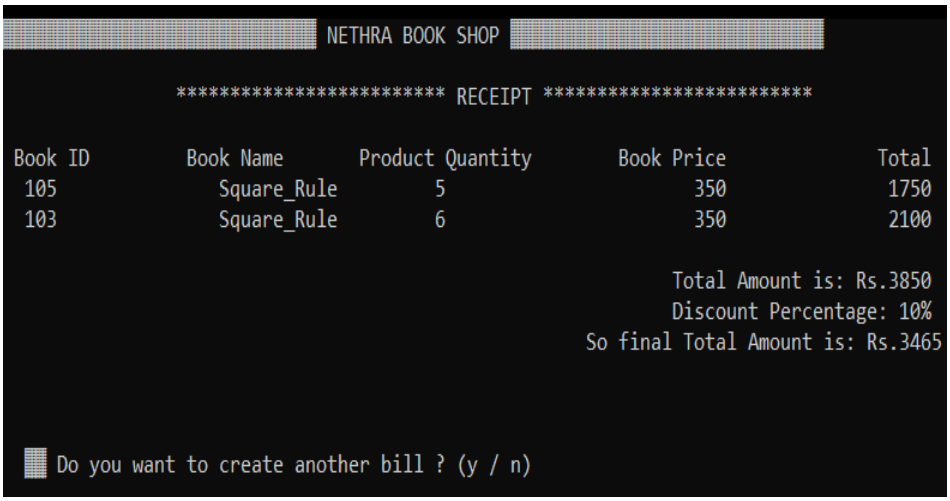
Test No.	Case 13
Scenario	View Orders This function has only admin. After selecting the view orders option, the system shows the orders that are done by users
Expected Result	Relevant book id and quantity

Actual Result	
Status	Pass

3.3.14 Make Orders

Test No.	Case 14
Scenario	Make Orders This function has only users. After selecting the make orders option, the system shows the books details in a table.
Expected Result	Book details in a table and the ability to enter book ID and quantity
Actual Result	
Status	Pass

3.3.15 Print Bill

Test No.	Case 15
Scenario	Print Bill This function has only users. After giving book ID and quantity, system prints the bill.
Expected Result	The receipt
Actual Result	
Status	Pass

3.4 User Acceptance Testing Questionnaires

Role / Position –

Name –

Email –

Contact –

❖ Underline your answer

1. Overall Satisfaction

i. On a scale of 1 to 5, how satisfied are you with this system?

- Very Dissatisfied
- Dissatisfied
- Normal
- Satisfied
- Very Satisfied

2. User Interface

ii. Did you find the system easy to use?

- Yes
- No

If no, please describe what parts need improvement.

3. Billing Process

iii. Were you able to generate bills without any difficulties?

- Yes
- No

If no, please describe what parts need improvement.

iv. Were the billing calculations accurate?

- Yes
- No

If no, please specify any inaccuracies you noticed.

4. Performance and Speed

v. Did you experience any performance issues, such as slow loading times?

- Yes
- No

If yes, please describe those the performance issues.

vi. Having problems updating, deleting books in this system?

- Yes
- No

If yes, please explain what those problems.

vii. Having problems entering books to create the bill?

- Yes
- No

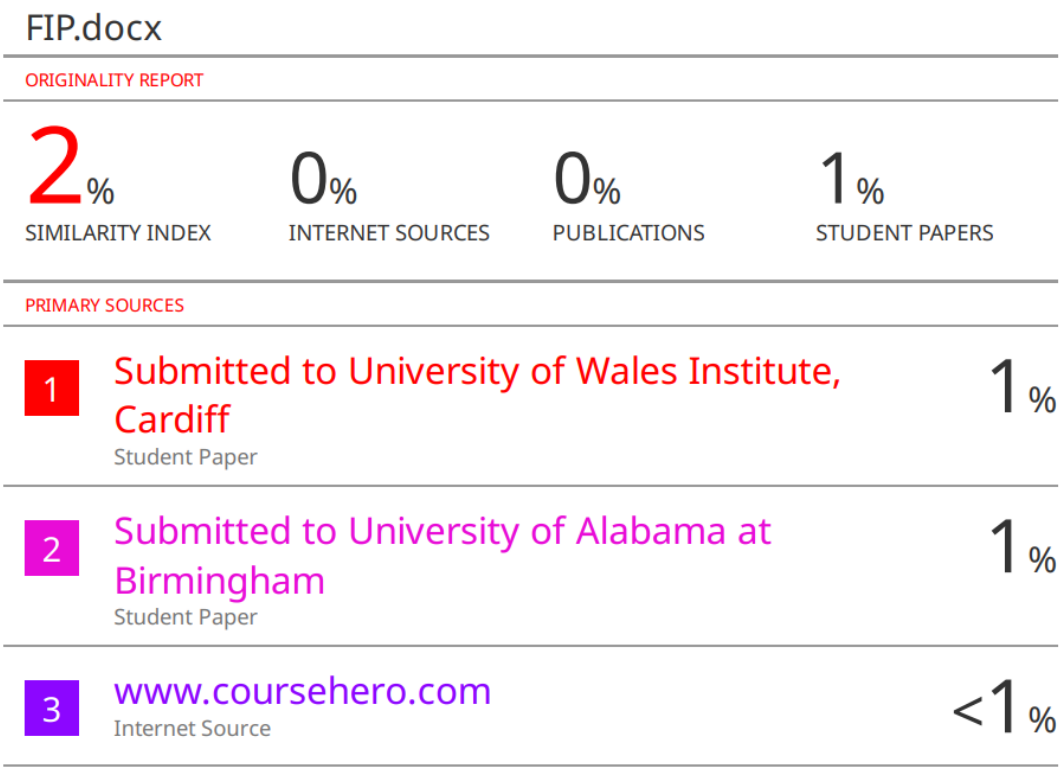
If yes, please explain what those problems.

5. Additional Comments

viii. Do you have any additional comments, suggestions for improvement, or features you'd like to see added to this system?

REFERENCES

- Bandakkanavar, R. (2023) *Software requirements specification document with example*, *Krazytech*. Available at: https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database (Accessed: 04 January 2024).
- Pawlicka, A. (2023) *HOW TO WRITE PROJECT SCOPE in software development*, *Selleo*. Available at: https://selleo.com/blog/how-to-write-project-scope-in-software-development (Accessed: 05 January 2024).



FIP.docx

ORIGINALITY REPORT

2%	0%	0%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Wales Institute, Cardiff Student Paper	1%
2	Submitted to University of Alabama at Birmingham Student Paper	1%
3	www.coursehero.com Internet Source	<1%