

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI KHOA CÔNG NGHỆ THÔNG TIN

-----oOo-----



BÁO CÁO BÀI TẬP LỚN

Môn học: Lập trình Java

Đề tài: Ứng dụng thu ngân cửa hàng sách

Giảng viên hướng dẫn: Vũ Huân

Sinh viên thực hiện

Phùng Gia Nghiên

Chu Thị Hồng Nhung

Lớp: CNTT5 – K62 – N09

Hà Nội, ngày 30 tháng 4 năm 2023

Mục lục

Phần 1: Giới thiệu chung.....	3
1. Giới thiệu dự án.	3
2. Mô tả dự án.	3
Phần 2: Hướng dẫn cài đặt	4
Phần 3: Xây dựng chương trình	6
1. Giới thiệu và nêu chức năng của các package	6
2. Giới thiệu và các tương tác trong giao diện.	7
2.1. Kết nối database.....	7
2.2. Trang đăng nhập, đăng ký	8
2.3. Trang quản lý thu ngân chính.	13
2.4 Trang thêm sách.....	18
2.5 Trang thay đổi thông tin sách.	19
Phần 4: Kết luận và hướng phát triển.....	24

Phần 1: Giới thiệu chung

1. Giới thiệu dự án.

Chúng em xin được giới thiệu về dự án của nhóm em, là một ứng dụng thu ngân được viết bằng ngôn ngữ lập trình Java. App của chúng em được thiết kế để quản lý sản phẩm trong cửa hàng, hỗ trợ tạo hóa đơn bán giúp việc tính tiền và quản lý sản phẩm được dễ dàng hơn. Trong quá trình làm dự án, nhóm chúng em tổ chức chương trình theo mô hình MVC để dễ quản lý và phát triển.

2. Mô tả dự án.

- App có các màn hình như sau:

- Màn hình đăng nhập, đăng ký.
- Màn hình chính
- Màn hình thêm sách
- Màn hình chỉnh sửa sách
- Màn hình chỉnh sửa giá, số lượng sách

- Một số chức năng chính

- Đăng nhập, đăng ký: Hỗ trợ tạo tài khoản và đăng nhập và ứng dụng.
- Màn hình chính: Hiện thông tin các quyển sách trong cửa hàng, hiển thị các sản phẩm trong đơn hàng khi mua, tìm kiếm sách.
- Màn hình thêm sách: Thêm một đầu sách mới.
- Màn hình chỉnh sửa sách: Chỉnh sửa thông tin của sách.
- Màn hình chỉnh sửa giá, số lượng sách: Thay đổi số lượng hoặc giá của sách.

Phần 2: Hướng dẫn cài đặt

Để chạy được app bạn cần thực hiện các bước sau :

1. Tải file của app trên gg driver về và giải nén tìm đến file test.jar khởi động lên .
2. Tạo cơ sở dữ liệu trên máy tính của bạn. Bạn có thể sử dụng hệ quản trị cơ sở dữ liệu MySQL.

2.1. Tạo một cơ sở dữ liệu mới tên là qlbooks chứa các bảng : “bill”, “bill_detail”, “books”, “sign_up”.

2.2. Tạo một cơ sở dữ liệu mới bằng cách chọn “Create new database” hoặc “New database” trong giao diện của MySQL.

2.3. Sau khi tạo xong cơ sở dữ liệu thì bạn cần tạo thêm các bảng bằng cách chọn “Create new table” hoặc “New table” trong giao diện của MySQL.

2.4. Đặt kiểu dữ liệu các bảng như hình

bill

#	Tên	Kiểu dữ liệu	Length/Set	Unsigned	Allow N...	Zerofill	Mặc định	Bình luận	Collation	Expression	Virtuality
1	id_out	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		utf8mb4_general_ci		
2	date	DATE		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL				
3	total_bill	DOUBLE		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0				

bill_detail

#	Tên	Kiểu dữ liệu	Length/Set	Unsigned	Allow N...	Zerofill	Mặc định	Bình luận	Collation	Expression	Virtuality
1	id_out	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		utf8mb4_general_ci		
2	idBook_out	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		utf8mb4_general_ci		
3	price	DOUBLE		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default				
4	amount	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default				

books

#	Tên	Kiểu dữ liệu	Length/Set	Unsigned	Allow N...	Zerofill	Mặc định	Bình luận	Collation	Expression	Virtuality
1	idBook	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		utf8mb4_general_ci		
2	nameBook	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	''		utf8mb4_general_ci		
3	cost	DOUBLE		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0				
4	typeBook	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	''		utf8mb4_general_ci		
5	authorBook	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	''		utf8mb4_general_ci		
6	number	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default				
7	publisher	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	''		utf8mb4_general_ci		
8	yearPublisher	INT	11	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	0				

sign_up

#	Tên	Kiểu dữ liệu	Length/Set	Unsigned	Allow N...	Zerofill	Mặc định	Bình luận	Collation	Expression	Virtuality
1	username	VARCHAR	50	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	No default		utf8mb4_general_ci		
2	password	VARCHAR	50	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	NULL		utf8mb4_general_ci		

Tùy chỉnh cổng kết nối của MySQL từ bất kỳ thành 3306 trong XAMPP, bạn cần thực hiện như sau:

1. Mở XAMPP Control Panel và bật Apache và MySQL.
2. Trong giao diện của XAMPP thì nhấp vào ô “Config của MySQL” và nhấp vào my.ini và cửa sổ NotePad hiện ra .
3. Trong giao diện của cửa sổ NotePad bạn đi tìm biến có tên là “port” và đặt nó bằng 3306. Nếu nó bằng 3306 rồi thì bỏ qua bước này và Ctrl + S để lưu lại cửa sổ NotePad
4. Khởi động lại MySQL bằng cách ấn vào nút “Stop” và sau đó lại ấn vào “Start” trên XAMPP Control Panel.

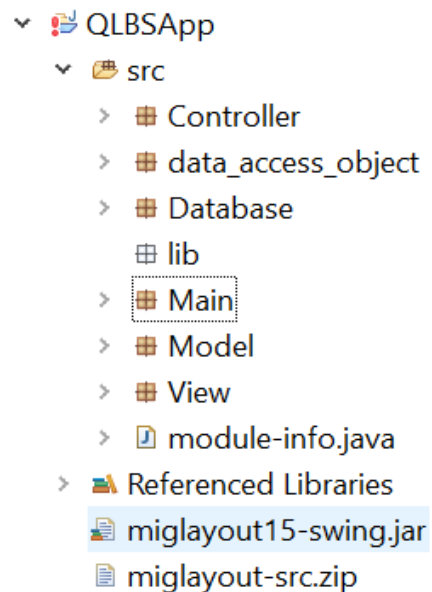
Đường dẫn gg driver đến file nén của app

[https://drive.google.com/drive/folders/11FNKSzI1](https://drive.google.com/drive/folders/11FNKSzI1Ju-Xyz6TLveiiGTtbQHlherg)

[Ju-Xyz6TLveiiGTtbQHlherg](https://drive.google.com/drive/folders/11FNKSzI1Ju-Xyz6TLveiiGTtbQHlherg)

Phần 3: Xây dựng chương trình

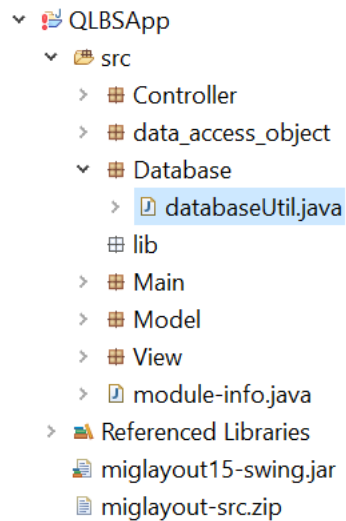
1. Giới thiệu và nêu chức năng của các package



- Model: Chứa các đối tượng (object) mô tả dữ liệu trong ứng dụng, được dùng để truyền dữ liệu giữa các thành phần trong ứng dụng, ví dụ như truyền dữ liệu từ Controller đến View.
- Database: Chứa phương thức liên kết với cơ sở dữ liệu.
- Data_access_object: Gồm các hàm để tương tác với cơ sở dữ liệu của từng đối tượng.
- Controller: chứa các lớp xử lý logic điều khiển (controller) trong ứng dụng. Các lớp này thường xử lý yêu cầu (request) từ phía người dùng và thực hiện các hành động tương ứng, ví dụ như truy xuất và cập nhật dữ liệu trong database.
- View: Chứa các class giao diện của ứng dụng.

2. Giới thiệu và các tương tác trong giao diện.

2.1. Kết nối database



- Hàm kết nối với cơ sở dữ liệu.

```
public class databaseUtil {
    public static Connection getConnection() {
        Connection c = null;

        try {
            com.mysql.jdbc.Driver driver = new com.mysql.jdbc.Driver();
            DriverManager.registerDriver(driver);

            String url = "jdbc:mysql://localhost:3306/qlbooks";
            String username = "root";
            String password = "";

            c = DriverManager.getConnection(url, username, password);

        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return c;
    }
}
```

Hàm đóng kết nối với cơ sở dữ liệu.

```
public static void closeConnection(Connection c) {
    if (c != null) {
        try {
            c.close();
        } catch (SQLException e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
}
```

2.2. Trang đăng nhập, đăng ký

2.2.1 Giao diện

2.2.2 Xây dựng giao diện và tương tác với giao diện

a) Xây dựng giao diện.

- View
 - AddBookView.java
 - DetailView.java
 - EmployeeInformationView.java
 - InvoiceView.java
 - QLBSView.java
 - View_login.java
 - View_signup.java
 - View_update_book.java
 - View_update_cost_book.java
 - View_update_number_book.java

The image displays two side-by-side Java Swing windows. The left window, titled 'Đăng nhập' (Login), features a large title 'Đăng Nhập' at the top. Below it, there are two input fields: 'username :' and 'password :'. At the bottom, there are two buttons: 'Đăng nhập' (Login) and 'Tạo tài khoản !' (Create account!). The right window, titled 'Đăng ký' (Register), features a large title 'Đăng Ký' at the top. Below it, there are three input fields: 'username :', 'password :', and 're-password :'. At the bottom, there is a single button: 'Đăng Ký' (Register).

b) Xây dựng tương tác.


```

package Model;
public class login {
    private String username;
    private String password;

    public login() {
        super();
    }

    public login(String username, String pass) {
        super();
        this.username = username;
        this.password = pass;
    }

    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    @Override
    public String toString() {
        return "login [username=" + username + ", password=" + password + "]";
    }
}

```

➤ Xây dựng các thuộc tính, phương thức quản lý thông tin tài khoản.

- Interface data

```

package data_access_object;

import java.util.ArrayList;

public interface data_interface<T> {

    public int insert(T t);

    public int update(T t);

    public int delete(T t);

    public ArrayList<T> selectAll();

    public T selectById(T t);

}

```

- File login_data

- ▼ data_access_object
 - > bill_data.java
 - > bill_detail_data.java
 - > books_data.java
 - > data_interface.java
 - > login_data.java

```

import java.sql.Connection;

public class login_data implements data_interface<login>{

    public static login_data getInstance() {
        return new login_data();
    }

    @Override
    public int insert(login t) {
        int check = 0 ;

        try {
            Connection connection = databaseUtil.getConnection();

            String sql = "INSERT INTO sign_up "
                + " VALUES (?, ?)";

            PreparedStatement pStatement = connection.prepareStatement(sql);

            pStatement.setString(1, t.getUsername());
            pStatement.setString(2, t.getPassword());

            check = pStatement.executeUpdate();
            databaseUtil.closeConnection(connection);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return check;
    }

    @Override
    public int update(login t) {
        int check = 0 ;

        try {
            Connection connection = databaseUtil.getConnection();

            String sql = "UPDATE sign_up "
                + " SET "
                + " password=?"
                + " WHERE username=?";

            PreparedStatement pStatement = connection.prepareStatement(sql);

            pStatement.setString(1, t.getPassword());
            pStatement.setString(2, t.getUsername());

            check = pStatement.executeUpdate();

            databaseUtil.closeConnection(connection);
        } catch (SQLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return check;
    }

    @Override
    public int delete(login t) {
        // TODO Auto-generated method stub
        return 0;
    }
}

```

```

@Override
public ArrayList<login> selectAll() {
    ArrayList<login> list = new ArrayList<>();

    try {
        Connection connection = databaseUtil.getConnection();

        String sql = "SELECT * FROM sign_up";

        PreparedStatement pStatement = connection.prepareStatement(sql);

        ResultSet resultSet = pStatement.executeQuery();

        while (resultSet.next()) {
            String username = resultSet.getString("username");
            String password = resultSet.getString("password");

            login login1 = new login(username, password);

            list.add(login1);
        }











        databaseUtil.closeConnection(connection);
    } catch (SQLException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return list;
}

@Override
public login selectById(login t) {
    // TODO Auto-generated method stub
    return null;
}

```

- Giải thích: Định nghĩa và xây dựng các phương thức tương tác với bảng login trên Database, bao gồm các phương thức thêm tài khoản, thay đổi tài khoản, chọn tất cả các tài khoản có trong database.

- File List_signup điều khiển việc đăng nhập, đăng ký:

- ▼  Controller
 - >  List_insert_bill.java
 - >  List_insert_book.java
 - >  List_reset.java
 - >  List_search_book.java
 - >  List_signup.java
 - >  List_update_book.java
 - >  List_update_cost_book.java
 - >  List_update_number_book.java
- >  data_access_object

```

public class List_signup implements ActionListener{

    private View_signup view_signup;
    private View_login view_login;

    public List_signup(View_signup view_signup) {
        this.view_signup = view_signup;
    }

    public List_signup(View_login view_login) {
        this.view_login = view_login;
    }
}

```

➤ Phương thức điều khiển hành động đăng ký

```
@Override
public void actionPerformed(ActionEvent e) {
    String ac = e.getActionCommand();
    boolean check = true;
    if (ac.equals("Đăng Ký")) {
        ArrayList<login> list = new ArrayList<>();

        list = login_data.getInstance().selectAll();
        for (login log_in : list) {
            if (this.view_signup.getjTextField_user().getText().equals(log_in.getUsername())) {
                check = false;
                JOptionPane.showMessageDialog(view_signup,
                    "tài khoản đã tồn tại ! ",
                    "Error",
                    JOptionPane.ERROR_MESSAGE
                );
            }
        }

        if (this.view_signup.getjTextField_user().getText().equals("")) {
            check = false;
            JOptionPane.showMessageDialog(view_signup,
                "không được bỏ trống! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }
        if (String.valueOf(this.view_signup.getjPasswordField_pass().getPassword()).equals("")) {
            check = false;
            JOptionPane.showMessageDialog(view_signup,
                "không được bỏ trống! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }
        if (String.valueOf(this.view_signup.getjPasswordField_re_pass().getPassword()).equals("")) {
            check = false;
            JOptionPane.showMessageDialog(view_signup,
                "không được bỏ trống! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }
        int checkpass = String.valueOf(this.view_signup.getjPasswordField_pass().getPassword())
            .compareTo(String.valueOf(this.view_signup.getjPasswordField_re_pass().getPassword()));
        if (checkpass > 0 || checkpass < 0) {
            check = false;
            JOptionPane.showMessageDialog(view_signup,
                "mật khẩu không trùng khớp ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }
        if (check == true) {
            String user = this.view_signup.getjTextField_user().getText();
            String pass = String.valueOf(this.view_signup.getjPasswordField_re_pass().getPassword());

            login login1 = new login(user, pass);

            login_data.getInstance().insert(login1);
            JOptionPane.showMessageDialog(view_signup,
                "đăng ký thành công ! ",
                "",
                JOptionPane.INFORMATION_MESSAGE
            );
            this.view_signup.mainSrceen_login();
            this.view_signup.dispose();
        }
    }
}
```

- Giải thích: Đọc tất cả các tài khoản có trong database lưu vào 1 ArrayList, check xem nếu tài khoản đăng ký đã có trong database

thì không được đăng ký, hiện lỗi nếu bỏ trống các field nhập dữ liệu.

➤ Phương thức điều khiển hành động đăng nhập.

```
boolean check_acc = false;
boolean check_pass = false;
if (ac.equals("Đăng nhập")) {
    ArrayList<login> list = new ArrayList<>();

    list = login_data.getInstance().selectAll();
    for (login log_in : list) {
        if (this.view_login.getTextField_user_login().getText().compareTo(log_in.getUsername()) == 0 ) {
            check_acc = true;
        }
    }

    for (login login : list) {
        if (String.valueOf(this.view_login.getPasswordField_pass_login().getPassword()).compareTo(login.getPassword()) == 0 ) {
            check_pass = true;
        }
    }

    if (check_acc == false ) {
        JOptionPane.showMessageDialog(view_login,
            "tài khoản không tồn tại ! ",
            "Error",
            JOptionPane.ERROR_MESSAGE
        );
        return;
    }

    if (check_pass == false ) {
        JOptionPane.showMessageDialog(view_login,
            "mật khẩu không đúng ! ",
            "Error",
            JOptionPane.ERROR_MESSAGE
        );
        return;
    }

    if (check_pass == true && check_acc == true) {
        JOptionPane.showMessageDialog(view_login,
            "đăng nhập thành công ! ",
            "",
            JOptionPane.INFORMATION_MESSAGE
        );
        this.view_login mainScreen();
        this.view_login.dispose();
    }
}

if (ac.equals("Tạo tài khoản !")) {
    this.view_signup.mainScreen();
    this.view_login.dispose();
}
}
```

- Giải thích: Kiểm tra tài khoản nhập có trong database hay không, nếu có cho phép đăng nhập, nếu không có hiện thông báo tài khoản không tồn tại.

2.3. Trang quản lý thu ngân chính.

2.3.1 Giao diện.

2.3.2 Xây dựng dao diện và tương tác với giao diện.







a) Xây dựng giao diện.

- ▼ View
 - > AddBookView.java
 - > DetailView.java
 - > EmployeeInformationView.java
 - > InvoiceView.java
 - > QLBSView.java
 - > View_login.java
 - > View_signup.java
 - > View_update_book.java
 - > View_update_cost_book.java
 - > View_update_number_book.java

b) Xây dựng tương tác.

- ▼ data_access_object
 - > bill_data.java
 - > bill_detail_data.java
 - > books_data.java
 - > data_interface.java
 - > login_data.java

- File bill_data, bill_data, bill_detail_data để đọc và tương tác với dữ liệu book, bill_detail và bill trong database.

- ▼  Model
 - >  bill_detail.java
 - >  bill.java
 - >  books.java
 - >  login.java
 - >  nhanvien.java

- Tạo 3 đối tượng book, bill, bill_detail gồm hàm constructor, các hàm getter và setter.
- Hàm hiện thông tin các đầu sách trong cửa hàng.

```
package Controller;

import java.awt.event.ActionEvent;

public class List_reset implements ActionListener{
    private QLBSView view;

    public List_reset(QLBSView view) {
        this.view = view;
    }
    @Override
    public void actionPerformed(ActionEvent e) {
        String acString = e.getActionCommand();
        if (acString.equals("Reset")) {
            this.view.reset_Information_book();
        }
        if (acString.equals("Đăng xuất")) {
            this.view.Screen_login();
            this.view.dispose();
        }
    }
}

public void reset_Information_book() {
    DefaultTableModel model = (DefaultTableModel) informationTable.getModel();
    ArrayList<books> list = books_data.getInstance().selectAll();

    model.setRowCount(0);

    for(books a:list) {
        model.addRow(new Object[] {
            a.getIdBook(),
            a.getNameBook(),
            a.getAuthorBook(),
            a.getPublisher(),
            a.getTypeBook(),
            a.getYearPublisher(),
            a.getNumber(),
            a.getCost()
        });
    }
}
```

- Giải thích: Lưu thông tin các quyển sách trong database vào ArrayList<books>, hiện thông tin vào bảng informationTable.

➤ Hàm hiển thị hàng khi nhập mã sách và số lượng.

```
@Override
public void actionPerformed(ActionEvent e) {

    String ac = e.getActionCommand();
    int sequenceNumber = 0;
    if (ac.equals("Nhập")) {

        ArrayList<books> list = new ArrayList<>();

        list = books_data.getInstance().selectAll();

        boolean check_id = false;

        books insert_book = new books();

        for (books check_id_book : list) {
            if (check_id_book.getIdBook().compareTo(this.view.getIdBookField().getText()) == 0 ) {
                check_id = true;
                insert_book = check_id_book;
            }
        }

        if (check_id == false) {
            JOptionPane.showMessageDialog(view,
                "không có idBook nào ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }

        boolean check_number = false;

        if (insert_book.getNumber() >= Integer.valueOf(this.view.getNumberBookField().getText()) ) {
            check_number = true;
        }

        if (check_number == false) {
            JOptionPane.showMessageDialog(view,
                "không đủ số lượng ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }

        if (check_id == true && check_number == true) {

            Double total_money = Integer.valueOf(this.view.getNumberBookField().getText())*insert_book.getCost();

            DefaultTableModel model = (DefaultTableModel) this.view.getTable().getModel();
            model.addRow(new Object[] {
                insert_book.getIdBook(),
                insert_book.getNameBook(),
                this.view.getNumberBookField().getText(),
                insert_book.getCost(),
                insert_book.getPublisher(),
                total_money
            });
            Double total_bill = 0.0 ;

            for (int i = 0; i < this.view.getTable().getRowCount(); i++) {
                total_bill += (Double) this.view.getTable().getValueAt(i, 5);
            }

            this.view.getTotalPriceLabel().setText(String.valueOf(total_bill));
            insert_book.setNumber(insert_book.getNumber() - Integer.valueOf(this.view.getNumberBookField().getText()));
            books_data.getInstance().update(insert_book);
        }

        bill_detail bill_update = new bill_detail(test.generateInvoiceNumber(), insert_book.getIdBook(),
            insert_book.getCost(), Integer.valueOf(this.view.getNumberBookField().getText()));
        bill_detail_data.getInstance().insert(bill_update);
    }
}
```


- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Nhập”, so sánh mã sách với các mã sách trong list, nếu có thì tiếp tục so sánh số lượng với số lượng của id sách trong list, nếu không đủ số lượng hiện ra thông báo. Nếu id và số lượng hợp lệ, sách sẽ đc hiện ra trong bảng billTable.

➤ Hàm tìm kiếm sách.

```
@Override
public void actionPerformed(ActionEvent e) {
    String ac = e.getActionCommand();

    if (ac.equals("Tìm kiếm")) {

        ArrayList<books> list = new ArrayList<>();

        list = books_data.getInstance().selectAll();

        boolean check_name_book = false;
        boolean check_author_book = false;

        books search_book = new books();

        for (books books_check_name : list) {
            if (books_check_name.getNameBook().equals(this.view.getSearchBookNameField().getText())) {
                check_name_book = true;
            }
            if (books_check_name.getAuthorBook().equals(this.view.getSearchAuthorField().getText()) ) {
                check_author_book = true;
            }
            if (check_name_book == true) {
                search_book = books_check_name;
                break;
            }
        }

        if (check_author_book == false) {
            JOptionPane.showMessageDialog(view,
                "không có tên tác giả nào ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }

        if (check_author_book == true && check_name_book == true) {
            DefaultTableModel model = (DefaultTableModel) this.view.getInformationTable().getModel();

            model.setRowCount(0);
            model.addRow(new Object[] {
                search_book.getIdBook(),
                search_book.getNameBook(),
                search_book.getAuthorBook(),
                search_book.getPublisher(),
                search_book.getTypeBook(),
                search_book.getYearPublisher(),
                search_book.getNumber(),
                search_book.getCost()
            });
        }
    }
}
```

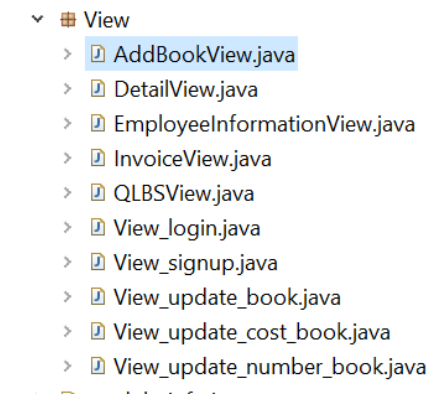
- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Tìm kiếm”, đọc dữ liệu đang nhập từ các text field sau đó so sánh với các quyển sách trong list. Nếu trùng khớp hiện thông tin sách ra bảng informationTable, nếu không hiện thông báo.

2.4 Trang thêm sách.

2.4.1 Thiết kế giao diện.

2.4.2 Xây dựng giao diện và tương tác với giao diện.

a) Xây dựng giao diện.



A screenshot of a Java Swing window titled 'Thêm sách' (Add Book). The window has a title bar with standard Windows controls (minimize, maximize, close). The main content area has a title 'THÊM SÁCH' in bold. Below the title, there are several input fields and a dropdown menu arranged vertically. The labels and corresponding input types are: 'Mã sách:' followed by a text field; 'Tên sách:' followed by a text field; 'Thể loại:' followed by a dropdown menu currently showing 'Sách Giáo trình'; 'Tác giả:' followed by a text field; 'Nhà XB:' followed by a text field; 'Năm XB:' followed by a text field; 'Giá:' followed by a text field; and 'Số lượng' followed by a text field. At the bottom of the window, there are two buttons: 'Thêm' (Add) and 'Thoát' (Exit).

b) Tương tác với giao diện.

```

@Override
public void actionPerformed(ActionEvent e) {

    String ac = e.getActionCommand();

    if (ac.equals("Thêm")) {
        String idBook = this.addBookView.getAddBookIDField().getText();
        String nameBook = this.addBookView.getAddBookNameField().getText();
        Double cost = Double.valueOf(this.addBookView.getAddCostField().getText());
        String typeBook = String.valueOf(this.addBookView.getAddKobBox().getSelectedItem());
        String authorBook = this.addBookView.getAddAuthorField().getText();
        int number = Integer.valueOf(this.addBookView.getAddNumberField().getText());
        String publisher = this.addBookView.getAddPublisherField().getText();
        int yearPublisher = Integer.valueOf(this.addBookView.getAddYearField().getText());

        books books1 = new books(idBook, nameBook, cost, typeBook, authorBook, number, publisher, yearPublisher);

        ArrayList<books> list = new ArrayList<>();
        list = books_data.getInstance().selectAll();

        boolean check = false;

        for (books books_check : list) {
            if (books_check.getIdBook().compareTo(idBook) == 0) {
                check = true;
            }
        }

        if (check == true) {
            JOptionPane.showMessageDialog(addBookView,
                "đã có idBook ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }

        if (check == false) {
            books_data.getInstance().insert(books1);
            JOptionPane.showMessageDialog(addBookView,
                "thêm thành công ! ",
                "",
                JOptionPane.INFORMATION_MESSAGE
            );
        }
    }

    if (ac.equals("Thoát")) {
        this.addBookView.dispose();
    }
}

```

- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Thêm”, đọc dữ liệu từ các text field, tạo một biến books book1 lưu thông tin từ các text field. Kiểm tra id của book1 đã có trong database hay chưa, nếu có hiện thông báo đã có id sách, nếu chưa có lưu book1 vào database.

2.5 Trang thay đổi thông tin sách.

2.5.1 Thiết kế giao diện.

2.5.2 Xây dựng giao diện và tương tác với giao diện.

a) Xây dựng giao diện

- ▼ View
 - > AddBookView.java
 - > DetailView.java
 - > EmployeeInformationView.java
 - > InvoiceView.java
 - > QLBSView.java
 - > View_login.java
 - > View_signup.java
 - > View_update_book.java
 - > View_update_cost_book.java
 - > View_update_number_book.java
- > module-info.java

The image displays three sequential screenshots of a Java Swing window titled "Thêm sách" (Add Book). Each window shows a form titled "CHỈNH SỬA SÁCH" (Edit Book). The first window shows the full form with fields for "Mã sách:", "Tên sách:", "Thể loại:" (set to "Sách Giáo trình"), "Tác giả:", "Nhà XB:", "Năm XB:", "Giá:", and "Số lượng:". The second window shows the form with "Mã sách:", "Tên sách:", and "Giá:" fields. The third window shows the form with "Mã sách:", "Tên sách:", and "Số lượng thêm:" fields. Each window has "Thêm" (Add) and "Thoát" (Exit) buttons at the bottom.

b) Tương tác với giao diện

- Thay đổi thông tin sách.
 - Hàm thay đổi thông tin sách.

```

@Override
public void actionPerformed(ActionEvent e) {
    String ac = e.getActionCommand();
    boolean check = false;
    if (ac.equals("Cập nhật")) {
        String idBook = this.update_book.getUpdate_BookIDField().getText();
        String nameBook = this.update_book.getUpdate_BookNameField().getText();
        Double cost = Double.valueOf(this.update_book.getUpdate_CostField().getText());
        String typeBook = String.valueOf(this.update_book.getUpdate_KobBox().getSelectedItem());
        String authorBook = this.update_book.getUpdate_AuthorField().getText();
        int number = Integer.valueOf(this.update_book.getUpdate_NumberField().getText());
        String publisher = this.update_book.getUpdate_PublisherField().getText();
        int yearPublisher = Integer.valueOf(this.update_book.getUpdate_YearField().getText());

        books books1 = new books(idBook, nameBook, cost, typeBook, authorBook, number, publisher, yearPublisher);

        ArrayList<books> list = new ArrayList<>();
        list = books_data.getInstance().selectAll();

        for (books books2 : list) {
            if (books2.getIdBook().compareTo(idBook) == 0 ) {
                check = true;
            }
        }
        if (check == false) {
            JOptionPane.showMessageDialog(update_book,
                "không có idBook nào ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }

        if (check == true ) {
            books_data.getInstance().update(books1);
            JOptionPane.showMessageDialog(update_book,
                "cập nhật thành công ! ",
                "...",
                JOptionPane.INFORMATION_MESSAGE
            );
        }
    }
    if (ac.equals("Thoát")) {
        this.update_book.dispose();
    }
}

```

- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Cập nhật”, đọc dữ liệu từ các text field, so sánh id sách với id sách trong list, nếu trùng khớp thì thay đổi thông tin của id trong database, nếu không trùng thì hiện thông báo. Khi click button “Thoát”, đóng màn hình.

- Thay đổi số lượng sách.

- Hàm thay đổi số lượng sách.

```

@Override
public void actionPerformed(ActionEvent e) {
    String ac = e.getActionCommand();

    boolean check = false;
    if (ac.equals("Thêm")) {

        String idBook = this.update_number_book.getUpdate_number_BookIDField().getText();
        String nameBook = this.update_number_book.getUpdate_number_BookNameField().getText();
        int number = Integer.valueOf(this.update_number_book.getUpdate_number_NumberField().getText());
        books books1 = new books();

        ArrayList<books> list = new ArrayList<>();
        list = books_data.getInstance().selectAll();
    }
}

```

```

for (books books2 : list) {
    if (books2.getIdBook().compareTo(idBook) == 0 ) {
        books1 = books2;
        check = true;
    }
}
if (check == false) {
    JOptionPane.showMessageDialog(update_number_book,
        "không có idBook nào ! ",
        "Error",
        JOptionPane.ERROR_MESSAGE
    );
    return;
}
if (check == true) {
    books1.setNumber(books1.getNumber()+number);
    books_data.getInstance().update(books1);
    JOptionPane.showMessageDialog(update_number_book,
        "cập nhật thành công ! ",
        "",
        JOptionPane.INFORMATION_MESSAGE
    );
}
}
if (ac.equals("Thoát")) {
    this.update_number_book.dispose();
}
}

```

- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Thêm”, đọc dữ liệu từ các text field, so sánh id sách với id sách trong list, nếu trùng khớp thì thay đổi thông tin của id trong database, nếu không trùng thì hiện thông báo. Khi click button “Thoát”, đóng màn hình.
- Hàm thay đổi giá sách.

```

@Override
public void actionPerformed(ActionEvent e) {
    String ac = e.getActionCommand();

    boolean check = false;

    if (ac.equals("Thêm")) {

        String idBook = this.update_cost_book.getUpdate_cost_BookIDField().getText();
        String nameBook = this.update_cost_book.getUpdate_cost_BookNameField().getText();
        Double cost = Double.valueOf(this.update_cost_book.getUpdate_cost_CostField().getText());
        books books1 = new books();

        ArrayList<books> list = new ArrayList<>();
        list = books_data.getInstance().selectAll();

        for (books books2 : list) {
            if (books2.getIdBook().compareTo(idBook) == 0 ) {
                books1 = books2;
                check = true;
            }
        }

        if (check == false) {
            JOptionPane.showMessageDialog(update_cost_book,
                "không có idBook nào ! ",
                "Error",
                JOptionPane.ERROR_MESSAGE
            );
            return;
        }
    }
}

```

```

        if (check == true) {
            books1.setCost(cost);
            books_data.getInstance().update(books1);
            JOptionPane.showMessageDialog(update_cost_book,
                "cập nhật thành công ! ",
                "",
                JOptionPane.INFORMATION_MESSAGE
            );
        }
    }
    if (ac.equals("Thoát")) {
        this.update_cost_book.dispose();
    }
}

```

- Giải thích: Tạo ArrayList<books> chứa các sách có trong database. Khi click button “Thêm”, đọc dữ liệu từ các text field, so sánh id sách với id sách trong list, nếu trùng khớp thì thay đổi thông tin của id trong database, nếu không trùng thì hiện thông báo. Khi click button “Thoát”, đóng màn hình.

Phần 4: Kết luận và hướng phát triển

- Nhóm chúng em đã hoàn thành “App thu ngân nhà sách”. Trong đó có một số ưu điểm và những nhược điểm sau:
 - + Ưu điểm: App đã lấy dữ liệu trên database xuống, dễ dàng truy xuất dữ liệu. Thêm vào đó app còn có các chức năng để đổi màu, đăng xuất khỏi app. Giao diện app thân thiện, dễ dàng sử dụng. Code dễ bảo trì do đặt class, và 1 hàm chỉ làm 1 nhiệm vụ
 - + Nhược điểm: Chưa tối ưu thao tác, các chức năng tìm kiếm, thay đổi thông tin sách bắt buộc phải điền hết thông tin gây ra nhiều thao tác không cần thiết.
 - Trong thời gian thực hiện dự án, nhờ sự chỉ bảo của thầy Vũ Huân và sự giúp đỡ của các bạn, nhóm chúng em đã thu được nhiều kết quả trong việc học lập trình Java.
 - Do thời gian và khả năng có hạn nên bài tập lớn của nhóm chúng em còn rất nhiều thiếu sót, chúng em rất mong nhận được sự góp ý, giúp đỡ của thầy và các bạn để bài tập của chúng em được hoàn thiện hơn.
- Chúng em xin cảm ơn!