**Practical No. 2**

**Aim**: Design Library Management System using swing components

**Theory:**

The provided Java program is a simple implementation of a Library Management System using Java Swing components.

**Class Definition:**

The program defines a class called LibraryManagementSystem that extends the JFrame class. This makes the class a Swing-based graphical user interface.

**Instance Variables:**

The class includes various instance variables, such as studentNameField, classNameField, rollNumberField, bookNameField, authorField, issueDateField, returnDateField, addButton, bookListTextArea, and bookList.

These variables represent Swing components (text fields, button, text area) and data structures (ArrayList) used in the program.

**Constructor:**

The class has a constructor (LibraryManagementSystem) responsible for initializing the GUI components, setting up the layout, and making the frame visible.

**GUI Components:**

The GUI includes text fields for student name, class name, roll number, book name, author, issue date, and return date.

A text area (bookListTextArea) is used to display the list of books added.

A button (addButton) triggers the addition of a book to the library.

**ActionListener:**

An ActionListener is implemented for the addButton. When the button is clicked, the addBook method is called.

**Methods:**

The addBook method retrieves input from the text fields, performs basic validation, constructs a string containing book details (including student and class information), and adds the book details to the bookList.

The updateBookList method creates a string representation of the books in the bookList and updates the text area.

The clearInputFields method resets the text fields to empty after adding a book.

**ArrayList (bookList):**

The bookList variable is an ArrayList of strings that stores the details of books added to the library.

**Main Method:**

The main method creates an instance of the LibraryManagementSystem class, initializing the GUI and starting the application.

**Swing Components Layout:**

The layout manager (GridLayout) is used to organize the input fields and labels in a grid format for better visual presentation.

**User Interaction:**

Users can input student details, book details, and view the list of added books by clicking the "Add Book" button.

**Validation:**

Basic validation checks are implemented to ensure that essential fields are notemptybefore adding a book.

**Source Code:**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.util.ArrayList;

class LibraryManagementSystem extends JFrame {

JTextField studentNameField, classNameField, rollNumberField, bookNameField, authorField, issueDateField, returnDateField;

JButton addButton;

JTextArea bookListTextArea;

ArrayList<String> bookList;

LibraryManagementSystem() {

bookList = new ArrayList<>();

setLayout(new BorderLayout());

bookListTextArea = new JTextArea();

JScrollPane scrollPane = new JScrollPane(bookListTextArea);

add(scrollPane, BorderLayout.CENTER);

JPanel inputPanel = new JPanel(new GridLayout(8, 2, 10, 10));

studentNameField = new JTextField();

classNameField = new JTextField();

rollNumberField = new JTextField();

bookNameField = new JTextField();

authorField = new JTextField();

issueDateField = new JTextField();

returnDateField = new JTextField();

addButton = new JButton("Add Book");

inputPanel.add(new JLabel("Student Name:")); inputPanel.add(studentNameField);

inputPanel.add(new JLabel("Class Name:")); inputPanel.add(classNameField);

inputPanel.add(new JLabel("Roll Number:")); inputPanel.add(rollNumberField);

inputPanel.add(new JLabel("Book Name:")); inputPanel.add(bookNameField);

inputPanel.add(new JLabel("Author:")); inputPanel.add(authorField);

inputPanel.add(new JLabel("Issue Date:")); inputPanel.add(issueDateField);

inputPanel.add(new JLabel("Return Date:")); inputPanel.add(returnDateField);

inputPanel.add(new JLabel()); // Empty space for layout

inputPanel.add(addButton);

add(inputPanel, BorderLayout.SOUTH);

addButton.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

addBook();

}

});

setTitle("Library Management System");

setSize(600, 400);

setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

setVisible(true);

}

void addBook() {

String studentName = studentNameField.getText();

String className = classNameField.getText();

String rollNumber = rollNumberField.getText();

String bookName = bookNameField.getText();

String author = authorField.getText();

String issueDate = issueDateField.getText();

String returnDate = returnDateField.getText();

if (!studentName.isEmpty() && !className.isEmpty() && !rollNumber.isEmpty() &&

!bookName.isEmpty() && !author.isEmpty() && !issueDate.isEmpty() && !returnDate.isEmpty()) {

String bookDetails = "Student Name: " + studentName + ", Class Name: " + className +

", Roll Number: " + rollNumber + "\n" +

"Book Name: " + bookName + ", Author: " + author +

", Issue Date: " + issueDate + ", Return Date: " + returnDate;

bookList.add(bookDetails);

updateBookList();

clearInputFields();

} else {

JOptionPane.showMessageDialog(this, "Please fill in all fields.");

}

}

void updateBookList() {

StringBuilder stringBuilder = new StringBuilder();

for (String bookDetails : bookList) {

stringBuilder.append(bookDetails).append("\n\n");

}

bookListTextArea.setText(stringBuilder.toString());

}

void clearInputFields() {

studentNameField.setText("");

classNameField.setText("");

rollNumberField.setText("");

bookNameField.setText("");

authorField.setText("");

issueDateField.setText("");

returnDateField.setText("");

}

public static void main(String[] args) {

new LibraryManagementSystem();

}

}

**Output:**



