**TRƯỜNG ĐẠI HỌC PHENIKAA KHOA CÔNG NGHỆ THÔNG TIN**

A logo of a university

Description automatically generated

**BÀI TẬP LỚN**

**Đề tài: Bài toán quản lý phòng GYMS**

**Học phần : Lập Trình Hướng Đối Tượng ( N13 )**

**Giảng viên : Phạm Đức Bắc**

**Nhóm : 12**

**Sinh viên : Nguyễn Thanh Nam - 23010056**

**Hoàng Thị Minh Ngọc – 23010669**

HÀ NỘI, THÁNG 11 Năm 2024

**TRƯỜNG ĐẠI HỌC PHENIKAA KHOA CÔNG NGHỆ THÔNG TIN**

A logo of a university

Description automatically generated

**BÀI TẬP LỚN**

**Đề tài: Bài toán quản lý phòng GYMS**

**Học phần : Lập Trình Hướng Đối Tượng ( N13 )**

**Giảng viên : Phạm Đức Bắc**

**Nhóm 12:**

|  |  |  |  |
| --- | --- | --- | --- |
| Họ tên sinh viên | Mã Số Sinh Viên | Điểm bằng số | Điểm bằng chữ |
| Nguyễn Thanh Nam | 23010056 |  |  |
| Hoàng Thị Minh Ngọc | 23010669 |  |  |

HÀ NỘI, THÁNG 11 Năm 2024

1

MỤC LỤC

Nội dung Trang

1. Mục lục…...……………………………………………………………….. 2
2. GIới thiệu bài toàn ..………………………………………………………. 3
   1. Bối cảnh.………………………………………………………… ……. 3
   2. Mục tiêu của bài toán………………………………………………….. 3
3. Phân tích bài toán………………………………………………………….. 3

2.1 Các tính năng chính……………………………………………………. 3

2.2 Mô tả lớp………………………………………………………………. 6

2.3 Cơ sở dữ liệu…………………………………………………………… 15

3. Cài đặt và sử dụng………………………………………………………….. 16

2

**1.Giới Thiệu Bài Toán**

**1.1 Bối cảnh**

Trong cuộc sống hiện nay, nhu cầu về sức khoẻ và thể dục thể thao ngày càng được quan tâm và chú trọng. Điều này dẫn đến sự phát triển mạnh mẽ của các phòng tập GYMS và trung tập thể dục thể thao, nơi cung cấp các dịch vụ tập luyện đa dạng cho khách hang. Tuy nhiên, khi số lượng hội viên tang lên và hoạt động của mô hình phòng tập trở nên phức tạp, việc quả lý phòng GYMS cũng trở nên khó khăn và cần có một phần mềm quản lý tự động hoá để tăng hiệu quả mô hình.

**1.2 Mục tiêu của bài toán**

Xây dựng một phần mềm quản lý phòng GYMS giúp tối ưu hoá quy trình vận hành, giảm thiểu sai sót và nâng cao hiệu suất quản lý. Hệ thống này sẽ hỗ trợ phòng GYMS trong việc quản lý thông tin huấn luyện viên, hội viện tập luyện, lịch tập, thanh toán, và theo dõi quá trình tập luyện.

**2.Phân tích bài toán**

**2.1.Các tính năng chính**

**Quản lý huấn luyện viên:** Nhập và lưu trữ thông tin huấn luyện viên bao gồm:

* ID huấn luyện viên
* Tên huấn luyện viên
* Địa chỉ huấn luyện viên
* Giới tính huấn luyện viên
* Số điện thoại của huấn luyện viên
* Trạng thái hoạt động của huấn luyện viên

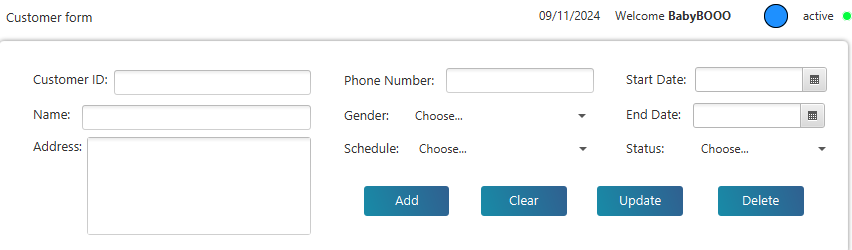
A screenshot of a computer

Description automatically generated

3

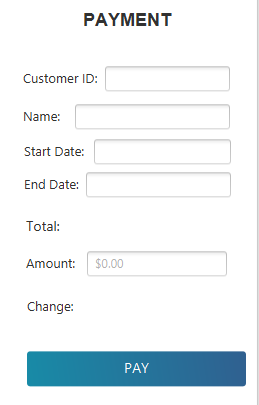
**Quản lý khách hang:** Nhập và lưu trữ thông tin khách hang bao gồm:

* ID khách hàng
* Tên khách hàng
* Số điện thoại khách hàng
* Địa chỉ khách hàng
* Giới tính khách hàng
* Thời gian tập luyện của khách hàng
* Trạng thái hoạt động của khách hàng
* Lựa chọn ngày bắt đầu
* Lựa chọn kết thúc



**Quản lý thanh toán :** Nhập thông tin khách hàng và thanh toán:

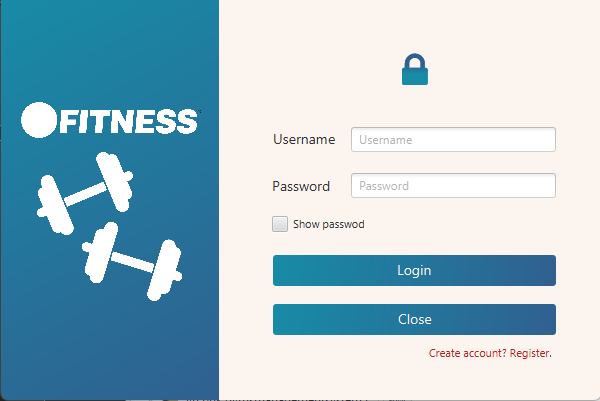
* **ID khách hàng**
* **Tên khách hàng**
* **Ngày bắt đầu**
* **Ngày kết thúc**
* **Số tiền thanh toán**



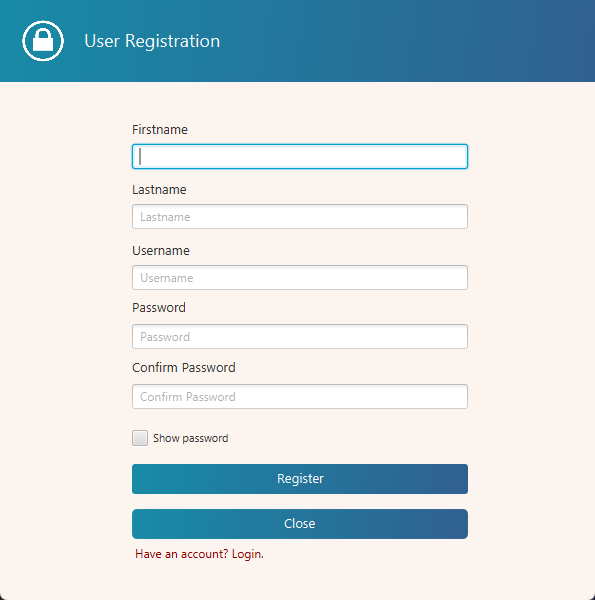
4

**Giao diện người dùng :** Nhập tên đăng nhập và tài khoản:

* Tài khoản chính: admin
* Mật khẩu: password

****

* Nếu chưa có tài khoản bạn có thể đăng kí tại đây:

****

5

**2.2.Mô tả lớp**

1. **Lớp Customer**

* Thuộc tính(Atribute):

public Customer(String customerId, String name, String gender, String phoneNumber, String address, String status, String schedule, String startDate, String endDate)

Constructor này nhận các giá trị khởi tạo cho các thuộc tính của khách hàng. Mỗi giá trị được chuyển vào SimpleStringProperty để hỗ trợ khả năng liên kết dữ liệu của JavaFX.

* **Các phương thức (Methods)**

1. **Getter và Property Methods**: Các phương thức getter và property trả về giá trị của các thuộc tính cũng như StringProperty tương ứng, giúp dễ dàng truy cập hoặc liên kết dữ liệu từ lớp này với giao diện người dùng trong JavaFX.
   * Ví dụ về getter:

java

Copy code

public String getCustomerId() {

return customerId.get();

}

* + Ví dụ về property:

java

Copy code

public StringProperty customerIdProperty() {

return customerId;

}

1. **Setter Methods**:
   * setGender(String gender): Thiết lập giới tính của khách hàng.
   * setStatus(String status): Thiết lập trạng thái của khách hàng.

Các setter này cho phép cập nhật giá trị các thuộc tính có thể thay đổi của khách hàng.

1. **calculateDaysBetweenStartAndEnd()**: Phương thức này tính số ngày giữa startDate và endDate, trả về kết quả dưới dạng long.
   * **Xử lý lỗi**: Nếu định dạng ngày không hợp lệ, phương thức sẽ in thông báo lỗi và trả về -1 để biểu thị lỗi trong tính toán.

6

Java

Copy code

public long calculateDaysBetweenStartAndEnd() {

try {

LocalDate start = LocalDate.parse(startDate.get());

LocalDate end = LocalDate.parse(endDate.get());

return ChronoUnit.DAYS.between(start, end);

} catch (DateTimeParseException e) {

System.err.println(“Error parsing dates: “ + e.getMessage());

return -1;

}

}

* Mã code chương trình:

package vn.pnk.gymsmanagementsystem.customer;

import java.time.LocalDate;

import java.time.format.DateTimeParseException;

import java.time.temporal.ChronoUnit;

import javafx.beans.property.SimpleStringProperty;

import javafx.beans.property.StringProperty;

public class Customer {

private final StringProperty customerId;

private final StringProperty name;

private final StringProperty gender;

private final StringProperty phoneNumber;

private final StringProperty address;

private final StringProperty status;

private final StringProperty schedule;

private final StringProperty startDate;

private final StringProperty endDate;

// Constructor

public Customer(String customerId, String name, String gender, String phoneNumber, String address, String status, String schedule, String startDate, String endDate) {

this.customerId = new SimpleStringProperty(customerId);

this.name = new SimpleStringProperty(name);

this.gender = new SimpleStringProperty(gender);

this.phoneNumber = new SimpleStringProperty(phoneNumber);

this.address = new SimpleStringProperty(address);

this.status = new SimpleStringProperty(status);

this.schedule = new SimpleStringProperty(schedule);

this.startDate = new SimpleStringProperty(startDate);

this.endDate = new SimpleStringProperty(endDate);

}

// Getters

public String getCustomerId() {

return customerId.get();

}

7

public String getName() {

return name.get();

}

public String getGender() {

return gender.get();

}

public String getPhoneNumber() {

return phoneNumber.get();

}

public String getAddress() {

return address.get();

}

public String getStatus() {

return status.get();

}

public String getSchedule() {

return schedule.get();

}

public String getStartDate() {

return startDate.get();

}

public String getEndDate() {

return endDate.get();

}

// Property methods for data binding

public StringProperty customerIdProperty() {

return customerId;

}

public StringProperty nameProperty() {

return name;

}

public StringProperty genderProperty() {

return gender;

}

public StringProperty phoneNumberProperty() {

return phoneNumber;

}

public StringProperty addressProperty() {

return address;

}

8

public StringProperty statusProperty() {

return status;

}

public StringProperty scheduleProperty() {

return schedule;

}

public StringProperty startDateProperty() {

return startDate;

}

public StringProperty endDateProperty() {

return endDate;

}

// Setters for properties that may change

public void setGender(String gender) {

this.gender.set(gender);

}

public void setStatus(String status) {

this.status.set(status);

}

// Method to calculate the number of days between start and end dates

public long calculateDaysBetweenStartAndEnd() {

try {

LocalDate start = LocalDate.parse(startDate.get());

LocalDate end = LocalDate.parse(endDate.get());

return ChronoUnit.DAYS.between(start, end);

} catch (DateTimeParseException e) {

System.err.println("Error parsing dates: " + e.getMessage());

return -1;

}

}

}

1. Lớp User

**Thuộc tính (Attributes)**

1. **id**: ID của người dùng, là một số nguyên để định danh duy nhất mỗi người dùng.
2. **firstname**: Tên của người dùng.
3. **lastname**: Họ của người dùng.
4. **username**: Tên đăng nhập của người dùng, được sử dụng để đăng nhập vào hệ thống.
5. **password**: Mật khẩu của người dùng.

9

**Constructor**

* **Constructor mặc định**:

java

Copy code

public User() {}

Constructor không tham số này cần thiết cho quá trình tuần tự hóa và giải tuần tự hóa XML.

* **Constructor tham số**:

java

Copy code

public User(int par, String username, String password, String lastname, String firstname, int id) {

this.id = id;

this.firstname = firstname;

this.lastname = lastname;

this.username = username;

this.password = password;

}

Khởi tạo đối tượng User với các giá trị cung cấp cho tất cả các thuộc tính.

**Phương thức (Methods)**

1. **Getter và Setter**:
   * Các phương thức getter có annotation @XmlElement để xác định rằng các thuộc tính này sẽ là phần tử con khi đối tượng User được chuyển đổi sang XML.

java

Copy code

@XmlElement

public String getUsername() {

return this.username;

}

public void setUsername(String username) {

this.username = username;

}

Các getter trả về giá trị của thuộc tính và setter cho phép thiết lập giá trị của thuộc tính đó.

10

* Mã Code:

package vn.pnk.gymsmanagementsystem.User;

import jakarta.xml.bind.annotation.XmlElement;

import jakarta.xml.bind.annotation.XmlRootElement;

@XmlRootElement

public class User {

private int id;

private String firstname;

private String lastname;

private String username;

private String password;

public User(){

}

public User (int par, String username, String password, String lastname, String firstname, int id) {

this.id = id;

this.firstname = firstname;

this.lastname = lastname;

this.username = username;

this.password = password;

}

@XmlElement

public String getUsername() {

return this.username;

}

public void setUsername(String username) {

this.username = username;

}

@XmlElement

public String getPassword() {

return this.password;

}

public void setPassword(String password) {

this.password = password;

}

@XmlElement

public String getLastname(){

return this.lastname;

}

11

public void setLastname(String lastname){

this.lastname = lastname;

}

@XmlElement

public String getFirstname(){

return this.firstname;

}

public void setFirstname(String firstname){

this.firstname = firstname;

}

@XmlElement

public int getId(){

return this.id;

}

public void setId(int id){

this.id = id;

}

}

1. Lớp User

**Thuộc tính**

1. **coachId**: Mã định danh của huấn luyện viên, giúp phân biệt các huấn luyện viên với nhau.
2. **name**: Tên của huấn luyện viên.
3. **gender**: Giới tính của huấn luyện viên.
4. **phoneNumber**: Số điện thoại của huấn luyện viên, dùng để liên lạc.
5. **address**: Địa chỉ của huấn luyện viên.
6. **status**: Tình trạng công việc của huấn luyện viên (có thể là Active, Inactive, v.v.).

**Constructor**

* **Constructor với tham số**:

java

Copy code

public Coach(String coachId, String name, String gender, String phoneNumber, String address, String status) {

this.coachId = new SimpleStringProperty(coachId);

this.name = new SimpleStringProperty(name);

this.gender = new SimpleStringProperty(gender);

this.phoneNumber = new SimpleStringProperty(phoneNumber);

this.address = new SimpleStringProperty(address);

this.status = new SimpleStringProperty(status);

}

12

Constructor này khởi tạo đối tượng Coach với các giá trị ban đầu cho tất cả các thuộc tính.

**Getter, Setter và Property Methods**

* **Getter**: Trả về giá trị của các thuộc tính dưới dạng chuỗi (String).
* **Setter**: Cho phép thiết lập giá trị của các thuộc tính.
* **Property Methods**: Cung cấp StringProperty của từng thuộc tính để hỗ trợ liên kết dữ liệu trong JavaFX.

Ví dụ về getter và property method:

java

Copy code

public String getName() {

return name.get();

}

public StringProperty nameProperty() {

return name;

}

public void setName(String name) {

this.name.set(name);

}

Các phương thức này cho phép:

* **Lấy giá trị**: getName() trả về tên hiện tại của huấn luyện viên.
* **Liên kết dữ liệu**: nameProperty() trả về StringProperty, giúp hỗ trợ liên kết với giao diện JavaFX.
* **Cập nhật giá trị**: setName(String name) thay đổi giá trị của name.
* **Mã Code**:

package vn.pnk.gymsmanagementsystem.coaches;

import javafx.beans.property.SimpleStringProperty;

import javafx.beans.property.StringProperty;

public class Coach {

private StringProperty coachId;

private StringProperty name;

private StringProperty gender;

private StringProperty phoneNumber;

private StringProperty address;

private StringProperty status;

13

public Coach(String coachId, String name, String gender, String phoneNumber, String address, String status) {

this.coachId = new SimpleStringProperty(coachId);

this.name = new SimpleStringProperty(name);

this.gender = new SimpleStringProperty(gender);

this.phoneNumber = new SimpleStringProperty(phoneNumber);

this.address = new SimpleStringProperty(address);

this.status = new SimpleStringProperty(status);

}

// Getter và Setter cho tất cả các thuộc tính

public String getCoachId() {

return coachId.get();

}

public StringProperty coachIdProperty() {

return coachId;

}

public void setCoachId(String coachId) {

this.coachId.set(coachId);

}

public String getName() {

return name.get();

}

public StringProperty nameProperty() {

return name;

}

public void setName(String name) {

this.name.set(name);

}

public String getGender() {

return gender.get();

}

public StringProperty genderProperty() {

return gender;

}

public void setGender(String gender) {

this.gender.set(gender);

}

public String getPhoneNumber() {

return phoneNumber.get();

}

14

public StringProperty phoneNumberProperty() {

return phoneNumber;

}

public void setPhoneNumber(String phoneNumber) {

this.phoneNumber.set(phoneNumber);

}

public String getAddress() {

return address.get();

}

public StringProperty addressProperty() {

return address;

}

public void setAddress(String address) {

this.address.set(address);

}

public String getStatus() {

return status.get();

}

public StringProperty statusProperty() {

return status;

}

public void setStatus(String status) {

this.status.set(status);

}

}

**2.3.Cơ sở dữ liệu**

* Dữ liệu được lưu trữ trong các file XML:
* Lớp User được lưu trữ trong file Users.xml
* Lớp Customer được lưu trữ trong file customers.xml
* Lớp Coach được lưu trữ trong file coach.xml
* Các giao dịch sau khi thực hiện được lưu trữ trong file transactions.xml
* Dữ liệu được lưu trữ và cập nhật sau mỗi lần thêm/xoá/cập nhật.
* Các file .xml được tự động khởi tạo nếu không tồn tại.

15

**3.Cài đặt và sử dụng**

Java Development Kit ( JDK ) : phiên bản JDK 23 hoặc mới hơn.

Apache Netbeans : phiên bản 23 hoặc mới hơn.

Hệ điều hành: Windows, macOS hoặc Linux.

Giao diện khi đăng nhập :

Khi mở ứng dụng sẽ xuất hiện màn hình đăng nhập tài khoản:

A screenshot of a login page

Description automatically generated

Khi ta đăng nhập hoặc đăng ký tài khoản ở góc dưới bên phải giao diện chính sẽ hiện lên như thế này:

A screenshot of a computer

Description automatically generated

Bạn lựa chọn các hành động thể thực hiện thao tác.

16

Khi muốn thêm huấn luyện viên ta chọn “ Coaches “ màn hình giao diện hiện lên và ta thao tác như sau:

A screenshot of a computer

Description automatically generated

Khi muốn thêm khách hàng ta chọn “ Customer” màn hình giao diện sẽ hiện lên và ta thao tác như sau:

A screenshot of a computer

Description automatically generated

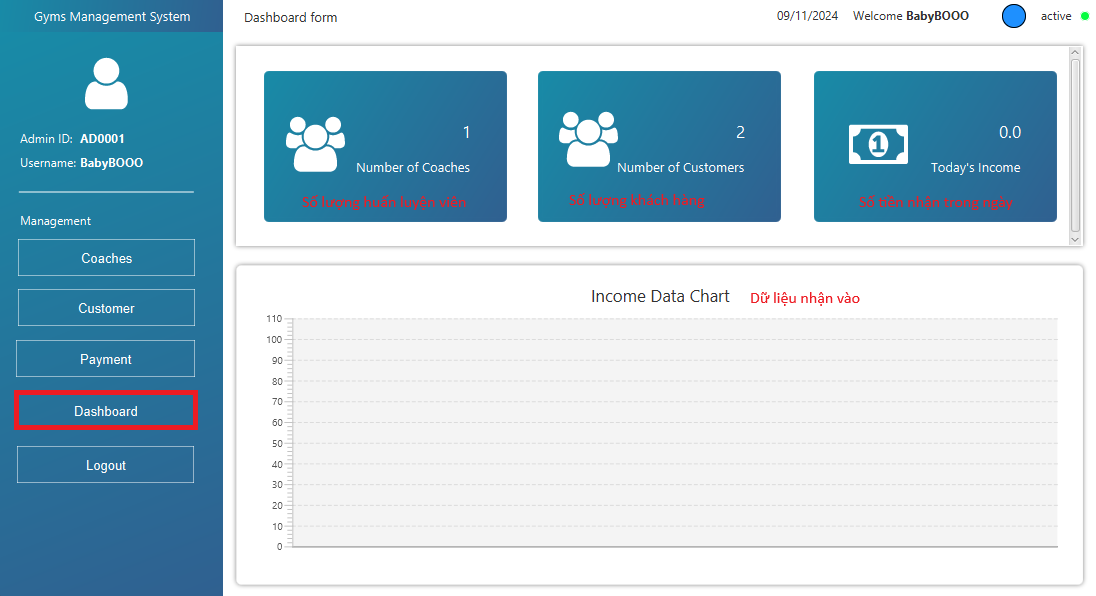
17

Khi muốn thanh toán ta chọn “ Payment” màn hình giao diện sẽ hiện lên và ta thao tác như sau :

A screenshot of a computer

Description automatically generated

Để xem thông tin chung của số lượng khách hàng , số lượng huấn luyện viên, số tiền nhận vào ta chọn “Dashboard” và thông tin hiện ra như sau:



18

Để tìm kiếm khách hàng theo ID của khách hàng ta chọn “Search” để tìm kiếm,  
giao diện tìm kiếm sẽ hiện ra và ta tìm như sau:

A screenshot of a computer

Description automatically generated

Để đăng xuất tài khoản và quay về giao diện đăng nhập ta chọn “Logout” , giao diện lúc này sẽ được quay về giao diện đăng nhập và tài khoản sẽ được đăng xuất.

A screenshot of a computer screen

Description automatically generated

19

**GIÁO VIÊN GIẢNG DẠY**

**Phạm Đức Bắc**

20