



# BÀI TẬP

## CHƯƠNG TRÌNH LẬP TRÌNH VIÊN

### JAVA

---

## MODULE 2: KỸ THUẬT LẬP TRÌNH VÀ CÔNG NGHỆ LƯU TRỮ



# BÀI 1: Biểu thức Lambda



*Mục tiêu chính:*

- Áp dụng biểu thức lambda trong việc giải quyết các bài toán

## 1.1. Xuất mảng

- ✓ **Yêu cầu:** Hãy viết lại đoạn code sau bằng cách sử dụng biểu thức Lambda:

```
List<integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7);
for(Integer n: list) {
    System.out.println(n);
}
```

- ✓ **Hướng dẫn:**

- Sử dụng biểu thức Lambda

```
List<integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7);
list.forEach(n -> System.out.println(n));
```

## 1.2. Tính thành tiền

- ✓ **Yêu cầu:** Sử dụng biểu thức Lambda để tính thành tiền cho mảng số lượng với đơn giá cho trước như sau:

```
List<Integer> soLuong = Arrays.asList(new Integer[]{2, 3, 5, 7,});
int donGia = 5200;
```

Kết quả sau khi tính:

```
run:
Thành tiền = 10400 VNĐ
Thành tiền = 15600 VNĐ
Thành tiền = 26000 VNĐ
Thành tiền = 36400 VNĐ
BUILD SUCCESSFUL (total time: 0 seconds)
```

- ✓ **Hướng dẫn:**

- Sử dụng biểu thức Lambda

```
List<Integer> soLuong = Arrays.asList(new Integer[]{2, 3, 5, 7,});
int donGia = 5200;
soLuong.forEach(element -> {
    System.out.println("Thành tiền = " + donGia * element + " VNĐ");
});
```

## 1.3. Tính diện tích hình tròn, chữ nhật, vuông

✓ **Yêu cầu: Sử dụng biểu thức Lambda để tính diện tích hình tròn, hình chữ nhật**

- Sử dụng console. Bán kính của hình tròn, chiều dài và chiều rộng được nhập vào từ bàn phím. Viết chương trình tính diện tích của hình tròn, hình chữ nhật.
- Interface cho sẵn như sau:

```
interface Area {
    public double calArea(double x1, double x2);
}
```

Kết quả sau khi tính:

```
run:
Nhập bán kính:
10
S = PI * (r*r) = 314.1592653589793
Nhập chiều dài:
15
Nhập chiều rộng:
8
S = W x H = 120.0
BUILD SUCCESSFUL (total time: 6 seconds)
```

✓ **Hướng dẫn:**

- Sử dụng biểu thức Lambda

```
BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Nhập bán kính: ");
double r = Double.parseDouble(input.readLine());
Area circle = (Double x1, Double x2) -> x1 * Math.pow(x2, 2);
System.out.println("S = PI * (r*r) = " + circle.calArea(Math.PI, r));
System.out.println("Nhập chiều dài: ");
double H = Double.parseDouble(input.readLine());
System.out.println("Nhập chiều rộng: ");
double W = Double.parseDouble(input.readLine());
Area rectangle = (Double x1, Double x2) -> x1 * x2;
System.out.println("S = W x H = " + rectangle.calArea(W, H));
```

- Ghi chú: Học viên có thể thiết kế form (thay vì sử dụng console) cho bài này.

## 1.4. Tính cộng, trừ, nhân chia hai số

✓ **Yêu cầu: Sử dụng biểu thức Lambda để tính cộng, trừ, nhân, chia hai số:**

Sau khi nhập hai số và nhấn "Tổng"

✓ **Với interface như sau:**

```
interface Operator {
    public Integer operate(Integer operand1, Integer operand2);
}
```

✓ **Hướng dẫn sử dụng:**

- Khi người dùng số thứ nhất và số thứ hai, bấm nút "Tổng"/ "Hiệu"/ "Tích"/ "Thương" thì hiển thị kết quả tương ứng.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmPhepTinh: JFrameForm
  - \* (Các thể hiện phía dưới đều nằm trong form)
    - txtSo1: TextField (nhập liệu)
    - txtSo2: TextField (nhập liệu)
    - txtKetQua: TextField (kết suất)
    - btnTong: Button (xử lý tính tổng)
    - btnHieu: Button (xử lý tính hiệu)
    - btnTich: Button (xử lý tính tích)
    - btnThuong: Button (xử lý tính thương)

▪ **Nhập:**

- Số thứ nhất
- Số thứ hai

▪ **Xuất:**

- Tổng/ hiệu/ tích/ thương của hai số

✓ **Hướng dẫn**

- Tạo interface Operator như yêu cầu
- Tạo lớp JFrame có tên là **frmPhepTinh**
- Gọi sử dụng interface, áp dụng biểu thức lambda để tính tổng và xuất kết quả khi người dùng nhấn nút "Tổng":

```
private void btnTongActionPerformed(java.awt.event.ActionEvent evt) {
    int soThuNhat = Integer.parseInt(txtSoThuNhat.getText());
    int soThuHai = Integer.parseInt(txtSoThuHai.getText());
    Operator op1 = (Integer operand1, Integer operand2) -> operand1 + operand2;
    int tong = op1.operate(soThuNhat, soThuHai);
    lblKetQua.setText("Tổng");
    txtKetQua.setText(String.valueOf(tong));
}
```

- HV làm tương tự cho các nút "Hiệu", "Tích", "Thương"

## 1.5. Sắp xếp danh sách chuỗi

✓ **Yêu cầu: Hãy tạo ra một danh sách các phần tử kiểu String. Sử dụng biểu thức Lambda thực hiện việc sắp xếp chuỗi theo:**

- Chiều dài chuỗi từ ngắn nhất tới dài nhất
- Chiều dài chuỗi từ dài nhất tới ngắn nhất
- Thứ tự Alphabet

✓ **Tóm tắt yêu cầu**

▪ **Nhập:**

- Danh sách các phần tử kiểu String

▪ **Xuất:**

- Các kết quả lần lượt theo từng yêu cầu

✓ **Hướng dẫn**

- Sử dụng Lambda Expression

# BÀI 2: Java Stream API



Mục tiêu chính:

- Áp dụng Stream API trong việc giải quyết các bài toán

## 2.1. Xử lý chuỗi với Stream

- ✓ **Yêu cầu:** Viết chương trình xử lý chuỗi với Stream như sau:
  - Tạo ra một List 1 kiểu String có 15 phần tử, trong đó có cả phần tử có giá trị ""
  - In List 1 vừa tạo
  - Trong List 1:
    - Đếm xem có bao nhiêu phần tử có giá trị ""
    - Đếm xem có bao nhiêu phần tử có chiều dài  $\geq 5$ .
    - Đếm xem có bao nhiêu phần tử giá trị bắt đầu là "a"
    - Đếm xem có bao nhiêu phần tử có giá trị chính xác là "happy"
  - Tạo một List 2 mới từ List 1
    - Chỉ gồm các phần tử có giá trị khác ""
  - Tạo một List 3 mới từ List 1
    - Chỉ gồm các phần tử có chiều dài  $\geq 3$  và  $\leq 6$
    - In chuỗi này với định dạng mỗi phần tử cách nhau bằng dấu ", "
  - Tạo ra một List 4 mới từ List 1
    - Mỗi phần tử sẽ có giá trị là chuỗi đang có gắn thêm vào chuỗi happy.
    - In List 4.
- **Nhập:**
  - Các phần tử String cho List 1 hoặc tạo mặc định List 1
- **Xuất:**
  - Theo các yêu cầu trên

### ✓ Hướng dẫn

- Sử dụng Stream để giải quyết yêu cầu

## 2.2. Xử lý số với Stream

- ✓ **Yêu cầu:** Viết chương trình xử lý số với Stream như sau:
  - Tạo ra một List 1 kiểu Integer có 15 phần tử. In List 1 vừa tạo
  - Tạo ra List 2 từ List 1 mà giá trị mỗi phần tử của List 2 bằng bình phương giá trị phần tử của List 1. In List 2 vừa tạo
  - Trong List 2: hãy thực hiện việc thống kê và cho biết
    - Giá trị lớn nhất trong List 2
    - Giá trị nhỏ nhất trong List 2

- Tổng giá trị của tất cả các phần tử trong List 2
- Trung bình của tất cả các phần tử trong List 2
- Tạo ra List 3 từ List 2 mà chỉ lấy các phần tử là số nguyên tố. In List 3 vừa tạo.
- Tạo ra List 4 từ List 2 mà chỉ lấy các phần tử là số chính phương. In List 4 vừa tạo.
- **Nhập:**
  - Các phần tử Integer cho List 1 hoặc tạo mặc định List 1
- **Xuất:**
  - Theo các yêu cầu trên

✓ **Hướng dẫn**

- Sử dụng Stream để giải quyết yêu cầu

### 2.3. Xử lý đối tượng với Stream

✓ **Yêu cầu:** Viết chương trình xử lý đối tượng với Stream như sau:

- Tạo ra một List kiểu Student có 10 phần tử. In List 1 vừa tạo.
- In List: danh sách Student kèm theo điểm trung bình.
- Cho biết trong danh sách Student này có bao nhiêu bạn có tuổi  $\geq 18$ .
- Cho biết trong danh sách này có bao nhiêu Student nào có firstName bắt đầu là "H". In ra sinh viên đầu tiên trong danh sách có firstName bắt đầu là "H".
- Trong List: hãy thực hiện việc thống kê và cho biết
  - Điểm trung bình lớn nhất
  - Điểm trung bình nhỏ nhất
  - Trung bình của điểm các trung bình
- Tạo một List 1 mới với các Very Good Student có điểm trung bình  $\geq 8$ . In list này.
- Ví dụ:

```
run:
List of Students:
Phuong Khuat      age = 17      mark 1 = 8.0 & mark 2 = 7.0      avg = 7.5
Hanh Nguyen       age = 19      mark 1 = 8.5 & mark 2 = 7.5      avg = 8.0
Hoa Huynh         age = 18      mark 1 = 7.5 & mark 2 = 6.5      avg = 7.0
Duyen Phan        age = 20      mark 1 = 7.5 & mark 2 = 7.5      avg = 7.5
Number of students have old >=18: 3
Number of Student have FirstName start 'H':2
First Student has FisrtName start at 'H':
Hanh Nguyen       age = 19      mark 1 = 8.5 & mark 2 = 7.5      avg = 8.0
Highest Avg Mark in List : 8.0
Lowest Avg Mark in List : 7.0
Sum of all Avg Mark : 30.0
Average of all Avg Marks : 7.5
List of Very Good Students:
Hanh Nguyen       age = 19      mark 1 = 8.5 & mark 2 = 7.5      avg = 8.0
```



- **Nhập:**

- Các phần tử Student cho List hoặc tạo mặc định List

- **Xuất:**

- Theo các yêu cầu trên

- ✓ **Hướng dẫn**

- Tạo lớp Student có các thuộc tính: firstName, lastName, age, mark1, mark2; xây dựng các phương thức khởi tạo, phương thức tính điểm tb = (mark1+mark2)/2, Phương thức toString() xuất thông tin Student
- Sử dụng Stream để giải quyết yêu cầu



# BÀI 3: Làm việc với dữ liệu JSON



Mục tiêu chính:

- Làm việc với tập tin .json: đọc, hiển thị, ghi tập tin .json

## 3.1. Thống kê nhân viên theo đơn vị

### ✓ Yêu cầu: Xây dựng chương trình đọc và thống kê nhân viên theo đơn vị:

- Khi chạy chương trình, sẽ hiển thị thống kê nhân viên theo đơn vị này như sau:
  - Tên công ty, địa chỉ, email, tổng số nhân viên
  - Thông tin từng đơn vị: tên đơn vị, số nhân viên

```
***** Thông tin công ty *****
Tên công ty: Công ty Dịch vụ Hoàng hôn Sớm
Địa chỉ: 11223 Trần hưng Đạo Q.1 TP HCM
Mail: hhsom2016@gmail.com
Điện thoại: 08-83222145
Tổng số đơn vị: 7 đơn vị
Tổng số nhân viên: 96
***** Thông tin đơn vị *****
1/Tên đơn vị: Đơn vị A1
Số nhân viên: 14
2/Tên đơn vị: Đơn vị A2
Số nhân viên: 15
3/Tên đơn vị: Đơn vị B1
Số nhân viên: 14
4/Tên đơn vị: Đơn vị B2
Số nhân viên: 20
5/Tên đơn vị: Đơn vị B3
Số nhân viên: 19
6/Tên đơn vị: Đơn vị B4
Số nhân viên: 14
```

### ✓ Tóm tắt yêu cầu

#### ▪ Qui tắc xử lý:

- Cung cấp tập tin: QLCT\_1.json
- Đọc và hiển thị kết quả như định dạng trên

### ✓ Hướng dẫn

- Cung cấp tập tin: QLCT\_1.json có cấu trúc như sau:

```
{'CONG_TY': [{'Dia_chi': '11223 Trần hưng Đạo Q.1 TP HCM ',
              'Dien_thoai': '08-83222145',
              'ID': 1,
              'Mail': 'hhsom2016@gmail.com',
```

```

        'Muc_Luong_Toai_thieu': 3500000,
        'Ten': 'Công ty Dịch vụ Hoàng hôn Sớm',
        'Tuoi_Toai_da': 50,
        'Tuoi_Toai_thieu': 20}],
'DON_VI': [{ 'ID': 1,
              'ID_CHI_NHANH': 1,
              'So_Nhan_vien': '14',
              'Ten': 'Đơn vị A1',
              'Ty_le': '14.58'},
            { 'ID': 2,
              'ID_CHI_NHANH': 1,
              'So_Nhan_vien': '15',
              'Ten': 'Đơn vị A2',
              'Ty_le': '15.62'},
            ...
          ]
    }

```

- Download json-simple-1.1.jar và cài vào project
- Áp dụng các bước thực hiện đã nêu trong phần lý thuyết để thực hiện yêu cầu của đề bài

### 3.2. Ghi dữ liệu quản lý giao dịch vào tập tin tập tin JSON

- ✓ **Yêu cầu: Xây dựng chương trình tạo các giao dịch và ghi các giao dịch vào tập tin JSON:**

Mô tả: Hệ thống quản lý 2 loại giao dịch:

- Giao dịch vàng: Mã giao dịch, ngày giao dịch (lấy giờ của hệ thống theo định dạng **ngay-thang-nam-gio-phut-giay**), đơn giá, số lượng, loại vàng có 3 loại 18k, 24k, 9999.
  - Thành tiền được tính như sau: thành tiền = số lượng \* đơn giá.
- Giao dịch tiền tệ: Mã giao dịch, ngày giao dịch (ngày/tháng/năm), tỷ giá (cũng là đơn giá), số lượng, loại tiền tệ có 3 loại: USD, EUR, AUD, loại giao dịch mua/bán. Thành tiền được tính như sau:
  - Nếu loại giao dịch là "mua" thì: thành tiền = số lượng \* tỷ giá
  - Nếu loại giao dịch là "bán" thì: thành tiền = (số lượng \* tỷ giá) \* 1.05

Dựa vào mô tả trên, hãy:

- Tạo lớp **GiaoDich** với các thuộc tính và phương thức chung (giao dịch vàng cũng là giao dịch).
- Tạo lớp **GiaoDichTienTe** kế thừa từ lớp GiaoDich với các thuộc tính riêng và phương thức cần thiết.
- Nhập xuất danh sách các giao dịch.
- Tính tổng số lượng cho từng loại.
- Tính tổng thành tiền cho từng loại.

Công việc của người dùng:

- Người dùng lần lượt tạo các giao dịch, chương trình hiển thị kết quả và thống kê. Chương trình sẽ hỏi người dùng có tiếp tục thực hiện giao dịch nữa hay không? Nếu có thì tiếp tục. nếu không thì dừng.
- Sau khi người dùng không tiếp tục thực hiện giao dịch nữa thì hỏi người dùng có muốn ghi vào tập tin hay không? Nếu người dùng chọn 1: Có => ghi vào tập tin. 0: Không => Không ghi
- Khi người dùng chọn ghi:
  - Lấy thông tin ngày hiện tại theo định dạng: **ngay-thang-nam-gio-phut-giay**
  - Lưu danh sách các giao dịch tiền và vàng vào tập tin **ngay-thang-nam-gio-phut-giay.json**

```

Quản lý giao dịch
Nhập mã:
gd001
Ngày giao dịch: 23-10-2017-15-03-41
Số lượng:
10
Chọn loại giao dịch: 1: Vàng, 2: Tiền
1
Chọn loại 18k/24k/9999:
18k
Nhập giá vàng:
2650000
GiaoDich [ma=gd001, ngay=23-10-2017-15-03-41, donGia=2650000.0, soLuong=10, loai=18k, tinhThanhTien()=2.65E7]
Tổng số lượng: 10
Tổng số tiền: 2.65E7

Bạn muốn tiếp tục giao dịch? nhập 1: Có, nhập khác: không
1
Nhập mã:
gd002
Ngày giao dịch: 23-10-2017-15-04-02
Số lượng:
100
Chọn loại giao dịch: 1: Vàng, 2: Tiền
2
Chọn loại USD/EUR/AUD:
USD
Nhập giá ngoại tệ:
22780
Khách hàng mua hay bán? 1: Mua, 2: Bán
1
GD mua: GiaoDich [ma=gd002, ngay=23-10-2017-15-04-02, donGia=22780.0, soLuong=100, loai=USD, tinhThanhTien()=2278000.0]
Tổng số lượng: 100
Tổng số tiền: 2278000.0

Bạn muốn tiếp tục giao dịch? nhập 1: Có, nhập khác: không
0
Bạn có muốn ghi lại giao dịch này không? 1: Có, Khác: Không
1
{"vangs":[{"ngay":"23-10-2017-15-03-41","ma":"gd001","dongia":2650000.0,"loai":"18k","soluong":10}], "tiens":[{"ngay":"23-10-2017-15-04-02","ma":"gd002","dongia":22780.0,"loai":"USD","soluong":100}]}
    
```

- Kiểm tra lại nội dung sau khi ghi: một tập tin mới được tạo ra theo đúng quy định đặt tên.

```
[
  "vangs": [
    {
      "ngay": "23-10-2017-15-03-41",
      "ma": "gd001",
      "dongia": 2650000.0,
      "loai": "18k",
      "soluong": 10
    }
  ],
  "tiens": [
    {
      "ngay": "23-10-2017-15-04-02",
      "ma": "gd002",
      "dongia": 22780.0,
      "mua": true,
      "loai": "USD",
      "soluong": 100
    }
  ]
]
```

#### ✓ Hướng dẫn sử dụng:

- Nhập mã giao dịch, số lượng
- Chọn loại giao dịch (1: Giao dịch Vàng, 2: Giao dịch Tiền tệ):
- Nếu chọn 1:
  - Chọn loại, nhập đơn giá => thêm vào list giao dịch vàng => hiển thị danh sách các giao dịch vàng: thông tin giao dịch, tổng số lượng, tổng số tiền
- Nếu chọn 2:
  - Chọn loại, nhập tỷ giá, nhập loại mua/bán => thêm vào list tương ứng => hiển thị thông tin các giao dịch, tổng số lượng, tổng thành tiền
- Sau mỗi lần thêm, chương trình sẽ hỏi người dùng có thêm tiếp hay không? Nếu chọn 1: có tiếp tục thêm, nếu chọn khác 1: dừng việc tạo giao dịch.
- Khi dừng tạo giao dịch, chương trình sẽ hỏi người dùng có ghi vào tập tin hay không? Nếu chọn 1: Có, nếu chọn khác 1: không
- In kết quả sau khi ghi tập tin.

#### ✓ Tóm tắt yêu cầu

##### ▪ Quy tắc xử lý:

- Ghi các giao dịch theo định dạng vào file json: nam-thang-ngay-gio-phut-giay.json

#### ✓ Hướng dẫn

- File .json sẽ có cấu trúc như sau:

```
[
  "vangs": [
    {
      "ngay": "23-10-2017-15-03-41",
      "ma": "gd001",
      "dongia": 2650000.0,
      "loai": "18k",
      "soluong": 10
    }
  ],
  "tiens": [
    {
      "ngay": "23-10-2017-15-04-02",
      "ma": "gd002",
      "dongia": 22780.0,
      "mua": true,
      "loai": "USD",
      "soluong": 100
    }
  ]
]
```

- Tạo lớp GiaoDich
- Tạo lớp GiaoDichTienTe kế thừa từ lớp GiaoDich
- Gọi các lớp đã tạo trên để thực hiện việc tạo, tính toán và hiển thị theo yêu cầu
- Áp dụng các bước đã được hướng dẫn để ghi nội dung vào file JSON mới theo yêu cầu

### 3.3. Thêm thông tin phòng karaoke

#### ✓ Yêu cầu: Xây dựng chương trình ghi thêm thông tin phòng karaoke vào tập tin .json:

- Ghi thêm thông tin phòng karaoke vào tập tin quan\_ly\_phong\_karaoke.json.
- Người dùng nhập thông tin của một phòng karaoke: gồm id, ten, mã số, id loại phòng (1 hoặc 2 hoặc 3), đơn giá, số khách tối đa, trạng thái (CON\_TRONG, CO\_KHACH).
- Ghi thông tin vừa nhập vào file.
- Thông báo kết quả ghi
- Xuất nội dung tập tin .json sau khi ghi.

```
Nhập thông tin phòng karaoke
nhập ID:
1
nhập Tên:
VIP 01
Nhập mã số:
VIP_01
Nhập ID loại phòng (1 hoặc 2 hoặc 3):
1
Nhập đơn giá:
65000
Nhập số khách tối đa:
10
Nhập trạng thái CON_TRONG hay CO_KHACH
CON_TRONG
Đã ghi file!
{"VIP_01":{"Trang_thai":"CON_TRONG","ID":1,"Ten":"VIP 01","Ma_so":"VIP_01","So_Khach_To_i_da":10,"ID_LOAI_PHONG":1,"Don_gia":65000}}
```

#### ✓ Hướng dẫn

- Tập tin quan\_ly\_phong\_karaoke.json được cung cấp sẵn có cấu trúc như sau:

```
{ } quan_ly_phong_karaoke.json
1 {
2   "VIP_01": {
3     "Trang_thai": "CON_TRONG",
4     "ID": 1,
5     "Ten": "VIP 01",
6     "Ma_so": "VIP_01",
7     "So_Khach_Toai_da": 10,
8     "ID_LOAI_PHONG": 1,
9     "Don_gia": 65000.0
10  },
11  "VIP_05": {
12    "Trang_thai": "CON_TRONG",
13    "ID": 5,
14    "Ten": "VIP 5",
15    "Ma_so": "VIP_05",
16    "So_Khach_Toai_da": 15,
17    "ID_LOAI_PHONG": 2,
18    "Don_gia": 75000.0
19  }
20 }
```

- Chú ý: chú ý cấu trúc của một phần tử trong tập tin JSON để thêm vào cho đúng.
- Áp dụng các bước đã được hướng dẫn trong lý thuyết để thực hiện việc ghi tiếp vào tập tin JSON đã có.

# BÀI 4: XML DOM



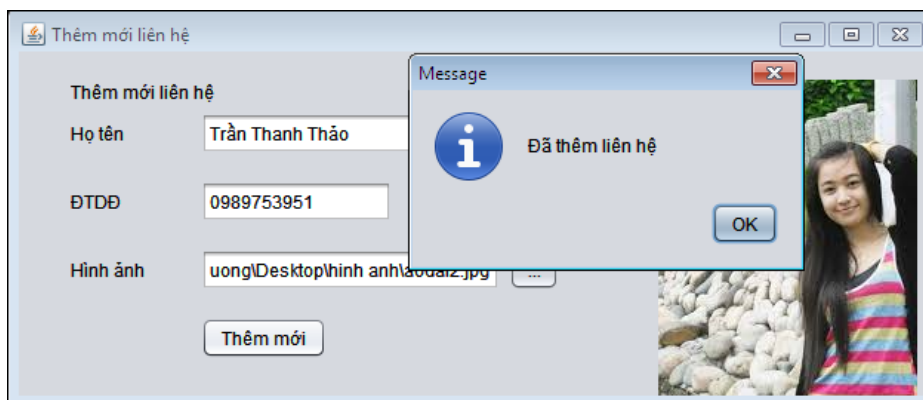
## Mục tiêu chính

- Sử dụng XML DOM trong việc đọc, ghi dữ liệu XML

## 4.1. Thêm Liên hệ vào tập tin xml

- ✓ **Yêu cầu:** Xây dựng chương trình Thêm mới Liên hệ có các thông tin sau vào file xml:

- Tên
- ĐTDD
- Hình ảnh



- ✓ **Hướng dẫn sử dụng:**
  - Nhập vào các ô nhập liệu các thông tin: tên, điện thoại di động, hình ảnh. Nhấn nút "Thêm" => Các thông tin vừa nhập sẽ được lưu vào tập tin lienhe.xml.
- ✓ **Tóm tắt yêu cầu**
  - **Thiết kế giao diện người dùng (sử dụng giao diện đã thiết kế ở bài Giao diện – module 1):**
    - frmManHinhThemLienHe: FrmThemLienHe (extends từ JFrame)
  - \* (Các thể hiện phía dưới đều nằm trong Frame)
    - txtTen: JTextField (nhập liệu)
    - txtDtdd: JTextField (nhập liệu)
    - fchHinhAnh: JFileChooser (chọn)
    - btnThem: JButton (Xử lý thêm)
  - **Nhập:**
    - Tên
    - ĐTDD
    - Hình ảnh...
  - **Xuất:**
    - "Đã thêm liên hệ" nếu thêm được hoặc "Không thể thêm Liên hệ"
  - **Qui tắc xử lý:**
    - Tạo tập tin lienhe.xml

- Lấy thông tin được nhập và ghi vào tập tin lienhe.xml

✓ **Thuật giải**

- Hiển thị đường dẫn file được chọn trong txtHinhAnh
- Khai báo biến ten nhận giá trị từ txtHoTen
- Khai báo biến dtdd nhận giá trị từ txtDtdd
- Khai báo biến hinhAnh nhận giá trị từ txtHinhAnh
- Xử lý ghi vào file xml các thông tin ten, dtdd, hinhAnh
- Xuất kết quả dưới dạng dialog thông báo

✓ **Hướng dẫn**

- Hiện thực phương thức btnThemMoiActionPerformed() để lấy thông tin người dùng nhập trên màn hình => xử lý và thêm liên hệ vào tập tin lienhe.xml

```
private void btnThemMoiActionPerformed(java.awt.event.ActionEvent evt) {
    String ten = txtHoTen.getText();
    String dtdd = txtDTDD.getText();
    File f = new File(txtHinhAnh.getText());
    String hinhAnh = f.getName();

    try {
        DocumentBuilderFactory docFactory = DocumentBuilderFactory.newInstance();
        DocumentBuilder docBuilder = docFactory.newDocumentBuilder();

        // root elements
        Document doc = docBuilder.newDocument();
        Element rootElement = doc.createElement("lienhes");
        doc.appendChild(rootElement);

        // lienhe
        Element lienhe = doc.createElement("lienhe");
        rootElement.appendChild(lienhe);

        // hoten element
        Element hoten = doc.createElement("hoten");
        hoten.appendChild(doc.createTextNode(ten));
        lienhe.appendChild(hoten);

        // dtdd element
        Element dtdd1 = doc.createElement("dtdd");
        dtdd1.appendChild(doc.createTextNode(dtdd));
        lienhe.appendChild(dtdd1);
    } catch (Exception e) {
        // TODO: Handle exception
    }
}
```



```
// hinhanh element
Element hinh = doc.createElement("hinhanh");
hinh.appendChild(doc.createTextNode(hinhAnh));
lienhe.appendChild(hinh);

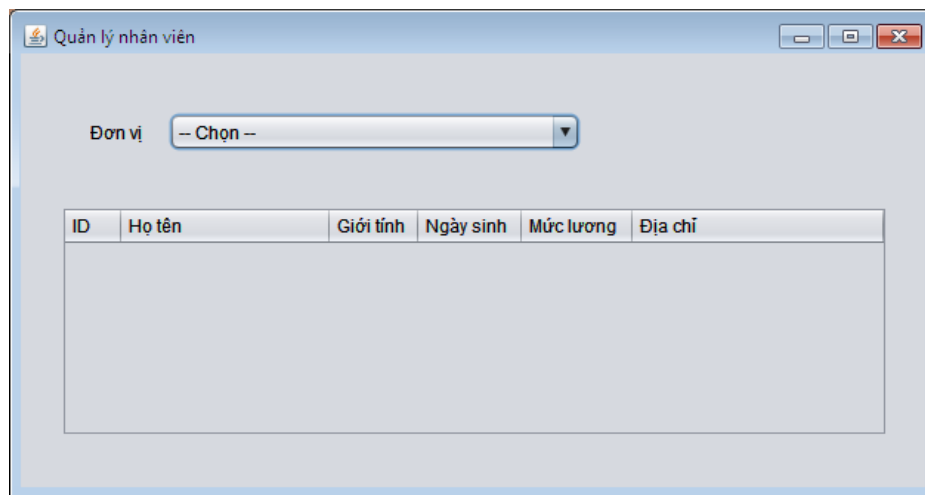
// write the content into xml file
TransformerFactory transformerFactory = TransformerFactory.newInstance();
Transformer transformer = transformerFactory.newTransformer();
DOMSource source = new DOMSource(doc);
StreamResult result = new StreamResult(new File("src/Bai4/lienhe.xml"));

transformer.transform(source, result);
JOptionPane.showMessageDialog(rootPane, "Đã thêm liên hệ");

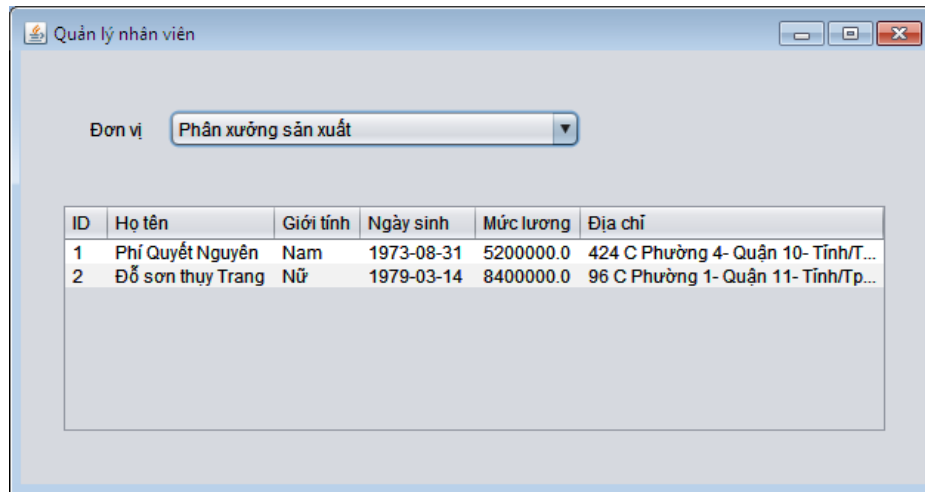
} catch (ParserConfigurationException | TransformerException pce) {
    System.out.println(pce.getMessage());
    JOptionPane.showMessageDialog(rootPane, "Không thể thêm liên hệ");
}
}
```

## 4.2. Đọc tập tin XML

- ✓ **Yêu cầu:** Xây dựng chương trình hiển thị nhân viên theo đơn vị như sau:



*Khi chạy chương trình*



*Khi chọn một đơn vị*

✓ **Hướng dẫn sử dụng:**

- Chọn một đơn vị => Hiển thị thông tin chi tiết của các nhân viên thuộc đơn vị được chọn.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmNhanVien: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- cbbDonVi: ComboBox (chọn lựa)
- tblNhanVien: Jtable (chỉ đọc)

▪ **Nhập:**

- Đơn vị

▪ **Xuất:**

- Thông tin nhân viên thuộc đơn vị được chọn

▪ **Qui tắc xử lý:**

- Mở tập tin don\_vi.xml
- Đưa tên đơn vị lấy được từ tập tin xml vào cbbDonVi
- Mở tập tin nhan\_vien.xml
- Đưa thông tin nhân viên lấy được từ tập tin xml vào tblNhanVien ứng với đơn vị được chọn

✓ **Hướng dẫn**

- Tạo tập tin don\_vi.xml có cấu trúc như sau:

```
<?xml version="1.0" encoding="UTF-8"?>
<cong_ty>
  <don_vi>
    <id>1</id>
    <ten>Phân xưởng sản xuất</ten>
  </don_vi>
  ...
</cong_ty>
```

- Tạo tập tin nhan\_vien.xml có cấu trúc như sau:

```
<?xml version="1.0" encoding="utf-8"?>
<don_vi>
    <nhan_vien>
        <id>1</id>
        <ho_ten>Phí Quyết Nguyên</ho_ten>
        <gioi_tinh>1</gioi_tinh>
        <ngay_sinh>1973-08-31</ngay_sinh>
        <muc_luong>5200000</muc_luong>
        <dia_chi>424 C Phường 4- Quận 10- Tỉnh/Tp CC</dia_chi>
        <id_don_vi>1</id_don_vi>
    </nhan_vien>
    ...
</don_vi>
```

- Lưu ý: HV tự thêm các nội dung đơn vị và nhân viên cho 2 tập tin này để có dữ liệu xử lý và hiển thị.
- Ở form Nhân viên, khai báo một biến toàn cục model kiểu table để chứa tạm dữ liệu trước khi hiển thị lên table và biến toàn cục kiểu combo box để chứa tạm dữ liệu trước khi hiển thị lên combo box:

```
ComboBoxModel cbbModel;
DefaultTableModel model = new DefaultTableModel();
```

- Trong phương thức khởi tạo form, khởi tạo các cột cho model vừa khai báo và đọc file don\_vi.xml lấy dữ liệu đưa lên combobox:

```
public FrmNhanVien() {
    ...
    model.addColumn("ID");
    model.addColumn("Họ tên");
    model.addColumn("Giới tính");
    model.addColumn("Ngày sinh");
    model.addColumn("Mức lương");
    model.addColumn("Địa chỉ");
    tblNhanVien.setModel(model);
    ArrayList<String> items = new ArrayList<>();
    items.add("-- Chọn --");
    // doc file xml don vi
    NodeList nList = docDSNode("src/module2/don_vi.xml", "don_vi");
    if (nList != null) {
        for (int temp = 0; temp < nList.getLength(); temp++) {
            Node nNode = nList.item(temp);
            if (nNode.getNodeType() == Node.ELEMENT_NODE) {
                Element eElement = (Element) nNode;
                String ten =
```

```

        eElement.getElementsByTagName("ten").item(0).getTextContent();
        items.add(ten);
    }
}

cbbModel = new DefaultComboBoxModel(items.toArray());
}

```

- Xử lý sự kiện "Chọn đơn vị": đọc file nhan\_vien.xml lấy dữ liệu đưa lên table các nhân viên thuộc đơn vị được chọn

# BÀI 5: Làm việc với CSDL



*Mục tiêu chính:*

Giúp cho học viên có thể làm việc với CSDL MySQL hoặc Oracle

## 5.1. Làm việc với CSDL MySQL

- ✓ **Yêu cầu:** Cho CSDL `phan_cong_nhan_vien_1_1_nam.sql`. Import vào MySQL và thực hiện các câu lệnh truy vấn

- **Ngữ cảnh:**

- Công ty X có 5 đơn vị (không bao gồm bộ phận quản lý), mỗi đơn vị có trung bình 25 nhân viên. Thông tin nhân viên được quản lý dựa theo biểu mẫu 1 (BM1). X quản lý việc phân công nhân viên dựa vào phiếu phân công (BM2). X hiện có 2 bộ phận quản lý với các công việc chính theo bảng mô tả công việc như sau:

Bộ phận	Công việc chính
Quản lý công ty	Quản lý hồ sơ nhân viên
Quản lý đơn vị	Phân công công việc cho các nhân viên thuộc đơn vị đang phụ trách

**BM1**
**Hồ sơ nhân viên**

Họ tên : .....Giới tính : .....CMND: .....  
 Ngày sinh : ..... Mức lương : .....  
 Địa chỉ : ..... Đơn vị : .....  
 Các ngoại ngữ có thể sử dụng : .....

Hình

**Ghi chú :** Tuổi nhân viên từ 18 đến 50. Nhân viên phải có khả năng sử dụng ít nhất 1 ngoại ngữ

**BM2**
**Phiếu phân công**

Họ tên : .....CMND: .....Đơn vị : .....  
 Loại công việc : .....Ngày bắt đầu : ..... Số ngày : .....  
**Ghi chú :**

- X hiện nay chỉ có 5 loại công việc, mỗi loại công việc yêu cầu một số ngoại ngữ đối với nhân viên được phân công ( nhân viên phải có khả năng sử dụng tất cả ngoại ngữ được yêu cầu của loại công việc )  
 - Giả sử tại 1 thời điểm nhân viên chỉ có thể được phân công trong tối đa 1 phiếu phân công

- **Ứng dụng phân công nhân viên gồm có các bảng sau:**

- Công ty (`cong_ty`)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	
Dien_thoai	varchar(200)	utf8_general_ci		Yes	NULL	
Dia_chi	varchar(200)	utf8_general_ci		Yes	NULL	
Tuoi_toi_thieu	int(11)			Yes	NULL	
Tuoi_toi_da	int(11)			Yes	NULL	

- Đơn vị (`don_vi`)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	

- Nhân viên (nhan\_vien)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ho_ten	varchar(200)	utf8_general_ci		Yes	NULL	
Gioi_tinh	int(1)			Yes	NULL	
Ngay_sinh	date			Yes	NULL	
CMND	varchar(200)	utf8_general_ci		Yes	NULL	
Muc_luong	double			Yes	NULL	
Dia_chi	varchar(200)	utf8_general_ci		Yes	NULL	
ID_DON_VI	int(11)			Yes	NULL	

- Loại công việc (loai\_cong\_viec)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	

- Ngoại ngữ (ngoai\_ngu)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	

- Yêu cầu (yeu\_cau)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
ID_LOAI_CONG_VIEC	int(11)			Yes	NULL	
ID_NGOAI_NGU	int(11)			Yes	NULL	

- Khả năng (kha\_nang)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
ID_NHAN_VIEN	int(11)			Yes	NULL	
ID_NGOAI_NGU	int(11)			Yes	NULL	

- Phiếu phân công (phieu\_phan\_cong)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ngay_bat_dau	date			Yes	NULL	
So_ngay	int(11)			Yes	NULL	
ID_NHAN_VIEN	int(11)			Yes	NULL	
ID_LOAI_CONG_VIEC	int(11)			Yes	NULL	

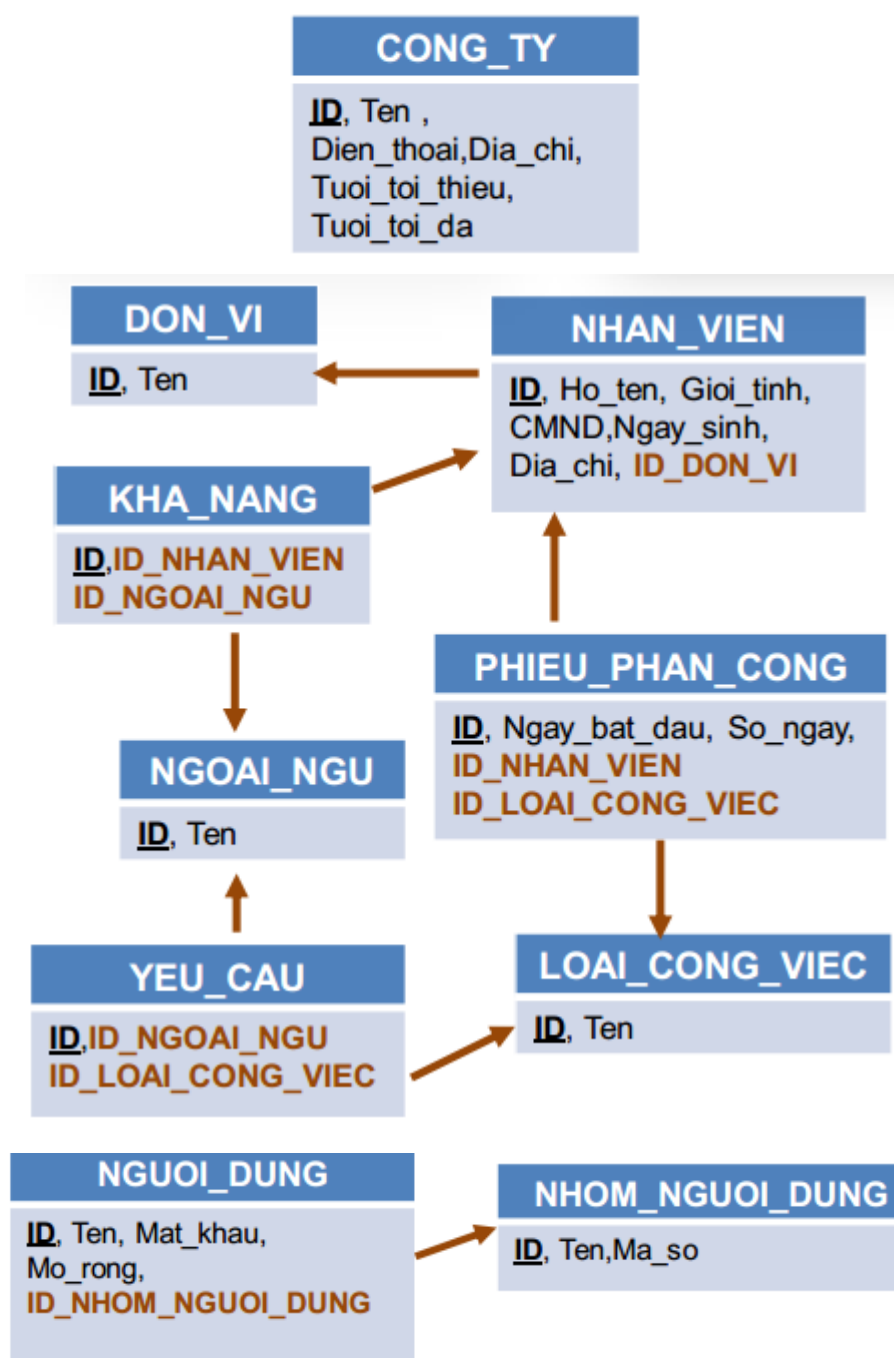
- Nhóm người dùng (nhom\_nguoi\_dung)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	
Ma_so	varchar(200)	utf8_general_ci		Yes	NULL	

- Người dùng (nguoi\_dung)

Name	Type	Collation	Attributes	Null	Default	Extra
<u>ID</u>	int(11)			No	None	AUTO_INCREMENT
Ten	varchar(200)	utf8_general_ci		Yes	NULL	
Mat_khau	varchar(200)	utf8_general_ci		Yes	NULL	
Mo_rong	varchar(200)	utf8_general_ci		Yes	NULL	
ID_NHOM_NGUOI_DUNG	int(11)			Yes	NULL	

- Môi quan hệ giữa các bảng:



▪ **Ghi chú:**

– Bảng NHOM\_NGUOI\_DUNG

Bao gồm các nhóm với tên và mã số tương ứng như sau:

Tên	Mã số
Nhân viên	NV
Quản lý đơn vị	QLDV
Quản lý công ty	QLCT

– Bảng NGUOI\_DUNG

Cột **mo\_rong**:

- Nếu nhóm tương ứng là "NV" => ID\_NHAN\_VIEN
- Nếu nhóm tương ứng là "QLDV" => ID\_DON\_VI

Dữ liệu hiện nay về người dùng với tên và mật khẩu tương ứng như sau:

Tên	Mật khẩu
NV_1	NV_1
QLDV_1	QLDV_1
QLCT_1	QLCT_1

▪ **Yêu cầu 1: Import CSDL vào MySQL**

– Import dữ liệu của CSDL **phan\_cong\_nhan\_vien\_1\_1\_nam** vào MySQL từ file dữ liệu sql được cung cấp sẵn (phan\_cong\_nhan\_vien\_1\_1\_nam.sql)

▪ **Yêu cầu 2: Thực hiện các câu lệnh truy vấn**

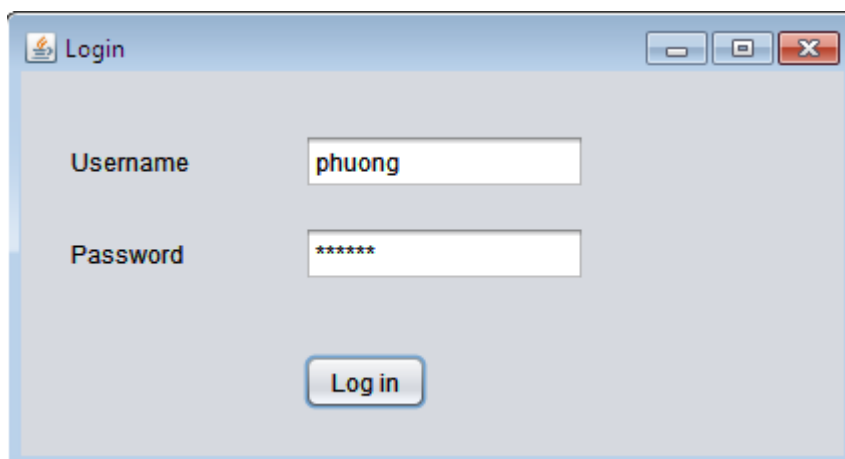
1. Liệt kê danh sách nhân viên gồm có các thông tin sau: họ tên, ngày sinh, địa chỉ.
2. Liệt kê danh sách nhân viên gồm có các thông tin sau: họ tên, CMND, mức lương. Chỉ liệt kê các nhân viên có họ tên bắt đầu là 'N'.
3. Liệt kê danh sách nhân viên sắp xếp giảm dần theo mức lương và tăng dần theo họ tên.
4. Liệt kê danh sách các nhân viên (họ tên) của đơn vị có ID=1 có thể được phân công vào loại công việc có ID=2.
5. Liệt kê danh sách các phiếu phân công trong tháng 11/2014.
6. Liệt kê danh sách các phiếu phân công trong quý 4 năm 2014 của nhân viên có tên là "Trần thanh thủy Lan".
7. Liệt kê danh sách các nhân viên mà trong họ tên có từ "Trang".
8. Liệt kê danh sách các nhân viên có mức lương từ 5.000.000 đến 8.000.000, danh sách được xếp theo thứ tự mức lương giảm dần.
9. Liệt kê danh sách các nhân viên có mức lương từ 9.000.000 trở lên.
10. Liệt kê danh sách các nhân viên có khả năng sử dụng từ 2 ngoại ngữ trở lên.



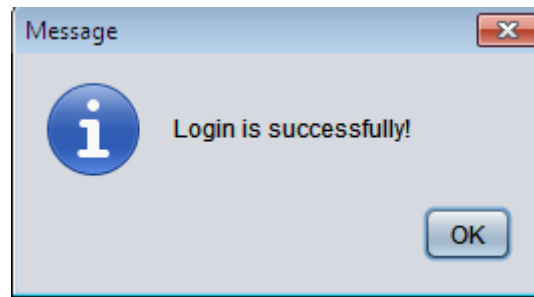
11. Liệt kê danh sách các nhân viên có khả năng sử dụng 3 thứ tiếng: tiếng Anh, Pháp, Đức.
12. Thống kê tổng số nhân viên theo từng đơn vị, gồm các thông tin: tên đơn vị, tổng số nhân viên.
13. Cho biết mức lương trung bình của các nhân viên thuộc "Đơn vị C".
14. Cho biết loại công việc nào có nhiều phiếu phân công nhất?
15. Cho biết nhân viên nào trẻ tuổi nhất của "Đơn vị D".
16. Cho biết mức lương thấp nhất, mức lương cao nhất của từng đơn vị.
17. Cho biết loại công việc nào yêu cầu cần phải biết tiếng "Brazil".
18. Cho biết loại công việc nào chỉ yêu cầu biết 1 ngoại ngữ.
19. Viết truy vấn và dựa vào kết quả truy vấn để trả lời câu hỏi sau: Có thể phân công nhân viên có ID=10 vào loại công việc có ID=1 với ngày bắt đầu là 20/1/2015 và số ngày = 2 không? Nếu câu trả lời là "Phân công được" thì không cần thực hiện gì thêm. Nếu câu trả lời là "Không phân công được" thì cho biết lý do tại sao?
20. Thêm mới 1 nhân viên vào bảng nhân viên với những thông tin sau: Trần thạch Anh, 1, 1980-10-10, 023485214, 7.900.000, 357 Lê Hồng Phong phường 2 Quận 10, 1
21. Thêm mới khả năng cho nhân viên vừa thêm: nhân viên "Trần thạch Anh" có thể sử dụng tiếng "Anh" và tiếng "Pháp".
22. Tạo một bảng tạm có tên là nhan\_vien\_don\_vi\_A có cấu trúc giống như bảng nhan\_vien và thêm các nhân viên thuộc "Đơn vị A" từ bảng nhan\_vien vào bảng này.
23. Hãy cập nhật lại mức lương của các nhân viên trong bảng nhan\_vien\_don\_vi\_A với mức lương mới bằng mức lương cũ + 5%.
24. Hãy xóa các nhân viên có năm sinh nhỏ hơn 1975 trong bảng nhan\_vien\_don\_vi\_A.
25. Hãy xóa các nhân viên trong bảng nhan\_vien\_don\_vi\_A chưa được phân công công việc nào.

## 5.2. Đăng nhập

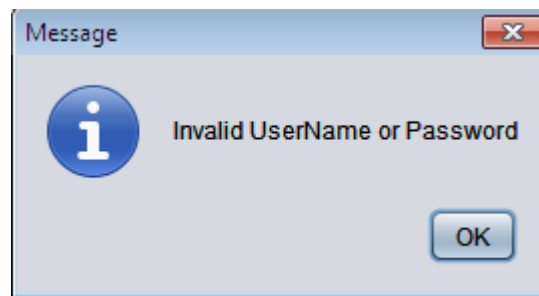
- ✓ **Yêu cầu: Xây dựng form đăng nhập**



*Form đăng nhập*



*Đăng nhập thành công*



*Đăng nhập không thành công*

✓ **Hướng dẫn sử dụng:**

- Người dùng nhập vào Username và Password. Sau đó nhấn nút “Login” để đăng nhập.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmDangNhap: JFrameForm

\* (Các thể hiện phía dưới đều nằm trong Frame)

- txtUsername: JTextField (nhập liệu)
- txtPassword: JPasswordField (nhập liệu)
- btnDangNhap: JButton (Xử lý)

▪ **Nhập:**

- Nhập username
- Nhập password

▪ **Xuất**

- Thông báo đăng nhập thành công/ thất bại

✓ **Hướng dẫn**

- Tạo CSDL có tên là **qllienhe**
- Trong CSDL **qllienhe** tạo bảng **user** như sau:

Name	Type	Collation	Attributes	Null	Default	Extra
<b>id</b>	int(11)			No	None	AUTO_INCREMENT
<b>username</b>	varchar(100)	utf8_general_ci		No	None	
<b>password</b>	varchar(100)	utf8_general_ci		No	None	

- Sau khi đã tạo bảng user, thêm một số dòng dữ liệu cho bảng này để làm việc, ví dụ:

id	username	password
1	phuong	123456
2	huy	654321
3	tung	1313131
4	khoa	123123

- Tạo project làm việc mới có tên là QuanLyLienHe
- Trong thư mục Libraries của project: add thêm thư viện MySQL JDBC Driver để làm việc với CSDL MySQL
- Trong thư mục Source Packages của project tạo package có tên là **bussiness** để chứa các lớp xử lý kết nối CSDL, làm việc với bảng trong CSDL
- Trong package bussiness:
  - Xây dựng lớp User có các thuộc tính là username và password, tạo các phương thức khởi tạo, getter và setter

```
public class User {
    public String Username;
    public String Password;

    public User() {
    }

    public User(String Username, String Password) {
        this.Username = Username;
        this.Password = Password;
    }

    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;
    }
}
```

- Xây dựng lớp QTCSDL để quản lý việc kết nối và xử lý lấy dữ liệu cần thiết cho ứng dụng. Lớp này có các thuộc tính final cần thiết cho việc kết nối

```
public class QLCSDL {
```

```
private static final String JDBC_URL = "jdbc:mysql://localhost:3306/
qllienhe?useUnicode=true&characterEncoding=utf8";

private static final String JDBC_DRIVER_CLASS = "com.mysql.jdbc.Driver";

private static final String JDBC_USERNAME = "root";

private static final String JDBC_PASSWORD = "123456";

// các phương thức cần thiết khác

}
```

- Trong lớp QLCSDL, xây dựng phương thức connect() để thực hiện việc kết nối:

```
public Connection connect() throws SQLException, ClassNotFoundException {
    Class.forName(JDBC_DRIVER_CLASS);
    Connection conn = DriverManager.getConnection(JDBC_URL, JDBC_USERNAME,
JDBC_PASSWORD);
    return conn;
}
```

- Trong lớp QLCSDL, xây dựng phương thức timUser(String username, String Password) để tìm trong bảng user của CSDL và trả về user tương ứng nếu tìm thấy, trả về null nếu không tìm thấy:

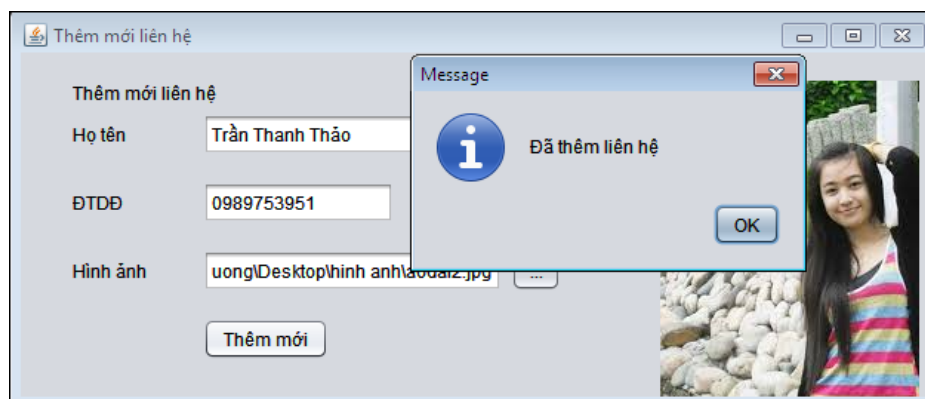
```
public User timUser(String username, String password) throws SQLException,
ClassNotFoundException{
    User user = null;
    try (Connection conn = this.connect()) {
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM user WHERE username like '" + username + "' and
password like '" + password + "'";
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            user = new User();
            user.setUsername(resultSet.getString("username"));
            user.setPassword(resultSet.getString("password"));
            break;
        }
    }
    return user;
}
```

- Trong thư mục Source Packages của project tạo package có tên là **forms** để chứa các form làm việc
- Trong package **forms**:
  - Thiết kế form frmDangNhap như trên.
  - Hiện thực phương thức btnDangNhapActionPerformed() để thực hiện việc đăng nhập và thông báo kết quả

```
private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
    QLCSDL qlcsdl = new QLCSDL();
    String username = txtUsername.getText();
    String password = txtPassword.getText();
    try {
        User user = qlcsdl.timUser(username, password);
        if(user==null){
            JOptionPane.showMessageDialog(null, "Invalid UserName or Password");
        }else{
            JOptionPane.showMessageDialog(null, "Login is successfully!");
        }
    } catch (SQLException | ClassNotFoundException | HeadlessException e) {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}
```

### 5.3. Thêm Liên hệ vào database

- ✓ **Yêu cầu: Viết chương trình thêm Liên hệ vào database**



- ✓ **Hướng dẫn sử dụng:**
  - Nhập vào các ô nhập liệu các thông tin: tên, điện thoại di động, hình ảnh. Nhấn nút “Thêm” => Các thông tin vừa nhập sẽ được lưu vào bảng lienhe trong CSDL.
- ✓ **Tóm tắt yêu cầu**
  - **Thiết kế giao diện người dùng (sử dụng giao diện đã thiết kế ở bài Giao diện – module 1):**
    - frmThemLienHe: JFrameForm
    - \* (Các thể hiện phía dưới đều nằm trong Frame)
      - txtTen: JTextField (nhập liệu)
      - txtDtdd: JTextField (nhập liệu)
      - fchHinhAnh: JFileChooser (chọn)
      - btnThem: JButton (Xử lý thêm)
  - **Nhập:**

- Tên
- ĐTDĐ
- Hình ảnh...

▪ **Xuất:**

- “Đã thêm liên hệ” nếu thêm được hoặc “Không thể thêm Liên hệ”

✓ **Thuật giải**

- Hiển thị đường dẫn file được chọn trong txtHinhAnh
- Khai báo biến hoten nhận giá trị từ txtHoTen
- Khai báo biến dtdd nhận giá trị từ txtDtdd
- Khai báo biến hinhAnh nhận giá trị từ txtHinhAnh
- Xử lý ghi vào bảng lienhe trong CSDL qllienhe thông tin liên hệ
- Xuất kết quả dưới dạng dialog thông báo

✓ **Hướng dẫn**

- Trong CSDL **qllienhe** tạo bảng **lienhe** có các cột: id, hoten, dtdd, hinhanh

Trong project QuanLyLienHe

- Trong package bussiness:
  - Xây dựng lớp LienHe có các thuộc tính là hoten, dtdd, hinhanh tạo các phương thức khởi tạo, getter và setter

```
public class LienHe {
    String hoten;
    String dtdd;
    String hinhanh;

    public LienHe(String hoten, String dtdd, String hinhanh) {
        this.hoten = hoten;
        this.dtdd = dtdd;
        this.hinhanh = hinhanh;
    }

    public LienHe() {
    }

    public String getHoten() {
        return hoten;
    }

    public void setHoten(String hoten) {
        this.hoten = hoten;
    }

    public String getDtdd() {
        return dtdd;
    }
}
```

```

public void setDtdd(String dtdd) {
    this.dtdd = dtdd;
}

public String getHinhanh() {
    return hinhanh;
}

public void setHinhanh(String hinhanh) {
    this.hinhanh = hinhanh;
}
}

```

- Trong lớp QLCSDL, xây dựng phương thức themLienHe(LienHe c) để thêm mới một liên hệ vào bảng lienhe trong CSDL:

```

public boolean themLienHe(LienHe c) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    boolean execute = false;
    try (Connection conn = this.connect()) {
        String sql = "INSERT INTO lienhe VALUES(NULL,'" + c.getHoten() + "','" + c.getDtdd() + "','"
+ c.getHinhanh() + "')";
        PreparedStatement statement = conn.prepareStatement(sql);
        execute = statement.execute();
    }
    catch (Exception ex){
        System.out.println(ex.getMessage());
    }
    return execute;
}

```

- Trong package forms, copy frmThemLienHe của module 1 – bài 1 vào và xử lý cho nút btnThem như sau:

```

private void btnThemMoiActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String hoten = txtHoTen.getText();
        String dtdd = txtDTDD.getText();
        File f = new File(txtHinhAnh.getText());
        String hinhAnh = f.getName();

        QLCSDL qlcsdl = new QLCSDL();
        LienHe c = new LienHe(hoten, dtdd, hinhAnh);
        if (qlcsdl.themLienHe(c)) {
            JOptionPane.showMessageDialog(rootPane, "Đã thêm liên hệ!");
        } else {

```

```

        JOptionPane.showMessageDialog(rootPane, "Không thêm được liên hệ!");
    }
} catch (SQLException | ClassNotFoundException | UnsupportedEncodingException ex) {
    Logger.getLogger(frmThemLienHe.class.getName()).log(Level.SEVERE, null, ex);
}
}

```

#### 5.4. Thêm sách vào bảng sách trong CSDL

- ✓ **Yêu cầu: Viết chương trình thêm sách vào bảng sách trong CSDL**

- ✓ **Hướng dẫn sử dụng:**
  - Nhập các thông tin Tên sách, Tác giả, NXB, Giá bìa. Nhấn nút “Thêm sách mới” => ghi thông tin sách vào bảng sách trong CSDL
  - Nhấn “Tiếp tục” => các thông tin đang có trên màn hình thêm sách sẽ bị xóa để người dùng tiếp tục nhập thông tin sách mới
- ✓ **Tóm tắt yêu cầu**
  - **Thiết kế giao diện người dùng:**
    - frmThemSach: JFrame
    - \* (Các thể hiện phía dưới đều nằm trong Frame)



- txtTenSach: JTextField (nhập liệu)
- txtTacGia: JTextField (nhập liệu)
- txtNXB: JTextField (nhập liệu)
- txtGiaBia: JTextField (nhập liệu)
- btnThemSach: JButton (Xử lý thêm sách vào bảng sách trong CSDL)
- btnTiepTuc: JButton (Xử lý xóa các thông tin trên màn hình)

▪ **Nhập:**

- Tên sách
- Tác giả
- NXB
- Giá bìa

▪ **Xuất:**

- Thông báo thêm sách thành công hoặc không thành công.

✓ **Hướng dẫn**

- Tạo CSDL có tên là QLSach
- Trong CSDL QLSach tạo bảng sách như sau:

Name	Type	Collation	Attributes	Null	Default	Extra
<b>id</b>	int(11)			No	None	AUTO_INCREMENT
<b>TenSach</b>	varchar(200)	utf8_general_ci		No	None	
<b>TacGia</b>	varchar(200)	utf8_general_ci		No	None	
<b>NXB</b>	varchar(200)	utf8_general_ci		No	None	
<b>GiaBia</b>	double			No	None	

Sau khi đã tạo bảng SACH, thêm một số dòng dữ liệu cho bảng này để làm việc, ví dụ:

id	TenSach	TacGia	NXB	GiaBia
1	Lá nằm trong lá	Nguyễn Nhật Ánh	NXB Trẻ	45000
2	Rừng xưa xanh lá	Bùi Ngọc Tấn	Hội nhà văn	36000
3	Ngôi trường đầu yêu	Thu Thảo	Trẻ	52000
4	Cà phê cùng Tony	Tony Buổi Sáng	Văn hóa thông tin	72000
5	Xì Xọn Online	Mao Tiểu Lạc	Mỹ Thuật	22500
6	Hoàng Tử Online	Kaivi	Văn học	69000

- Tạo project làm việc mới có tên là QuanLySach
- Trong thư mục Libraries của project: add thêm thư viện MySQL JDBC Driver để làm việc với CSDL MySQL
- Trong thư mục Source Packages của project tạo package có tên là **bussiness** để chứa các lớp xử lý kết nối CSDL, làm việc với bảng trong CSDL
- Trong package bussiness:
  - Xây dựng class Sach gồm có các thuộc tính của sách : TenSach, TacGia, NXB, GiaBia, xây dựng 2 phương thức khởi tạo : một phương thức không có tham số và một phương thức có 4 tham số là tên sách, tác giả, nhà xuất bản, giá bìa, các hàm geter và setter.

```

public class Sach {
    String TenSach;
    String TacGia;
    String NXB;
    double GiaBia;
    public Sach() {
    }
    public Sach(String TenSach, String TacGia, String NXB, double GiaBia) {
        this.TenSach = TenSach;
        this.TacGia = TacGia;
        this.NXB = NXB;
        this.GiaBia = GiaBia;
    }
    public double getGiaBia() {
        return GiaBia;
    }
    public void setGiaBia(double GiaBia) {
        this.GiaBia = GiaBia;
    }
    public String getTenSach() {
        return TenSach;
    }
    public void setTenSach(String TenSach) {
        this.TenSach = TenSach;
    }
    public String getTacGia() {
        return TacGia;
    }
    public void setTacGia(String TacGia) {
        this.TacGia = TacGia;
    }
    public String getNXB() {
        return NXB;
    }
    public void setNXB(String NXB) {
        this.NXB = NXB;
    }
}

```

- Copy lại lớp QTCSĐL đã viết ở bài trên, bỏ đi các phương thức thừa và viết thêm phương thức thêmSach(Sach s)

```
public void themSach(Sach s) throws ClassNotFoundException,
SQLException,UnsupportedEncodingException
{
    try(Connection conn = this.connect())
    {
        String sql = "INSERT INTO sach VALUES(NULL," +
        s.tenSach + "," + s.tacGia + "," + s.NXB + "," + s.giaBia + ")";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    } catch(Exception ex){
        System.out.println(ex.getMessage());
    }
}
```

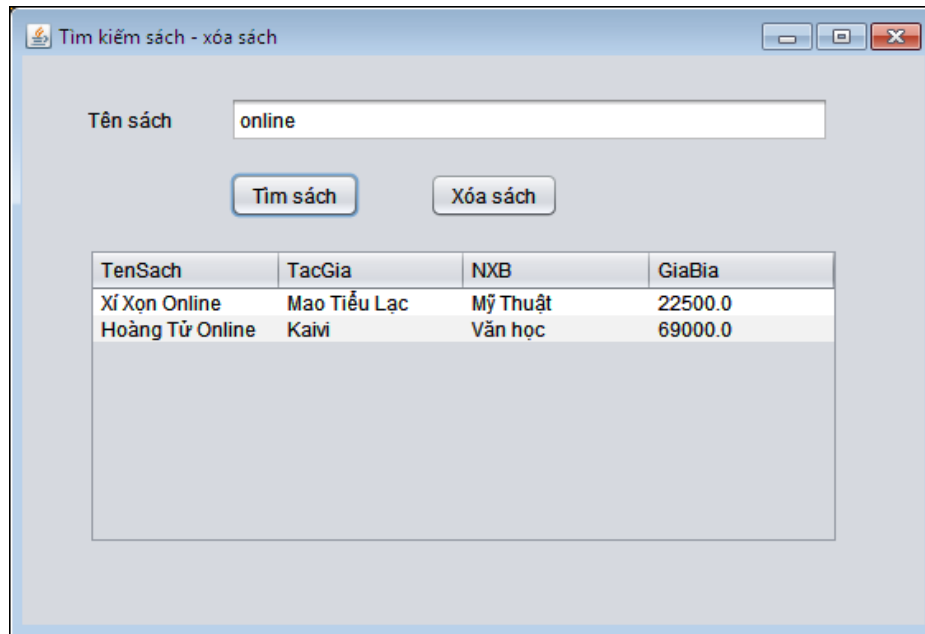
- Trong thư mục Source Packages của project tạo package có tên là forms để chứa các form của project.
  - Trong package forms, tạo frmThemSach như hình trên
  - Ở FrmThemSach, hiện thực phương thức btnThemSachActionPerformed() trong đó gọi sử dụng phương thức themSach(Sach s) đã viết trong lớp QTCSDL để thực hiện việc thêm một sách mới vào CSDL.

```
private void btThemSachActionPerformed(java.awt.event.ActionEvent evt) {
    String tensach = txtTenSach.getText();
    String tacgia = txtTacGia.getText();
    String nxb = txtNXB.getText();
    double giabia = Double.parseDouble(txtGiaBia.getText());
    try
    {
        Sach s = new Sach(tensach, tacgia, nxb, giabia);
        QLCSdl qlcsdl = new QLCSdl();
        qlcsdl.themSach(s);
        JOptionPane.showMessageDialog(rootPane,"Da Them Sach Moi!");
    }
    catch(SQLException | ClassNotFoundException | UnsupportedEncodingException ex)
    {
        JOptionPane.showMessageDialog(rootPane,ex);
    }
}
```

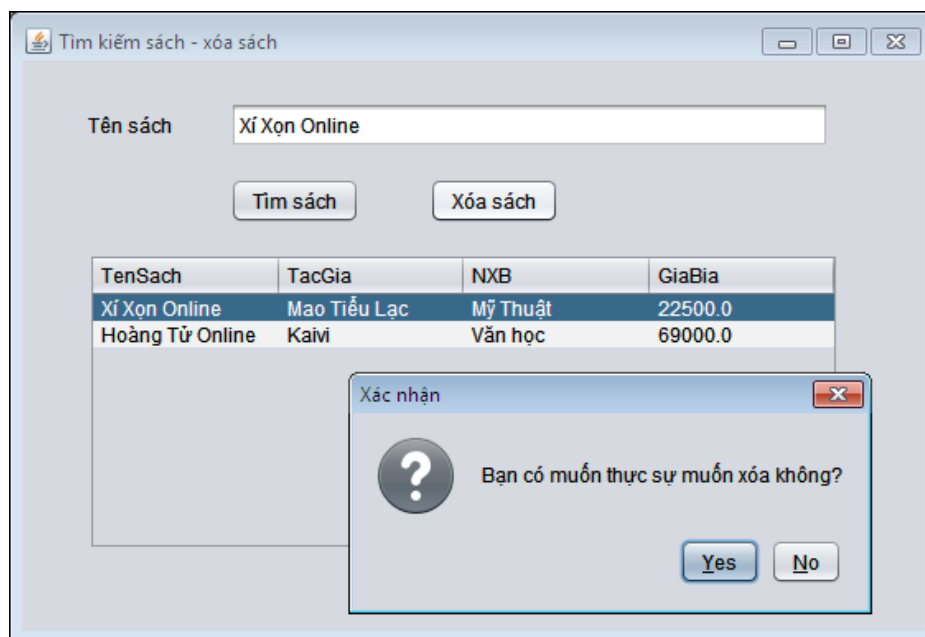
- Đối với nút "Tiếp tục", xóa bỏ nội dung của các điều khiển trên màn hình để tiếp tục thêm mới.

## 5.5. Tìm kiếm sách – xóa sách

- ✓ **Yêu cầu: Viết chương trình Tìm kiếm sách – xóa sách của bảng sach trong CSDL**



*Kết quả tìm sách với tên sách tương đối*



*Chọn một sách trên table và xóa sách được chọn*

✓ **Hướng dẫn sử dụng:**

- Nhập tên sách (tương đối), nhấn "Tìm sách" => Hiện thị danh sách tương ứng
- Chọn một sách trong danh sách, nhấn "Xóa sách" => Xóa và cập nhật lại danh sách

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmTimXoaSach: JFrame
- \* (Các thể hiện phía dưới đều nằm trong Frame)
  - txtTenSach: JTextField (nhập liệu)
  - tblSach: JTable (kết xuất, chỉ đọc)

- btnTimSach: JButton (Xử lý tìm kiếm và hiển thị kết quả)
- btnXoaSach: JButton (Xử lý xóa sách)

▪ **Nhập:**

- Tên sách

▪ **Xuất:**

- Danh sách sách ứng với tên sách

✓ **Hướng dẫn**

- Sử dụng bảng sách trong CSDL QLSach đã tạo ở trên
- Trong lớp QLCSDL
  - Xây dựng phương thức dsSach(String tenSach) để tìm tất cả các sách trong bảng sach có tên gần đúng với tenSach

```
public List<Sach> dsSach(String tenSach) throws ClassNotFoundException, SQLException
{
    List<Sach> rs;
    try(Connection conn = this.connect()){
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM sach WHERE TenSach like '%" + tenSach + "%'";
        ResultSet resultSet = statement.executeQuery(sql);
        while(resultSet.next())
        {
            Sach p = new Sach();
            p.setTenSach(resultSet.getString("TenSach"));
            p.setTacGia(resultSet.getString("TacGia"));
            p.setNXB(resultSet.getString("NXB"));
            p.setGiaBia(resultSet.getDouble("GiaBia"));
            rs.add(p);
        }
    }
    return rs;
}
```

- Xây dựng phương thức xoaSach(String tenSach) để xóa sách có tên khớp với tenSach

```
public void xoaSach(String tenSach) throws ClassNotFoundException,
SQLException,UnsupportedEncodingException
{
    try(Connection conn = this.connect())
    {
        String sql = "DELETE FROM SACH WHERE TenSach = '" + tenSach + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
    }
}
```

```
        statement.execute();
    }
}
```

- Tạo JFrame form frmTimXoaSach để thực hiện việc tìm kiếm và xóa sách.
  - Khai báo một biến toàn cục model kiểu table để chứa tạm dữ liệu trước khi hiển thị lên table
  - Khai báo biến toàn cục qlcsdl là đối tượng quản lý CSDL

```
DefaultTableModel model = new DefaultTableModel();
QLCSDL qlcsdl = new QLCSDL();
```

- Trong phương thức khởi tạo form, khởi tạo các cột cho model vừa khai báo

```
public FrmTimXoaSach() {
    ...
    model.addColumn("Tên sách");
    model.addColumn("Tác giả");
    model.addColumn("NXB");
    model.addColumn("Giá bìa");
}
```

- Xử lý sự kiện "Tìm sách": Với các sách tìm được theo tên sách, tạo một biến listSach chứa các sách này, sau đó đưa từng sách trong listSach vào model và gán model này cho Jtable tblSach đã chuẩn bị

```
private void btTimActionPerformed(java.awt.event.ActionEvent evt) {
    List<Sach> listSach;
    String tensach = txtTenSach.getText();
    try {
        listSach = qlcsdl.dsSach(tensach);
        if (listSach.size() <= 0) {
            JOptionPane.showMessageDialog(null, "Không tìm thấy sách");
        } else {
            for (Sach s : listSach) {
                model.addRow(new Object[]{s.getTenSach(), s.getTacGia(), s.getNXB(), s.getGiaBia()});
            }
            tblSach.setModel(model);
        }
    } catch (SQLException | ClassNotFoundException | HeadlessException ex) {
        JOptionPane.showMessageDialog(rootPane, "Lỗi " + ex.getMessage());
    }
}
```

- Xử lý sự kiện click chuột trên tblSach: lấy thông tin tên sách

```
private void tblSachMouseClicked(java.awt.event.MouseEvent evt) {
```

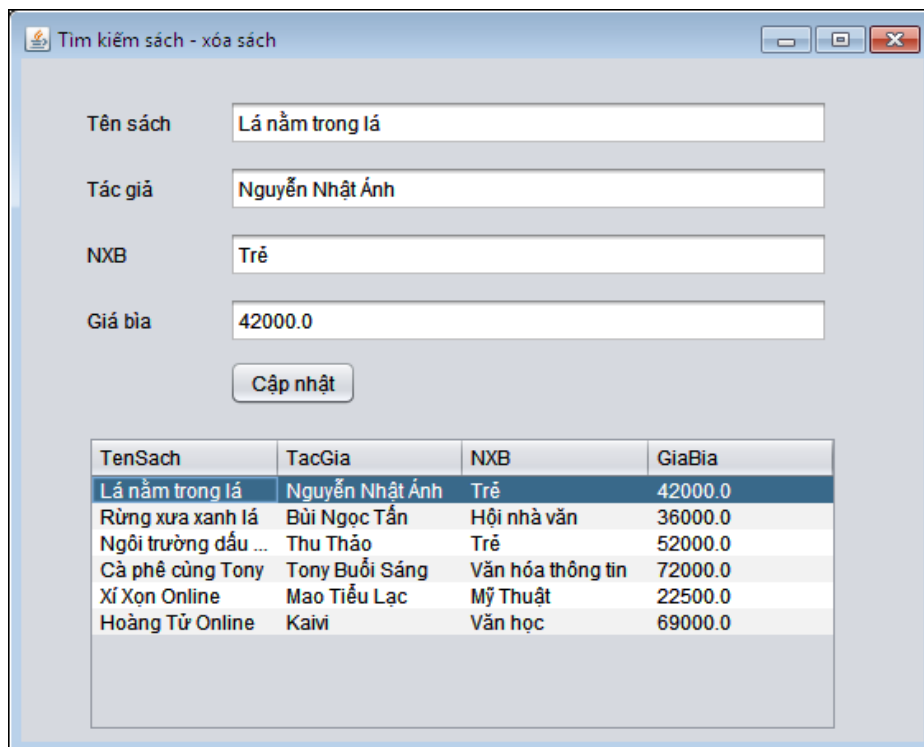
```
txtTenSach.setText(String.valueOf(model.getValueAt(tblSach.getSelectedRow(), 0)));
}
```

- Xử lý sự kiện "Xóa sách": xóa sách được chọn khỏi bảng sách trong CSDL và đồng thời xóa sách này khỏi model

```
private void btXoaActionPerformed(java.awt.event.ActionEvent evt) {
    int reply = JOptionPane.showConfirmDialog(rootPane, "Bạn có muốn xóa sách: " +
    txtTen.getText() + "?", "Confirm", JOptionPane.YES_NO_OPTION);
    if (reply == JOptionPane.YES_OPTION) {
        try {
            qlcsdl.xoaSach(txtTen.getText());
        } catch (SQLException | ClassNotFoundException | UnsupportedEncodingException ex) {
            JOptionPane.showMessageDialog(rootPane, "Lỗi " + ex.getMessage());
        }
        model.removeRow(tblSach.getSelectedRow());
        txtTen.setText("");
    }
}
```

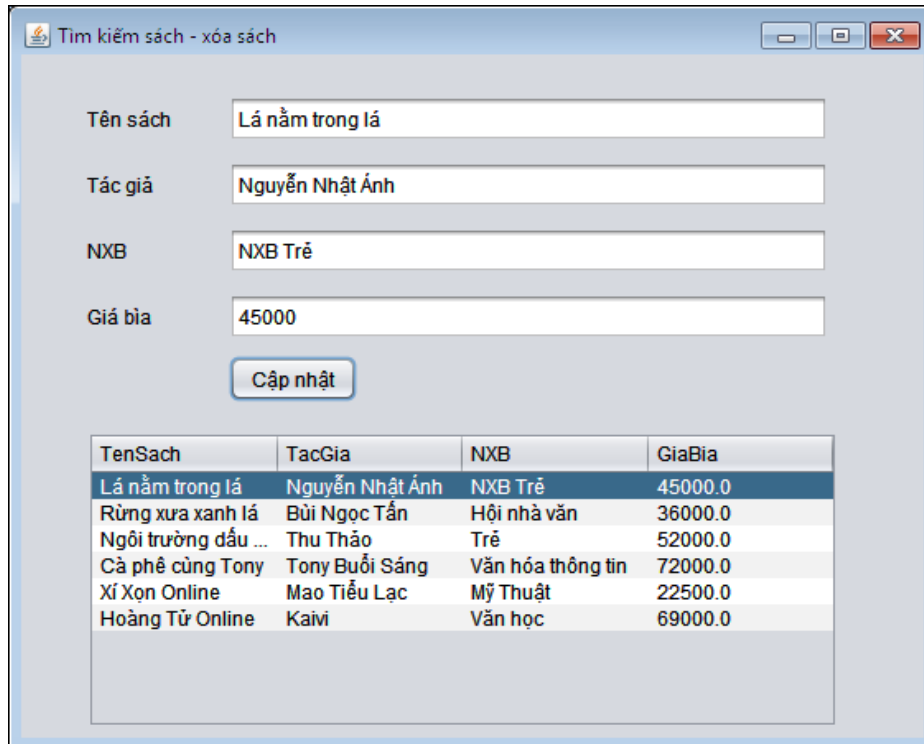
## 5.6. Cập nhật sách

- ✓ **Yêu cầu: Viết chương trình cập nhật thông tin sách của bảng sách trong CSDL**



TenSach	TacGia	NXB	GiaBia
Lá nằm trong lá	Nguyễn Nhật Ánh	Trẻ	42000.0
Rừng xưa xanh lá	Bùi Ngọc Tấn	Hội nhà văn	36000.0
Ngôi trường đầu ...	Thu Thảo	Trẻ	52000.0
Cà phê cùng Tony	Tony Buổi Sáng	Văn hóa thông tin	72000.0
Xí Xọn Online	Mao Tiểu Lạc	Mỹ Thuật	22500.0
Hoàng Tử Online	Kaivi	Văn học	69000.0

*Chọn sách cần cập nhật*



TenSach	TacGia	NXB	GiaBia
Lá nằm trong lá	Nguyễn Nhật Ánh	NXB Trẻ	45000.0
Rừng xưa xanh lá	Bùi Ngọc Tấn	Hội nhà văn	36000.0
Ngôi trường đầu ...	Thu Thảo	Trẻ	52000.0
Cà phê cùng Tony	Tony Buổi Sáng	Văn hóa thông tin	72000.0
Xí Xon Online	Mao Tiểu Lạc	Mỹ Thuật	22500.0
Hoàng Tử Online	Kaivi	Văn học	69000.0

*Cập nhật và hiển thị lại thông tin*

✓ **Hướng dẫn sử dụng:**

- Chọn sách cần cập nhật trên table, cập nhật thông tin trên text field, nhấn "Cập nhật" => Cập nhật lại thông tin sách trong CSDL và trên table.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmCapNhatSach: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- txtTenSach: JTextField (kết xuất, chỉ đọc)
- txtTacGia: JTextField (nhập liệu)
- txtNXB: JTextField (nhập liệu)
- txtGiaBia: JTextField (nhập liệu)
- tblSach: JTable (kết xuất, chỉ đọc)
- btnCapNhat: JButton (Xử lý cập nhật sách trong CSDL và trên bảng)

▪ **Nhập:**

- Chọn sách trên table

▪ **Xuất:**

- Thông tin sách được chọn

✓ **Hướng dẫn**

- Sử dụng bảng sách trong CSDL QLSach đã tạo ở trên
- Trong lớp QLCSL
  - Xây dựng phương thức dsSach() để lấy tất cả các sách trong bảng sach

```
public List<Sach> dsSach() throws ClassNotFoundException, SQLException
```



```
{
    List<Sach> rs;
    try(Connection conn = this.connect()){
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM SACH";
        ResultSet resultSet = statement.executeQuery(sql);
        while(resultSet.next())
        {
            Sach p = new Sach();
            p.setTenSach(resultSet.getString("TenSach"));
            p.setTacGia(resultSet.getString("TacGia"));
            p.setNXB(resultSet.getString("NXB"));
            p.setGiaBia(resultSet.getDouble("GiaBia"));
            rs.add(p);
        }
    }
    return rs;
}
```

- Xây dựng phương thức capNhatSach(Sach s) để cập nhật sách có tên sách khớp với tên sách của đối tượng s

```
public void capnhatSach(Sach s) throws ClassNotFoundException, SQLException,
UnsupportedEncodingException {
    try (Connection conn = this.connect()) {
        String sql = "UPDATE sach SET TacGia = '" + s.getTacGia() + "', NXB = '" + s.getNXB() + "',
        GiaBia = '" + s.getGiaBia() + "' WHERE TenSach = '" + s.getTenSach() + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    }
}
```

- Tạo JFrame form frmCapNhatSach để thực hiện việc cập nhật sách.
  - Khai báo một biến toàn cục model kiểu table để chứa tạm dữ liệu trước khi hiển thị lên table
  - Khai báo biến toàn cục qlcsdl là đối tượng quản lý CSDL

```
DefaultTableModel model = new DefaultTableModel();
QLCSDL qlcsdl = new QLCSDL();
```

- Trong phương thức khởi tạo form, khởi tạo các cột cho model vừa khai báo, đọc cách sách trong bảng sách và đưa vào model sau đó gán model cho tblSach đã chuẩn bị

```
public frmCapNhatSach() throws ClassNotFoundException, SQLException {
    initComponents();
```

```

model.addColumn("Tên sách");
model.addColumn("Tác giả");
model.addColumn("NXB");
model.addColumn("Giá bìa");

List<Sach> listSach = qlcsdl.dsSach();
if (listSach.size() <= 0) {
    JOptionPane.showMessageDialog(null, "Không có sách nào!");
} else {
    for (Sach s : listSach) {
        model.addRow(new Object[]{s.getTenSach(), s.getTacGia(), s.getNXB(), s.getGiaBia()});
    }
}
tblSach.setModel(model);
}

```

- Xử lý sự kiện click chuột trên tblSach: lấy thông tin tên sách

```

private void tblSachMouseClicked(java.awt.event.MouseEvent evt) {
    txtTenSach.setText(String.valueOf(model.getValueAt(tblSach.getSelectedRow(), 0)));
    txtTacGia.setText(String.valueOf(model.getValueAt(tblSach.getSelectedRow(), 1)));
    txtNXB.setText(String.valueOf(model.getValueAt(tblSach.getSelectedRow(), 2)));
    txtGiaBia.setText(String.valueOf(model.getValueAt(tblSach.getSelectedRow(), 3)));
}

```

- Xử lý sự kiện "Cập nhật": Lấy thông tin trên các TextField mà người dùng đã cập nhật => cập nhật thông tin sách trong bảng sách và gán lại các giá trị này cho dòng trên table

```

private void btCapNhatActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String tenSach = txtTenSach.getText();
    String tacGia = txtTacGia.getText();
    String NXB = txtNXB.getText();
    double giaBia = Double.parseDouble(txtGiaBia.getText());
    Sach s = new Sach(tenSach, tacGia, NXB, giaBia);
    try {
        qlcsdl.capnhatSach(s);
        model.setValueAt(s.tenSach, tblSach.getSelectedRow(), 0);
        model.setValueAt(s.tacGia, tblSach.getSelectedRow(), 1);
        model.setValueAt(s.NXB, tblSach.getSelectedRow(), 2);
        model.setValueAt(s.giaBia, tblSach.getSelectedRow(), 3);
    } catch (SQLException | ClassNotFoundException | UnsupportedEncodingException ex) {
    }
}

```

```

        JOptionPane.showMessageDialog(rootPane, "Lỗi " + ex.getMessage());
    }
}

```

# BÀI 6: Biểu thức chính quy, Quốc tế hóa – Địa phương hóa



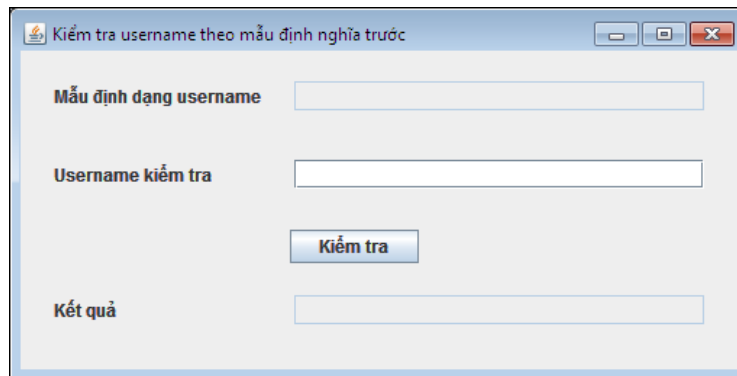
*Mục tiêu chính:*

- Giúp cho học viên có thể làm việc với biểu thức chính quy
- Xây dựng ứng dụng đa ngôn ngữ

## 6.1. Username format

✓ **Yêu cầu: Xây dựng chương trình kiểm tra username nhập vào.**

- Học viên xây dựng mẫu biểu thức chính quy của username. Sau đó sử dụng mẫu định dạng này để kiểm tra và thông báo cho người sử dụng biết username mà người sử dụng nhập vào có hợp lệ hay không.



✓ **Hướng dẫn sử dụng:**

- Nhập vào username. Nhấn nút "Kiểm tra" => Hiển thị kết quả: Username hợp lệ/ không hợp lệ

✓ **Tóm tắt yêu cầu**

### ▪ Thiết kế giao diện người dùng:

- frmKTUsername: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- txtBieuThuc: JTextField (kết xuất, chỉ đọc)
- txtUsername: JTextField (nhập liệu)
- txtKetQua: JTextArea (kết xuất, chỉ đọc)
- btnKiemTra: JButton (xử lý kiểm tra username so với biểu thức chính quy và hiển thị)

### ▪ Nhập:

- Username

### ▪ Xuất:

- Username hợp lệ/ không hợp lệ

### ▪ Quy tắc xử lý :

- Định dạng biểu thức Username hợp lệ: dài từ 6 – 20 ký tự gồm có ký tự thường, ký số, ký tự "\_", ký tự "-".

✓ **Hướng dẫn**

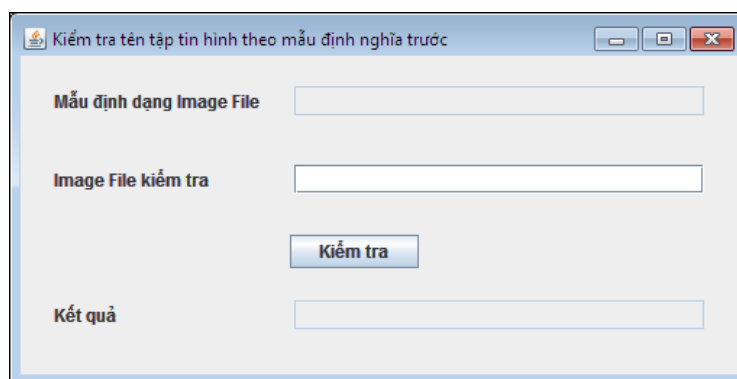
- Ví dụ một số username hợp lệ:
  - npkhuong34
  - dumien\_2002
  - mekong-2002
  - hoclt-4\_t3h
- Ví dụ một số username không hợp lệ
  - Ktp - ngắn hơn 6 ký tự
  - ktp@nguyen - ký tự @ không được sử dụng
  - nguyenthithutrang\_123456789\_- - dài hơn 20 ký tự
- Trong form frmKTUsername:
  - Thiết lập giá trị ban đầu cho txtBieuThuc là mẫu username theo quy tắc xử lý nêu trên.
  - Hiện thực phương thức btnKiemTraActionPerformed() để kiểm tra giữa giá trị người dùng nhập và mẫu như sau:

```
Pattern p = Pattern.compile(txtBieuThuc.getText());
Matcher m = p.matcher(txtUsername.getText());
if(m.matches()){
    txtKetQua.setText("User name hợp lệ");
}
else txtKetQua.setText("Username không hợp lệ");
```

## 6.2. Image file format

### ✓ Yêu cầu: Xây dựng chương trình kiểm tra tên tập tin hình nhập vào.

- Học viên xây dựng mẫu biểu thức chính quy của định dạng tên tập tin hình. Sau đó sử dụng mẫu định dạng này để kiểm tra và thông báo cho người sử dụng biết image file mà người sử dụng nhập vào có hợp lệ hay không.



### ✓ Hướng dẫn sử dụng:

- Nhập vào tên tập tin hình. Nhấn nút "Kiểm tra" => Hiện thị kết quả: Tên tập tin hình hợp lệ/ không hợp lệ

### ✓ Tóm tắt yêu cầu

- **Thiết kế giao diện người dùng:**
  - frmKTTenTapTin: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- txtBieuThuc: JTextField (kết xuất, chỉ đọc)
- txtTenTapTin: JTextField (nhập liệu)
- txtKetQua: JTextArea (kết xuất, chỉ đọc)
- btnKiemTra: JButton (xử lý kiểm tra tên tập tin hình so với biểu thức chính quy và hiển thị)
- **Nhập:**
  - Tên tập tin hình
- **Xuất:**
  - Tên tập tin hình hợp lệ/ không hợp lệ
- **Qui tắc xử lý :**
  - Định dạng tên tập tin hình hợp lệ: là chuỗi không có khoảng trắng, tiếp đó là dấu "." Và cuối cùng là phần mở rộng "jpg" hoặc "png" hoặc "gif" hoặc "bmp". Phần mở rộng không phân biệt chữ hoa, chữ thường.

✓ **Hướng dẫn**

- Ví dụ một số tên tập tin hình hợp lệ:
  - "a.jpg", "a.gif", "a.png", "a.bmp",
  - ".jpg", ".gif", ".png", ".bmp",
  - "a.JPG", "a.GIF", "a.PNG", "a.BMP",
  - "a.JpG", "a.GiF", "a.PnG", "a.BmP",
  - "jpg.jpg", "gif.gif", "png.png", "bmp.bmp"
- Ví dụ một số tên tập tin hình không hợp lệ
  - ".jpg", ".gif", ".png", ".bmp" – không có tên tập tin
  - ".jpg", ".gif", ".png", ".bmp" – không cho phép tên tập tin có khoảng trắng
  - "a.txt", "a.exe", "a.", "a.mp3" - phần mở rộng không dành cho tập tin hình ảnh
  - "jpg", "gif", "png", "bmp" – thiếu phần tên tập tin và dấu chấm.
- Phần source code học viên thực hiện tương tự như bài trên

### 6.3. Date format

✓ **Yêu cầu: Xây dựng chương trình kiểm tra ngày/tháng/năm nhập vào.**

- Học viên xây dựng mẫu biểu thức chính quy của định dạng dd/mm/yyyy. Sau đó sử dụng mẫu định dạng này để kiểm tra và thông báo cho người sử dụng biết giờ mà người sử dụng nhập vào có hợp lệ hay không.

✓ **Hướng dẫn sử dụng:**

- Nhập vào ngày tháng năm. Nhấn nút “Kiểm tra” => Hiển thị kết quả: Ngày hợp lệ/ không hợp lệ

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmKTNgayThangNam: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- txtBieuThuc: JTextField (kết xuất, chỉ đọc)
- txtNgayThangNam: JTextField (nhập liệu)
- txtKetQua: JTextArea (kết xuất, chỉ đọc)
- btnKiemTra: JButton (xử lý kiểm tra giờ so với biểu thức chính quy và hiển thị)

▪ **Nhập:**

- Ngày tháng năm

▪ **Xuất:**

- Ngày hợp lệ/ không hợp lệ

▪ **Qui tắc xử lý :**

- Định dạng ngày hợp lệ: dd/mm/yyyy và các quy tắc về ngày
- Gợi ý: `(0?[1-9]|[12][0-9]|3[01])/(0?[1-9]|1[012])/((19|20)\d\d)`

✓ **Thuật giải**

- Không có

✓ **Hướng dẫn**

- Ví dụ một số ngày hợp lệ:

- “1/1/2010” , “01/01/2020” , “31/1/2010” , “31/01/2020” , “29/2/2008” , “29/02/2008” ,  
“28/2/2009” , “28/02/2009” , “31/3/2010” ,  
“31/8/2010” , “31/08/2010” , “30/9/2010” , “30/09/2010”

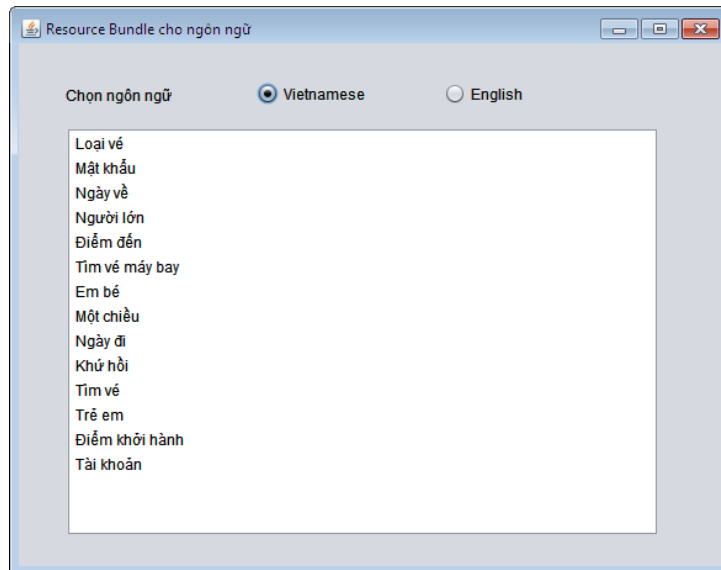
- Ví dụ một số ngày không hợp lệ

- “32/1/2010” , “32/01/2020” – ngày vượt quá [1-31]
- “1/13/2010” – tháng vượt quá [1-12]
- “29/a/2008” , “a/02/2008” , “333/2/2008” , “29/02/200a” – tháng không hợp lệ, ngày không hợp lệ

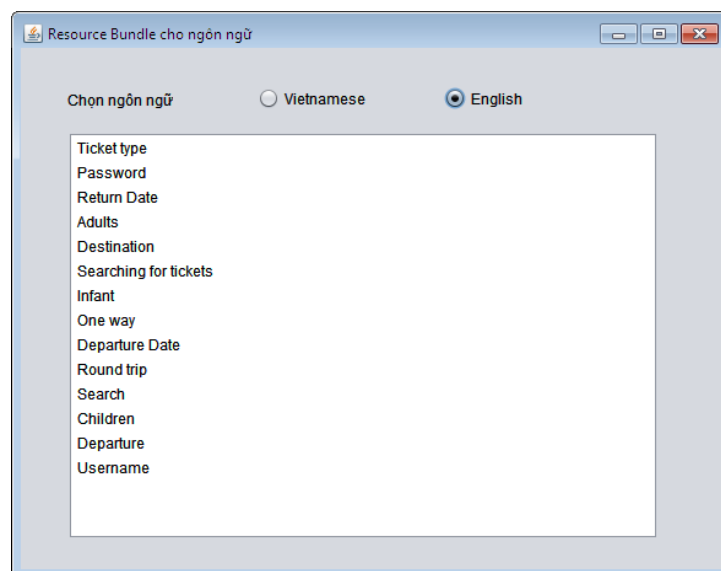
- Phần source code học viên thực hiện tương tự như bài trên

## 6.4. Đa ngôn ngữ

- ✓ **Yêu cầu:** Viết một ứng dụng đọc resource bundle cho ngôn ngữ tiếng Việt và tiếng Anh như sau:



*Chọn tiếng Việt*



*Chọn tiếng Anh*

✓ **Hướng dẫn sử dụng:**

- Chọn ngôn ngữ => Hiển thị nội dung đúng với ngôn ngữ được chọn

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmResource: JFrame (chính thuộc tính defaultCloseOperation của JFrame là DISPOSE)

\* (Các thể hiện phía dưới đều nằm trong form)

- btnLang: Button Group
  - rbtViet: Radio Button (chọn)
  - rbtEng: Radio Button (chọn)
- lstResult: List (kết xuất)

▪ **Nhập:**

- Việt/ Anh



▪ **Xuất:**

- Nội dung trong file resource bundle tương ứng

✓ **Hướng dẫn**

- Tạo 2 file resource cho tiếng Việt và tiếng Anh và lưu trong package làm việc:

```
# File resource resource_vi_VN.properties
user=Tài khoản
pass=Mật khẩu
title=Tìm vé máy bay
ticket_type=Loại vé
roundtrip=Khứ hồi
oneway=Một chiều
departure=Điểm khởi hành
destination=Điểm đến
fromDate=Ngày đi
toDate=Ngày về
adults=Người lớn
children=Trẻ em
infant=Em bé
search=Tìm vé
```

```
# File resource resource_en_US.properties
user=Username
pass=Password
title=Searching for tickets
ticket_type=Ticket type
roundtrip=Round trip
oneway=One way
departure=Departure
destination=Destination
fromDate=Departure Date
toDate=Return Date
adults= Adults
children=Children
infant=Infant
search=Search
```

- Xử lý hiển thị khi người dùng chọn “English”

```
private void rdbAnhActionPerformed(java.awt.event.ActionEvent evt) {
    String language = "en";
```

```

        String country = "US";

        Locale currentLocal = new Locale(language, country);

        ResourceBundle message;

        message = ResourceBundle.getBundle("Bai6/resource_en_US",
currentLocal);

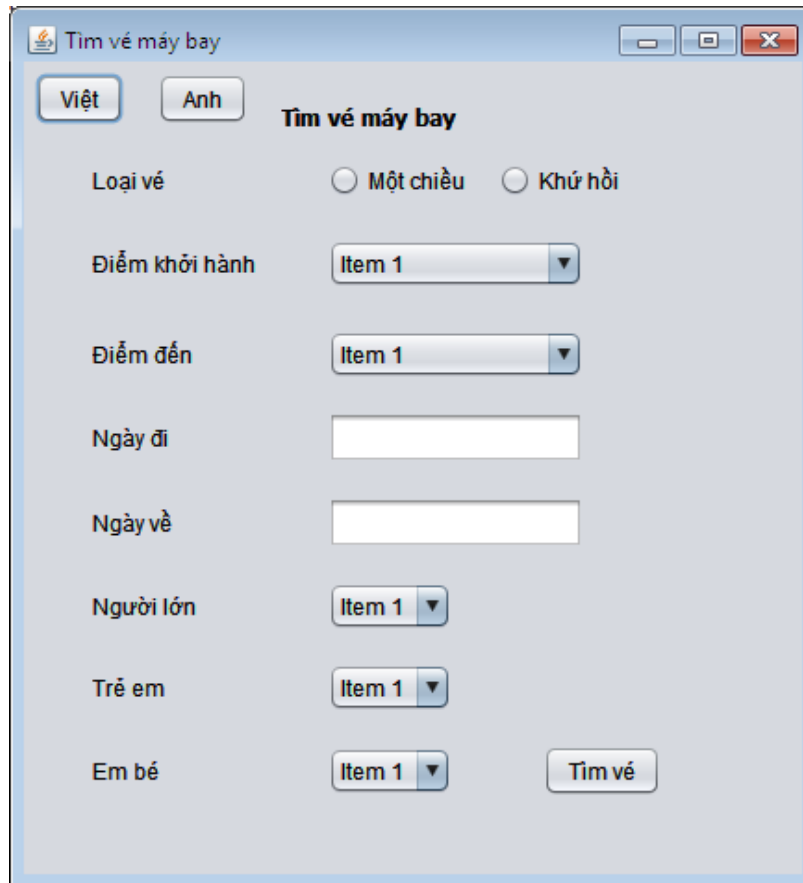
        DefaultListModel list = new DefaultListModel();
        Enumeration <String> keys = message.getKeys();
        while (keys.hasMoreElements()) {
            String key = keys.nextElement();
            String value = message.getString(key);
            String kq = value;
            list.addElement(kq);
        }
        lstResult.setModel(list);
    }

```

- Thực hiện tương tự khi người dùng chọn "Vietnamese"

## 6.5. Form tìm kiếm vé máy bay

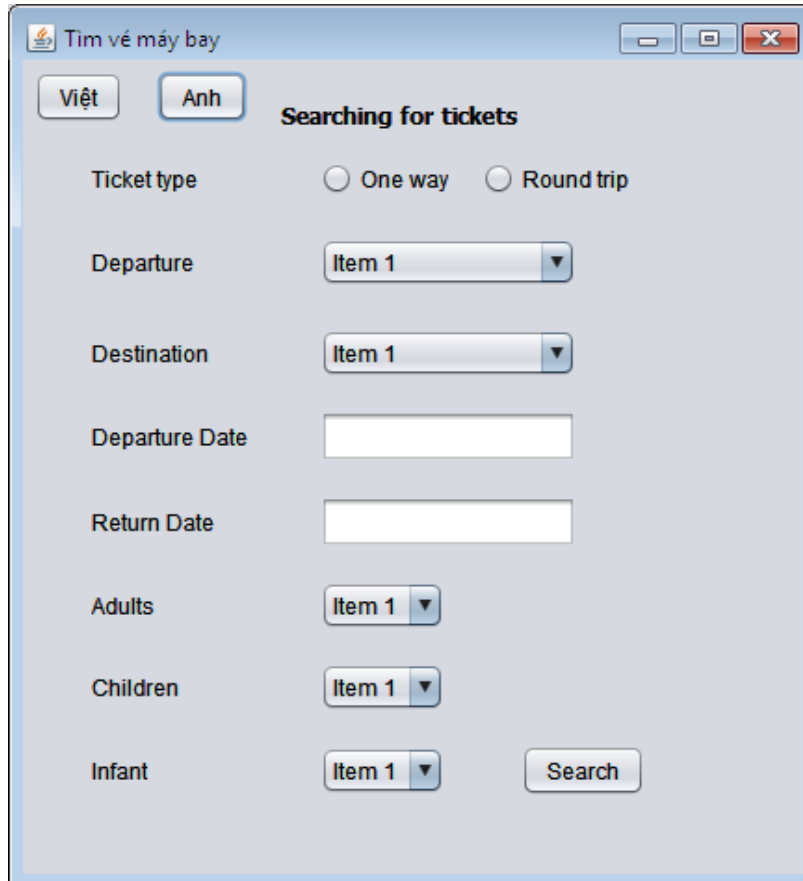
- ✓ **Yêu cầu:** Viết một ứng dụng đọc resource bundle cho ngôn ngữ tiếng Việt và tiếng Anh để hiển thị form "Tìm vé máy bay" theo ngôn ngữ được chọn:



The screenshot shows a web form titled "Tìm vé máy bay" (Find flight). At the top, there are two buttons: "Việt" (selected) and "Anh". Below the buttons, the form contains the following fields and controls:

- Loại vé** (Ticket type): Two radio buttons, "Một chiều" (One way) and "Khứ hồi" (Round trip).
- Điểm khởi hành** (Departure): A dropdown menu showing "Item 1".
- Điểm đến** (Destination): A dropdown menu showing "Item 1".
- Ngày đi** (Departure Date): A text input field.
- Ngày về** (Return Date): A text input field.
- Người lớn** (Adults): A dropdown menu showing "Item 1".
- Trẻ em** (Children): A dropdown menu showing "Item 1".
- Em bé** (Infant): A dropdown menu showing "Item 1".
- Tìm vé** (Find ticket): A button at the bottom right.

*Form mặc định/ Chọn Button "Việt"*



The screenshot shows the same web form titled "Tìm vé máy bay" (Find flight), but with the "Anh" button selected. The form content is identical to the previous one, but the text is in English:

- Ticket type**: Two radio buttons, "One way" and "Round trip".
- Departure**: A dropdown menu showing "Item 1".
- Destination**: A dropdown menu showing "Item 1".
- Departure Date**: A text input field.
- Return Date**: A text input field.
- Adults**: A dropdown menu showing "Item 1".
- Children**: A dropdown menu showing "Item 1".
- Infant**: A dropdown menu showing "Item 1".
- Search**: A button at the bottom right.

*Chọn Button "Anh"*

✓ **Hướng dẫn sử dụng:**

- Chọn “Việt” hoặc “Anh” => Hiện thị form bằng ngôn ngữ đã chọn

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmBookingTicket: JFrame (chính thuộc tính defaultCloseOperation của JFrame là DISPOSE)

\* (Các thể hiện phía dưới đều nằm trong form)

- btnViet: Button (xử lý hiển thị form tiếng Việt)
- btnAnh: Button (xử lý hiển thị form tiếng Anh)
- rbt1Chieu: Radio Button (chọn)
- rbtKH: Radio Button (chọn)
- cbbDKH: Combo Box (chọn)
- cbbDD: Combo Box (chọn)
- txtNgayDi: Text Field (nhập liệu)
- txtNgayVe: Text Field (kết xuất)
- cbbNL: Combo Box (chọn)
- cbbTE: Combo Box (chọn)
- cbbEB: Combo Box (chọn)
- btnTimVe: Button

✓ **Hướng dẫn**

- Sử dụng lại 2 file resource của bài trên
- Trong frmBookingTicket, sử dụng Resource Bundle để lấy và hiển thị các value có trong tập tin resource\_vi\_VN.properties và gán những value này cho các điều khiển tương ứng trên màn hình
- Gợi ý: Sử dụng phương thức : message.getString("từ khóa")
- Ví dụ: lblTitle.setText(message.getString("Adults"));
- Hiện thực phương thức btnVietActionPerformed() để thực hiện Việt hóa:

```
String language = "vi";
String country = "VN";
Locale currentLocal = new Locale(language, country);
ResourceBundle message;
message = ResourceBundle.getBundle("Bai6/resource_vi_VN", currentLocal);

lblTitle.setText(message.getString("title"));
lblLoaiVe.setText(message.getString("ticket_type"));
```

```

rbt1Chieu.setText(message.getString("oneway"));
rbtKH.setText(message.getString("roundtrip"));
lblKH.setText(message.getString("departure"));
lblDiemDen.setText(message.getString("destination"));
lblNgayDi.setText(message.getString("fromDate"));
lblNgayVe.setText(message.getString("toDate"));
lblNguoiLon.setText(message.getString("adults"));
lblTreEm.setText(message.getString("children"));
lblEmBe.setText(message.getString("infant"));
btnTim.setText(message.getString("search"));

```

- Làm tương tự cho Anh hóa
- **Hoặc** HV có thể viết một phương thức để hiển thị kết quả dùng gọi sử dụng cho cả phương thức khởi tạo, chọn nút “Việt”, chọn nút “Anh” như sau:

```

private void DaNgonNgu(String language, String country, String path){
    Locale currentLocal = new Locale(language, country);
    ResourceBundle message;
    message = ResourceBundle.getBundle(path, currentLocal);

    lblTitle.setText(message.getString("title"));
    lblLoaiVe.setText(message.getString("ticket_type"));
    rbt1Chieu.setText(message.getString("oneway"));
    rbtKH.setText(message.getString("roundtrip"));
    lblKH.setText(message.getString("departure"));
    lblDiemDen.setText(message.getString("destination"));
    lblNgayDi.setText(message.getString("fromDate"));
    lblNgayVe.setText(message.getString("toDate"));
    lblNguoiLon.setText(message.getString("adults"));
    lblTreEm.setText(message.getString("children"));
    lblEmBe.setText(message.getString("infant"));
    btnTim.setText(message.getString("search"));
}

```

- Lúc này, hiện thực phương thức btnAnhActionPerformed() để thực hiện Anh hóa sẽ đơn giản như sau:



```
DaNgonNgu("en", "US", "Bai6/resource_en_US");
```

# BÀI 7: Design Pattern



*Mục tiêu chính:*

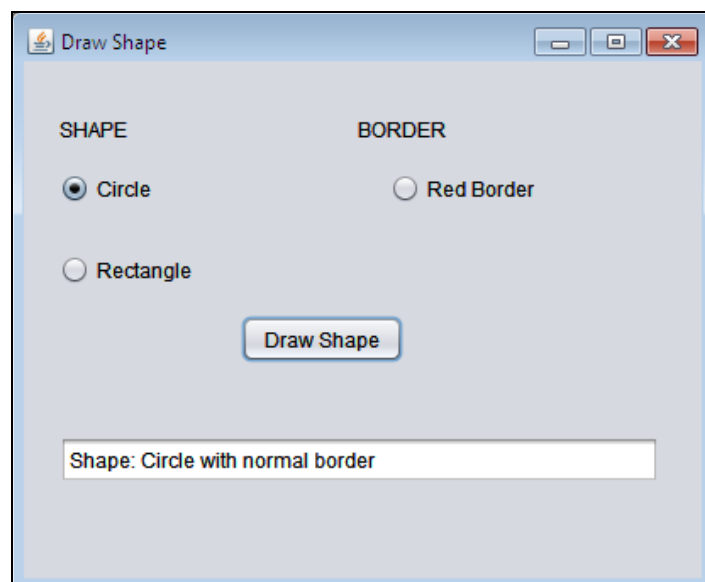
*Giúp cho học viên hiểu và sử dụng pattern trong thiết kế mô hình cho ứng dụng*

## 7.1. Vẽ hình (Draw Shape)

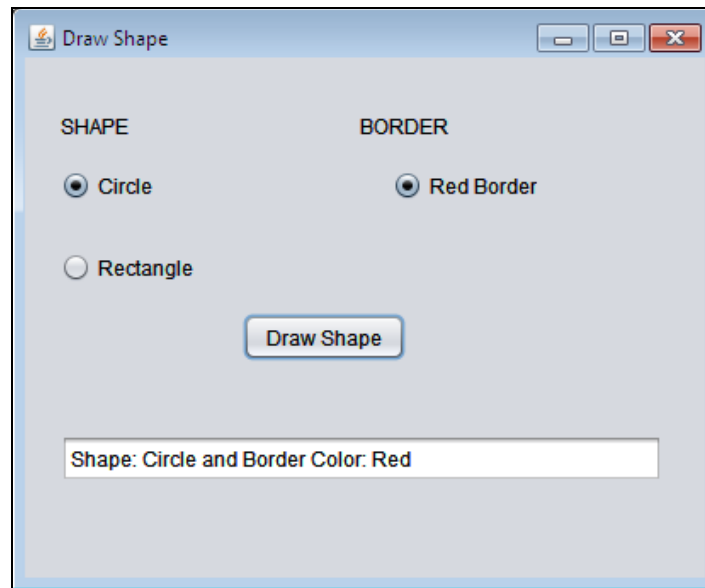
✓ **Yêu cầu:** Xây dựng chương trình mô phỏng việc vẽ hình (Draw Shape) như sau:

Chương trình mô tả như sau:

- Chương trình được xây dựng để vẽ hình và trang trí cho hình vẽ.
- Sẽ có hai hình để vẽ là hình tròn và hình chữ nhật với đường viền thông thường (sau này có thể mở rộng bài toán với nhiều hình khác).
- Sẽ có một cách trang trí cho hình là tô màu đường viền của hình vẽ là màu đỏ (sau này có thể mở rộng bài toán với nhiều cách trang trí khác nhau)
- Bất kể hình nào được chọn vẽ thì đều có thể trang trí lại hình vẽ.



*Khi người dùng chọn Circle*



Draw Shape

SHAPE

☒ Circle

☐ Rectangle

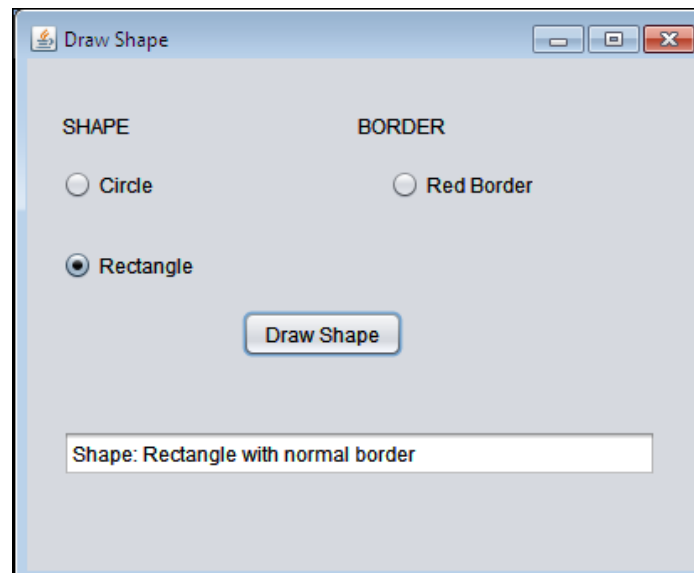
BORDER

☒ Red Border

Draw Shape

Shape: Circle and Border Color: Red

*Khi người dùng chọn Circle + Red Border*



Draw Shape

SHAPE

☐ Circle

☒ Rectangle

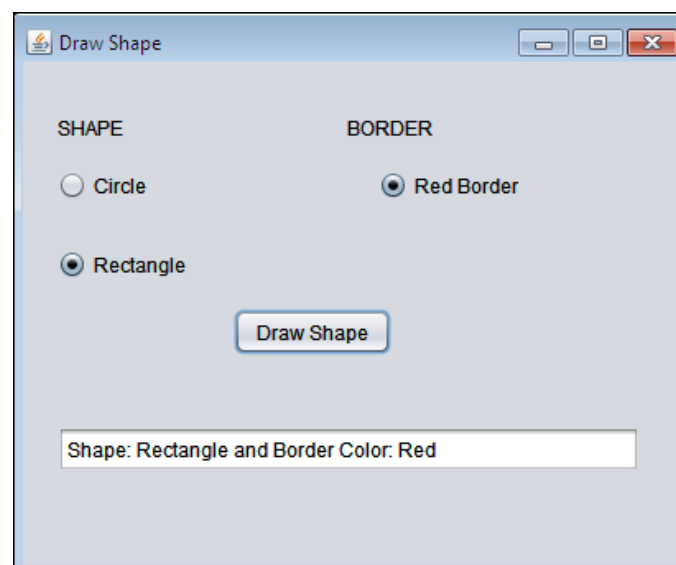
BORDER

☐ Red Border

Draw Shape

Shape: Rectangle with normal border

*Khi người dùng chọn Rectangle*



Draw Shape

SHAPE

☐ Circle

☒ Rectangle

BORDER

☒ Red Border

Draw Shape

Shape: Rectangle and Border Color: Red

*Khi người dùng chọn Rectangle + Red Border*



✓ **Hướng dẫn sử dụng:**

- Chọn một hình để vẽ trong danh sách các hình.
- Chọn hoặc không chọn trang trí
- Nhấn Button Vẽ hình

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmDecoratorPatternDemo: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- rbtCircle: JRadioButton (chọn lựa)
- rbtRectangle: JRadioButton (chọn lựa)
- grpShape: ButtonGroup (gồm có rbtCircle và rbtRectangle)
- rbtRedBorder: JRadioButton (chọn lựa)
- txtResult: JTextField (kết xuất)
- btnDraw: JButton (xử lý vẽ hình)

▪ **Nhập:**

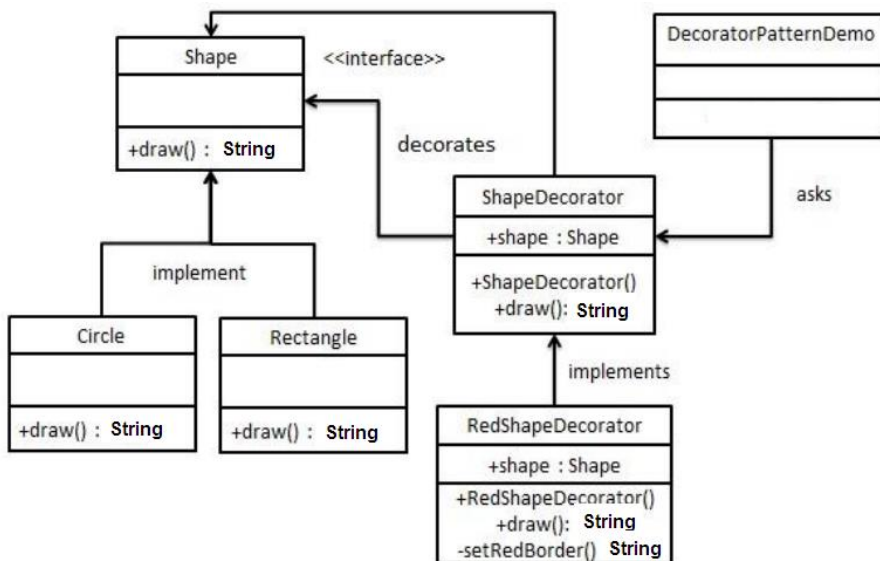
- Không nhập, chỉ chọn/hoặc không chọn các radio button trên màn hình.

▪ **Xuất:**

- Nếu người dùng chọn hình vẽ mà không chọn trang trí thì hiển thị thông tin sau:
  - Shape : Circle/Rectangle with normal border.
- Nếu người dùng chọn hình vẽ và chọn trang trí thì hiển thị thông tin sau:
  - Shape : Circle/Rectangle and Red Border: Red

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Decorator Pattern:



- Tạo interface Shape

```
public interface Shape {
    String draw();
}
```

```
}
```

- Tạo lớp Circle là lớp cụ thể implement interface Shape

```
public class Circle implements Shape{
    @Override
    public String draw() {
        return "Shape: Circle";
    }
}
```

- Tạo lớp Rectangle là lớp cụ thể implement interface Shape

```
public class Rectangle implements Shape{
    @Override
    public String draw() {
        return "Shape: Rectangle";
    }
}
```

- Sau đó tạo lớp abstract decorator ShapeDecorator implement interface Shape và có đối tượng Shape là biến thực thể (instance variable)

```
public abstract class ShapeDecorator implements Shape {
    protected Shape shape;
    public ShapeDecorator(Shape shape) {
        this.shape = shape;
    }
    public Shape getShape() {
        return shape;
    }
    public void setShape(Shape shape) {
        this.shape = shape;
    }
    @Override
    public String draw(){
        return shape.draw();
    }
}
```

- Tạo lớp RedShapeDecorator là lớp cụ thể implementing ShapeDecorator.

```
public class RedShapeDecorator extends ShapeDecorator{
    public RedShapeDecorator(Shape shape) {
        super(shape);
    }
}
```

```

    }
    @Override
    public String draw() {
        return this.shape.draw() + " and " + setRedBorder(shape);
    }
    private String setRedBorder(Shape Shape){
        return "Border Color: Red";
    }
}

```

- Tạo form frmDecoratorPatternDemo, là lớp demo sẽ sử dụng RedShapeDecorator để trang trí các đối tượng Shapes.

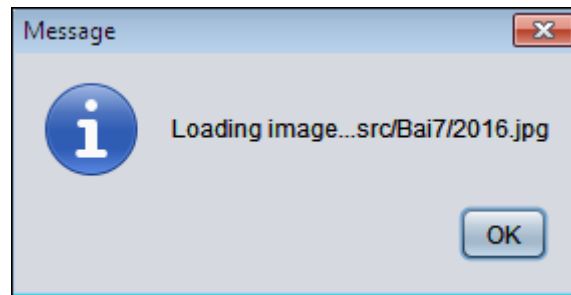
```

private void btnDrawActionPerformed(java.awt.event.ActionEvent evt) {
    String result = "";
    if (btgShape.getSelection().equals(rbtCircle.getModel())) {
        Circle c = new Circle();
        result = c.draw();
        if (rbtRedBorder.isSelected() == true) {
            Shape red = new RedShapeDecorator(c);
            result = red.draw();
        }
        else{
            result+= " with normal border";
        }
    }
    if (btgShape.getSelection().equals(rbtRectangle.getModel())) {
        Rectangle r = new Rectangle();
        result = r.draw();
        if (rbtRedBorder.isSelected() == true) {
            Shape red = new RedShapeDecorator(r);
            result = red.draw();
        }
        else{
            result+= " with normal border";
        }
    }
    txtResult.setText(result);
}

```

## 7.2. Proxy Image

- ✓ **Yêu cầu:** Xây dựng chương trình hiển thị hình ảnh như sau:



Khi đang tải



Sau khi tải xong

- Chương trình hiển thị hình ảnh sau khi nó được tải lên. Thay vì tải hình ảnh trực tiếp, hãy sử dụng lớp ImageProxy để trì hoãn việc tải cho đến khi việc tải hình ảnh hoàn thành.

✓ **Tóm tắt yêu cầu**

- **Thiết kế giao diện người dùng:**

- frmProxyPatternDemo: JFrame
  - \* (Các thể hiện phía dưới đều nằm trong Frame)
    - lblImage: Label (kết xuất)

- **Nhập:**

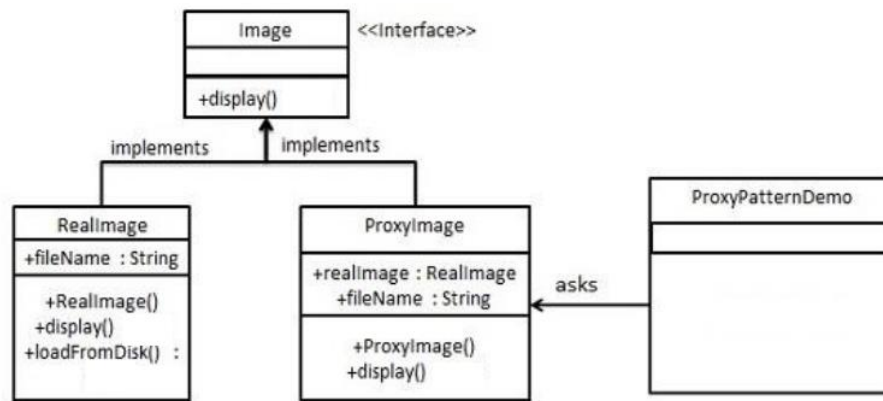
- Không

- **Xuất:**

- Hình ảnh sau khi tải xong

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Proxy Pattern:



- Tạo một interface Image:

```

public interface Image {
    ImageIcon display();
}
    
```

- Tạo lớp RealImage implements interface Image

```

public class RealImage implements Image{
    String filename;
    public String getFilename() {
        return filename;
    }
    public void setFilename(String filename) {
        this.filename = filename;
    }
    public RealImage(String filename) {
        this.filename = filename;
        loadFromDisk(filename);
    }
    @Override
    public ImageIcon display() {
        ImageIcon img = new ImageIcon(filename);
        return img;
    }
    private void loadFromDisk(String filename){
        JOptionPane.showMessageDialog(null, "Loading image..." + filename);
    }
}
    
```

- Tạo lớp ProxyImage là một lớp Proxy giúp giảm bộ nhớ đối tượng RealImage khi đang tải.

```

public class ProxyImage implements Image {
    private RealImage realImage;
}
    
```

```
private String fileName;

public ProxyImage(String fileName) {
    this.fileName = fileName;
}

@Override
public ImageIcon display() {
    if (realImage == null) {
        realImage = new RealImage(fileName);
    }
    return realImage.display();
}

public String getFileName() {
    return fileName;
}

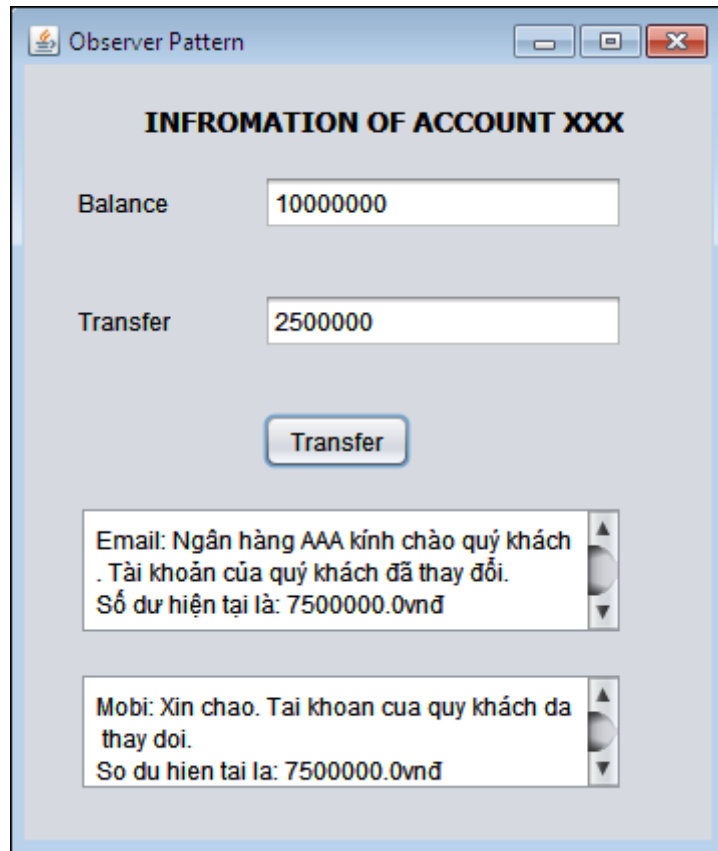
public void setFileName(String fileName) {
    this.fileName = fileName;
}
}
```

- Tạo JFrame frmProxyPatternDemo, là lớp demo sẽ sử dụng ProxyImage để có được đối tượng Image tải lên và hiển thị nó khi cần trong phương thức khởi tạo của form

```
public frmProxyPatternDemo() {
    initComponents();
    Image img = new ProxyImage("src/Bai7/2016.jpg");
    lblImage.setIcon(img.display());
}
```

### 7.3. Giao dịch qua tài khoản

- ✓ **Yêu cầu:** Xây dựng chương trình với mô tả như sau:



Giả sử một khách hàng sử dụng thẻ của ngân hàng ABC có đăng ký dịch vụ thông báo khi có sự thay đổi số tiền trong tài khoản thông qua email và điện thoại di động.

Chương trình hiển thị số tiền hiện có trong tài khoản của một khách hàng. Khi khách hàng chuyển tiền hoặc rút tiền với một số tiền (nhỏ hơn số tiền hiện có trong tài khoản) thì giá trị trong tài khoản sẽ thay đổi đồng thời khách hàng sẽ nhận được một **email** thông báo là: "Ngân hàng AAA kính chào quý khách. Tài khoản của quý khách đã thay đổi. Số dư hiện tại là: yyy vnd", và đồng thời **điện thoại di động** cũng nhận được một tin nhắn là "Mobi: Xin chào. Tai khoan cua quy khách đa thay doi. So du hien tai la yyy vnd"

✓ **Hướng dẫn sử dụng:**

- Mặc định tài khoản có 10.000.000 vnd
- Người dùng nhập vào Số tiền cần rút (nhỏ hơn số tiền hiện có trong tài khoản)
- Nhấn "Rút tiền" => chương trình hiển thị thông tin sau khi vừa giao dịch xong ở cả email và điện thoại di động.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmObserverPatternDemo: JFrame
- \* (Các thể hiện phía dưới đều nằm trong Frame)
  - txtBalance: JTextField (kết xuất)
  - txtTransfer: JTextField (nhập liệu)
  - txtaEmail: JTextArea (kết xuất)
  - txtaMobiPhone: JTextArea (kết xuất)
  - btnTransfer: JButton (Xử lý giao dịch)

▪ **Nhập:**

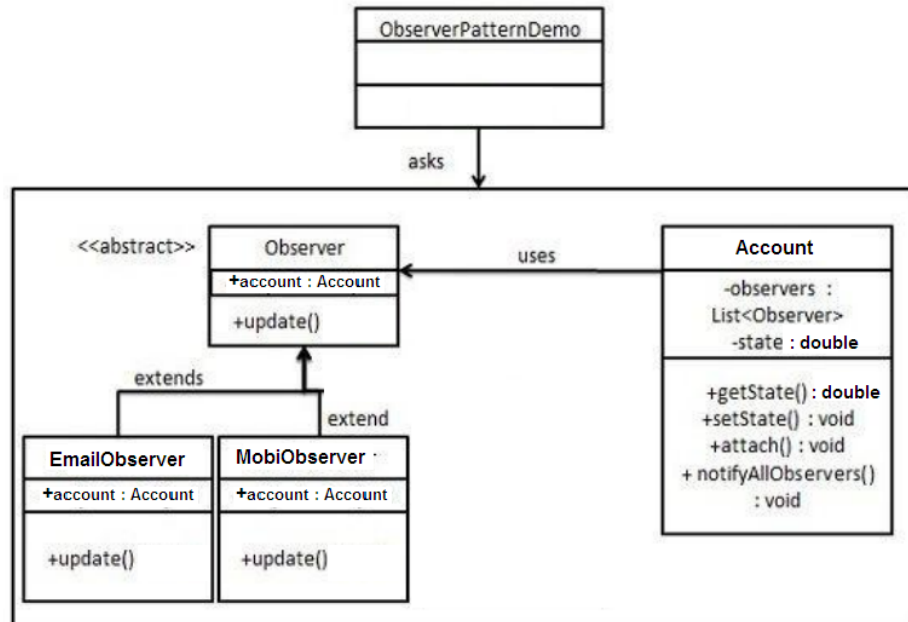
- Số tiền cần rút

▪ **Xuất:**

- Thông báo sau khi rút tiền qua email và điện thoại di động

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Observer Pattern



- Observer pattern sử dụng 3 lớp đối tượng actor là: Account, Observer và Client. Account là một đối tượng có các phương thức là để đính kèm và tách các observer với đối tượng client.
- Tạo ra lớp Account

```

public class Account {
    private List<Observer> observers = new ArrayList<>();
    private double state;
    public double getState() {
        return state;
    }
    public void setState(double state) {
        this.state = state;
        notifyAllObservers();
    }
    public void attach(Observer observer) {
        observers.add(observer);
    }
    public void notifyAllObservers() {
        for (Observer observer : observers) {
            observer.update();
        }
    }
}

```



- Tạo lớp abstract Observer

```
public abstract class Observer {
    protected Account account;
    public abstract String update();
}
```

- Tạo lớp MobiObserver là lớp cụ thể mở rộng từ Observer abstract.

```
public class MobiObserver extends Observer {
    public MobiObserver(Account account) {
        this.account = account;
        this.account.attach(this);
    }
    @Override
    public String update() {
        return "Mobi: Xin chào. Tài khoản của quý khách đã thay đổi. \nSố dư hiện tại là: " +
            account.getState() + "vnd";
    }
}
```

- Tạo lớp EmailObserver là lớp cụ thể mở rộng từ Observer abstract.

```
public class EmailObserver extends Observer {
    public EmailObserver(Account account) {
        this.account = account;
        this.account.attach(this);
    }
    @Override
    public String update() {
        return "Email: Ngân hàng AAA kính chào quý khách. Tài khoản của quý khách đã thay đổi.\nSố
            dư hiện tại là: " + account.getState() + "vnd";
    }
}
```

- Tạo form frmObserverPatternDemo, là lớp demo sẽ sử dụng Account và các lớp đối tượng cụ thể để hiển thị observer pattern khi đổi trạng thái.

- Hiện thực phương thức btnTransferActionPerformed() để thực hiện công việc trên

```
private void btnTransferActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        double balance = Double.parseDouble(txtBalance.getText());
        double transfer = Double.parseDouble(txtTransfer.getText());
        if (transfer > balance) {
            JOptionPane.showMessageDialog(rootPane, "Transfer phải <= Banlance");
        }
    }
}
```

```

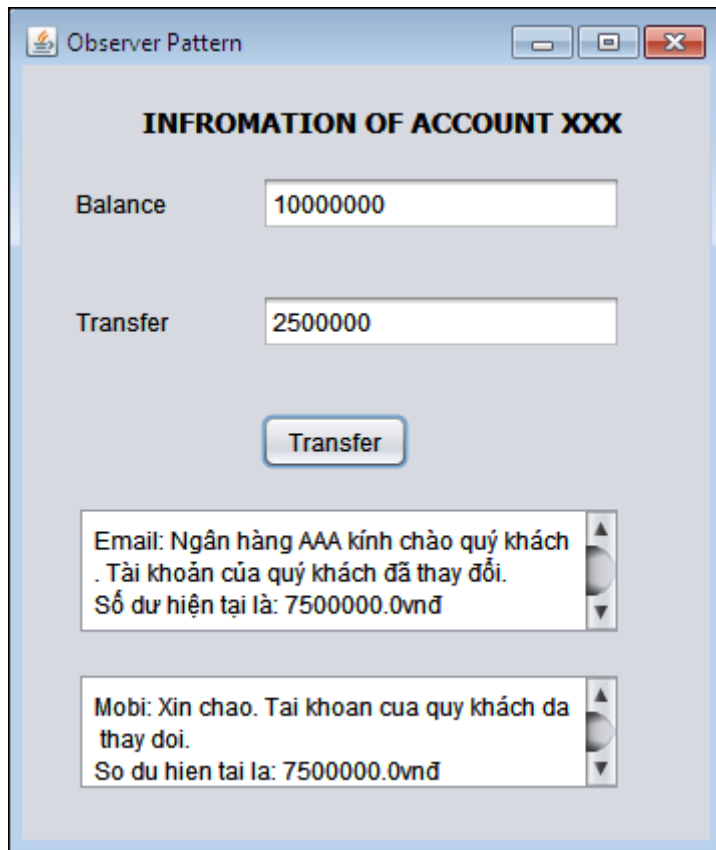
    } else {
        double money = balance - transfer;
        Account account = new Account();
        account.setState(money);

        txtaEmail.setText(new EmailObserver(account).update());
        txtaMobi.setText(new MobiObserver(account).update());
    }
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(rootPane, "Lỗi nhập liệu: " + ex.getMessage());
}
}

```

#### 7.4. Giao dịch qua tài khoản

- ✓ **Yêu cầu:** Xây dựng chương trình với mô tả như sau:



Giả sử một khách hàng sử dụng thẻ của ngân hàng ABC có đăng ký dịch vụ thông báo khi có sự thay đổi số tiền trong tài khoản thông qua email và điện thoại di động.

Chương trình hiển thị số tiền hiện có trong tài khoản của một khách hàng. Khi khách hàng chuyển tiền hoặc rút tiền với một số tiền (nhỏ hơn số tiền hiện có trong tài khoản) thì giá trị trong tài khoản sẽ thay đổi đồng thời khách hàng sẽ nhận được một **email** thông báo là: "Ngân hàng AAA kính chào quý khách. Tài khoản của quý khách đã thay đổi. Số dư hiện tại là: yyy vnd", và đồng thời **điện thoại di động** cũng nhận được một tin nhắn là "Mobi: Xin chào. Tài khoản của quý khách đã thay đổi. Số dư hiện tại là yyy vnd"

- ✓ **Hướng dẫn sử dụng:**

- Mặc định tài khoản có 10.000.000 vnd

- Người dùng nhập vào Số tiền cần rút (nhỏ hơn số tiền hiện có trong tài khoản)
- Nhấn “Rút tiền” => chương trình hiển thị thông tin sau khi vừa giao dịch xong ở cả email và điện thoại di động.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmObserverPatternDemo: JFrame
- \* (Các thể hiện phía dưới đều nằm trong Frame)
  - txtBalance: JTextField (kết xuất)
  - txtTransfer: JTextField (nhập liệu)
  - txtaEmail: JTextArea (kết xuất)
  - txtaMobiPhone: JTextArea (kết xuất)
  - btnTransfer: JButton (Xử lý giao dịch)

▪ **Nhập:**

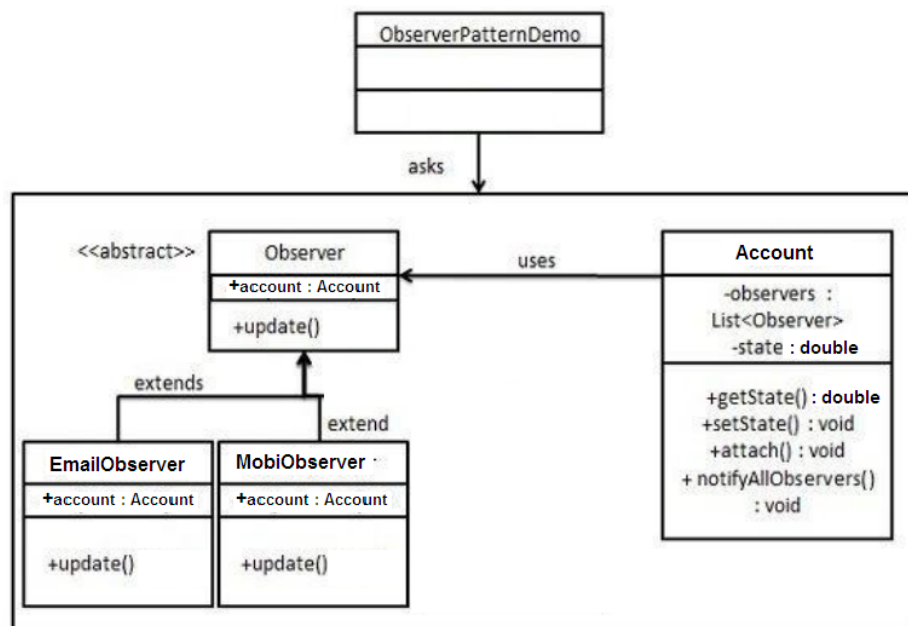
- Số tiền cần rút

▪ **Xuất:**

- Thông báo sau khi rút tiền qua email và điện thoại di động

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Observer Pattern



- Observer pattern sử dụng 3 lớp đối tượng actor là: Account, Observer và Client. Account là một đối tượng có các phương thức là để đính kèm và tách các observer với đối tượng client.
- Tạo ra lớp Account

```

public class Account {
    private List<Observer> observers = new ArrayList<>();
    private double state;
    public double getState() {
    }
}
    
```

```

        return state;
    }
    public void setState(double state) {
        this.state = state;
        notifyAllObservers();
    }
    public void attach(Observer observer) {
        observers.add(observer);
    }
    public void notifyAllObservers() {
        for (Observer observer : observers) {
            observer.update();
        }
    }
}

```

- Tạo lớp abstract Observer

```

public abstract class Observer {
    protected Account account;
    public abstract String update();
}

```

- Tạo lớp MobiObserver là lớp cụ thể mở rộng từ Observer abstract.

```

public class MobiObserver extends Observer {
    public MobiObserver(Account account) {
        this.account = account;
        this.account.attach(this);
    }
    @Override
    public String update() {
        return "Mobi: Xin chào. Tài khoản của quý khách đã thay đổi. \nSố dư hiện tại là: " +
            account.getState() + "vnd";
    }
}

```

- Tạo lớp EmailObserver là lớp cụ thể mở rộng từ Observer abstract.

```

public class EmailObserver extends Observer {
    public EmailObserver(Account account) {
        this.account = account;
        this.account.attach(this);
    }
}

```

```
@Override
public String update() {
    return "Email: Ngân hàng AAA kính chào quý khách. Tài khoản của quý khách đã thay đổi.\nSố
    dư hiện tại là: " + account.getState() + "vnd";
}
}
```

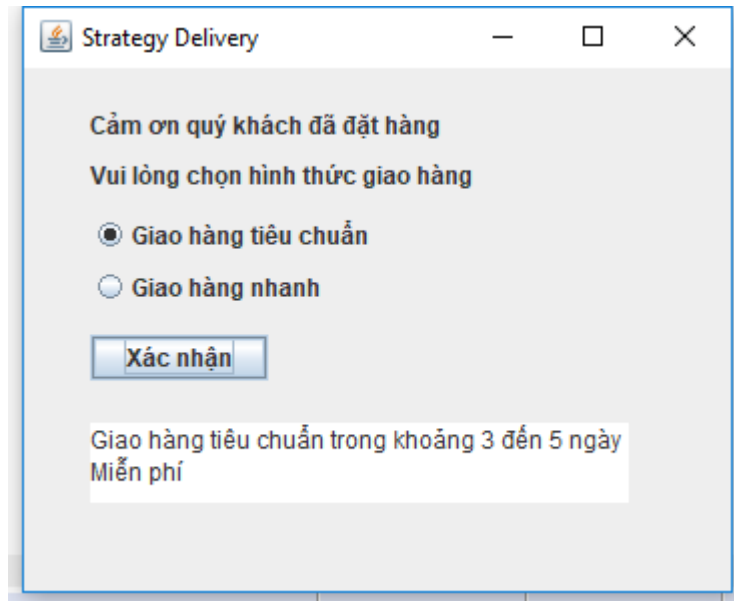
- Tạo form frmObserverPatternDemo, là lớp demo sẽ sử dụng Account và các lớp đối tượng cụ thể để hiển thị observer pattern khi đổi trạng thái.
  - Hiện thực phương thức btnTransferActionPerformed() để thực hiện công việc trên

```
private void btnTransferActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        double balance = Double.parseDouble(txtBalance.getText());
        double transfer = Double.parseDouble(txtTransfer.getText());
        if (transfer > balance) {
            JOptionPane.showMessageDialog(rootPane, "Transfer phải <= Banlance");
        } else {
            double money = balance - transfer;
            Account account = new Account();
            account.setState(money);

            txtaEmail.setText(new EmailObserver(account).update());
            txtaMobi.setText(new MobiObserver(account).update());
        }
    } catch (NumberFormatException ex) {
        JOptionPane.showMessageDialog(rootPane, "Lỗi nhập liệu: " + ex.getMessage());
    }
}
```

## 7.5. Hình thức giao hàng - Strategy

- ✓ **Yêu cầu: Xây dựng chương trình cho phép người dùng chọn lựa hình thức giao hàng như sau:**
  - Thực hiện việc giao hàng với 2 hình thức: Giao hàng tiêu chuẩn và Giao hàng nhanh. Việc giao hàng phụ thuộc vào lựa chọn của khách hàng khi đặt hàng.



Strategy Delivery

Cảm ơn quý khách đã đặt hàng

Vui lòng chọn hình thức giao hàng

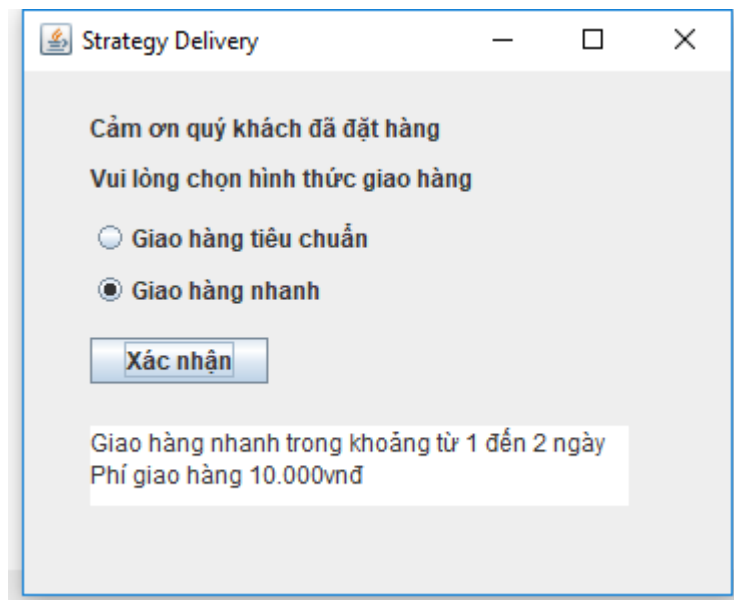
☒ Giao hàng tiêu chuẩn

☐ Giao hàng nhanh

Xác nhận

Giao hàng tiêu chuẩn trong khoảng 3 đến 5 ngày  
Miễn phí

Giao hàng tiêu chuẩn



Strategy Delivery

Cảm ơn quý khách đã đặt hàng

Vui lòng chọn hình thức giao hàng

☐ Giao hàng tiêu chuẩn

☒ Giao hàng nhanh

Xác nhận

Giao hàng nhanh trong khoảng từ 1 đến 2 ngày  
Phí giao hàng 10.000vnd

Giao hàng nhanh

✓ **Hướng dẫn sử dụng:**

- Người dùng chọn hình thức giao hàng => chương trình hiển thị thời gian giao hàng và phí giao hàng.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmStrategyDemo: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

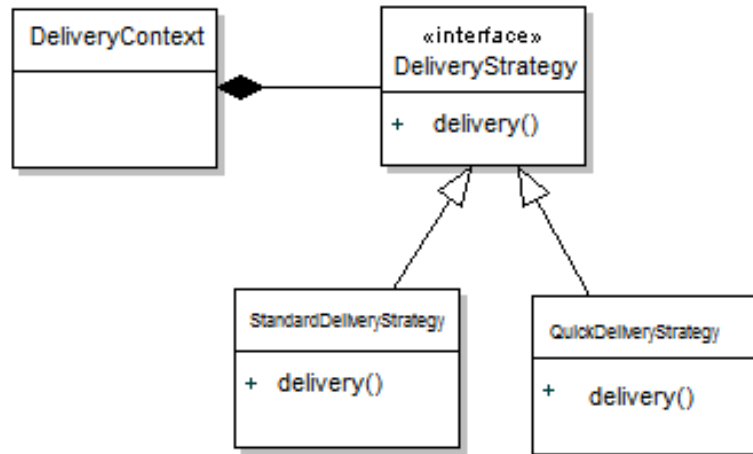
- rbtTieuChuan: RadioButton
- rbtNhanh: RadioButton
- rbtgChon: ButtonGroup
- txtaKetQua: TextArea (kết xuất - hiển thị kết quả)
- btnXacNhan: JButton (Xử lý hiển thị kết quả chọn hình thức giao hàng)

▪ **Nhập:**

- Loại
- **Xuất:**
  - Thông tin giao hàng.

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Strategy:



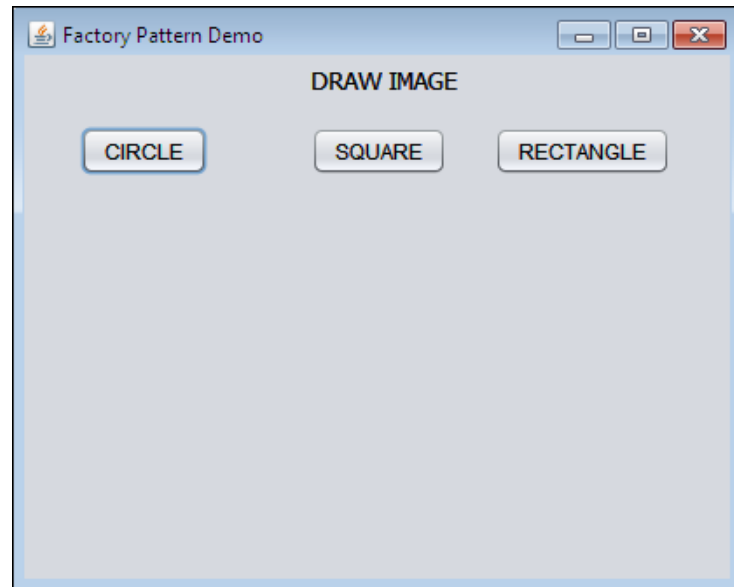
- Xây dựng các Interface và class như mô hình trên (tham khảo trên slide và hiệu chỉnh lại cho phù hợp)
- Gọi sử dụng các lớp đã xây dựng trong formStrategyDemo:

```

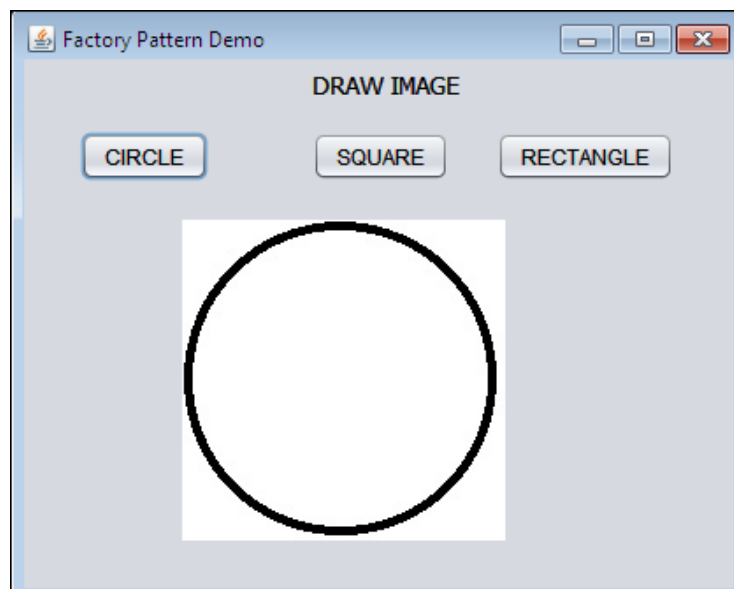
btnXacNhan.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent arg0) {
        if(rbtTieuChuan.isSelected()) {
            DeliveryContext context1 = new DeliveryContext(new
            StandardDeliveryStrategy());
            txtaKetQua.setText(context1.executeDelivery());
        }
        if(rbtNhanh.isSelected()) {
            DeliveryContext context2 = new DeliveryContext(new
            QuickDeliveryStrategy());
            txtaKetQua.setText(context2.executeDelivery());
        }
    }
});
    
```

## 7.6. Vẽ hình – Factory Design Pattern

- ✓ **Yêu cầu:** Xây dựng chương trình Vẽ hình như sau:



Form khởi tạo



Chọn nút "Circle"

Chương trình được xây dựng để vẽ hình. Người dùng chọn một hình bất kỳ thì sẽ hiển thị nội dung của hình được chọn (CIRCLE/ SQUARE/ RECTANGLE)

✓ **Hướng dẫn sử dụng:**

- Người dùng nhấn một trong 3 nút: "CIRCLE", "SQUARE", "RECTANGLE" thì hình được chọn sẽ hiển thị.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- frmFactoryPatternDemo: JFrame

\* (Các thể hiện phía dưới đều nằm trong Frame)

- btnCircle: JButton (Xử lý tạo hình tròn)
- btnSquare: JButton (Xử lý tạo hình vuông)
- btnRectangle: JButton (Xử lý tạo hình chữ nhật)
- lblImage: JLabel (Kết xuất)

▪ **Nhập:**



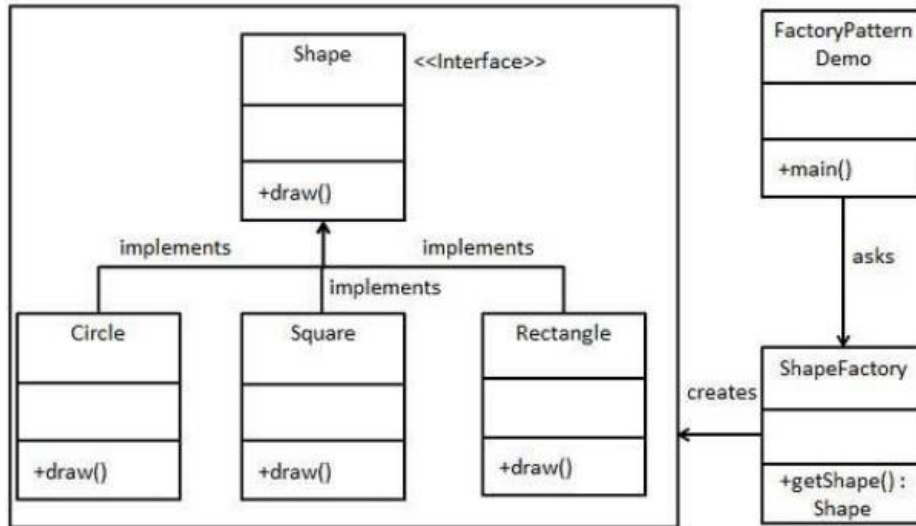
- Bấm 1 trong 3 nút

▪ **Xuất:**

- Hình vẽ tương ứng hiển thị

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Factory Pattern:



- Tạo lớp interface Shape

```

public interface Shape {
    ImageIcon draw();
}
    
```

- Tạo lớp Circle cụ thể implement từ interface Shape.

```

public class Circle implements Shape{
    @Override
    public ImageIcon draw() {
        ImageIcon img = new ImageIcon("src/Bai7/factorypattern/circle.jpg");
        return img;
    }
}
    
```

- Tạo lớp Square cụ thể implement từ interface Shape.

```

public class Square implements Shape{
    @Override
    public ImageIcon draw() {
        ImageIcon img = new ImageIcon("src/Bai7/factorypattern/square.jpg");
        return img;
    }
}
    
```

- Tạo lớp Rectangle cụ thể implement từ interface Shape.

```
public class Rectangle implements Shape{
    @Override
    public ImageIcon draw() {
        ImageIcon img = new ImageIcon("src/Bai7/factorypattern/rectangle.jpg");
        return img;
    }
}
```

- Tạo 1 lớp ShapeFactory để xử lý việc chọn hình

```
public class ShapeFactory {
    public Shape getShape(String shapeType){
        if(shapeType==null){
            return null;
        }
        if(shapeType.equalsIgnoreCase("CIRCLE")){
            return new Circle();
        }
        else if(shapeType.equalsIgnoreCase("SQUARE")){
            return new Square();
        }
        else if(shapeType.equalsIgnoreCase("RECTANGLE")){
            return new Rectangle();
        }
        return null;
    }
}
```

- Tạo JFrame form FactoryPatternDemo, là lớp demo sẽ sử dụng ShapeFactory để lấy một đối tượng Shape, nó sẽ chuyển các thông tin (CIRCLE / RECTANGLE / SQUARE) vào ShapeFactory để có được loại của đối tượng mà nó cần
  - Hiện thực phương thức btnCircleActionPerformed() để hiển thị hình ảnh

```
private void btnCircleActionPerformed(java.awt.event.ActionEvent evt) {
    ShapeFactory shapeFactory = new ShapeFactory();
    //get an object of Circle and call its draw method.
    Shape shape1 = shapeFactory.getShape("CIRCLE");
    //call draw method of Circle
    lblImage.setIcon(shape1.draw());
}
```

- Hiện thực phương thức btnSquareActionPerformed() để hiển thị hình ảnh

```
private void btnSquareActionPerformed(java.awt.event.ActionEvent evt) {
```

```
ShapeFactory shapeFactory = new ShapeFactory();
Shape shape1 = shapeFactory.getShape("SQUARE");
lblImage.setIcon(shape1.draw());
}
```

- Hiện thực phương thức btnRectangleActionPerformed() để hiển thị hình ảnh

```
private void btnRectangleActionPerformed(java.awt.event.ActionEvent evt) {
    ShapeFactory shapeFactory = new ShapeFactory();
    Shape shape1 = shapeFactory.getShape("RECTANGLE");
    lblImage.setIcon(shape1.draw());
}
```

## 7.7. Tạo xe hơi – Builder Design Pattern

- ✓ **Yêu cầu:** Xây dựng chương trình cho phép người dùng tạo ra chiếc xe theo ý mình bằng cách: chọn xe mặc định 4 bánh, màu đen hoặc tạo xe với số bánh và màu sắc, hoặc tạo xe chỉ chọn số bánh màu mặc định, hoặc chỉ chọn màu sắc số bánh mặc định:

Bạn muốn tạo xe như thế nào?

1. Tiêu chuẩn 4 bánh, màu đen
2. Tùy chọn số bánh và màu
3. Tùy chọn số bánh, màu đen
4. 4 bánh, tùy chọn màu

1

Car [wheels = 4, color = Black]

Bạn có tiếp tục tạo xe? 1: Có, Khác 1: Không

1

Bạn muốn tạo xe như thế nào?

1. Tiêu chuẩn 4 bánh, màu đen
2. Tùy chọn số bánh và màu
3. Tùy chọn số bánh, màu đen
4. 4 bánh, tùy chọn màu

2

Nhập số bánh:

8

Nhập màu:

Yellow

Car [wheels = 8, color = Yellow]

Bạn có tiếp tục tạo xe? 1: Có, Khác 1: Không

1

Bạn muốn tạo xe như thế nào?

1. Tiêu chuẩn 4 bánh, màu đen
2. Tùy chọn số bánh và màu
3. Tùy chọn số bánh, màu đen
4. 4 bánh, tùy chọn màu

3

Nhập số bánh:

8

Car [wheels = 8, color = Black]

Bạn có tiếp tục tạo xe? 1: Có, Khác 1: Không

1

Bạn muốn tạo xe như thế nào?

1. Tiêu chuẩn 4 bánh, màu đen
2. Tùy chọn số bánh và màu
3. Tùy chọn số bánh, màu đen
4. 4 bánh, tùy chọn màu

4

Nhập màu:

Red

Car [wheels = 4, color = Red]

Bạn có tiếp tục tạo xe? 1: Có, Khác 1: Không

0

<

✓ **Hướng dẫn sử dụng:**

- Người dùng chọn các loại 1, 2, 3, 4 để tạo ra chiếc xe tương ứng.

✓ **Tóm tắt yêu cầu**

▪ **Nhập:**

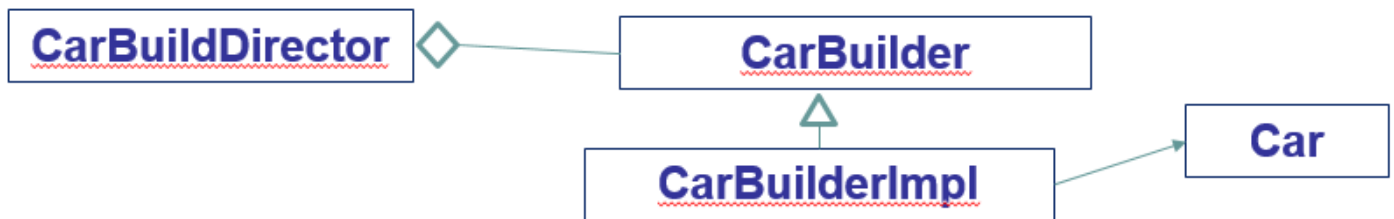
- Số bánh, màu sắc

▪ **Xuất:**

- Thông tin chiếc xe tương ứng

✓ **Hướng dẫn**

- Mô hình gợi ý dùng Builder Pattern:



- Xây dựng các Interface và class như mô hình trên (tham khảo trên slide và hiệu chỉnh lại cho phù hợp)
- Gọi sử dụng các lớp đã xây dựng trong main:

```

public class CarBuildDirector {
    private CarBuilder builder;

    public CarBuildDirector(final CarBuilder builder) {
        this.builder = builder;
    }

    public Car construct() {
        return builder.setWheels(4).setColor("Black").build();
    }

    public Car construct1(int numWheels, String color) {
        return builder.setWheels(numWheels).setColor(color).build();
    }

    public Car construct2(int numWheels) {
        return builder.setWheels(numWheels).setColor("Black").build();
    }

    public Car construct3(String color) {
        return builder.setWheels(4).setColor(color).build();
    }
}
    
```

```

    public static void main(final String[] arguments) throws NumberFormatException,
    IOException {
        BufferedReader input = new BufferedReader(new InputStreamReader(System.in));
        final CarBuilder builder = new CarBuilderImpl();
        final CarBuildDirector carBuildDirector = new CarBuildDirector(builder);
        int i = 1;
        while (i == 1) {
            System.out.println("Bạn muốn tạo xe như thế nào?\n1. Tiêu chuẩn 4 bánh,
            màu đen\n"
                               + "2. Tùy chọn số bánh và màu\n" + "3. Tùy chọn số bánh,
            màu đen\n" + "4. 4 bánh, tùy chọn màu");
            int select = Integer.parseInt(input.readLine());
            if (select == 1) {
                System.out.println(carBuildDirector.construct());
            } else if (select == 2) {
                System.out.println("Nhập số bánh:");
                int wheels = Integer.parseInt(input.readLine());
                System.out.println("Nhập màu:");
                String color = input.readLine();
                System.out.println(carBuildDirector.construct1(wheels, color));
            } else if (select == 3) {
                System.out.println("Nhập số bánh:");
                int wheels = Integer.parseInt(input.readLine());
                System.out.println(carBuildDirector.construct2(wheels));
            } else if (select == 4) {
                System.out.println("Nhập màu:");
                String color = input.readLine();
                System.out.println(carBuildDirector.construct3(color));
            }
            else {
                System.out.println("Chỉ có 4 loại để lựa chọn!");
            }
            System.out.println("\nBạn có tiếp tục tạo xe? 1: Có, Khác 1: Không");
            i = Integer.parseInt(input.readLine());
        }
    }
}

```

# BÀI 8: Thread



## Mục tiêu chính:

- Một số khái niệm về chương trình, tiến trình, chương trình đơn luồng (single thread) và chương trình đa luồng (multi thread)

## 8.1. Ứng dụng đa luồng

- ✓ **Yêu cầu:** Viết chương trình tạo và thực thi nhiều luồng độc lập trên console như sau:

```
Output - JavaSE8_Programmer2 (run)
run:
Creating Google
Starting Google
Creating Yahoo
Starting Yahoo
Running Google
Thread: Google, 3
Creating Facebook
Running Yahoo
Starting Facebook
Thread: Yahoo, 3
Running Facebook
Thread: Facebook, 3
Thread: Yahoo, 2
Thread: Facebook, 2
Thread: Google, 2
Thread: Facebook, 1
Thread: Google, 1
Thread: Yahoo, 1
Thread Google exiting.
Thread Yahoo exiting.
Thread Facebook exiting.
BUILD SUCCESSFUL (total time: 0 seconds)
```

*Kết quả khi thực thi 3 luồng*

- ✓ **Tóm tắt yêu cầu**

- **Nhập:**

- Không có

- **Xuất:**

- Kết quả chạy 3 luồng

- ✓ **Hướng dẫn**

- Tạo một lớp ThreadDemo kế thừa từ lớp Thread:

```
public class ThreadDemo extends Thread {
    private Thread t;
    private String threadName;

    ThreadDemo( String name){
        threadName = name;
        System.out.println("Creating " + threadName );
    }
}
```



```

    }

    @Override
    public void run() {
        System.out.println("Running " + threadName );
        try {
            for(int i = 3; i > 0; i--) {
                System.out.println("Thread: " + threadName + ", " + i);
                // Let the thread sleep for a while.
                Thread.sleep(50);
            }
        } catch (InterruptedException e) {
            System.out.println("Thread " + threadName + " interrupted.");
        }
        System.out.println("Thread " + threadName + " exiting.");
    }

    @Override
    public void start ()
    {
        System.out.println("Starting " + threadName );
        if (t == null)
        {
            t = new Thread (this, threadName);
            t.start ();
        }
    }
}

```

- Tạo và thực thi 3 luồng độc lập:

```

public class ThreadTest {
    public static void main(String[] args) {
        ThreadDemo T1 = new ThreadDemo("Google");
        T1.start();

        ThreadDemo T2 = new ThreadDemo("Yahoo");
        T2.start();

        ThreadDemo T3 = new ThreadDemo("Facebook");
    }
}

```



```
T3.start();

    }

}
```

## 8.2. Tính tổng các phần tử trong mảng

### ✓ Yêu cầu: Sử dụng Thread để tính tổng các phần tử trong mảng

- Chương trình được xây dựng để tính tổng cho một mảng có n phần tử (n lớn) có giá trị ngẫu nhiên từ 0 - 10
- Yêu cầu tạo ra Thread để tính tổng một mảng con (1 phần trong mảng lớn)
- Hãy tạo ra một số Thread để thực hiện việc tính tổng này

### ✓ Hướng dẫn sử dụng:

- Nhập vào số phần tử cho mảng => Tạo mảng ngẫu nhiên với n phần tử.
- Nhập vào số Thread cần tạo => Tổng mảng dựa trên số Thread này.

### ✓ Tóm tắt yêu cầu

#### ▪ Nhập:

- Số phần tử
- Số Thread

#### ▪ Xuất:

- Tổng của mảng

### ✓ Hướng dẫn

- Tạo SumThread kế thừa từ Thread để tính tổng mảng con:

```
/**
 *
 * @author ktpuong
 * @version 1.0
 * @since 09-2016
 * This thread finds the sum of a subsection of an array.
 */

public class SumThread extends Thread {
    private int lo, hi;
    private int[] arr;
    private int sum = 0;

    public SumThread(int lo, int hi, int[] arr) {
        this.lo = lo;
        this.hi = hi;
        this.arr = arr;
    }
}
```

```

    }

    @Override
    public void run() {
        for (int i = this.lo; i < this.hi; i++) {
            this.sum += this.arr[i];
        }
    }

    public int getLo() {
        return lo;
    }

    public void setLo(int lo) {
        this.lo = lo;
    }

    public int getHi() {
        return hi;
    }

    public void setHi(int hi) {
        this.hi = hi;
    }

    public int[] getArr() {
        return arr;
    }

    public void setArr(int[] arr) {
        this.arr = arr;
    }

    public int getSum() {
        return sum;
    }

    public void setSum(int sum) {

```

```

        this.sum = sum;
    }
}

```

– Main

```

/**
 * @author ktphuong
 * @version 1.0
 * @since 09-2016
 * Using Thread to calculate Sum of Array
 */
public class SumOfArrayThread {

    /**
     * @param args the command line arguments
     * @throws java.lang.InterruptedException
     */
    public static void main(String[] args) throws InterruptedException {
        Scanner input = new Scanner(System.in);
        System.out.println("Input n:");
        int n = input.nextInt();
        int arr[] = new int[n];
        Random random = new Random();
        for (int i = 0; i < n; i++) {
            arr[i] = random.nextInt(10);
        }
        System.out.println("Input num of Threads:");
        int numThreads = input.nextInt();
        int sum = sum(arr, numThreads);
        String strArr = "";
        for (int value : arr) {
            strArr += value + " ";
        }
        System.out.println("Array: " + strArr);
        System.out.println("Sum: " + sum);
    }

    // Sum of arr

```

```

public static int sum(int[] arr, int numThreads) throws InterruptedException {
    int len = arr.length;
    int sum = 0;
    // Create and start numThreads.
    SumThread[] ts = new SumThread[numThreads];
    for (int i = 0; i < numThreads; i++) {
        ts[i] = new SumThread((i * len) / numThreads, ((i + 1) * len / numThreads), arr);
        ts[i].start();
    }
    // Wait for the threads to finish and sum their results.
    for (int i = 0; i < numThreads; i++) {
        ts[i].join();
        sum += ts[i].getSum();
    }
    return sum;
}
}

```

### 8.3. Tìm giá trị lớn nhất trong mảng

- ✓ **Yêu cầu: Sử dụng Thread để tìm giá trị lớn nhất trong mảng**
  - Chương trình được xây dựng để tìm giá trị lớn nhất trong một mảng có n phần tử (n lớn) có giá trị ngẫu nhiên từ 0 - 100
  - Yêu cầu tạo ra Thread để tìm max trong một mảng con (1 phần trong mảng lớn)
  - Hãy tạo ra một số Thread để thực hiện việc tìm max này
- ✓ **Hướng dẫn sử dụng:**
  - Nhập vào số phần tử cho mảng => Tạo mảng ngẫu nhiên với n phần tử.
  - Nhập vào số Thread cần tạo => Giá trị lớn nhất của mảng dựa trên số Thread này
- ✓ **Tóm tắt yêu cầu**
  - **Nhập:**
    - Số phần tử
    - Số Thread
  - **Xuất:**
    - Giá trị lớn nhất
- ✓ **Hướng dẫn**
  - Làm tương tự bài tính tổng

# BÀI 9: Tổng kết



*Mục tiêu chính:*

*Giúp cho học viên có thể xây dựng hoàn chỉnh ứng dụng windows forms với Java*

## 9.1. Xây dựng ứng dụng Quản lý cửa hàng Phương Perfume

### ✓ Mô tả ứng dụng

- Cửa hàng nước hoa Phương Perfume chuyên kinh doanh các mặt hàng nước hoa trực tiếp tại cửa hàng và online.
- Hiện tại cửa hàng có các bộ phận với các công việc chính như sau:
  - Nhân viên bán hàng: giới thiệu sản phẩm, chăm sóc khách hàng, bán hàng và giải quyết các đơn hàng của khách.
  - Nhân viên kế toán: quản lý đơn hàng và thống kê
- Cửa hàng nước hoa yêu cầu xây dựng website giúp cửa hàng quảng bá và bán sản phẩm, hỗ trợ khách hàng có thể xem và đặt hàng trực tiếp trên mạng và xây dựng một ứng dụng windows forms giúp cửa hàng quản lý thông tin sản phẩm, tạo quảng cáo, đơn hàng, thống kê, quản lý thông tin khách hàng...
- Ứng dụng được xây dựng có thể hỗ trợ nhân viên trong cửa hàng thực hiện tốt các công việc của mình và giúp cửa hàng có thể thu hút nhiều khách đến tham quan và mua hàng.

⇒ **Xây dựng windows forms quản lý cửa hàng nước hoa (làm việc ở module này)**

⇒ Xây dựng website giới thiệu, quảng bá sản phẩm và bán hàng online (làm việc ở module sau)

✓ **Thiết kế CSDL cho ứng dụng (tập tin phuong\_perfume.sql được đính kèm): gồm có các bảng sau:**

- Bảng loai (loại)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
2	tenloai	varchar(200)	utf8_general_ci		No	None	

- Bảng thuonghieu (thương hiệu)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
2	tenthuonghieu	varchar(300)	utf8_general_ci		No	None	
3	hinhanh	varchar(300)	utf8_general_ci		No	None	

- Bảng sanpham (sản phẩm)

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	tensanpham	varchar(300)	utf8_general_ci		No	None	
<input type="checkbox"/> 3	mota	text	utf8_general_ci		Yes	NULL	
<input type="checkbox"/> 4	hinhanh	varchar(300)	utf8_general_ci		No	None	
<input type="checkbox"/> 5	dongia	double			No	None	
<input type="checkbox"/> 6	dongiaKM	double			Yes	NULL	
<input type="checkbox"/> 7	soluong	int(11)			No	None	
<input type="checkbox"/> 8	ngaytao	date			No	None	
<input type="checkbox"/> 9	hienthi	int(11)			No	1	
<input type="checkbox"/> 10	id_loai	int(11)			No	None	
<input type="checkbox"/> 11	id_thuonghieu	int(11)			No	None	

– Bảng vai tro (vai trò)

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	<u>tenvaitro</u>	varchar(100)	utf8_general_ci		No	None	
<input type="checkbox"/> 3	mota	text	utf8_general_ci		Yes	NULL	

– Bảng nguoidung (người dùng)

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	email	varchar(100)	utf8_general_ci		No	None	
<input type="checkbox"/> 3	password	varchar(100)	utf8_general_ci		No	None	
<input type="checkbox"/> 4	hoten	varchar(200)	utf8_general_ci		No	None	
<input type="checkbox"/> 5	diachi	varchar(300)	utf8_general_ci		No	None	
<input type="checkbox"/> 6	dtdd	varchar(20)	utf8_general_ci		No	None	
<input type="checkbox"/> 7	id_vaitro	int(11)			No	1	

– Bảng trangthaidonhang (trạng thái đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	tentrangthai	varchar(100)	utf8_general_ci		No	None	

– Bảng donhang (đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Extra
<input type="checkbox"/> 1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
<input type="checkbox"/> 2	id_khachhang	int(11)			No	None	
<input type="checkbox"/> 3	ngaydathang	date			No	None	
<input type="checkbox"/> 4	tennguoinhanhang	varchar(200)	utf8_general_ci		No	None	
<input type="checkbox"/> 5	dienthoainguoinhan	varchar(20)	utf8_general_ci		No	None	
<input type="checkbox"/> 6	diachigiaohang	varchar(300)	utf8_general_ci		No	None	
<input type="checkbox"/> 7	ghichu	text	utf8_general_ci		Yes	NULL	
<input type="checkbox"/> 8	thanhtoan	tinyint(4)			No	0	
<input type="checkbox"/> 9	id_trangthai	int(11)			No	1	

– Bảng chitietdonhang (chi tiết đơn hàng)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
2	id_donhang	int(11)			No	None	
3	id_sanpham	int(11)			No	None	
4	soluong	int(11)			No	None	

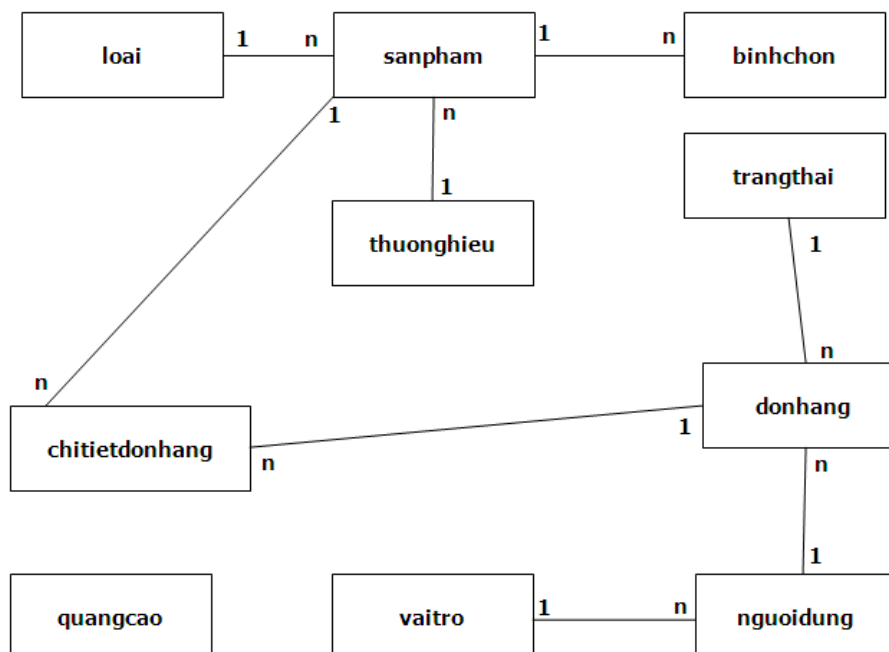
– Bảng quangcao (quảng cáo)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
2	hinhanh	varchar(200)	utf8_general_ci		No	None	
3	thongdiep	varchar(300)	utf8_general_ci		No	None	
4	thongtinchitiet	text	utf8_general_ci		Yes	NULL	
5	ngaydang	date			Yes	NULL	

– Bảng binhchon (bình chọn)

#	Name	Type	Collation	Attributes	Null	Default	Extra
1	<u>id</u>	int(11)			No	None	AUTO_INCREMENT
2	ten	varchar(200)	utf8_general_ci		No	None	
3	email	varchar(200)	utf8_general_ci		No	None	
4	diem	int(11)			No	None	
5	id_sanpham	int(11)			No	None	

Mối quan hệ giữa các bảng:

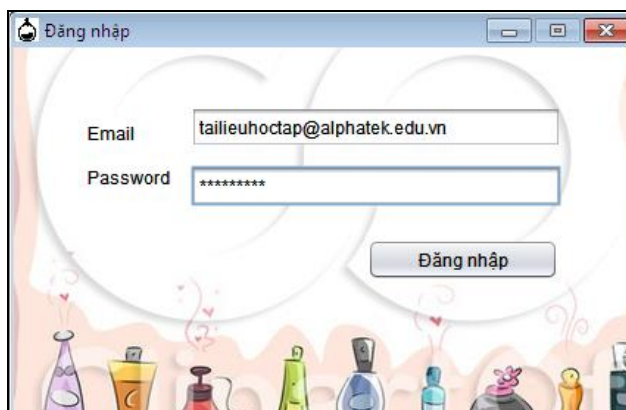


Lưu ý:

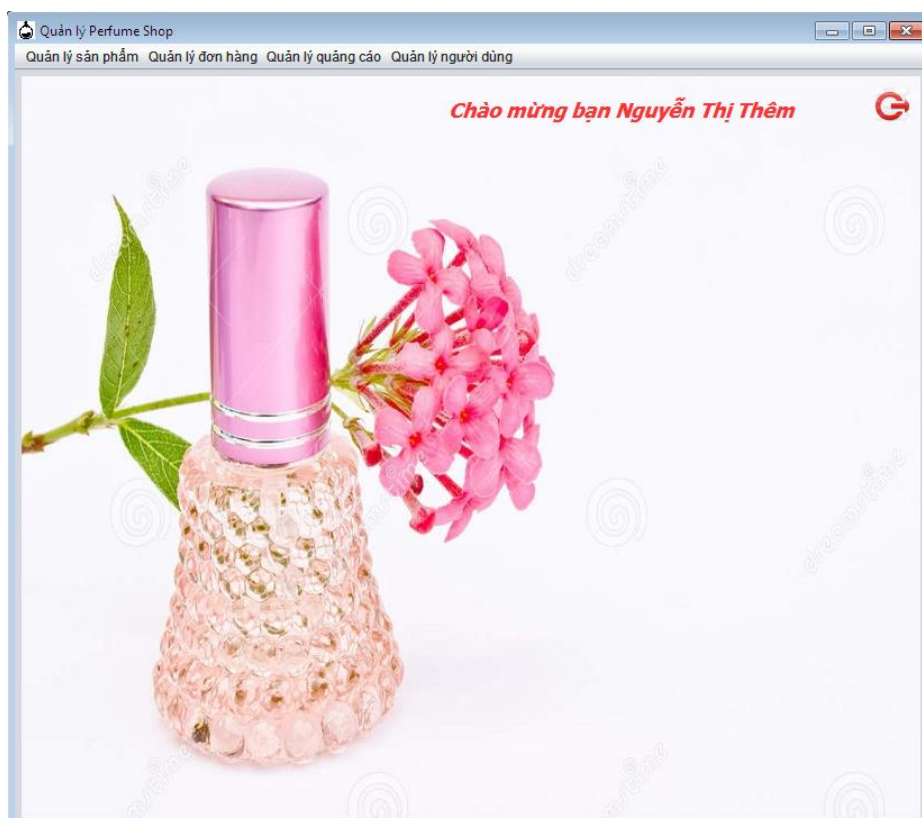
- Học viên được cung cấp CSDL **phuong\_perfume.sql** => Import CSDL phuong\_perfume.sql vào MySQL
- Thư mục **img** (chứa các hình ảnh cần thiết cho ứng dụng) => Học viên tạo Project và đưa vào project để sử dụng
- Trong project, ở thư mục src, học viên tạo package model, trong package model, tạo tất cả các lớp cần thiết tương ứng với các bảng của CSDL.

- Trong model, xây dựng lớp QLCSDL, trong lớp này có phương thức để kết nối đến CSDL phuong\_perfume

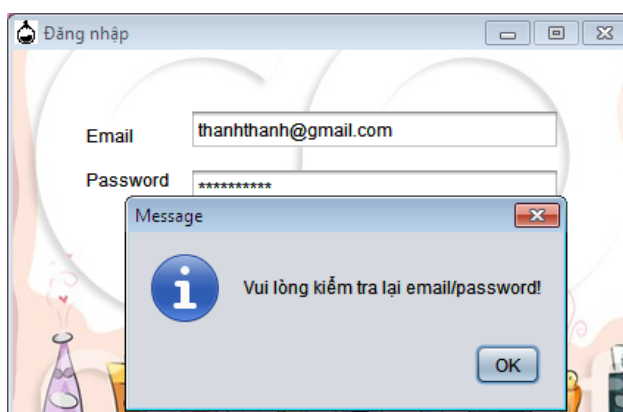
### 1. Yêu cầu 1: Đăng nhập vào hệ thống



*Form Đăng nhập*



*Đăng nhập thành công => Vào Form chính của ứng dụng*



*Đăng nhập không thành công => Hiện thị thông báo*



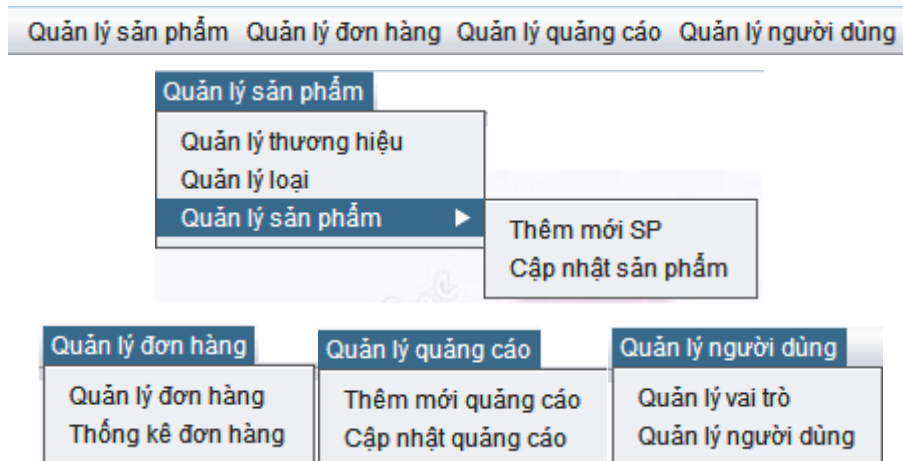
✓ **Hướng dẫn sử dụng:**

- Nhập email, password => nhấn "Đăng nhập" => hệ thống sẽ kiểm tra thông tin => nếu có tài khoản này và vai trò không phải là khách hàng thì đăng nhập thành công và chuyển vào màn hình làm việc chính của ứng dụng. Ngược lại, đăng nhập thất bại, hiển thị thông báo yêu cầu người dùng kiểm tra lại thông tin.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- Học viên thiết kế giao diện form đăng nhập phía trên
- Học viên thiết kế giao diện form chính như trên, trên form chính có menu chứa các công việc chính của ứng dụng



▪ **Nhập:**

- Email
- Password

▪ **Xuất:**

- Kết quả đăng nhập (thành công/ không thành công)

✓ **Hướng dẫn**

- Trong lớp QLCSDL, xây dựng phương thức phù hợp cho việc đăng nhập.
- Gợi ý:

```
public NgườiDung dangnhapNgườiDung(String email, String password) throws SQLException,
ClassNotFoundException {
    NgườiDung nd = null;
    try (Connection conn = this.connect()) {
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM nguoidung WHERE email like '" + email + "' and password like '"
+ password + "'";
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            nd = new NgườiDung();
            nd.setId(resultSet.getInt("id"));
            nd.setEmail(resultSet.getString("email"));
        }
    }
}
```

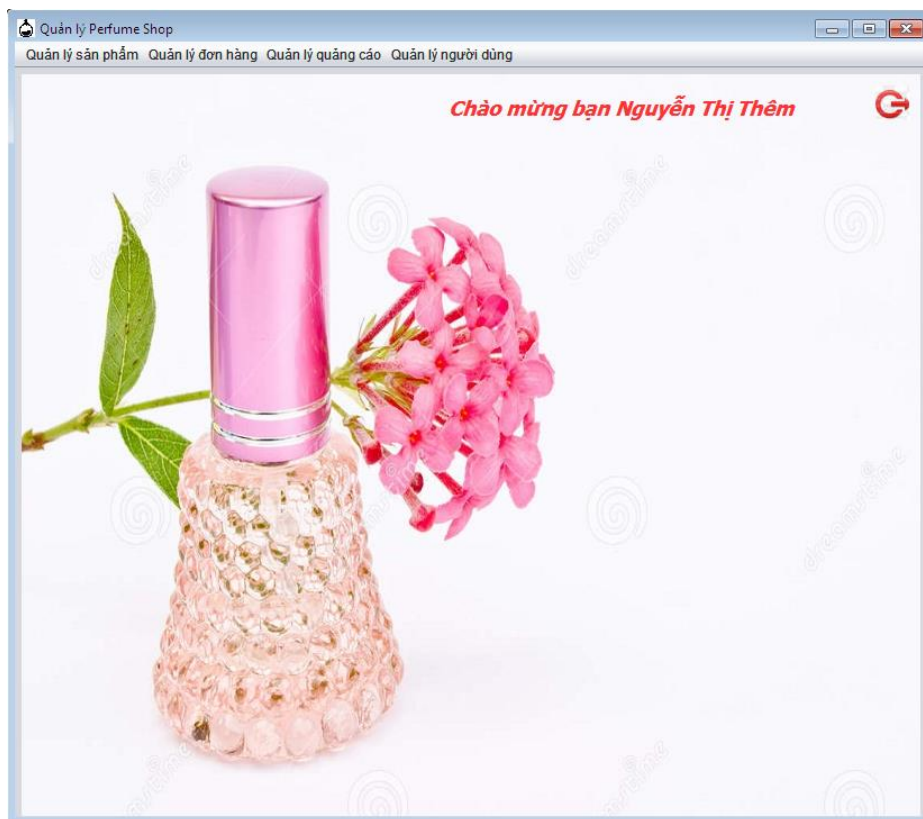
```

        nd.setPassword(resultSet.getString("password"));
        nd.setHoten(resultSet.getString("hoten"));
        nd.setDiachi(resultSet.getString("diachi"));
        nd.setDtdd(resultSet.getString("dtdd"));
        nd.setId_vaitro(resultSet.getInt("id_vaitro"));
    }
}
return nd;
}


```

- Xử lý việc đăng nhập trên form

## 2. Yêu cầu 2: Đăng xuất khỏi hệ thống



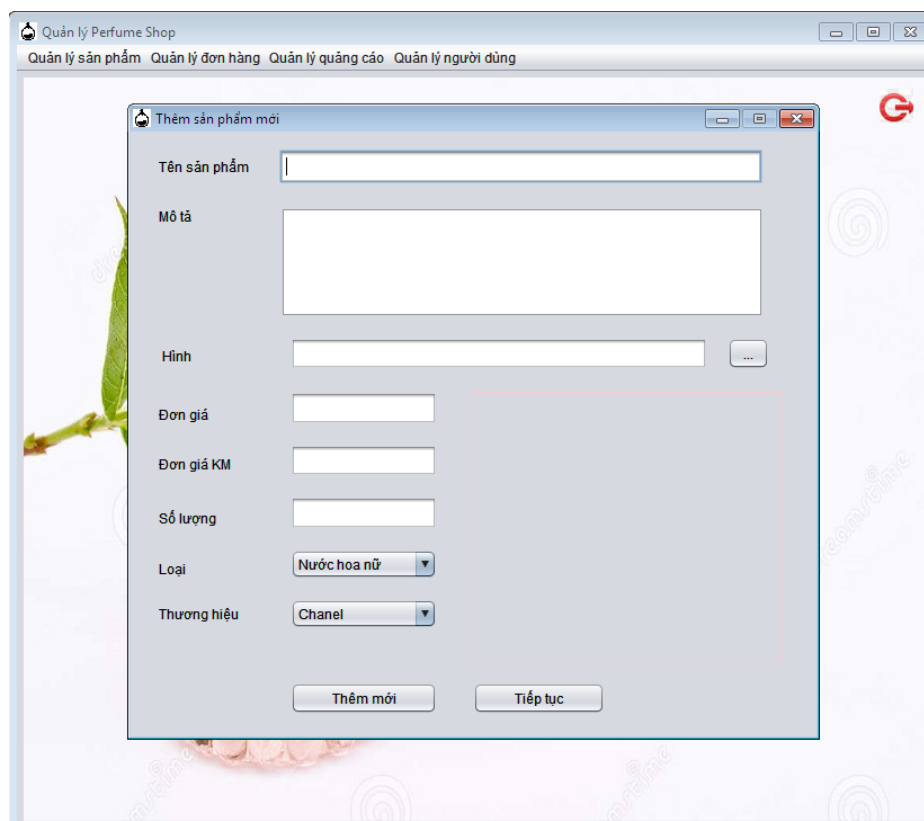
### ✓ Hướng dẫn sử dụng:

- Nhấn vào icon  đăng xuất ngay sau câu chào => Hệ thống sẽ đóng form chính quay trở về form Đăng nhập

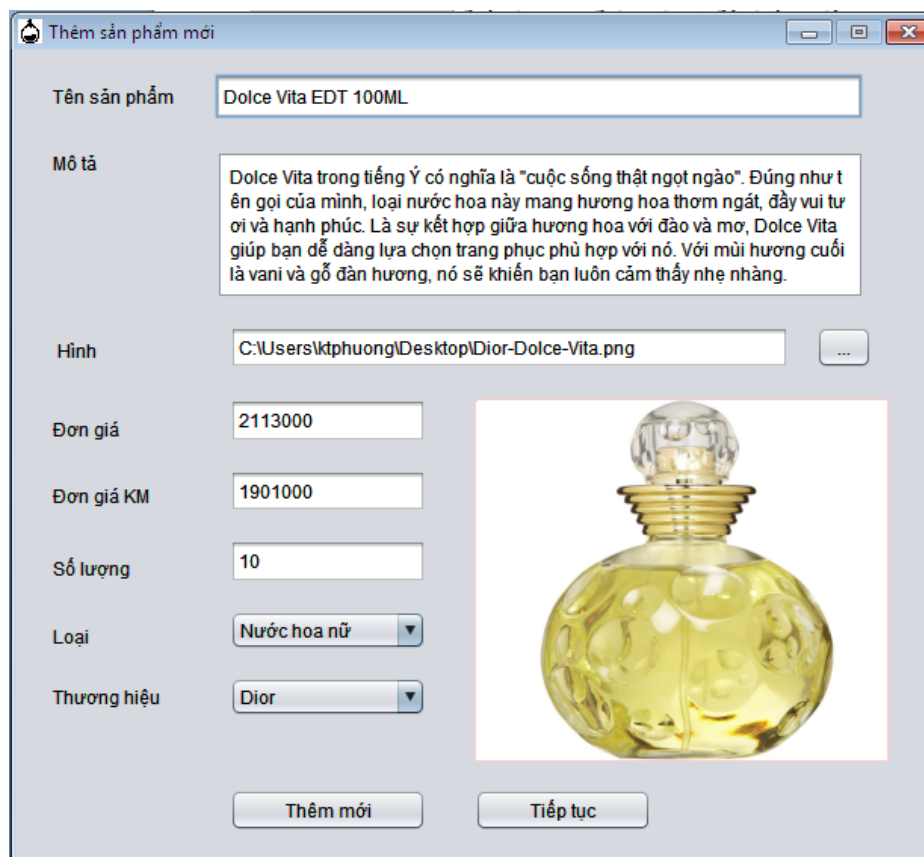
### ✓ Hướng dẫn

- Xử lý đóng form chính và mở form đăng nhập

## 3. Yêu cầu 3: Thêm sản phẩm mới



From "Thêm sản phẩm mới" sẽ được mở ra khi người dùng chọn chức năng này trên menu



Thêm sản phẩm mới

✓ **Hướng dẫn sử dụng:**

- Nhập các thông tin: tên sản phẩm, mô tả, hình, đơn giá, đơn giá khuyến mãi, số lượng, chọn loại và thương hiệu => nhấn "Thêm mới" để thêm sản phẩm.

- Nhấn “Tiếp tục” để xóa các thông tin đang có cho người dùng tiếp tục thêm mới sản phẩm

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- Học viên thiết kế giao diện Thêm sản phẩm mới như hình phía trên, trên đó có 2 combobox có sẵn loại và thương hiệu cho người dùng chọn.
- Form này là form nằm trong form chính, được mở ra khi người dùng chọn chức năng “Thêm sản phẩm mới” trên menu.

▪ **Nhập:**

- Tên sản phẩm, mô tả, hình, đơn giá, đơn giá khuyến mãi, số lượng, chọn loại và thương hiệu

▪ **Xuất:**

- Kết quả thêm thành công/ không thành công

✓ **Hướng dẫn**

- Trong lớp QLCSDL, xây dựng thêm phương thức phù hợp cho việc hiển thị danh sách loại và thương hiệu, thêm sản phẩm.
- Gợi ý:

```
// danh sách loại
public List<Loai> dsLoai() throws SQLException, ClassNotFoundException {
    List<Loai> rs;
    try (Connection conn = this.connect()) {
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM loai";
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            Loai loai = new Loai();
            loai.setId(resultSet.getInt("id"));
            loai.setTenloai(resultSet.getString("tenloai"));
            rs.add(loai);
        }
    }
    return rs;
}

// danh sách thương hiệu
public List<ThuongHieu> dsThuongHieu() throws SQLException, ClassNotFoundException {
    List<ThuongHieu> rs;
    try (Connection conn = this.connect()) {
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        String sql = "SELECT * FROM thuonghieus";
    }
}
```

```

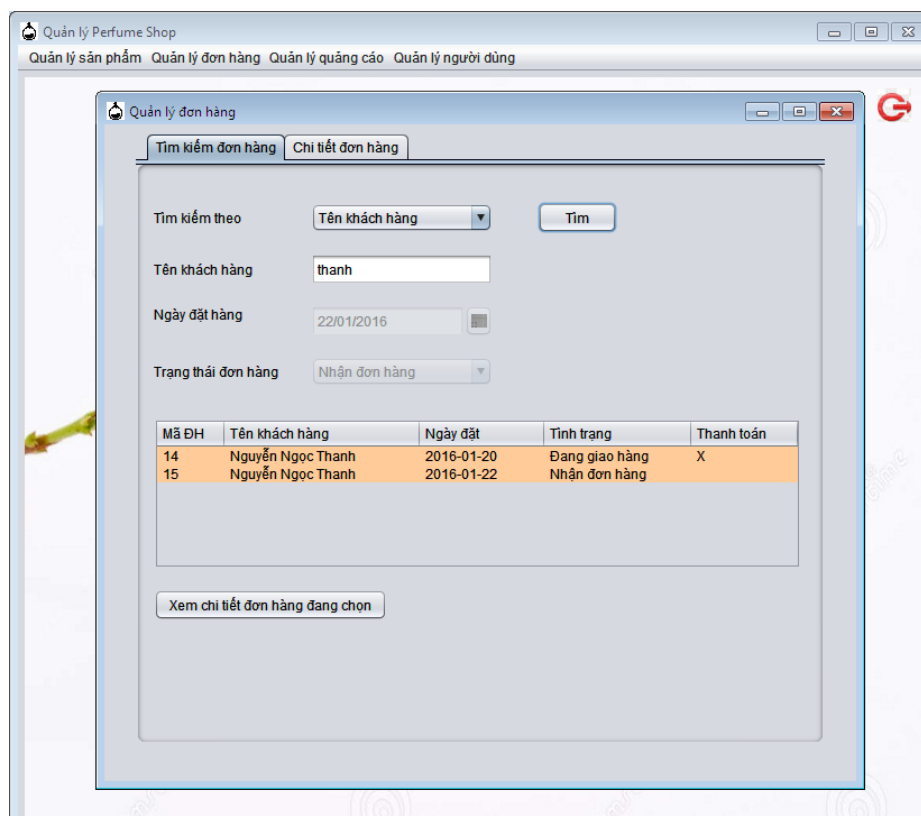
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            ThuongHieu th = new ThuongHieu();
            th.setId(resultSet.getInt("id"));
            th.setTenthuonghieu(resultSet.getString("tenthuonghieu"));
            th.setHinhanh(resultSet.getString("hinhanh"));
            rs.add(th);
        }
    }
    return rs;
}

// thêm mới sản phẩm
public boolean themSanPham(SanPham sanpham) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    boolean execute = false;
    SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-dd");
    try (Connection conn = this.connect()) {
        String sql = "INSERT INTO sanpham VALUES(NULL,'" + sanpham.tensanpham + "','" +
sanpham.mota + "','" + sanpham.hinhanh + "','" + sanpham.dongia + "','" + sanpham.dongiaKM +
 "','" + sanpham.soluong + "','" + df.format(sanpham.ngaytao) + "','" + 1 + "','" + sanpham.id_loai
+ "','" + sanpham.id_thuonghieu + "')";
        PreparedStatement statement = conn.prepareStatement(sql);
        execute = statement.executeUpdate();
    }
    return execute;
}

```

- Xử lý việc đưa danh sách loại/ thương hiệu vào combo box
- Xử lý việc chọn và hiển thị hình ảnh
- Xử lý việc đưa hình ảnh vào thư mục img chung của ứng dụng
- Xử lý việc thêm sản phẩm vào CSDL

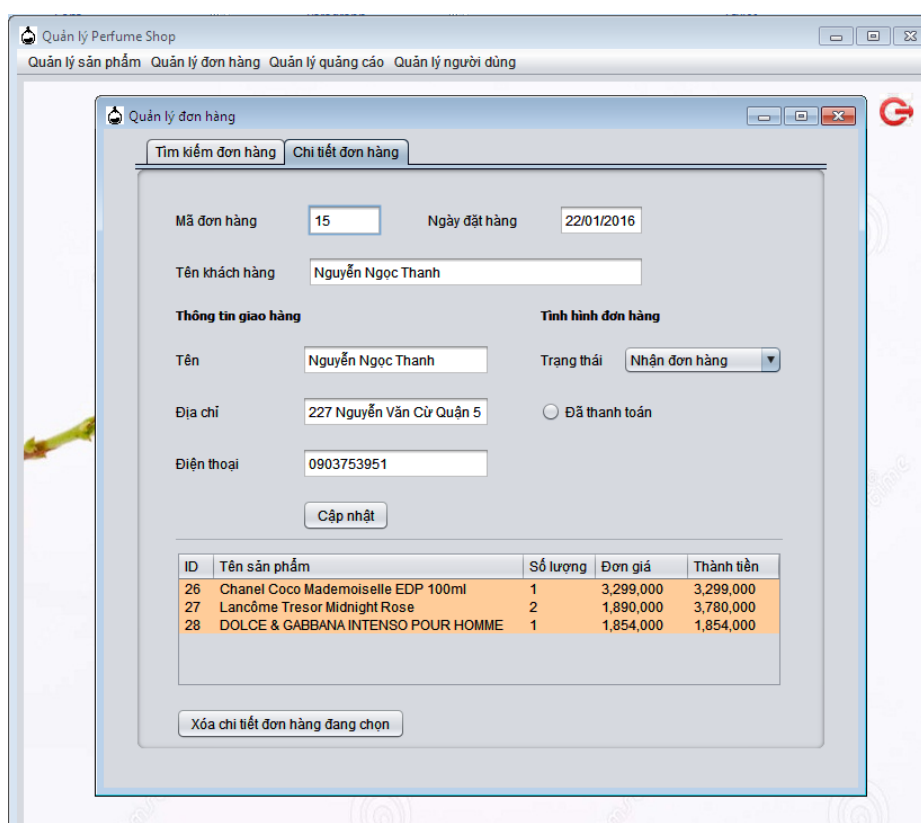
#### 4. Yêu cầu 4: Quản lý đơn hàng



Mã ĐH	Tên khách hàng	Ngày đặt	Tình trạng	Thanh toán
14	Nguyễn Ngọc Thanh	2016-01-20	Đang giao hàng	X
15	Nguyễn Ngọc Thanh	2016-01-22	Nhận đơn hàng	

Xem chi tiết đơn hàng đang chọn

From "Quản lý đơn hàng" sẽ được mở ra khi người dùng chọn chức năng này trên menu. Tab hiển thị là tab "Tìm kiếm đơn hàng"



Mã đơn hàng: 15 Ngày đặt hàng: 22/01/2016

Tên khách hàng: Nguyễn Ngọc Thanh

**Thông tin giao hàng**

Tên: Nguyễn Ngọc Thanh

Địa chỉ: 227 Nguyễn Văn Cừ Quận 5

Điện thoại: 0903753951

**Tình hình đơn hàng**

Trạng thái: Nhận đơn hàng

☐ Đã thanh toán

Cập nhật

ID	Tên sản phẩm	Số lượng	Đơn giá	Thành tiền
26	Chanel Coco Mademoiselle EDP 100ml	1	3,299,000	3,299,000
27	Lancôme Tresor Midnight Rose	2	1,890,000	3,780,000
28	DOLCE & GABBANA INTENSO POUR HOMME	1	1,854,000	1,854,000

Xóa chi tiết đơn hàng đang chọn

Tab "Chi tiết đơn hàng"

✓ **Hướng dẫn sử dụng:**

- Chọn tiêu chí tìm kiếm theo Tên khách hàng/ ngày đặt hàng/ trạng thái đơn hàng sau đó điền và chọn thông tin thích hợp > Nhấn "Tìm" > Danh sách đơn hàng phù hợp với thông tin tìm kiếm sẽ hiển thị trên table.
- Chọn một đơn hàng trên table sau đó chọn tab "Chi tiết đơn hàng" hoặc nhấn nút "Xem chi tiết đơn hàng đang chọn" > hiển thị thông tin của đơn hàng kèm theo chi tiết đơn hàng.
- Cập nhật thông tin giao hàng và chọn tình trạng giao hàng > nhấn "Cập nhật" > Thông tin đơn hàng sẽ được cập nhật.
- Chọn một chi tiết đơn hàng > Nhấn "Xóa chi tiết đơn hàng đang chọn" > Chi tiết đơn hàng sẽ được xóa khi đơn hàng còn ở trạng thái "Nhận đơn hàng"

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- Học viên thiết kế giao diện Quản lý đơn hàng như hình phía trên. Sử dụng thêm điều khiển Date Chooser Combo để chọn ngày.
- Form này là form nằm trong form chính, được mở ra khi người dùng chọn chức năng "Quản lý đơn hàng" trên menu.

▪ **Nhập:**

- Tên khách hàng/ ngày đặt hàng/ trạng thái đơn hàng

▪ **Xuất:**

- Danh sách đơn hàng phù hợp với tiêu chí tìm kiếm

✓ **Hướng dẫn**

- Trong lớp QLCSDDL, xây dựng thêm các phương thức phù hợp cho việc hiển thị danh sách trạng thái đơn hàng, tìm kiếm đơn hàng, cập nhật đơn hàng, chi tiết đơn hàng, xóa chi tiết đơn hàng
- Gợi ý:

```
// danh sách đơn hàng theo SQL
public List<DonHang> dsDonHangTheoSQL(String sql) throws SQLException, ClassNotFoundException
{
    List<DonHang> rs;
    try (Connection conn = this.connect()) {
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            DonHang dh = new DonHang();
            dh.setId(resultSet.getInt("id"));
            dh.setId_khachhang(resultSet.getInt("id_khachhang"));
            dh.setDiachigiaohang(resultSet.getString("diachigiaohang"));
            dh.setDienthoainguoinhan(resultSet.getString("dienthoainguoinhan"));
            dh.setGhichu(resultSet.getString("ghichu"));
            dh.setId_trangthai(resultSet.getInt("id_trangthai"));
            dh.setNgaydathang(resultSet.getDate("ngaydathang"));
            dh.setTenkhachhang(resultSet.getString("hoten"));
            dh.setTennguoinhanhang(resultSet.getString("tennguoinhanhang"));
        }
    }
}
```

```

        dh.setThanhtoan(resultSet.getInt("thanhtoan"));
        rs.add(dh);
    }
}
return rs;
}

// lấy đơn hàng theo ID
public DonHang layDonHangTheoID(int ID) throws SQLException, ClassNotFoundException {
    DonHang dh = null;
    try (Connection conn = this.connect()) {
        java.sql.Statement statement = conn.createStatement();
        ResultSet resultSet = statement.executeQuery("Select * from donhang inner join nguoidung
on nguoidung.id = donhang.id_khachhang where donhang.id = " + ID + "");
        while (resultSet.next()) {
            dh = new DonHang();
            dh.setId(resultSet.getInt("id"));
            dh.setId_khachhang(resultSet.getInt("id_khachhang"));
            dh.setDiachigiaohang(resultSet.getString("diachigiaohang"));
            dh.setDienthoainguoinhan(resultSet.getString("dienthoainguoinhan"));
            dh.setGhichu(resultSet.getString("ghichu"));
            dh.setId_trangthai(resultSet.getInt("id_trangthai"));
            dh.setNgaydathang(resultSet.getDate("ngaydathang"));
            dh.setTenkhachhang(resultSet.getString("hoten"));
            dh.setTennguoinhan(resultSet.getString("tennguoinhanhang"));
            dh.setThanhtoan(resultSet.getInt("thanhtoan"));
        }
    }
    return dh;
}

// danh sách CTĐH theo sql
public List<ChiTietDonHang> dsCTDHTheoSQL(String sql) throws SQLException,
ClassNotFoundException {
    List<ChiTietDonHang> rs;
    try (Connection conn = this.connect()) {
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            ChiTietDonHang ctdh = new ChiTietDonHang();
            ctdh.setId(resultSet.getInt("id"));

```



```

        ctdh.setId_donhang(resultSet.getInt("id_donhang"));
        ctdh.setId_sanpham(resultSet.getInt("id_sanpham"));
        ctdh.setSoluong(resultSet.getInt("soluong"));
        rs.add(ctdh);
    }
}
return rs;
}

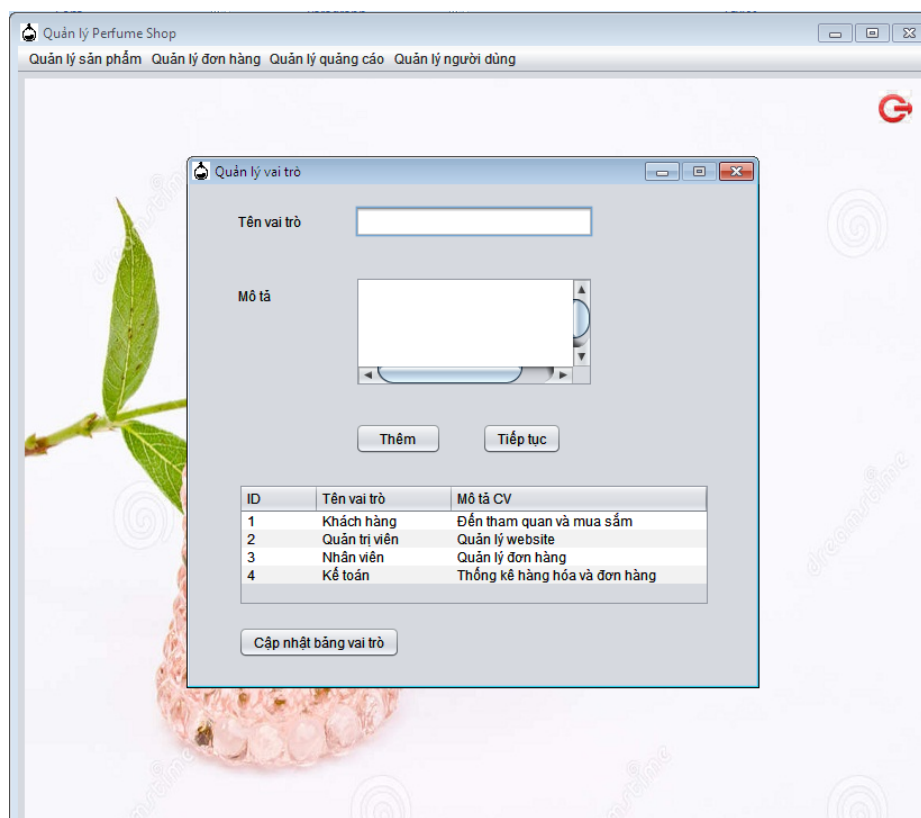
// cập nhật đơn hàng
public void capNhatDonHang(DonHang d) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    try (Connection conn = this.connect()) {
        String sql = "UPDATE DonHang SET tennguoinhanhang = '" + d.getTennguoinhanhang() + "',
dienthoainguoinhan = '" + d.getDienthoainguoinhan() + "', diachigiaohang = '" +
d.getDiachigiaohang() + "', thanhtoan = '" + d.getThanhtoan() + "', id_trangthai = '" +
d.getId_trangthai() + "' WHERE id = '" + d.getId() + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    }
}

// xóa CTĐH
public void xoaCTDH(int id) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    try (Connection conn = this.connect()) {
        String sql = "DELETE FROM chitietdonhang WHERE id = '" + id + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    }
}
}

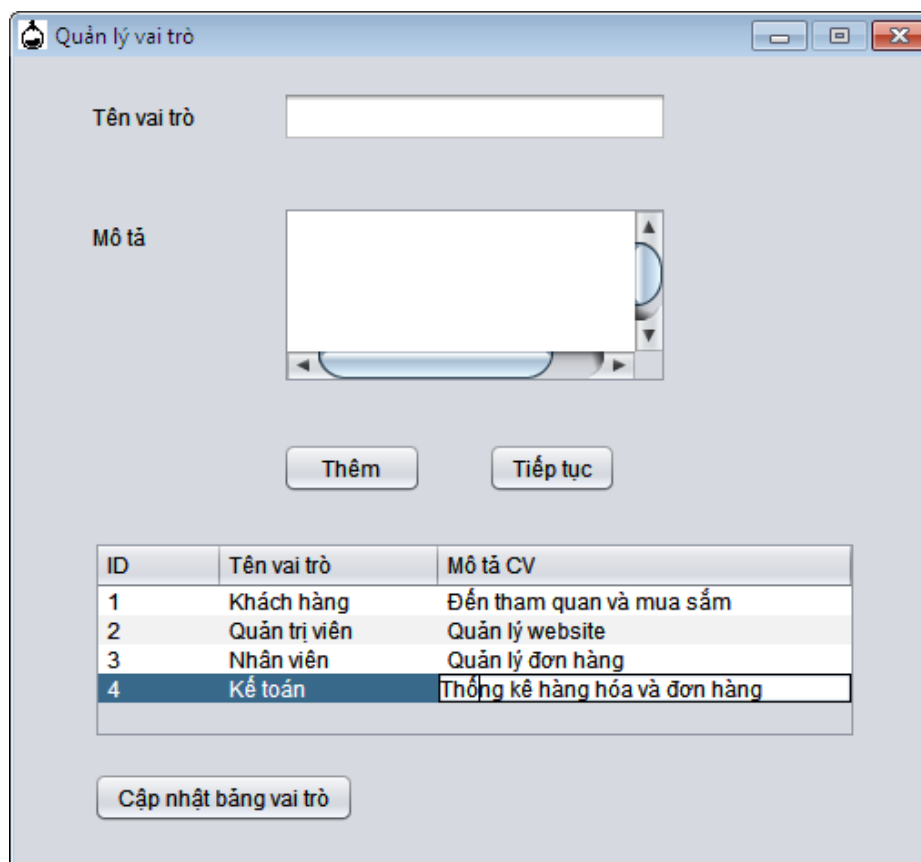
```

- Xử lý việc đưa danh sách trạng thái đơn hàng vào combo box
- Xử lý việc tìm kiếm và in xuất danh sách đơn hàng vào table
- Xử lý việc hiển thị đơn hàng
- Xử lý việc cập nhật đơn hàng
- Xử lý việc hiển thị danh sách chi tiết đơn hàng của đơn hàng
- Xử lý việc xóa chi tiết đơn hàng của đơn hàng

## 5. Yêu cầu 5: Quản lý vai trò



From "Quản lý vai trò" sẽ được mở ra khi người dùng chọn chức năng này trên menu.



Cập nhật vai trò trực tiếp trên table

#### ✓ Hướng dẫn sử dụng:

- Nhập tên vai trò, mô tả > Nhấn "Thêm" > Vai trò mới sẽ được thêm vào CSDL và đồng thời hiển thị trên table.

- Nhấn “Tiếp tục” để xóa nội dung đang có trên các điều khiển
- Cập nhật tên vai trò, mô tả CV trực tiếp trên table > Nhấn “Cập nhật bảng vai trò” để cập nhật nhiều vai trò một lần.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- Học viên thiết kế giao diện Quản lý vai trò như hình phía trên. Form này vừa có thể thêm mới vừa có thể cập nhật trực tiếp trên table.
- Form này là form nằm trong form chính, được mở ra khi người dùng chọn chức năng “Quản lý vai trò” trên menu.

▪ **Nhập:**

- Tên vai trò, mô tả

▪ **Xuất:**

- Các vai trò trên table

✓ **Hướng dẫn**

- Trong lớp QLCSDL, xây dựng thêm phương thức thêm mới và cập nhật vai trò
- Gợi ý:

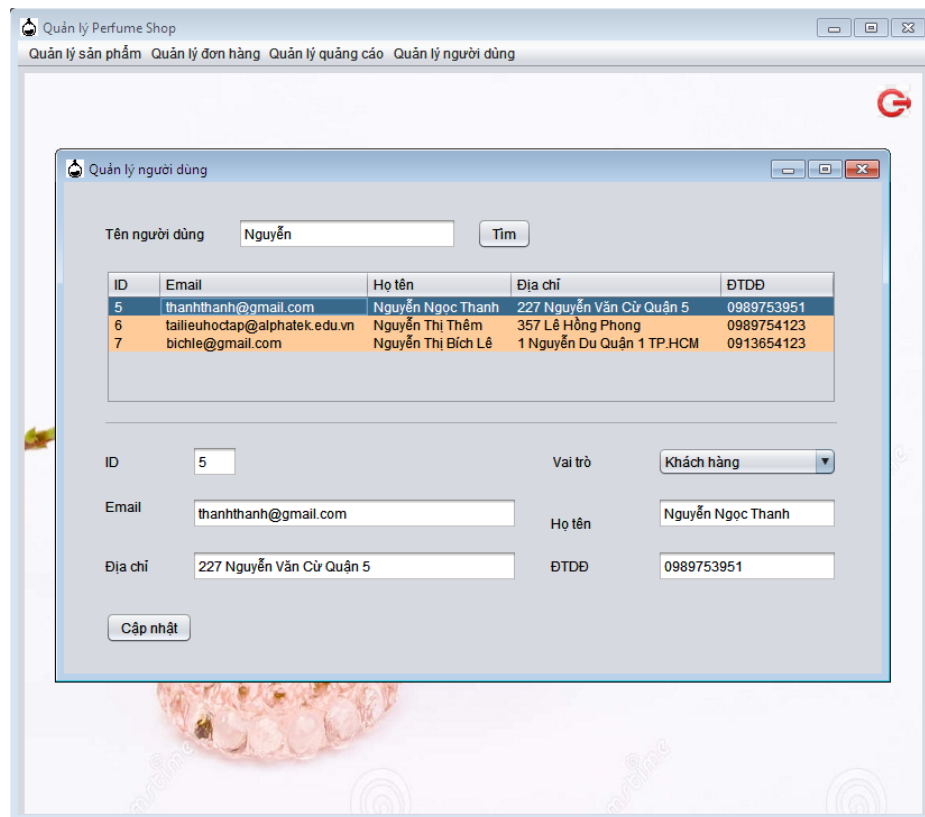
```
// thêm vai trò
public int themVaiTro(VaiTro vt) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    int lastInsertID = 0;
    try (Connection conn = this.connect()) {
        java.sql.Statement statement = conn.createStatement();
        String sql = "INSERT vaitro VALUES(null, '" + vt.getTenvaitro() + "', '" + vt.getMota() + "')";
        statement.executeUpdate(sql);
        ResultSet rs= statement.executeQuery("select id from vaitro order by id desc limit 0,1");
        while (rs.next()) {
            lastInsertID = rs.getInt("id");
            break;
        }
    }
    return lastInsertID;
}

// Cập nhật vai trò
public void capNhatVaiTro(VaiTro vt) throws SQLException, ClassNotFoundException,
UnsupportedEncodingException {
    try (Connection conn = this.connect()) {
        String sql = "UPDATE vaitro SET tenvaitro = '" + vt.tenvaitro + "', mota = '" + vt.mota + "'
WHERE id = '" + vt.getId() + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    }
}
```

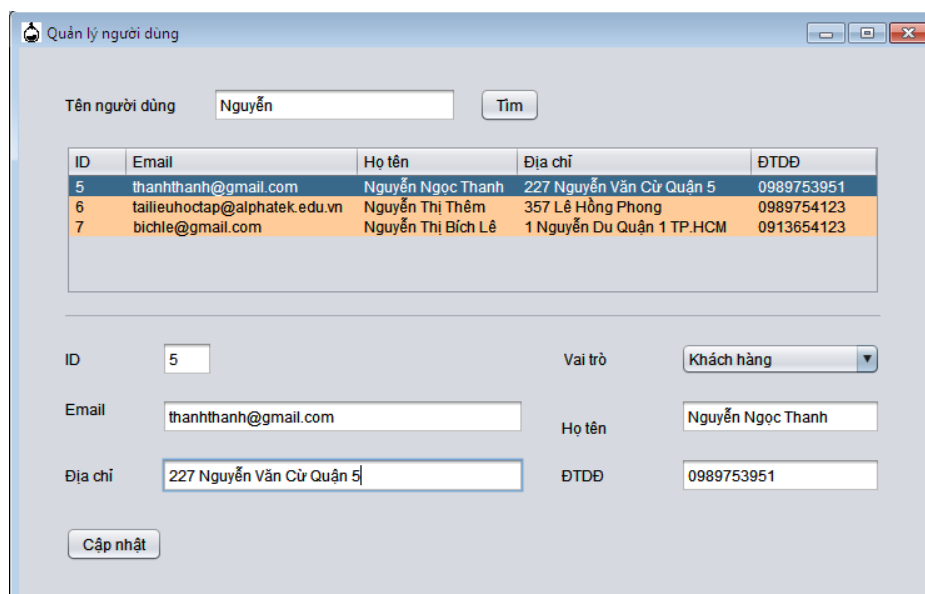
```
}
}
```

- Xử lý việc thêm mới vai trò vào CSDL
- Xử lý việc hiển thị danh sách vai trò trên table
- Xử lý việc cập nhật các vai trò trực tiếp trên table

## 6. Yêu cầu 6: Quản lý người dùng



From "Quản lý người dùng" sẽ được mở ra khi người dùng chọn chức năng này trên menu.



*Cập nhật thông tin người dùng*

### ✓ Hướng dẫn sử dụng:

- Nhập tên người dùng (tìm tương đối) => Nhấn nút "Tìm" => Danh sách người dùng tìm thấy sẽ hiển thị trên table
- Chọn một người dùng trên table => Thông tin người dùng sẽ hiển thị trên các điều khiển => Cập nhật địa chỉ, số điện thoại và vai trò => Nhấn cập nhật.

✓ **Tóm tắt yêu cầu**

▪ **Thiết kế giao diện người dùng:**

- Học viên thiết kế giao diện Quản lý người dùng như hình phía trên.
- Form này là form nằm trong form chính, được mở ra khi người dùng chọn chức năng "Quản lý người dùng" trên menu.

▪ **Nhập:**

- Tên người dùng

▪ **Xuất:**

- Danh sách người dùng thỏa tiêu chí tìm kiếm

✓ **Hướng dẫn**

- Trong lớp QLCSDL, xây dựng thêm phương thức phù hợp cho việc hiển thị danh sách người dùng và cập nhật người dùng.
- Gợi ý:

```
// danh sách người dùng theo SQL
public List<NguoiDung> dsNguoiDung(String sql) throws SQLException, ClassNotFoundException {
    List<NguoiDung> rs;
    try (Connection conn = this.connect()) {
        rs = new ArrayList<>();
        java.sql.Statement statement = conn.createStatement();
        ResultSet resultSet = statement.executeQuery(sql);
        while (resultSet.next()) {
            NguoiDung nd = new NguoiDung();
            nd.setId(resultSet.getInt("id"));
            nd.setEmail(resultSet.getString("email"));
            nd.setPassword(resultSet.getString("password"));
            nd.setHoten(resultSet.getString("hoten"));
            nd.setDiachi(resultSet.getString("diachi"));
            nd.setDtdd(resultSet.getString("dtdd"));
            nd.setId_vaitro(resultSet.getInt("id_vaitro"));
            rs.add(nd);
        }
    }
    return rs;
}

// cập nhật người dùng
public void capNhatNguoiDung(NguoiDung nd) throws SQLException, ClassNotFoundException,
```

```

UnsupportedEncodingException {
    try (Connection conn = this.connect()) {
        String sql = "UPDATE nguoidung SET diachi = '" + nd.getDiachi() + "', dtdd = '" +
        nd.getDtdd() + "', id_vaitro = '" + nd.getId_vaitro() + "' WHERE id = '" + nd.getId() + "'";
        PreparedStatement statement = conn.prepareStatement(sql);
        statement.execute();
    }
}

```

- Xử lý việc tìm kiếm và hiển thị người dùng
- Xử lý việc hiển thị danh sách vai trò vào combobox khi chọn một người dùng trên table
- Xử lý việc cập nhật thông tin người dùng vào CSDL

*Lưu ý: Các yêu cầu chứng năng còn lại, học viên tự thiết kế và xử lý tương tự như những chức năng trên*

# Mục lục

<b>BÀI 1: Biểu thức Lambda .....</b>	<b>2</b>
1.1. Xuất mảng.....	2
1.2. Tính thành tiền .....	2
1.3. Tính diện tích hình tròn, chữ nhật, vuông .....	2
1.4. Tính cộng, trừ, nhân chia hai số .....	3
1.5. Sắp xếp danh sách chuỗi .....	5
<b>BÀI 2: Java Stream API .....</b>	<b>6</b>
2.1. Xử lý chuỗi với Stream.....	6
2.2. Xử lý số với Stream.....	6
2.3. Xử lý đối tượng với Stream.....	7
<b>BÀI 3: Làm việc với dữ liệu JSON .....</b>	<b>9</b>
3.1. Thống kê nhân viên theo đơn vị .....	9
3.2. Ghi dữ liệu quản lý giao dịch vào tập tin tập tin JSON.....	10
3.3. Thêm thông tin phòng karaoke .....	13
<b>BÀI 4: XML DOM .....</b>	<b>15</b>
4.1. Thêm Liên hệ vào tập tin xml .....	15
4.2. Đọc tập tin XML.....	17
<b>BÀI 5: Làm việc với CSDL .....</b>	<b>21</b>
5.1. Làm việc với CSDL MySQL.....	21
5.2. Đăng nhập.....	25
5.3. Thêm Liên hệ vào database .....	29
5.4. Thêm sách vào bảng sách trong CSDL.....	32
5.5. Tìm kiếm sách – xóa sách .....	35
5.6. Cập nhật sách .....	39
<b>BÀI 6: Biểu thức chính quy, Quốc tế hóa – Địa phương hóa .....</b>	<b>44</b>



<b>6.1. Username format .....</b>	<b>44</b>
<b>6.2. Image file format.....</b>	<b>45</b>
<b>6.3. Date format.....</b>	<b>46</b>
<b>6.4. Đa ngôn ngữ .....</b>	<b>47</b>
<b>6.5. Form tìm kiếm vé máy bay .....</b>	<b>50</b>
<b>BÀI 7: Design Pattern.....</b>	<b>55</b>
<b>7.1. Vẽ hình (Draw Shape) .....</b>	<b>55</b>
<b>7.2. Proxy Image .....</b>	<b>59</b>
<b>7.3. Giao dịch qua tài khoản .....</b>	<b>62</b>
<b>7.4. Gửi tài liệu tham khảo .....</b>	<b>66</b>
<b>7.5. Vẽ hình – Factory Design Pattern.....</b>	<b>71</b>
<b>BÀI 8: Thread .....</b>	<b>79</b>
<b>8.1. Ứng dụng đa luồng .....</b>	<b>79</b>
<b>BÀI 9: Tổng kết.....</b>	<b>85</b>
<b>9.1. Xây dựng ứng dụng Quản lý cửa hàng Phương Perfume .....</b>	<b>85</b>
<b>Mục lục.....</b>	<b>103</b>