##### BOOK MANAGEMENT SYSTEM

##### A PROJECT REPORT

***Under the guidance of***

**Khagendra Shah**

***Submitted by***

### Pratik Baishnav

*In partial fulfillment of the requirement for the award of the degree*

*Of*

**B.Sc.IT Sixth Semester**

****

18-July-2015

**College for Professional Studies**

**LC of Sikkim Manipal University (01537)**

### VIVA VOCE SHEET

We have examined the viva-voce examination of the graduate research report presented by

**Pratik Baishnav**

Entitled

**Book Management System**

And found the report to be original work of the student and written according to the prescribed format of the University. We recommend the report to be accepted as partial fulfillment of the requirements for the B.Sc.IT Sixth semester.

**Viva-Voce Committee**

1. Supervisor/Project Guide

1. Program Coordinator
2. Chief Program Coordinator
3. External Examiner

Date: 18 July 2015

### 

### BONAFIDE CERTIFICATE

Certified that this project report titled**“Book Management System”** is bonafide work of **“Pratik Baishnav”**who carried out the project work under my supervision.

**CERTIFIED**

**………………………………..**

**Khagendra Shah**

**Project Supervisor**

Department of Computer Science

College for Professional Studies

LC of Sikkim Manipal University

LC Code: 01537

### 

### Certificate from University Learning Centre

This is to certify that **Pratik Baishnav** of B.Sc.IT has worked on the project as per the course curriculum of B.Sc.IT Sixth Semester. This project entitled “**Book Management System**” is the original work of Pratik Baishnav and was carried out under the supervision of **Khagendra Shah** as per the guidelines provided by the university. As per the student’s declaration this is certified that his projecthas not been presented anywhere as a part of any other academic work.

**………………………**

**P.Kejriwal**

**Founder**

**………………………**

**Er. PankajJalan**

**Executive (Director)**

**………………………**

**Er. Prakash Kumar**

**Director (Academics)**

### 

### Certificate from University Learning Centre

This is to certify that **Pratik Baishnav** of B.Sc.IT has worked on the project as per the course curriculum of B.Sc.IT Sixth Semester. This project entitled “**Book Management System**” is the original work of **Pratik Baishnav** and was carried out under the supervision of **Khagendra Shah** as per the guidelines provided by the university. As per the student’s declaration this is certified that his project has not been presented anywhere as a part of any other academic work.

**………………………**

**Prof. Dr. Mahendra Singh**

**Chief Academic Advisor**

**………………………**

**Ramesh Suwal**

**Program Coordinator**

**………………………**

**Uday Kant Jha**

**Chief Coordinator**

### Student declaration

I hereby declare that the project report entitled

**Book Management System**

Submitted in partial fulfillment of the requirements for the degree of

B.Sc.IT Sixth Semester

To Sikkim-Manipal University, India, is my original work and not submitted for the award of any other degree, diploma, fellowship, or any other similar title or prizes

Place: Kathmandu Pratik Baishnav

Date: 18-July-2015 Reg. No: 551217295

### 

### ACKNOWLEDGEMENT

I consider it a privilege to express through these pages of this report, a few words of gratitude and respect to all who guided and inspired me in completion of this project. The successful accomplishment of this project work requires efforts of several intellect minds. Several people have contributed their experiences and ideas to make this project.

Words appear inadequate and hollow, when I take this opportunity to express my deep sense of gratitude to “**Er. Pankaj Jalan”,** the administrator Director of CPS College, to “**Er. Prakash Kumar**”, the academic director, to “**Mr. Uday Kant Jha**”, the program coordinator, to my respectable project guides “**Mr. KhagendraSah**” and respectable teachers **Mr. Sanjeev Subedi** and **Mr. Chandra Kishor Gauro** and all the teachers and staff for their cooperation and coordination in providing necessary information and resources for our project. Especially I like to thank to **Mr. LaxmanSubedi** for inspiration and proper guidance.

I express my sincere thanks to my classmates, friends and my family my brother Pukar Baishnav for the most for relentless support and help to make this project as it is should be. And at last but not the least I would like to thank my close friends Bikash Shrestha, Sunil Shrestha, Seeba Shrestha, Prasanga Poudel, Pasang Norbu Lama, Sazeed Shakya, Sushila Chhunju and so on for support and providing me different ideas and their valuable time for this project. I thank all the people for their help directly and indirectly to complete my major project. Thank You.

**Pratik Baishnav**

**B.Sc.IT Sixth Semester**

**Roll no: 551217295**

### 

### ABSTRACT

The client uses MS Excel, and maintains their records, however it is not possible them to share the data from multiple system in multi user environment, there is lot of duplicate work, and chance of mistake. When the records are changed they need to update each and every excel file. There is no option to find and print previous saved records. There is no security; anybody can access any report and sensitive data, also no reports to summary report. This Book Management System is used to overcome the entire problem which they are facing currently, and making complete atomization of manual system to computerized system.

This project entitled “Book Management System” is a major project work for partial fulfillment of B.Sc.IT Sixth Semester. It is a small software application which is helpful in the different educational institutes. This project is database oriented program and it is developed using SQLSERVER 2008 R2 and C#.NET.

Hence, this software is helpful for the staffs of a library who want to improve the efficiency of the book registration and want to be competitive in this day to day changing world. This project is also a small step towards development of big professional software.

**Table of Content**

1.[Title Page I](#_Toc409780139)

2. [VIVA VOCE SHEET II](#_Toc409780140)

3. [BONAFIDE CERTIFICATE III](#_Toc409780141)

4. [Certificate from University Learning Centre IV](#_Toc409780142)

5. [Certificate from University Learning Centre V](#_Toc409780143)

6. [Student declaration VI](#_Toc409780144)

7. [ACKNOWLEDGEMENT VII](#_Toc409780145)

8. [ABSTRACT VIII](#_Toc409780146)

[9. Chapters 4](#_Toc409780147)

[1. Introduction 4](#_Toc409780148)

[2. Objective/Scopes on Book Management System 4](#_Toc409780149)

[3. THEORETICAL BACKGROUND 5](#_Toc409780150)

[4. DEFINITION OF PROBLEM 6](#_Toc409780151)

[5. FEASIBILITY STUDY 7](#_Toc409780152)

[6. SYSTEM PLANNING 8](#_Toc409780153)

[6.1 Gantt Chart 8](#_Toc409780154)

[7. METHODOLOGY ADOPTED 9](#_Toc409780155)

[7.1 Project Methodology 9](#_Toc409780156)

[7.2 Hardware and Software Used 12](#_Toc409780157)

[8. LIFE CYCLE OF PROJECT 13](#_Toc409780158)

[8.1 System Architecture Overview 13](#_Toc409780159)

[8.2 Testing Method 16](#_Toc409780161)

[9. DATA DICTONARY 21](#_Toc409780162)

[9.1. Schemas 21](#_Toc409780163)

[10. SNAPSHOTS AND CODE OF PROJECT 25](#_Toc409780164)

[10.1 SNAPSHOTS 25](#_Toc409780165)

[10.2 CODES 31](#_Toc409780165)

[11. USER OPERATIONS AND MANUAL 67](#_Toc409780166)

[11.1 Support and Maintenance 67](#_Toc409780167)

[11.2 Future Scope 68](#_Toc409780168)

[11.3 Backup and Recovery 68](#_Toc409780169)

[11.4 Security 69](#_Toc409780170)

[12. Conclusion and Recommendation 70](#_Toc409780171)

[13. Bibliography 71](#_Toc409780172)

**LIST OF TABLES**

[table 1: Table\_Author](#_Toc409722102) 22

[table 2: Table\_Book](#_Toc409722103) 23

[table 3: Table\_Category](#_Toc409722104) 23

[table 4: Table\_Fine](#_Toc409722105) 23

[table 5: Table\_Issue](#_Toc409722106) 24

table 6: Table\_Login 24

table 7: Table\_StudentDetail 24

**LIST OF FIGURES**

[Figure 1: ER Diagram 13](#_Toc409734013)

[Figure 2: Context Level DFD 14](file:///C:\Users\user\Desktop\main.docx#_Toc409734014)

[Figure 3: Flow Chart 15](file:///C:\Users\user\Desktop\main.docx#_Toc409734015)

[Figure 4: Schema 22](#_Toc409734016)

[Figure 5: Login Panel 25](file:///C:\Users\user\Desktop\main.docx#_Toc409734017)

[Figure 6: Admin Form 25](file:///C:\Users\user\Desktop\main.docx#_Toc409734018)

[Figure 7: User Form 26](file:///C:\Users\user\Desktop\main.docx#_Toc409734019)

[Figure 8: Add User 26](file:///C:\Users\user\Desktop\main.docx#_Toc409734020)

[Figure 9: Add student 27](file:///C:\Users\user\Desktop\main.docx#_Toc409734021)

[Figure 10: add book 27](file:///C:\Users\user\Desktop\main.docx#_Toc409734022)

[Figure 11: user details 28](file:///C:\Users\user\Desktop\main.docx#_Toc409734023)

[Figure 12: view book 28](file:///C:\Users\user\Desktop\main.docx#_Toc409734024)

[Figure 13: change user details 29](file:///C:\Users\user\Desktop\main.docx#_Toc409734025)

[Figure 14: change book details 29](file:///C:\Users\user\Desktop\main.docx#_Toc409734026)

[Figure 15: view fine 30](file:///C:\Users\user\Desktop\main.docx#_Toc409734027)

FIGURE 16: REPORT 30

**Abbreviation**

DFD = Data Flow Diagram

ERD = Entity Relationship Diagram

SQLSERVER 2008 R2 = My Structured Query Language

B.Sc.IT = Bachelors of Science in Information Technology

IT = Information Technology

SMU = Sikkim Manipal University

CPS = College for Professional Studies

LBEF = Lord Buddha Education Foundation

# 

# 9. Chapters

## 1. Introduction

Book Management System is desktop based small computer application. Book Management System is focused for areas like institutes which offer books library and made for saving records of books. This application is very useful to store records of books.

The project is entitled “Book Management System” and before discussing anything about the project Book Management System, a brief discussion of related basic concept is necessary.

As a software developer or as a programmer, we are expected to design and developed any program that works correctly, efficiently and the time is easy to be used by every person, who may or may not be well versed with computer and its capabilities.

The Project is based on the Book Management System, Being the Information System it requires extensive use of some Data base Management System to store, manipulate and handle the huge and complex record, In RDBMS we can act various attributes with the database like editing the records, Modifications Deletions of the records, View the records in various formats, listing the database etc. Project can be categorized by their functioning and relation with their database and other tools can categorize project. Since this project has been developed based on the Relation Data Base Management System So Proposed system comes under RDBMS (Relational Database Management System) category, as there is need to store and manipulate a huge amount of data related to patients as per various queries.

## ****2. Objective/****Scopes****on Book Management System****

This is C#.NET Projects on Book Management System, which provided a lot of facility to their user. The objective and scope of my Project Book Management System is to record the details various activities of user. It will simplify the task and

reduce the paper work. During implementation every user will be given appropriate training to suit their specific needs. Specific support will also be provided at key points within the academic calendar. Training will be provided on a timely basis, and you will be trained as the new is Book Management System rolled out to your area of responsibility.

At the moment we are in the very early stages, so it is difficult to put a specific time on the training, but we will keep people informed as plans are developed. The system is very user friendly and it is anticipated that functions of the system will be easily accessed by administrators and user staffs.

Hence the application for the institute management has been designed to remove all the deficiency from which the present system is suffering and to ensure.

## 3. THEORETICAL BACKGROUND

The project is entitled “Book Management System” and before discussing anything about the project Book Management System, a brief discussion of related basic concept is necessary.

As a software developer or as a programmer, we are expected to design and developed any program that works correctly, efficiently and the time is easy to be used by every person, who may or may not be well versed with computer and its capabilities.

The Project is based on the Book Management System, Being the Information System it requires extensive use of some Database Management System to store, manipulate and handle the huge and complex record, In RDBMS we can act various attributes with the database like editing the records, Modifications Deletions of the records, View the records in various formats, listing the database etc. Project can be categorized by their functioning and relation with their database and other tools can categorize project. Since this project has been developed based on the Relation Data Base Management System So Proposed system comes under RDBMS (Relational Database Management System) category, as there is need to store and manipulate a huge amount of data related to patients as per various queries.

## 

## 4. DEFINITION OF PROBLEM

The existing system only provides text-based interface, which is not as user-friendly as Graphical user Interface. Since the system is implemented in Manual, so the response is very slow. The transactions are executed in off-line mode, hence on-line data capture and modification is not possible. On-line reports cannot be generated.

Hence, there is a need of reformation of the system with more advantages and flexibility. The Book Management System eliminates most of the limitations of the existing software. It has the following objectives:

**Enhancement:**

The main objective of Book Management System is to enhance and upgrade the existing system by increasing its efficiency and effectiveness. The software improves the working methods by replacing the existing manual system with the computer-based system.

**Automation:**

The Book Management System automates each and every activity of the manual system and increases its throughput. Thus the response time of the system is very less and it works very fast.

**Accuracy:**

The Book Management System provides the uses a quick response with very accurate information regarding the users etc. Any details or system in an accurate manner, as and when required.

**User-Friendly:**

The software Book Management System has a very user-friendly interface. Thus the users will feel very easy to work on it. The software provides accuracy along

with a pleasant interface. Make the present manual system more interactive, speedy and user friendly.

**Availability:**

The transaction reports of the system can be retried as and when required. Thus, there is no delay in the availability of any information, whatever needed, can be captured very quickly and easily.

**Maintenance Cost:**

Reduce the cost of maintenance.

## 5. FEASIBILITY STUDY

The feasibility study is an evaluation and analysis of the potential of a proposed project which is based on extensive investigation and research to support the process of decision making. It aim to objectively and rationally uncover the strengths and weaknesses of an existing business or proposed venture, opportunities and threats present in the environment, the resources required to carry through, and ultimately the prospects for success. In its simplest terms, the two criteria to judge feasibility are cost required and value to be attained.

Feasibility study conducted once problem is clearly understood. There are three types of feasibility study: technical feasibility, operational feasibility, economic feasibility.

**Technical feasibility:**

A technical feasibility study of function, performance and constraints can affect the ability to achieve an acceptable system. The software developed for managing Book Management System is used in computer as application software with C#.NET as front end and SQLSERVER 2008 R2 Server as back end so the project is technically feasible.

**Operational feasibility:**

The main purpose of this program is to develop a Desktop based application which facilitates on providing computer based or electronic based system replacing old paper work system. No computer trainings or experts are needed to handle this software. Therefore this project is operational feasible.

**Economic feasibility:**

Economic feasibility analysis includes a broad range of tests that include long term cooperative income strategies, cost of resources needed for development, cost benefit analysis. In existing system they had to maintain many registers/books is a costly affair. This can be reduced by keeping data in the digital format that is reliable and cheaper .Since the development cost for system satisfies the organization therefore the software is economically feasible.

## 6.SYSTEM PLANNING

### 6.1 Gantt Chart

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| S.N. | Tasks | Start | End | Duration  (days) | 2015 | | |
| May | June | July |
| 1 | **Analysis** | 02/05/15 | 14/05/15 | 12 |  | | |
| 2 | **Design/coding** | 16/05/15 | 05/06/15 | 34 |  | | |
| 3 | **Documentation** | 10/06/15 | 15/06/15 | 5 |  | | |
| 4 | **Implementation** | 17/06/15 | 24/06/15 | 8 |  | | |
| 5 | **Review** | 01/07/15 | 07/07/15 | 7 |  | | |
| 6 | **Testing** | 07/07/15 | 17/07/15 | 10 |  | | |

## 

## 7. METHODOLOGY ADOPTED

### 7.1 Project Methodology

In software engineering, a methodology refers to software development methodology (also known as software development life cycle), which is a division of software development work into distinct phases or activities with the intent of better planning and management. It may include the pre-definition of specific deliverables and artifacts that are created and completed by a project team to develop or maintain an application.

Project management methodologies are all about specifying the best way to initiate, plan and execute projects. Common methodologies include waterfall prototyping, iterative and incremental development, spiral development, rapid application development and extreme programming.

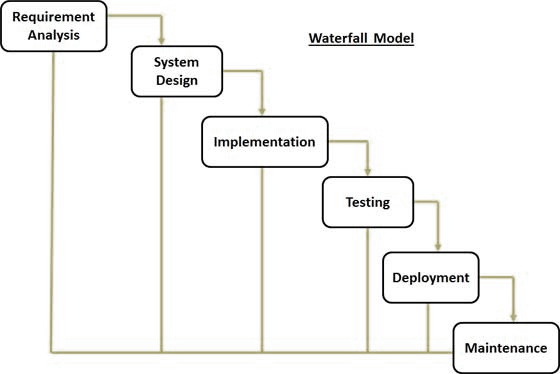
Several software development approaches have been used since the origin of information technology, in two main categories. Typically an approach or a combination of approaches is chosen by management or a development team. In order to ensure that this project is in par with the objectives, I used the waterfall model approach in developing the systems.

**Waterfall Model**

Waterfall model has evolved from what has been termed more traditional project management methodologies, employing a sequential top-down approach to project management. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The waterfall model illustrates the software development process in a linear sequential flow; hence it also referred to as a linear sequential life cycle model. The waterfall approach, the whole process of software development is divided into six different phases, typically, the outcome of one phase acts as input for the nest phase sequentially.

Following is diagrammatic representation of different phases of waterfall model:



The sequential phases in Waterfall model are:

1. Requirement Analysis

All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

1. System Design

The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

1. Implementation

With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which referred to as Unit Testing.

1. Integration and Testing

All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

1. Deployment of System

Once the functional and non-functional testing is done, the product is deployed in the customer environment or released into the market.

1. Maintenance

There are some issues which come up with in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name “Waterfall Model”. In this model, phases do not overlap.

### 

### 7.2 Hardware and Software Used

#### 7.2.1 Hardware Configuration

This system is developed under following hardware configuration:

1. **Intel i3 Personal Computer**
2. **4 GB DDR-III RAM**
3. **500 GB HDD**
4. **Microsoft Compatible Key Board**
5. **Optical Mouse**

#### 7.2.2 Description of Software

This system is developed using the following software:

1. **Operating System : Windows 8.1**
2. **IDE Platform : Visual Basic 2012**
3. **Front-End Technology : C#.NET**
4. **Back-End Technology : SQLSERVER 2008 R2**

## 

## 

## 8. LIFE CYCLE OF PROJECT

### 8.1 System Architecture Overview

#### 8.1.1 Entity-Relationship Diagram

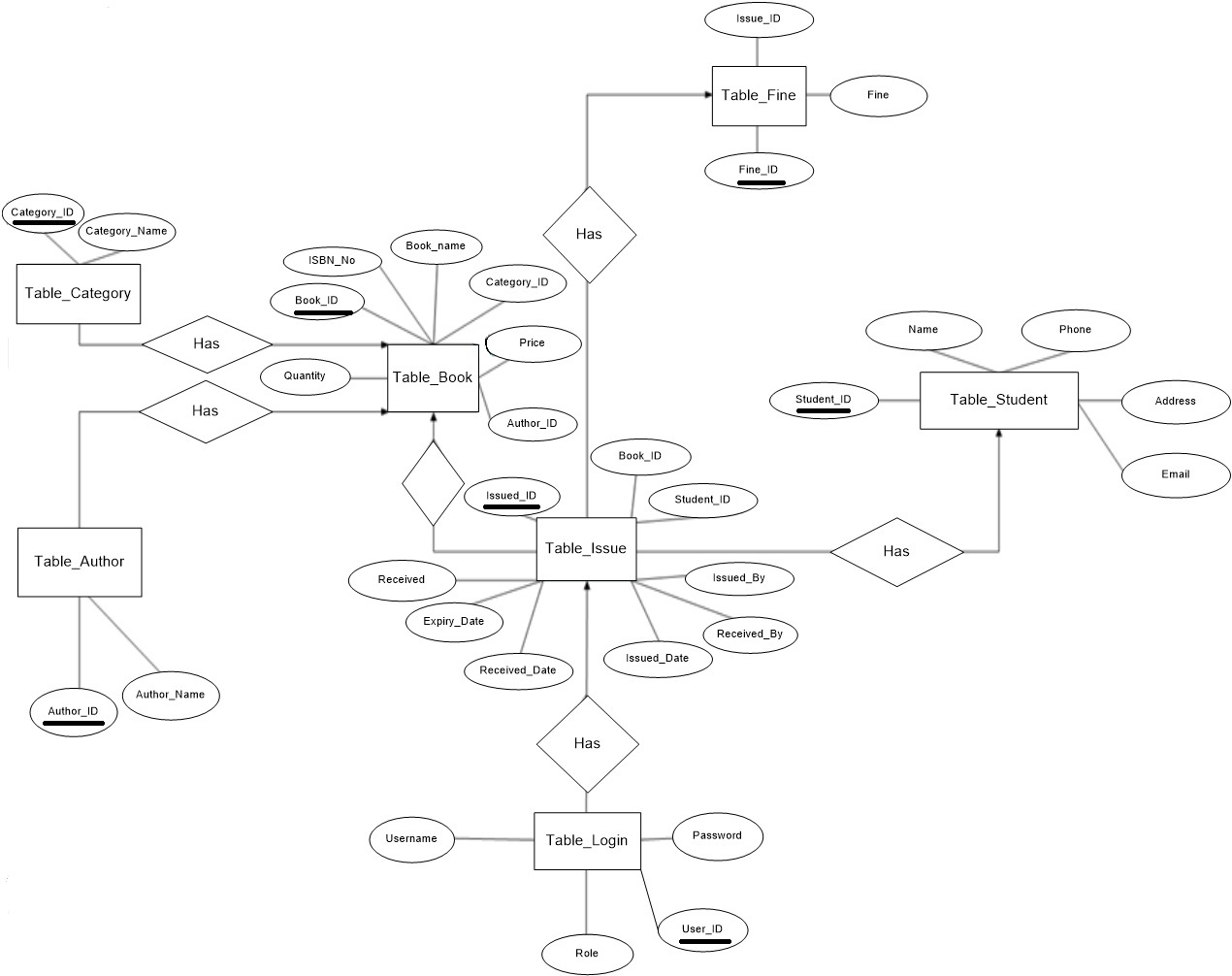


Figure : ER Diagram

#### 8.1.2 Data Flow Diagram

##### 8.1.2.1 Context Level DFD

Figure : Context Level DFD

##### 8.1.3. System Flow Chart

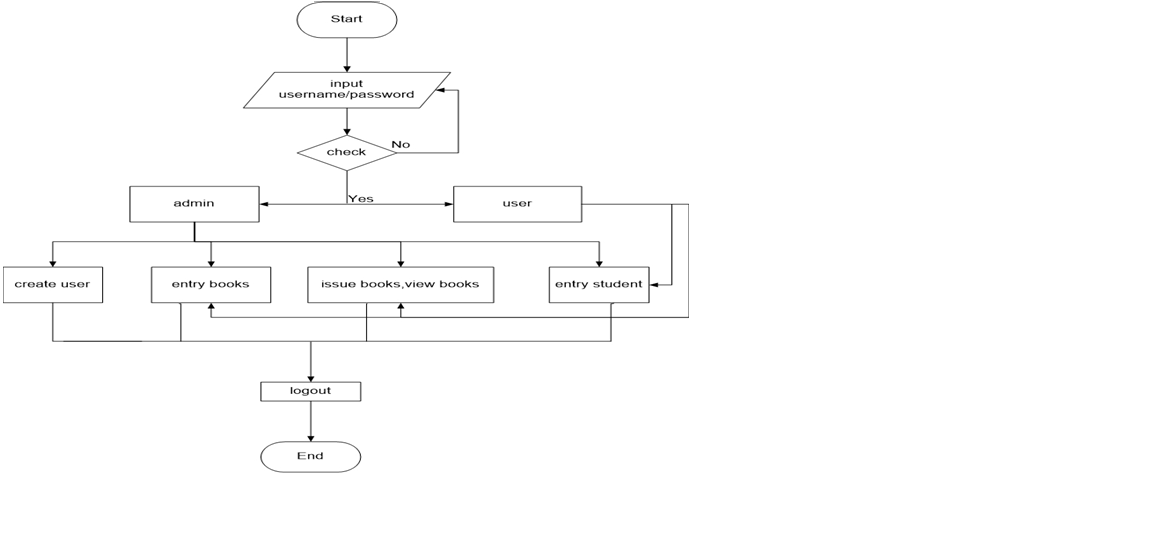
****

Figure : Flow Chart

### 8.2 Testing Method

Software Testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. It can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).

#### 8.2.1 Coding Tools

In software engineering, the terms “front end” and “back end” are distinctions which refer to the separation of concerns between a presentation layer and a data access layer respectively. The front end is an interface between the user and the back end. The front and back ends may be distributed amongst one or more systems.

##### 8.2.1.1 Back End Tools

This activity deals with the design of the physical database. A key is to determine how the access paths art to be implemented.

Program design: In conjunction with database design is a decision on the programming language to be used and the flowcharting, coding, and debugging procedure prior to conversion. The operating system limits the programming languages that will run of the system.

System and program test preparation. Each aspect of the system has a separate test requirement. System testing is done after all programming and testing completed the test on system and program test requirements become a part of design specifications a prerequisite to implementation.

In contrast to the system testing is acceptance testing, which puts the system through a procedure design to convince the user that the proposed system will meet the stated requirements. Acceptance testing is technically similar to system testing but politically it is different.

##### 8.2.1.2 Front End Tools

The systems objectives outlined during the feasibility study serve as the basic from which the work of system design is initiated. Much of the activities involved at this stage are of technical nature requiring a certain degree of experience in designing systems, sound knowledge of computer related technology and through understanding of computers available in the market and the various facilities provided by the vendors. Nevertheless, a system cannot be designed in isolation without the active involvement of the user. The user has a vital role to play at this stage too. As we know that data collected during feasibility study wills we utilized systematically during the system design. It should, however be kept in mind that detailed study of the existing system is not necessarily over with the completion of the feasibility study. Depending on the plan of feasibility study, the level of detailed study will vary and the system design stage will also vary in the amount of investigation that still needs to be done. This investigation is generally an urgent activity during the system. Sometimes, but rarely, this investigation may form a separate stage between feasibility study and computer system design. Designing a new system is a creative process, which calls for logical as well as lateral thinking. The logical approach involves systematic moves towards the end product keeping in mind the capabilities of the personnel and the equipment at each decision making step. Lateral thought implies encompassing of ideas beyond the usual functions and equipment. This is to ensure that no efforts are being made to fit previous solutions into new situations.

##### 8.2.2 Testing Cases

A test case is a set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly. The process of developing test cases can also help find problems in the requirements or design of an application.

##### Test Case 01

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Pre-conditions: User run the application and login form displays asking to input correct  Username and password. User inputs correct username and password value.  . | | | | | |
| Step | Action | Expected System Response | P/F | Comment |
| 1 | User inputs value “admin” in both fields. | The system waits user to click button. |  |  |
| 2 | User click on login button | The system runs the Admin form. |  |  |

|  |
| --- |
| Post Condition: The user “admin” is logged in into system. |

##### Test Case 02

|  |
| --- |
| Test Case #: 02 Test Case Name: User Log In Failure  System: Windows 8.1 OS Short Description: Test user fail  Designed by: Pratik Baishnav Design Date: 2015-05-16  Executed by: Pratik Baishnav Execution Date: 2015-05-16 |

|  |
| --- |
| Pre-conditions: User run the application and login form displays asking to input correct  Username and password. User inputs incorrect username and password value.  . |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | Action | Expected System Response | P/F | Comment |
| 1 | User inputs value “abcde” in both fields. | The system displays message saying incorrect username and password. |  |  |

|  |
| --- |
| Post Condition: The unauthorized users are not accessed. |

##### Test Case 03

|  |
| --- |
| Test Case #: 03 Test Case Name: Add Book  System: Windows 8.1 OS Short Description: Add Book  Designed by: Pratik Baishnav Design Date: 2015-05-20  Executed by: Pratik Baishnav Execution Date: 2015-05-20 |

|  |
| --- |
| Pre-conditions: Application is running. In dashboard, user adds new books  in the system. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | Action | Expected System Response | P/F | Comment |
| 1 | User clicks the button and adds books. | The system process the query and adds new books in the database. |  |  |
| 2 | User clicks on the cancel button. | The system cancels to add new books in the database. |  |  |

|  |
| --- |
| Post Condition: As the user adds or cancels to add new books in the database. |

##### Test Case 04

|  |
| --- |
| Test Case #: 04 Test Case Name: add/edit/delete  System: Windows 8.1 OS Description: Test add, edit, delete  Designed by: Pratik Baishnav Design Date: 2015-05-22  Executed by: Pratik Baishnav Execution Date: 2015-05-22 |

|  |
| --- |
| Pre-conditions: The application has different kinds of button for their specific  actions to be performed. And view completion of those action after that. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | Action | Expected System Response | P/F | Comment |
| 1 | User click different action buttons | The system appears edit menu with add, edit and delete buttons. |  |  |
| 2 | User click and modifies data or records | The system updates the data and records and displays it. |  |  |

|  |
| --- |
| Post Condition: The users successfully add, edit and delete records and display it. |

##### Test Case 05

|  |
| --- |
| Test Case #: 05 Test Case Name: modify users  System: Windows 8.1 OS Short Description: Test users  Designed by: Pratik Baishnav Design Date: 2015-05-26  Executed by: Pratik Baishnav Execution Date: 2015-05-26 |

|  |
| --- |
| Pre-conditions: Admin will to modify and add users. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Step | Action | Expected System Response | P/F | Comment |
| 1 | Admin add, delete | There is user view and admin add or delete specific users. |  |  |
| 2 | Admin modifies users and access type | In same user view, admin can edit user credentials like username, password and user type (here it is of two types: Admin and User) |  |  |

|  |
| --- |
| Post Condition: Users are controlled by Admin and changes credentials. |

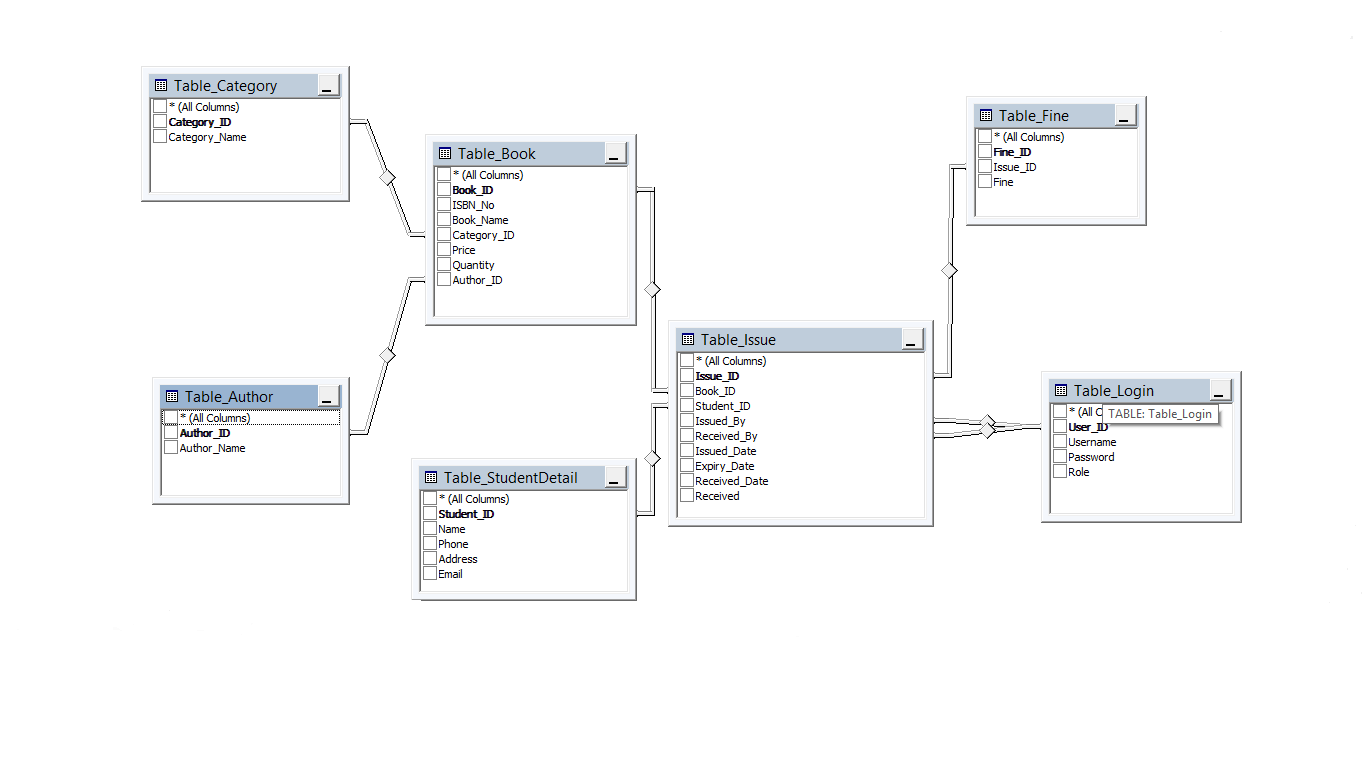
## 9. DATA DICTONARY

### 9.1. Schemas

A schema shows the organizational structure of a database. It should show the entities (the tables in the database), and their characteristics (i.e. their properties/attributes/fields). It should clearly identify the primary key in each table and the links and relationships between tables. The example below is just one way a

schema can be represented. There are other acceptable methods that students can use to present this information.

Figure 4: Schema



**Author Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Author\_ID | int |
| Author\_Name | nvarchar(50) |

Table 1: Table\_Author

**Book Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Book\_ID | int |
| ISBN\_No | nvarchar(50) |
| Book\_Name | nvarchar(50) |
| Category\_ID | int |
| Price | decimal(18, 2) |
| Quantity | int |
| Author\_ID | int |

Table 2: Table\_Book

**Category Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Category\_ID | int |
| Category\_Name | nvarchar(50) |

Table 3: Table\_Category

**Fine Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Fine\_ID | int |
| Issue\_ID | int |
| Fine | nvarchar(50) |

Table 4: Table\_Fine

**Issue Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Issue\_ID | int |
| Book\_ID | int |
| Student\_ID | int |
| Issued\_By | int |
| Received\_By | int |
| Issued\_Date | datetime |
| Expiry\_Date | date |
| Received\_Date | date |
| Received | nvarchar(50) |

Table 5: Table\_Issue

**Login Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| User\_ID | int |
| Username | nvarchar(50) |
| Password | nvarchar(50) |
| Role | nvarchar(50) |

**Table 6: Table\_Login**

**Student Detail Table**

|  |  |
| --- | --- |
| **Column** | **Data Type** |
| Student\_ID | int |
| Name | nvarchar(50) |
| Phone | nvarchar(50) |
| Address | nvarchar(50) |
| Email | nvarchar(50) |

**Table 7: Table\_StudentDetail**

## 

## 10.SNAPSHOTS AND CODE OF PROJECT

### 10.1 SNAPSHOTS

Figure 5: Login Panel

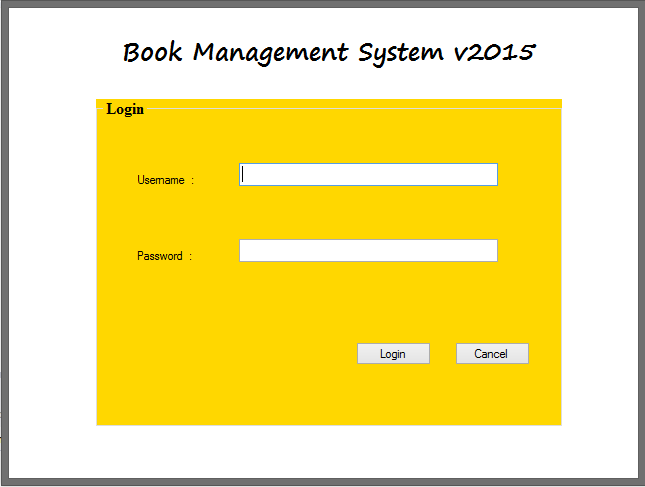


Figure 6: Admin Form

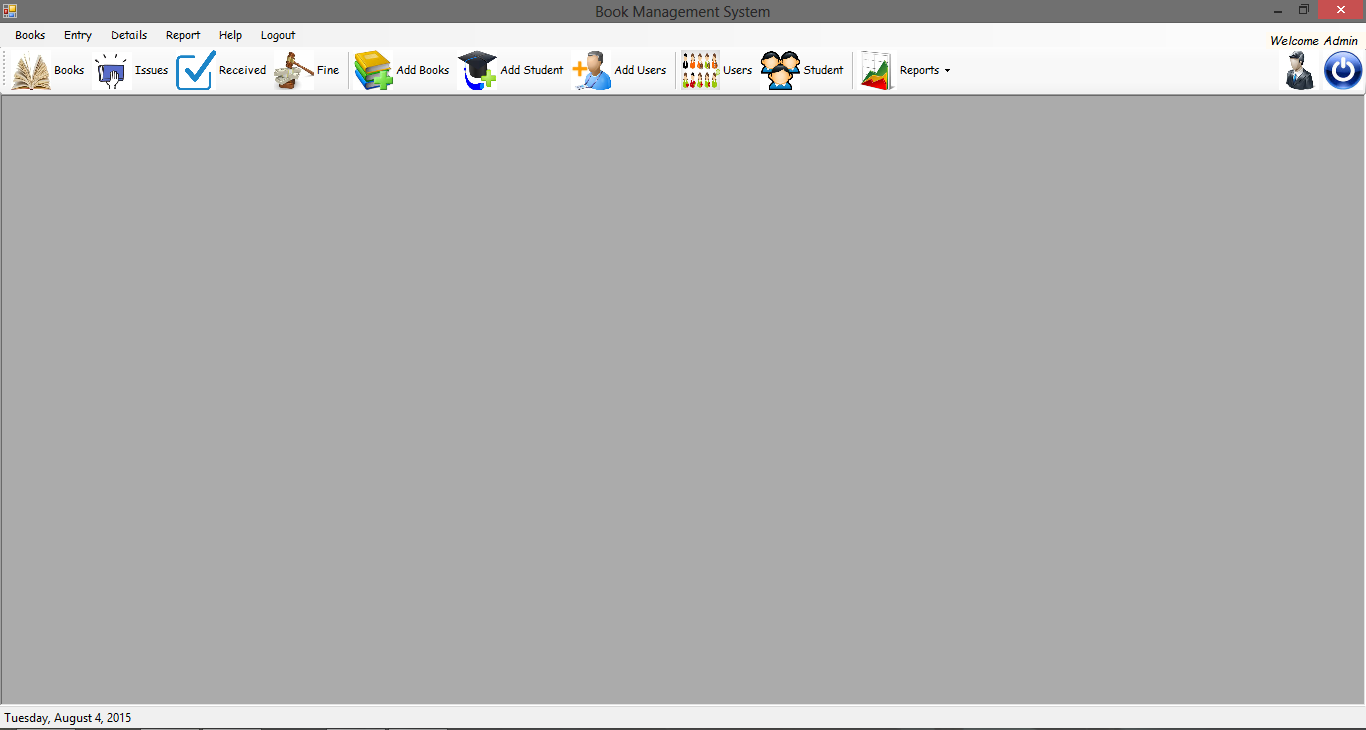


Figure 7: User Form

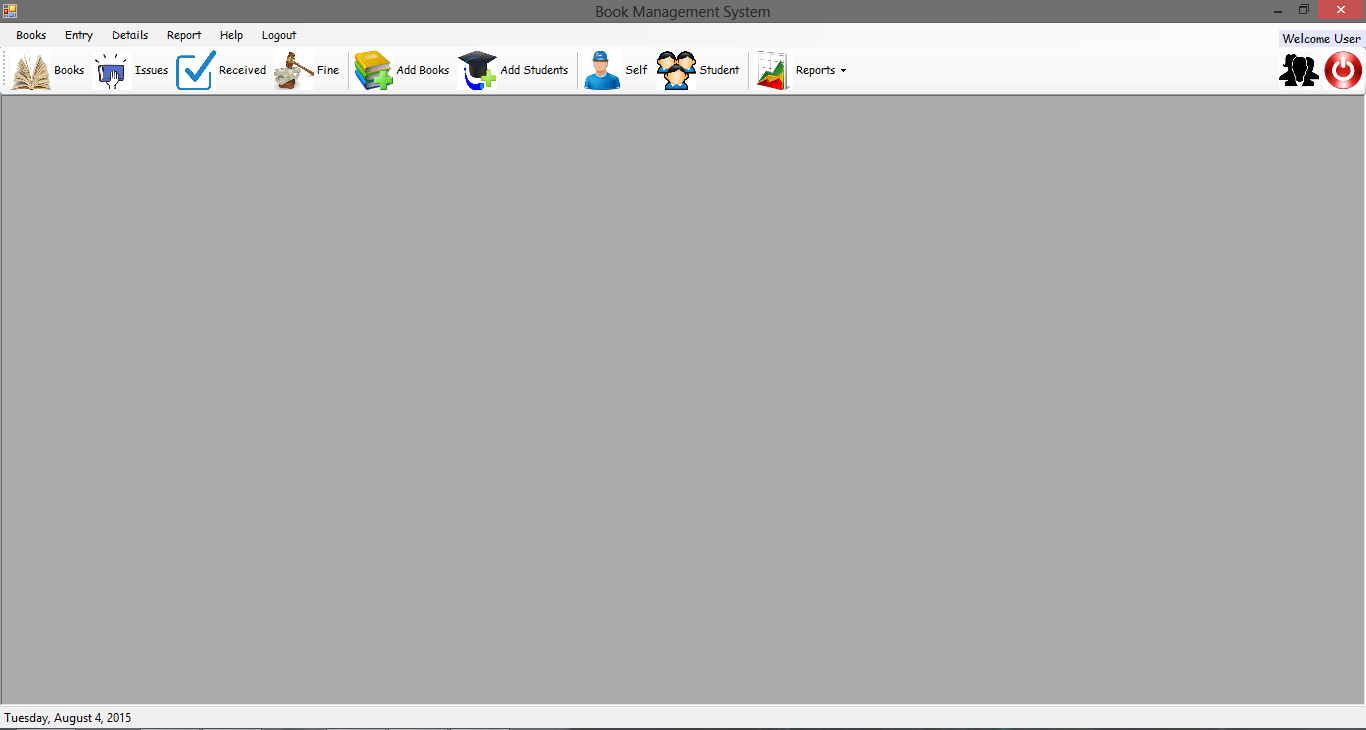


Figure 8: Add User

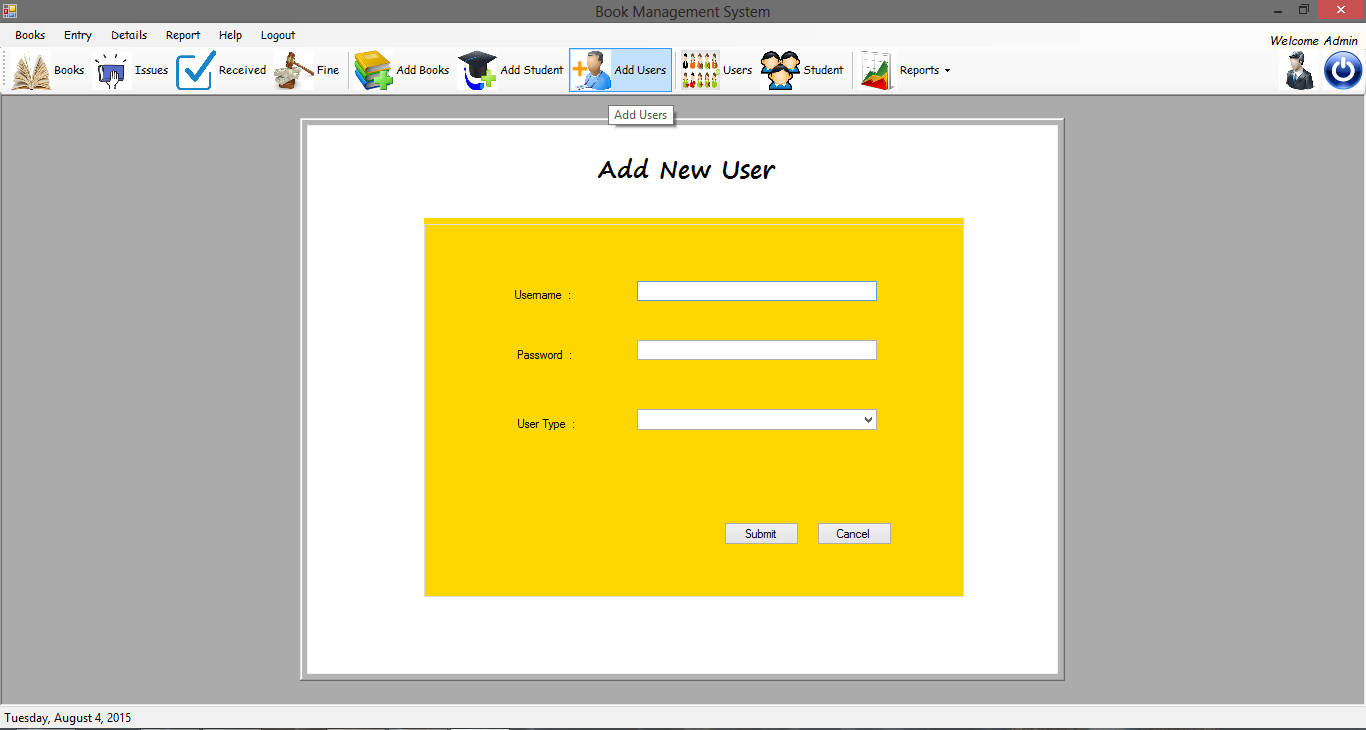


Figure 9: Add Student

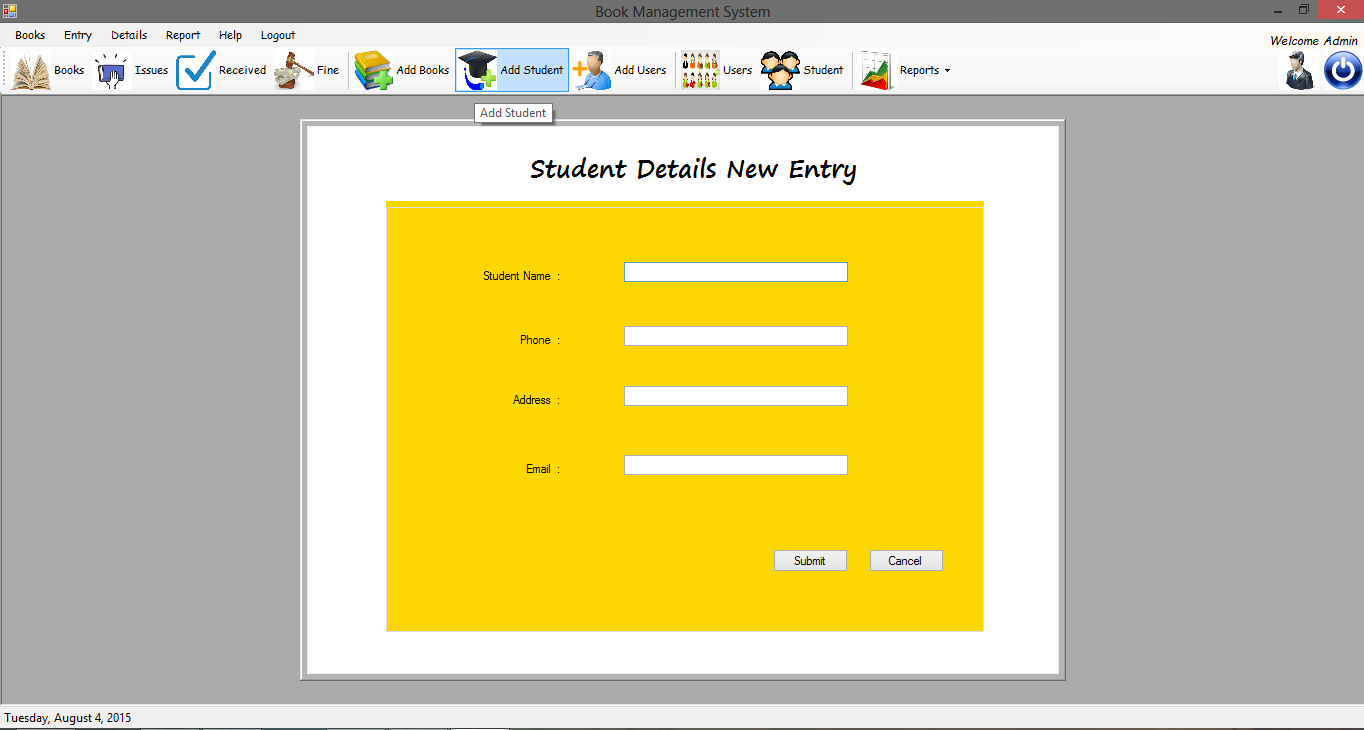


Figure 10: Add Book

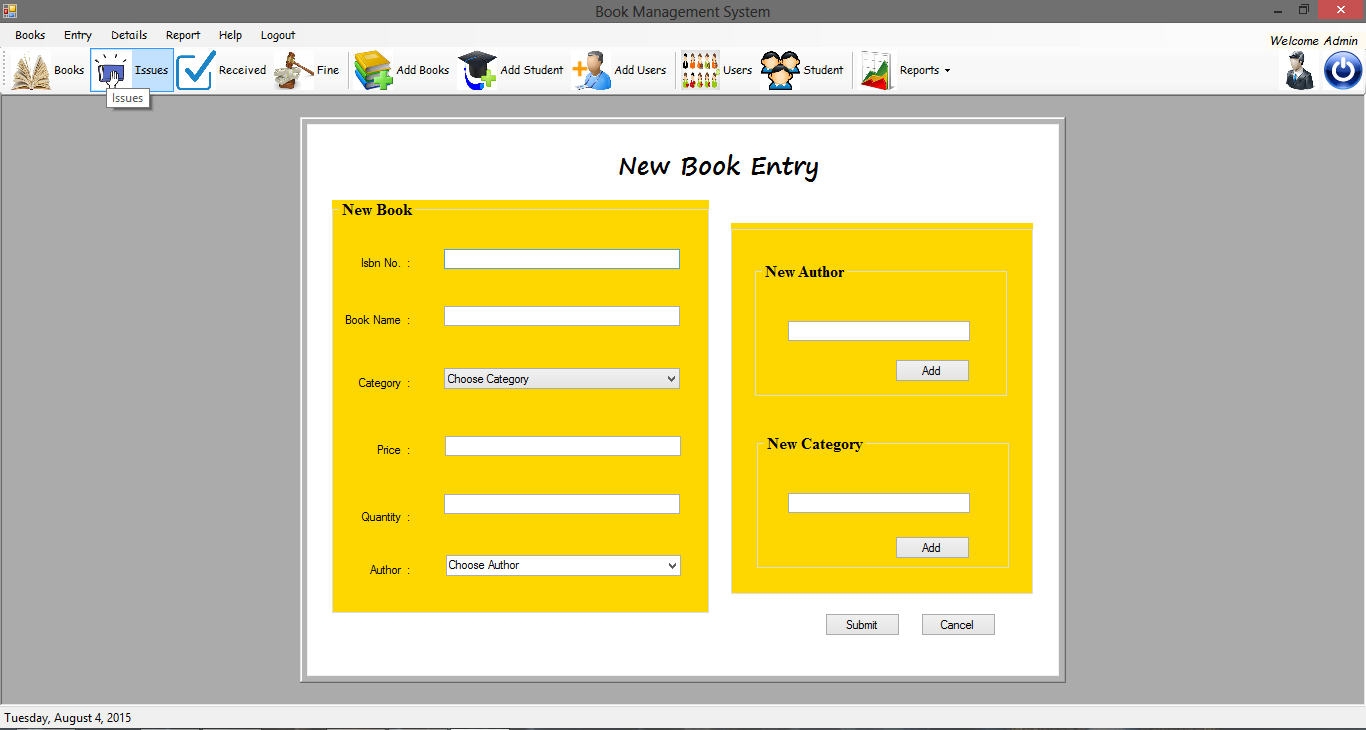


Figure 11: User Details

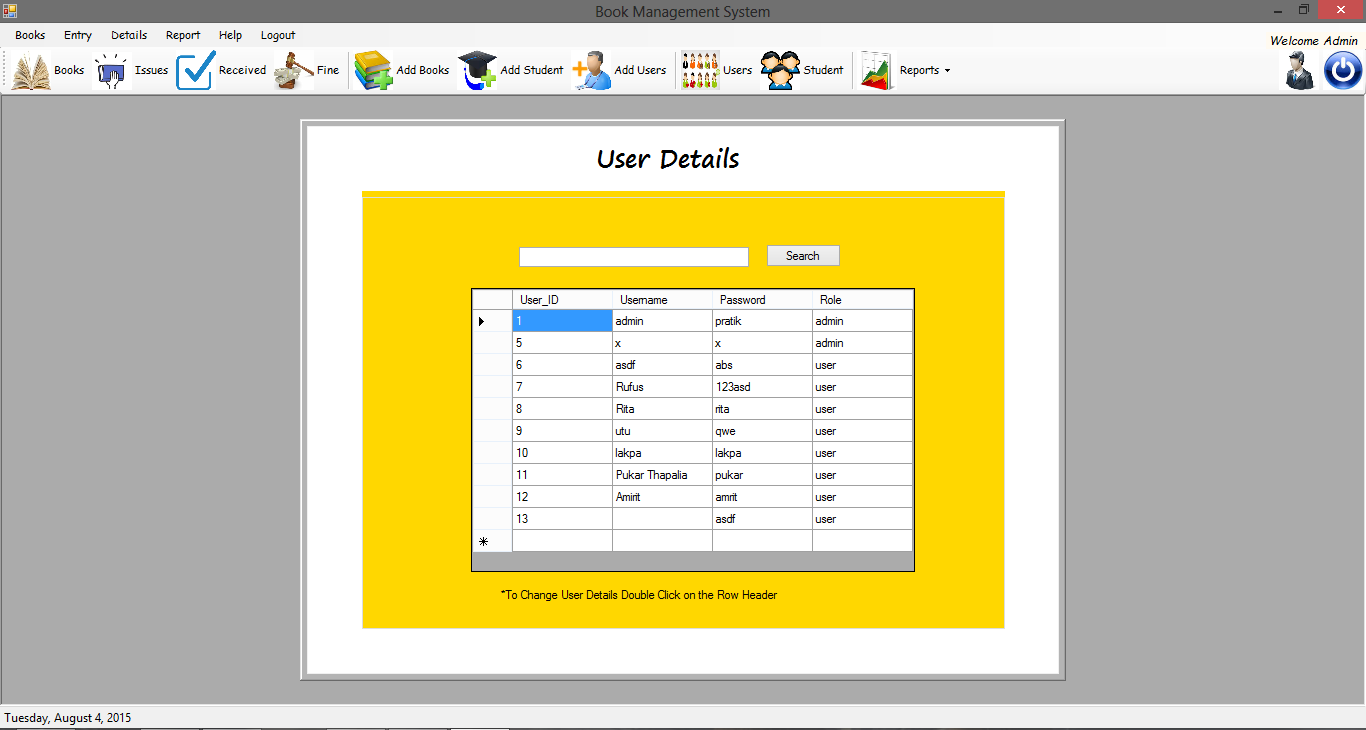


Figure 12: View Books

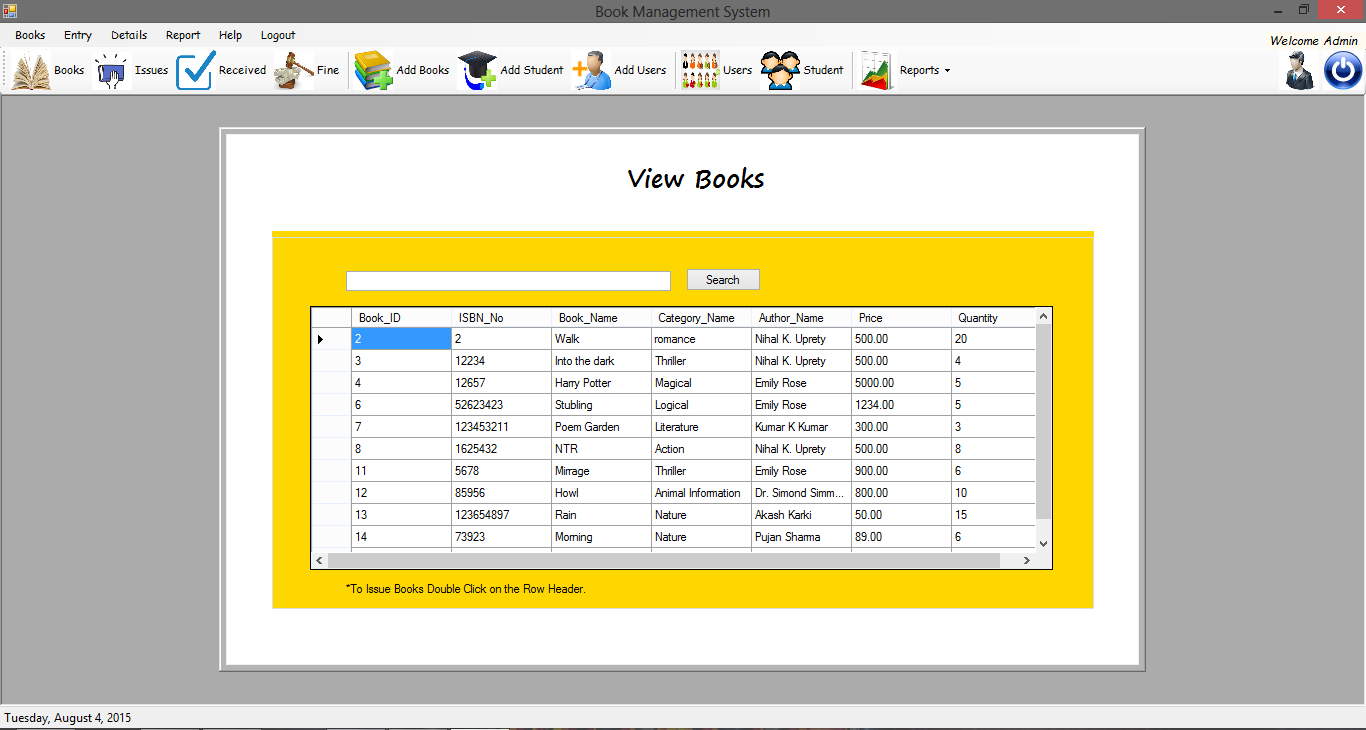


Figure 13: Change User Detail

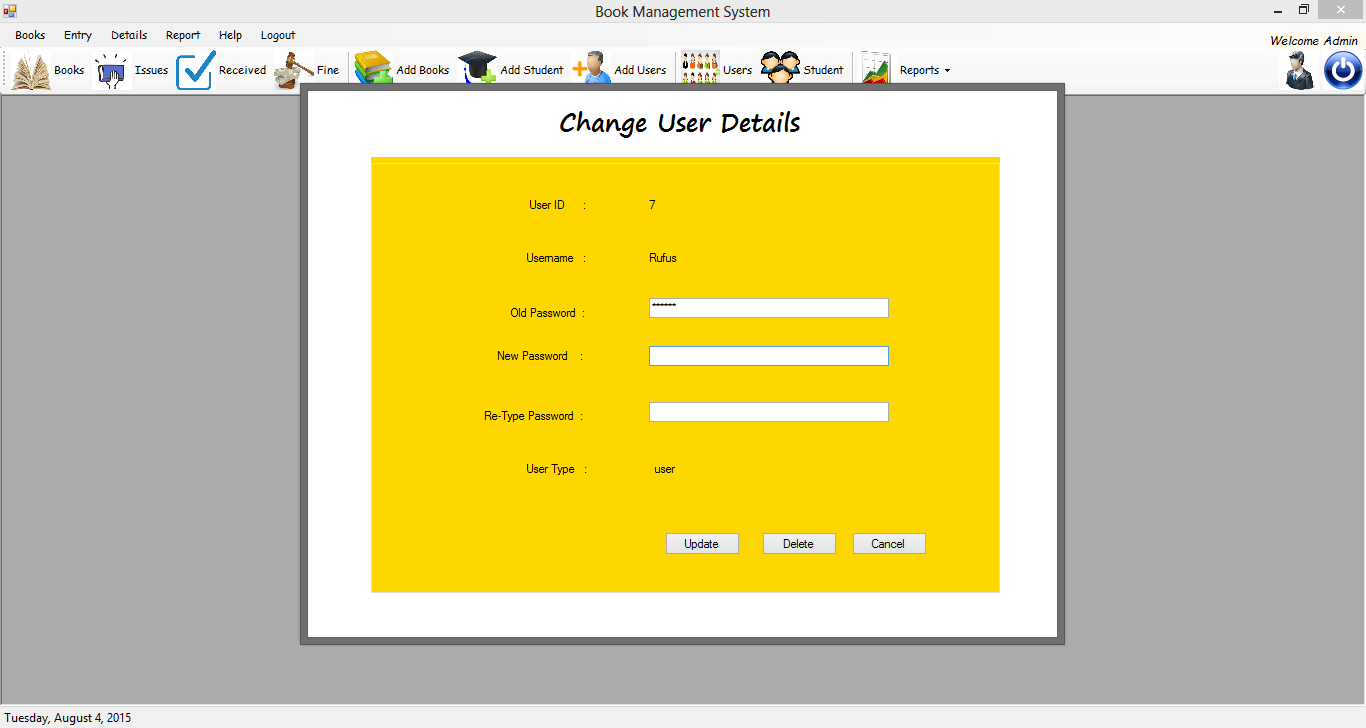


Figure 14: Change Book Details

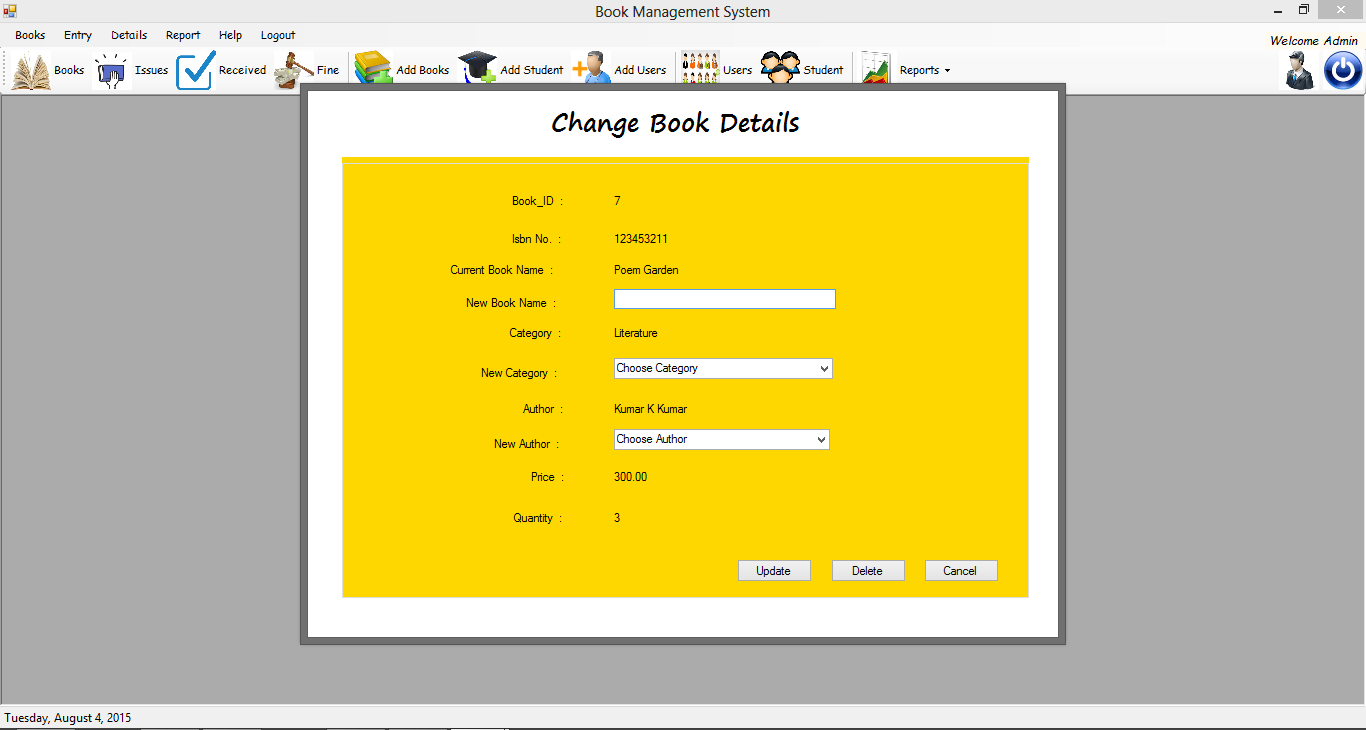


Figure 15: View Fine

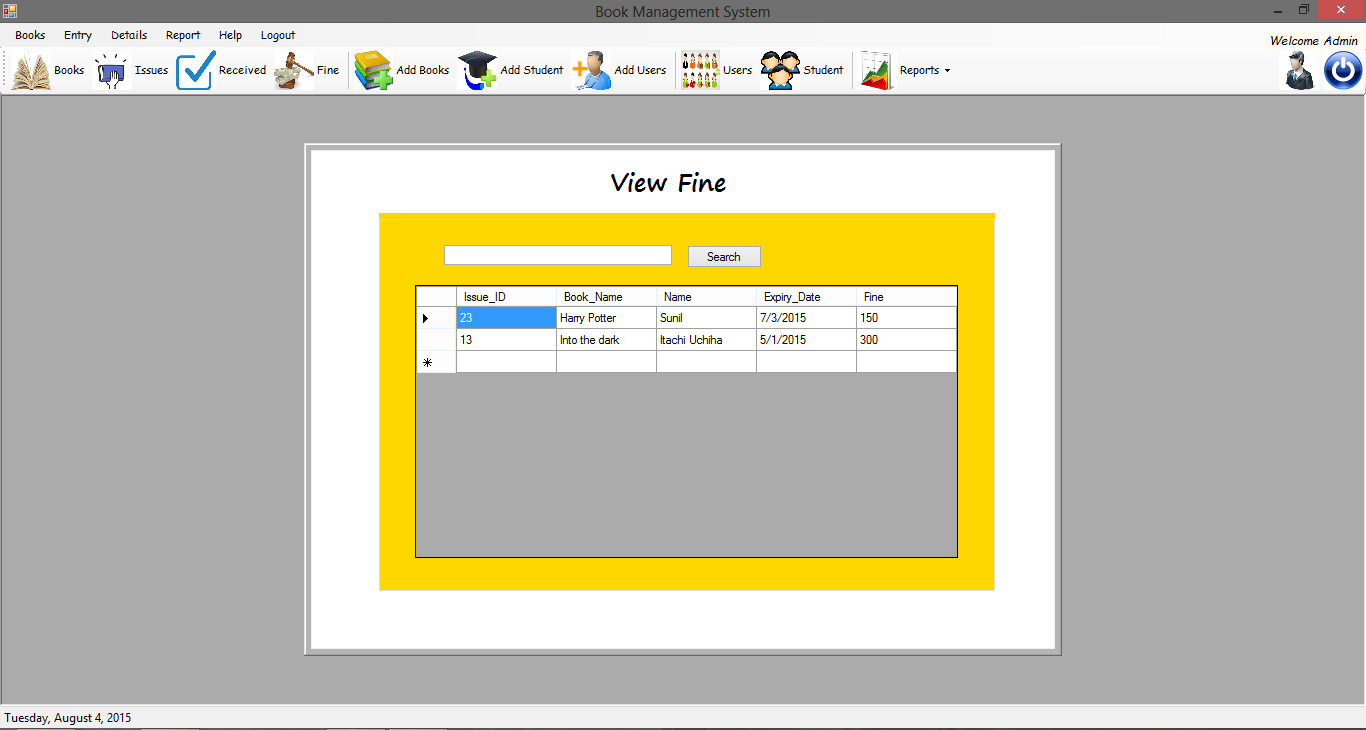
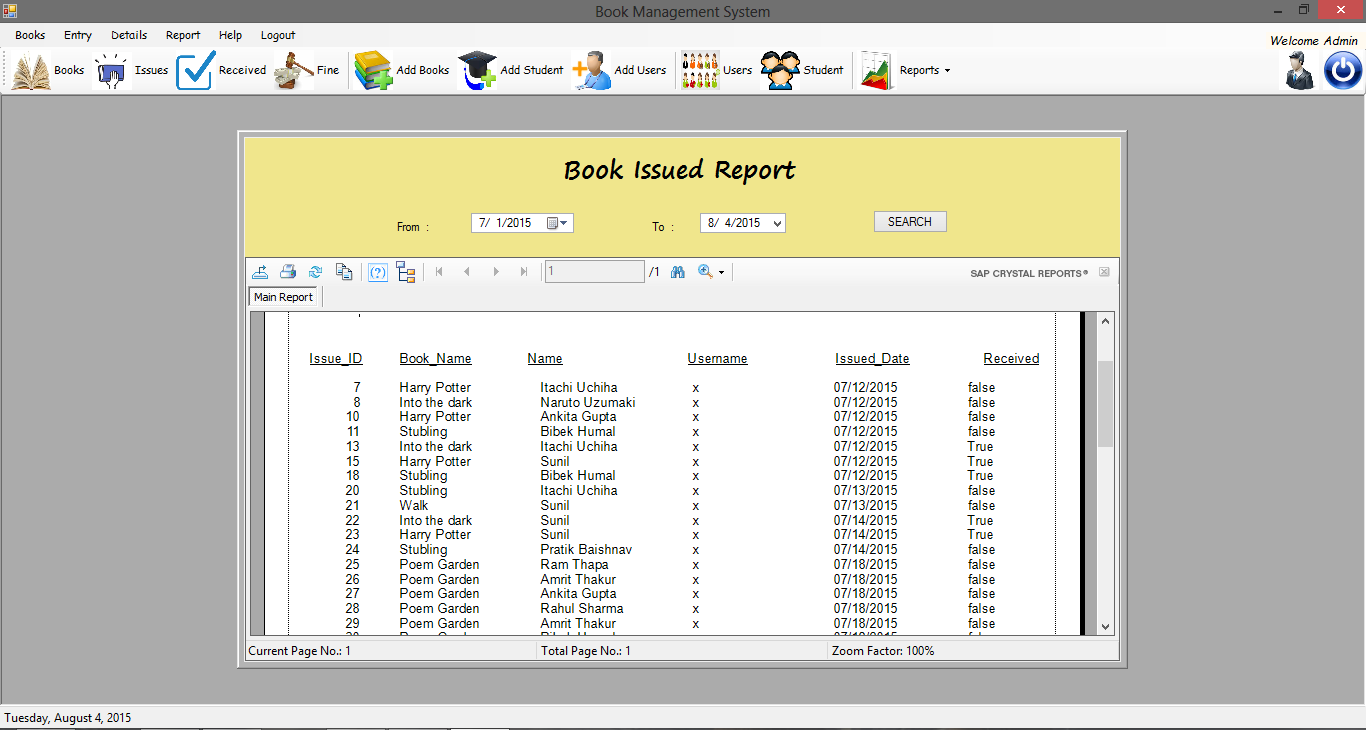


Figure 16: Report



**10.2 SOURCE CODE**

* **Login**

string connectionStr = "data source=DELL\\SQLEXPRESS; Integrated Security=true; Initial Catalog=BookMgmtSystem;";

private void btnLogin\_Click\_1(object sender, EventArgs e)

{

string username = txtUsername.Text;

string password = txtPassword.Text;

if (username == "" && password == "")

{

MessageBox.Show("No Username or Password");

}

else if (username == "")

{

MessageBox.Show("Please enter a Username");

}

else if (password == "")

{

MessageBox.Show("Please enter a Password");

}

else

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Username=@a and Password=@b";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtUsername.Text);

cmd.Parameters.AddWithValue("@b", txtPassword.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

DataTable dt = ds.Tables["t"];

if (dt.Rows.Count > 0 && dt.Rows[0][3].ToString() == "admin")

{

MDIAdmin frm = new MDIAdmin();

Program.User\_ID = Convert.ToInt32(dt.Rows[0][0].ToString());

Program.Username = dt.Rows[0][1].ToString();

Program.Usertype = "admin";

frm.Show();

this.Hide();

}

else if (dt.Rows.Count > 0 && dt.Rows[0][3].ToString() == "user")

{

MDIUser frmUser = new MDIUser();

Program.User\_ID = Convert.ToInt32(dt.Rows[0][0].ToString());

Program.Username = dt.Rows[0][1].ToString();

Program.Usertype = "user";

frmUser.Show();

this.Hide();

}

else

{

MessageBox.Show("Invalid Username or Password");

txtUsername.Text = "";

txtPassword.Text = "";

txtUsername.Focus();

}

}

}

private void btnCancel\_Click\_1(object sender, EventArgs e)

{

this.Close();

}

private void FormLogin\_Load(object sender, EventArgs e)

{

}

* **Add Book**

string connectionStr = "data source=DELL\\SQLEXPRESS; Integrated Security=true; Initial Catalog=BookMgmtSystem;";

private void btnSubmit\_Click(object sender, EventArgs e)

{

if (txtBookname.Text == "" || txtIsbnno.Text == "" || txtPrice.Text == "" || txtQuantity.Text == "" || cbxAuthor.SelectedValue.ToString() == "" || cbxCategory.SelectedValue.ToString() == "")

{

MessageBox.Show("Any of the above fields cannot be empty");

}

else if (Convert.ToInt32(txtIsbnno.Text.ToString()) < 0)

{

MessageBox.Show("The Isbn No. Cannot Be Negative");

txtIsbnno.Text = "";

txtIsbnno.Focus();

}

else

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Book where ISBN\_No=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtIsbnno.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

DataTable dt = ds.Tables["t"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The Book containing this ISBN No. Already Exists");

}

else

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "insert into Table\_Book(ISBN\_No,Book\_Name,Category\_ID,Price,Quantity,Author\_ID) values(@g,@b,@c,@d,@e,@f)";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@g", txtIsbnno.Text);

cmd2.Parameters.AddWithValue("@b", txtBookname.Text);

cmd2.Parameters.AddWithValue("@c", Convert.ToInt32(cbxCategory.SelectedValue.ToString()));

cmd2.Parameters.AddWithValue("@d", Convert.ToDecimal(txtPrice.Text));

cmd2.Parameters.AddWithValue("@e", Convert.ToInt32(txtQuantity.Text));

cmd2.Parameters.AddWithValue("@f", Convert.ToInt32(cbxAuthor.SelectedValue.ToString()));

con2.Open();

int i = cmd2.ExecuteNonQuery();

con2.Close();

if (i > 0)

{

MessageBox.Show("Book Added Successfully");

txtIsbnno.Text = "";

txtBookname.Text = "";

txtNewcategory.Text = "";

txtPrice.Text = "";

txtQuantity.Text = "";

txtNewAuthor.Text = "";

cbxAuthor.Text = "Choose Author";

cbxCategory.Text = "Choose Category";

txtIsbnno.Focus();

}

else

{

MessageBox.Show("Book Addition Unsuccessful");

txtIsbnno.Text = "";

txtBookname.Text = "";

txtNewcategory.Text = "";

txtPrice.Text = "";

txtQuantity.Text = "";

txtNewAuthor.Text = "";

cbxAuthor.Text = "Choose Author";

cbxCategory.Text = "Choose Category";

txtIsbnno.Focus();

}

}

}

}

private void btnNewauthor\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "insert into Table\_Author values(@a)";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtNewAuthor.Text);

con.Open();

int i = cmd.ExecuteNonQuery();

con.Close();

if (i > 0)

{

MessageBox.Show("New Author Created Successfully");

}

else

{

MessageBox.Show("New Author Creation Unsuccessful");

}

LoadAuthor();

}

private void btnNewcategory\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "insert into Table\_Category values(@a)";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtNewcategory.Text);

con.Open();

int i = cmd.ExecuteNonQuery();

con.Close();

if (i > 0)

{

MessageBox.Show("New Category Created Successfully");

}

else

{

MessageBox.Show("New Category Creation Unsuccessful");

}

LoadCategory();

}

public void LoadAuthor()

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Author";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

DataTable dt = ds.Tables["t"];

if (dt.Rows.Count > 0)

{

//inserting choose author

DataRow dr = dt.NewRow();

dr["Author\_Name"] = "Choose Author";

dt.Rows.InsertAt(dr, 0);

cbxAuthor.DataSource = dt;

cbxAuthor.DisplayMember = "Author\_Name";

cbxAuthor.ValueMember = "Author\_ID";

}

}

public void LoadCategory()

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Category";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

//inserting choose category

DataRow dr = dt.NewRow();

dr["Category\_Name"] = "Choose Category";

dt.Rows.InsertAt(dr, 0);

cbxCategory.DataSource = dt;

cbxCategory.DisplayMember = "Category\_Name";

cbxCategory.ValueMember = "Category\_ID";

}

}

private void FormEntrybook\_Load(object sender, EventArgs e)

{

LoadAuthor();

LoadCategory();

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

* **Add User**

string connectionStr = "Data Source=DELL\\SQLEXPRESS; Integrated Security=True; Initial Catalog=BookMgmtSystem;";

private void btnSubmit\_Click\_1(object sender, EventArgs e)

{

if (txtUsername.Text == "")

{

MessageBox.Show("Please enter an username.");

}

else if (txtPassword.Text == "")

{

MessageBox.Show("Please enter a password.");

}

else

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Username = @username";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@username", txtUsername.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("This username already in use.");

txtUsername.Focus();

}

else

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "insert into Table\_Login(Username,Password,Role) values(@a,@b,@c)";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", txtUsername.Text);

cmd2.Parameters.AddWithValue("@b", txtPassword.Text);

cmd2.Parameters.AddWithValue("@c", cbxUserType.Text);

con2.Open();

int i = cmd2.ExecuteNonQuery();

con2.Close();

if (i > 0)

{

MessageBox.Show("User Created Successfully");

txtPassword.Text = "";

txtUsername.Text = "";

cbxUserType.Text = "";

txtUsername.Focus();

}

else

{

MessageBox.Show("User Creation Unsuccessful");

txtPassword.Text = "";

txtUsername.Text = "";

cbxUserType.Text = "";

txtUsername.Focus();

}

}

}

}

private void btnCancel\_Click\_1(object sender, EventArgs e)

{

this.Close();

}

* **Add Student**

string connectionStr = "Data Source=DELL\\SQLEXPRESS; Integrated Security=True; Initial Catalog=BookMgmtSystem;";

private void btnSubmit\_Click\_1(object sender, EventArgs e)

{

if (txtAddress.Text == "" || txtEmail.Text == "" || txtPhone.Text == "" || txtStudentName.Text == "")

{

MessageBox.Show("Any of the above fields cannot be empty");

}

else if (Convert.ToInt32(txtPhone.Text.Length.ToString()) <= 9)

{

MessageBox.Show("Invalid NUMBER");

}

else

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_StudentDetail where Phone=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtPhone.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The Student with This Phone Number already Exists");

txtAddress.Text = "";

txtEmail.Text = "";

txtPhone.Text = "";

txtStudentName.Text = "";

txtStudentName.Focus();

}

else

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "insert into Table\_StudentDetail(Name,Phone,Address,Email) values(@a,@b,@c,@d)";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", txtStudentName.Text);

cmd2.Parameters.AddWithValue("@b", txtPhone.Text);

cmd2.Parameters.AddWithValue("@c", txtAddress.Text);

cmd2.Parameters.AddWithValue("@d", txtEmail.Text);

con2.Open();

int i = cmd2.ExecuteNonQuery();

con2.Close();

if (i > 0)

{

MessageBox.Show("Student Details Entered Successfully");

txtAddress.Text = "";

txtEmail.Text = "";

txtPhone.Text = "";

txtStudentName.Text = "";

txtStudentName.Focus();

}

else

{

MessageBox.Show("Student Details Entered Unsuccessful");

txtAddress.Text = "";

txtEmail.Text = "";

txtPhone.Text = "";

txtStudentName.Text = "";

txtStudentName.Focus();

}

}

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

* **View Books**

string connectionStr = "Data Source=DELL\\SQLEXPRESS; Integrated Security=True; Initial Catalog=BookMgmtSystem;";

private void FormView\_Load(object sender, EventArgs e)

{

SqlConnection con=new SqlConnection(connectionStr);

string sql = "SELECT Table\_Book.Book\_ID, Table\_Book.ISBN\_No, Table\_Book.Book\_Name, Table\_Category.Category\_Name, Table\_Author.Author\_Name, Table\_Book.Price, Table\_Book.Quantity FROM Table\_Author INNER JOIN Table\_Book ON Table\_Author.Author\_ID = Table\_Book.Author\_ID INNER JOIN Table\_Category ON Table\_Book.Category\_ID = Table\_Category.Category\_ID";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

datagridViewBooks.DataSource = ds.Tables["a"];

}

private void datagridViewBooks\_RowHeaderMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDoIssue fdi = new FormDoIssue();

fdi.lblBookIdDoIssue.Text = datagridViewBooks.CurrentRow.Cells[0].Value.ToString();

fdi.Show();

this.Hide();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Book where Book\_Name=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

if (ds.Tables["a"].Rows.Count > 0)

{

datagridViewBooks.DataSource = ds.Tables["a"];

}

else

{

MessageBox.Show("Such Book name Does not exsists");

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Book where Book\_Name like @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text + "%");

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

datagridViewBooks.DataSource = ds.Tables["t"];

}

* **Issue Books**

private void btnIssue\_Click\_1(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Issue where Student\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtStudentID.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 3)

{

MessageBox.Show("The Student With This Id already exceeded 3 Books.");

}

else

{

DateTime issueDate = DateTime.Today.Date;

DateTime expiryDate = issueDate.AddDays(20).Date;

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "insert into Table\_Issue (Book\_ID,Student\_ID,Issued\_By,Issued\_Date,Expiry\_Date,Received) values(@g,@a,@b,@c,@d,'false')";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@g", Convert.ToInt32(lblBookIdDoIssue.Text));

cmd2.Parameters.AddWithValue("@a", txtStudentID.Text);

cmd2.Parameters.AddWithValue("@b", Program.User\_ID);

cmd2.Parameters.AddWithValue("@c", issueDate);

cmd2.Parameters.AddWithValue("@d", expiryDate);

con2.Open();

int i = cmd2.ExecuteNonQuery();

con2.Close();

if (i > 0)

{

MessageBox.Show("Book Issued Successfully");

this.Close();

}

else

{

MessageBox.Show("Book Issue Unsuccessful");

}

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

private void button1\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Issue where Book\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", lblBookIdDoIssue.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 5)

{

MessageBox.Show("Book Out Of Stock.");

}

else

{

MessageBox.Show("Book available");

}

}

* **View Issues**

string connectionStr = "data source=DELL\\SQLEXPRESS; Integrated Security=True; Initial catalog=BookMgmtSystem;";

private void FormViewIssues\_Load(object sender, EventArgs e)

{

loadgrid();

}

public void loadgrid()

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "SELECT Table\_Issue.Issue\_ID, Table\_Book.Book\_Name, Table\_Issue.Issued\_Date, Table\_Login.Username, Table\_Issue.Expiry\_Date, Table\_StudentDetail.Name FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Login

ON Table\_Issue.Issued\_By = Table\_Login.User\_ID where Table\_Issue.Received='False'";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

datagridViewIssues.DataSource = ds.Tables["a"];

}

* **Change Book Details**

string connectionStr = "data source=DELL\\SQLEXPRESS; integrated security=true; initial catalog=BookMgmtSystem;";

private void btnUpdate\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Book where Book\_Name=@a and Book\_ID=@b";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtNewBookNameChangeBookDetailsUpdate.Text);

cmd.Parameters.AddWithValue("@b", lblBookIdChangeBookDetailsUpdate.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "e");

DataTable dt = ds.Tables["e"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The New Book Name cannot be same as the Old Book Name");

txtNewBookNameChangeBookDetailsUpdate.Text = "";

txtNewBookNameChangeBookDetailsUpdate.Focus();

}

else

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "update Table\_Book set Book\_Name=@b,Category\_ID=@c,Author\_ID=@d where Book\_ID=@a";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", lblBookIdChangeBookDetailsUpdate.Text);

cmd2.Parameters.AddWithValue("@b", txtNewBookNameChangeBookDetailsUpdate.Text);

cmd2.Parameters.AddWithValue("@c", Convert.ToInt32(cbxNewCategoryChangeBookDetailsUpdate.SelectedValue.ToString()));

cmd2.Parameters.AddWithValue("@d", Convert.ToInt32(cbxNewAuthorChangeBookDetailsUpdate.SelectedValue.ToString()));

con2.Open();

cmd2.ExecuteNonQuery();

con2.Close();

MessageBox.Show("Update Successfull");

this.Close();

}

}

public void LoadAuthor()

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Author";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

DataTable dt = ds.Tables["t"];

if (dt.Rows.Count > 0)

{

//inserting choose author

DataRow dr = dt.NewRow();

dr["Author\_Name"] = "Choose Author";

dt.Rows.InsertAt(dr, 0);

cbxNewAuthorChangeBookDetailsUpdate.DataSource = dt;

cbxNewAuthorChangeBookDetailsUpdate.DisplayMember = "Author\_Name";

cbxNewAuthorChangeBookDetailsUpdate.ValueMember = "Author\_ID";

}

}

private void FormDetailsChangeBookDetailsUpdate\_Load(object sender, EventArgs e)

{

LoadAuthor();

LoadCategory();

}

public void LoadCategory()

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Category";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

//inserting choose category

DataRow dr = dt.NewRow();

dr["Category\_Name"] = "Choose Category";

dt.Rows.InsertAt(dr, 0);

cbxNewCategoryChangeBookDetailsUpdate.DataSource = dt;

cbxNewCategoryChangeBookDetailsUpdate.DisplayMember = "Category\_Name";

cbxNewCategoryChangeBookDetailsUpdate.ValueMember = "Category\_ID";

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

private void btnDelete\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Are you sure want to delete?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "delete from Table\_Book where Book\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", lblBookIdChangeBookDetailsUpdate.Text);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Deleted Successfully");

this.Close();

}

}

* **Change User Details**

string connectionStr = "Data Source=DELL\\SQLEXPRESS; Integrated security=True; Initial Catalog=BookMgmtSystem;";

private void btnUpdate\_Click\_1(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Password=@a and User\_ID=@b ";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtPasswordChangeDetails.Text);

cmd.Parameters.AddWithValue("@b", lblUserIdChangeUserDetailsUpdate.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The New Password cannot be same as the Old Password");

txtPasswordChangeDetails.Text = "";

txtRetypePasswordChangeDetails.Text = "";

txtPasswordChangeDetails.Focus();

}

else if(txtPasswordChangeDetails.Text == txtRetypePasswordChangeDetails.Text)

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "Update Table\_Login set Password=@a where User\_ID=@b";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", txtPasswordChangeDetails.Text);

cmd2.Parameters.AddWithValue("@b", lblUserIdChangeUserDetailsUpdate.Text);

con2.Open();

cmd2.ExecuteNonQuery();

con2.Close();

MessageBox.Show("Updated Successfully");

this.Close();

}

else

{

MessageBox.Show("The above Passwords did not matched");

txtPasswordChangeDetails.Text = "";

txtRetypePasswordChangeDetails.Text = "";

txtPasswordChangeDetails.Focus();

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

private void btnDelete\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Are you sure want to delete?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "delete from Table\_Login where User\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", lblUserIdChangeUserDetailsUpdate.Text);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Deleted Successfully");

this.Close();

}

}

* **Change Student Details**

string connectionStr = "data source=DELL\\SQLEXPRESS; integrated security=true; initial catalog=BookMgmtSystem;";

private void btnUpdate\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_StudentDetail where Phone=@a and Student\_ID=@b ";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtNewPhoneNoChangeStudentDetailsUpdate.Text);

cmd.Parameters.AddWithValue("@b", lblStudentIdChangeStudentDetailsUpdate.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The New Phone no. cannot be same as the Old Phone no.");

txtNewPhoneNoChangeStudentDetailsUpdate.Text = "";

txtNewPhoneNoChangeStudentDetailsUpdate.Focus();

}

else

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "update Table\_StudentDetail set Phone=@a,Email=@b where Student\_ID=@c";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", txtNewPhoneNoChangeStudentDetailsUpdate.Text);

cmd2.Parameters.AddWithValue("@b", txtNewEmailChangeStudentDetailsUpdate.Text);

cmd2.Parameters.AddWithValue("@c", lblStudentIdChangeStudentDetailsUpdate.Text);

con2.Open();

cmd2.ExecuteNonQuery();

con2.Close();

MessageBox.Show("Updated Successfully");

this.Close();

}

}

private void btnDelete\_Click(object sender, EventArgs e)

{

if (MessageBox.Show("Are you sure want to delete?", "Delete", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "delete from Table\_StudentDetail where Student\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", lblStudentIdChangeStudentDetailsUpdate.Text);

con.Open();

cmd.ExecuteNonQuery();

con.Close();

MessageBox.Show("Deleted Successfully");

this.Close();

}

}

private void btnCancel\_Click(object sender, EventArgs e)

{

this.Close();

}

* **Change Self Details**

string connectionStr = "data source=DELL\\SQLEXPRESS; Integrated Security=true; Initial Catalog=BookMgmtSystem;";

private void FormIndividualUserDetailsUpdate\_Load(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login ";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

DataTable dt = ds.Tables["t"];

lblUserIdIndividualUserDetailsUpdate.Text = Program.User\_ID.ToString();

lblUsernameIdIndividualUserDetailsUpdate.Text = dt.Rows[0][1].ToString();

lblUserTypeIndividualUserDetailsUpdate.Text = dt.Rows[0][3].ToString();

txtOldPasswordIndividualUserDetailsUpdate.Text = dt.Rows[0][2].ToString();

}

private void btnUpdate\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Password=@a and User\_ID=@b ";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtPasswordIndividualUserDetailsUpdate.Text);

cmd.Parameters.AddWithValue("@b", Program.User\_ID);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

if (dt.Rows.Count > 0)

{

MessageBox.Show("The New Password cannot be same as the Old Password");

txtPasswordIndividualUserDetailsUpdate.Text = "";

txtRetypePasswordIndividualUserDetailsUpdate.Text = "";

txtPasswordIndividualUserDetailsUpdate.Focus();

}

else if (txtPasswordIndividualUserDetailsUpdate.Text == txtRetypePasswordIndividualUserDetailsUpdate.Text)

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "Update Table\_Login set Password=@a where User\_ID=@b";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", txtPasswordIndividualUserDetailsUpdate.Text);

cmd2.Parameters.AddWithValue("@b", Program.User\_ID);

con2.Open();

cmd2.ExecuteNonQuery();

con2.Close();

MessageBox.Show("Updated Successfully");

this.Close();

}

else

{

MessageBox.Show("The above Passwords did not matched");

txtPasswordIndividualUserDetailsUpdate.Text = "";

txtRetypePasswordIndividualUserDetailsUpdate.Text = "";

txtPasswordIndividualUserDetailsUpdate.Focus();

}

}

* **Receive Books**

private void datagridViewIssues\_RowHeaderMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

int Issue\_ID = Convert.ToInt32(datagridViewIssues.CurrentRow.Cells[0].Value.ToString());

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select Expiry\_Date from Table\_Issue where Issue\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", Issue\_ID);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

DateTime expiredDate = Convert.ToDateTime(dt.Rows[0][0].ToString());

int Expired = expiredDate.CompareTo(DateTime.Now);

if (Expired == -1)

{

int fine;

int leftDays =( DateTime.Now - expiredDate).Days;

if (leftDays < 8)

fine = 100;

else if (leftDays < 15)

fine = 150;

else if (leftDays < 22)

fine = 200;

else

fine = 300;

if (MessageBox.Show("This issue has been expired and the fine is Rs."+fine+". Paid?", "Book Return", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

//update

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "Update Table\_Issue set Received\_By=@a,Received\_Date=@b,Received='True' where Issue\_ID=@c";

SqlCommand cmd2 = new SqlCommand(sql2,con2);

cmd2.Parameters.AddWithValue("@a", Program.User\_ID);

cmd2.Parameters.AddWithValue("@b", DateTime.Now);

cmd2.Parameters.AddWithValue("@c", Issue\_ID);

con2.Open();

int i= cmd2.ExecuteNonQuery();

con2.Close();

//insert

SqlConnection con3 = new SqlConnection(connectionStr);

string sql3 = "Insert into Table\_Fine(Issue\_ID,Fine) values(@a,@b)";

SqlCommand cmd3 = new SqlCommand(sql3, con3);

cmd3.Parameters.AddWithValue("@a", Issue\_ID);

cmd3.Parameters.AddWithValue("@b", fine);

con3.Open();

int j= cmd3.ExecuteNonQuery();

con3.Close();

if (i > 0 && j > 0)

{

MessageBox.Show("Book Received successfully");

loadgrid();

}

}

}

else

{

if (MessageBox.Show("Returned?", "Book Return", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "Update Table\_Issue set Received\_By=@a,Received\_Date=@b,Received='True' where Issue\_ID=@c";

SqlCommand cmd2 = new SqlCommand(sql2, con2);

cmd2.Parameters.AddWithValue("@a", Program.User\_ID);

cmd2.Parameters.AddWithValue("@b", DateTime.Now);

cmd2.Parameters.AddWithValue("@c", Issue\_ID);

con2.Open();

int i = cmd2.ExecuteNonQuery();

con2.Close();

if (i > 0)

{

MessageBox.Show("Book Received successfully");

loadgrid();

}

}

}

* **View Received Books**

private void btnSearch\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "SELECT Table\_Issue.Issue\_ID, Table\_Book.Book\_Name, Table\_Issue.Issued\_Date, Table\_Login.Username, Table\_Issue.Expiry\_Date, Table\_StudentDetail.Name FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Login ON Table\_Issue.Issued\_By = Table\_Login.User\_ID where Table\_Issue.Received='False' and Table\_Book.Book\_Name=@a or Table\_StudentDetail.Name=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text);

// cmd.Parameters.AddWithValue("@b", Convert.ToInt32(txtSearchId.Text.ToString()));

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

if (ds.Tables["a"].Rows.Count > 0)

{

datagridViewIssues.DataSource = ds.Tables["a"];

}

else

{

MessageBox.Show("No Issue Found");

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "SELECT Table\_Issue.Issue\_ID, Table\_Book.Book\_Name, Table\_Issue.Issued\_Date, Table\_Login.Username, Table\_Issue.Expiry\_Date, Table\_StudentDetail.Name FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Login ON Table\_Issue.Issued\_By = Table\_Login.User\_ID where Table\_Issue.Received='False' and Table\_Book.Book\_Name like @a or Table\_StudentDetail.Name like @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text + "%");

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

datagridViewIssues.DataSource = ds.Tables["t"];

}

* **Charge Fine**

int Issue\_ID = Convert.ToInt32(datagridViewIssues.CurrentRow.Cells[0].Value.ToString());

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select Expiry\_Date from Table\_Issue where Issue\_ID=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", Issue\_ID);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

DataTable dt = ds.Tables["a"];

DateTime expiredDate = Convert.ToDateTime(dt.Rows[0][0].ToString());

int Expired = expiredDate.CompareTo(DateTime.Now);

if (Expired == -1)

{

int fine;

int leftDays =( DateTime.Now - expiredDate).Days;

if (leftDays < 8)

fine = 100;

else if (leftDays < 15)

fine = 150;

else if (leftDays < 22)

fine = 200;

else

fine = 300;

if (MessageBox.Show("This issue has been expired and the fine is Rs."+fine+". Paid?", "Book Return", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

//update

SqlConnection con2 = new SqlConnection(connectionStr);

string sql2 = "Update Table\_Issue set Received\_By=@a,Received\_Date=@b,Received='True' where Issue\_ID=@c";

SqlCommand cmd2 = new SqlCommand(sql2,con2);

cmd2.Parameters.AddWithValue("@a", Program.User\_ID);

cmd2.Parameters.AddWithValue("@b", DateTime.Now);

cmd2.Parameters.AddWithValue("@c", Issue\_ID);

con2.Open();

int i= cmd2.ExecuteNonQuery();

con2.Close();

//insert

SqlConnection con3 = new SqlConnection(connectionStr);

string sql3 = "Insert into Table\_Fine(Issue\_ID,Fine) values(@a,@b)";

SqlCommand cmd3 = new SqlCommand(sql3, con3);

cmd3.Parameters.AddWithValue("@a", Issue\_ID);

cmd3.Parameters.AddWithValue("@b", fine);

con3.Open();

int j= cmd3.ExecuteNonQuery();

con3.Close();

if (i > 0 && j > 0)

{

MessageBox.Show("Book Received successfully");

loadgrid();

}

* **View Fine**

string connectionstr = "data source=DELL\\SQLEXPRESS; Integrated Security=True; Initial catalog=BookMgmtSystem;";

private void FormViewFine\_Load(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionstr);

string sql = "SELECT Table\_Fine.Issue\_ID, Table\_Book.Book\_Name, Table\_StudentDetail.Name, Table\_Issue.Expiry\_Date, Table\_Fine.Fine FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Fine ON Table\_Issue.Issue\_ID = Table\_Fine.Issue\_ID";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

datagridViewFine.DataSource = ds.Tables["a"];

}

private void btnSearch\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionstr);

string sql = "SELECT Table\_Fine.Issue\_ID, Table\_Book.Book\_Name, Table\_StudentDetail.Name, Table\_Issue.Expiry\_Date, Table\_Fine.Fine FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Fine ON Table\_Issue.Issue\_ID = Table\_Fine.Issue\_ID where Table\_Book.Book\_Name=@a or Table\_StudentDetail.Name=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

if (ds.Tables["a"].Rows.Count > 0)

{

datagridViewFine.DataSource = ds.Tables["a"];

}

else

{

MessageBox.Show("No Fine Charged");

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionstr);

string sql = "SELECT Table\_Fine.Issue\_ID, Table\_Book.Book\_Name, Table\_StudentDetail.Name, Table\_Issue.Expiry\_Date, Table\_Fine.Fine FROM Table\_Book INNER JOIN Table\_Issue ON Table\_Book.Book\_ID = Table\_Issue.Book\_ID INNER JOIN Table\_StudentDetail ON Table\_Issue.Student\_ID = Table\_StudentDetail.Student\_ID INNER JOIN Table\_Fine ON Table\_Issue.Issue\_ID = Table\_Fine.Issue\_ID where Table\_Book.Book\_Name like @a or Table\_StudentDetail.Name like @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text + "%");

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

datagridViewFine.DataSource = ds.Tables["a"];

}

* **View User Details**

private void FormDetailsChangeUserDetails\_Load(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

datagridUserDetails.DataSource = ds.Tables["t"];

}

private void datagridUserDetails\_RowHeaderMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeUserDetailsUpdate frm = new FormDetailsChangeUserDetailsUpdate();

frm.lblUserIdChangeUserDetailsUpdate.Text = datagridUserDetails.CurrentRow.Cells[0].Value.ToString();

frm.lblUsernameChangeUserDetailsUpdate.Text = datagridUserDetails.CurrentRow.Cells[1].Value.ToString();

frm.txtOldPasswordUserDetailsUpdate.Text = datagridUserDetails.CurrentRow.Cells[2].Value.ToString();

frm.lblUserTypeChangeUserDetailsUpdate.Text = datagridUserDetails.CurrentRow.Cells[3].Value.ToString();

frm.Show();

this.Hide();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Username = @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text );

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

if (ds.Tables["t"].Rows.Count > 0)

{

datagridUserDetails.DataSource = ds.Tables["t"];

}

else

{

MessageBox.Show("Username Does not exsists");

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_Login where Username like @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text + "%");

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

datagridUserDetails.DataSource = ds.Tables["t"];

}

* **View Student Details**

string connectionStr = "data source=DELL\\SQLEXPRESS; integrated security=true; initial catalog=BookMgmtSystem;";

private void FormDetailsChangeStudentDetails\_Load(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_StudentDetail";

SqlCommand cmd = new SqlCommand(sql, con);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "a");

datagridStudentDetails.DataSource = ds.Tables["a"];

}

private void datagridStudentDetails\_RowHeaderMouseDoubleClick(object sender, DataGridViewCellMouseEventArgs e)

{

FormDetailsChangeStudentDetailsLUpdate frm = new FormDetailsChangeStudentDetailsLUpdate();

frm.lblStudentIdChangeStudentDetailsUpdate.Text = datagridStudentDetails.CurrentRow.Cells[0].Value.ToString();

frm.lblStudentNameChangeStudentDetailsUpdate.Text = datagridStudentDetails.CurrentRow.Cells[1].Value.ToString();

frm.lblPhoneNoChangeStudentDetailsUpdate.Text = datagridStudentDetails.CurrentRow.Cells[2].Value.ToString();

frm.lblAddressChangeStudentDetailsUpdate.Text = datagridStudentDetails.CurrentRow.Cells[3].Value.ToString();

frm.lblEmailChangeStudentDetailsUpdate.Text = datagridStudentDetails.CurrentRow.Cells[4].Value.ToString();

frm.Show();

this.Hide();

}

private void btnSearch\_Click(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_StudentDetail where Name=@a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text);

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

if (ds.Tables["t"].Rows.Count > 0)

{

datagridStudentDetails.DataSource = ds.Tables["t"];

}

else

{

MessageBox.Show("Student name Does not exsists");

}

}

private void txtSearch\_TextChanged(object sender, EventArgs e)

{

SqlConnection con = new SqlConnection(connectionStr);

string sql = "select \* from Table\_StudentDetail where Name like @a";

SqlCommand cmd = new SqlCommand(sql, con);

cmd.Parameters.AddWithValue("@a", txtSearch.Text + "%");

SqlDataAdapter da = new SqlDataAdapter(cmd);

DataSet ds = new DataSet();

da.Fill(ds, "t");

datagridStudentDetails.DataSource = ds.Tables["t"];

}

* **Book Issued Report**

private void btnSearch\_Click(object sender, EventArgs e)

{

CReportIssue crpt = new CReportIssue();

ParameterFieldDefinitions crParameterFieldDefinitions;

ParameterFieldDefinition crParameterFieldDefinition;

ParameterValues crParameterValues = new ParameterValues();

ParameterDiscreteValue crParameterDiscreteValue = new ParameterDiscreteValue();

//for parameter fromdate

crParameterDiscreteValue.Value = dateTimePickerFrom.Text;

crParameterFieldDefinitions = crpt.DataDefinition.ParameterFields;

crParameterFieldDefinition = crParameterFieldDefinitions["fromdate"];

crParameterValues = crParameterFieldDefinition.CurrentValues;

crParameterValues.Clear();

crParameterValues.Add(crParameterDiscreteValue);

crParameterFieldDefinition.ApplyCurrentValues(crParameterValues);

//for parameter todate

crParameterDiscreteValue.Value = dateTimePickerTo.Text;

crParameterFieldDefinitions = crpt.DataDefinition.ParameterFields;

crParameterFieldDefinition = crParameterFieldDefinitions["todate"];

crParameterValues = crParameterFieldDefinition.CurrentValues;

crParameterValues.Add(crParameterDiscreteValue);

crParameterFieldDefinition.ApplyCurrentValues(crParameterValues);

crystalReportViewer1.ReportSource = crpt;

crystalReportViewer1.Refresh();

}

* **Fine Report**

private void btnSearch\_Click(object sender, EventArgs e)

{

CrystalReport1 crpt = new CrystalReport1();

ParameterFieldDefinitions crParameterFieldDefinitions;

ParameterFieldDefinition crParameterFieldDefinition;

ParameterValues crParameterValues = new ParameterValues();

ParameterDiscreteValue crParameterDiscreteValue = new ParameterDiscreteValue();

//for parameter fromdate

crParameterDiscreteValue.Value = dateTimePicker1.Text;

crParameterFieldDefinitions = crpt.DataDefinition.ParameterFields;

crParameterFieldDefinition = crParameterFieldDefinitions["FromDate"];

crParameterValues = crParameterFieldDefinition.CurrentValues;

crParameterValues.Clear();

crParameterValues.Add(crParameterDiscreteValue);

crParameterFieldDefinition.ApplyCurrentValues(crParameterValues);

//for parameter todate

crParameterDiscreteValue.Value = dateTimePicker2.Text;

crParameterFieldDefinitions = crpt.DataDefinition.ParameterFields;

crParameterFieldDefinition = crParameterFieldDefinitions["ToDate"];

crParameterValues = crParameterFieldDefinition.CurrentValues;

crParameterValues.Add(crParameterDiscreteValue);

crParameterFieldDefinition.ApplyCurrentValues(crParameterValues);

crystalReportViewer1.ReportSource = crpt;

crystalReportViewer1.Refresh();

}

* **Admin Form**

private void MDIAdmin\_Load(object sender, EventArgs e)

{

labelDate.Text = DateTime.Today.ToLongDateString();

}

private void aToolStripMenuItem\_Click(object sender, EventArgs e)

{

if(ActiveMdiChild !=null)

{

ActiveMdiChild.Close();

}

FormEntryBook frm = new FormEntryBook();

frm.MdiParent = this;

frm.Show();

}

private void viewToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewBooks frm = new FormViewBooks();

frm.MdiParent = this;

frm.Show();

}

private void addStudentDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormStudentDetails frm = new FormStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void addUserToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormAddUser frm = new FormAddUser();

frm.MdiParent = this;

frm.Show();

}

private void issuesToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewIssues frm = new FormViewIssues();

frm.MdiParent = this;

frm.Show();

}

private void changeDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeUserDetails frm = new FormDetailsChangeUserDetails();

frm.MdiParent = this;

frm.Show();

}

private void changeBookDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeBookDetails frm = new FormDetailsChangeBookDetails();

frm.MdiParent = this;

frm.Show();

}

private void changeStudentDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeStudentDetails frm = new FormDetailsChangeStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void logoutToolStripMenuItem\_Click(object sender, EventArgs e)

{

FormLogin frm = new FormLogin();

frm.Show();

this.Close();

}

private void viewReceivedToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewReceived frm = new FormViewReceived();

frm.MdiParent = this;

frm.Show();

}

private void selfDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormIndividualUserDetailsUpdate frm = new FormIndividualUserDetailsUpdate();

frm.MdiParent = this;

frm.Show();

}

private void issuedReportToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

ReportIssued frm = new ReportIssued();

frm.MdiParent = this;

frm.Show();

}

private void viewFineToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewFine frm = new FormViewFine();

frm.MdiParent = this;

frm.Show();

}

private void aboutUsToolStripMenuItem1\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormHelpAboutUs frm = new FormHelpAboutUs();

frm.MdiParent = this;

frm.Show();

}

private void guideToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormHelpGuide frm = new FormHelpGuide();

frm.MdiParent = this;

frm.Show();

}

private void btnLogout\_Click(object sender, EventArgs e)

{

FormLogin frm = new FormLogin();

frm.Show();

this.Close();

}

private void btnBooks\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewBooks frm = new FormViewBooks();

frm.MdiParent = this;

frm.Show();

}

private void btnIssues\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewIssues frm = new FormViewIssues();

frm.MdiParent = this;

frm.Show();

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewReceived frm = new FormViewReceived();

frm.MdiParent = this;

frm.Show();

}

private void btnFine\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewFine frm = new FormViewFine();

frm.MdiParent = this;

frm.Show();

}

private void toolStripButton2\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeUserDetails frm = new FormDetailsChangeUserDetails();

frm.MdiParent = this;

frm.Show();

}

private void toolStripButton3\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeStudentDetails frm = new FormDetailsChangeStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void btnIssuedReport\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

ReportIssued frm = new ReportIssued();

frm.MdiParent = this;

frm.Show();

}

private void btnAddBooks\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormEntryBook frm = new FormEntryBook();

frm.MdiParent = this;

frm.Show();

}

private void AddUsers\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormAddUser frm = new FormAddUser();

frm.MdiParent = this;

frm.Show();

}

private void AddStudent\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormStudentDetails frm = new FormStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void fineReportToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormReportFine frm = new FormReportFine();

frm.MdiParent = this;

frm.Show();

}

private void fineReportToolStripMenuItem1\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormReportFine frm = new FormReportFine();

frm.MdiParent = this;

frm.Show();

}

* **User Form**

private void viewBooksToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewBooks frm = new FormViewBooks();

frm.MdiParent = this;

frm.Show();

}

private void viewIssuesToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewIssues frm = new FormViewIssues();

frm.MdiParent = this;

frm.Show();

}

private void viewReceivedToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewReceived frm = new FormViewReceived();

frm.MdiParent = this;

frm.Show();

}

private void viewFineToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewFine frm = new FormViewFine();

frm.MdiParent = this;

frm.Show();

}

private void addBookToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormEntryBook frm = new FormEntryBook();

frm.MdiParent = this;

frm.Show();

}

private void addStudentDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormStudentDetails frm = new FormStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void changeBookDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeBookDetails frm = new FormDetailsChangeBookDetails();

frm.MdiParent = this;

frm.Show();

}

private void changeStudentDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeStudentDetails frm = new FormDetailsChangeStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void changeSelfDetailsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormIndividualUserDetailsUpdate frm = new FormIndividualUserDetailsUpdate();

frm.MdiParent = this;

frm.Show();

}

private void issuedReportToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

ReportIssued frm = new ReportIssued();

frm.MdiParent = this;

frm.Show();

}

private void aboutUsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormHelpAboutUs frm = new FormHelpAboutUs();

frm.MdiParent = this;

frm.Show();

}

private void userManualToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormHelpGuide frm = new FormHelpGuide();

frm.MdiParent = this;

frm.Show();

}

private void logoutToolStripMenuItem\_Click(object sender, EventArgs e)

{

FormLogin frm = new FormLogin();

frm.Show();

this.Close();

}

private void BtnBook\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewBooks frm = new FormViewBooks();

frm.MdiParent = this;

frm.Show();

}

private void btnIssues\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewIssues frm = new FormViewIssues();

frm.MdiParent = this;

frm.Show();

}

private void btnReceived\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewReceived frm = new FormViewReceived();

frm.MdiParent = this;

frm.Show();

}

private void btnFine\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormViewFine frm = new FormViewFine();

frm.MdiParent = this;

frm.Show();

}

private void btnAddBooks\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormEntryBook frm = new FormEntryBook();

frm.MdiParent = this;

frm.Show();

}

private void toolStripButton3\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormStudentDetails frm = new FormStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void btnStudent\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormDetailsChangeStudentDetails frm = new FormDetailsChangeStudentDetails();

frm.MdiParent = this;

frm.Show();

}

private void btnSelf\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormIndividualUserDetailsUpdate frm = new FormIndividualUserDetailsUpdate();

frm.MdiParent = this;

frm.Show();

}

private void btnIssuedReport\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

ReportIssued frm = new ReportIssued();

frm.MdiParent = this;

frm.Show();

}

private void btnLogout\_Click(object sender, EventArgs e)

{

FormLogin frm = new FormLogin();

frm.Show();

this.Close();

}

private void MDIUser\_Load(object sender, EventArgs e)

{

labelDate.Text = DateTime.Today.ToLongDateString();

}

private void fineReportToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormReportFine frm = new FormReportFine();

frm.MdiParent = this;

frm.Show();

}

private void fineReportToolStripMenuItem1\_Click(object sender, EventArgs e)

{

if (ActiveMdiChild != null)

{

ActiveMdiChild.Close();

}

FormReportFine frm = new FormReportFine();

frm.MdiParent = this;

frm.Show();

}

## 

## 11. USER OPERATIONS AND MANUAL

User operational manual includes the handling of my project on various regards. Some of them are as below:-

### 11.1 Support and Maintenance

This software will undoubtedly undergo change after it is delivered to the customer. Change will occur because errors have been encountered, because the software must be adapted to accommodate changes in its external environments. This software can also undergo change if the admin or user wants to add new features based on functional or performance enhancement. This software maintenance phase may reply each of the preceding software engineering life cycle steps to existing program rather than a new one.

Though maintenance is not considered a part of software development, it is a very important activity in the life of a software product. Usually, the total it is very important activity in the life of a software product. Usually, the total cost of maintenance phase is much higher than the development cost of software. Normally the maintenance to developed cost ratio is suggested as 80:20, 70:30 and 60:40

Maintenance work is based on existing software, as compared to development work that creates new software. In other words, maintenance revolves around understating the existing software and maintenance spend most of their time trying to understand the software they have to modify. Understanding the software means that understanding not only the code but also the related documents .During the modification of the software, the effect the change has to be clearly understood by the maintainer. To test whether those aspects of the system that are not supposed to be modified are operation as they were before modification, regression testing is done. In regression testing, we use old test case to test whether new errors have been introduced or not.

Maintenance involves understanding the existing software, understanding the effect of changes to both code and document, testing the new parts and retesting the old parts that were not changed. In order to make the maintainer job easier, it is necessary to prepare some supporting documents during software development. The complexity of the maintenance task, coupled with the neglect of maintenance concerns during development, makes maintenance the most costly activity in the life of software product.

### 11.2Future Scope

This project can be used in the Book after adding some more useful modules in the project for which Book are providing services.

Utmost care and back-up procedures must be established to ensure 100% successful implementation of the computerized Book system. In case of system failure, the organization should be in a position to process the transaction with another organization or if the worst comes to the worst, it should be in a position to complete it manually.

### 11.3 Backup and Recovery

Backup and recovery of data is most essential part of any software development. There are many circumstances which may leads to manipulation or loss of whole data such as virus attack, failure of storage device, etc. In such a case it is most important to recover data for which backup of data need to be done.

Using this software, user can backup data into hard-drive or other storage device like pen-drive.. Using the backup file, we can restore it at any time by providing the location of backup file.

### 

### 11.4 Security

Current practices for developing secure systems are still closer to art than to an engineering discipline. Security is still treated as an add-on and is therefore not integrated into software development practices and tools. Experienced security artisans are still the key to achieving acceptable levels of security.  
Regarding to this software, it is secured on the basis that no other people can access the software as it provides the log in feature.

**a)Operating system level security**

Use of username and password protect the data of Books to be viewed by other people which are highly used for security. Therefore as I have built this software in the windows 8.1 platform using C#.NET, this software bears all the security that this operating system and Dot NET Framework has.

**b) Database security**

Database has it has own username and password which save the data of database. SQLSERVER 2008 R2 has its own security level which prevent from illegal copying of data.

**c) Application security**

In addition to the operating level security this software has its own application securities too.

**d)Program strength and weakness**

The program has prepared according to the detail study of system so it provides a robust operation. It uses windows interface and can organize its entire program with within MDI parent windows.

# 

# 12. Conclusion and Recommendation

**Conclusion**

My application “Book Management System “is responsible for the creating and controlling students and providing the authority for them and making registration process easier and faster and more convenient for book management activities.

Book Management System is desktop based small computer application. It is focused for areas like institutes which offer book management activities and save their records for future reference.

**Recommendation**

The materials presented in this report can be extended to access the applicability of an application. Based on the achieved results the following areas for further study are recommended.

Through the entire project, it is recommended to issue the research on the different institutes for the different types of registration process which are widely used in institutes and educational sectors.

# 

**13. Bibliography**

# 

* [http://www.C#.NETguicodexample.com/C#.NETdesktopguiSQLSERVER 2008 R21.html](http://www.javaguicodexample.com/javadesktopguimysql1.html)
* <https://www.youtube.com/playlist?list=PLB04B4E5D9B58C13D>
* <https://www.youtube.com/watch?v=ppGttvNUD90>
* [http://www.onC#.NET.com/pub/a/onC#.NET/2003/09/17/C#.NET.html](http://www.onjava.com/pub/a/onjava/2003/09/17/macosxjava.html)
* [https://www.C#.NET.net/forum/topic/C#.NET-tools/C#.NET-development-tools/how-develop-C#.NET-desktop-application-0](https://www.java.net/forum/topic/java-tools/java-development-tools/how-develop-java-desktop-application-0)
* [http://www.C#.NETdesktop.org/articles/jdic/index.html](http://www.javadesktop.org/articles/jdic/index.html)

**ONLINE RESOURCES**

* www.youtube.com/user/ProgrammingKnowledge
* www.tutorialspoint.com/C#.NET/
* www.C#.NETbeginner.com/