

Index

SL	Topic	Page
1	Introduction	2
2	Motivation	3
3	Objectives	3-4
4	Technology and Tools	4
5	System Design	5-7
6	Implementation	7-12
7	Result	13-17
8	Limitation	18
9	Conclusion	18
10	References	19

Introduction:

This software project is a library management software system with all the basic as well as some innovative features for managing a library. It consists of a large database of various books available in the library. It also lists various books issued to respective readers. The system keeps track of all the books readily available and also the books that have been issued to various readers for the time period for which the books have been issued. The system also handles books database. An ILS usually comprises a relational database, software to interact with that database, and two graphical user interfaces (one for patrons, one for staff). Most ILS es separate software functions into discrete programs called modules, each of them integrated with a unified interface. Thus, this innovative library management system provides enhanced library functionality for this modern world.

Prior to computerization, library tasks were performed manually and independently from one another. Selectors ordered materials with ordering slips, cataloguers manually catalogued sources and indexed them with the card catalog system (in which all bibliographic data was kept on a single index card), fines were collected by local bailiffs, and users signed books out manually, indicating their name on clue cards which were then kept at the circulation desk. Early mechanization came in 1936, when the University of Texas began using a punch card system to manage library circulation. While the punch card system allowed for more efficient tracking of loans, library services were far from being integrated, and no other library task was affected by this change.

In our project, we have designed a well relational database to store books and student information. They can issue books from this application. It also works for online and offline as per as need our environment.

Motivation

Technology is always changing, and techies often pride themselves on keeping up with the times. If they're stuck for months maintaining old, broken code written long ago on older technologies, they'll become frustrated. It might be time to revamp that software and modernize. In our country there are many libraries but most of libraries do not use any software to manage books and issue. So that the lost books from library and it's an unmanaged system. On the other side when they used manually book keeping system it's waste lots of time. It also a boring system to work long time.

In this case we decided to make an application to face this situation and protest this boring working process.

Feasibility Study

The existing system is clearly understood the next step is to conduct the feasibility study, which is a high-level version of the entire System Analysis and Design process. The objective is to determine whether the proposed system is feasible. The three tests of feasibility have been carried out:

- Technical Feasibility
- Economic Feasibility
- Operational Feasibility

Objectives

A library management software where admin can add/view/delete librarian and librarian can add/view books, issue, view issued books and return books. This is Java Projects on Library System, which provided a lot of facility to their user. The objective and scope of my Project Library 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 Library 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, academics, students and applicants. Hence the management system for the College management has been designed to remove all the deficiency from which the present system is suffering and to ensure. Abstract of Library System 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 Library System is used to overcome the entire problem which they are facing currently, and making complete atomization of manual system to computerized system.

- Build and manage library collections in support of academic programs
- Provide timely access to requested materials.
- Simplify search/discovery of library resources.
- Build Digital Library infrastructure.
- Build the information literacy/library instruction program.
- Increase library outreach and marketing efforts.
- Undertake systematic review of reference services designed both to both adapt to changes in facilities and technology and to improve quality of service delivered.

Technology and Tools

Tools to be used

1. Use any IDE to develop the project. We have used intellij idea/ Netbeans.
2. MySQL for the database.

Front End and Back End

Front End: Java Swing

Back End: MySQL

System Design

Functional Requirements

1. Admin

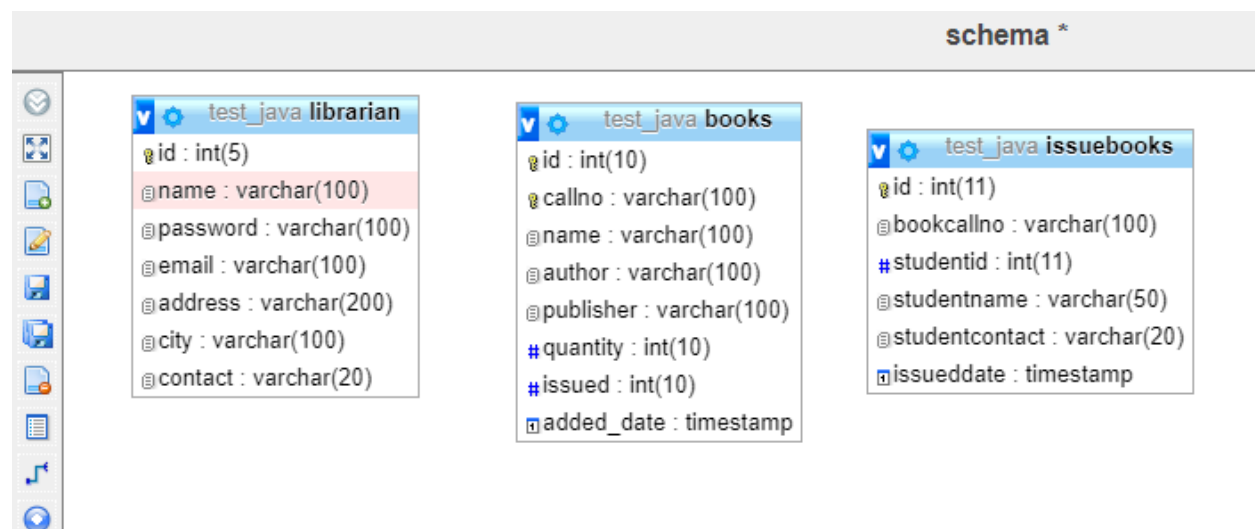
2. Can add/view/delete librarian
3. Can logout

2. Librarian

1. Can add/view books
2. Can issue books
3. View issued books
4. Return Books
5. Can logout

ER DESIGN

It is clear that the physical objects from the previous section – the customers, employees, cards, media, and library branches – correspond to entities in the Entity-Relationship model, and the operations to be done on those entities – holds, checkouts, and so on – correspond to relationships. However, a good design will minimize redundancy and attempt to store all the required information in as small a space as possible. After some consideration, we have decided on the following design:



1 books

Creation: Nov 27, 2018 at 08:50 AM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(10)		No		auto_increment			
callno	varchar(100)		No					
name	varchar(100)		No					
author	varchar(100)		No					
publisher	varchar(100)		No					
quantity	int(10)		No					
issued	int(10)		Yes	NULL				
added_date	timestamp		No	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP			

2 issuebooks

Creation: Nov 27, 2018 at 08:08 AM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(11)		No		auto_increment			
bookcallno	varchar(100)		No					
studentid	int(11)		No					
studentname	varchar(50)		No					
studentcontact	varchar(20)		No					
issueddate	timestamp		No	CURRENT_TIMESTAMP	on update CURRENT_TIMESTAMP			

3 librarian

Creation: Nov 27, 2018 at 08:08 AM

Column	Type	Attributes	Null	Default	Extra	Links to	Comments	MIME
id	int(5)		No		auto_increment			
name	varchar(100)		No					
password	varchar(100)		No					
email	varchar(100)		No					
address	varchar(200)		No					
city	varchar(100)		No					
contact	varchar(20)		No					

Implementation

PHYSICAL DATABASE DESIGN

The next step was to create the physical database and input some sample data. In order to turn the relational design into a database, we ran the following script in

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `test_java`
--
-- Table structure for table `books`
--

CREATE TABLE `books` (
  `id` int(10) NOT NULL,
  `callno` varchar(100) NOT NULL,
  `name` varchar(100) NOT NULL,
  `author` varchar(100) NOT NULL,
  `publisher` varchar(100) NOT NULL,
  `quantity` int(10) NOT NULL,
  `issued` int(10) DEFAULT NULL,
  `added_date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```

-----

-- Table structure for table `issuebooks`
--

CREATE TABLE `issuebooks` (
  `id` int(11) NOT NULL,
  `bookcallno` varchar(50) NOT NULL,
  `studentid` int(11) NOT NULL,
  `studentname` varchar(50) NOT NULL,
  `studentcontact` varchar(20) NOT NULL,
  `issueddate` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `issuebooks`
--

-- Table structure for table `librarian`
--

CREATE TABLE `librarian` (
  `id` int(5) NOT NULL,
  `name` varchar(100) NOT NULL,
  `password` varchar(100) NOT NULL,
  `email` varchar(100) NOT NULL,
  `address` varchar(200) NOT NULL,
  `city` varchar(100) NOT NULL,
  `contact` varchar(20) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Indexes for dumped tables
-- Indexes for table `books`
--
ALTER TABLE `books`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `callno` (`callno`),
  ADD UNIQUE KEY `callno_2` (`callno`);

--

-- Indexes for table `issuebooks`
--
ALTER TABLE `issuebooks`
  ADD PRIMARY KEY (`id`);

--

-- Indexes for table `librarian`
--
ALTER TABLE `librarian`
  ADD PRIMARY KEY (`id`);

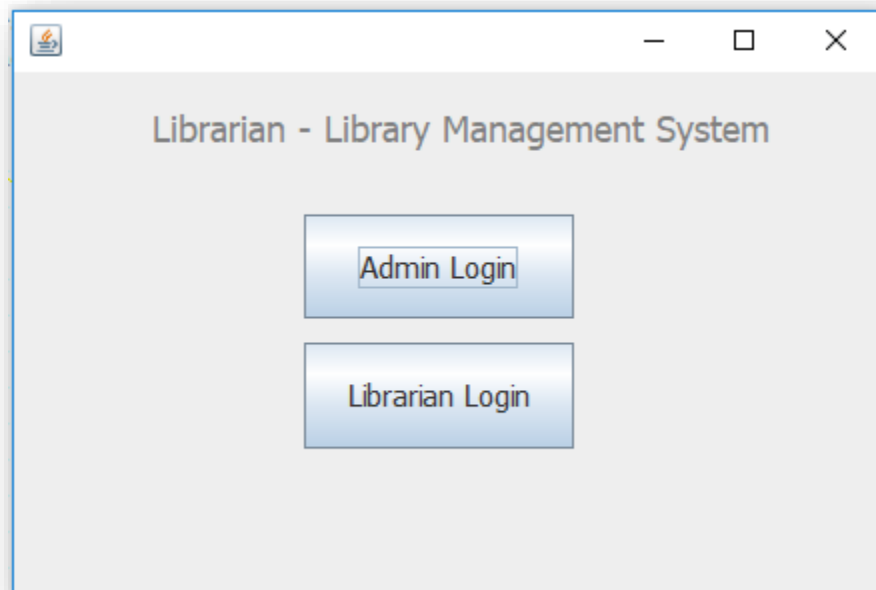
```

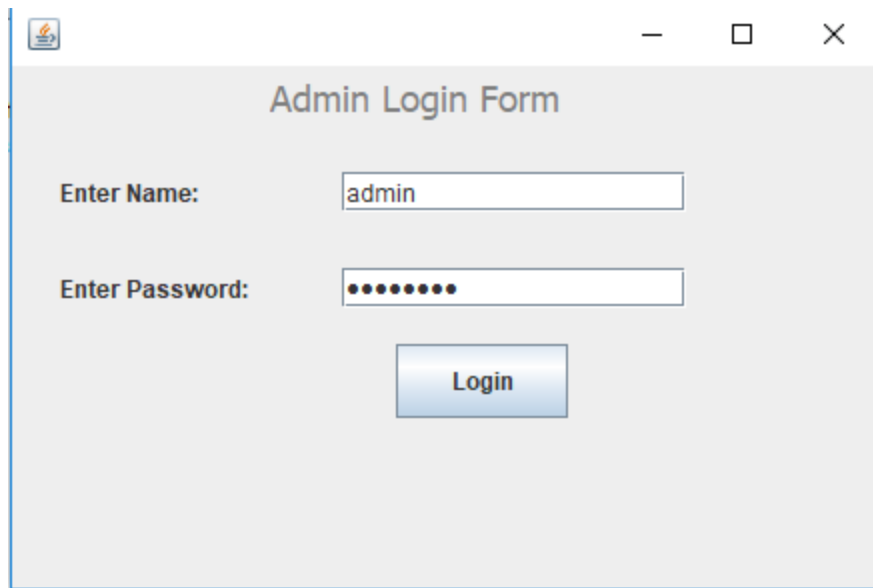


```
--
-- AUTO_INCREMENT for dumped tables
--
--
-- AUTO_INCREMENT for table `books`
--
ALTER TABLE `books`
  MODIFY `id` int(10) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=5;
--
-- AUTO_INCREMENT for table `issuebooks`
--
ALTER TABLE `issuebooks`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;
--
-- AUTO_INCREMENT for table `librarian`
```

GUI DESIGN

The first step in designing the GUI was to choose a means of accessing the database. After evaluating various options, we settled on using the JDBC API. By using JDBC we could separate the application logic from the DBMS as well as from clients. In addition to simplifying operations on the database, this architecture makes extending the functionality of our system easier. When adding a new feature or improving an existing one, we will not need to change the database; it will only be necessary to modify the Java portion of the code.





Create database connection used following code:

```
Class.forName("com.mysql.jdbc.Driver");  
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/test_java","","");
```

To insert data into database we have used these lines of codes

```
Connection con=DB.getConnection();  
  
status=updatebook(bookcallno);//updating quantity and issue  
  
if(status>0){  
    PreparedStatement ps=con.prepareStatement("insert into  
issuebooks(bookcallno,studentid,studentname,studentcontact) values(?,?,?,?)");  
    ps.setString(1,bookcallno);  
    ps.setInt(2,studentid);  
    ps.setString(3,studentname);  
    ps.setString(4,studentcontact);  
    status=ps.executeUpdate();  
}  
  
con.close();
```

To view books or retrieve data from database we have used these codes:

```
Connection con=DB.getConnection();  
  
    PreparedStatement ps=con.prepareStatement("select * from  
books",ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);  
  
    ResultSet rs=ps.executeQuery();
```

```

ResultSetMetaData rsmd=rs.getMetaData();

int cols=rsmd.getColumnCount();

column=new String[cols];

for(int i=1;i<=cols;i++){

    column[i-1]=rsmd.getColumnName(i);

}


rs.last();

int rows=rs.getRow();

rs.beforeFirst();


data=new String[rows][cols];

int count=0;

while(rs.next()){

    for(int i=1;i<=cols;i++){

        data[count][i-1]=rs.getString(i);

    }

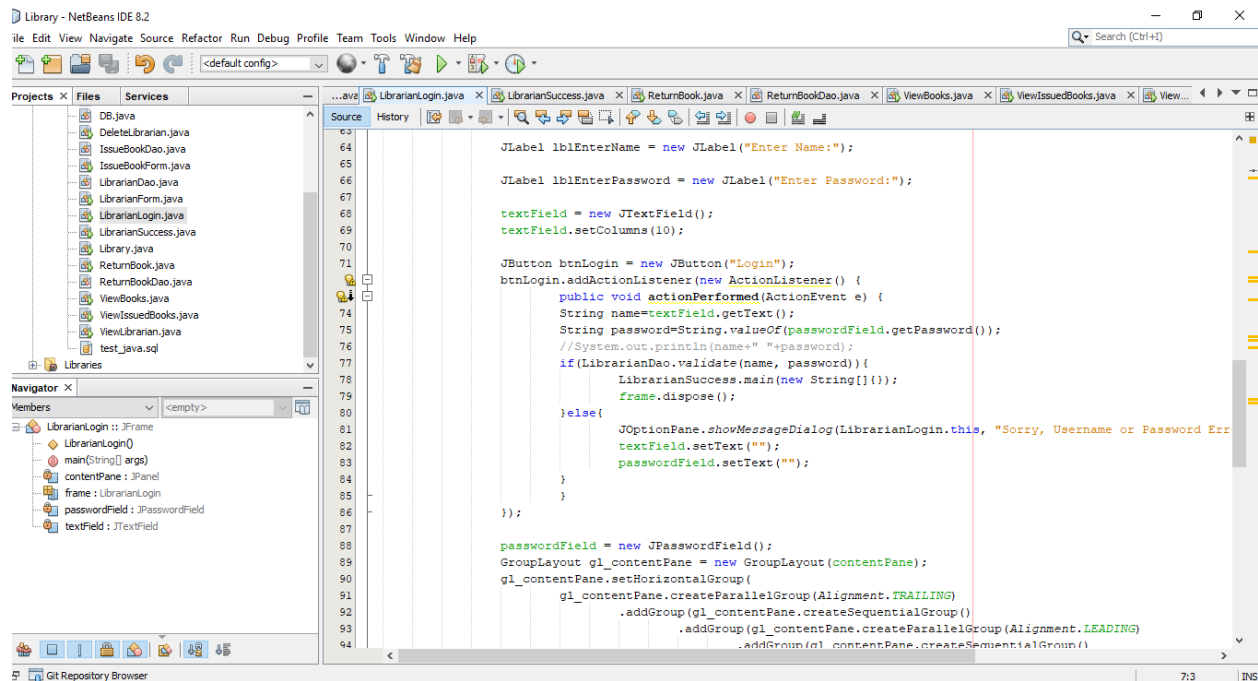
    count++;

}

con.close();

```

id	callno	name	author	publisher	quantity	issued	added_date
1	A@4	C In Depth	Shrivastav	BPB	2	2	2016-07-20 01:3...
2	B@1	DBMS	Korth	Pearson	3	0	2016-07-19 00:3...
3	G@12	Let's see	Yashwant Kanetkar	BPB	10	0	2016-07-19 05:0...
4	121	AAA	aa	aa	10	0	2018-11-27 08:5...
5	978	Mastering Java	Mosharrof Rubel	Odommo Prokash	10		2018-11-27 10:1...



To delete data from database:

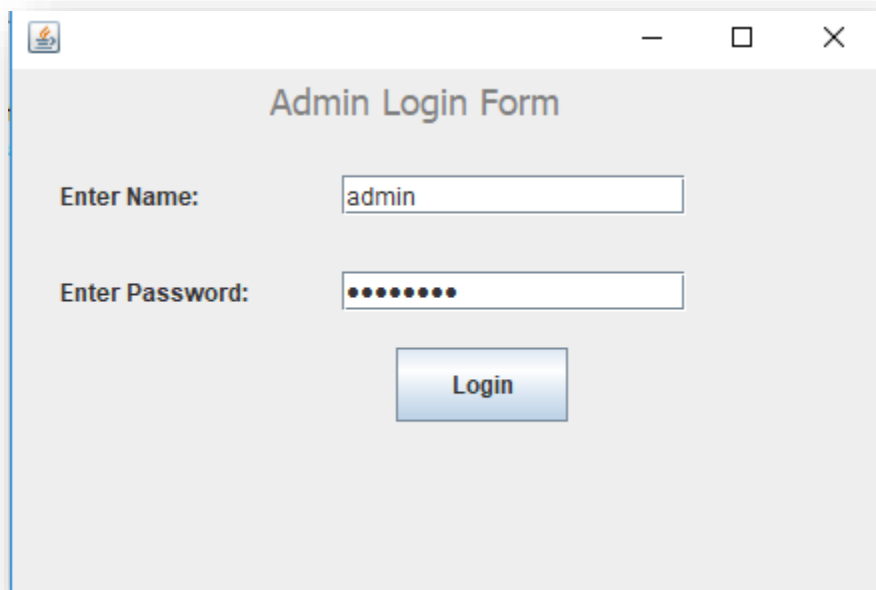
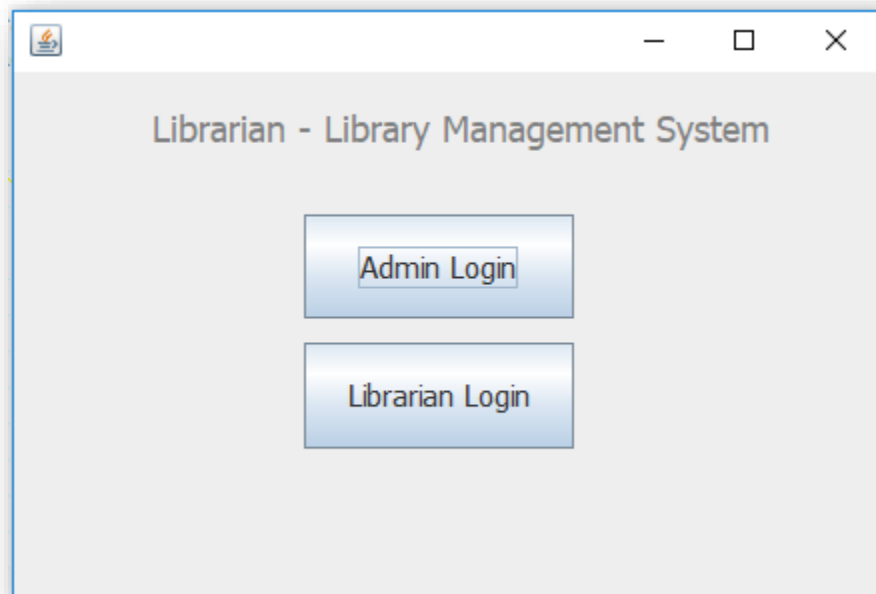
```

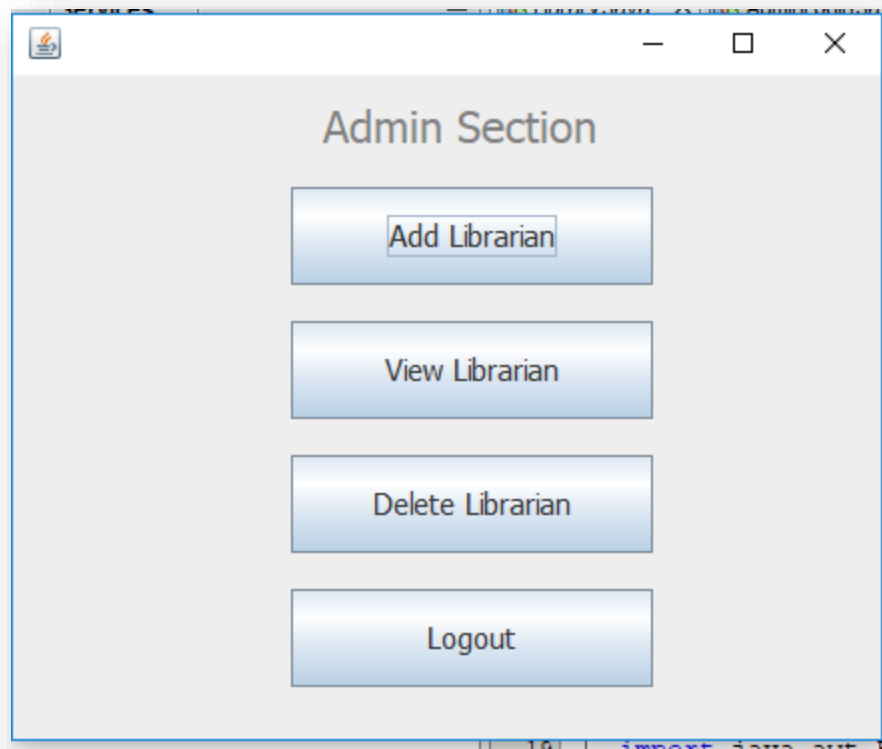
btnDelete.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        String sid=textField.getText();
        if(sid==null || sid.trim().equals("")){
            JOptionPane.showMessageDialog(DeleteLibrarian.this,"Id
can't be blank");
        }else{
            int id=Integer.parseInt(sid);
            int i=LibrarianDao.delete(id);
            if(i>0){
                JOptionPane.showMessageDialog(DeleteLibrarian.this,"Record deleted successfully!");
            }else{
                JOptionPane.showMessageDialog(DeleteLibrarian.this,"Unable to delete given id!");
            }
        }
    }
});

```

Results:

All of screen shots are following bellow





A screenshot of a web application window titled "Add Librarian". The window has a light gray background and a blue border. At the top, there is a title bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. Below the title bar, the text "Add Librarian" is centered. Underneath, there are six input fields with labels to their left: "Name:" (Ariful Islam), "Password:" (masked with dots), "Em..." (hello@ariful.net), "Address:" (Dhaka), "City:" (Dhaka), and "Contact No:" (01700000000000). Below the input fields, there are two blue buttons with white text: "Add Librarian" and "Back".

id	name	password	email	address	city	contact
1	Ariful	aaa	a@b.com	Dhaka	Dhaka	999832...
9	Ariful Isa...	123				
10	A	A	a	a	a	123
11	Ariful Isl...	123456	hello@a...	Dhaka	Dhaka	017000...

Librarian Section - Librarian

Add Books

View Books

Issue Book

View Issued Books

Return Book

Logout

Add Books

Call No:

Name:


Auth...

Publisher:

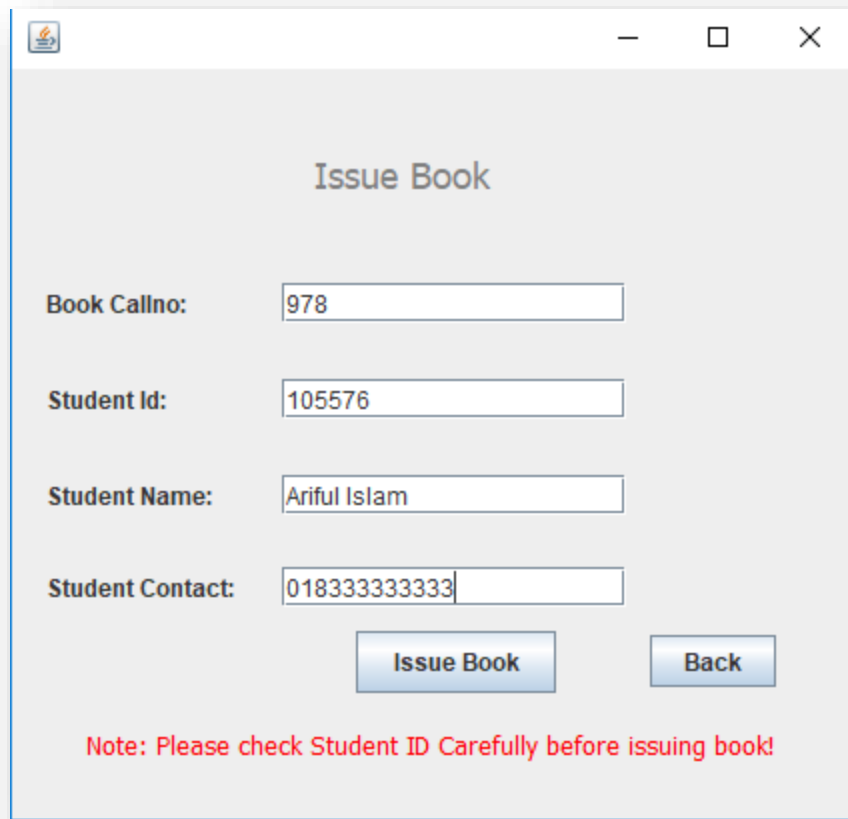
Quantity:

✕

Message

 **Books added successfully!**

id	callno	name	author	publisher	quantity	issued	added_date
1	A@4	C In Depth	Shrivastav	BPB	2	2	2016-07-20 01:3...
2	B@1	DBMS	Korth	Pearson	3	0	2016-07-19 00:3...
3	G@12	Let's see	Yashwant Kanetkar	BPB	10	0	2016-07-19 05:0...
4	121	AAA	aa	aa	10	0	2018-11-27 08:5...
5	978	Mastering Java	Mosharrof Rubel	Odommo Prokash	10		2018-11-27 10:1...



The 'Issue Book' window is a Java Swing application with a light gray background. It features a title bar with a standard icon, a minus button, a maximize button, and a close button. The main content area is titled 'Issue Book' in a bold, black font. Below the title, there are four input fields with labels to their left: 'Book Callno:' with the value '978', 'Student Id:' with the value '105576', 'Student Name:' with the value 'Ariful Islam', and 'Student Contact:' with the value '01833333333'. At the bottom of the form, there are two buttons: 'Issue Book' and 'Back'. Below the buttons, a red text note reads: 'Note: Please check Student ID Carefully before issuing book!'.

Issue Book

Book Callno: 978

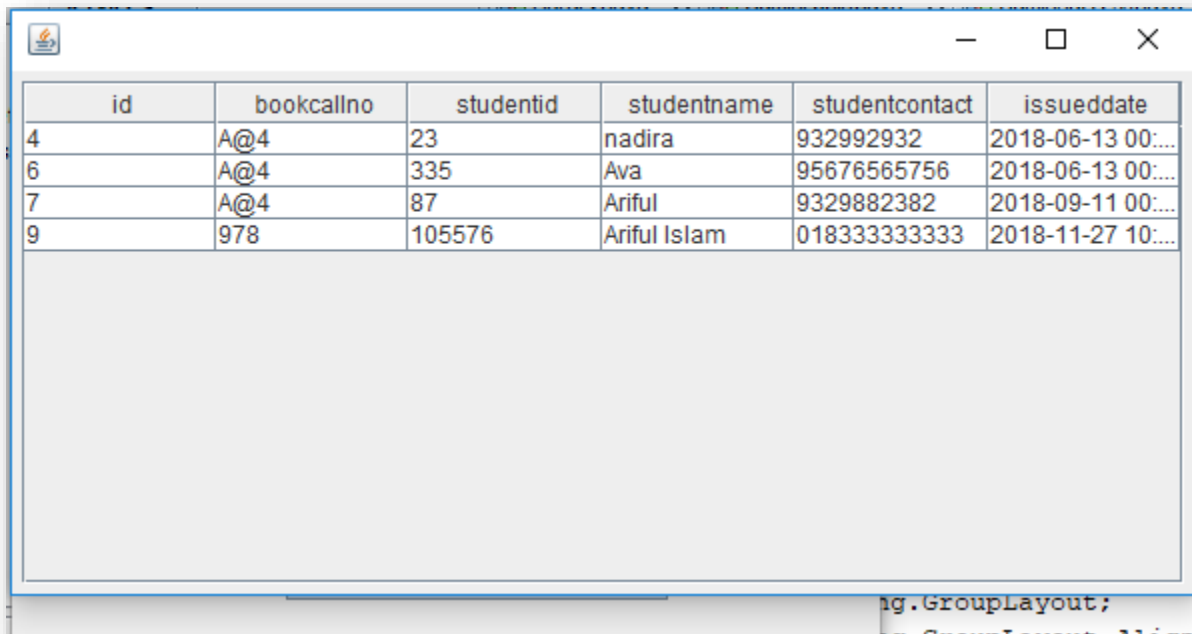
Student Id: 105576

Student Name: Ariful Islam

Student Contact: 01833333333

Issue Book Back

Note: Please check Student ID Carefully before issuing book!



The 'Table' window is a Java Swing application with a light gray background. It features a title bar with a standard icon, a minus button, a maximize button, and a close button. The main content area contains a table with 6 columns: 'id', 'bookcallno', 'studentid', 'studentname', 'studentcontact', and 'issuedate'. The table has 4 data rows. Below the table, there is a large, empty rectangular area.

id	bookcallno	studentid	studentname	studentcontact	issuedate
4	A@4	23	nadira	932992932	2018-06-13 00:...
6	A@4	335	Ava	95676565756	2018-06-13 00:...
7	A@4	87	Ariful	9329882382	2018-09-11 00:...
9	978	105576	Ariful Islam	01833333333	2018-11-27 10:...

Limitation:

- ❖ Difficulty in reports generating: Either no reports generating in a current system or they are generated with great difficulty reports take time to generate in the current system.
- ❖ Manual operator control: Manual operator control is there and leads to a lot of chaos and errors.
- ❖ Inability of sharing the data: Data cannot be shared in the existing system. This means that no two persons can use the same data in existing system. Also, the two departments in an organization cannot interact with each other without the actual movement of data.
- ❖ No data export system and import system.

Future implementation:

We will implement those limitations on upcoming version.

Conclusion

Our project is only a humble venture to satisfy the needs in a library. Several user-friendly coding has also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses. Last but not least it is not the work that played the ways to success but ALMIGHTY.

References

1. https://en.wikipedia.org/wiki/Integrated_library_system
2. https://www.annauniv.edu/academic_courses/docs/ugthesis.pdf
3. <https://www.javatpoint.com>
4. <https://docs.oracle.com>
5. <https://www.tutorialspoint.com/swing/>
6. <https://www.phpmyadmin.net/>
7. <https://www.codecademy.com/articles/sql-commands>

Download this project from GitHub

<https://github.com/prodhan/Java-Project-Librarian>