**Booklist.java**

package application;

import java.io.\*;

public class Booklist {

private int BOOKID;

private String BOOKNAME;

private int BOOKEDITION;

private String AUTHOR;

public Booklist(int BOOKID, String BOOKNAME, int BOOKEDITION, String AUTHOR) {

this.BOOKID = BOOKID;

this.BOOKNAME = BOOKNAME;

this.BOOKEDITION = BOOKEDITION;

this.AUTHOR = AUTHOR;

}

//getters and setters methods

public int getBOOKID() {

return BOOKID;

}

public void setBOOKID(int BOOKID) {

this.BOOKID = BOOKID;

}

public String getBOOKNAME() {

return BOOKNAME;

}

public void setBOOKNAME(String BOOKNAME) {

this.BOOKNAME = BOOKNAME;

}

public int getBOOKEDITION() {

return BOOKEDITION;

}

public void setBOOKEDITION(int BOOKEDITION) {

this.BOOKEDITION = BOOKEDITION;

}

public String getAUTHOR() {

return AUTHOR;

}

public void setAUTHOR(String AUTHOR) {

this.AUTHOR = AUTHOR;

}

public String toString() {

return "Booklist{" +

", BOOKID='" + BOOKID + '\'' +

", BOOKNAME='" + BOOKNAME + '\'' +

", BOOKEDITION='" + BOOKEDITION + '\'' +

", AUTHOR='" + AUTHOR +

'}';

}

}

**BookRegistration.java**

package application;

public class BookRegistration {

private String MEMBERID;

private int BOOKID;

public BookRegistration(String MEMBERID, int BOOKID) {

this.MEMBERID = MEMBERID;

this.BOOKID = BOOKID;

}

//get and set methods

public String getMEMBERID() {

return MEMBERID;

}

public void setMEMBERID(String MEMBERID) {

this.MEMBERID = MEMBERID;

}

public int getBOOKID() {

return BOOKID;

}

public void setBOOKID(int BOOKID) {

this.BOOKID = BOOKID;

}

public String toString() {

return "REGISTERLIBRARYBOOK{" +

", MEMBERID='" + MEMBERID + '\'' +

", BOOKID=" + BOOKID +

'}';

}

}

**LoginInformation.java**

package application;

public class LoginInformation {

private String libraryLoginUserName;

private String libraryLoginPassword;

public LoginInformation(String libraryLoginUserName, String libraryLoginPassword) {

this.libraryLoginUserName = libraryLoginUserName;

this.libraryLoginPassword = libraryLoginPassword;

}

public String getLibraryUserName() {

return libraryLoginUserName;

}

public String getLibraryPassword() {

return libraryLoginPassword;

}

}

**Main.java**

package application;

import java.io.IOException;

import javafx.application.Application;

import javafx.fxml.FXMLLoader;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.layout.\*;

import javafx.stage.Stage;

/\*\* Main application\*/

public class Main extends Application {

public static void main(String[] args) {

launch(args);

}

public void start(Stage primaryStage) throws IOException {

Parent loginWindow = FXMLLoader.load(getClass().getResource("/views/LoginWindow.fxml"));

Scene loginWindowScene = new Scene(loginWindow);

primaryStage.setTitle("LIBRARY SERVICES");

primaryStage.setScene(loginWindowScene);

primaryStage.show();

}

}

**Members.java**

package application;

public class Members {

private String MEMBERID;

private String libraryMemberName;

private String libraryMemberEmail;

private String libraryMemberContactNumber;

//to get member details

public Members(String MEMBERID, String libraryMemberName, String libraryMemberEmail, String libraryMemberContactNumber) {

this.setMEMBERID(MEMBERID);

this.setlibraryMemberName(libraryMemberName);

this.setlibraryMemberEmail(libraryMemberEmail);

this.setlibraryMemberContactNumber(libraryMemberContactNumber);

}

//get and set methods

public String getMEMBERID() {

return MEMBERID;

}

public void setMEMBERID(String MEMBERID) {

this.MEMBERID = MEMBERID;

}

public String getlibraryMemberName() {

return libraryMemberName;

}

public void setlibraryMemberName(String libraryMemberName) {

this.libraryMemberName = libraryMemberName;

}

public String getlibraryMemberEmail() {

return libraryMemberEmail;

}

public void setlibraryMemberEmail(String libraryMemberEmail) {

this.libraryMemberEmail = libraryMemberEmail;

}

public String getlibraryMemberContactNumber() {

return libraryMemberContactNumber;

}

public void setlibraryMemberContactNumber(String libraryMemberContactNumber) {

this.libraryMemberContactNumber = libraryMemberContactNumber;

}

public String toString() {

return "Members{" +

"memberName='" + libraryMemberName + '\'' +

", memberId='" + MEMBERID + '\'' +

", memberEmail='" + libraryMemberEmail + '\'' +

", memberContactNumber='" + libraryMemberContactNumber + '\'' +

'}';

}

}

**NewUsers.java**

package application;

public class NewUsers {

private String fullName;

private String userEmailID;

private String userName;

private String userPassword;

private String userType;

//method for getting user info

public NewUsers(String fullName, String userEmailID, String userName, String userPassword, String userType) {

this.fullName = fullName;

this.userEmailID = userEmailID;

this.userName = userName;

this.userPassword = userPassword;

this.userType = userType;

}

public String getFullName() {

return fullName;

}

public String getUserEmailID() {

return userEmailID;

}

public String getUserName() {

return userName;

}

public String getPassword() {

return userPassword;

}

public String getUserType() {

return userType;

}

}

**AdministrationWindowController.java**

package controllers;

import java.io.IOException;

import java.net.URL;

import java.sql.SQLException;

import java.util.ResourceBundle;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.fxml.Initializable;

import javafx.scene.Node;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.\*;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.layout.AnchorPane;

import javafx.scene.paint.Color;

import javafx.stage.Stage;

import models.\*;

import javax.swing.\*;

import application.Booklist;

public class AdministrationWindowController implements Initializable {

// Book Table

@FXML

private ObservableList <Booklist> libraryBooklist;

@FXML

private TableView <Booklist> libraryAvailableBooksTable;

@FXML

private TextField libraryBookID;

@FXML

private TextField libraryBookName;

@FXML

private TextField libraryBookPrice;

@FXML

private TextField bookAuthor;

@FXML

private Label bookAdditionStatus;

//initialization

public void initialize(URL location, ResourceBundle resources) {

try {

getAVailableBooks();

updateAvailableBooks();

} catch (SQLException sqlException) {

sqlException.printStackTrace();

}

}

//function to get the availability of Books from database

private void getAVailableBooks() throws SQLException{

libraryBooklist = FXCollections.observableArrayList();

DataBaseOperationBook bookOp = new DataBaseOperationBookImplementation();

libraryBooklist = bookOp.getAllBookList();

}

//function to update the table when a new Book is added

private void updateAvailableBooks(){

TableColumn<Booklist, Integer> libraryBookID = new TableColumn<>("BOOKID");

libraryBookID.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKID"));

TableColumn<Booklist, String> libraryBookName = new TableColumn<>("BOOKNAME");

libraryBookName.setCellValueFactory(new PropertyValueFactory<Booklist, String>("BOOKNAME"));

TableColumn<Booklist, Integer> libraryBookPrice = new TableColumn<>("BOOKEDITION");

libraryBookPrice.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKEDITION"));

TableColumn<Booklist, String> bookAuthor = new TableColumn<>("AUTHOR");

bookAuthor.setCellValueFactory(new PropertyValueFactory<Booklist, String>("AUTHOR"));

libraryAvailableBooksTable.setItems(libraryBooklist);

libraryAvailableBooksTable.getColumns().addAll(libraryBookID, libraryBookName, libraryBookPrice, bookAuthor);

libraryAvailableBooksTable.setVisible(true);

}

//function to add new books

@FXML

private void addNewBook(ActionEvent action) throws SQLException {

//check if all the fields are filled

if(!libraryBookID.getText().isEmpty() && !libraryBookName.getText().isEmpty() && !libraryBookPrice.getText().isEmpty() && !bookAuthor.getText().isEmpty()) {

int BookCode = Integer.parseInt(libraryBookID.getText());

String BookTitle = libraryBookName.getText();

int BookDuration = Integer.parseInt(libraryBookPrice.getText());

String BookProfessor = bookAuthor.getText();

Booklist newBook = new Booklist(BookCode,BookTitle,BookDuration,BookProfessor);

DataBaseOperationBook addBookDBOps = new DataBaseOperationBookImplementation();

//condition to check if the Book is already inserted

if(addBookDBOps.insertBook(newBook))

{

bookAdditionStatus.setText(BookCode + " Book inserted successfully");

}

else

{

bookAdditionStatus.setText(BookCode+ " Book already inserted/there was an error while inserting");

}

libraryBookID.clear();

libraryBookName.clear();

libraryBookPrice.clear();

bookAuthor.clear();

getAVailableBooks();

updateAvailableBooks();

}

else

bookAdditionStatus.setText("Please entery all the fields!!");

}

@FXML

public void adminWindowSignOut(ActionEvent action) throws IOException {

Parent loginWindow = FXMLLoader.load(getClass().getResource("/views/LoginWindow.fxml"));

Scene loginWindowScene = new Scene(loginWindow);

Stage loginMainWindow = (Stage)((Node)action.getSource()).getScene().getWindow();

loginMainWindow.setScene(loginWindowScene);

loginMainWindow.show();

}

}

**LoginWindowController.java**

package controllers;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.fxml.Initializable;

import javafx.scene.Node;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.ComboBox;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.RadioButton;

import javafx.scene.control.TextField;

import javafx.scene.layout.AnchorPane;

import javafx.stage.Stage;

import application.\*;

import models.\*;

import java.io.IOException;

import java.net.URL;

import java.sql.SQLException;

import java.util.ResourceBundle;

public class LoginWindowController implements Initializable{

//SignUp Window

@FXML

private TextField libraryFullNameSignUp;

@FXML

private PasswordField libraryPasswordSignUp;

@FXML

private TextField libraryEmailIdSignUp;

@FXML

private TextField libraryUserNameSignUp;

@FXML

private ComboBox <String> libraryUserTypesSignUp;

private ObservableList <String> libraryUserTypes;

@FXML

private Label libraryUserStatusSignUp;

//Login Window

@FXML

private TextField libraryUserNameLogin;

@FXML

private PasswordField libraryPasswordLogin;

@FXML

private Label libraryStatusLogin;

@FXML

private AnchorPane loginWindowPane;

@FXML

private AnchorPane signUpWindowPane;

private static String getUserName;

@Override

public void initialize(URL location, ResourceBundle resources) {

libraryUserTypes = FXCollections.observableArrayList("LIBRARY\_ADMIN","LIBRARY\_MEMBER");

libraryUserTypesSignUp.setItems(libraryUserTypes);

libraryUserTypesSignUp.getSelectionModel().selectLast();

libraryUserStatusSignUp.setText(null);

}

//function for sign in button

@FXML

private void librarySignUpUserButton(ActionEvent actionEvent) throws SQLException {

//get all the details from fxml window

String librarySignUpUserName = libraryUserNameSignUp.getText();

String librarySignUpPassword = libraryPasswordSignUp.getText();

String librarySignUpUserType = libraryUserTypesSignUp.getValue();

String librarySignUpFullName = libraryFullNameSignUp.getText();

String librarySignUpEmailID = libraryEmailIdSignUp.getText();

//check if all the required fields are entered

if (!librarySignUpFullName.isEmpty() && !librarySignUpUserName.isEmpty() && !librarySignUpPassword.isEmpty() && librarySignUpUserType != null){

//clear the sign up status

libraryUserStatusSignUp.setText(null);

//insert new user into the database

NewUsers libraryUser = new NewUsers(librarySignUpFullName, librarySignUpEmailID, librarySignUpUserName, librarySignUpPassword, librarySignUpUserType);

DataBaseOperationBookManagement libraryUserDBOps = new DataBaseOperationBookManagementImplementation();

//check if the user trying insert is already available

if (libraryUserDBOps.userExists(libraryUser)){

libraryUserStatusSignUp.setText("User "+librarySignUpFullName + " is already created, please login with user name and password");

return;

}

//if user is not available the insert new user

if (libraryUserDBOps.insertNewUser(libraryUser)){

libraryUserStatusSignUp.setText(librarySignUpFullName+" Sign Up successful!");

signUpClear();

}

//if there are any other errors while signing up

else{

libraryUserStatusSignUp.setText(librarySignUpFullName+" Failed to Sign Up");

signUpClear();

}

}

//if the required fields are empty

else{

libraryUserStatusSignUp.setText("Please fill all the required details!");

signUpClear();

}

}

//enable the login window pane

public void loginSignUpPaneWindow(ActionEvent actionEvent) {

loginWindowPane.setVisible(true);

signUpWindowPane.setVisible(false);//setting sign up window pane to null

}

//clear the sign up window pane fields

public void signUpClear() {

libraryUserNameSignUp.clear();

libraryPasswordSignUp.clear();

libraryUserTypesSignUp.getSelectionModel().selectLast();

libraryFullNameSignUp.clear();

libraryEmailIdSignUp.clear();

}

//handle user login button

@FXML

private void libraryUserLoginButton(ActionEvent actionEvent) throws SQLException {

//for getting the fxml feild values from login window pane

String libraryUserLoginName = libraryUserNameLogin.getText();

String libraryUserPassword = libraryPasswordLogin.getText();

//check if any fields are empty in the login window pane

if (!libraryUserLoginName.isEmpty() && !libraryUserPassword.isEmpty()){

LoginInformation userloginInfo = new LoginInformation(libraryUserLoginName, libraryUserPassword);

DataBaseOperationBookManagement libraryUserDBOps = new DataBaseOperationBookManagementImplementation();

//verify the entered credentials

if (libraryUserDBOps.verifyUserLoginCredential(userloginInfo)){

getUserName = userloginInfo.getLibraryUserName();

String userType = libraryUserDBOps.getLoginUserType(userloginInfo);

try {

Parent LibraryServiceDashboard;

//if signed in user is library student enable the student window

if (userType.equals("LIBRARY\_MEMBER")) {

LibraryServiceDashboard = FXMLLoader.load(getClass().getResource("/views/MemberWindow.fxml"));

}

//else enable the admin window

else {

LibraryServiceDashboard = FXMLLoader.load(getClass().getResource("/views/AdministrationWindow.fxml"));

}

Scene LibraryServiceDashboardScene = new Scene(LibraryServiceDashboard);

Stage window = (Stage)((Node)actionEvent.getSource()).getScene().getWindow();

window.setScene(LibraryServiceDashboardScene);

window.show();

} catch (IOException sqlException) {

sqlException.printStackTrace();

}

}

//invalid credentials

else{

libraryStatusLogin.setText("Entered Credentials are invalid");

loginClear();

}

}

//if the fields are empty

else{

libraryStatusLogin.setText("Please enter all the required fields!");

loginClear();

}

}

//enable the sign up window pane

@FXML

public void signUpLoginPageButton(ActionEvent action) {

loginWindowPane.setVisible(false);//disable the login window pane

signUpWindowPane.setVisible(true);

}

public void loginClear() {

libraryUserNameLogin.clear();

libraryPasswordLogin.clear();

}

//get the entered user name

public static String getUsername(){

return getUserName;

}

}

**MemberWindowController.java**

package controllers;

import java.io.IOException;

import java.net.URL;

import java.sql.SQLException;

import java.util.ResourceBundle;

import javax.swing.text.TabableView;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.fxml.FXML;

import javafx.fxml.FXMLLoader;

import javafx.fxml.Initializable;

import javafx.scene.Node;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.ComboBox;

import javafx.scene.control.Label;

import javafx.scene.control.PasswordField;

import javafx.scene.control.TableColumn;

import javafx.scene.control.TableView;

import javafx.scene.control.TextField;

import javafx.scene.control.cell.PropertyValueFactory;

import javafx.scene.layout.AnchorPane;

import javafx.scene.layout.BorderPane;

import javafx.stage.Stage;

import models.\*;

import application.\*;

import javafx.fxml.Initializable;

import controllers.\*;

public class MemberWindowController implements Initializable {

//Member fxml variables

@FXML

private Label libraryWelcomeMemberName;

@FXML

private Label libraryWelcomeMemberId;

@FXML

private AnchorPane libraryMemberWelcomeWindowPane;

@FXML

private AnchorPane libraryMemberInformationPane;

@FXML

private AnchorPane libraryMemberBookRegistrationPane;

@FXML

private Label libraryMemberName;

@FXML

private Label libraryMemberId;

@FXML

private Label libraryMemberBookInformation;

@FXML

private TextField libraryMemberEmailId;

@FXML

private TextField libraryMemberContactNumber;

@FXML

private Label libraryMemberUpdate;

//Book table

@FXML

private TableView<Booklist> libraryAvailableBooksTable;

private ObservableList <Booklist> availableBooklist;

@FXML

private Label libraryBookRegistrationStatus;

@FXML

private TableView<Booklist> libraryBookRegisteredTable;

private ObservableList <Booklist> registeredBooklist;

@FXML

private Label libraryBookRemovalStatus;

private Members libraryMember;

private String libraryBookMemberId;

//initializing the database connection

DataBaseOperationBookManagement MemberOp = null;

DataBaseOperationBook BookRegistrationOps = null;

DataBaseOperationBook BookOp = null;

//constructor

public MemberWindowController() {

MemberOp = new DataBaseOperationBookManagementImplementation();

BookRegistrationOps = new DataBaseOperationBookImplementation();

BookOp = new DataBaseOperationBookImplementation();

}

@Override

public void initialize(URL location, ResourceBundle resources) {

//set the Member window pane

libraryMemberWelcomeWindowPane.setVisible(true);

libraryMemberUpdate.setText(null);

libraryBookRegistrationStatus.setText(null);

libraryBookRemovalStatus.setText(null);

try{

//functions to perform Book registration operations

updateTheMemberInformationDetails();

getlibraryMemberInformationDetails();

getlibraryAvailableBookDetails();

updatelibraryAvaialableBookTable();

getlibraryMemberRegisteredBooks();

updatelibraryMemberRegisteredBooks();

} catch (SQLException sqlException){

sqlException.printStackTrace();

}

}

//method to get the Books registered by Member

private void getlibraryMemberRegisteredBooks() throws SQLException {

registeredBooklist = BookRegistrationOps.getRegisteredbooks(libraryMember);

}

//method to update the registered Books in Member table

private void updatelibraryMemberRegisteredBooks() {

//set the column values and populate values

TableColumn<Booklist, Integer> libraryBookCode = new TableColumn<>("Code");

libraryBookCode.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKID"));

TableColumn<Booklist, String> libraryBookTitle = new TableColumn<>("Title");

libraryBookTitle.setCellValueFactory(new PropertyValueFactory<Booklist, String>("BOOKNAME"));

TableColumn<Booklist, Integer> libraryBookDuration = new TableColumn<>("EDITION");

libraryBookDuration.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKEDITION"));

TableColumn<Booklist, String> libraryProfessor = new TableColumn<>("AUTHOR");

libraryProfessor.setCellValueFactory(new PropertyValueFactory<Booklist, String>("AUTHOR"));

//set the registered Book table

libraryBookRegisteredTable.setItems(registeredBooklist);

libraryBookRegisteredTable.getColumns().addAll(libraryBookCode, libraryBookTitle, libraryBookDuration, libraryProfessor);

libraryBookRegisteredTable.setVisible(true);

}

//function to get the available Books

void getlibraryAvailableBookDetails() throws SQLException{

availableBooklist = FXCollections.observableArrayList();

availableBooklist = BookOp.getAllBookList();

}

//function to update the available Book table

void updatelibraryAvaialableBookTable() {

//set the column names and populate values

TableColumn<Booklist, Integer> libraryBookCode = new TableColumn<>("Code");

libraryBookCode.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKID"));

TableColumn<Booklist, String> libraryBookTitle = new TableColumn<>("Title");

libraryBookTitle.setCellValueFactory(new PropertyValueFactory<Booklist, String>("BOOKNAME"));

TableColumn<Booklist, Integer> libraryBookDuration = new TableColumn<>("EDITION");

libraryBookDuration.setCellValueFactory(new PropertyValueFactory<Booklist, Integer>("BOOKEDITION"));

TableColumn<Booklist, String> libraryProfessor = new TableColumn<>("AUTHOR");

libraryProfessor.setCellValueFactory(new PropertyValueFactory<Booklist, String>("AUTHOR"));

//set the available Book table

libraryAvailableBooksTable.setItems(availableBooklist);

libraryAvailableBooksTable.getColumns().addAll(libraryBookCode, libraryBookTitle, libraryBookDuration, libraryProfessor);

libraryAvailableBooksTable.setVisible(true);

}

//method to get the Member details

private void updateTheMemberInformationDetails() throws SQLException {

libraryBookMemberId = LoginWindowController.getUsername();

libraryMember = MemberOp.getLibraryMember(libraryBookMemberId);

libraryWelcomeMemberName.setText(libraryMember.getlibraryMemberName());

libraryWelcomeMemberId.setText(libraryMember.getMEMBERID());

}

//method to update the Member information

private void getlibraryMemberInformationDetails() {

libraryMemberName.setText(libraryMember.getlibraryMemberName());

libraryMemberId.setText(libraryMember.getMEMBERID());

libraryMemberBookInformation.setText("Management Information System");

libraryMemberEmailId.setText(libraryMember.getlibraryMemberEmail());

libraryMemberContactNumber.setText(libraryMember.getlibraryMemberContactNumber());

}

//handle Member information button to enable Member information table

@FXML

private void libraryMemberInformationButton(ActionEvent action) {

libraryMemberBookRegistrationPane.setVisible(false);//disable the Member registered Book pane

libraryMemberWelcomeWindowPane.setVisible(false);//disable the Member welcome window pane

libraryMemberInformationPane.setVisible(true);

}

//handle Member information button to enable Book registration window table

@FXML

private void libraryMemberBookRegistrationButton(ActionEvent action) {

libraryMemberInformationPane.setVisible(false);//disable the Member information window pane

libraryMemberWelcomeWindowPane.setVisible(false);//disable the Member welcome window pane

libraryMemberBookRegistrationPane.setVisible(true);

}

//handle Member update button

@FXML

private void libraryMemberUpdateButton(ActionEvent action) {

String newEmailId = libraryMemberEmailId.getText();

String newContactNumber = libraryMemberContactNumber.getText();

boolean emailStatus ;

boolean contactNumberStatus;

//update Member information

emailStatus = MemberOp.updateLibraryMemberEmail(libraryBookMemberId, newEmailId);

if (emailStatus)

libraryMember.setlibraryMemberEmail(newEmailId);

contactNumberStatus = MemberOp.updateLibraryMemberContactNumber(libraryBookMemberId, newContactNumber);

if (contactNumberStatus)

libraryMember.setlibraryMemberContactNumber(newContactNumber);

if (emailStatus && contactNumberStatus) {

libraryMemberUpdate.setText("Information updated successfully.");

} else {

libraryMemberUpdate.setText("Error updating information. Please recheck!");

}

}

//method to register the Books from the available Books

@FXML

private void libraryBookRegisterButton(ActionEvent action) throws SQLException {

Booklist choosenBook = libraryAvailableBooksTable.getSelectionModel().getSelectedItem();

if (choosenBook == null){

libraryBookRegistrationStatus.setText("Select a Book first!");

return;

}

BookRegistration BookRegistration = new BookRegistration(libraryBookMemberId, choosenBook.getBOOKID());

//check if Book is already registered

boolean exists = BookRegistrationOps.bookExists(choosenBook, libraryMember);

if (exists){

libraryBookRegistrationStatus.setText("Already registered for " + choosenBook.getBOOKID());

return;

}

boolean regStatus = BookRegistrationOps.registerTobook(BookRegistration);

if (regStatus){

registeredBooklist.add(choosenBook);

libraryBookRegistrationStatus.setText("Registration for " + choosenBook.getBOOKID() + " successful.");

} else{

libraryBookRegistrationStatus.setText("Failed to add " + choosenBook.getBOOKID() + "!");

}

}

//function to remove the Books from the registered Books

@FXML

private void libraryBookRemovalButton(ActionEvent action) {

libraryBookRemovalStatus.setText(null);

Booklist choosenBook = libraryBookRegisteredTable.getSelectionModel().getSelectedItem();

if (choosenBook == null){

libraryBookRemovalStatus.setText("Select a Book first!");

return;

}

boolean removeStatus = BookRegistrationOps.removeRegsiteredbook(choosenBook, libraryMember);

if (removeStatus){

registeredBooklist.remove(choosenBook);

libraryBookRemovalStatus.setText("Removed " + choosenBook.getBOOKID() + " successfully.");

} else{

libraryBookRemovalStatus.setText("Failed to remove " + choosenBook.getBOOKID() + "!");

}

}

//handle sign out button

@FXML

public void MemberWinodwSignOut(ActionEvent action) throws IOException {

Parent loginWindow = FXMLLoader.load(getClass().getResource("/views/LoginWindow.fxml"));

Scene loginWindowScene = new Scene(loginWindow);

Stage loginMainWindow = (Stage)((Node)action.getSource()).getScene().getWindow();

loginMainWindow.setScene(loginWindowScene);

loginMainWindow.show();

}

}

**DataBaseOperationBook.java**

package models;

import javafx.collections.ObservableList;

import application.Booklist;

import application.BookRegistration;

import application.Members;

import java.sql.ResultSet;

import java.sql.SQLException;

public interface DataBaseOperationBook {

//function to get book details

ObservableList <Booklist> getAllBookList() throws SQLException;

Booklist getBookDetail(int library\_bookNumber) throws SQLException;

//function to register to book

boolean registerTobook(BookRegistration registration);

boolean bookExists(Booklist librarybook, Members libraryStudent) throws SQLException;

//function to remove the book from the registered books

boolean removeRegsiteredbook(Booklist librarybook, Members libraryStudent);

ObservableList <Booklist> getRegisteredbooks(Members libraryStudent) throws SQLException;

ObservableList <Members> getRegisteredMembers(Booklist librarybook) throws SQLException;

//function to insert a new book

boolean insertBook(Booklist newbook);

}

**DataBaseOperationBookImplementation.java**

package models;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import application.BookRegistration;

import application.Booklist;

import application.Members;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

public class DataBaseOperationBookImplementation implements DataBaseOperationBook {

//initializing database connection

DBConnect conn = null;

Statement stmt = null;

PreparedStatement prepareStmt = null;

//constructor

public DataBaseOperationBookImplementation() {

conn = new DBConnect();

}

//calling function to call the list of books

@Override

public ObservableList <Booklist> getAllBookList() throws SQLException {

stmt = conn.connect().createStatement();

String getbookDetail = String.format("SELECT \* FROM BOOKREGISTRY");

ResultSet outputResult = stmt.executeQuery(getbookDetail);

//get all the available books

ObservableList <Booklist> listOfbooks = FXCollections.observableArrayList();

while(outputResult.next()){

int librarybookCode = outputResult.getInt("BOOKID");

String BOOKNAME = outputResult.getString("BOOKNAME");

int BOOKEDITION = outputResult.getInt("BOOKEDITION");

String libraryAUTHOR = outputResult.getString("AUTHOR");

Booklist newbook = new Booklist(librarybookCode, BOOKNAME, BOOKEDITION, libraryAUTHOR);

listOfbooks.add(newbook);

}

return listOfbooks;

}

//calling function to get the book details

@Override

public Booklist getBookDetail(int BOOKID) throws SQLException {

//sql statement to get the book details

String getbookDetail = String.format("SELECT \* FROM BOOKREGISTRY WHERE BOOKID=%d",

BOOKID);

stmt = conn.connect().createStatement();

//execution of sql statement

ResultSet outputResult = stmt.executeQuery(getbookDetail);

//checking the result

if(outputResult.next()){

String BOOKNAME = outputResult.getString("BOOKNAME");

int BOOKEDITION = outputResult.getInt("BOOKEDITION");

String AUTHOR = outputResult.getString("AUTHOR");

return new Booklist(BOOKID, BOOKNAME, BOOKEDITION, AUTHOR);

}

return null;

}

//calling function to register book

@Override

public boolean registerTobook(BookRegistration registration) {

//insert books to the Member id

String insertIntoMemberRegistration = String.format("INSERT INTO BORROWEDBOOKS VALUES('%s', %d)",

registration.getMEMBERID(),

registration.getBOOKID());

try{

//executing sql statement

stmt = conn.connect().createStatement();

stmt.executeUpdate(insertIntoMemberRegistration);

} catch (SQLException sqlException){

sqlException.printStackTrace();

return false;

}

return true;

}

//function to check the existence of book

@Override

public boolean bookExists(Booklist LIBRARYBOOK, Members libraryMemberId) throws SQLException {

//sql statement to get the book details to check existence

String getBookList = String.format("SELECT BOOKREGISTRY.BOOKID\n" +

"FROM BOOKREGISTRY, BORROWBOOKS\n" +

"WHERE BORROWBOOKS.MEMBERID='%s' "

+ "AND BOOKREGISTRY.BOOKID='%s' "

+ "AND BOOKREGISTRY.BOOKID=BOOKREGISTRY.BOOKID",

libraryMemberId.getMEMBERID(),

LIBRARYBOOK.getBOOKID());

stmt = conn.connect().createStatement();

//execution of sql statement

ResultSet outputResult = stmt.executeQuery(getBookList);

return outputResult.next();

}

//function to remove the registered books

@Override

public boolean removeRegsiteredbook(Booklist registeredbook, Members registeredMember) {

//sql statement to remove the book

String removeRegisteredbook = String.format("DELETE FROM BORROWEDBOOKS WHERE MEMBERID='%s' AND BOOKID=%d",

registeredMember.getMEMBERID(),

registeredbook.getBOOKID());

try{

stmt = conn.connect().createStatement();

stmt.executeUpdate(removeRegisteredbook);

return true;

} catch (SQLException sqlException) {

sqlException.printStackTrace();

}

return false;

}

//calling function to get the list of registered books by Member

@Override

public ObservableList<Booklist> getRegisteredbooks(Members libraryMember) throws SQLException {

stmt = conn.connect().createStatement();

//sql statement to get books registered by Member

String getRegisteredbook = String.format("SELECT BOOKREGISTRY.BOOKID, "

+ "BOOKREGISTRY.BOOKNAME, "

+ "BOOKREGISTRY.BOOKEDITION, "

+ "BOOKREGISTRY.AUTHOR "

+ "FROM BOOKREGISTRY, BORROWEDBOOKS "

+ "WHERE BORROWEDBOOKS.MEMBERID='%s' "

+ "AND BORROWEDBOOKS.BOOKID=BOOKREGISTRY.BOOKID",

libraryMember.getMEMBERID());

ResultSet outputResult = stmt.executeQuery(getRegisteredbook);

ObservableList <Booklist> BookList = FXCollections.observableArrayList();

while(outputResult.next()){

int bookCode = outputResult.getInt("BOOKID");

String bookTitle = outputResult.getString("BOOKNAME");

int bookDuration = outputResult.getInt("BOOKEDITION");

String AUTHOR = outputResult.getString("AUTHOR");

Booklist book = new Booklist(bookCode, bookTitle, bookDuration, AUTHOR);

BookList.add(book);

}

return BookList;

}

//function to get the list of Members registered to book

@Override

public ObservableList <Members> getRegisteredMembers(Booklist LIBRARYBOOK) throws SQLException {

String query = String.format("SELECT LIBRARY\_MEMBER.MEMBERID, "

+ "LIBRARY\_MEMBER.MEMBERNAME, "

+ "LIBRARY\_MEMBER.MEMBEREMAIL, "

+ "LIBRARY\_MEMBER.MEMBERMOBILE "

+ "FROM BORROWEDBOOKS, LIBRARY\_MEMBER "

+ "WHERE BORROWEDBOOKS.BOOKID=%d "

+ "AND LIBRARY\_MEMBER.MEMBERID=BORROWEDBOOKS.MEMBERID",

LIBRARYBOOK.getBOOKID());

stmt = conn.connect().createStatement();

ResultSet outputResult = stmt.executeQuery(query);

ObservableList <Members> MemberList = FXCollections.observableArrayList();

while(outputResult.next()){

String MemberId = outputResult.getString("MEMBERID");

String MemberName = outputResult.getString("MEMBERNAME");

String MemberEmail = outputResult.getString("MEMBEREMAIL");

String MemberContactNumber = outputResult.getString("MEMBERMOBILE");

MemberList.add(new Members(MemberId, MemberName, MemberEmail, MemberContactNumber));

}

return MemberList;

}

//function to insert new book

@Override

public boolean insertBook(Booklist newbook) {

boolean queryResult = false;

try {

//sql prepared statements to insert into new book

String insertNewBook = "INSERT INTO BOOKREGISTRY(BOOKID,BOOKNAME,BOOKEDITION,AUTHOR) " + "VALUES (?,?,?,?)";

prepareStmt = conn.connect().prepareStatement(insertNewBook);

prepareStmt.setInt(1,newbook.getBOOKID());

prepareStmt.setString(2, newbook.getBOOKNAME());

prepareStmt.setInt(3, newbook.getBOOKEDITION());

prepareStmt.setString(4, newbook.getAUTHOR());

String getbookDetails = String.format("SELECT BOOKID FROM BOOKREGISTRY WHERE BOOKID = '%s'", newbook.getBOOKID());

stmt = conn.connect().createStatement();

ResultSet outputResult = stmt.executeQuery(getbookDetails);

if(outputResult.next()) {

queryResult = false;

}

else {

int insertRecordStatus = prepareStmt.executeUpdate();

if(insertRecordStatus>0)

queryResult = true;

else

queryResult = false;

}

}

catch (SQLException sqlException) {

sqlException.printStackTrace();

return false;

}

if(queryResult)

{

return true;

}

else

{

return false;

}

}

}

**DataBaseOperationBookManagement.java**

package models;

import java.sql.SQLException;

import application.LoginInformation;

import application.Members;

import application.NewUsers;

public interface DataBaseOperationBookManagement {

//new user verification and sign up

boolean insertNewUser(NewUsers newUser);

boolean verifyUserLoginCredential(LoginInformation loginInfo);

String getLoginUserType(LoginInformation loginInfo) throws SQLException;

boolean userExists(NewUsers newUser) throws SQLException;

//Member database update

boolean updateLibraryMemberEmail(String libraryMemberId, String libraryMemberEmailId);

boolean updateLibraryMemberContactNumber(String libraryMemberId, String libraryMemberContactNumber);

Members getLibraryMember(String libraryMemberId) throws SQLException;

}

**DataBaseOperationBookManagementImplementation.java**

package models;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import application.LoginInformation;

import application.Members;

import java.sql.PreparedStatement;

import application.NewUsers;

import models.DataBaseOperationBookManagement;

public class DataBaseOperationBookManagementImplementation implements DataBaseOperationBookManagement {

//initializing the database connection

DBConnect conn = null;

Statement stmt = null;

PreparedStatement prepareStmt = null;

//constructor

public DataBaseOperationBookManagementImplementation() {

conn = new DBConnect();

}

//calling the method to insert new user

*@Override*

public boolean insertNewUser(NewUsers newUser) {

boolean queryResult = false;

try {

//prepared statement to insert new users

String insertUserTypeQuery = "INSERT INTO LIBRARYUSERLOGIN(USERNAME,PASSWORD,USERTYPE) "+ "VALUES (?,?,?)";

prepareStmt = conn.connect().prepareStatement(insertUserTypeQuery);

prepareStmt.setString(1,newUser.getUserName());

prepareStmt.setString(2, newUser.getPassword());

prepareStmt.setString(3, newUser.getUserType());

String roleName = newUser.getUserType();

int insertUserTypeStatus = prepareStmt.executeUpdate();

if(insertUserTypeStatus > 0) {

queryResult = true;

}

else {

queryResult = false;

}

stmt = conn.connect().createStatement();

//prepared statements

String insertQuery = "INSERT INTO " +roleName+"(MEMBERID,MEMBERNAME,MEMBEREMAIL) " + "VALUES (?, ?, ?)";

prepareStmt = conn.connect().prepareStatement(insertQuery);

prepareStmt.setString(1,newUser.getUserName());

prepareStmt.setString(2, newUser.getFullName());

prepareStmt.setString(3, newUser.getUserEmailID());

int insertRecordStatus = prepareStmt.executeUpdate();

if(insertRecordStatus > 0) {

queryResult = true;

}

else {

queryResult = false;

}

}

catch (SQLException sqlException) {

sqlException.printStackTrace();

return false;

}

return queryResult;

}

//calling method to verify user credentials

*@Override*

public boolean verifyUserLoginCredential(LoginInformation loginInfo) {

//sql statment to get user name and password from the database

String veryUserLogin = String.*format*("SELECT \* FROM LIBRARYUSERLOGIN WHERE USERNAME='%s' AND PASSWORD='%s'",

loginInfo.getLibraryUserName(),

loginInfo.getLibraryPassword());

try {

//verify the user login

stmt = conn.connect().createStatement();

ResultSet outputResult = stmt.executeQuery(veryUserLogin);

if (outputResult.next()){

return true;

}

} catch (SQLException sqlException) {

sqlException.printStackTrace();

}

return false;

}

//method to get the type of user who has logged in

*@Override*

public String getLoginUserType(LoginInformation loginInfo) throws SQLException {

//sql statement to get the user name

String getUserType = String.*format*("SELECT USERTYPE FROM LIBRARYUSERLOGIN WHERE USERNAME='%s'",

loginInfo.getLibraryUserName());

stmt = conn.connect().createStatement();

ResultSet outputResult = stmt.executeQuery(getUserType);

if (outputResult.next()){

return outputResult.getString("USERTYPE");

}

return null;

}

//method to check the user existence

*@Override*

public boolean userExists(NewUsers newUser) throws SQLException {

//sql query to get the user login details

String query = String.*format*("SELECT USERNAME FROM LIBRARYUSERLOGIN WHERE USERNAME='%s' AND USERTYPE='%s'",newUser.getUserName(),newUser.getUserType());

stmt = conn.connect().createStatement();

ResultSet outputResult = stmt.executeQuery(query);

return (outputResult.next());

}

//method to update Member email

*@Override*

public boolean updateLibraryMemberEmail(String LibraryMemberId, String LibraryMemberEmailId) {

//sql query to update the email id

String updateEmailId = String.*format*("UPDATE LIBRARY\_MEMBER SET MEMBEREMAIL='%s' WHERE MEMBERID='%s'",

LibraryMemberEmailId,

LibraryMemberId);

try{

//execution of sql statement

stmt = conn.connect().createStatement();

stmt.executeUpdate(updateEmailId);

return true;

} catch (SQLException sqlException){

sqlException.printStackTrace();

}

return false;

}

//method to update contact information

*@Override*

public boolean updateLibraryMemberContactNumber(String LibraryMemberId, String LibraryMemberContactNumber) {

//sql statement to update contact information

String updateContactNumber = String.*format*("UPDATE LIBRARY\_MEMBER SET MEMBERMOBILE='%s' WHERE MEMBERID='%s'",

LibraryMemberContactNumber,

LibraryMemberId);

try{

//command to execute sql statements

stmt = conn.connect().createStatement();

stmt.executeUpdate(updateContactNumber);

return true;

} catch (SQLException sqlException){

sqlException.printStackTrace();

}

return false;

}

//method to get Member details

*@Override*

public Members getLibraryMember(String LibraryMemberId) throws SQLException {

//sql query to get Member details

stmt = conn.connect().createStatement();

//stmt.executeUpdate(sql);

String getMemberInfo = String.*format*("SELECT \* FROM LIBRARY\_MEMBER WHERE MEMBERID='%s'", LibraryMemberId);

//command to execute sql command

ResultSet outputResult = stmt.executeQuery(getMemberInfo);

if (!outputResult.next())

return null;

String LibraryMemberName = outputResult.getString("MEMBERNAME");

String LibraryMemberEmail = outputResult.getString("MEMBEREMAIL");

String LibraryMemberContactNumber = outputResult.getString("MEMBERMOBILE");

return new Members(LibraryMemberId, LibraryMemberName, LibraryMemberEmail, LibraryMemberContactNumber);

}

}

**DBConnect.java**

package models;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnect {

// Code database URL

static final String ***DB\_URL*** = "jdbc:mysql://www.papademas.net:3307/510fp?autoReconnect=true&useSSL=false";

// Database credentials

static final String ***USER*** = "fp510", ***PASS*** = "510";

public Connection connect() throws SQLException {

return DriverManager.*getConnection*(***DB\_URL***, ***USER***, ***PASS***);

}

}

**AdministrationWindow.fxml**

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<?**import** javafx.scene.control.Button?>

<?**import** javafx.scene.control.Label?>

<?**import** javafx.scene.control.Tab?>

<?**import** javafx.scene.control.TabPane?>

<?**import** javafx.scene.control.TableView?>

<?**import** javafx.scene.control.TextField?>

<?**import** javafx.scene.image.Image?>

<?**import** javafx.scene.image.ImageView?>

<?**import** javafx.scene.layout.AnchorPane?>

<?**import** javafx.scene.layout.BorderPane?>

<?**import** javafx.scene.layout.ColumnConstraints?>

<?**import** javafx.scene.layout.GridPane?>

<?**import** javafx.scene.layout.RowConstraints?>

<?**import** javafx.scene.text.Font?>

<?**import** javafx.scene.text.Text?>

<**BorderPane** prefHeight=*"507.0"* prefWidth=*"917.0"* style=*"-fx-background-color: #088080;"* xmlns=*"http://javafx.com/javafx/8.0.241"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"controllers.AdministrationWindowController"*>

<**left**>

<**AnchorPane** prefHeight=*"509.0"* prefWidth=*"331.0"* style=*"-fx-background-color: #088080;"* BorderPane.alignment=*"CENTER"*>

<**children**>

<**ImageView** fitHeight=*"318.0"* fitWidth=*"406.0"* pickOnBounds=*"true"* preserveRatio=*"true"*>

</**ImageView**>

<**Text** fill=*"#f2f2f2"* layoutX=*"89.0"* layoutY=*"239.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"Administration Window"* textAlignment=*"CENTER"* wrappingWidth=*"228.05992126464844"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"32.0"* />

</**font**>

</**Text**>

</**children**>

</**AnchorPane**>

</**left**>

<**right**>

<**AnchorPane** prefHeight=*"557.0"* prefWidth=*"441.0"* BorderPane.alignment=*"CENTER"*>

<**children**>

<**TabPane** minHeight=*"400.0"* minWidth=*"500.0"* prefHeight=*"200.0"* prefWidth=*"200.0"* tabClosingPolicy=*"UNAVAILABLE"*>

<**tabs**>

<**Tab** text=*"Available Book"*>

<**content**>

<**AnchorPane** minHeight=*"0.0"* minWidth=*"0.0"* prefHeight=*"399.0"* prefWidth=*"500.0"*>

<**children**>

<**TableView** fx:id=*"libraryAvailableBooksTable"* layoutX=*"-4.0"* layoutY=*"88.0"* prefHeight=*"459.0"* prefWidth=*"503.0"* />

<**Text** fill=*"#f8f7f7"* layoutX=*"67.0"* layoutY=*"52.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"LIST OF AVAILABLE BOOKS"* textAlignment=*"CENTER"* wrappingWidth=*"381.4700241088867"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"26.0"* />

</**font**>

</**Text**>

</**children**>

</**AnchorPane**>

</**content**>

</**Tab**>

<**Tab** text=*"Add Books"*>

<**content**>

<**AnchorPane** minHeight=*"0.0"* minWidth=*"0.0"* prefHeight=*"180.0"* prefWidth=*"200.0"*>

<**children**>

<**GridPane** layoutX=*"35.0"* layoutY=*"113.0"* prefHeight=*"163.0"* prefWidth=*"394.0"*>

<**columnConstraints**>

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"194.66668701171875"* minWidth=*"10.0"* prefWidth=*"194.66668701171875"* />

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"226.33331298828125"* minWidth=*"10.0"* prefWidth=*"199.99997965494788"* />

</**columnConstraints**>

<**rowConstraints**>

<**RowConstraints** maxHeight=*"41.333343505859375"* minHeight=*"10.0"* prefHeight=*"32.66667175292969"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"41.99999491373697"* minHeight=*"10.0"* prefHeight=*"37.33334350585938"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"35.333343505859375"* minHeight=*"10.0"* prefHeight=*"32.666656494140625"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"34.333343505859375"* minHeight=*"10.0"* prefHeight=*"31.0"* vgrow=*"SOMETIMES"* />

</**rowConstraints**>

<**children**>

<**Label** prefHeight=*"20.0"* prefWidth=*"174.0"* text=*"BOOK ID"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"BOOK NAME"* GridPane.rowIndex=*"1"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"BOOK EDITION"* GridPane.rowIndex=*"2"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"AUTHOR"* GridPane.rowIndex=*"3"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**TextField** fx:id=*"libraryBookID"* prefHeight=*"30.0"* prefWidth=*"227.0"* GridPane.columnIndex=*"1"* />

<**TextField** fx:id=*"libraryBookName"* layoutX=*"178.0"* layoutY=*"13.0"* prefHeight=*"31.0"* prefWidth=*"227.0"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"1"* />

<**TextField** fx:id=*"libraryBookPrice"* layoutX=*"178.0"* layoutY=*"48.0"* prefHeight=*"30.0"* prefWidth=*"227.0"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"2"* />

<**TextField** fx:id=*"bookAuthor"* layoutX=*"178.0"* layoutY=*"81.0"* prefHeight=*"30.0"* prefWidth=*"227.0"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"3"* />

</**children**>

</**GridPane**>

<**Button** layoutX=*"241.0"* layoutY=*"314.0"* mnemonicParsing=*"false"* onAction=*"#addNewBook"* text=*"ADD"*>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Button**>

<**Label** fx:id=*"bookAdditionStatus"* layoutX=*"134.0"* layoutY=*"267.0"* prefHeight=*"17.0"* prefWidth=*"355.0"* />

<**Text** fill=*"#f8f7f7"* layoutX=*"130.0"* layoutY=*"50.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"ADD NEW BOOKS"* textAlignment=*"CENTER"* wrappingWidth=*"253.47007751464844"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"18.0"* />

</**font**>

</**Text**>

</**children**></**AnchorPane**>

</**content**>

</**Tab**>

</**tabs**>

</**TabPane**>

</**children**>

</**AnchorPane**>

</**right**>

<**bottom**>

<**Button** mnemonicParsing=*"false"* onAction=*"#adminWindowSignOut"* prefHeight=*"36.0"* prefWidth=*"100.0"* text=*"Sign Out"* BorderPane.alignment=*"CENTER"*>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Button**>

</**bottom**>

</**BorderPane**>

**LoginWindow.fxml**

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<?**import** javafx.geometry.Insets?>

<?**import** javafx.scene.control.Button?>

<?**import** javafx.scene.control.ComboBox?>

<?**import** javafx.scene.control.Label?>

<?**import** javafx.scene.control.PasswordField?>

<?**import** javafx.scene.control.TextField?>

<?**import** javafx.scene.image.Image?>

<?**import** javafx.scene.image.ImageView?>

<?**import** javafx.scene.layout.AnchorPane?>

<?**import** javafx.scene.layout.ColumnConstraints?>

<?**import** javafx.scene.layout.GridPane?>

<?**import** javafx.scene.layout.RowConstraints?>

<?**import** javafx.scene.paint.Color?>

<?**import** javafx.scene.paint.LinearGradient?>

<?**import** javafx.scene.paint.Stop?>

<?**import** javafx.scene.text.Font?>

<?**import** javafx.scene.text.Text?>

<**AnchorPane** xmlns=*"http://javafx.com/javafx/8.0.241"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"controllers.LoginWindowController"*>

<**children**>

<**AnchorPane** fx:id=*"loginWindowPane"* layoutX=*"-7.0"* layoutY=*"1.0"* maxHeight=*"-Infinity"* maxWidth=*"-Infinity"* minHeight=*"-Infinity"* minWidth=*"-Infinity"* prefHeight=*"515.0"* prefWidth=*"1018.0"* style=*"-fx-background-color: #008080;"*>

<**children**>

<**Text** fill=*"#e0e5e8"* fontSmoothingType=*"LCD"* layoutX=*"68.0"* layoutY=*"118.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"LIBRARY SERVICES"* textAlignment=*"CENTER"* wrappingWidth=*"539.8763427734375"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"52.0"* />

</**font**>

<**stroke**>

<**LinearGradient** cycleMethod=*"REFLECT"* endX=*"1.0"* endY=*"1.0"*>

<**stops**>

<**Stop**>

<**color**>

<**Color** />

</**color**>

</**Stop**>

<**Stop** offset=*"1.0"*>

<**color**>

<**Color** red=*"1.0"* green=*"1.0"* blue=*"1.0"* />

</**color**>

</**Stop**>

</**stops**>

</**LinearGradient**>

</**stroke**>

</**Text**>

<**Text** layoutX=*"679.0"* layoutY=*"85.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"Login To The Library Services"* textAlignment=*"CENTER"* wrappingWidth=*"354.13677978515625"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Text**>

<**Label** layoutX=*"818.0"* layoutY=*"104.0"* text=*"User Name"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Label**>

<**TextField** fx:id=*"libraryUserNameLogin"* layoutX=*"764.0"* layoutY=*"133.0"* prefHeight=*"25.0"* prefWidth=*"185.0"* />

<**AnchorPane** layoutX=*"169.0"* layoutY=*"214.0"* prefHeight=*"251.0"* prefWidth=*"310.0"*>

<**children**>

<**ImageView** fitHeight=*"232.0"* fitWidth=*"232.0"* layoutX=*"52.0"* layoutY=*"-52.0"*>

<**image**>

<**Image** url=*"@../../../../../../Downloads/264745.png"* />

</**image**>

</**ImageView**>

</**children**>

</**AnchorPane**>

<**Label** layoutX=*"818.0"* layoutY=*"166.0"* text=*"Password"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Label**>

<**Button** layoutX=*"773.0"* layoutY=*"242.0"* mnemonicParsing=*"false"* onAction=*"#libraryUserLoginButton"* text=*"Login"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Button**>

<**Button** layoutX=*"878.0"* layoutY=*"243.0"* mnemonicParsing=*"false"* onAction=*"#signUpLoginPageButton"* text=*"Sign Up"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Button**>

<**Label** fx:id=*"libraryStatusLogin"* layoutX=*"777.0"* layoutY=*"301.0"* prefHeight=*"17.0"* prefWidth=*"169.0"* />

<**PasswordField** fx:id=*"libraryPasswordLogin"* layoutX=*"766.0"* layoutY=*"193.0"* prefHeight=*"25.0"* prefWidth=*"185.0"* />

</**children**>

</**AnchorPane**>

<**AnchorPane** fx:id=*"signUpWindowPane"* prefHeight=*"517.0"* prefWidth=*"1011.0"* style=*"-fx-background-color: #008080"* visible=*"false"*>

<**children**>

<**ImageView** fitHeight=*"298.0"* fitWidth=*"344.0"* layoutX=*"150.0"* layoutY=*"50.0"* pickOnBounds=*"true"* preserveRatio=*"true"*>

<**image**>

<**Image** url=*"@264745.png"* />

</**image**>

</**ImageView**>

<**Text** fontSmoothingType=*"LCD"* layoutX=*"51.0"* layoutY=*"400.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"Library Services"* textAlignment=*"CENTER"* wrappingWidth=*"488.5430006980896"*>

<**font**>

<**Font** name=*"System Bold"* size=*"39.0"* />

</**font**>

</**Text**>

<**Text** layoutX=*"700.0"* layoutY=*"50.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"Please Fill In The Details To Sign Up"* wrappingWidth=*"251.044921875"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Text**>

<**GridPane** layoutX=*"604.0"* layoutY=*"104.0"* prefHeight=*"265.0"* prefWidth=*"390.0"*>

<**columnConstraints**>

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"133.0"* minWidth=*"10.0"* prefWidth=*"126.0"* />

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"150.0"* minWidth=*"10.0"* prefWidth=*"144.0"* />

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"144.0"* minWidth=*"10.0"* prefWidth=*"144.0"* />

</**columnConstraints**>

<**rowConstraints**>

<**RowConstraints** maxHeight=*"74.0"* minHeight=*"10.0"* prefHeight=*"37.66668446858724"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"74.0"* minHeight=*"10.0"* prefHeight=*"30.99998219807943"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"74.0"* minHeight=*"8.0"* prefHeight=*"35.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"152.0"* minHeight=*"10.0"* prefHeight=*"39.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"183.0"* minHeight=*"10.0"* prefHeight=*"36.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"206.0"* minHeight=*"10.0"* prefHeight=*"48.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** maxHeight=*"206.0"* minHeight=*"10.0"* prefHeight=*"48.0"* vgrow=*"SOMETIMES"* />

</**rowConstraints**>

<**children**>

<**Label** text=*"User Name"* GridPane.rowIndex=*"2"*>

<**padding**>

<**Insets** left=*"20.0"* />

</**padding**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**Label** text=*"Password"* GridPane.rowIndex=*"3"*>

<**padding**>

<**Insets** left=*"20.0"* />

</**padding**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**Label** text=*"Role"* GridPane.rowIndex=*"4"*>

<**padding**>

<**Insets** left=*"20.0"* />

</**padding**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**TextField** fx:id=*"libraryUserNameSignUp"* prefHeight=*"30.0"* prefWidth=*"138.0"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"2"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

</**TextField**>

<**Button** mnemonicParsing=*"false"* onAction=*"#librarySignUpUserButton"* prefHeight=*"35.0"* prefWidth=*"130.0"* text=*"Sign Up"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"5"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"2.0"* />

</**GridPane.margin**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Button**>

<**Button** mnemonicParsing=*"false"* onAction=*"#loginSignUpPaneWindow"* prefHeight=*"35.0"* prefWidth=*"130.0"* text=*"Login"* GridPane.columnIndex=*"2"* GridPane.rowIndex=*"5"*>

<**GridPane.margin**>

<**Insets** left=*"2.0"* right=*"10.0"* />

</**GridPane.margin**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Button**>

<**PasswordField** fx:id=*"libraryPasswordSignUp"* prefHeight=*"30.0"* prefWidth=*"260.0"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"3"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

</**PasswordField**>

<**Label** fx:id=*"libraryUserStatusSignUp"* textAlignment=*"CENTER"* GridPane.columnSpan=*"2147483647"* GridPane.halignment=*"CENTER"* GridPane.rowIndex=*"6"* GridPane.valignment=*"CENTER"*>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

<**GridPane.margin**>

<**Insets** bottom=*"10.0"* />

</**GridPane.margin**>

</**Label**>

<**Label** text=*"Full Name"*>

<**GridPane.margin**>

<**Insets** left=*"20.0"* />

</**GridPane.margin**>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**ComboBox** fx:id=*"libraryUserTypesSignUp"* layoutX=*"10.0"* prefHeight=*"25.0"* prefWidth=*"189.0"* promptText=*"Select Role"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"4"* />

<**Label** text=*"Email"* GridPane.rowIndex=*"1"*>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

<**padding**>

<**Insets** left=*"20.0"* />

</**padding**>

</**Label**>

<**TextField** fx:id=*"libraryFullNameSignUp"* prefHeight=*"30.0"* prefWidth=*"138.0"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

</**TextField**>

<**TextField** fx:id=*"libraryEmailIdSignUp"* prefHeight=*"30.0"* prefWidth=*"138.0"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"1"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

</**TextField**>

<**Label** fx:id=*"libraryUserStatusSignUp"* prefHeight=*"27.0"* prefWidth=*"265.0"* text=*"Label"* GridPane.columnIndex=*"1"* GridPane.columnSpan=*"2"* GridPane.rowIndex=*"6"* />

</**children**>

</**GridPane**>

</**children**>

</**AnchorPane**>

</**children**>

</**AnchorPane**>

**MemberWindow.fxml**

<?**xml** version=*"1.0"* encoding=*"UTF-8"*?>

<?**import** javafx.geometry.Insets?>

<?**import** javafx.scene.control.Button?>

<?**import** javafx.scene.control.Label?>

<?**import** javafx.scene.control.TableView?>

<?**import** javafx.scene.control.TextField?>

<?**import** javafx.scene.image.Image?>

<?**import** javafx.scene.image.ImageView?>

<?**import** javafx.scene.layout.AnchorPane?>

<?**import** javafx.scene.layout.ColumnConstraints?>

<?**import** javafx.scene.layout.GridPane?>

<?**import** javafx.scene.layout.RowConstraints?>

<?**import** javafx.scene.text.Font?>

<?**import** javafx.scene.text.Text?>

<**AnchorPane** prefHeight=*"507.0"* prefWidth=*"930.0"* style=*"-fx-background-color: #088080;"* xmlns=*"http://javafx.com/javafx/8.0.241"* xmlns:fx=*"http://javafx.com/fxml/1"* fx:controller=*"controllers.MemberWindowController"*>

<**children**>

<**AnchorPane** fx:id=*"libraryMemberInformationPane"* prefHeight=*"507.0"* prefWidth=*"957.0"* visible=*"false"*>

<**children**>

<**GridPane** layoutX=*"84.0"* layoutY=*"103.0"* prefHeight=*"311.0"* prefWidth=*"509.0"*>

<**columnConstraints**>

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"240.0"* minWidth=*"10.0"* prefWidth=*"156.0"* />

<**ColumnConstraints** hgrow=*"SOMETIMES"* maxWidth=*"334.0"* minWidth=*"10.0"* prefWidth=*"334.0"* />

</**columnConstraints**>

<**rowConstraints**>

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

<**RowConstraints** minHeight=*"10.0"* prefHeight=*"30.0"* vgrow=*"SOMETIMES"* />

</**rowConstraints**>

<**children**>

<**Label** text=*"MEMBER NAME"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"MEMBER ID"* GridPane.rowIndex=*"1"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"BOOK INFORMATION"* GridPane.rowIndex=*"2"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"MEMBER EMAIL ID"* GridPane.rowIndex=*"3"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** text=*"PHONE"* GridPane.rowIndex=*"4"*>

<**font**>

<**Font** name=*"System Bold"* size=*"14.0"* />

</**font**>

</**Label**>

<**Label** fx:id=*"libraryMemberName"* text=*"memberName"* GridPane.columnIndex=*"1"*>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

<**GridPane.margin**>

<**Insets** left=*"10.0"* />

</**GridPane.margin**>

</**Label**>

<**Label** fx:id=*"libraryMemberId"* text=*"memberID"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"1"*>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

<**GridPane.margin**>

<**Insets** left=*"10.0"* />

</**GridPane.margin**>

</**Label**>

<**Label** fx:id=*"libraryMemberBookInformation"* text=*"BOOK INFORMATION"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"2"*>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

<**GridPane.margin**>

<**Insets** left=*"10.0"* />

</**GridPane.margin**>

</**Label**>

<**TextField** fx:id=*"libraryMemberEmailId"* prefHeight=*"26.0"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"3"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

</**TextField**>

<**TextField** fx:id=*"libraryMemberContactNumber"* prefHeight=*"26.0"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"4"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* right=*"10.0"* />

</**GridPane.margin**>

<**font**>

<**Font** size=*"14.0"* />

</**font**>

</**TextField**>

<**Button** alignment=*"CENTER\_RIGHT"* mnemonicParsing=*"false"* nodeOrientation=*"LEFT\_TO\_RIGHT"* onAction=*"#libraryMemberUpdateButton"* stylesheets=*"@../../../LibraryService/src/application/button.css"* text=*"Update"* textFill=*"WHITE"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"6"*>

<**GridPane.margin**>

<**Insets** left=*"10.0"* />

</**GridPane.margin**>

</**Button**>

<**Label** fx:id=*"libraryMemberUpdate"* prefHeight=*"57.0"* prefWidth=*"234.0"* text=*"Label"* GridPane.columnIndex=*"1"* GridPane.rowIndex=*"6"*>

<**GridPane.margin**>

<**Insets** left=*"100.0"* />

</**GridPane.margin**>

</**Label**>

</**children**>

</**GridPane**>

<**Label** layoutX=*"343.0"* layoutY=*"49.0"* text=*"member Information"*>

<**font**>

<**Font** size=*"16.0"* />

</**font**>

</**Label**>

</**children**>

</**AnchorPane**>

<**AnchorPane** fx:id=*"libraryMemberBookRegistrationPane"* prefHeight=*"507.0"* prefWidth=*"971.0"* visible=*"false"*>

<**children**>

<**Label** layoutX=*"103.0"* layoutY=*"63.0"* prefHeight=*"25.0"* prefWidth=*"177.0"* text=*"Available Books"*>

<**font**>

<**Font** size=*"16.0"* />

</**font**>

</**Label**>

<**TableView** fx:id=*"libraryAvailableBooksTable"* layoutX=*"8.0"* layoutY=*"120.0"* prefHeight=*"249.0"* prefWidth=*"487.0"* />

<**Label** layoutX=*"597.0"* layoutY=*"63.0"* prefHeight=*"25.0"* prefWidth=*"143.0"* text=*"Books Regsitered"*>

<**font**>

<**Font** size=*"16.0"* />

</**font**>

</**Label**>

<**Button** layoutX=*"160.0"* layoutY=*"477.0"* mnemonicParsing=*"false"* onAction=*"#libraryBookRegisterButton"* prefHeight=*"30.0"* prefWidth=*"143.0"* stylesheets=*"@../../../LibraryService/src/application/button.css"* text=*"Register"* textFill=*"WHITE"* />

<**Label** fx:id=*"libraryBookRegistrationStatus"* layoutX=*"132.0"* layoutY=*"413.0"* prefHeight=*"30.0"* prefWidth=*"234.0"* text=*"Registration Status For Selected Book"* />

<**Button** layoutX=*"640.0"* layoutY=*"477.0"* mnemonicParsing=*"false"* onAction=*"#libraryBookRemovalButton"* prefHeight=*"30.0"* prefWidth=*"109.0"* stylesheets=*"@../../../LibraryService/src/application/button.css"* text=*"Remove"* textFill=*"WHITE"* />

<**TableView** fx:id=*"libraryBookRegisteredTable"* layoutX=*"481.0"* layoutY=*"120.0"* prefHeight=*"249.0"* prefWidth=*"487.0"* />

<**Label** fx:id=*"libraryBookRemovalStatus"* alignment=*"TOP\_LEFT"* layoutX=*"633.0"* layoutY=*"416.0"* prefHeight=*"30.0"* prefWidth=*"256.0"* text=*"Status for removed Book"* wrapText=*"true"* />

</**children**>

</**AnchorPane**>

<**AnchorPane** fx:id=*"libraryMemberWelcomeWindowPane"* layoutY=*"60.0"* prefHeight=*"447.0"* prefWidth=*"965.0"*>

<**children**>

<**Label** fx:id=*"libraryWelcomeMemberId"* layoutX=*"775.0"* layoutY=*"179.0"* prefHeight=*"25.0"* prefWidth=*"124.0"* text=*"member ID"*>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**Text** fill=*"#f2f3f5"* layoutX=*"535.0"* layoutY=*"91.0"* strokeType=*"OUTSIDE"* strokeWidth=*"0.0"* text=*"WELCOME TO LIBRARY SERVICES"* textAlignment=*"CENTER"* wrappingWidth=*"410.3992919921875"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"24.0"* />

</**font**>

</**Text**>

<**Label** layoutX=*"600.0"* layoutY=*"132.0"* prefHeight=*"23.0"* prefWidth=*"146.0"* text=*"MEMBER NAME"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"18.0"* />

</**font**>

</**Label**>

<**Label** layoutX=*"601.0"* layoutY=*"181.0"* prefHeight=*"21.0"* prefWidth=*"112.0"* text=*"MEMBER ID"*>

<**font**>

<**Font** name=*"Arial Bold"* size=*"18.0"* />

</**font**>

</**Label**>

<**Label** fx:id=*"libraryWelcomeMemberName"* layoutX=*"774.0"* layoutY=*"129.0"* prefHeight=*"25.0"* prefWidth=*"131.0"* text=*"member"*>

<**font**>

<**Font** name=*"System Bold"* size=*"16.0"* />

</**font**>

</**Label**>

<**AnchorPane** layoutX=*"-1.0"* prefHeight=*"447.0"* prefWidth=*"516.0"*>

<**children**>

<**ImageView** fitHeight=*"269.0"* fitWidth=*"273.0"* layoutX=*"136.0"* layoutY=*"72.0"*>

<**image**>

<**Image** url=*"@../../../../../../Downloads/8750683.png"* />

</**image**>

</**ImageView**>

</**children**>

</**AnchorPane**>

</**children**>

</**AnchorPane**>

<**Button** mnemonicParsing=*"false"* onAction=*"#libraryMemberInformationButton"* prefHeight=*"32.0"* prefWidth=*"193.0"* text=*"INFORMATION"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Button**>

<**Button** layoutX=*"400.0"* layoutY=*"1.0"* mnemonicParsing=*"false"* onAction=*"#libraryMemberBookRegistrationButton"* prefHeight=*"32.0"* prefWidth=*"195.0"* text=*"REGISTRATION"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Button**>

<**Button** layoutX=*"879.0"* mnemonicParsing=*"false"* onAction=*"#MemberWinodwSignOut"* prefHeight=*"32.0"* prefWidth=*"92.0"* style=*"-fx-background-radius: 2;"* text=*"Sign Out"*>

<**font**>

<**Font** name=*"System Bold"* size=*"15.0"* />

</**font**>

</**Button**>

</**children**>

</**AnchorPane**>