

SILIGURI INSTITUTE OF TECHNOLOGY

IT 892

PLACEMENT CELL MANAGEMENT SYSTEM

IT-PROJECT-2021-6

Name of Students	Roll No.
1. SOUBHIK DUTTA	11900217003
2. PAYEL DAS	11900217010
3. ARPAN BHAKTA	11900217027
4. ADARSH RAI	11900217030

Under the Guidance

of

Prof. MR ASIT BARMAN

Submitted to the Department of **Information Technology** in partial fulfilment of the requirements for the award of the degree Bachelor of Technology in **Information Technology**.

Year of Submission: 2021



Siliguri Institute of Technology

DECLARATION

■ This is to certify that Report entitled “**PLACEMENT CELL MANAGEMENT SYSTEM**” which is submitted by me in partial fulfilment of the requirement for the award of degree B.Tech. in **Information Technology** at **Siliguri Institute of Technology** under **Maulana Abul Kalam Azad University of Technology**, West Bengal. We took the help of other materials in our dissertation which have been properly acknowledged. This report has not been submitted to any other Institute for the award of any other degree.

Date:

SN	Name of the Student	Roll No	Signature
1	SOUBHIK DUTTA	11900217003	
2	PAYEL DAS	11900217010	
3	ARPAN BHAKTA	11900217027	
4	ADARSH RAI	11900217030	

CERTIFICATE

This is to certify that the project report entitled PLACEMENT CELL MANAGEMENT SYSTEM submitted to **the Department of Information Technology of Siliguri Institute of Technology** in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in Information Technology** during the academic year **2019-20**, is a bonafide record of the project work carried out by them under my guidance and supervision.

Project Group Number: 6			
SN	Name of the students	Registration No	Roll No
1.	SOUBHIK DUTTA		11900217003
2.	PAYEL DAS		11900217010
3.	ARPAN BHAKTA		11900217027
4.	ADARSH RAI		11900217030

Signature of Project Guide

Name of the Guide:MR.ASIT BARMAN

Department

Acknowledgement

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful.

We take this opportunity to express our deep gratitude towards our project mentor, **Mr Asit Barman** for giving such valuable suggestions, guidance and encouragement during the development of this project work.

Signature of all the group members with date

- 1.
- 2.
- 3.
- 4.

TABLE OF CONTENTS

CHAPTER 1 : INTRODUCTION	1
1.1 LITERATURE REVIEW	7
1.2 OBJECTIVE	7
1.3 SCOPE	8
CHAPTER 2 : SYSTEM ANALYSIS	10
2.1 IDENTIFICATION OF NEED	11
2.2 FEASIBILITY STUDY	12
2.3 PROJECT PLANNING	13
2.4 PROJECT SCHEDULING	15
2.5 SOFTWARE REQUIREMENTS	16
CHAPTER 3: PLACEMENT CELL DESIGN	17
3.1 DATA FLOW DIAGRAM	19
3.3 DATABASE DESIGN	22
3.4 USER INTERFACE DESIGN	25
CHAPTER 4 : CODING	31
4.1SNIPPETS OF CODE	32
CHAPTER 5 : TESTING	38
5.1 WHITE BOX TESTING	39
5.2 BLACK BOX TESTING	42
5.3 MANDATORY TESTING METHODS	46
CHAPTER 6 : CONCLUSION AND RECOMMENDATION	50
CHAPTER 7: REFERENCING AND APPENDICES	53

ABSTRACT

Placement cell management system is a useful system that will help students in interacting with recruiting organizations. This project is made for ease in the process of placement.

In this project, the system will maintain the records of the student such as name, branch, contact no, email id, enrollment year etc. All the details have to be given by the student while registering. The admin from the college has to log in and verify the details given by the student while registering in the portal.

The whole system would be online so that the process would make it easier for the students, college officials and the recruiting organisations. At first, when the portal opens the following process could be made:

- Register
- Login
- Help
- Reports
- News and events(upcoming companies for the drive).

This project will also give the statistics of the student as well as the companies visiting for the drives which can be fetched by the placement officer of the college and can visualize the placed students so that the student can be barred from sitting in the upcoming drives. The students can also see the companies coming to the college for the placement drives. The companies can take the database from the portal and can mail them for further rounds.

CHAPTER – 1

INTRODUCTION

Placement cell management system is a very useful open-source application software for providing user flexibility to perform various actions that are usually used by a placement officer, recruiting organizations and automating the operations performed over the information. It reduces the workload of management of the students as well as companies.

1.1 OBJECTIVE

The project's aims and objectives that will be achieved after the completion of the system was carried out in this subchapter. The succession of the system also will be evaluated through this subchapter. The project objectives are:

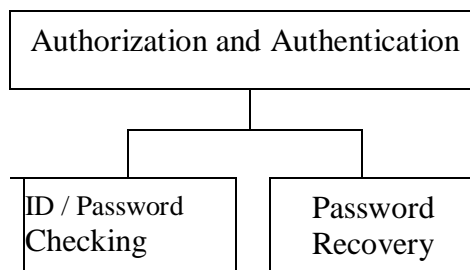
- To provide open-source application software.
- To have a record of every student in the portal.
- To have a record of every organization visiting the college
- To design a user-friendly graphical user interface.
- To complete the system according to the project schedule.
- To produce a technical report that documents the phases, tasks and deliverables in the project.
- To ease the process of the student, recruiting organization and the placement officer.

The project will also help the students to get regular updates about the upcoming drives in the college.

1.2 SCOPE

In this subchapter, the project scope will carry out what modules were contained inside the software. it will describe the part which is the placement cell used by student/employee /TPO.

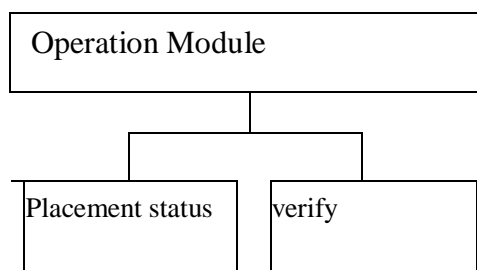
1.2.1 Authorization and Authentication Module:



This module is used by student/company/placement officers.

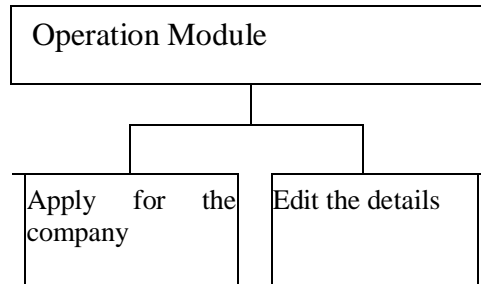
They have to log in using this module or change the password using this module if they don't know or have forgotten the password.

1.2.2: Operation Module(placement officer(TPO)):



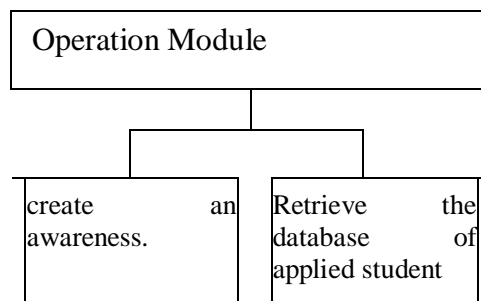
In this module, the placement officer can verify the student or check the placement status/report of the student.

1.2.3: Operation Module(student):



In this module, students can apply for the company or edit the details.

1.2.3: Operation Module(employee):



In this module, the recruiter will create awareness about the upcoming recruitment drive or retrieve the database of the applied student.

There is a **HELP** bar at last which have some frequently asked questions(faq).it will help the users to get a good idea of the questions they have.

CHAPTER – 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF THE NEED

2.1.1 EXISTING SYSTEM: There is no consolidated application software for placement in our college. The current database in our college is retrieved using google forms for an upcoming placement drive. Whenever there is a drive the data is being taken using google forms. The placed students don't get barred from filling up the upcoming drive forms if they get placed into 2 companies.

2.1.2 PROPOSED SYSTEM: First and foremost this project will give a consolidated application software for placement activities. It will help recruiters to fetch data from the software and ease the process of placement drive.it will help the students to get timely updates about the upcoming drives.it has a user-friendly interface which will help recruiters/tpo/student to navigate through.

2.2 FEASIBILITY STUDY

It involves the analysis of the problem & collection of all relevant information relating to the product such as

items that would be input to the system, processing required to carry those data, the output data required to be produced by the system as well the various constraints on the behaviour of the system. "application based placement management system" has undergone the feasibility study so that the proposed system is possible for development deployment in our college. The feasibility study concentrates on the following, such as Operational Feasibility, Technical Feasibility, Economic Feasibility.

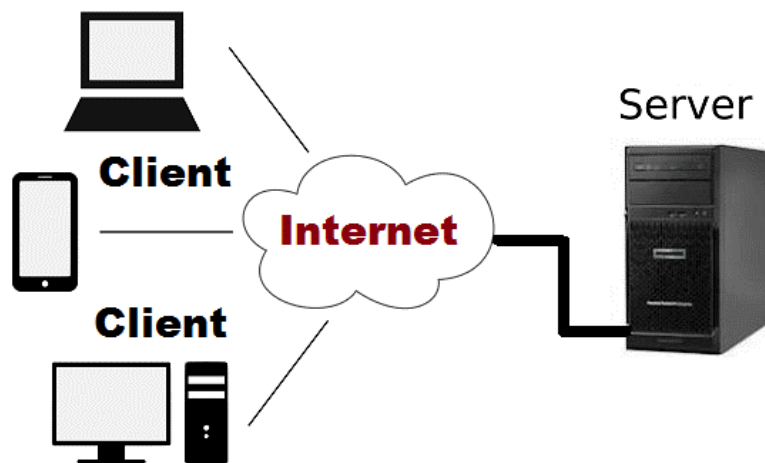
2.3 PROJECT PLANNING

Firstly, we collected the requirement for the Placement cell management system, and then we analyzed it. After analyzing the requirements, we proceeded to the design stage. In the design phase, the user interface (UI) and the database design is designed. The next phase in the waterfall model is the Construction phase. It is an important phase in the waterfall model and it is a time-consuming phase depending on the programmer's ability. In placement Management System development, the construction phase is using Eclipse IDE to write the program. After the coding phase, we proceed to the integration phase. In this phase, the software shares a database to make the integration between two applications. The next phase after integration is the testing and debugging phase. The testing module is separated into module testing, system testing, unit testing and user acceptance test. Once there is a bug found, it will be solved immediately before the system is launched to ensure the system launched is bug-free.

Lastly, it is the installation and maintenance phase where the system will be installed at the user side. After installing the system, maintenance is required to ensure the system is always on and up to date with the latest technologies or latest business processes.

2.3.1. SYSTEM ARCHITECTURE

The application-based Management system uses client/server engineering. It is an online web portal designed using JAVA and SQL.

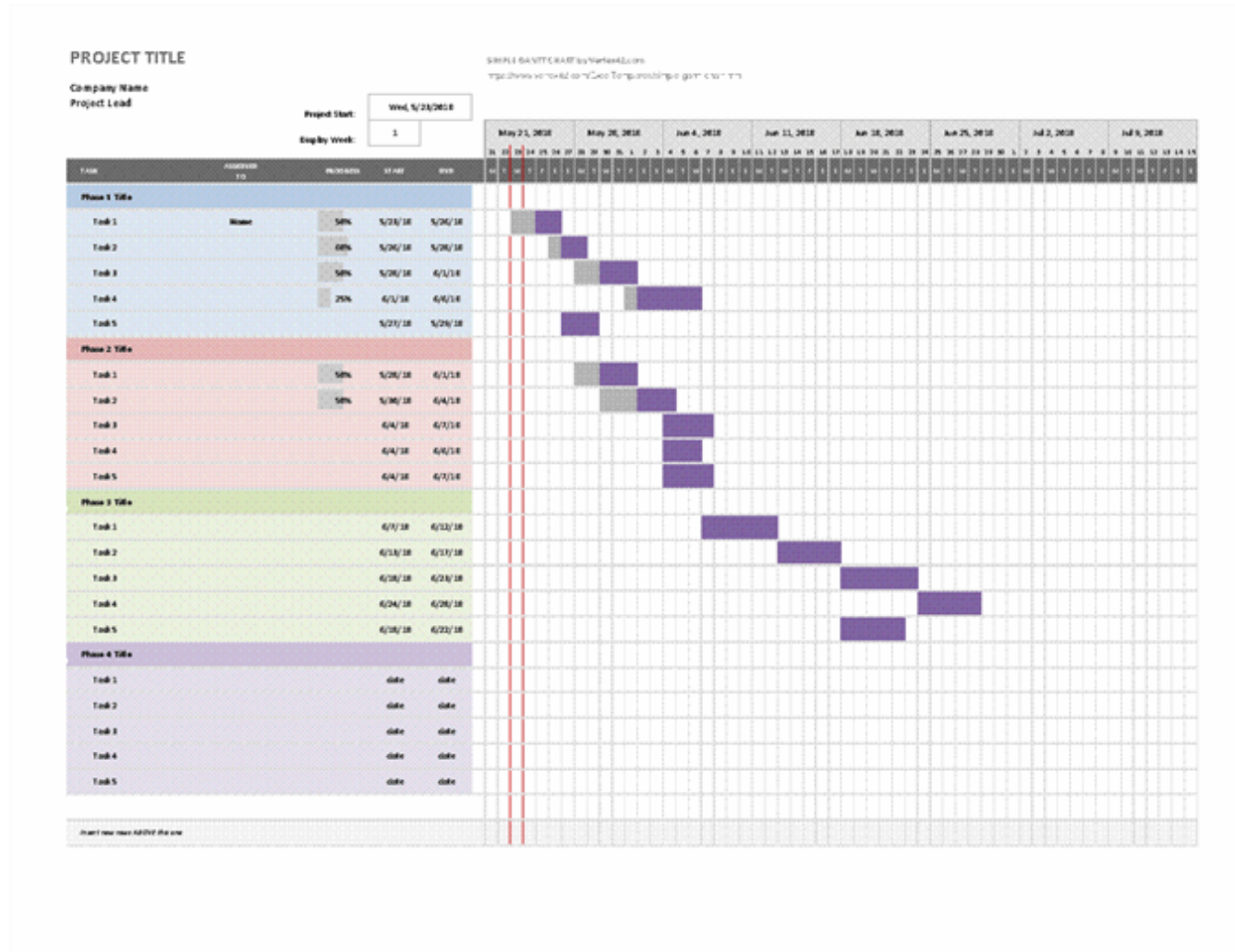


2.3.2 Database outline:

The database connected to this application is MySQL which is worked within the tomcat server or free host server. Three tables are made which are user record, placement record, student record.

2.4 SCHEDULING

PROJECT



2.5 SOFTWARE REQUIREMENTS SPECIFICATIONS(SRS)

2.5.1. Hardware Requirements:

- Processor: Intel i3 5th Gen Processor 2.00 GHz is used, where we can keep developing the Placement cell Management System without the need to worry that the pc cannot support it.
- Ram: 4 Gb is used to support Eclipse IDE and MySQL and to avoid any problem that may arise during the development phase.

2.5.2. Software Requirements:

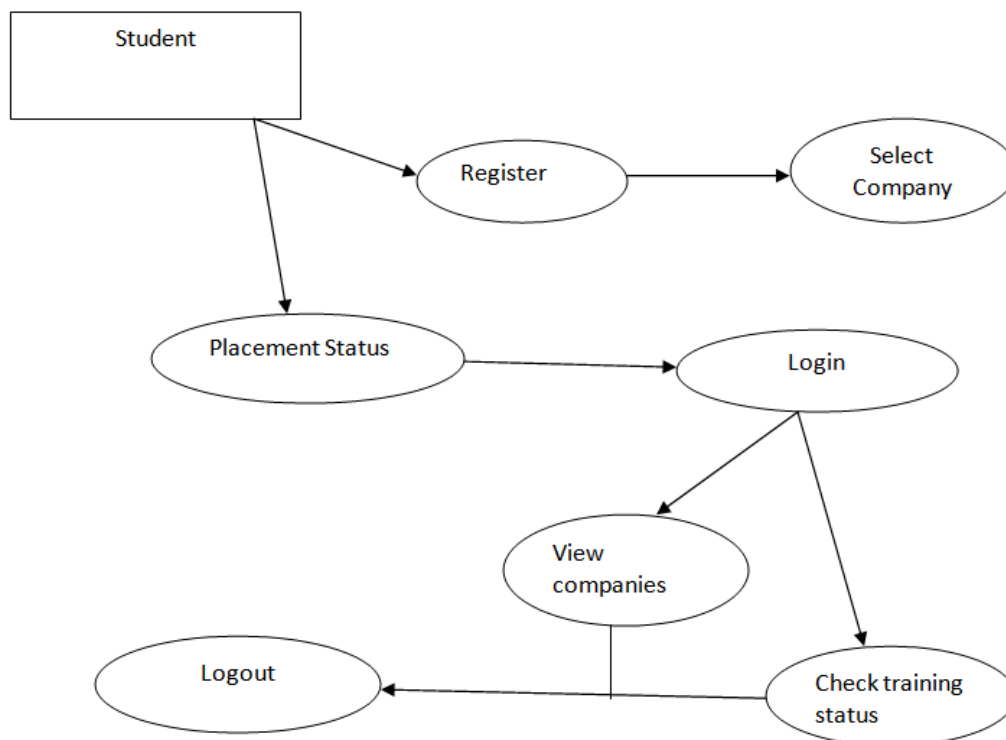
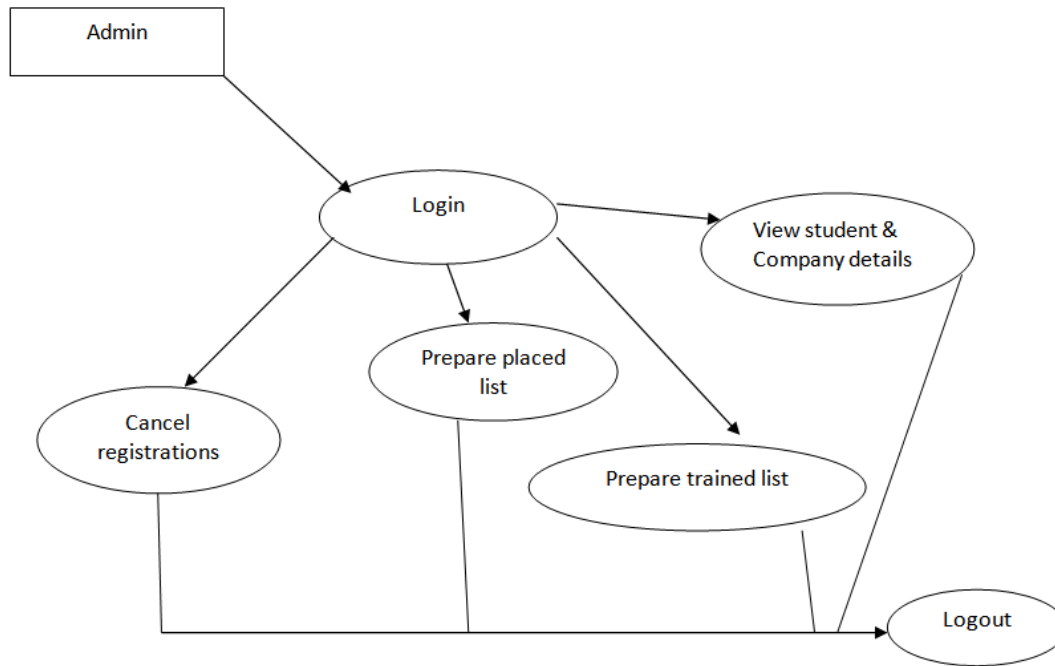
- Operating system - Windows 10 is used as the operating system as it is the latest stable build of Microsoft;
- IDE- Eclipse IDE is used
- XAMPP

CHAPTER - 3

**SYSTEM DESIGN
DOCUMENT (SDD)**

The purpose of this section is mainly to discuss the system design aspect of the placement cell management system. Design goals will be provided in the introduction of the document to identify the qualities that our system will focus on. An overview of the current system architecture will be included for comparison with the proposed system architecture. Besides that, the proposed system architecture, its subsystem decomposition, hardware and software mapping, persistent data management, access control and security, global software control, and subsystem services will also be included in this document. Ultimately, the goal of this system design document is to provide design specifications of the placement cell management system to facilitate our project implementation process.

3.1 DATA-FLOW DIAGRAM(DFD)



The DFD of our placement cell Management System

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and non-technical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.

There are four basic symbols to depict the processes in a DFD.



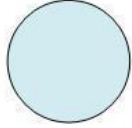





- 1. External entity:** An outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.

- 2. Process:** Any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the

data flow based on business rules. A short label is used to describe the process, such as “Submit payment.”

3. Datastore: Files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as “Orders.”

4. Data flow: The route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labelled with a short data-name, like “Billing details.”

Notation	De Marco & Yourdon	Gane and Sarson
External Entity		
Process		
Data Store		
Data Flow		

3.2 DATABASE DESIGN

For persistent data management, a relational database created by MySQL will be used as the database management system (DBMS) of the Placement cell management system(PCMS). This database is used to store all data related to the PCMS, including student information, course scheduling data. PCMS shall implement some authentication mechanism for access to the database. The following figure shows the relational database schema of the PCMS.

Exit Survey

ExitSurveys

- id INT
- surveyDate DATETIME
- studentName VARCHAR(45)
- sect VARCHAR(15)
- degreeProgram VARCHAR(45)
- termGradDateSemester VARCHAR(15)
- termGradDateYear VARCHAR(5)
- contact1Name VARCHAR(45)
- contact1PhoneHome VARCHAR(20)
- contact1PhoneWork VARCHAR(20)
- contact1PhoneCell VARCHAR(20)
- contact1Address VARCHAR(100)
- contact1Email VARCHAR(45)
- contact1OtherOption VARCHAR(20)
- contact1Name VARCHAR(45)
- contact2PhoneHome VARCHAR(20)
- contact2PhoneWork VARCHAR(20)
- contact2PhoneCell VARCHAR(20)
- contact2Address VARCHAR(100)
- contact2Email VARCHAR(45)
- assessQ1 VARCHAR(45)
- assessQ2 VARCHAR(45)
- assessQ3 VARCHAR(45)
- assessComment VARCHAR(200)
- furtherStudySchool VARCHAR(45)
- furtherStudyMajor VARCHAR(45)
- furtherStudyScholarship VARCHAR(45)
- jobSearchDuration VARCHAR(10)
- jobSearchNumInterview VARCHAR(10)
- jobSearchNumOffer VARCHAR(10)
- jobSearchAvgSalary VARCHAR(10)
- jobCompany VARCHAR(100)
- jobCity VARCHAR(45)
- jobTitle VARCHAR(45)
- jobCompanyContact VARCHAR(45)
- jobCompanyWeb VARCHAR(100)
- jobSalary VARCHAR(10)
- networkingQ1 VARCHAR(45)
- networkingQ2 VARCHAR(45)

GradSeniorSurveys

- id INT
- surveyDate DATETIME
- degreeProgram VARCHAR(45)
- termgraduateSemester VARCHAR(15)
- termgraduateYear VARCHAR(5)
- Obj1 VARCHAR(45)
- Obj2 VARCHAR(45)
- Obj3 VARCHAR(45)
- Obj4 VARCHAR(45)
- Obj5 VARCHAR(45)
- Outcome1 VARCHAR(45)
- Outcome2 VARCHAR(45)
- Outcome3 VARCHAR(45)
- Outcome4 VARCHAR(45)
- Outcome5 VARCHAR(45)
- Outcome6 VARCHAR(45)
- Outcome7 VARCHAR(45)
- Outcome8 VARCHAR(45)
- Outcome9 VARCHAR(45)
- Outcome10 VARCHAR(45)

Course Schedule

Courses

- id INT
- courseTitle VARCHAR(100)
- courseNum VARCHAR(10)
- section INT
- scheduleStartTime DATETIME
- scheduleEndTime DATETIME
- instructor VARCHAR(45)
- room VARCHAR(20)
- credHours INT
- cm VARCHAR(10)
- MaxStudent INT
- notes VARCHAR(200)
- weekday VARCHAR(15)
- scheduleType INT
- semesterId INT

Semesters

- id INT
- SemesterTime VARCHAR(10)
- From DATETIME
- To DATETIME

MODULARIZATION

- **We have used waterfall modularization**

The Waterfall approach was the first SDLC Model to be used widely in Software Engineering to ensure the success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.

The following illustration is a representation of the phases of the Waterfall Model.

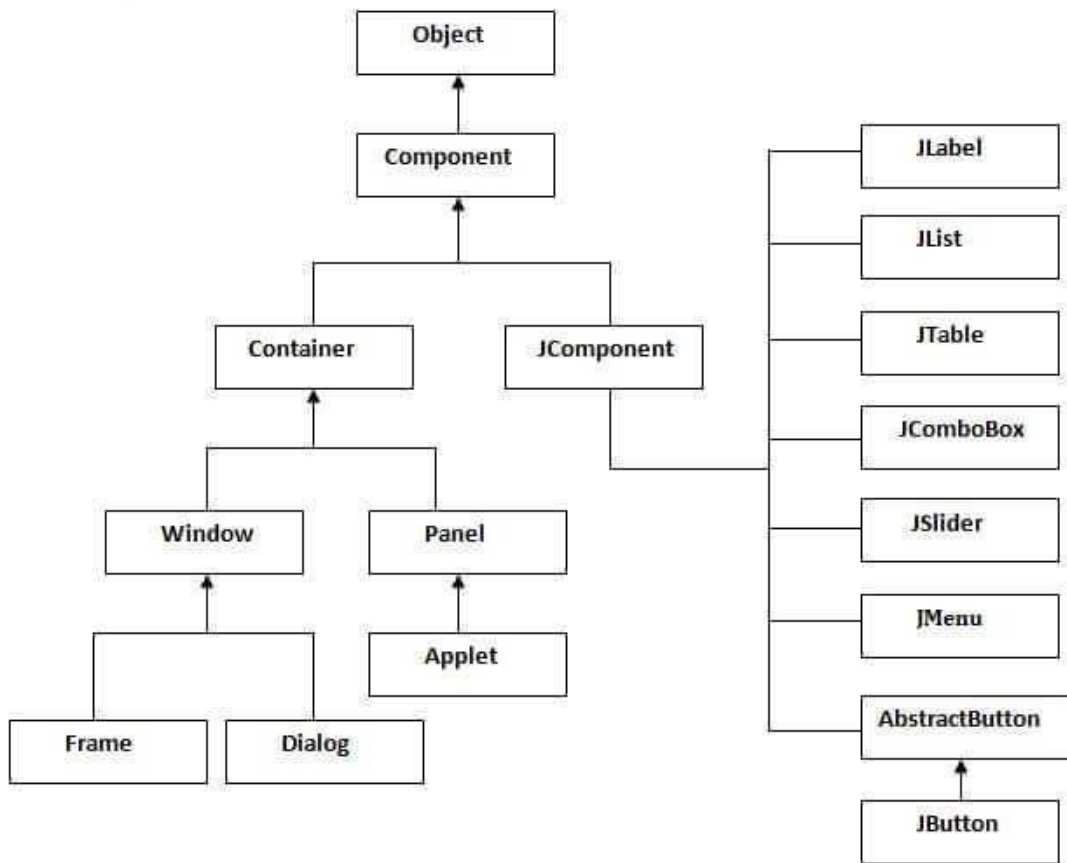
The sequential phases in the Waterfall model are –

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – The requirement specifications from the first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – With inputs from the 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 is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing each unit. Post integration the entire system is tested for any faults and failures.
- **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.
- **Maintenance** – There are some issues that come up 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.

3.3 USER INTERFACE DESIGN

We used Java AWT and Java Swing for designing the GUI.

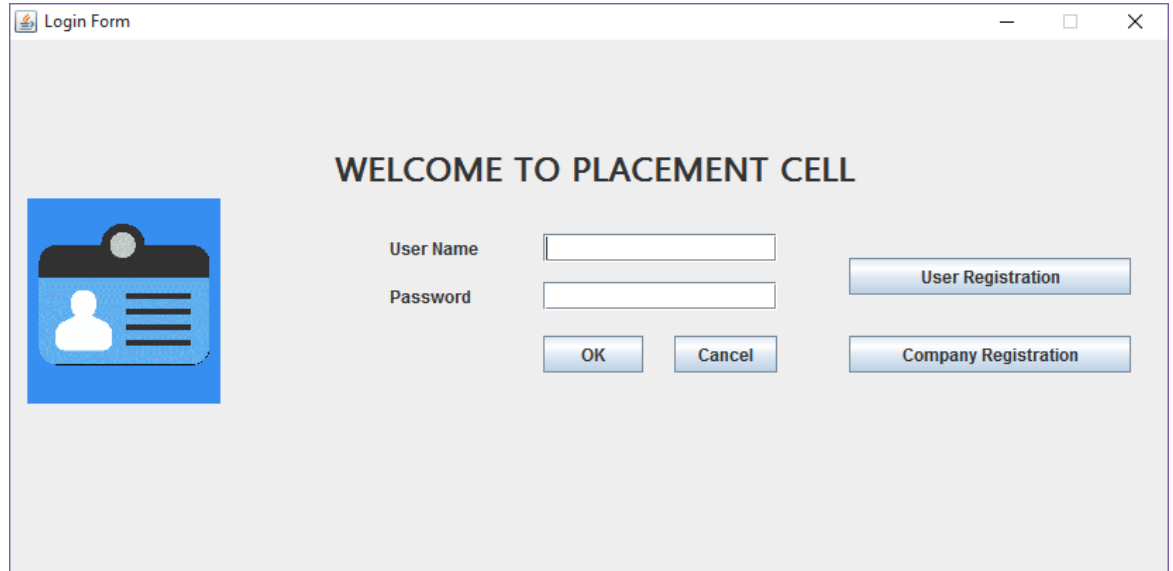


The Hierarchy of Swing API

We used the concept of the grid layout present in swing in order to design the required pages in our project. This is basically used to arrange all the components in a rectangular grid.

Constructors of GridLayout class

1. **GridLayout()**: creates a grid layout with one column per component in a row.
2. **GridLayout(int rows, int columns)**: creates a grid layout with the given rows and columns but no gaps between the components.
3. **GridLayout(int rows, int columns, int hgap, int vgap)**: creates a grid layout with the given rows and columns along with given horizontal and vertical gaps.

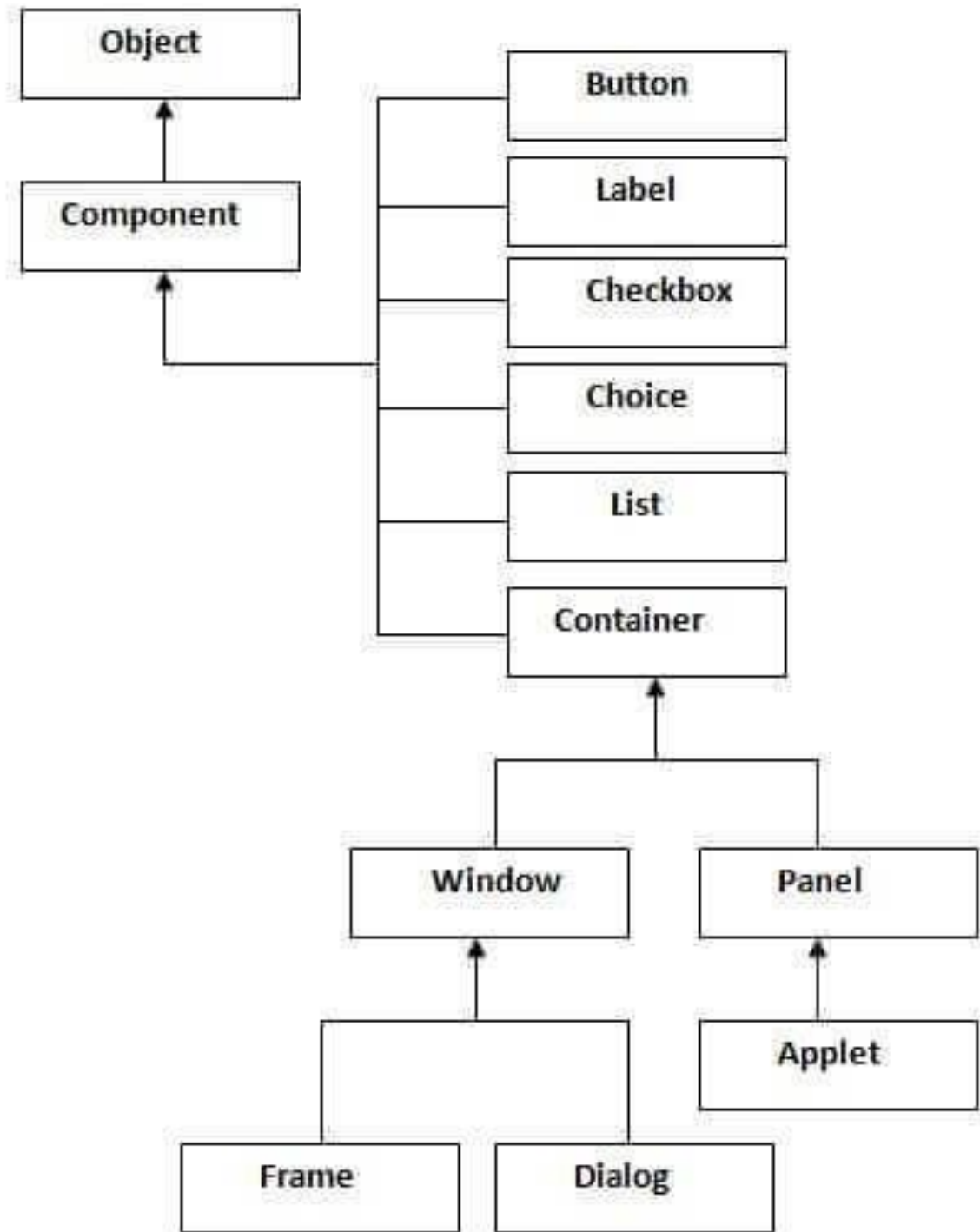


The login form was designed using the grid layout

Java AWT (Abstract Window Toolkit) is *an API to develop GUI or window-based applications* in java.

Java AWT components are platform-dependent i.e. components are displayed according to the view of the operating system. AWT is heavyweight i.e. its components are using the resources of the OS.

The java.awt package provides classes for AWT api such as TextField, Label, TextArea, RadioButton, CheckBox, Choice, List etc.



The AWT Hierarchy

Container

The Container is a component in AWT that can contain other components like buttons, text fields, labels etc. The classes that extend the Container class are known as containers such as Frame, Dialog and Panel.

Window

The window is the container that has no borders and menu bars. You must use a frame, dialog or another window for creating a window.

Panel

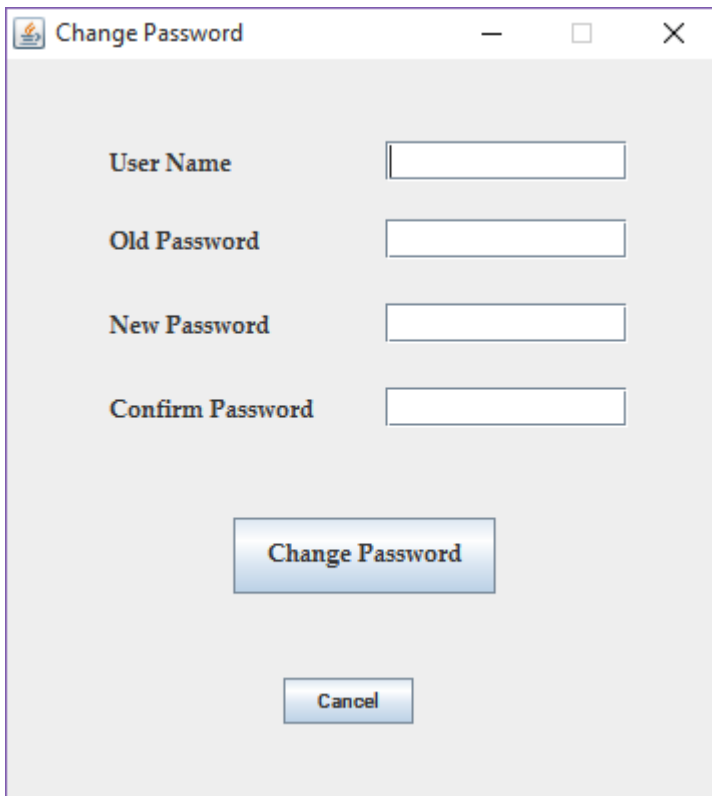
The Panel is the container that doesn't contain title bar and menu bars. It can have other components like buttons, textfield etc.

Frame

The Frame is the container that contains the title bar and can have menu bars. It can have other components like buttons, textfield etc.



The 'User Registration' window features a title bar with a standard icon, a minus sign, a maximize button, and a close button. The main content area is divided into two sections. On the left, under the heading 'User Details', there are five text input fields labeled 'Name', 'User Name', 'Password', 'Email ID', and 'Contact No.'. On the right, there is a vertical stack of five buttons: 'New', 'Save', 'Delete', 'Update', and 'Get Data'. The 'New', 'Save', and 'Get Data' buttons are highlighted with a blue gradient, while 'Delete' and 'Update' are in a lighter grey.



The 'Change Password' window has a title bar with a standard icon, a minus sign, a maximize button, and a close button. The main content area contains four text input fields labeled 'User Name', 'Old Password', 'New Password', and 'Confirm Password'. At the bottom of the window, there are two buttons: 'Change Password' and 'Cancel'. Both buttons have a blue gradient.

Some snaps of when the PCMS is run

Chapter - 4

CODING

Code of login page for applicants/students:-

```
if(app.isSelected()){
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        clog=DriverManager.getConnection("jdbc:odbc:go");
        st=clog.createStatement();
        ResultSet rsLog1=st.executeQuery("select * from Applicant");

        while(rsLog1.next()){
            boolean b1=rsLog1.getString(8).equals((String)usr.getText());
            boolean b2=rsLog1.getString(9).equals((String)pwd.getText());
            if( b1 && b2 ){
                JOptionPane.showMessageDialog(null,"done");           //Remove this line
                new empdetail(rsLog1.getInt(7));
                flag=1;
                break;
            }
        }
        if(flag!=1 && i<=3){
            ++i;
            JOptionPane.showMessageDialog(null,"Invalid Entry");
        }
        else if(flag!=1 && i>3){
            System.exit(0);
        }
        else if(flag==1){
            setVisible(false);
        }
    }
    catch(Exception excp){
        JOptionPane.showMessageDialog(null,excp);
        System.exit(0);
    }
}
```

Code of login page of employer/recruiter:-

```
else if(emp.isSelected()){
    try{
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        clog=DriverManager.getConnection("jdbc:odbc:go");
        st=clog.createStatement();
        ResultSet rsLog2=st.executeQuery("select * from Company");

        while(rsLog2.next()){
            boolean b1=rsLog2.getString(5).equals((String)usr.getText());
            boolean b2=rsLog2.getString(6).equals((String)pwd.getText());
            if( b1 && b2 ){
                JOptionPane.showMessageDialog(null,"done");           //Remove this line
                new appdetail(rsLog2.getInt(4));
                flag=1;
                break;
            }
        }
        if(flag!=1 && i<=3){
            ++i;
            JOptionPane.showMessageDialog(null,"Invalid Entry");
        }
        else if(flag!=1 && i>3){
            System.exit(0);
        }
        else if(flag==1){
            setVisible(false);
        }
    }
    catch(Exception excp){
        JOptionPane.showMessageDialog(null,excp);
        System.exit(0);
    }
}
}
```

Code of making the form for the students by recruiter:

```
getContentPane().add(p, BorderLayout.CENTER);
getContentPane().add(q, BorderLayout.SOUTH);

setTitle("Placement Office");

lusr=new JLabel("Login Name: ");
lpwd=new JLabel("Password: ");
lname=new JLabel("Name: ");
lage=new JLabel("Age: ");
lsex=new JLabel("Sex: ");
ledu=new JLabel("Education: ");
lextra=new JLabel("Extra: ");
lexpect=new JLabel("Field: ");
lsal=new JLabel("Minimum Expected Salary: ");

tusr=new JTextField(10);
tpwd=new JTextField(10);
tname=new JTextField(20);
tage=new JTextField(2);
textra=new JTextField(15);

rbml=new JRadioButton("Male");
rbfml=new JRadioButton("Female");
ButtonGroup bgrp=new ButtonGroup();
bgrp.add(rbml);
bgrp.add(rbfml);

cbedu=new JComboBox();
cbedu.addItem("Diploma");
cbedu.addItem("Degree");
cbedu.addItem("Post-Graduate");

cbsal=new JComboBox();
cbsal.addItem("5,000");
cbsal.addItem("10,000");
cbsal.addItem("15,000");
cbsal.addItem("20,000");

it=new JRadioButton("Software Engg.");
civil=new JRadioButton("Civil Engg.");
mech=new JRadioButton("Mechanical Engg.");
ButtonGroup bg2=new ButtonGroup();
```

```

bg2.add(it);
bg2.add(civil);
bg2.add(mech);

bSub=new JButton("Submit");
bRes=new JButton("Reset");

p.setLayout(new GridLayout(12,2));
Blank=new JLabel("");
Blank1=new JLabel("");
Blank2=new JLabel("");
Blank3=new JLabel("");
Blank4=new JLabel("");

p.add(lusr);
p.add(tusr);

p.add(lpwd);
p.add(tpwd);

p.add(lname);
p.add(tname);

p.add(lage);
p.add(tage);

p.add(lsex);
p.add(rbm1);

p.add(Blank);
p.add(rbfm1);

p.add(ledu);
p.add(cbedu);

p.add(lextra);
p.add(textra);

p.add(lexpect);
p.add(it);

p.add(Blank4);
p.add(civil);

```

```

p.add(Blank1);
p.add(mech);

p.add(lsal);
p.add(cbsal);

q.add(bSub);
q.add(bRes);

bSub.addActionListener(this);
bRes.addActionListener(this);
setVisible(true);
setBounds(200,200,500,500);
setDefaultCloseOperation(EXIT_ON_CLOSE);
}

public void actionPerformed(ActionEvent e){
    Object source=e.getSource();

    if(source==bRes){
        tname.setText("");
        tage.setText("");
        cbedu.setSelectedItem("Diploma");
        textra.setText("");
        tusr.setText("");
        tpwd.setText("");
    }

    else{

        if(it.isSelected())field=1;
        else if(civil.isSelected())field=2;
        else if(mech.isSelected())field=3;

        if(rbml.isSelected())sex="MALE";
        else if(rbfml.isSelected())sex="FEMALE";

        try{
            int age=Integer.parseInt(tage.getText());}
        catch(Exception ex){tage.setText("");}
    }
}

```


CHAPTER – 5

TESTING

5.1 WHITE BOX TESTING

White Box Testing is a software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, Open box testing, Transparent box testing, Code-based testing and Glass box testing.

It is one of two parts of the Box Testing approach to software testing. Its counterpart, Blackbox testing, involves testing from an external or end-user type perspective. On the other hand, White box testing in software engineering is based on the inner workings of an application and revolves around internal testing.

The term "WhiteBox" was used because of the see-through box concept. The clear box or WhiteBox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "[Black Box Testing](#)" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested.

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

5.1.1 WHITE BOX TESTING - HOW IT IS DONE?

The process of doing the white box testing can be done in two major steps :

UNDERSTAND THE SOURCE CODE

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

CREATE TEST CASES AND EXECUTE

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have a knowledge of the code and is often done by the developer.

5.1.2 WHITE BOX TESTING - WHAT ARE THE ADVANTAGES ?

Code optimization by finding hidden errors.

White box test cases can be easily automated.

Testing is more thorough as all code paths are usually covered.

Testing can start early in SDLC even if GUI is not available.

We must keep in mind that white box testing usually is more time consuming and takes a lot of efforts of the developers to test a particular operation. Also as it requires end-to-end knowledge of the source code, it becomes really difficult to

test if a person is not the developer and hence increases the complexity of the testing procedure.

5.2 BLACK BOX TESTING

Black Box Testing is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.



The above Black-Box can be any software system you want to test. For Example, an operating system like Windows, a website like Google, a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

5.2.1 How to do BlackBox Testing

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially, the requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

5.2.2 Types of Black Box Testing

There are many types of Black Box Testing but the following are the prominent ones -

- **Functional testing** - This black box testing type is related to the functional requirements of a system; it is done by software testers.
- **Non-functional testing** - This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.
- **Regression testing** - Regression Testing is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

5.2.3 Tools used for Black Box Testing:

Tools used for Black box testing largely depend on the type of black box testing you are doing.

- For Functional/ Regression Tests you can use - QTP, Selenium

- For Non-Functional Tests, you can use - LoadRunner, Jmeter

5.2.4 Black Box Testing and Software Development Life Cycle (SDLC)

Black box testing has its own life cycle called Software Testing Life Cycle (STLC) and it is relative to every stage of Software Development Life Cycle of Software Engineering.

- **Requirement** - This is the initial stage of SDLC and in this stage, a requirement is gathered. Software testers also take part in this stage.
- **Test Planning & Analysis** - Testing Types applicable to the project are determined. A Test Plan is created which determines possible project risks and their mitigation.
- **Design** - In this stage Test cases/scripts are created on the basis of software requirement documents
- **Test Execution**- In this stage Test Cases prepared are executed. Bugs if any are fixed and re-tested.

5.3 MANDATORY TESTING METHODS

1. Alpha Testing

Alpha testing is done when the software is almost 60–80% complete. There is no fixed testing cycle. Each cycle might go up to two weeks. Alpha testing involves both black box and white box testing.

Generally, alpha testing is used for off-the-shelf software development and is done before beta testing.

Alpha testing is a type of acceptance testing which is conducted to identify the bugs before the product is released in the market. It is possible to make minor changes as a result of alpha testing.

2. Beta Testing

Beta testing is also known as pre-release testing. It is the second step of testing which is performed before the app is released commercially to the public. Typically, the beta version of the software is released to a limited number of users.

Beta testing involves only one or two cycles with each test cycle lasting around three to six weeks. Beta testing ensures that there are no major failures in the software, and also gives developers an idea of whether it satisfies the business requirements from the end user's perspective.

There are two versions of beta testing:

- **Open beta version:** the software is open for a wide audience. Anyone who is interested can report bugs and also suggest additional features to improve the final version of the software.
- **Closed beta version:** The software is released to a selected group of end-users and is by invitation only.

The biggest advantage of beta testing is that the percentage of product failure is greatly reduced because the software is validated by the customer.

3. Backend testing

Backend testing or database testing takes place on the server side. The databases used are MYSQL, DB2, Oracle, SQL and so on. If database testing is not done, it can result in serious complications like deadlock, loss of data and disk corruption. Database testing includes the following processes:

- Avoiding data duplication
- Verifying keys and indexes
- Validating Schema tables
- Database server approvals

Also, keep in mind that backend testing is extremely different from black box testing. Backend testing gives total control over the algorithm in the software. It also allows debugging through log files.

4. GUI testing

The main aim of Graphical User Interface testing is to approve the GUI as per the demands and needs of the client.

The user's first impression will be the design and appearance of the application or software. If it does not appeal to them, they will never come back to the application or software. This is where GUI testing comes into play.

It involves validating user interface aspects like the main menu, icon, toolbar, dialogue boxes, menu bar, windows and many more. The most popular GUI testing tools are Selenium, Cucumber, SilkTest, QTP and TestComplete.

Chapter - 6

CONCLUSION

This project helped us to know about the end to end flow of an application. To complete this project we had to read about Databases, Java as a language and also about the toolkits in order to design the application. We came to know about various testing techniques which are used in the process of software development. We also came to know about the practicality of various Software development models used in software engineering. Though we followed the waterfall model in developing our project, we also read about the other methodologies used in the industry. The process of building an application from scratch also helped us as a team to have good bonds between us. There were situations where we thought of leaving this idea but we somehow managed to fight back and try to stick to the plans we made initially. The motivation to build this project was our college's way of taking info from the students. We found that we were given google forms to fill our placement details, which in turn made us realise that we need a portal, a portal which will be accessible by both - the professors and the students. The students would fill their placement

details and this would generate an excel like file to the professor's end. This was the motivation for us and we started working on the same. At first we had no idea how we would build such an application, which stack would we use in our project, these were some of the major concerns as we started our journey. But as time passed by, we all learnt new technologies, and chose java as the language and it was really difficult to have an agreement on the same. Some of our team mates used python and used python to satisfy their needs. I personally thank them, ones who came out of their comfort zone and learnt about the language in order to help us in achieving our goals. Along with technicalities this project really helped to develop our soft skills to some extent, as it takes a lot of efforts to convey the message that you disagree, without hurting a friend.

Chapter - 7

REFERENCES

1. https://www.youtube.com/watch?v=t6NQtfokZr8&ab_channel=CodeJava
2. <https://www.geeksforgeeks.org/short-note-on-project-scheduling/>
3. <https://www.geeksforgeeks.org/rules-for-data-flow-diagram/>
4. <https://www.udemy.com/course/database-design-and-management/>
5. <https://www.javatpoint.com/java-swing>
6. <https://www.softwaretestinghelp.com/types-of-software-testing/>
7. <https://docs.oracle.com/en/java/>
8. <https://www.quackit.com/database/tutorial/>
9. <http://www.successcds.net/learn-english/writing-skills/report-writing-format-topics-samples.html>
10. https://www.w3schools.com/mysql/mysql_rdbms.asp

SILIGURI INSTITUTE OF TECHNOLOGY

IT 892

LIBRARY MANAGEMENT SYSTEM

BY

IT_PROJ_2021_Group-02

Name of Students	Roll No.
1. Himadri Bhattacharya	11900217019
2. Chayan Karmakar	11900217021
3. Biswajit Sharma	11900217022
4. Ayan Dutta	11900217026

Under the Guidance

of

Prof. Ms. Sathi Ball

Submitted to the Department of **Information Technology** in partial fulfillment of the requirements for the award of the degree Bachelor of Technology in **Information Technology**.

Year of Submission: 2021



Siliguri Institute of Technology

P.O. SUKNA, SILIGURI, DIST. DARJEELING, PIN: 734009

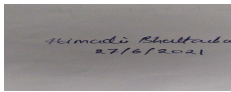
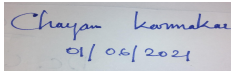
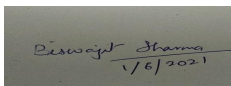
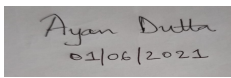
Tel: (0353)2778002/04, Fax: (0353) 2778003

DECLARATION

-

This is to certify that Report entitled “**Library Management System**” which is submitted by me in partial fulfillment of the requirement for the award of degree B. Tech. in **Information Technology** at **Siliguri Institute of Technology** under **Maulana Abul Kalam Azad University of Technology**, West Bengal. We took the help of other materials in our dissertation which have been properly acknowledged. This report has not been submitted to any other Institute for the award of any other degree.

Date: 05/07/2021

S	Name of the Student	Roll No	Signature
N			
1	Himadri Bhattacharya	1190021701 9	
2	Chayan Karmakar	1190021702 1	
3	Biswajit Sharma	1190021702 2	
4	Ayan Dutta	1190021702 6	

CERTIFICATE

This is to certify that the project report entitled “**LIBRARY MANAGEMENT SYSTEM**” submitted to **Department of Information Technology of Siliguri Institute of Technology** in partial fulfilment of the requirement for the award of the degree of **Bachelor of Technology in Information Technology** during the academic year **2020-21**, is a bonafide record of the project work carried out by them under my guidance and supervision.

Project Group Number : 5			
SN	Name of the students	Registration No	Roll No
1.	Himadri Bhattacharya	171190110101	11900217019
2.	Chayan Karmakar	171190110099	11900217021
3.	Biswajit Sharma	171190110098	11900217022
4.	Ayan Dutta	171190110094	11900217026

Signature of P

Name of the G

Signature of the HOD

Department of Information Technology

ACKNOWLEDGEMENT

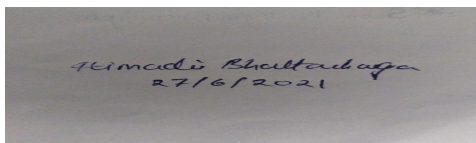
The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful.

We take this opportunity to express our deep gratitude towards our project mentor, **Ms. Sathi Ball** for giving such valuable suggestions, guidance and encouragement during the development of this project work.

-

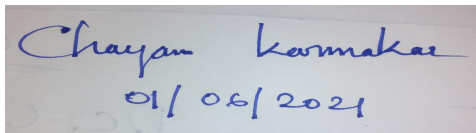
Signature of all the group members with date

1.



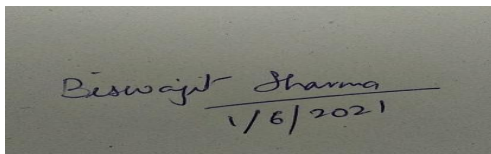
Himadri Bhattacharya
27/6/2021

2.



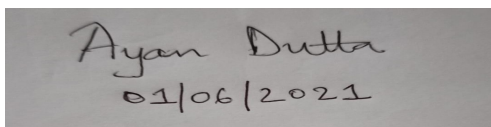
Chayan Karmakar
01/06/2021

3.



Biswajit Sharma
1/6/2021

4.



Ayan Dutta
01/06/2021

TABLE OF CONTENTS

CHAPTER 1 : INTRODUCTION01 - 03

- 1.1 OBJECTIVE01
- 1.2 SCOPE 02-03
- 1.3 OPERATION ENVIRONMENT 03

CHAPTER 2 : SYSTEM ANALYSIS04 - 15

- 2.1 IDENTIFICATION OF NEED 04
- 2.2 PRELIMINARY INVESTIGATION 05
- 2.3 SYSTEM ARCHITECTURE 06
- 2.4 FEASIBILITY STUDY 07
- 2.5 PROJECT PLANNING 08
- 2.6 PROJECT SCHEDULING 09
- 2.7 SOFTWARE REQUIREMENTS SPECIFICATION10
- 2.8 SOFTWARE ENGINEERING PARADIGM APPLIED 11
- 3.1 DATA FLOW DIAGRAM 12-13
- 3.2 ENTITY RELATIONSHIP DIAGRAM 14-15

CHAPTER 3 : SYSTEM DESIGN16 - 20

- 3.1 MODULARISATION DETAILS 16 -17
- 3.2 DATABASE DESIGN 17 -19
- 3.3 USER INTERFACE DESIGN 20

CHAPTER 4 : CODING21 - 30

- 4.1 SOURCE CODE21 - 30

CHAPTER 5 : TESTING 31 - 35

- 5.1 UNIT TESTING31 - 32
- 5.2 SYSTEM TESTING33 - 35

CHAPTER 6 : CONCLUSION AND RECOMMENDATION36

- 6.1 CONCLUSION36
- 6.2 RECOMMENDATIONS36

CHAPTER 7: REFERENCING AND APPENDICES 37

- 7.1 REFERENCES37
-

ABSTRACT

Library Management System is a computerized system used to store and retrieve information related to books and students and transactions of books. The project is aimed at making the work simple and easier for librarians. It is projected towards enhancing the relationship between students and library by making it convenient for the students and the librarians such that when the student access books the librarians can manage books easily using this software.

The software first asks the user to **Login** along with two other options, viz. **Signup** and **Forgot Password**. If the user does not have an account, he can sign up, where the software will ask the user to set the following details:

- Username;
- Name;
- Password;
- Security Question and its Answer.

There are a bunch of security questions for the user to choose from. The username of any 2 account can't be same. If user already has an account, the user can just login by entering username and password. There is also an option for Forgot Password in case the user has forgotten the password. There the username, name, security question and answer have to be entered and it will show the password for that account. After login there are two sections, **Operation** and **Action**. Under operation there is **New Book** to add new book in database, **New Student** to register new student and **Statistics** that shows issue and return details. In New Book and New Student there can't be two same Book Id and Student Id. Under action, there is issue book to issue a book for a student, return book to take record if a student has returned the book issued. The issue book and return book options take inputs for both student and book. One has to put registered book details and student details. Once a student returns his or her book the issue book record gets deleted.

The main purpose of the project is to remove paperwork for the Librarians and store the data in a better and easy way and hence removing errors which generally occurs while doing everything manually.

Chapter 1

INTRODUCTION

Library Management System is a very useful open source desktop application for providing user flexibility to perform various actions that are usually used by a Librarian and automating the operations performed over the information about the members, book issues and returns and all other operations. It reduces the workload of management as most of the manual work done is significantly reduced. Thus this innovative library management system provides a path for transition into the rapidly growing digitalized era of this modern world.

1.1 OBJECTIVE

The project's aims and objectives that will be achieved after completion of the system were carried out in this sub chapter. The succession of the system also will be evaluated through this sub chapter. The project objectives are:

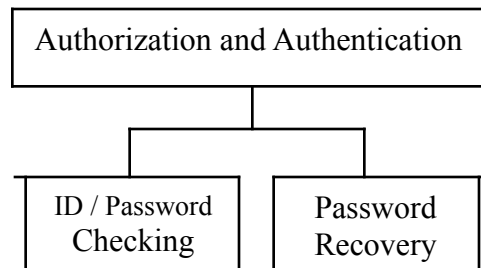
- To provide an open source software solution and promote the use of paperless/digital work in every library possible;
- To record every transaction in computerized system so that problem such as record file missing won't happen again;
- To design a user friendly graphical user interface which suits even the computer-novice users;
- To complete the system according to project schedule;
- To produce technical report that documents the phases, tasks and deliverables in the project.

Library Management System provides a user-friendly data entry in purpose to make the input entry easier to understand and use. It is also created to ensure that the books and student information is stored properly in order to maintain their security.

1.2 SCOPE

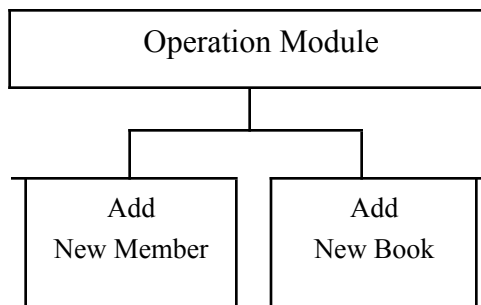
In this sub chapter, project scope will carry out what modules were containing inside the Library Management System. I will be describing the part which is library system which used by librarian.

1.2.1 Authorization and Authentication Module:



This module is used by user which means librarian in the library. They need to login to the system using their id and password.

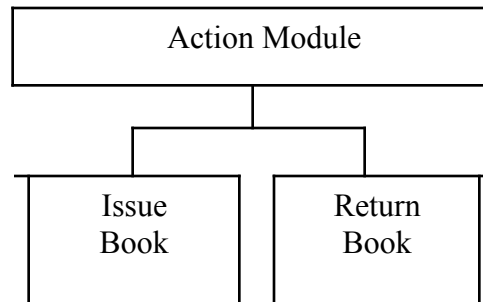
1.2.2 Operation Module:



This module is accessed by the user(librarian) to maintain member's profile or record such as Search, Add, etc. This module can be used by the user(librarian) to maintain the book inventory record such as Search, Add, etc.

There is a **Statistics** functionality added in this module which gives the overall image of the inventory records of the system.

1.2.3 Action Module:



This module is used by the user(librarian) to maintain the details of books issued and returned on a particular date to/from a particular member.

There is an **About Us** functionality added in this module which displays the name and contact info of the developers of this project.

1.3 OPERATION ENVIRONMENT

The table shown below is the minimum requirement for the Library Management System:

Processor	Intel Pentium 233Ghz or better performance
Operating System	Microsoft Windows 7 or above
Memory	2GB RAM
Screen Resolution	Monitor with screen resolution minimum 1024 x 768
Hard disk Space	Minimum 5GB to include database usage for future
Database	SQLite

Chapter 2

SYSTEM ANALYSIS

2.1. IDENTIFICATION OF NEED

2.1.1 EXISTING SYSTEM:

The current Library Management System does not eliminate the process of searching books within the library campus. Students have to find books manually. They have to wait until they are not provided with their library card and token. For receiving book, they have to show their library card and wait in line for their turns. The librarian also has to look manually on which day which book was taken by which member making the system time consuming.

2.1.2 PROPOSED SYSTEM:

Proposed system is a Computerized Library Management System. Through our software user can add members, add books, search members, search books, borrow and return books in quick time. Our proposed system so many advantages like User friendly interface, Fast access to database, less error prone, more Storage Capacity, Search facility, Look and Feel Environment and Quick Transaction. All the manual difficulties in managing the Library have been rectified by implementing computerization.

2.2. PRELIMINARY INVESTIGATION

Library Management System is an application refer to other library system and it is suitable to use by small and medium size library. It is used by the librarian to manage the library using a computerized system. The system was developed and designed to help librarian record every book transaction with minimal complications. It is convenient and time saving than the traditional paper-dependent approach.

All these modules are able to help librarian to manage the library with more convenience and in a more efficient way as compared to library systems which are not computerized.

In addition, the Statistics module included in Library Management System, gives the user the ability to view Issued Books' and Returned Books' reports of a particular day.

All these modules help the user(librarian) to manage the library with more convenience and efficiency compared to those libraries without computerized system.

2.3. SYSTEM ARCHITECTURE

The computer based Management application uses client/server engineering. It is an offline app designed using JAVA and SQLite.



2.3.1. Database outline:

The database connected to this application is SQLITE which is worked with in free host server. Four tables are made which are; clients, flight, city, and book. Client data, booking data would be spared in clients and book table. Admin of system has the capacity to refresh flight data by including new flight, change client data or booking data, contact with the client if there is a mistake or flight cancellation or any an emergency.

2.4. FEASIBILITY STUDY

2.4.1 Evaluation of Project Objectives:

Objectives	Function / Module	Status
Eliminate paper work	Operation Module	Achieved
To add as well as track member and book renting history	Action Module	Achieved

2.4.2 Evaluation of Projects Strengths and Weaknesses:

Strengths

- User interface is considered user friendly and ease of use.
- Some listview control has been included to allow user view the history or some important data without print out the report.
- Search features are included in several modules to ease user so that they can filter the data easily.
- Allow user to keep track member renting history.
- System is able to check the book whether is under reservation or not.

Weakness

- Doesn't allow user to generate bar code of the book without using additional bar code generator software.
 - No feature of online book reservation in the system for the members.
 - System didn't implement smart card technology so member card can be generated using the system.
-

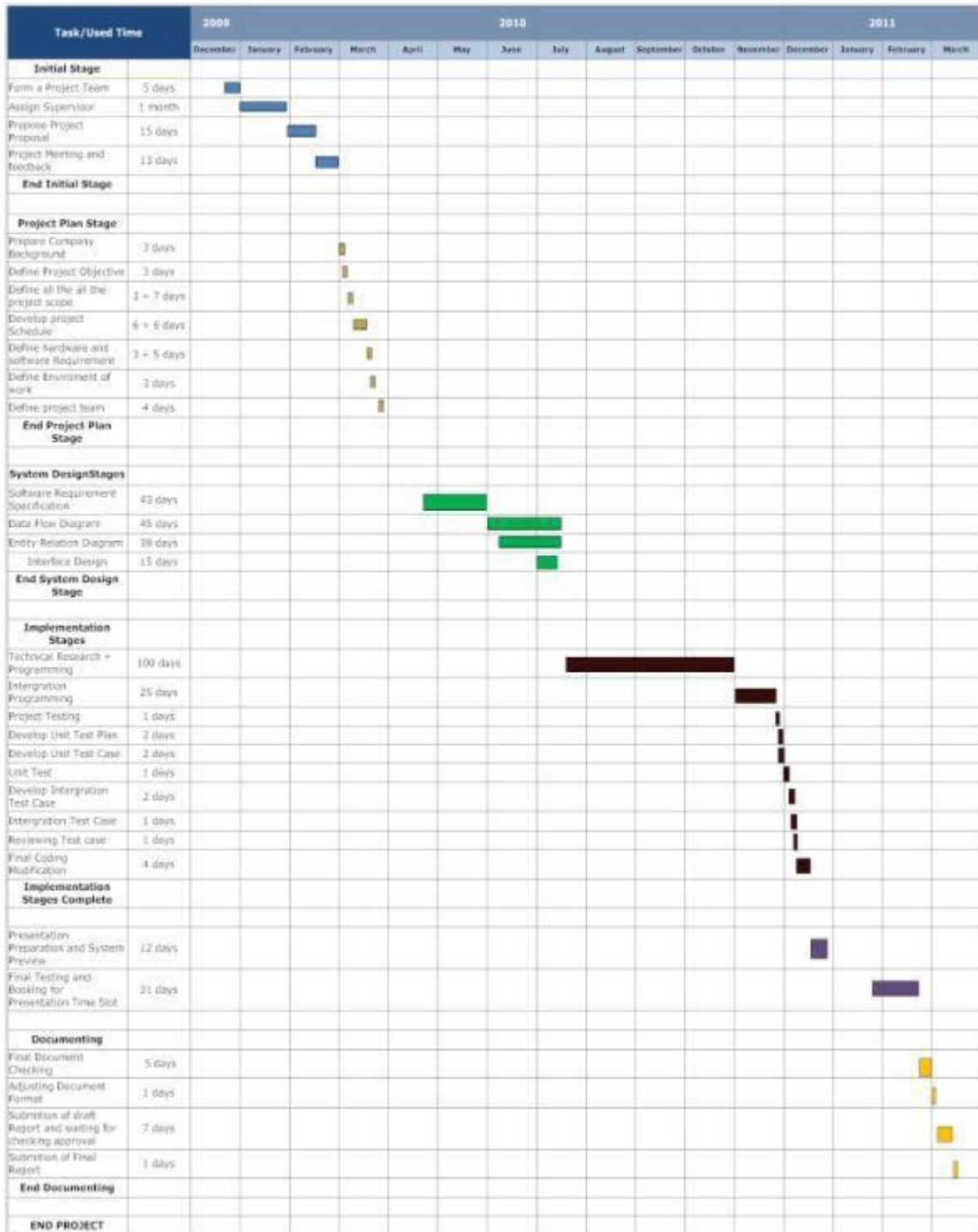
2.5. PROJECT PLANNING

In Royce's original waterfall model, the waterfall model originally consists of 7 phases which is Requirement Specification, Design, Construction, Integration, Testing and Debugging, Installation and Maintenance.

Firstly, we collect the requirement for the library system, and then we analyzed it. After analyzing the requirement, we proceeded to design stage. In the design phase, the user interface (UI) and the database design is designed. The next phase in waterfall model is Construction phase. It is an important phase in waterfall model and it is a time consuming phase depends on programmer's ability. In Library Management System development, construction phase is using Eclipse IDE to write the program. After the coding phase, we proceed to integration phase. In this phase, the software shares a database to make integration between two applications. The next phase after integration is the testing and debugging phase. The testing module is separated into module testing, system testing, unit testing and user acceptance test. Once there is a bug found, it will be solved immediately before the system is launched to ensure the system launched is bug free.

Lastly, it is installation and maintenance phase where the system will be installed at user side. After installed the system, maintenance is compulsory needed to ensure the system is always-on and up to date with latest technologies or latest business process.

2.6. PROJECT SCHEDULING



2.7. SOFTWARE REQUIREMENTS SPECIFICATIONS(SRS)

2.7.1. Hardware Requirements:

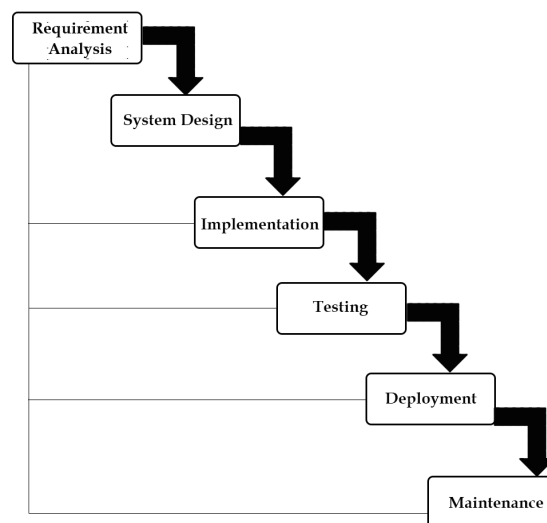
- Processor: Intel i3 6th Gen Processor 2.00 Ghz is used, where we can keep developing the Library Management System without need to worry that the pc cannot support.
- Ram: 4 Gb is used to support Eclipse IDE and MySQL and to avoid any problem that may arise during the development phase.

2.7.2. Software Requirements:

- Operating system - Windows 10 is used as the operating system as it is the latest stable build of Microsoft;
 - IDE- Eclipse IDE is used
 - Database SQLite- SQLite is used as database as it easy to maintain and retrieve records by simple queries which are in English language which are easy to understand and easy to write;
-

2.8. SOFTWARE ENGINEERING PARADIGM APPLIED

The software used to develop Library Management System is Eclipse IDE and MySQL as the database. On the other hand, the methodology we used to develop this system is waterfall model. Waterfall model is one of a system development life cycle(SDLC) model. Users proceed to next phase if and only if current phase is complete. Users are not allowed go back to previous phases if there are any mistake so they named it waterfall model, just like the water is always fall down from the waterfall and not flow upward.



The waterfall model was selected as the SDLC model due to the following reasons:

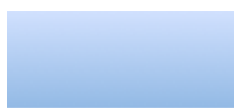
- Requirements were very well documented, clear and fixed.
 - Technology was adequately understood.
 - Simple and easy to understand and use.
 - There were no ambiguous requirements.
 - Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
 - Clearly defined stages.
 - Well understood milestones.
 - Easy to arrange tasks.
-

2.9. DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a diagram that describes the flow of data and the processes that change data throughout a system. A structured analysis and design tool that can be used for flowcharting in place of or in association with information. Oriented and process oriented system flowcharts. When analysts prepare the Data Flow Diagram, they specify the user needs at a level of detail that virtually determines the information flow into and out of the system and the required data resources. This network is constructed by using a set of symbols that do not imply physical implementations. The Data Flow Diagram reviews the current physical system, prepares input and output specification, specifies the implementation plan etc.

Four basic symbols are used to construct data flow diagrams. They are symbols that represent data source, data flows, and data transformations and data storage. The points at which data are transformed are represented by enclosed figures, usually circles, which are called nodes.

2.9.1. Data Flow Diagram Symbols:



Source or Destination of Data



Data Flow



Process

2.9.2. Steps to Construct Data Flow Diagram:

Four Steps are generally used to construct a DFD.

- Process should be named and referred for easy reference. Each name should be representative of the reference.
- The destination of flow is from top to bottom and from left to right.
- When a process is distributed into lower level details they are numbered.
- The names of data stores, sources and destinations are written in capital letters.

Rules for constructing a Data Flow Diagram-

- Arrows should not cross each other.
- Squares, Circles, Files must bear a name.
- Decomposed data flow squares and circles can have same names. Draw all data flow around the outside of the diagram.

2.9.3. Zero Level DFD:

2.9.4. Entity-Relationship Diagram:

In software engineering, an **entity-relationship model (ER model)** is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that

lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities(things) and the relationships that can exist among them. However, variants of the idea existed previously, and have been devised subsequently such as super type and subtype data entities and commonality relationships.

An entity–relationship model is a systematic way of describing and defining a business process. The process is modeled as components (*entities*) that are linked with each other by *relationship* that express the dependencies and requirements between them, such as: *one building may be divided into zero or more apartments, but one apartment can only be located in one building.*

Entities may have various properties (*attributes*) that characterize them. Diagrams created to represent these entities, attributes, and relationships graphically are called entity–relationship diagrams.

An ER model is typically implemented as a database. In the case of a relational database, which stores data in tables, every row of each table represents one instance of an entity. Some data fields in these tables point to indexes in other tables; such pointers represent the relationships.

The three schema approach to software engineering uses three levels of ER models that may be developed.

An entity may be defined as a thing capable of an independent existence that can be uniquely identified. An entity is an abstraction from the complexities of a domain. When we speak of an entity, we normally speak of some aspect of the real world that can be distinguished from other aspects of the real world.

A relationship captures how entities are related to one another. Relationships can be thought of as verbs, linking two or more nouns.

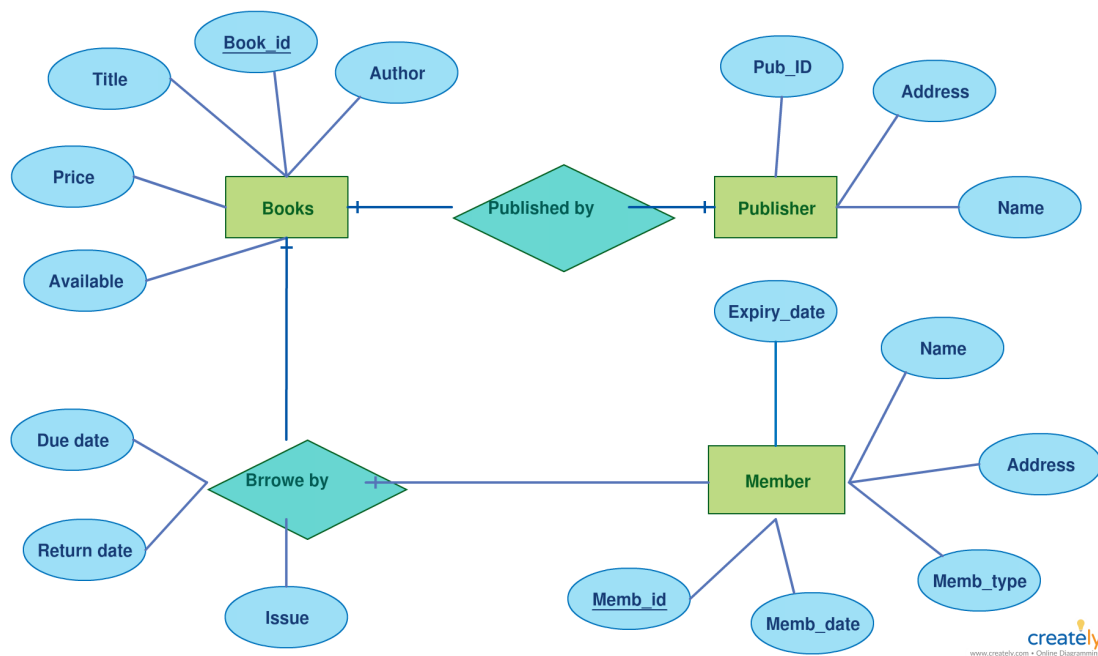
Cardinality constraints are expressed as follows:

- a double line indicates a *participation constraint*, totality or
-

subjectivity: all entities in the entity set must participate in *at least one* relationship in the relationship set;

- an arrow from entity set to relationship set indicates a key constraint, i.e. injectivity: each entity of the entity set can participate in *at most one* relationship in the relationship set;
- A thick line indicates both, i.e. bijectivity: each entity in the entity set is involved in *exactly one* relationship.
- An underlined name of an attribute indicates that it is a key two different entities or relationships with this attribute always have different values for this attribute.

E-R Diagram for Library Management System



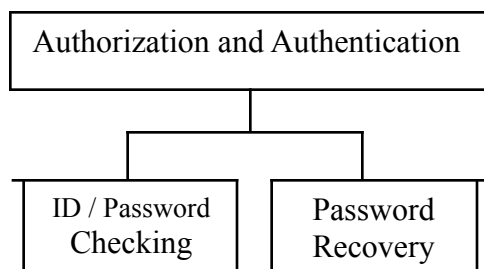
Chapter 3

SYSTEM DESIGN

3.1. MODULARISATION DETAILS

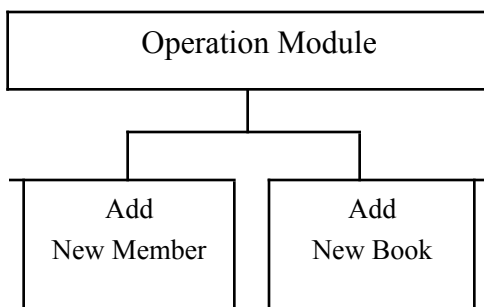
In this sub chapter, project scope will carry out what modules were containing inside the Library Management System. I will be describing the part which is library system which used by librarian.

1.2.1 Authorization and Authentication Module:



This module is used by user which means librarian in the library. They need to login to the system using their id and password.

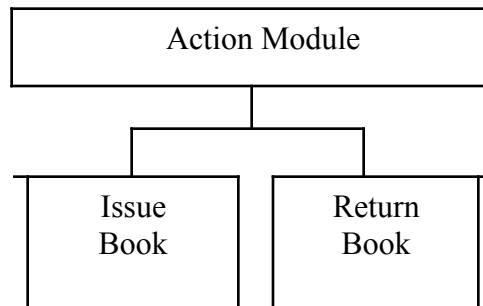
1.2.2 Operation Module:



This module is accessed by the user(librarian) to maintain member's profile or record such as Search, Add, etc. This module can be used by the user(librarian) to maintain the book inventory record such as Search, Add, etc.

There is a **Statistics** functionality added in this module which gives the overall image of the inventory records of the system.

1.2.3 Action Module:



This module is used by the user(librarian) to maintain the details of books issued and returned on a particular date to/from a particular member.

There is an **About Us** functionality added in this module which displays the name and contact info of the developers of this project.

3.2. DATABASE DESIGN

A database is an organized mechanism that has capability of storing information through which a user can retrieve stored information in an effective and efficient manner. The data is the purpose of any database and must be protected.

The database design is two level processes. In the first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called information Level design and it is taken independent of any individual DBMS.

In the following snapshots we display the way we have used SQL Server as the backend RDBMS for our project and the various

entities that have been used along with their table definition and table data.

3.2.1. Database Tables:

LIST OF TABLES

TABLE – Account :

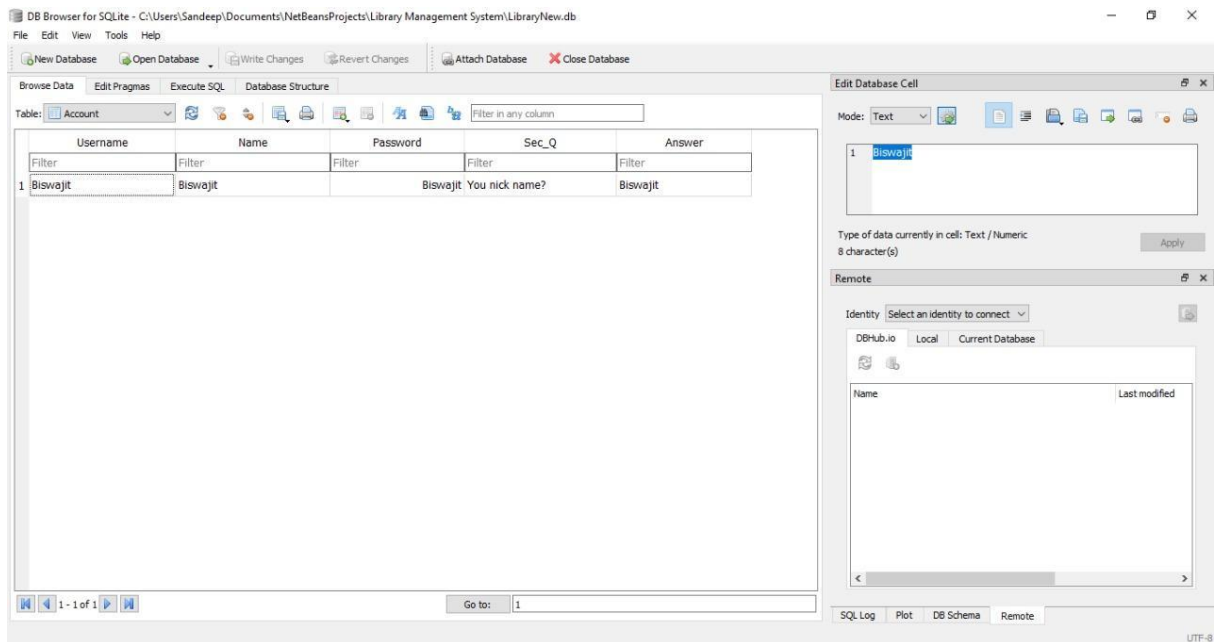


TABLE – Student :

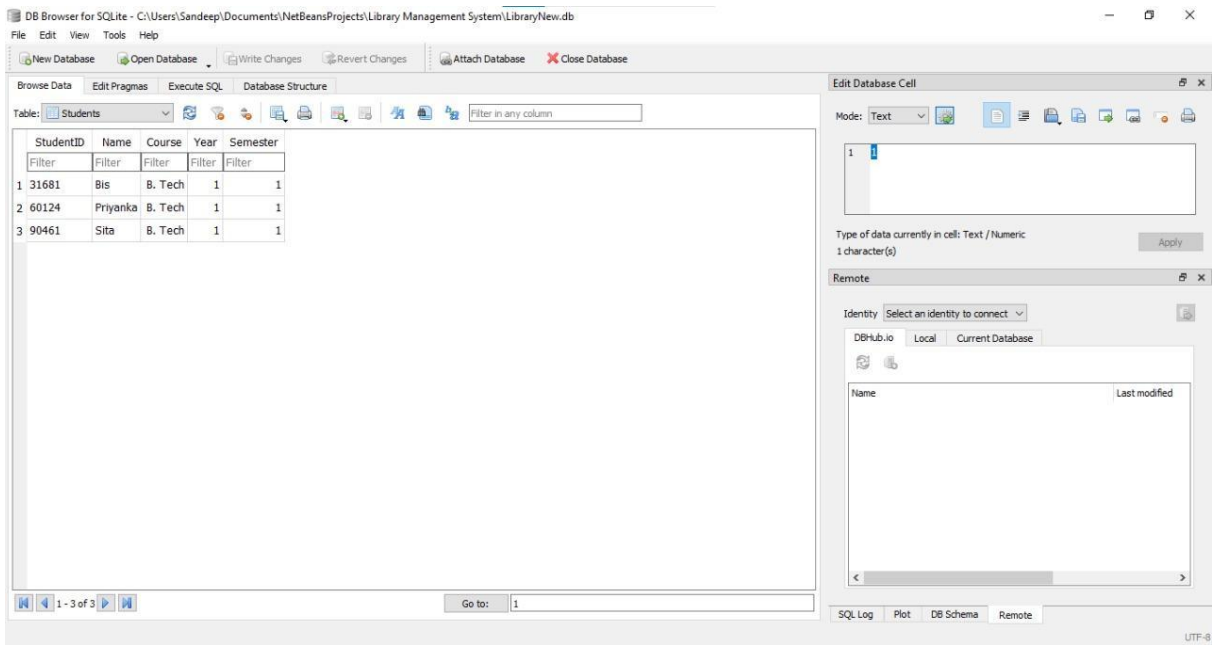


TABLE - Books:

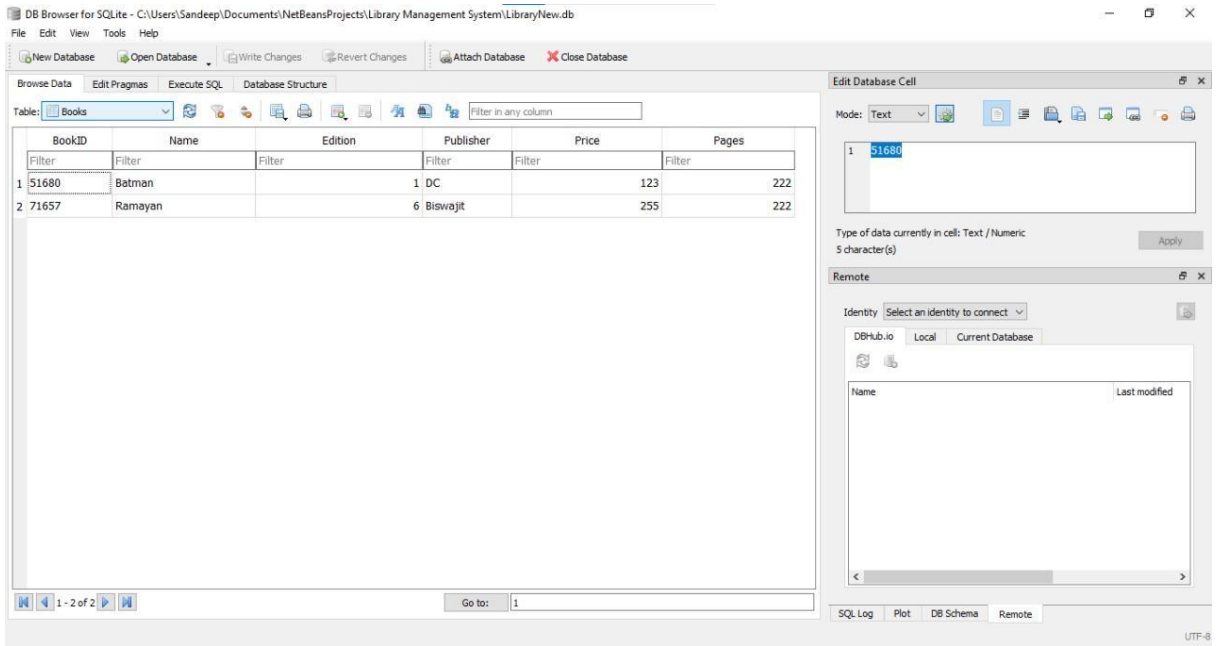
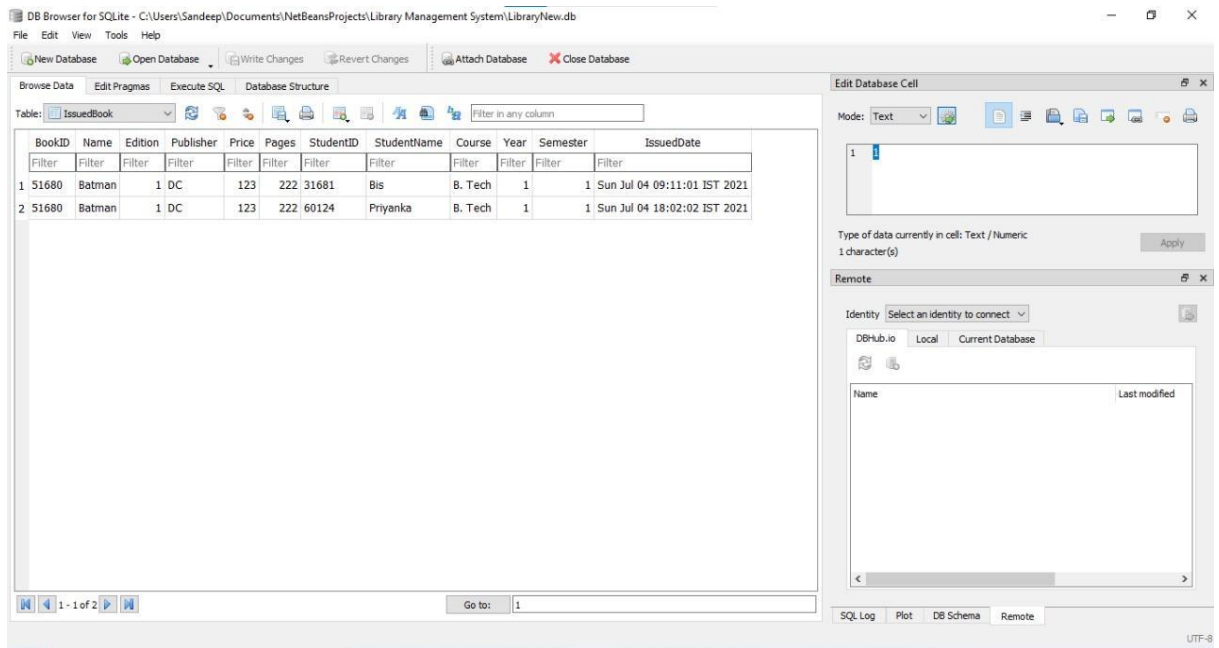
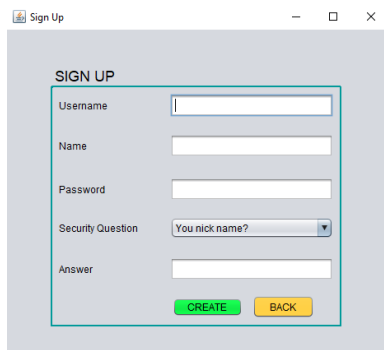


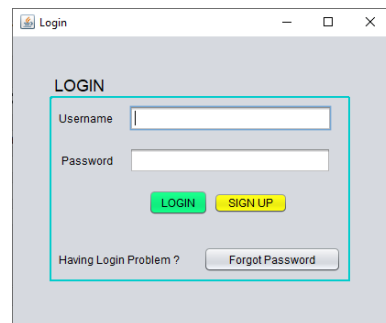
TABLE - IssueBooks :



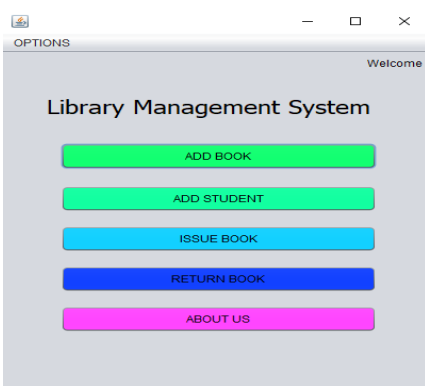
3.3. USER INTERFACE DESIGN

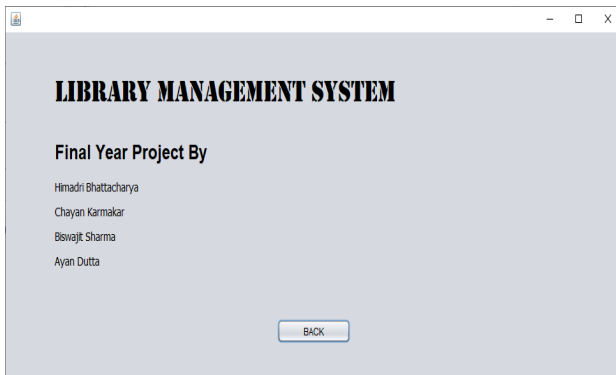


Sign up page to create new user



Login page to log in to the user dashboard





**It's a user dashboard.
Here we can add Book, Student.
And also return book...**

Information About us.

Here we can Issue our book

Chapter 4

CODING



4.1. SOURCE CODE

```
import javax.swing.UIManager;

/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
/**
 *
 * @author Sandeep
 */
public class Home extends javax.swing.JFrame {

    /**
     * Creates new form Home
     */
    public Home() {
        initComponents();
    }
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
```

```
private void initComponents() {  
    jLabel1 = new javax.swing.JLabel();  
    jLabel2 = new javax.swing.JLabel();  
    addStudent = new javax.swing.JButton();  
    addBook = new javax.swing.JButton();  
    returnBook = new javax.swing.JButton();  
    issueBook = new javax.swing.JButton();  
    aboutUs = new javax.swing.JButton();  
    jMenuBar1 = new javax.swing.JMenuBar();  
    jMenu1 = new javax.swing.JMenu();  
    jMenuItem1 = new javax.swing.JMenuItem();  
    jMenuItem2 = new javax.swing.JMenuItem();  
  
    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);  
  
    jLabel1.setFont(new java.awt.Font("Tahoma", 0, 24)); // NOI18N  
    jLabel1.setText("Library Management System");  
  
    jLabel2.setText("Welcome");  
    addStudent.setBackground(new java.awt.Color(0, 255, 153));  
    addStudent.setText("ADD STUDENT");  
    addStudent.addActionListener(new java.awt.event.ActionListener() {  
        public void actionPerformed(java.awt.event.ActionEvent evt) {  
            addStudentActionPerformed(evt);  
        }  
    });  
}
```

```
addBook.setBackground(new java.awt.Color(0, 255, 102));
addBook.setText("ADD BOOK");
addBook.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        addBookActionPerformed(evt);
    }
});
returnBook.setBackground(new java.awt.Color(0, 51, 255));
returnBook.setText("RETURN BOOK");
returnBook.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        returnBookActionPerformed(evt);
    }
});
issueBook.setBackground(new java.awt.Color(0, 204, 255));
issueBook.setText("ISSUE BOOK");
issueBook.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        issueBookActionPerformed(evt);
    }
});
aboutUs.setBackground(new java.awt.Color(255, 51, 255));
aboutUs.setText("ABOUT US");
aboutUs.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        aboutUsActionPerformed(evt);
    }
});
```

```
jMenu1.setText("OPTIONS");

jMenuItem1.setText("Exit");
jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem1ActionPerformed(evt);
    }
});
jMenu1.add(jMenuItem1);

jMenuItem2.setBackground(new java.awt.Color(255, 0, 0));
jMenuItem2.setText("Logout");
jMenuItem2.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jMenuItem2ActionPerformed(evt);
    }
});
jMenu1.add(jMenuItem2);
jMenuBar1.add(jMenu1);
setJMenuBar(jMenuBar1);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .add(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .add(layout.createSequentialGroup()
                    .addGap(41, 41, 41)

```

```

        .addComponent(jLabel1, javax.swing.GroupLayout.DEFAULT_SIZE, 326, Short.MAX_VALUE)
        .addGap(33, 33, 33))
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup())
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup())
        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
            .addComponent(addBook, javax.swing.GroupLayout.PREFERRED_SIZE, 297,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(addStudent, javax.swing.GroupLayout.PREFERRED_SIZE, 297,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(returnBook, javax.swing.GroupLayout.PREFERRED_SIZE, 297,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(issueBook, javax.swing.GroupLayout.PREFERRED_SIZE, 297,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(aboutUs, javax.swing.GroupLayout.PREFERRED_SIZE, 297,
                javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(49, 49, 49))
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup())
    .addComponent(jLabel2)
    .addContainerGap()))
);

layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup())
        .addGap(6, 6, 6)
        .addComponent(jLabel2)
        .addGap(18, 18, 18)
        .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 58,
            javax.swing.GroupLayout.PREFERRED_SIZE)

```

```

        .addGap(18, 18, 18)

        .addComponent(addBook, javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(21, 21, 21)

        .addComponent(addStudent, javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(18, 18, 18)

        .addComponent(issueBook, javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(18, 18, 18)

        .addComponent(returnBook, javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addGap(18, 18, 18)

        .addComponent(aboutUs, javax.swing.GroupLayout.PREFERRED_SIZE, 33,
javax.swing.GroupLayout.PREFERRED_SIZE)

        .addContainerGap(125, Short.MAX_VALUE))
    );

    setSize(new java.awt.Dimension(416, 539));
    setLocationRelativeTo(null);
} // </editor-fold> // GEN-END: initComponents

private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
{ // GEN-FIRST: event_jMenuItem1ActionPerformed

    // TODO add your handling code here:

    System.exit(0);

} // GEN-LAST: event_jMenuItem1ActionPerformed

private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
{ // GEN-FIRST: event_jMenuItem2ActionPerformed

```

```
// TODO add your handling code here:

setVisible(false);

Login log = new Login();

log.setVisible(true);

} //GEN-LAST:event_jMenuItem2ActionPerformed

private void addBookActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_addBookActionPerformed

    // TODO add your handling code here:

    setVisible(false);

    AddBook newbook = new AddBook();

    newbook.setVisible(true);

} //GEN-LAST:event_addBookActionPerformed

private void addStudentActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_addStudentActionPerformed

    // TODO add your handling code here:

    setVisible(false);

    AddStudent newbook = new AddStudent();

    newbook.setVisible(true);

} //GEN-LAST:event_addStudentActionPerformed

private void issueBookActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_issueBookActionPerformed

    // TODO add your handling code here:

    setVisible(false);

    IssueBook issue = new IssueBook();

    issue.setVisible(true);

} //GEN-LAST:event_issueBookActionPerformed

private void returnBookActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_returnBookActionPerformed
```

```
// TODO add your handling code here:

setVisible(false);

ReturnBook ret = new ReturnBook();

ret.setVisible(true);

} //GEN-LAST:event_returnBookActionPerformed

private void aboutUsActionPerformed(java.awt.event.ActionEvent evt)
{ //GEN-FIRST:event_aboutUsActionPerformed

    // TODO add your handling code here:

    setVisible(false);

    AboutUs about = new AboutUs();

    about.setVisible(true);

} //GEN-LAST:event_aboutUsActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {

    /* Set the Nimbus look and feel */

    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {

        // setTheme(String themeName, String licenseKey, String logoString)

        // select the Look and Feel

        UIManager.setLookAndFeel("com.jtattoo.plaf.acryl.AcrylLookAndFeel");
```

```

        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {

            /*if ("Nimbus".equals(info.getName())) {

                javax.swing.UIManager.setLookAndFeel(info.getClassName());

                break;

            }*/

            UIManager.setLookAndFeel("com.jtattoo.plaf.smart.SmartLookAndFeel");

        }

    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);

    } catch (InstantiationException ex) {

        java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);

    } catch (IllegalAccessException ex) {

        java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

        java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);

    }

}

//</editor-fold>

```

```

/* Create and display the form */

```

```

java.awt.EventQueue.invokeLater(new Runnable() {

    public void run() {

        new Home().setVisible(true);

    }

});

}

```

```
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton aboutUs;
private javax.swing.JButton addBook;
private javax.swing.JButton addStudent;
private javax.swing.JButton issueBook;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JMenu jMenuItem1;
private javax.swing.JMenuBar jMenuItemBar1;
private javax.swing.JMenuItem jMenuItem1;
private javax.swing.JMenuItem jMenuItem2;
private javax.swing.JButton returnBook;
// End of variables declaration//GEN-END:variables
}
```

Chapter 5

TESTING

5.1. UNIT TESTING

Unit testing is undertaken when a module has been created and successfully reviewed. In order to test a single module, we need to provide a complete environment i.e. besides the module we would require

- The procedures belonging to other modules that the module under test calls;
- Non local data structures that module accesses;
- A procedure to call the functions of the module under test with appropriate parameters

Unit testing was done on each and every module that is described under module description of chapter 3

1. Test for the **Authorization and Authentication** module:

- Testing user login frame-This form is used for log in of administrator of the system. Here, we enter the username and password if both are correct, administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask for username and password and in case the user has forgotten his credentials, the forgot password option comes to the rescue. If there is no existing user, the user creates a new user using the signup option.

2. Test for the **Operation** module:

- Student account addition- In this section the admin can verify student details from student academic info and then only
-

add student details to main library database it contains add and delete buttons if user click add button data will be added to student database and if he clicks delete button the student data will be deleted

- Book Addition- User can enter details of book and can add the details to the main book table. He can also view the books requests.

3. Test for the **Action** module:

- Test for Issue Book -This form is used for issuing books to Students. Here we enter Book details, student details and date of issue. If all these are correct, the book is issued to the particular student and a book issue transaction is recorded in the database.
 - Test for Return Book- This form is used for maintain record of the books returned back by the Students. Here we enter Book details, student details, date of issue and date of return. If all these are correct, the book is returned to the library and a book return transaction is recorded in the database.
-

5.2. SYSTEM TESTING

This chapter will show the test plan which programmer used to test the system. System Testing is an important phase in SDLC to ensure the system is bug free. If the system hasn't been test properly, user will meet a lot problem when using the system.

5.2.1. Test Plan:

Tester Name : Ayan Dutta			
Module Name : Authorization and Authentication module			
User ID:	M0022	Password	1234123
Expected Result: Invalid ID			
Comments from Supervisor / Moderator : none			

Tester Name : Ayan Dutta	
Module Name : Authorization and Authentication module	
User ID:	M0001
Expected Result: Invalid Password	
Comments from Supervisor / Moderator : none	

Tester Name : Biswajit Sharma	
Module Name : Operation module/Add new member	
Test method	Key in correct information
Expected Result: Message box show user 'AAA' successfully added	

Comments from Supervisor / Moderator: none

Tester Name : Biswajit Sharma

Module Name : Operation module/Add new book

BOOK ID: 9771985821003

Expected Result:

Book successfully added

Show database to show that book and book copy has been update

Comments from Supervisor / Moderator: none

Tester Name : Himadri Bhattacharya

Module Name : Operation module/Statistics

Expected Result:

Show all the transactions of books in the system

Comments from Supervisor / Moderator :none

Tester Name : Chayan Karmakar

Module Name : Action module/Issue Book

BOOK ID: 584961352652402

Expected Result:

Book was return to database

Try login m0001to show that listview borrowed has nothing and history got one record.

Available book +1

Comments from Supervisor / Moderator : none

Tester Name : Chayan Karmakar	
Module Name : Action module/Return Book	
BOOK ID:	584961352652402
Expected Result: Book was return to database Try login m0001to show that listview borrowed has nothing and history got one record. Available book +1	
Comments from Supervisor / Moderator : none	

5.2.2. Conclusion:

System testing phase made us understand the importance of bug free system. If one system contains bugs, users will be frustrated and meet with some problems. Computerized system should be created to help user reduce the work so it is very importance to ensure the system is bug free.

We must test the system completely and perfectly to ensure the system is bug free. No matter how powerful a system, it will not consider as good system if it contains bugs. The system which is bug free is more powerful and more helpful to end user because it does not produce problems and bugs.

Chapter 6

CONCLUSION AND RECOMMENDATIONS

6.1. CONCLUSION

In Library Management System, the streamlining of the process of managing various aspects in a library without human interaction is highly needed so as to perform well in the highly competitive market. This paper work surveyed some of the existing library management systems of previous work. We will try to solve the existing issues by incorporating new ideas and improvements.

The project “**LIBRARY MANAGEMENT SYSTEM**” provides an entry-level computerized version of library management system which will benefit the staff of the library. It makes entire process paperless where the user can add and search books and members, book transactions for a member, generate reports.

This project has been appreciated by all the users in the organization. It is easy to use, since it uses the GUI provided in the user dialog. User friendly screens are provided. The usage of software increases the efficiency, decreases the effort. It has been efficiently employed as a Site management mechanism. It has been thoroughly tested and implemented.

6.2. RECOMMENDATIONS

In future, we would like to keep working on this project and make new additions to provide users with more advanced features and more detailed information. We have set our sights on the following additions in future: -

- Readers usually tend to forget the date to return their library books, so an **E-mail Alert System** would come in handy for the members to save themselves from getting fined;
 - A **Book Rating** feature where members can rate the book only on the basis of how useful was it to them for the time they had it which in a way throws more light on what to expect from the book before renting it.
-

Chapter 7

REFERENCING AND APPENDICS

7.1. REFERENCES

1. <https://www.javatpoint.com/library-management-system-in-java-swing>
 2. <https://www.tutorialspoint.com/swing/index.htm>
 3. <https://www.edureka.co/blog/library-management-system-project-in-java>
 4. <https://www.sqlite.org/doclist.html>
 5. <https://www.youtube.com/watch?v=esIfjnbrJo8&list=PLKe9hJMFerTWfBOug40-2uSANxa8dtE>
-