|  |
| --- |
|  |
| **DỰ ÁN MẪU** |
| **NGÀNH CÔNG NGHỆ THÔNG TIN (ỨNG DỤNG PHẦN MỀM)** |
|  |



|  |
| --- |
| Hồ Chí Minh, 2021 |

|  |
| --- |
| **GIẢNG VIÊN HƯỚNG DẪN: Thạc Sỹ Lê Anh Tú** |
| **SINH VIÊN THỰC HIỆN: Trần Nguyên Hải**  **MSSV: PS1469** |
|  |
|  |
|  |
|  |

MỤC LỤC

[ĐIỂM CẢI TIẾN CỦA PHẦN MỀM SO VỚI PHIÊN BẢN MẪU 3](#_Toc34246868)

[1 Phân tích 5](#_Toc34246869)

[1.1 Hiện trạng 5](#_Toc34246870)

[1.2 Yêu cầu hệ thống 5](#_Toc34246871)

[1.3 Use case 5](#_Toc34246872)

[2 Thiết kế 9](#_Toc34246873)

[2.1 Mô hình triển khai 9](#_Toc34246874)

[2.2 Thiết kế CSDL 10](#_Toc34246875)

[2.2.1 Sơ đồ quan hệ thực thể 10](#_Toc34246876)

[2.2.2 Thiết kế chi tiết các thực thể 11](#_Toc34246877)

[2.3 Thiết kế giao diện 13](#_Toc34246878)

[2.3.1 Sơ đồ tổ chức giao diện 13](#_Toc34246879)

[2.3.2 Thiết kế giao diện cho các chức năng nghiệp vụ 14](#_Toc34246880)

[3 Thực hiện viết mã 52](#_Toc34246881)

[3.1 Viết mã tạo CSDL 52](#_Toc34246882)

[3.1.1 Tạo CSDL 52](#_Toc34246883)

[3.1.2 SQL truy vấn và thao tác 54](#_Toc34246884)

[3.1.3 Các thủ tục lưu tổng hợp thống kê 57](#_Toc34246885)

[3.2 Lập trình JDBC và các lớp hỗ trợ 58](#_Toc34246886)

[3.2.1 Lớp hỗ trợ 59](#_Toc34246887)

[3.2.2 Model class - Các lớp mô tả dữ liệu 75](#_Toc34246888)

[3.2.3 DAO Class - Các lớp truy xuất dữ liệu 85](#_Toc34246889)

[3.3 Viết mã cho ứng dụng 102](#_Toc34246890)

[3.3.1 Màn hình chào (chaoJDialog) 102](#_Toc34246891)

[3.3.2 Form đăng nhập (dangNhapJDialog) 104](#_Toc34246892)

[3.3.3 Đổi mật khẩu (doiMatKhauJDialog) 106](#_Toc34246893)

[3.3.4 Màn hình giới thiệu (gioiThieuJDialog) 107](#_Toc34246894)

[3.3.5 Form chính (formCha) 108](#_Toc34246895)

[3.3.6 Form nhân viên (nhanVienJInternalFrame) 113](#_Toc34246896)

[3.3.7 Form người học (nguoiHocJInternalFrame) 123](#_Toc34246897)

[3.3.8 Form chuyên đề (chuyenDeJInternalFrame) 132](#_Toc34246898)

[3.3.9 Form khóa học (khoaHocJInternalFrame) 141](#_Toc34246899)

[3.3.10 Form học viên (hocVienJInternalFrame) 149](#_Toc34246900)

[3.3.11 Form thống kê (thongKeJInternalFrame) 154](#_Toc34246901)

[4 Kiểm thử 161](#_Toc34246902)

[4.1 UtilityHelper 161](#_Toc34246903)

[4.2 Kiểm thử form nhân viên 161](#_Toc34246904)

[4.3 KIỂM THỬ FORM NGƯỜI HỌC 161](#_Toc34246905)

[4.4 KIỂM THỬ FORM CHUYÊN ĐỀ 162](#_Toc34246906)

[4.5 KIỂM THỬ FORM KHÓA HỌC 163](#_Toc34246907)

[4.6 KIỂM THỬ FORM HỌC VIÊN 164](#_Toc34246908)

[4.7 KIỂM THỬ FORM THỐNG KÊ 164](#_Toc34246909)

[4.8 KIỂM THỬ FORM ĐĂNG NHẬP 164](#_Toc34246910)

[4.9 KIỂM THỬ FORM ĐỔI MẬT KHẨU 164](#_Toc34246911)

[5 Đóng gói và triển khai 164](#_Toc34246912)

[5.1 Hướng dẫn chuyển đổi jar thành exe 165](#_Toc34246913)

[5.2 DÙNG INNOSETUP ĐÓNG GÓI PHẦN MỀM 180](#_Toc34246914)

[5.3 Hướng dẫn cài đặt triển khai 187](#_Toc34246915)

[5.4 Hướng dẫn sử dụng phần mềm 187](#_Toc34246916)

# Phân tích

## Hiện trạng

Trung tâm đào tạo tin học ngắn hạn LapTrinhCity đang đào tạo các khóa tin học ngắn hạn theo chuyên đề như Photoshop, Java, Web…

Việc quản lý khóa học, học viên, bảng điểm và doanh thu đang thực hiện thông qua excel. Hiện Trung tâm đang gặp khó khăn khi số lượng người học ngày một nhiều, dữ liệu ngày càng lớn nên việc quản lý excel gặp rất nhiều khó khăn, dễ sai sót mà không bảo mật.

## Yêu cầu hệ thống

LapTrinhCity mong muốn xây dựng một phần mềm để giải quyết khó khăn trên.

**⎫ Yêu cầu chức năng nghiệp vụ:**

o Quản lý người đăng ký học

o Quản lý các chuyên đề

o Quản lý các khóa học được tạo ra từ các chuyên đề

o Quản lý học viên của các khóa học o Tổng hợp bảng điểm của các khóa học

o Thống kê doanh thu, số lượng người học từng chuyên đề theo năm

o Tổng hợp số lượng người học đăng ký theo tháng

**⎫ Yêu cầu về bảo mật**

o Tất cả mọi thành viên phải đăng nhập mới sử dụng được phần mềm

o Trưởng phòng đào tạo được phép thực hiện tất cả các chức năng

o Nhân viên phòng đào tạo không được phép xóa dữ liệu và cũng không được xem thông tin về doanh thu

**⎫ Yêu cầu về môi trường công nghệ**

o Ứng dụng phải được thực với công nghệ Swing và JDBC chạy trên mọi hệ điều hành với môi trường JDK tối thiểu 1.8

o Hệ quản trị CSDL SQL Server 2008 trở lên

## Use case

Use case là sơ đồ tổng quan về mặt chức năng và phân vai trò người sử dụng. Dựa vào yêu cầu hệ thống của khách hàng, chúng ta có thể phác thảo sơ đồ use case như sau.



**⎫ CHI TIẾT CÁC CHỨC NĂNG**

o Mỗi chức năng quản lý bao gồm các chức năng con

♣ Xem: xem tất cả và chi tiết một mục

♣ Thêm: thêm mới vào cơ sở dữ liệu

♣ Xóa: xóa theo mã ♣ Sửa: cập nhật dữ liệu đang xem

♣ Tìm kiếm: tìm kiếm theo điều kiện

♣ Điều hướng: di chuyển đến dữ liệu của bản ghi chi tiết trước và sau

♣ Các chức năng thao tác dữ liệu (thêm, sửa, xóa) cần được kiểm lỗi hợp lý với dữ liệu

o Chức năng tổng hợp thống kê số liệu

♣ Bảng điểm theo khóa học phải có cấu trúc là:

• Mã người học

• Họ và tên

• Điểm

♣ Tổng hợp điểm từng chuyên đề

• Chuyên đề

• Số học viên

• Điểm cao nhất

• Điểm thấp nhất

• Điểm trung bình

♣ Doanh thu từng chuyên đề theo năm phải có cấu trúc:

• Chuyên đề

• Số khóa

• Số học viên

• Doanh thu

• Học phí thấp nhất

• Học phí cao nhất

• Học phí trung bình

♣ Thống kê số người học từng năm phải có cấu trúc:

• Năm

• Số học viên

• Ngày đăng ký sớm nhất

• Ngày đăng ký muộn nhất

o Chức năng đăng nhập và đăng xuất

♣ Đăng nhập: để sử dụng phần mềm, bắt buộc phải đăng nhập

♣ Đăng xuất: đăng nhập lại với người dùng khác hoặc dừng lại đi đâu đó

**⎫ CHI TIẾT VỀ YÊU CẦU BẢO MẬT**

o Tất cả phải đăng nhập mới được sử dụng các chức năng trong phần mềm

o Trưởng phòng: được phép sử dụng tất cả

o Nhân viên:

♣ Không được phép thực hiện chức năng xóa trong các chức năng quản lý

♣ Không được phép xem chức năng thống kê doanh thu

**⎫ THÔNG TIN CÁC THỰC THỂ**

o Người học

♣ Mã người đăng ký học

♣ Họ và tên

♣ Ngày sinh

♣ Giới tính

♣ Số điện thoại

♣ Email

♣ Ghi chú

o Chuyên đề

♣ Mã chuyên đề

♣ Tên chuyên đề

♣ Học phí

♣ Thời lượng (tính theo giờ)

♣ Hình logo

♣ Mô tả chuyên đề

o Khóa học

♣ Mã khóa học

♣ Mã chuyên đề

♣ Học phí

♣ Thời lượng

♣ Ngày khai giảng

♣ Ghi chú o Học viên

♣ Mã học viên (số báo danh)

♣ Mã khóa học

♣ Mã người học

♣ Điểm trung bình

o Nhân viên

♣ Mã nhân viên (dùng để đăng nhập)

♣ Mật khẩu

♣ Họ và tên

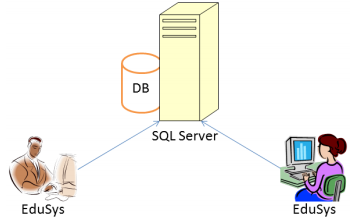
♣ Vai trò (trưởng phòng hay nhân viên)

o Chú ý: Học phí và thời lượng của chuyên đề có thể bị thay đổi mà không làm ảnh hưởng đến học phí và thời lượng của những người đã học.

# Thiết kế

## Mô hình triển khai

Ứng dụng phần mềm được xây dựng để phục vụ cho nhiều người dùng nhưng cơ sở dữ liệu thì lưu trữ tập trung



⎫ Cần một máy cài SQL Server 2008+. Máy này cần hệ điều hành window xp trở lên

⎫ Các máy nhân viên phòng đào tạo cài phần mềm edusys. Các máy nhân viên cần JDK 1.8+ với hệ điều hành bất ký

## Thiết kế CSDL

### Sơ đồ quan hệ thực thể

Trong hệ thống đã được mô tả trên chúng ta thấy có các thực thể sau

⎫ Chuyên đề: quản lý chuyên đề đào tạo ngắn hạn

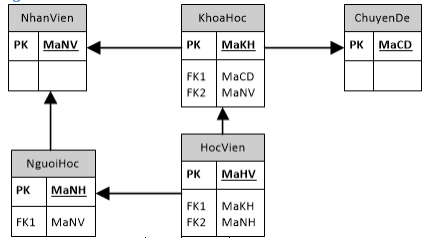
⎫ Khóa học: quản lý thông tin khóa học được tạo ra từ chuyên đề. Có nghĩa là mỗi chuyên đề có thể tạo ra nhiều khóa học.

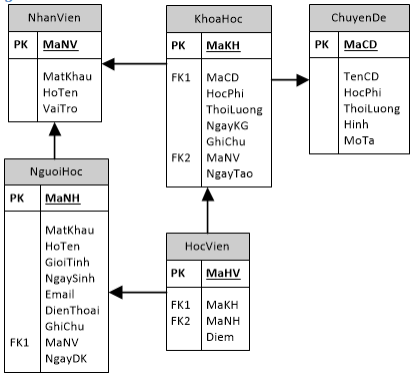
⎫ Người học: quản lý thông tin của người đăng ký học

⎫ Học viên: Học viên là người học đăng ký học chuyên đề. Mỗi học viên có thể đăng ký học nhiều chuyên đề khác nhau mà không cần phải nhập lại.

⎫ Nhân viên: là người quản trị cần có thông tin đăng nhập đồng thời được ghi nhận mỗi khi nhân viên tạo người học mới hoặc khóa học mới.

Sau khi phân tích chúng ta có thể thấy các thực thể có mối quan hệ với nhau như sơ đồ ERD sau



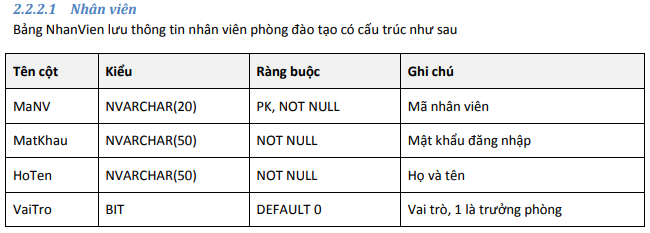


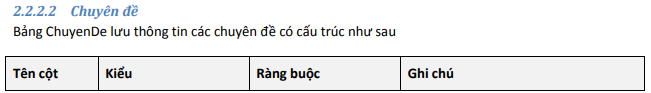
Yêu cầu ràng buộc quan hệ:

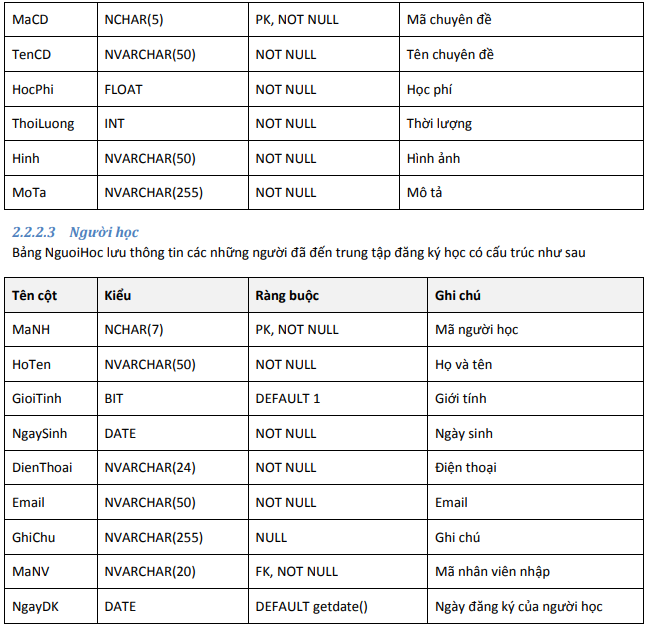
⎫ Tất cả các relationship đều là CASCADE (cập nhật dây chuyền)

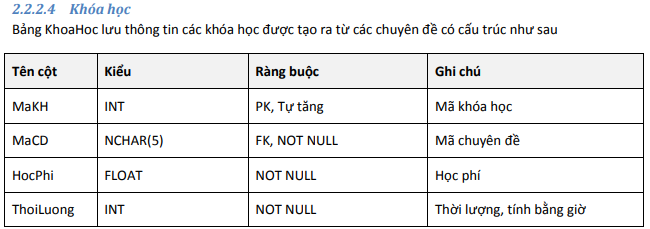
⎫ Tất cả các relationship đều là NO ACTION ngoại trừ KhoaHoc-HocVien là CASCADE

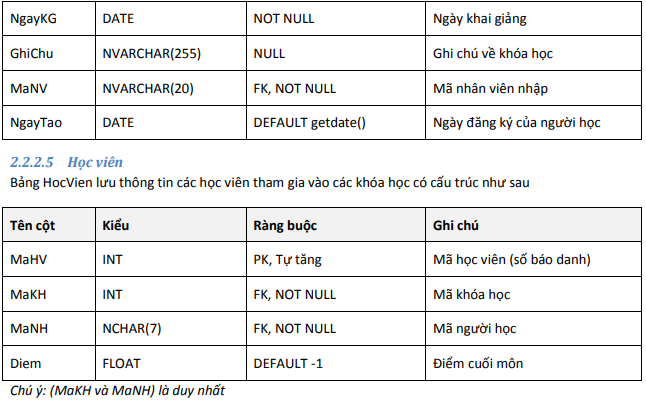
### Thiết kế chi tiết các thực thể











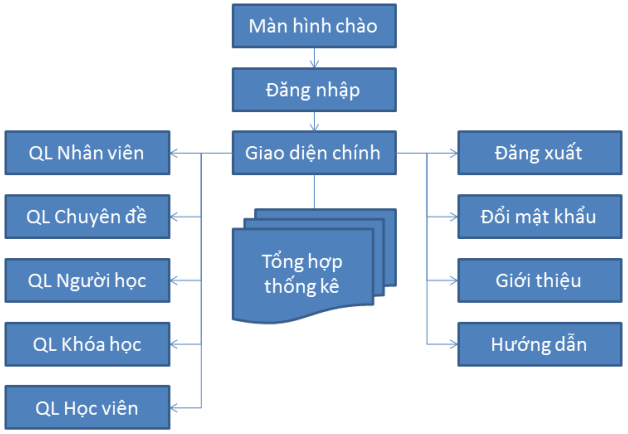
## Thiết kế giao diện

### Sơ đồ tổ chức giao diện

Giao diện chính là một cửa sổ chứa menu chính và toolbar. Thông qua đó để đi đến các giao diện thành viên để thực hiện các chức năng trong hệ thống.

Theo yêu cầu thì mọi nhân viên phải đăng nhập trước khi sử dụng ứng dụng nên form đăng nhập xuất hiện trước để yêu cầu đăng nhập.

Ngoài ra mỗi ứng dụng trong thời gian khởi động cần có một màn hình chào cùng với thanh tiến trình để người có cảm giác ứng dụng đang khởi động.



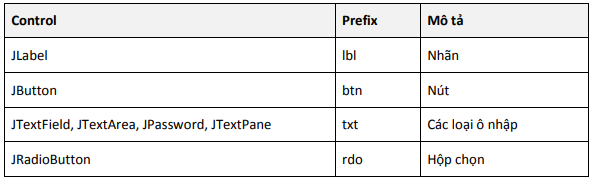
### Thiết kế giao diện cho các chức năng nghiệp vụ

Trong phần thiết kế giao diện các bạn nên sử dụng gói hình ảnh được cung cấp sẵn trên tài nguyên học tập của LMS. Nếu các bạn không muốn sử dụng gói hình ảnh này thì bạn có thể chọn gói hình ảnh phù hợp của các bạn.

Lưu ý quan trọng:

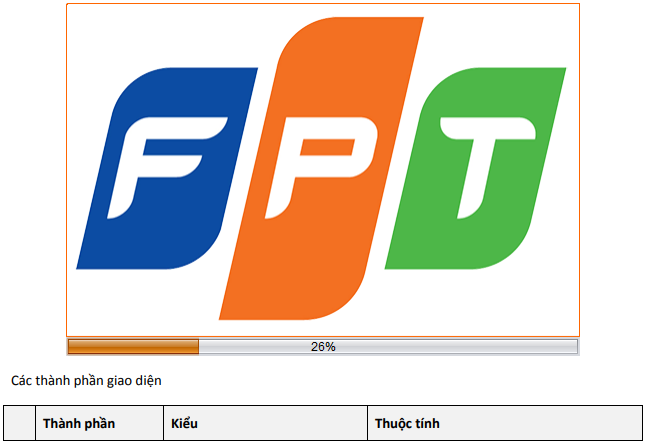
⎫ Các bạn phải đặt tên đúng như hướng dẫn vì các tên này ảnh hưởng đến việc viết mã sau này. Nếu các bạn đặt tên khác thì phải chỉnh mã sau này theo tên mà các bạn đã đặt.

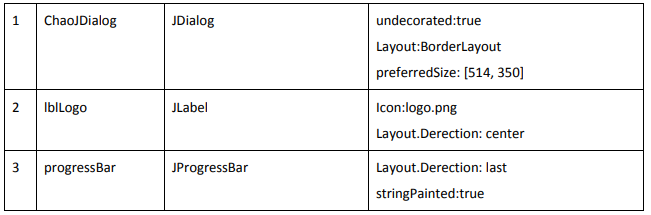
⎫ Trong lập trình giao diện người ta thường qui ước tiếp đầu ngữ của tên các thành phần giao diện như sau



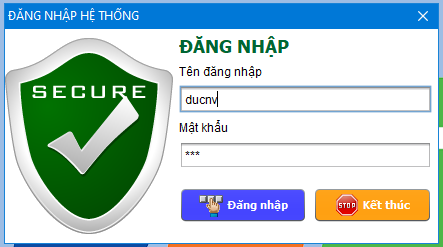


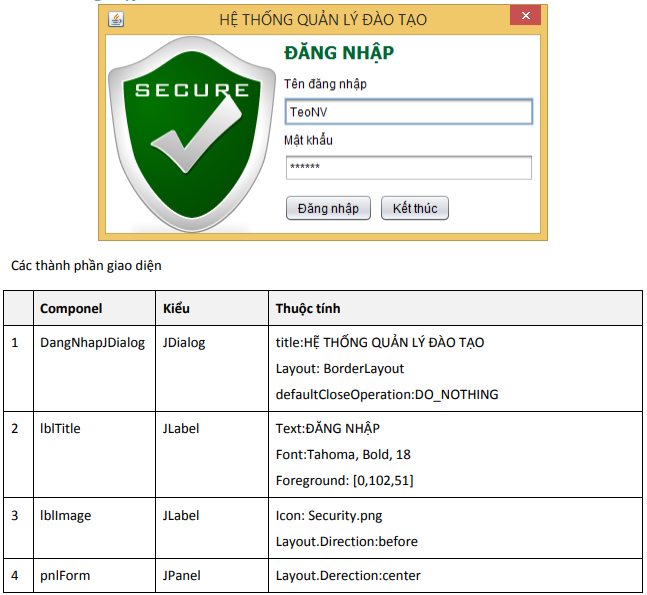
#### Thiết kế màn hình chào

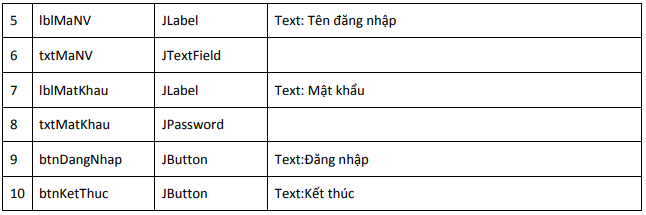




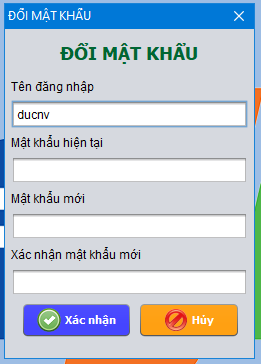
#### Thiết kế Form đăng nhập



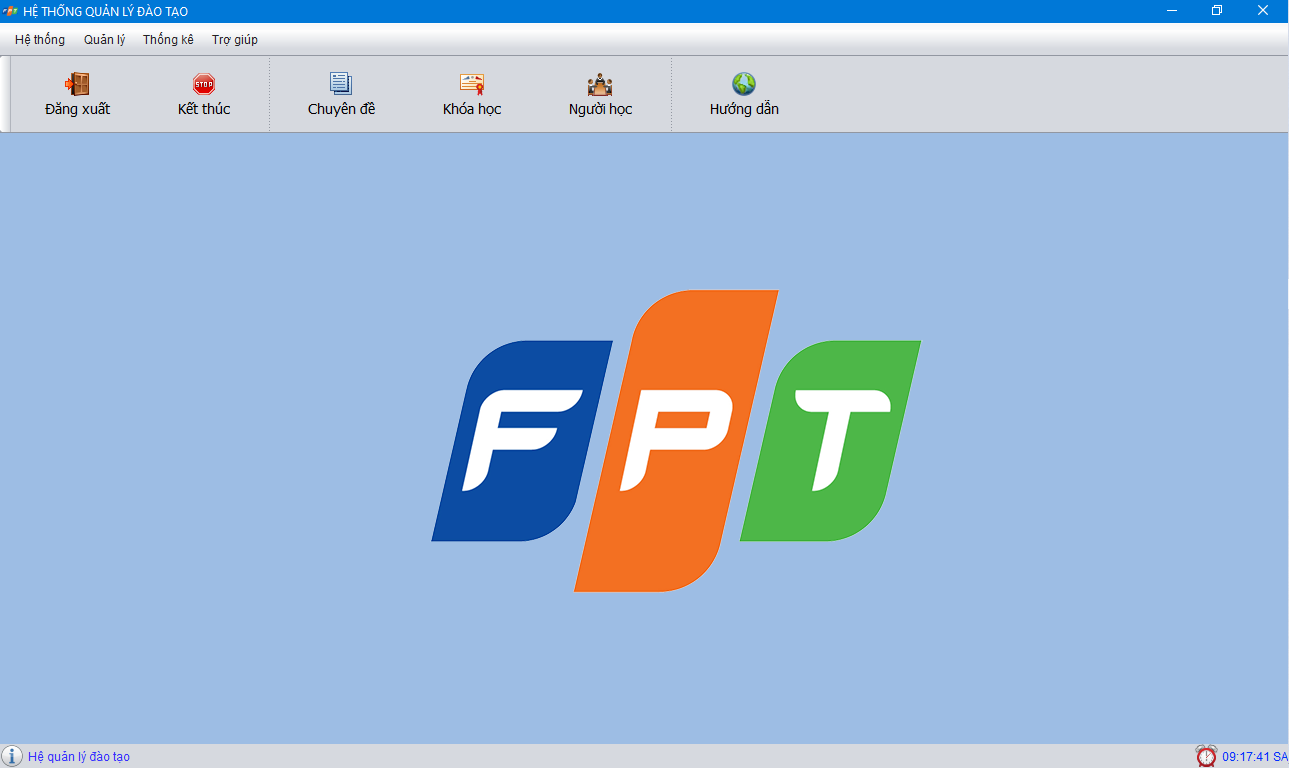


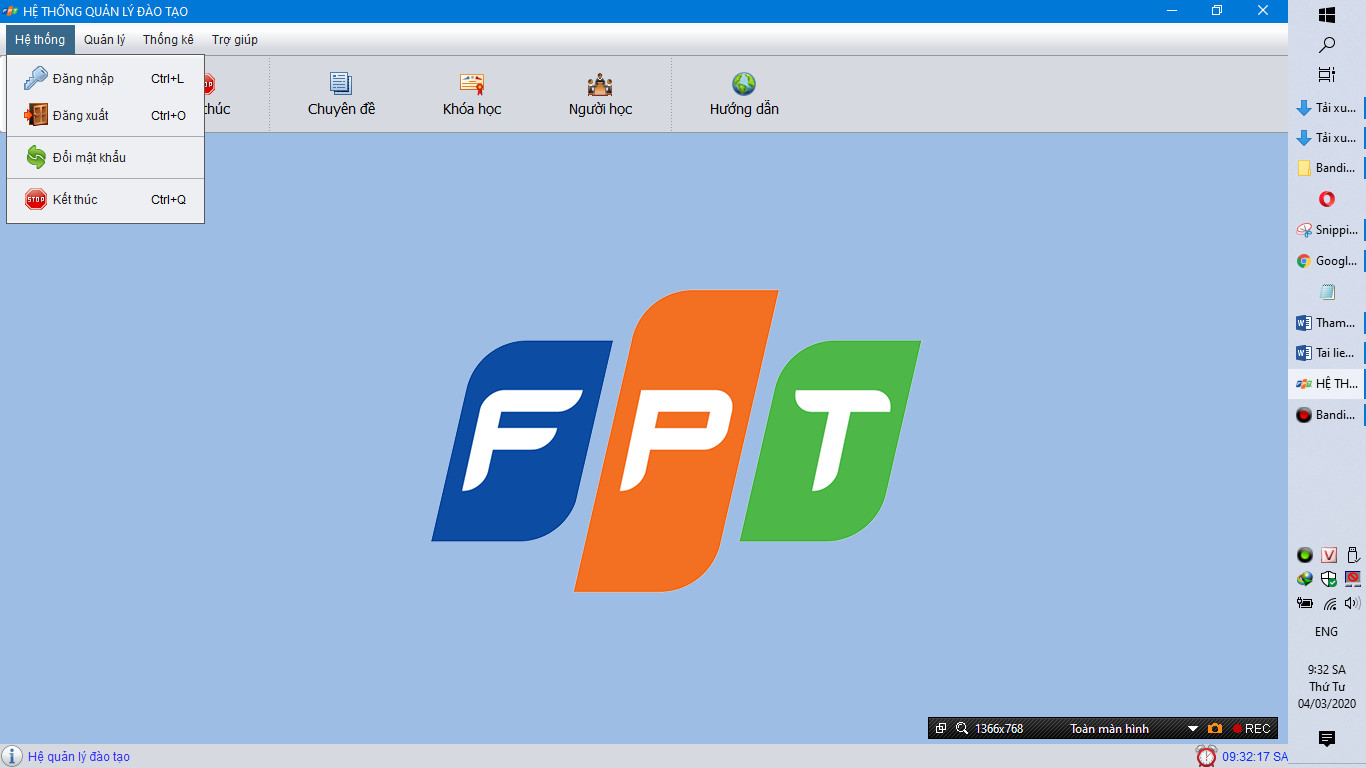


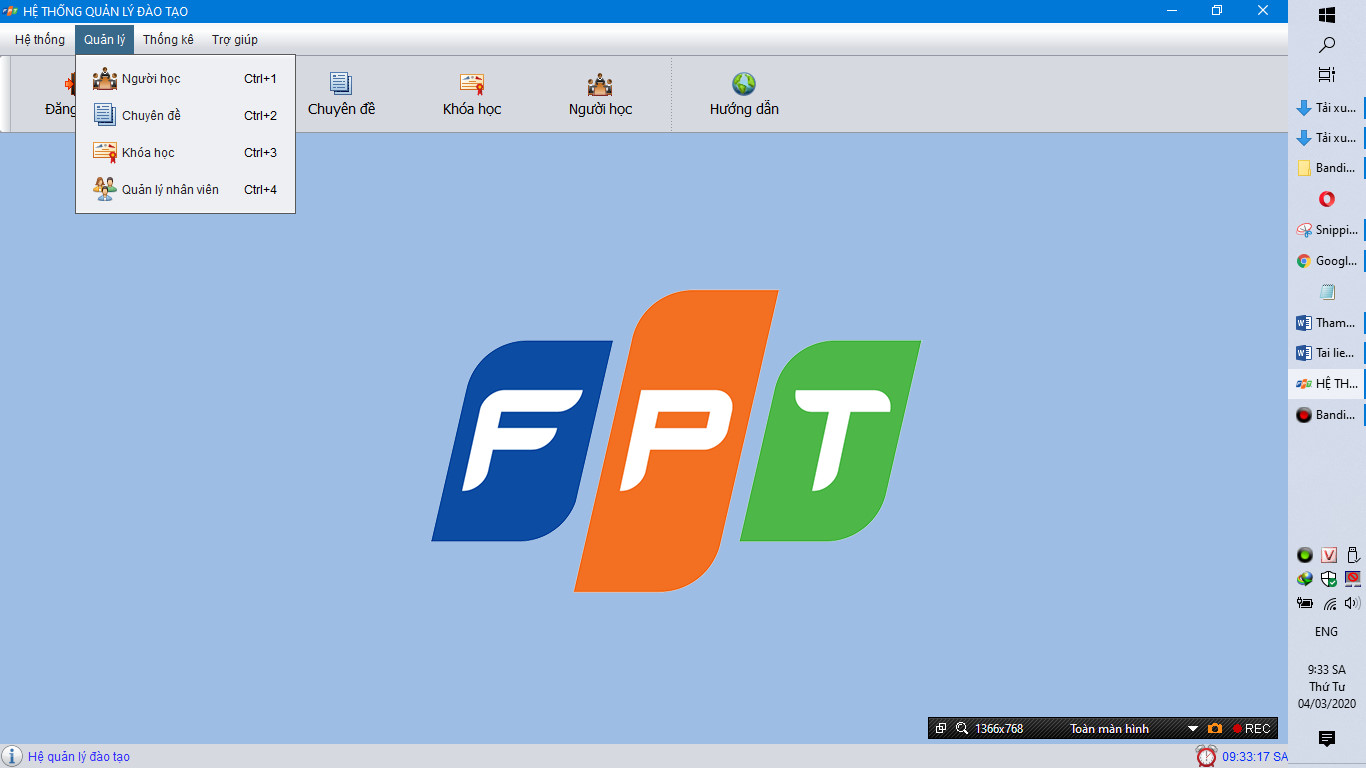
#### Đổi mật khẩu



#### Giao diện chính

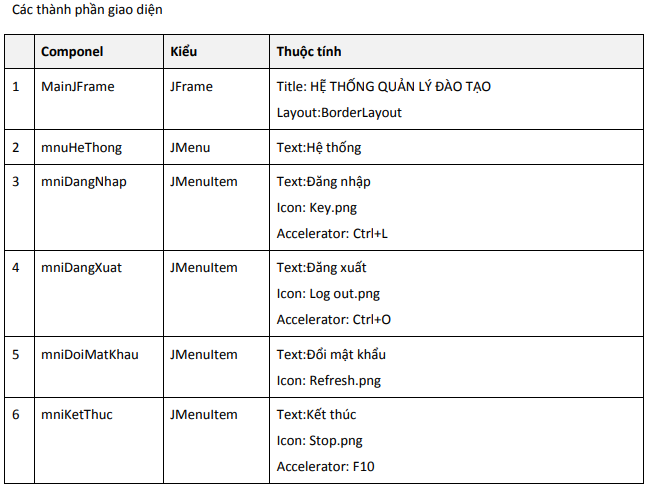




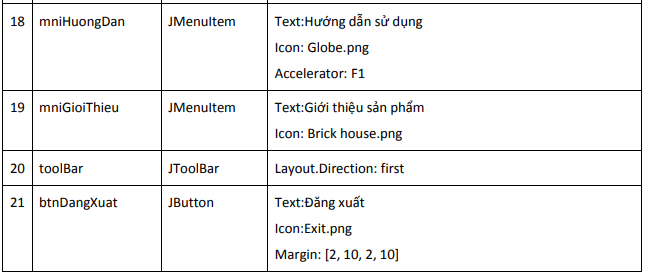


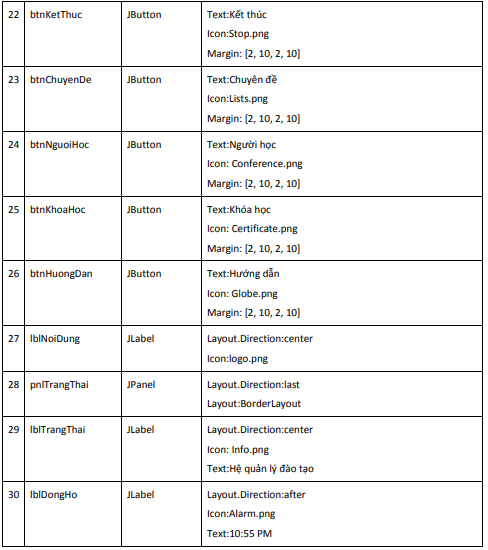






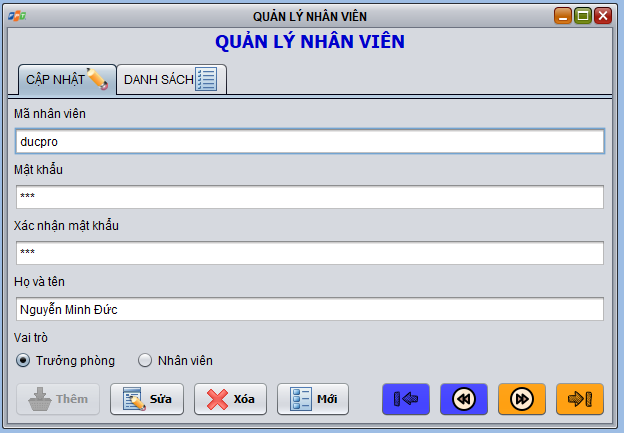






#### Quản lý nhân viên

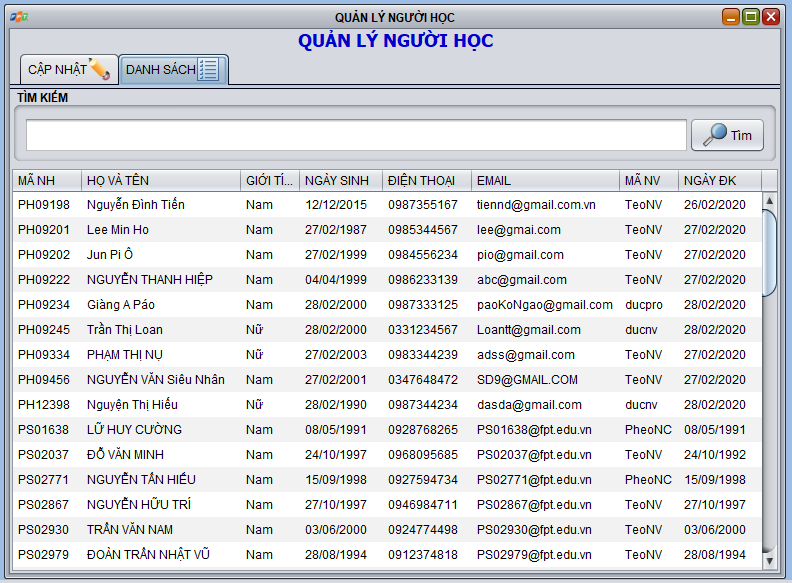


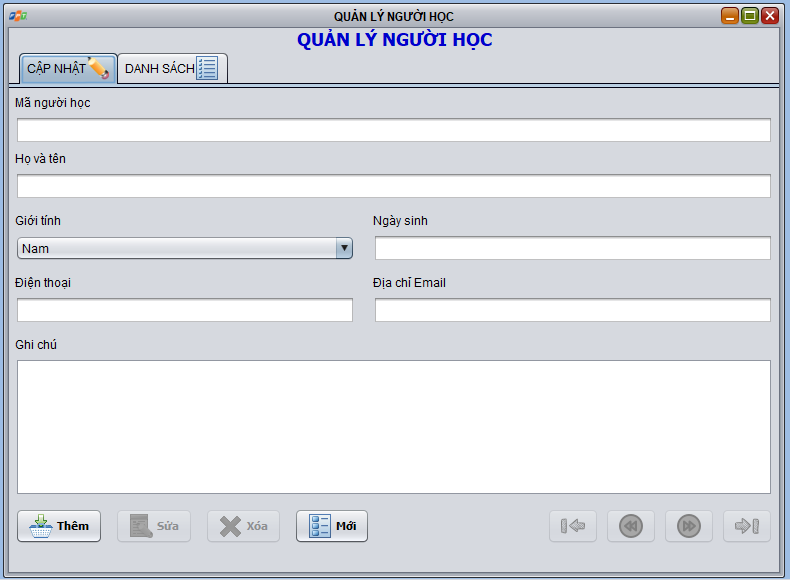


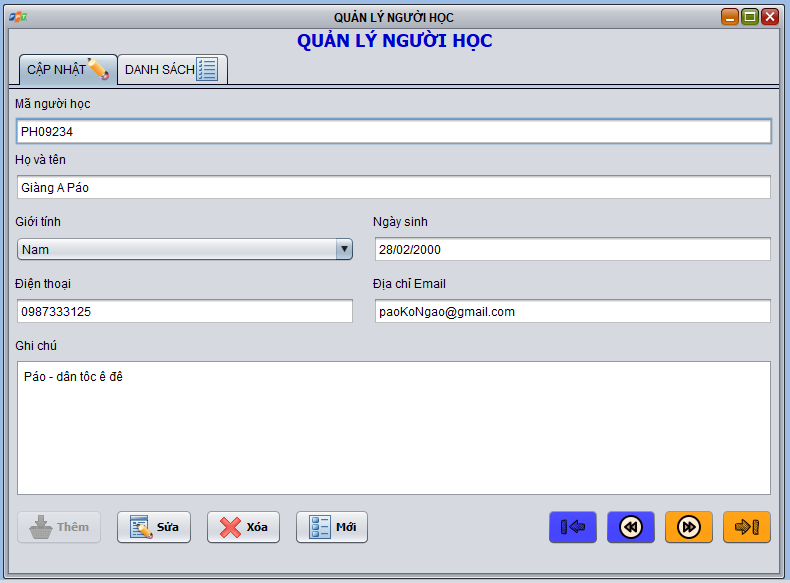
Các thành phần giao diện

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | NhanVienJFrame | JFrame | Title: QUẢN LÝ NHÂN VIÊN |
|  |  |  |  | defaultCloseOperation:DISPONSE |
|  |  |  |  |  |
|  | 2 | lblTitle | JLabel | Text: QUẢN LÝ NHÂN VIÊN QUẢN TRỊ |
|  |  |  |  | Foreground: [0,0,204] |
|  |  |  |  | Font:Tahoma, Bold, 14 |
|  |  |  |  |  |
|  | 3 | tabs | JTabbedPane |  |
|  |  |  |  |  |
|  | 4 | pnlEdit | JPanel | Tab Title: CẬP NHẬT |
|  |  |  |  |  |
|  | 5 | pnlList | JPanel | Tab Title: DANH SÁCH |
|  |  |  |  |  |
|  | 6 | lblMaNV | JLabel | Text: Mã nhân viên |
|  |  |  |  |  |
|  | 7 | txtMaNV | JTextField |  |
|  |  |  |  |  |
| 8 | | lblMatKhau | JLabel | Text: Mật khẩu |
|  | |  |  |  |
| 9 | | txtMatKhau | JPassword |  |
|  | |  |  |  |
| 10 | | lblXacNhanMK | JLabel | Text: Xác nhận mật khẩu |
|  | |  |  |  |
| 11 | | txtXacNhanMK | JPassword |  |
|  | |  |  |  |
| 12 | | lblHoTen | JLabel | Text: Họ và tên |
|  | |  |  |  |
| 13 | | txtHoTen | JTextField |  |
|  | |  |  |  |
| 14 | | lblVaiTro | JLabel | Text: Vai trò |
|  | |  |  |  |
| 15 | | bgrVaiTro | ButtonGroup |  |
|  | |  |  |  |
| 16 | | rdoTruongPhong | JRadioButton | Text: Trưởng phòng |
|  | |  |  | Selected: true |
|  | |  |  | buttonGroup: bgrVaiTro |
|  | |  |  |  |
| 17 | | rdoNhanVien |  | Text: Nhân viên |
|  | |  |  | buttonGroup: bgrVaiTro |
|  | |  |  |  |
| 18 | | btnInsert | JButton | Text:Thêm |
|  | |  |  |  |
| 19 | | btnUpdate | JButton | Text:Sửa |
|  | |  |  |  |
| 20 | | btnDelete | JButton | Text:Xóa |
|  | |  |  |  |
| 21 | | btnClear | JButton | Text:Mới |
|  | |  |  |  |
| 22 | | btnFirst | JButton | Text: |< |
|  | |  |  |  |
| 23 | | btnPrev | JButton | Text: << |
|  | |  |  |  |
| 24 | | btnNext | JButton | Text: >> |
|  | |  |  |  |
| 25 | | btnLast | JButton | Text: >| |
|  | |  |  |  |
| 26 | | tblGridView | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  |  |

#### Quản lý người học







Các thành phần giao diện

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | NguoiHocJFrame | JFrame | Title: QUẢN LÝ NGƯỜI HỌC |
|  |  |  |  | defaultCloseOperation:DISPONSE |
|  |  |  |  |  |
|  | 2 | lblTitle | JLabel | Text: QUẢN LÝ NGƯỜI HỌC |
|  |  |  |  | Foreground: [0,0,204] |
|  |  |  |  | Font:Tahoma, Bold, 14 |
|  |  |  |  |  |
|  | 3 | tabs | JTabbedPane |  |
|  |  |  |  |  |
|  | 4 | pnlEdit | JPanel | Tab Title: CẬP NHẬT |
|  |  |  |  |  |
|  | 5 | pnlList | JPanel | Tab Title: DANH SÁCH |
|  |  |  |  |  |
|  | 6 | lblMaNH | JLabel | Text: Mã người học |
|  |  |  |  |  |
|  | 7 | txtMaNH | JTextField |  |
|  |  |  |  |  |
|  | 8 | lblHoTen | JLabel | Text: Họ và tên |
|  |  |  |  |  |
|  | 9 | txtHoTen | JTextField |  |
|  |  |  |  |  |
|  | 10 | lblGioiTinh | JLabel | Text: Giới tính |
|  |  |  |  |  |
|  | 11 | cboGioiTinh | JComboBox | Model: Nam, Nữ |
|  |  |  |  |  |
|  | 12 | lblNgaySinh | JLabel | Text: Ngày sinh |
|  |  |  |  |  |
|  | 13 | txtNgaySinh | JTextField |  |
|  |  |  |  |  |
|  | 14 | lblDienThoai | JLabel | Text: Điện thoại |
|  |  |  |  |  |
|  | 15 | txtDienThoai | JTextField |  |
|  |  |  |  |  |
|  | 16 | lblEmail | JLabel | Text: Email |
|  |  |  |  |  |
|  | 17 | txtEmail | JTextField |  |
|  |  |  |  |  |
|  | 18 | lblGhiChu | JLabel | Text: Ghi chú |
|  |  |  |  |  |
|  | 19 | txtGhiChu | JTextArea |  |
|  |  |  |  |  |
|  | 20 | btnInsert | JButton | Text:Thêm |
|  |  |  |  |  |
| 21 | | btnUpdate | JButton | Text:Sửa |
|  | |  |  |  |
| 22 | | btnDelete | JButton | Text:Xóa |
|  | |  |  |  |
| 23 | | btnClear | JButton | Text:Mới |
|  | |  |  |  |
| 24 | | btnFirst | JButton | Text: |< |
|  | |  |  |  |
| 25 | | btnPrev | JButton | Text: << |
|  | |  |  |  |
| 26 | | btnNext | JButton | Text: >> |
|  | |  |  |  |
| 27 | | btnLast | JButton | Text: >| |
|  | |  |  |  |
| 28 | | tblGridView | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  |  |
| 28 | | pnlTimKiem | JPanel | Border: TitledBorder(Title: Tìm Kiếm) |
|  | |  |  |  |
| 30 | | txtTimKiem | JTextField |  |
|  | |  |  |  |
| 31 | | btnTimKiem | JButton | Tìm |

#### Quản lý chuyên đề

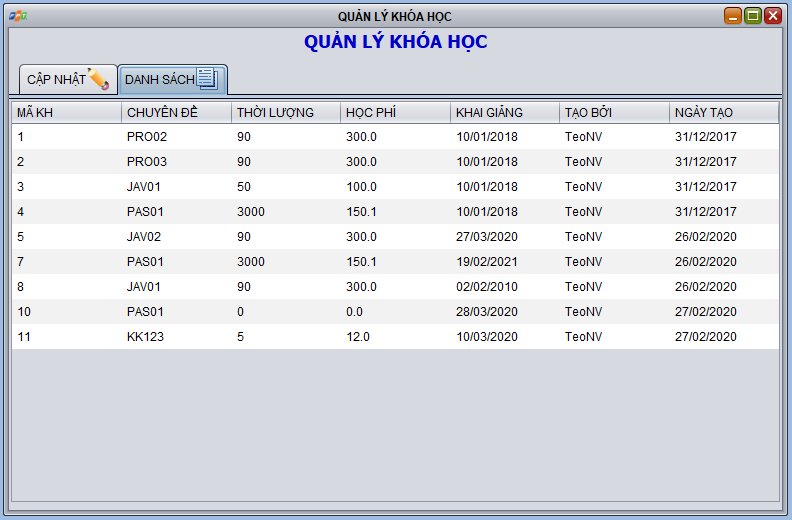


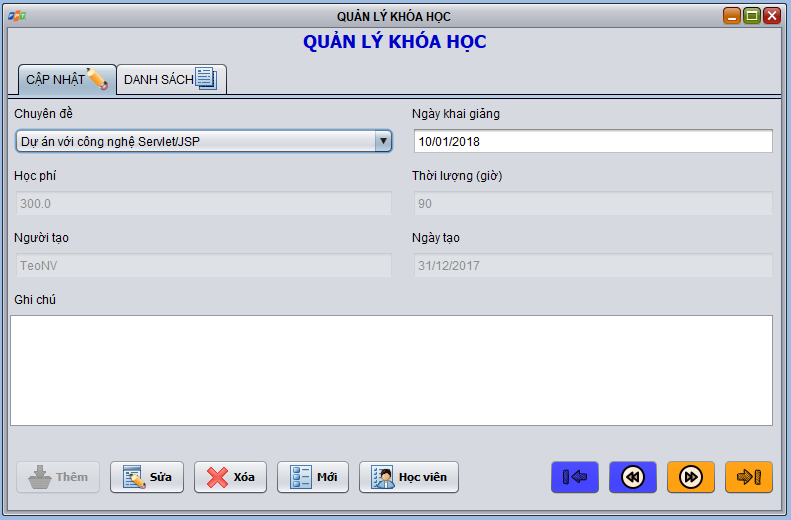


Các thành phần giao diện:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | ChuyenDeJFrame | JFrame | Title: QUẢN LÝ CHUYÊN ĐỀ |
|  |  |  |  | defaultCloseOperation:DISPONSE |
|  |  |  |  |  |
|  | 2 | lblTitle | JLabel | Text: QUẢN LÝ CHUYÊN ĐỀ |
|  |  |  |  | Foreground: [0,0,204] |
|  |  |  |  | Font:Tahoma, Bold, 14 |
|  |  |  |  |  |
|  | 3 | tabs | JTabbedPane |  |
|  |  |  |  |  |
|  | 4 | pnlEdit | JPanel | Tab Title: CẬP NHẬT |
|  |  |  |  |  |
|  | 5 | pnlList | JPanel | Tab Title: DANH SÁCH |
|  |  |  |  |  |
| 6 | | lblMaCD | JLabel | Text: Mã chuyên đề |
|  | |  |  |  |
| 7 | | txtMaCD | JTextField |  |
|  | |  |  |  |
| 8 | | lblTenCD | JLabel | Text: Tên chuyên đề |
|  | |  |  |  |
| 9 | | txtTenCD | JTextField |  |
|  | |  |  |  |
| 10 | | lblHocPhi | JLabel | Text: Học phí |
|  | |  |  |  |
| 11 | | txtHocPhi | JTextField |  |
|  | |  |  |  |
| 12 | | lblThoiLuong | JLabel | Text: Thời lượng (giờ) |
|  | |  |  |  |
| 13 | | txtThoiLuong | JTextField |  |
|  | |  |  |  |
| 14 | | lblMoTa | JLabel | Text: Mô tả chuyên đề |
|  | |  |  |  |
| 15 | | txtMoTa | JTextArea |  |
|  | |  |  |  |
| 16 | | lblHinh | JLabel | Text: Hình logo |
|  | |  |  |  |
| 17 | | lblHinh | JLabel | Text: |
|  | |  |  |  |
| 18 | | btnInsert | JButton | Text:Thêm |
|  | |  |  |  |
| 19 | | btnUpdate | JButton | Text:Sửa |
|  | |  |  |  |
| 20 | | btnDelete | JButton | Text:Xóa |
|  | |  |  |  |
| 21 | | btnClear | JButton | Text:Mới |
|  | |  |  |  |
| 22 | | btnFirst | JButton | Text: |< |
|  | |  |  |  |
| 23 | | btnPrev | JButton | Text: << |
|  | |  |  |  |
| 24 | | btnNext | JButton | Text: >> |
|  | |  |  |  |
| 25 | | btnLast | JButton | Text: >| |
|  | |  |  |  |
| 26 | | tblGridView | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  |  |
| 27 | | fileChooser | JFileChooser | dialogTitle: Chọn logo chuyên đề |
|  | |  |  |  |

#### Quản lý khóa học

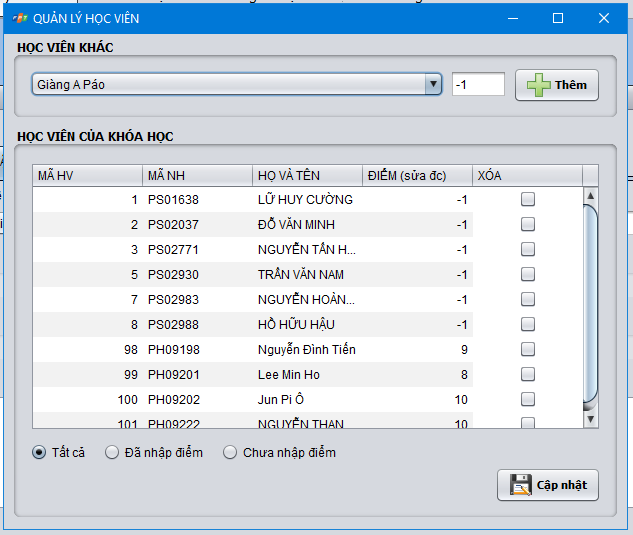




Các thành phần giao diện:

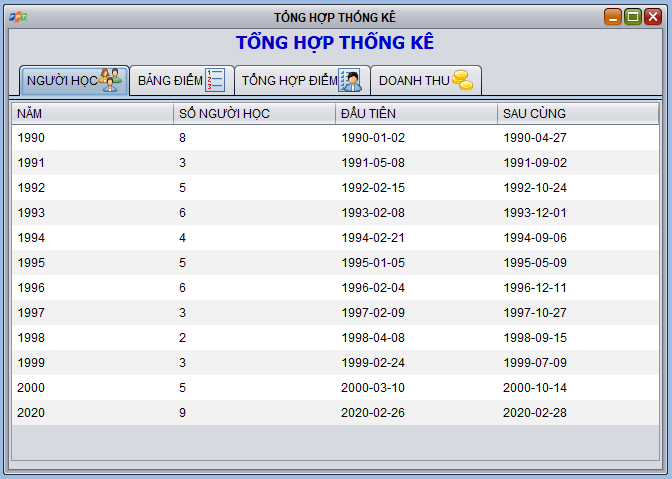
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** | |
|  |  |  |  |  | |
|  | 1 | KhoaHocJFrame | JFrame | Title: QUẢN LÝ KHÓA HỌC | |
|  |  |  |  |  | |
|  | 2 | lblTitle | JLabel | Text: QUẢN LÝ KHÓA HỌC | |
|  |  |  |  | Foreground: [0,0,204] | |
|  |  |  |  | Font:Tahoma, Bold, 14 | |
|  |  |  |  |  | |
|  | 3 | tabs | JTabbedPane |  | |
|  |  |  |  |  | |
|  | 4 | pnlEdit | JPanel | Tab Title: CẬP NHẬT | |
|  |  |  |  |  | |
|  | 5 | pnlList | JPanel | Tab Title: DANH SÁCH | |
|  |  |  |  |  | |
|  | 6 | lblChuyenDe | JLabel | Text: Chuyên đề | |
|  |  |  |  |  | |
|  | 7 | cboChuyenDe | JComboBox | Model: | |
|  |  |  |  | Editable: false | |
|  |  |  |  |  | |
|  | 8 | lblNgayKG | JLabel | Text: Ngày khai giảng |  |
|  |  |  |  |  |  |
|  | 9 | txtNgayKG | JTextField |  |  |
|  |  |  |  |  |  |
|  | 10 | lblHocPhi | JLabel | Text: Học phí |  |
|  |  |  |  |  |  |
|  | 11 | txtHocPhi | JTextField | Editable: false |  |
|  |  |  |  | Enable: false |  |
|  |  |  |  |  |  |
|  | 12 | lblThoiLuong | JLabel | Text: Thời lượng (giờ) |  |
|  |  |  |  |  |  |
|  | 13 | txtThoiLuong | JTextField | Editable: false |  |
|  |  |  |  | Enable: false |  |
|  |  |  |  |  |  |
|  | 14 | lblMaNV | JLabel | Text: Người tạo |  |
|  |  |  |  |  |  |
|  | 15 | txtMaNV | JTextField | Editable: false |  |
|  |  |  |  | Enable: false |  |
|  |  |  |  |  |  |
|  | 16 | lblNgayTao | JLabel | Text: Ngày tạo |  |
|  |  |  |  |  |  |
|  | 17 | txtNgayTao | JLabel | Editable: false |  |
|  |  |  |  | Enable: false |  |
|  |  |  |  |  |  |
|  | 18 | lblGhiChu | JLabel | Text: Ghi chú |  |
|  |  |  |  |  |  |
|  | 19 | txtGhiChu | JTextArea |  |  |
|  |  |  |  |  |  |
|  | 20 | btnInsert | JButton | Text:Thêm |  |
|  |  |  |  |  |  |
|  | 21 | btnUpdate | JButton | Text:Sửa |  |
|  |  |  |  |  |  |
|  | 22 | btnDelete | JButton | Text:Xóa |  |
|  |  |  |  |  |  |
|  | 23 | btnClear | JButton | Text:Mới |  |
|  |  |  |  |  |  |
|  | 24 | btnFirst | JButton | Text: |< |  |
|  |  |  |  |  |  |
|  | 25 | btnPrev | JButton | Text: << |  |
|  |  |  |  |  |  |
|  | 26 | btnNext | JButton | Text: >> |  |
|  |  |  |  |  |  |
|  | 27 | btnLast | JButton | Text: >| |  |
|  |  |  |  |  |  |
|  | 28 | tblGridView | JTable | Model: như hình |  |
|  |  |  |  | Height: 25 |  |
|  |  |  |  |  |  |
| 29 | | btnHocVien | JButton | Text: Học viên | |
|  | |  |  |  | |

#### Quản lý học viên

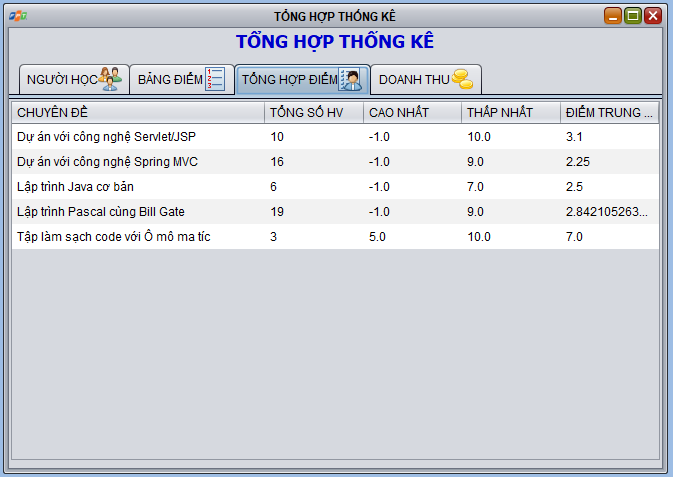


|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | HocVienJFrame | JFrame | Title: QUẢN LÝ HỌC VIÊN KHÓA HỌC |
|  |  |  |  | defaultCloseOperation:DISPONSE |
|  |  |  |  |  |
|  | 2 | pnlHVKhac | JPanel | Border: TitledBorder(Title: HỌC VIÊN KHÁC) |
|  |  |  |  |  |
|  | 3 | pnlHVKH | JPanel | Border: TitledBorder(Title: HỌC VIÊN CỦA KHÓA HỌC) |
|  |  |  |  |  |
|  | 4 | cboNguoiHoc | JComboBox | Model: |
|  |  |  |  | Editable: false |
|  |  |  |  |  |
|  | 5 | txtDiem | JTextField |  |
|  |  |  |  |  |
| 6 | | btnThem | JButton | Text: Thêm |
|  | |  |  |  |
| 7 | | tblGridView | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  |  |
| 8 | | bgrPhanLoai | ButtonGroup |  |
|  | |  |  |  |
| 9 | | rdoTatCa | JButtonGroup | Text: Tất cả |
|  | |  |  | Selected: true |
|  | |  |  | ButtonGroup:bgrPhanLoai |
|  | |  |  |  |
| 10 | | rdoDaNhap | JButtonGroup | Text: Đã nhập điểm |
|  | |  |  | ButtonGroup:bgrPhanLoai |
|  | |  |  |  |
| 11 | | rdoChuaNhap | JButtonGroup | Text: Chưa nhập điểm |
|  | |  |  | ButtonGroup:bgrPhanLoai |
|  | |  |  |  |
| 12 | | btnCapNhat | JButton | Text: Cập nhật |
|  | |  |  |  |

#### Tổng hợp thống kê











Các thành phần giao diện:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | ThongKeJFrame | JFrame | Title: TỔNG HỢP – THỐNG KÊ |
|  |  |  |  | defaultCloseOperation:DISPONSE |
|  |  |  |  |  |
|  | 2 | lblTitle | JLabel | Text: TỔNG HỢP – THỐNG KÊ |
|  |  |  |  |  |
|  | 3 | tabs | JTabbedPane |  |
|  |  |  |  |  |
|  | 4 | pnlNguoiHoc | JPanel | Tab Title: NGƯỜI HỌC |
|  |  |  |  | Layout: BorderLayout |
|  |  |  |  |  |
|  | 5 | tblNguoiHoc | JTable | Model: như hình |
|  |  |  |  | Height: 25 |
|  |  |  |  |  |
|  | 6 | pnlBangDiem | JPanel | Tab Title: BẢNG ĐIỂM |
|  |  |  |  | Layout: BorderLayout |
|  |  |  |  |  |
| 7 | | pnlKhoaHoc | JPanel | Layout: BorderLayout |
|  | |  |  | Layout.Direction: first |
|  | |  |  |  |
| 8 | | lblKhoaHoc | JLabel | Text: KHÓA HỌC |
|  | |  |  | Layout.Direction: before |
|  | |  |  |  |
| 9 | | cboKhoaHoc | JComboBox | Model: |
|  | |  |  | Editable: false |
|  | |  |  | Layout.Direction: center |
|  | |  |  |  |
| 10 | | tblBangDiem | JTable | Layout.Direction: center |
|  | |  |  | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  |  |
| 11 | | pnlKhoaHoc | JPanel | Tab Title: TỔNG HỢP |
|  | |  |  | Layout: BorderLayout |
|  | |  |  |  |
| 12 | | tblTongHop | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  | Layout.Direction: center |
|  | |  |  |  |
| 13 | | pnlDoanhThu | JPanel | Tab Title: DOANH THU |
|  | |  |  | Layout: BorderLayout |
|  | |  |  |  |
| 14 | | pnlName | JPanel | Layout: BorderLayout |
|  | |  |  | Layout.Direction: first |
|  | |  |  |  |
| 15 | | lblNam | JLabel | Text: NĂM |
|  | |  |  | Layout.Direction: before |
|  | |  |  |  |
| 16 | | cboNam | JComboBox | Model: |
|  | |  |  | Editable: false |
|  | |  |  | Layout.Direction: center |
|  | |  |  |  |
| 17 | | tblDoanhThu | JTable | Model: như hình |
|  | |  |  | Height: 25 |
|  | |  |  | Layout.Direction: center |
|  | |  |  |  |

#### Giới thiệu



Các thành phần giao diện:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Component** | **Kiểu** | **Thuộc tính** |
|  |  |  |  |  |
|  | 1 | GioiThieuJDialog | JDialog | Undecorated: True |
|  |  |  |  | preferredSize: [512, 505] |
|  |  |  |  | alwaysOnTop: True |
|  |  |  |  | Layout: BorderLayout |
|  |  |  |  |  |
|  | 2 | lblTitle | JLabel | Icon: logo-small.png |
|  |  |  |  | Layout.Direction: first |
|  |  |  |  |  |
|  | 3 | txtGioiThieu | JTextPane | Text: như hình |
|  |  |  |  | Layout.Direction: center  Editable: false |
|  |  |  |  |  |

#### Hướng dẫn sử dụng



Thiết kế giao diện:

Thiết kế các trang web hướng dẫn sử dụng phần mềm. Trong phần viết mã sẽ hướng dẫn cách mở các trang web này khi người dùng nhấp vào mục hướng dẫn trong menu trợ giúp.

# Thực hiện viết mã

## Viết mã tạo CSDL

### Tạo CSDL

CREATE DATABASE Polypro

GO

USE Edusys

GO

CREATE TABLE NhanVien(

MaNV nvarchar(50) NOT NULL,

MatKhau nvarchar(50) NOT NULL,

HoTen nvarchar(50) NOT NULL,

VaiTro bit NOT NULL DEFAULT 0,

PRIMARY KEY(MaNV)

)

GO

CREATE TABLE ChuyenDe(

MaCD nchar(5) NOT NULL,

TenCD nvarchar(50) NOT NULL,

HocPhi float NOT NULL DEFAULT 0,

ThoiLuong int NOT NULL DEFAULT 30,

Hinh nvarchar(50) NOT NULL DEFAULT 'chuyende.png', MoTa nvarchar(255) NOT NULL,

PRIMARY KEY(MaCD),

UNIQUE(TenCD),

CHECK(HocPhi >= 0 AND ThoiLuong > 0)

)

GO

CREATE TABLE NguoiHoc(

MaNH nchar(7) NOT NULL,

HoTen nvarchar(50) NOT NULL,

NgaySinh date NOT NULL,

GioiTinh bit NOT NULL DEFAULT 0,

DienThoai nvarchar(50) NOT NULL,

Email nvarchar(50) NOT NULL,

GhiChu nvarchar(max) NULL,

MaNV nvarchar(50) NOT NULL,

NgayDK date NOT NULL DEFAULT getdate(),

PRIMARY KEY(MaNH)

)

GO

CREATE TABLE KhoaHoc(

MaKH int IDENTITY(1,1) NOT NULL,

MaCD nchar(5) NOT NULL,

HocPhi float NOT NULL DEFAULT 0,

ThoiLuong int NOT NULL DEFAULT 0,

NgayKG date NOT NULL,

GhiChu nvarchar(50) NULL,

MaNV nvarchar(50) NOT NULL,

NgayTao date NOT NULL DEFAULT getdate(),

PRIMARY KEY(MaKH),

CHECK(HocPhi >= 0 AND ThoiLuong > 0),

FOREIGN KEY (MaCD) REFERENCES ChuyenDe(MaCD) ON UPDATE CASCADE, FOREIGN KEY (MaNV) REFERENCES NhanVien(MaNV) ON UPDATE CASCADE

)

GO

CREATE TABLE HocVien(

MaHV int IDENTITY(1,1) NOT NULL,

MaKH int NOT NULL,

MaNH nchar(7) NOT NULL,

Diem float NOT NULL,

PRIMARY KEY(MaHV),

UNIQUE(MaKH, MaNH),

FOREIGN KEY (MaKH) REFERENCES KhoaHoc(MaKH) ON DELETE CASCADE, FOREIGN KEY (MaNH) REFERENCES NguoiHoc(MaNH) ON UPDATE CASCADE

)

### SQL truy vấn và thao tác

#### NhanVien

|  |  |
| --- | --- |
|  |  |
| **Hành động** | **Câu lệnh SQL** |
|  |  |
| Thêm mới | INSERT INTO NhanVien (MaNV, MatKhau, HoTen, VaiTro) VALUES (?, ?, ?, ?) |
|  |  |
| Cập nhật theo mã | UPDATE NhanVien SET MatKhau=?, HoTen=?, VaiTro=? WHERE MaNV=? |
|  |  |
| Xóa theo mã | DELETE FROM NhanVien WHERE MaNV=? |
|  |  |
| Truy vấn tất cả | SELECT \* FROM NhanVien |
|  |  |
| Truy vấn theo mã | SELECT \* FROM NhanVien WHERE MaNV=? |
|  |  |

#### ChuyenDe

|  |  |
| --- | --- |
| **Hành động** | **Câu lệnh SQL** |
|  |  |
| Thêm mới | INSERT INTO ChuyenDe (MaCD, TenCD, HocPhi, ThoiLuong, Hinh, MoTa) VALUES (?, ?, |
|  | ?, ?, ?, ?) |
|  |  |
| Cập nhật theo mã | UPDATE ChuyenDe SET TenCD=?, HocPhi=?, ThoiLuong=?, Hinh=?, MoTa=? WHERE MaCD=? |
|  |  |
| Xóa theo mã | DELETE FROM ChuyenDe WHERE MaCD=? |
|  |  |
| Truy vấn tất cả | SELECT \* FROM ChuyenDe |
|  |  |
| Truy vấn theo mã | SELECT \* FROM ChuyenDe WHERE MaCD=? |
|  |  |

#### NguoiHoc

|  |  |  |
| --- | --- | --- |
| **Hành động** | **Câu lệnh SQL** | |
|  |  |  |
| Thêm mới | INSERT | INTO NguoiHoc (MaNH, HoTen, NgaySinh, GioiTinh, DienThoai, Email, |
|  | GhiChu, MaNV, NgayDK) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?) | |
|  |  |  |
| Cập nhật theo mã | UPDATE | NguoiHoc SET HoTen=?, NgaySinh=?, GioiTinh=?, DienThoai=?, Email=?, |
|  | GhiChu=?, MaNV=?, NgayDK=? WHERE MaNH=? | |
|  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Xóa theo mã | |  | DELETE FROM NguoiHoc WHERE MaNH=? |  |
|  |  | |  |  |  |
|  | Truy vấn tất cả | |  | SELECT \* FROM NguoiHoc |  |
|  |  | |  |  |  |
|  | Truy vấn theo mã | |  | SELECT \* FROM NguoiHoc WHERE MaNH=? |  |
|  |  |  |  |  |  |

#### KhoaHoc

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | |  |  |  |
|  | **Hành động** | |  | **Câu lệnh SQL** |  |
|  |  | |  |  | |
|  | Thêm mới | |  | INSERT INTO KhoaHoc (MaCD, HocPhi, ThoiLuong, NgayKG, GhiChu, MaNV, NgayTao) | |
|  |  |  |  | VALUES (?, ?, ?, ?, ?, ?, ?) |  |
|  |  | |  |  | |
|  | Cập nhật theo mã | |  | UPDATE KhoaHoc SET MaCD=?, HocPhi=?, ThoiLuong=?, NgayKG=?, GhiChu=?, MaNV=?, | |
|  |  |  |  | NgayTao=? WHERE MaKH=? |  |
|  |  | |  |  |  |
|  | Xóa theo mã | |  | DELETE FROM KhoaHoc WHERE MaKH=? |  |
|  |  | |  |  |  |
|  | Truy vấn tất cả | |  | SELECT \* FROM KhoaHoc |  |
|  |  | |  |  |  |
|  | Truy vấn theo mã | |  | SELECT \* FROM KhoaHoc WHERE MaKH=? |  |
|  |  |  |  |  |  |

#### HocVien

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | **Hành động** |  | **Câu lệnh SQL** |  |
|  |  |  |  |  |
|  | Thêm mới |  | INSERT INTO HocVien (MaKH, MaNH, Diem) VALUES (?, ?, ?) |  |
|  |  |  |  |  |
|  | Cập nhật theo mã |  | UPDATE HocVien SET MaKH=?, MaNH=?, Diem=? WHERE MaHV=? |  |
|  |  |  |  |  |
|  | Xóa theo mã |  | DELETE FROM HocVien WHERE MaHV=? |  |
|  |  |  |  |  |
|  | Truy vấn tất cả |  | SELECT \* FROM HocVien |  |
|  |  |  |  |  |
|  | Truy vấn theo mã |  | SELECT \* FROM HocVien WHERE MaHV=? |  |
|  |  |  | |  |

### Các thủ tục lưu tổng hợp thống kê

#### Số người học qua từng năm

|  |  |  |  |
| --- | --- | --- | --- |
| CREATE PROC sp\_ThongKeNguoiHoc | | | |
| AS BEGIN | |  |  |
|  | SELECT |  |  |
|  | YEAR(NgayDK) Nam, | | |
|  | COUNT(\*) SoLuong, | | |
|  | MIN(NgayDK) DauTien, | | |
|  | MAX(NgayDK) CuoiCung | | |
|  | FROM NguoiHoc | | |
|  | GROUP BY YEAR(NgayDK) | | |
| END |  |  |  |

#### Doanh thu theo chuyên đề

CREATE PROC sp\_ThongKeDoanhThu(@Year INT)

AS BEGIN

SELECT

TenCD ChuyenDe,

COUNT(DISTINCT kh.MaKH) SoKH,

COUNT(hv.MaHV) SoHV,

SUM(kh.HocPhi) DoanhThu,

MIN(kh.HocPhi) ThapNhat,

MAX(kh.HocPhi) CaoNhat,

AVG(kh.HocPhi) TrungBinh

FROM KhoaHoc kh

JOIN HocVien hv ON kh.MaKH=hv.MaKH

JOIN ChuyenDe cd ON cd.MaCD=kh.MaCD

WHERE YEAR(NgayKG) = @Year

GROUP BY TenCD

END

#### Học viên theo chuyên đề

CREATE PROC sp\_ThongKeDiem

AS BEGIN

SELECT

TenCD ChuyenDe,

COUNT(MaHV) SoHV,

MIN(Diem) ThapNhat,

MAX(Diem) CaoNhat,

AVG(Diem) TrungBinh

FROM KhoaHoc kh

JOIN HocVien hv ON kh.MaKH=hv.MaKH

JOIN ChuyenDe cd ON cd.MaCD=kh.MaCD

GROUP BY TenCD

END

#### Bảng điểm

CREATE PROC sp\_BangDiem(@MaKH INT)

AS BEGIN

SELECT

nh.MaNH,

nh.HoTen,

hv.Diem

FROM HocVien hv

JOIN NguoiHoc nh ON nh.MaNH=hv.MaNH

WHERE hv.MaKH = @MaKH

ORDER BY hv.Diem DESC

END

## Lập trình JDBC và các lớp hỗ trợ

Trong dự án này chúng ta chọn mô hình lập trình với CSDL như hình sau:



Trong đó:

⎫ UI: các thành phần giao diện

⎫ DB: cơ sở dữ liệu

⎫ Jdbc: là lớp tiện ích cung cấp các hàm tiện ích làm việc với CSDL thông qua câu lệnh sql hoặc lời gọi thủ tục lưu

⎫ Model: là các lớp mô tả dữ liệu theo cấu trúc các bảng trong CSDL

⎫ DAO: là các lớp thao tác và truy vấn dữ liệu. Nó có nhiệm vụ chuyển đổi Model sang SQL và ngược lại.

Theo mô hình này thì lập trình giao diện làm việc với CSDL thông qua các lớp model. Model như là một gói dữ liệu được sử dụng để giao tiếp giữa giao diện và các thành phần DAO. DAO mới thực sự là lập trình CSDL nó làm nhiệm vụ nhận model từ giao diện, chuyển đổi sang SQL và sử dụng Jdbc để thực hiện các thao tác dữ liệu. Ngược lại thì truy vấn dữ liệu từ CSDL, chuyển đổi thành model để hiển thị lên các thành phần giao diện.

*Việc tách thành các phần riêng biết giúp quản lý dự án tốt hơn, dễ bảo trì nâng cấp hơn.*

### Lớp hỗ trợ

Các lớp hỗ trợ chứa các phương thức tiện ích giúp chúng ta thực hiện các xử lý trong ứng dụng đơn giản hơn, ngắn gọn hơn, dễ hiểu hơn. Trong phần này chúng ta xây dựng 2 lớp là XDate và Jdbc như sau.

#### Lớp tiện ích dateHelper

Chứa các phương thức chuyển đổi thời gian và chuỗi giúp lập trình sau này khi xử lý thời gian dễ dàng hơn.

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package helper;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.Calendar;

import java.util.Date;

/\*\*

\*

\* **@author** Jason

\*/

public class **dateHelper** {

public static final SimpleDateFormat *DATE\_FORMATER* = new SimpleDateFormat("dd/MM/yyyy");

//chuuyển String sang Date

/\*

@param date truyền vào date kiểu String

@param pattern truyền vào kiểu

return trả về date kiểu Date

\*/

public static Date ***toDate***(String date,String...pattern){

try {

if(pattern.length>0)*DATE\_FORMATER*.applyPattern(pattern[0]);

if(date==null)return dateHelper.*now*();

return *DATE\_FORMATER*.parse(date);

} catch (ParseException ex) {

throw new RuntimeException(ex);

}

}

//chuyển Date sang String

/\*

@param date chuyền vào date kiểu date

@param pattern định dạng date

return date kiểu String đã định theo dạng pattern

\*/

public static String ***toString***(Date date, String...pattern){

if(pattern.length>0)*DATE\_FORMATER*.applyPattern(pattern[0]);

if(date==null)date=dateHelper.*now*();

return *DATE\_FORMATER*.format(date);

}

//lấy giờ hiện tại

public static Date ***now***() {

return new Date(); //new Date lấy giờ hiện tại

}

//thêm 1 số ngày vào mốc thời gian

/\*

@param date kiểu Date

@param days số ngày thêm, kiểu int

return date kiểu Date đã thêm số ngày

\*/

public static Date ***addDays***(Date date, int days){

//date.setTime(date.getTime()+days\*24\*60\*60\*1000);

//setTime gán cho biến date 1 mốc thời gian được chuyển từ milisecon (long)

//getTime chuyển mốc thời gian của biến date thành milisecon (long)

Calendar cal=Calendar.*getInstance*();

cal.setTime(date);

cal.add(Calendar.*DAY\_OF\_MONTH*,days);

return cal.getTime();

}

//thêm 1 số ngày vào mốc thời gian hiện tại

/\*

@param days số ngày thêm, kiểu int

return date kiểu Date đã thêm số ngày vào date hiện tại

\*/

public static Date ***add***(int days){

Calendar cal=Calendar.*getInstance*();

cal.add(Calendar.*DAY\_OF\_MONTH*,days);

return cal.getTime();

}

}

#### Lớp tiện ích jdbcHelper

Chứa các phương thức thực hiện các câu lệnh thao tác và truy vấn được gọn nhẹ hơn.

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package helper;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.logging.Level;

import java.util.logging.Logger;

/\*\*

\*

\* **@author** Jason

\*/

public class **jdbcHelper** {

public static String *driver*="com.microsoft.sqlserver.jdbc.SQLServerDriver";

public static String *dburl*="jdbc:sqlserver://localhost:1433;databaseName=Polypro";

public static String *username*="sa";

public static String *password*="songlong";

//nạp driver

static{

try {

Class.*forName*(*driver*);

} catch (ClassNotFoundException ex) {

Logger.*getLogger*(jdbcHelper.class.getName()).log(Level.*SEVERE*, null, ex);

}

}

//nạp truyền giá trị đối số vào prepareStatement

//có thể statment không có đối số

//prepareStatement có thể là prepareStatement hoặc CallableStatements

/\*

@param sql câu lệnh sql statement (có đối số hoặc không)

@param args mảng đối số của câu lệnh sql (có hoặc không)

return pstmt là PrepareStatement đã được truyền đối số

\*/

public static PreparedStatement ***preparedStatement***(String sql,Object...args)throws SQLException{

Connection con=DriverManager.*getConnection*(*dburl*, *username*, *password*);

PreparedStatement pstmt=null;

if(sql.startsWith("{"))pstmt=con.prepareCall(sql); //có thể gán biến kiểu PreparedStatement là prepareCall (CallableStatement)

else pstmt=con.prepareStatement(sql);

for(int i=0; i<args.length;i++){

pstmt.setObject(i+1, args[i]);

}

return pstmt;

}

//thao tác (INSERT, UPDATE, DELETE)

//thực thi prepareStatement (đã được truyền đối số ở hàm trên)

//khi làm chỉ cần gọi hàm này, từ trong hàm này nó sẽ gọi ra hàm preparedStatement ở trên

/\*

@param sql (String) câu lệnh sql statement (có đối số hoặc không)

@param args mảng đối số của câu lệnh sql (có hoặc không)

\*/

public static void ***executeUpdate***(String sql,Object...args){

try {

PreparedStatement pstmt= *preparedStatement*(sql, args);

try{

pstmt.executeUpdate();

}finally{

pstmt.getConnection().close(); //đóng Connection từ statement

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

}

}

//thao tác truy vấn (SELECT)

//thực thi prepareStatement (đã được truyền đối số ở hàm trên)

//khi làm chỉ cần gọi hàm này, từ trong hàm này nó sẽ gọi ra hàm preparedStatement ở trên

/\*

@param sql (String) câu lệnh sql statement (có đối số hoặc không)

@param args mảng đối số của câu lệnh sql (có hoặc không)

\*/

public static ResultSet ***executeQuery***(String sql, Object...args){

try {

PreparedStatement pstmt=*preparedStatement*(sql, args);

try{

return pstmt.executeQuery();

}finally{

//pstmt.getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException(ex); //throw các lỗi khi chạy CT, VD không có return khi try bị lỗi

}

}

}

#### Lớp tiện ích dialogHelper

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package helper;

import java.awt.Component;

import javax.swing.JOptionPane;

/\*\*

\*

\* **@author** Jason

\*/

public class **dialogHelper** {

/\*\*

\* **Hiển** **thị** **thông** **báo** **cho** **người** **dùng**

\* **@param** parent là cửa sổ chứa thông báo

\* **@param** message là thông báo

\*/

public static void ***alert***(Component parent, String message) {

JOptionPane.*showMessageDialog*(parent, message,

"Hệ thống quản lý đào tạo", JOptionPane.*INFORMATION\_MESSAGE*);

}

/\*\*

\* **Hiển** **thị** **thông** **báo** **và** **yêu** **cầu** **người** **dùng** **xác** **nhận**

\* **@param** parent là cửa sổ chứa thông báo

\* **@param** message là câu hỏi yes/no

\* **@return** là kết quả nhận được true/false

\*/

public static boolean ***confirm***(Component parent, String message) {

int result = JOptionPane.*showConfirmDialog*(parent, message,

"Hệ thống quản lý đào tạo",

JOptionPane.*YES\_NO\_OPTION*, JOptionPane.*QUESTION\_MESSAGE*);

return result == JOptionPane.*YES\_OPTION*;

}

/\*\*

\* **Hiển** **thị** **thông** **báo** **yêu** **cầu** **nhập** **dữ** **liệu**

\* **@param** parent là cửa sổ chứa thông báo

\* **@param** message là thông báo nhắc nhở nhập

\* **@return** là kết quả nhận được từ người sử dụng nhập vào

\*/

public static String ***prompt***(Component parent, String message) {

return JOptionPane.*showInputDialog*(parent, message,

"Hệ thống quản lý đào tạo", JOptionPane.*INFORMATION\_MESSAGE*);

}

}

#### Lớp tiện ích shareHelper

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package helper;

import java.awt.Image;

import java.io.File;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.nio.file.StandardCopyOption;

import javax.swing.ImageIcon;

import model.nhanVien;

/\*\*

\*

\* **@author** Jason

\*/

public class **shareHelper** {

/\*\*

\* Ảnh biểu tượng của ứng dụng, xuất hiện trên mọi cửa sổ

\*/

public static final Image *APP\_ICON*;

public static final ImageIcon *APP\_ICON\_1*;

static{

// Tải biểu tượng ứng dụng

//CÁCH TẢI ẢNH TỪ TRONG PROJECT

//icon là thư mục con của src

String file = "/icon/fpt.png"; //icon là thư mục con của src

*APP\_ICON* = new ImageIcon(shareHelper.class.getResource(file)).getImage();

*APP\_ICON\_1* = new ImageIcon(shareHelper.class.getResource(file));

}

/\*\*

\* **Sao** **chép** **file** **logo** **chuyên** **đề** **vào** **thư** **mục** **logos (tạo** **nếu** **chưa** **có** **thư** **mục** **logos**)

\* **@param** file là đối tượng file ảnh

\* **@return** chép được hay không

\*/

public static boolean ***saveLogo***(File file){

File dir = new File("logos"); //khai báo thư mục logos ngang hàng với src

// Tạo thư mục nếu chưa tồn tại

if(!dir.exists()){

dir.mkdirs();

}

File newFile = new File(dir, file.getName());

try {

// Copy vào thư mục logos (đè nếu đã tồn tại)

Path source = Paths.*get*(file.getAbsolutePath());

Path destination = Paths.*get*(newFile.getAbsolutePath());

Files.*copy*(source, destination, StandardCopyOption.*REPLACE\_EXISTING*);

return true;

}

catch (Exception ex) {

return false;

}

}

/\*\*

\* **Đọc** **hình** **ảnh** **logo** **chuyên** **đề** **trong** **thư** **mục** **logos** **theo** **tenFile**

\* **@param** fileName là tên file logo

\* **@return** ImageIcon ảnh đọc được

\*/

public static ImageIcon ***readLogo***(String fileName){

File path = new File("logos", fileName);

return new ImageIcon(new ImageIcon(path.getAbsolutePath()).getImage().getScaledInstance(180, 180, Image.*SCALE\_DEFAULT*));

}

/\*\*

\* Đối tượng này chứa thông tin người sử dụng sau khi đăng nhnập

\*/

public static nhanVien *USER* = null;

/\*\*

\* Xóa thông tin của người sử dụng khi có yêu cầu đăng xuất

\*/

public static void ***logoff***() {

shareHelper.*USER* = null;

}

/\*\*

\* **Kiểm** **tra** **xem** **đăng** **nhập** **hay** **chưa**

\* **@return** đăng nhập hay chưa

\*/

public static boolean ***authenticated***() {

return shareHelper.*USER* != null;

}

}

#### Lớp tiện ích utilityHelper

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package helper;

import static java.awt.Color.pink;

import static java.awt.Color.white;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import javax.swing.JPasswordField;

import javax.swing.JTextArea;

import javax.swing.JTextField;

/\*\*

\*

\* **@author** Jason

\*/

public class **utilityHelper** {

public static String ***getRank***(double diem) {

String xepLoai = "Xuất sắc";

if (diem < 0) {

xepLoai = "Chưa nhập";

} else if (diem < 3) {

xepLoai = "Kém";

} else if (diem < 5) {

xepLoai = "Yếu";

} else if (diem < 6.5) {

xepLoai = "Trung bình";

} else if (diem < 7.5) {

xepLoai = "Khá";

} else if (diem < 9) {

xepLoai = "Giỏi";

}

return xepLoai;

}

/\*

1-15 kí tự

a-z, A-Z, 0-9

\*/

public static boolean ***checkMaNV***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "[a-zA-Z0-9]{1,15}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải có 1-15 kí tự**\n**chữ hoa, thường không dấu hoặc số.");

return false;

}

}

/\*

đúng 7 kí tự

a-z, A-Z, 0-9

\*/

public static boolean ***checkMaNH***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "[a-zA-Z0-9]{7}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải có đúng 7 kí tự**\n**chữ thường, chữ hoa hoặc số");

return false;

}

}

/\*

đúng 5 kí tự

a-z, A-Z, 0-9

\*/

public static boolean ***checkMaCD***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "[a-zA-Z0-9]{5}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải có đúng 5 kí tự**\n**chữ thường, chữ hoa hoặc số");

return false;

}

}

//pass từ 3-16 kí tự

public static boolean ***checkPass***(JPasswordField txt) {

txt.setBackground(*white*);

if (txt.getPassword().length > 2 && txt.getPassword().length < 17) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải có từ 3-16 kí tự.");

return false;

}

}

public static boolean ***isValidDate***(String inDate) {

if (inDate == null) {

return false;

}

//set the format to use as a constructor argument

SimpleDateFormat dateFormat = new SimpleDateFormat("dd/MM/yyyy");

if (inDate.trim().length() != dateFormat.toPattern().length()) {

return false;

}

dateFormat.setLenient(false);

try {

//parse the inDate parameter

dateFormat.parse(inDate.trim());

} catch (ParseException pe) {

return false;

}

return true;

}

//định dạng dd/MM/yyyy (ko dc)(hoặc d/M/yyyy nếu là số 0 đứng trước)

public static boolean ***checkDate***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

// String rgx = "^(?:(?:31(\\/|-|\\.)(?:0?[13578]|1[02]))\\1|(?:(?:29|30)(\\/|-|\\.)(?:0?[1,3-9]|1[0-2])\\2))(?:(?:1[6-9]|[2-9]\\d)?\\d{2})$|^(?:29(\\/|-|\\.)0?2\\3(?:(?:(?:1[6-9]|[2-9]\\d)?(?:0[48]|[2468][048]|[13579][26])|(?:(?:16|[2468][048]|[3579][26])00))))$|^(?:0?[1-9]|1\\d|2[0-8])(\\/|-|\\.)(?:(?:0?[1-9])|(?:1[0-2]))\\4(?:(?:1[6-9]|[2-9]\\d)?\\d{2})$";

// if (id.matches(rgx)) {

// return true;

// } else {

// txt.setBackground(pink);

// dialogHelper.alert(txt.getRootPane(), txt.getName() + " không đúng định dạng Date.");

// return false;

// }

if (*isValidDate*(id)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " không đúng định dạng dd/MM/yyyy");

return false;

}

}

//gồm các ký tự chữ đấu cách

//từ 3-25 kí tự

public static boolean ***checkName***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "^[A-Za-zÀÁÂÃÈÉÊÌÍÒÓÔÕÙÚÝàáâãèéêìíòóôõùúýĂăĐđĨĩŨũƠơƯưẠ-ỹ ]{3,25}$";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là tên tiếng việt hoặc không đấu**\n**từ 3-25 kí tự");

return false;

}

}

//bất kì kí tự nào

//từ 3-50 kí tự

public static boolean ***checkTenCD***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = ".{3,50}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải từ 3-50 kí tự.");

return false;

}

}

//bất kì kí tự nào

//từ 3-255 kí tự

public static boolean ***checkMoTaCD***(JTextArea txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = ".{3,255}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải từ 3-255 kí tự.");

return false;

}

}

//gồm 10 số

//các đầu 3 số của nhà mạng

public static boolean ***checkSDT***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "(086|096|097|098|032|033|034|035|036|037|038|039|089|090|093|070|079|077|078|076|088|091|094|083|084|085|081|082|092|056|058|099|059)[0-9]{7}";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải gồm 10 số**\n**đúng các đầu số của nhà mạng.");

return false;

}

}

public static boolean ***checkEmail***(JTextField txt) {

txt.setBackground(*white*);

String id = txt.getText();

String rgx = "^[a-zA-Z][a-zA-Z0-9\_**\\**.]{2,32}@[a-zA-Z0-9]{2,10}(**\\**.[a-zA-Z0-9]{2,4}){1,2}$";

if (id.matches(rgx)) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " không đúng định dạng");

return false;

}

}

//gio là int >0

public static boolean ***checkThoiLuong***(JTextField txt) {

txt.setBackground(*white*);

try {

int hour = Integer.*parseInt*(txt.getText());

if (hour >= 0) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải lớn hơn bằng 0.");

return false;

}

} catch (NumberFormatException e) {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là số nguyên.");

return false;

}

}

//học phí là float >0

public static boolean ***checkHocPhi***(JTextField txt) {

txt.setBackground(*white*);

try {

float hp = Float.*parseFloat*(txt.getText());

if (hp >= 0) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là lớn hơn bằng 0.");

return false;

}

} catch (NumberFormatException e) {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là số thực.");

return false;

}

}

public static boolean ***checkDiem***(JTextField txt) {

txt.setBackground(*white*);

try {

float hp = Float.*parseFloat*(txt.getText());

if ((hp >= 0 && hp <= 10)||hp==-1) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là trong khoảng 0-10");

return false;

}

} catch (NumberFormatException e) {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), txt.getName() + " phải là số thực.");

return false;

}

}

public static boolean ***checkNullText***(JTextField txt) {

txt.setBackground(*white*);

if (txt.getText().trim().length() > 0) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), "Không được để trống " + txt.getName());

return false;

}

}

public static boolean ***checkNullText***(JTextArea txt) {

txt.setBackground(*white*);

if (txt.getText().trim().length() > 0) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), "Không được để trống " + txt.getName());

return false;

}

}

public static boolean ***checkNullPass***(JPasswordField txt) {

txt.setBackground(*white*);

if (txt.getPassword().length > 0) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(txt.getRootPane(), "Không được để trống " + txt.getName());

return false;

}

}

}

### Model class - Các lớp mô tả dữ liệu

#### nhanVien

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

/\*\*

\*

\* **@author** Jason

\*/

public class **nhanVien** {

private String maNV;

private String matKhau;

private String hoTen;

private boolean vaiTro=false;

@Override

public String **toString**(){

return this.hoTen;

}

public String **getMaNV**() {

return maNV;

}

public void **setMaNV**(String maNV) {

this.maNV = maNV;

}

public String **getMatKhau**() {

return matKhau;

}

public void **setMatKhau**(String matKhau) {

this.matKhau = matKhau;

}

public String **getHoTen**() {

return hoTen;

}

public void **setHoTen**(String hoTen) {

this.hoTen = hoTen;

}

public boolean **isVaiTro**() {

return vaiTro;

}

public void **setVaiTro**(boolean vaiTro) {

this.vaiTro = vaiTro;

}

}

#### chuyenDe

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

/\*\*

\*

\* **@author** Jason

\*/

public class **chuyenDe** {

private String maCD;

private String tenCD;

private double hocPhi;

private int thoiLuong;

private String hinh;

private String moTa;

@Override

public String **toString**(){

return this.tenCD;

}

public String **getMaCD**() {

return maCD;

}

public void **setMaCD**(String maCD) {

this.maCD = maCD;

}

public String **getTenCD**() {

return tenCD;

}

public void **setTenCD**(String tenCD) {

this.tenCD = tenCD;

}

public double **getHocPhi**() {

return hocPhi;

}

public void **setHocPhi**(double hocPhi) {

this.hocPhi = hocPhi;

}

public int **getThoiLuong**() {

return thoiLuong;

}

public void **setThoiLuong**(int thoiLuong) {

this.thoiLuong = thoiLuong;

}

public String **getHinh**() {

return hinh;

}

public void **setHinh**(String hinh) {

this.hinh = hinh;

}

public String **getMoTa**() {

return moTa;

}

public void **setMoTa**(String moTa) {

this.moTa = moTa;

}

}

#### nguoiHoc

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

import helper.dateHelper;

import java.util.Date;

/\*\*

\*

\* **@author** Jason

\*/

public class **nguoiHoc** {

private String maNH;

private String hoTen;

private Date ngaySinh;

private boolean gioiTinh;

private String dienThoai;

private String email;

private String ghiChu;

private String maNV;

private Date ngayDK= dateHelper.*now*();

@Override

public String **toString**(){

return this.hoTen;

}

public String **getMaNH**() {

return maNH;

}

public void **setMaNH**(String maNH) {

this.maNH = maNH;

}

public String **getHoTen**() {

return hoTen;

}

public void **setHoTen**(String hoTen) {

this.hoTen = hoTen;

}

public Date **getNgaySinh**() {

return ngaySinh;

}

public void **setNgaySinh**(Date ngaySinh) {

this.ngaySinh = ngaySinh;

}

public boolean **isGioiTinh**() {

return gioiTinh;

}

public void **setGioiTinh**(boolean gioiTinh) {

this.gioiTinh = gioiTinh;

}

public String **getDienThoai**() {

return dienThoai;

}

public void **setDienThoai**(String dienThoai) {

this.dienThoai = dienThoai;

}

public String **getEmail**() {

return email;

}

public void **setEmail**(String email) {

this.email = email;

}

public String **getGhiChu**() {

return ghiChu;

}

public void **setGhiChu**(String ghiChu) {

this.ghiChu = ghiChu;

}

public String **getMaNV**() {

return maNV;

}

public void **setMaNV**(String maNV) {

this.maNV = maNV;

}

public Date **getNgayDK**() {

return ngayDK;

}

public void **setNgayDK**(Date ngayDK) {

this.ngayDK = ngayDK;

}

}

#### khoaHoc

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

import helper.dateHelper;

import java.util.Date;

/\*\*

\*

\* **@author** Jason

\*/

public class **khoaHoc** {

private int maKH;

private String maCD;

private double hocPhi;

private int thoiLuong;

private Date ngayKG;

private String ghiChu;

private String maNV;

private Date ngayTao = dateHelper.*now*();

@Override

public String **toString**(){

return this.maCD+" ("+this.ngayKG+")";

}

public int **getMaKH**() {

return maKH;

}

public void **setMaKH**(int maKH) {

this.maKH = maKH;

}

public String **getMaCD**() {

return maCD;

}

public void **setMaCD**(String maCD) {

this.maCD = maCD;

}

public double **getHocPhi**() {

return hocPhi;

}

public void **setHocPhi**(double hocPhi) {

this.hocPhi = hocPhi;

}

public int **getThoiLuong**() {

return thoiLuong;

}

public void **setThoiLuong**(int thoiLuong) {

this.thoiLuong = thoiLuong;

}

public Date **getNgayKG**() {

return ngayKG;

}

public void **setNgayKG**(Date ngayKG) {

this.ngayKG = ngayKG;

}

public String **getGhiChu**() {

return ghiChu;

}

public void **setGhiChu**(String ghiChu) {

this.ghiChu = ghiChu;

}

public String **getMaNV**() {

return maNV;

}

public void **setMaNV**(String maNV) {

this.maNV = maNV;

}

public Date **getNgayTao**() {

return ngayTao;

}

public void **setNgayTao**(Date ngayTao) {

this.ngayTao = ngayTao;

}

}

#### hocVien

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

/\*\*

\*

\* **@author** Jason

\*/

public class **hocVien** {

private int maHV;

private int maKH;

private String maNH;

private double diem = -1.0;

@Override

public String **toString**(){

return this.toString();

}

public int **getMaHV**() {

return maHV;

}

public void **setMaHV**(int maHV) {

this.maHV = maHV;

}

public int **getMaKH**() {

return maKH;

}

public void **setMaKH**(int maKH) {

this.maKH = maKH;

}

public String **getMaNH**() {

return maNH;

}

public void **setMaNH**(String maNH) {

this.maNH = maNH;

}

public double **getDiem**() {

return diem;

}

public void **setDiem**(double diem) {

this.diem = diem;

}

}

### DAO Class - Các lớp truy xuất dữ liệu

DAO (Data Access Object) là các lớp giúp giúp làm việc với các bảng trong CSDL. Mỗi lớp được định nghĩa một cách đầy đủ nhất về các thao tác và truy vấn dữ liệu sẽ được sử dụng trong ứng dụng. DAO sau đó sẽ được sử dụng trong các điều khiển sự kiện của giao diện để làm việc với CSDL một cách thuận tiện.

Trong phạm vi ứng dụng này chúng ta cần xây dựng mỗi DAO chứa tối thiểu các hành động sau

#### nhanVienDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package dao;

import helper.jdbcHelper;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import model.nhanVien;

/\*\*

\*

\* **@author** Jason

\*/

public class **nhanVienDAO** {

//đọc 1 nhân viên từ 1 bản ghi (1 ResultSet)

public nhanVien **readFromResultSet**(ResultSet rs) throws SQLException{

nhanVien model=new nhanVien();

model.setMaNV(rs.getString("MaNV"));

model.setMatKhau(rs.getString("MatKhau"));

model.setHoTen(rs.getString("HoTen"));

model.setVaiTro(rs.getBoolean("VaiTro"));

return model;

}

//thực hiện truy vấn lấy về 1 tập ResultSet rồi điền tập ResultSet đó vào 1 List

public List<nhanVien> **select**(String sql,Object...args){

List<nhanVien> list=new ArrayList<>();

try {

ResultSet rs=null;

try{

rs=jdbcHelper.*executeQuery*(sql, args);

while(rs.next()){

list.add(readFromResultSet(rs));

}

}finally{

rs.getStatement().getConnection().close(); //đóng kết nối từ resultSet

}

} catch (SQLException ex) {

throw new RuntimeException();

}

return list;

}

/\*\*

\* **Thêm** **mới** **thực** **thể** **vào** **CSDL**

\* **@param** entity là thực thể chứa thông tin bản ghi mới

\*/

public void **insert**(nhanVien entity) {

String sql="INSERT INTO NhanVien (MaNV, MatKhau, HoTen, VaiTro) VALUES (?, ?, ?, ?)";

jdbcHelper.*executeUpdate*(sql,

entity.getMaNV(),

entity.getMatKhau(),

entity.getHoTen(),

entity.isVaiTro());

}

/\*\*

\* **Cập** **nhật** **thực** **thể** **vào** **CSDL**

\* **@param** entity là thực thể chứa thông tin bản ghi cần cập nhật

\*/

public void **update**(nhanVien entity) {

String sql="UPDATE NhanVien SET MatKhau=?, HoTen=?, VaiTro=? WHERE MaNV=?";

jdbcHelper.*executeUpdate*(sql,

entity.getMatKhau(),

entity.getHoTen(),

entity.isVaiTro(),

entity.getMaNV());

}

/\*\*

\* **Xóa** **bản** **ghi** **khỏi** **CSDL**

\* **@param** maNV là mã của bản ghi cần xóa

\*/

public void **delete**(String maNV) {

String sql="DELETE FROM NhanVien WHERE MaNV=?";

jdbcHelper.*executeUpdate*(sql, maNV);

}

/\*\*

\* **Truy** **vấn** **tất** **cả** **các** **các** **thực** **thể**

\* **@return** danh sách các thực thể

\*/

public List<nhanVien> **select**() {

String sql="SELECT \* FROM NhanVien";

return select(sql); //trong 1 class có thể có 2 method trùng tên (nhưng param khác nhau)

}

/\*\*

\* **Truy** **vấn** **thực** **thể** **theo** **mã** **id**

\* **@param** id là mã của bản ghi được truy vấn

\* **@return** thực thể chứa thông tin của bản ghi

\*/

public nhanVien **findById**(String id) {

String sql="SELECT \* FROM NhanVien WHERE MaNV=?";

List<nhanVien> list=select(sql, id);

return list.size()>0?list.get(0):null; //có thể trả về là null

}

}

#### nguoiHocDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package dao;

import helper.jdbcHelper;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import model.nguoiHoc;

/\*\*

\*

\* **@author** Jason

\*/

public class **nguoiHocDAO** {

private nguoiHoc **readFromResultSet**(ResultSet rs) throws SQLException{

nguoiHoc entity=new nguoiHoc();

entity.setMaNH(rs.getString("MaNH"));

entity.setHoTen(rs.getString("HoTen"));

entity.setNgaySinh(rs.getDate("NgaySinh"));

entity.setGioiTinh(rs.getBoolean("GioiTinh"));

entity.setDienThoai(rs.getString("DienThoai"));

entity.setEmail(rs.getString("Email"));

entity.setGhiChu(rs.getString("GhiChu"));

entity.setMaNV(rs.getString("MaNV"));

entity.setNgayDK(rs.getDate("NgayDK"));

return entity;

}

private List<nguoiHoc> **select**(String sql, Object...args){

List<nguoiHoc> list=new ArrayList<>();

try {

ResultSet rs=null;

try{

rs=jdbcHelper.*executeQuery*(sql, args);

while(rs.next()){

list.add(readFromResultSet(rs));

}

}finally{

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException();

}

return list;

}

/\*\*

\* **Thêm** **mới** **thực** **thể** **vào** **CSDL**

\* **@param** entity là thực thể chứa thông tin bản ghi mới

\*/

public void **insert**(nguoiHoc entity) {

String sql="INSERT INTO NguoiHoc (MaNH, HoTen, NgaySinh, GioiTinh, DienThoai, Email, GhiChu, MaNV) VALUES (?, ?, ?, ?, ?, ?, ?, ?)";

jdbcHelper.*executeUpdate*(sql,

entity.getMaNH(),

entity.getHoTen(),

entity.getNgaySinh(),

entity.isGioiTinh(),

entity.getDienThoai(),

entity.getEmail(),

entity.getGhiChu(),

entity.getMaNV());

}

/\*\*

\* **Cập** **nhật** **thực** **thể** **vào** **CSDL**

\* **@param** model là thực thể chứa thông tin bản ghi cần cập nhật

\*/

public void **update**(nguoiHoc model) {

String sql="UPDATE NguoiHoc SET HoTen=?, NgaySinh=?, GioiTinh=?, DienThoai=?, Email=?, GhiChu=?,MaNV=? WHERE MaNH=?";

jdbcHelper.*executeUpdate*(sql,

model.getHoTen(),

model.getNgaySinh(),

model.isGioiTinh(),

model.getDienThoai(),

model.getEmail(),

model.getGhiChu(),

model.getMaNV(),

model.getMaNH());

}

/\*\*

\* **Xóa** **bản** **ghi** **khỏi** **CSDL**

\* **@param** id là mã của bản ghi cần xóa

\*/

public void **delete**(String id) {

String sql="DELETE FROM NguoiHoc WHERE MaNH=?";

jdbcHelper.*executeUpdate*(sql, id);

}

/\*\*

\* **Truy** **vấn** **tất** **cả** **các** **các** **thực** **thể**

\* **@return** list danh sách các thực thể

\*/

public List<nguoiHoc> **select**() {

String sql="SELECT \* FROM NguoiHoc";

return select(sql);

}

/\*\*

\* Truy vấn người học theo keyword

\*/

public List<nguoiHoc> **selectByKeyword**(String keyword) {

String sql="SELECT \* FROM NguoiHoc WHERE HoTen LIKE ?";

return select(sql, "%"+keyword+"%");

}

//truy xuất tất cả học viên không học khóa học maKH

public List<nguoiHoc> **selectByCourse**(Integer makh){ //để là Integer cho đúng kiểu Object

String sql="SELECT \* FROM NguoiHoc WHERE MaNH NOT IN (SELECT MaNH FROM HocVien WHERE MaKH=?)";

return select(sql, makh);

}

//truy xuất người học theo maNH

public nguoiHoc **findById**(String manh){

String sql="SELECT \* FROM NguoiHoc WHERE MaNH=?";

List<nguoiHoc> list = select(sql, manh);

return list.size() > 0 ? list.get(0) : null;

}

}

#### chuyenDeDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package dao;

import helper.jdbcHelper;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import model.chuyenDe;

/\*\*

\*

\* **@author** Jason

\*/

public class **chuyenDeDAO** {

private chuyenDe **readFromResultSet**(ResultSet rs) throws SQLException{

chuyenDe model=new chuyenDe();

model.setMaCD(rs.getString("MaCD"));

model.setHinh(rs.getString("Hinh"));

model.setHocPhi(rs.getDouble("HocPhi"));

model.setMoTa(rs.getString("MoTa"));

model.setTenCD(rs.getString("TenCD"));

model.setThoiLuong(rs.getInt("ThoiLuong"));

return model;

}

private List<chuyenDe> **select**(String sql, Object...args){

List<chuyenDe> list=new ArrayList<>();

try {

ResultSet rs=null;

try{

rs=jdbcHelper.*executeQuery*(sql, args);

while(rs.next()){

list.add(readFromResultSet(rs));

}

}finally{

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException();

}

return list;

}

/\*\*

\* **Thêm** **mới** **thực** **thể** **vào** **CSDL**

\* **@param** entity là thực thể chứa thông tin bản ghi mới

\*/

public void **insert**(chuyenDe entity) {

String sql="INSERT INTO ChuyenDe (MaCD, TenCD, HocPhi, ThoiLuong, Hinh, MoTa) VALUES (?, ?, ?, ?, ?, ?)";

jdbcHelper.*executeUpdate*(sql,

entity.getMaCD(),

entity.getTenCD(),

entity.getHocPhi(),

entity.getThoiLuong(),

entity.getHinh(),

entity.getMoTa());

}

/\*\*

\* **Cập** **nhật** **thực** **thể** **vào** **CSDL**

\* **@param** entity là thực thể chứa thông tin bản ghi cần cập nhật

\*/

public void **update**(chuyenDe entity) {

String sql="UPDATE ChuyenDe SET TenCD=?, HocPhi=?, ThoiLuong=?, Hinh=?, MoTa=? WHERE MaCD=?";

jdbcHelper.*executeUpdate*(sql,

entity.getTenCD(),

entity.getHocPhi(),

entity.getThoiLuong(),

entity.getHinh(),

entity.getMoTa(),

entity.getMaCD());

}

/\*\*

\* **Xóa** **bản** **ghi** **khỏi** **CSDL**

\* **@param** id là mã của bản ghi cần xóa

\*/

public void **delete**(String id) {

String sql="DELETE FROM ChuyenDe WHERE MaCD=?";

jdbcHelper.*executeUpdate*(sql, id);

}

/\*\*

\* **Truy** **vấn** **tất** **cả** **các** **các** **thực** **thể**

\* **@return** list danh sách các thực thể

\*/

public List<chuyenDe> **select**() {

String sql="SELECT \* FROM ChuyenDe";

return select(sql);

}

/\*\*

\* **Truy** **vấn** **thực** **thể** **theo** **mã** **id**

\* **@param** id là mã của bản ghi được truy vấn

\* **@return** thực thể chứa thông tin của bản ghi

\*/

public chuyenDe **findById**(String id) {

String sql="SELECT \* FROM ChuyenDe WHERE MaCD=?";

List<chuyenDe> list=select(sql,id);

return list.size()>0?list.get(0):null;

}

}

#### khoaHocDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package model;

import helper.dateHelper;

import java.util.Date;

/\*\*

\*

\* **@author** Jason

\*/

public class **khoaHoc** {

private int maKH;

private String maCD;

private double hocPhi;

private int thoiLuong;

private Date ngayKG;

private String ghiChu;

private String maNV;

private Date ngayTao = dateHelper.*now*();

@Override

public String **toString**(){

return this.maCD+" ("+this.ngayKG+")";

}

public int **getMaKH**() {

return maKH;

}

public void **setMaKH**(int maKH) {

this.maKH = maKH;

}

public String **getMaCD**() {

return maCD;

}

public void **setMaCD**(String maCD) {

this.maCD = maCD;

}

public double **getHocPhi**() {

return hocPhi;

}

public void **setHocPhi**(double hocPhi) {

this.hocPhi = hocPhi;

}

public int **getThoiLuong**() {

return thoiLuong;

}

public void **setThoiLuong**(int thoiLuong) {

this.thoiLuong = thoiLuong;

}

public Date **getNgayKG**() {

return ngayKG;

}

public void **setNgayKG**(Date ngayKG) {

this.ngayKG = ngayKG;

}

public String **getGhiChu**() {

return ghiChu;

}

public void **setGhiChu**(String ghiChu) {

this.ghiChu = ghiChu;

}

public String **getMaNV**() {

return maNV;

}

public void **setMaNV**(String maNV) {

this.maNV = maNV;

}

public Date **getNgayTao**() {

return ngayTao;

}

public void **setNgayTao**(Date ngayTao) {

this.ngayTao = ngayTao;

}

}

#### hocVienDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package dao;

import helper.jdbcHelper;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

import model.hocVien;

/\*\*

\*

\* **@author** Jason

\*/

public class **hocVienDAO** {

//đọc 1 nhân viên từ 1 bản ghi (1 ResultSet)

public hocVien **readFromResultSet**(ResultSet rs) throws SQLException{

hocVien model=new hocVien();

model.setMaHV(rs.getInt("MaHV"));

model.setMaKH(rs.getInt("KH"));

model.setMaNH(rs.getString("MaNH"));

model.setDiem(rs.getDouble("Diem"));

return model;

}

//thực hiện truy vấn lấy về 1 tập ResultSet rồi điền tập ResultSet đó vào 1 List

public List<hocVien> **select**(String sql,Object...args){

List<hocVien> list=new ArrayList<>();

try {

ResultSet rs=null;

try{

rs=jdbcHelper.*executeQuery*(sql, args);

while(rs.next()){

list.add(readFromResultSet(rs));

}

}finally{

rs.getStatement().getConnection().close(); //đóng kết nối từ resultSet

}

} catch (SQLException ex) {

throw new RuntimeException();

}

return list;

}

public void **insert**(hocVien model) {

String sql="INSERT INTO HocVien(MaKH, MaNH, Diem) VALUES(?, ?, ?)";

jdbcHelper.*executeUpdate*(sql,

model.getMaKH(),

model.getMaNH(),

model.getDiem());

}

public void **update**(hocVien model) {

String sql="UPDATE HocVien SET MaKH=?, MaNH=?, Diem=? WHERE MaHV=?";

jdbcHelper.*executeUpdate*(sql,

model.getMaKH(),

model.getMaNH(),

model.getDiem(),

model.getMaHV());

}

public void **delete**(Integer MaHV){

String sql="DELETE FROM HocVien WHERE MaHV=?";

jdbcHelper.*executeUpdate*(sql, MaHV);

}

public List<hocVien> **select**() {

String sql="SELECT \* FROM HocVien";

return select(sql); //trong 1 class có thể có 2 method trùng tên (nhưng param khác nhau)

}

public hocVien **findById**(String id) {

String sql="SELECT \* FROM HocVien WHERE MaHV=?";

List<hocVien> list=select(sql, id);

return list.size()>0?list.get(0):null; //có thể trả về là null

}

}

#### thongKeDAO

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package dao;

import helper.jdbcHelper;

import helper.utilityHelper;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\*

\* **@author** Jason

\*/

public class **thongKeDAO** {

/\*

thống kê số người học của trung tâm theo từng năm

return 1 <Object[]> list : Năm - số lượng - ngày người đầu tiên đk - ngày người cc đk

\*/

public List<Object[]> **getNguoiHoc**() {

List<Object[]> list = new ArrayList<>();

try {

ResultSet rs = null;

try {

String sql = "{call sp\_ThongKeNguoiHoc}";

rs = jdbcHelper.*executeQuery*(sql);

while (rs.next()) {

Object[] model = {

rs.getInt("Nam"),

rs.getInt("SoLuong"),

rs.getDate("DauTien"),

rs.getDate("CuoiCung")

};

list.add(model);

}

} finally {

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

}

return list;

}

/\*

bảng điểm của các học viên trong khóa học

@param makh mã khóa học

@return <Object[]> list : mã NH - họ và tên - điểm - xếp loại

\*/

public List<Object[]> **getBangDiem**(Integer makh) {

List<Object[]> list = new ArrayList<>();

try {

ResultSet rs = null;

try {

String sql = "{call sp\_BangDiem (?)}";

rs = jdbcHelper.*executeQuery*(sql, makh);

while (rs.next()) {

Object[] model = {

rs.getString("MaNH"),

rs.getString("HoTen"),

rs.getDouble("Diem"),

utilityHelper.*getRank*(rs.getDouble("Diem"))

};

list.add(model);

}

} finally {

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

ex.printStackTrace();

throw new RuntimeException(ex);

}

return list;

}

/\*

tổng hợp điểm của theo từng chuyên đề

@return <Object[]> list : tên chuyên đề - số HV - điểm thấp nhất - điểm cao nhất - điểm trung bình

\*/

public List<Object[]> **getDiemTheoChuyenDe**() {

List<Object[]> list = new ArrayList<>();

try {

ResultSet rs = null;

try {

String sql = "{call sp\_ThongKeDiem}";

rs = jdbcHelper.*executeQuery*(sql);

while (rs.next()) {

Object[] model = {

rs.getString("ChuyenDe"),

rs.getInt("SoHV"),

rs.getDouble("ThapNhat"),

rs.getDouble("CaoNhat"),

rs.getDouble("TrungBinh")

};

list.add(model);

}

} finally {

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

}

return list;

}

/\*

tổng hợp doanh thu từng chuyên đề (theo từng năm)

@param int nam

@return <Object[]> list : tên chuyên đề - số KH - số HV - doanh thu - HP cao nhất - HP thấp nhất - HP trung bình

\*/

public List<Object[]> **getDoanhThu**(int nam) {

List<Object[]> list = new ArrayList<>();

try {

ResultSet rs = null;

try {

String sql = "{call sp\_ThongKeDoanhThu (?)}";

rs = jdbcHelper.*executeQuery*(sql, nam);

while (rs.next()) {

Object[] model = {

rs.getString("ChuyenDe"),

rs.getInt("SoKH"),

rs.getInt("SoHV"),

rs.getDouble("DoanhThu"),

rs.getDouble("ThapNhat"),

rs.getDouble("CaoNhat"),

rs.getDouble("TrungBinh")

};

list.add(model);

}

} finally {

rs.getStatement().getConnection().close();

}

} catch (SQLException e) {

throw new RuntimeException(e);

}

return list;

}

public List<Integer> **getNamKhaiGiang**() {

List<Integer> list = new ArrayList<>();

try {

ResultSet rs = null;

try {

rs = jdbcHelper.*executeQuery*("select distinct year(NgayKG) as nam from KhoaHoc order by year(NgayKG) desc");

while (rs.next()) {

int nam = rs.getInt(1);

list.add(nam);

}

}finally{

rs.getStatement().getConnection().close();

}

} catch (SQLException ex) {

throw new RuntimeException(ex);

}

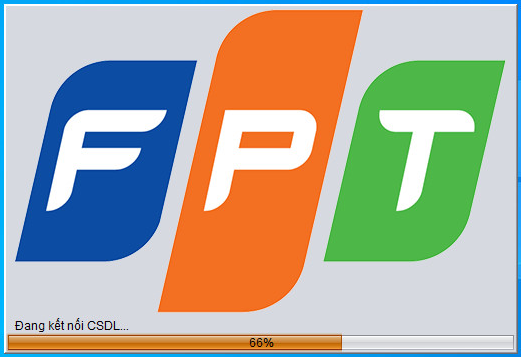
return list;

}

}

## Viết mã cho ứng dụng

### Màn hình chào (chaoJDialog)



HOẠT ĐỘNG

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
|  |  |
| Constructor | init() |
|  |  |

HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN

public void **init**(){

setLocationRelativeTo(null);

//thread là 1 khỗi mã thực hiện nhiệm vụ

Thread t=new Thread(){

int i=-1;

@Override

public void **run**(){

while(true){

try {

i++;

prg.setValue(i);

if(i==20)lblStatus.setText(" Đang khởi tạo cái modun...");

if(i==50)lblStatus.setText(" Đang kết nối CSDL...");

if(i==90)lblStatus.setText(" Chuẩn bị vào chương trình...");

if(i==100){

ChaoJDialog.this.dispose(); //đóng ChaoJDialog

break;

}

Thread.*sleep*(20); //thread tạm dừng hoạt động trong 20 ms

} catch (InterruptedException ex) {

break;

}

}

}

};

t.start(); //thread bắt đầu hoạt động

}

### Form đăng nhập (dangNhapJDialog)



MÔ TẢ HOẠT ĐỘNG

|  |  |
| --- | --- |
| **Sự kiện** | **Công việc** |
|  |  |
| Constructor | Init() |
| btnDangNhap | txtMaNV.setBackground(*white*);  txtMatKhau.setBackground(*white*);  if(txtMaNV.getText().trim().length()>0){  if(txtMatKhau.getPassword().length>0){  login();  }else{  txtMatKhau.setBackground(*pink*);  dialogHelper.*alert*(this, "Không được để trống tên mật khẩu");  }  }else{  txtMaNV.setBackground(*pink*);  dialogHelper.*alert*(this, "Không được để trống tên đăng nhập");  } |
| btnKetThuc | exit() |

HƯỚNG DẪN CÀI ĐẶT MÃ

Bổ sung mã sau vào cuối lớp DangNhapJFrame.

nhanVienDAO dao=new nhanVienDAO(); //tạo đối tượng nhanVienDAO

void **init**(){

setLocationRelativeTo(null);

}

void **login**() {

String manv = txtMaNV.getText();

String matKhau = new String(txtMatKhau.getPassword());

try {

nhanVien nhanVien = dao.findById(manv);

/\*

manv là tên đăng nhập

nhanVien findbyId(String manv)

\*/

if(nhanVien != null){ //nếu manv đúng

String matKhau2 = nhanVien.getMatKhau();

if(matKhau.equals(matKhau2)){ //nếu mật khẩu đúng

shareHelper.*USER* = nhanVien;

dialogHelper.*alert*(this, "Đăng nhập thành công!");

this.dispose();

}

else{

dialogHelper.*alert*(this, "Sai mật khẩu!");

}

}

else{

dialogHelper.*alert*(this, "Sai tên đăng nhập!");

}

}

catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

void **exit**(){

if(dialogHelper.*confirm*(this, "Bạn có muốn thoát khỏi ứng dụng không?")){

System.*exit*(0);

}

}

### Đổi mật khẩu (doiMatKhauJDialog)



|  |  |
| --- | --- |
| Sự kiện | Công việc |
| Constructor | init() |
| btnXacNhan | if(utilityHelper.*checkNullPass*(txtMatKhau)&&  utilityHelper.*checkNullPass*(txtMatKhauMoi)&&  utilityHelper.*checkNullPass*(txtXacNhanMKM)){  doiMatKhau();  } |
| btnHuy | this.dispose(); |

nhanVienDAO dao=new nhanVienDAO();

public void **init**(){

txtMaNV.setText(shareHelper.*USER*.getMaNV());

}

public void **doiMatKhau**(){

txtXacNhanMKM.setBackground(Color.*white*);

txtMatKhau.setBackground(Color.*white*);

String matKhau=new String(txtMatKhau.getPassword());

String matKhauMoi=new String(txtMatKhauMoi.getPassword());

String xacNhanMKM=new String(txtXacNhanMKM.getPassword());

if(matKhau.equals(shareHelper.*USER*.getMatKhau())){

if(matKhauMoi.equals(xacNhanMKM)){

shareHelper.*USER*.setMatKhau(matKhauMoi);

dao.update(shareHelper.*USER*);

dialogHelper.*alert*(this, "Đổi mật khẩu thành công!!");

this.dispose();

}else{

txtXacNhanMKM.setBackground(Color.*pink*);

dialogHelper.*alert*(this, "Mật khẩu xác nhận không trùng mật khẩu");

}

}else{

txtMatKhau.setBackground(Color.*pink*);

dialogHelper.*alert*(this, "Mật khẩu cũ nhập không chính xác!");

}

}

### Màn hình giới thiệu (gioiThieuJDialog)



|  |  |
| --- | --- |
| Sự kiện | Công việc |
| Mouse Clicked | this.dispose(); |

### Form chính (formCha)



MÔ TẢ HOẠT ĐỘNG

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Sự kiện** | **Công việc** |  |
|  |  |  |  |
|  | Constructor | Init() |  |
|  |  |  |  |
|  | Đăng nhập | openLogin(); |  |
|  |  |  |  |
|  | Đăng xuất | Logoff() |  |
|  |  |  |  |
| Đổi mật khẩu | | if (shareHelper.*authenticated*()) {  new doiMatKhauJDialog(this, true).setVisible(true);  } else {  dialogHelper.*alert*(this, "Vui lòng đăng nhập trước khi đổi mật khẩu");  } | |
|  | Kết thuc | Exit() |  |
|  |  |  |  |
|  | Chuyên đề | openChuyenDe() |  |
|  |  |  |  |
|  | Người học | openNguoiHoc() |  |
|  |  |  |  |
|  | Học viên | openHocVien() |  |
|  |  |  |  |
|  | Khóa học | openKhoaHoc() |  |
|  |  |  |  |
|  | Nhân viên | openNhanVien() |  |
|  |  |  |  |
| Thống kê người học từng năm | | openThongKe(1) | |
|  | |  | |
| Bảng điểm khóa | | openThongKe(2) | |
|  | |  | |
| Điểm từng khóa | | openThongKe(3) | |
|  | |  | |
| Doanh thư từng chuyên đề | | openThongKe(4) | |
|  | |  | |
| Hướng dẫn | | openAbout() | |
|  | |  | |
| Giới thiệu | | openWebsite() | |
|  | |  | |
| HƯỚNG DẪN CÀI ĐẶT MÃ NGUỒN | |  | |

void **init**() {

setIconImage(shareHelper.*APP\_ICON*); //đặt icon góc trên trái

setLocationRelativeTo(null); //cho ra giữa màn hình

setExtendedState(JFrame.*MAXIMIZED\_BOTH*); //cho toàn màn hình

/\*

làm đồng hồ

Timer t=new Timer(s, ActionListener); sk lặp lại sau s ms

t.start() / t.stop()

\*/

new Timer(1000, new ActionListener() {

SimpleDateFormat sdf = new SimpleDateFormat("hh:mm:ss a");

@Override

public void **actionPerformed**(ActionEvent e) {

lblDongHo.setText(sdf.format(new Date()));

}

}).start();

showChao(); // gọi chaoJDialog

openLogin(); // gọi dangNhapJDialog

}

public void **showChao**() {

new ChaoJDialog(this, true).setVisible(true);

}

public void **openLogin**() {

for (JInternalFrame frmChild : jdesktop.getAllFrames()) {

frmChild.dispose();

}

new dangNhapJDialog(this, true).setVisible(true);

}

void **logoff**() {

shareHelper.*logoff*();

this.openLogin();

}

void **exit**() {

if (dialogHelper.*confirm*(this, "Bạn thực sự muốn kết thúc?")) {

System.*exit*(0);

}

}

public void **openChuyenDe**() {

if (shareHelper.*authenticated*()) {

chuyenDeJInternalFrame frmChuyenDe = new chuyenDeJInternalFrame();

openX(frmChuyenDe);

} else {

dialogHelper.*alert*(this, "Vui lòng đăng nhập!");

}

}

public void **openNguoiHoc**() {

if (shareHelper.*authenticated*()) {

nguoiHocJInternalFrame frmNguoiHoc = new nguoiHocJInternalFrame();

openX(frmNguoiHoc);

} else {

dialogHelper.*alert*(this, "Vui lòng đăng nhập!");

}

}

void **openAbout**() {

new gioiThieuJDialog(this, true).setVisible(true);

}

void **openWebsite**() {

try {

Desktop.*getDesktop*().browse(new File("help/index.html").toURI());

//thư mục help đặt ngang hàng với src

} catch (IOException ex) {

dialogHelper.*alert*(this, "Không tìm thấy file hướng dẫn!");

}

}

public void **openX**(JInternalFrame x) {

for (JInternalFrame frmChild : jdesktop.getAllFrames()) {

frmChild.dispose();

}

x.setLocation(this.getWidth() / 2 - x.getWidth() / 2,

(this.getHeight() - 20) / 2 - x.getHeight() / 2 - 60);

jdesktop.add(x);

x.setVisible(true);

}

public void **openKhoaHoc**() {

if (shareHelper.*authenticated*()) {

khoaHocJInternalFrame frmKhoaHoc = new khoaHocJInternalFrame();

openX(frmKhoaHoc);

} else {

dialogHelper.*alert*(this, "Vui lòng đăng nhập!");

}

}

public void **openNhanVien**() {

if (shareHelper.*authenticated*()) {

nhanVienJInternalFrame frmNhanVien = new nhanVienJInternalFrame();

openX(frmNhanVien);

} else {

dialogHelper.*alert*(this, "Vui lòng đăng nhập!");

}

}

public void **openThongKe**(int index) {

if (shareHelper.*authenticated*()) {

thongKeJInternalFrame frmThongKe = new thongKeJInternalFrame(index);

openX(frmThongKe);

} else {

dialogHelper.*alert*(this, "Vui lòng đăng nhập!");

}

}

public void **openGioiThieu**() {

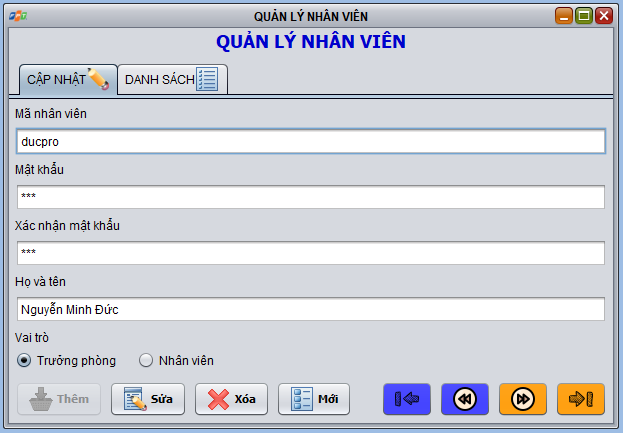
gioiThieuJDialog gt = new gioiThieuJDialog(this, true);

gt.setLocationRelativeTo(null);

gt.setVisible(true);

}

### Form nhân viên (nhanVienJInternalFrame)



|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init() |
| JinternalFrameOpened | this.load();  this.setStatus(true); |
| tblGridView.mouseClicked | if(evt.getClickCount() == 2){  this.index = tblGridView.rowAtPoint(evt.getPoint());  if (this.index >= 0) {  this.edit();  tabs.setSelectedIndex(0);  }  } |
| btnInsert | if (utilityHelper.*checkNullText*(txtMaNV)  && utilityHelper.*checkNullPass*(txtMatKhau)  && utilityHelper.*checkNullPass*(txtXacNhanMK)  && utilityHelper.*checkNullText*(txtHoTen)) {  if (utilityHelper.*checkMaNV*(txtMaNV)  && utilityHelper.*checkPass*(txtMatKhau)  && utilityHelper.*checkName*(txtHoTen)) {  if (checkTrungMa(txtMaNV)) {  insert();  }  }  } |
| btnUpdate | if (utilityHelper.*checkNullPass*(txtMatKhau)  && utilityHelper.*checkNullPass*(txtXacNhanMK)  && utilityHelper.*checkNullText*(txtHoTen)) {  if (utilityHelper.*checkPass*(txtMatKhau)  && utilityHelper.*checkName*(txtHoTen)) {  update();  }  } |
| btnDelete | if(shareHelper.*USER*.isVaiTro()){  if(checkChinhMinh(txtMaNV)){  delete();  }  }else{  dialogHelper.*alert*(this, "Chỉ trưởng phòng mới được phép xóa");  } |
| btnClear | clear() |
| btnFirst | this.index=0;  this.edit(); |
| btnPrev | this.index--;  this.edit(); |
| btnNext | this.index++;  this.edit(); |
| btnLast | this.index = tblGridView.getRowCount() -1;  this.edit(); |

int index = 0; // vị trí của nhân viên đang đc chọn

nhanVienDAO dao = new nhanVienDAO();

void **init**() {

setFrameIcon(shareHelper.*APP\_ICON\_1*); //set Icon cho JInternalFrame

tabs.setSelectedIndex(1); //chuyển tabsPanel sang tab 2

}

//lấy list nhanVien từ CSDL điền vào bảng

//LƯU Ý: CHỈ TRƯỞNG PHÒNG MỚI XEM ĐƯỢC MẬT KHẨU CỦA NGƯỜI KHÁC

void **load**() {

/\*

2 cách kết nối model với bảng:

cách 1: DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

cách 2: tblGridView.setModel(model);

\*/

DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

model.setRowCount(0); //đưa số dòng về 0 (xóa bảng)

try {

List<nhanVien> list = dao.select(); //lấy tất cả nhân viên trong CSDL đưa vào list

/\*

3 cách đưa dữ liệu vào bảng:

cách 1: dùng mảng 2 chiều

String[] header ={"ten 1", "ten 2",...};

Object[][] data ={{x1, x2,...}, {c1, c2,...},...};

DefaultTableModel model = new DefaultTableModel(header, data);

tblGridView.setModel(model);

cách 2: dùng mảng 1 chiều (như trong bài này)

tblGridView.setModel(model);

for(...){

Object[] row ={x1, x2,...}

model.addRow(row);

}

cách 3: dùng vector (như cách 2 nhưng vector thay cho mảng 1 chiều

tblGridView.setModel(model);

for(...){

Vector row=new Vector();

row.add(x1);

row.add(x2);

...

model.addRow(row);

}

\*/

for (nhanVien nv : list) {

Object[] row = {

nv.getMaNV(),

shareHelper.*USER*.isVaiTro()?nv.getMatKhau():matKhauToSao(nv.getMatKhau()),

nv.getHoTen(),

nv.isVaiTro() ? "Trưởng phòng" : "Nhân viên"

};

model.addRow(row);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

//chỉ trưởng phòng mới đc xem mật khẩu người khác

public String **matKhauToSao**(String pass){

String sao="";

for(int i=0;i<pass.length();i++){

sao+="\*";

}

return sao;

}

//lấy thông tin trên form để

//thêm (đăng kí) nhân viên - trưởng phòng vào CSDL

void **insert**() {

//getModel() viết ở dưới, lấy thông tin trên form điền vào đt model

nhanVien model = getModel();

/\*

2 cách so sánh 2 mật khẩu:

cách 1: chuyển String về char[] rồi so sanh 2 mảng

char[] x= chuoi.toCharArray();

so sánh 2 char[]: boolean b = Array.equals(mang1, mang2);

cách 2: chuyển char[] về String rồi so sanh 2 chuỗi

String chuoi = new String(mangChar); (cách dưới)

hoặc for(char x: mang){

chuoi+=x.toString();

}

rồi so sánh 2 chuỗi: boolean b = chuoi1.equals(chuoi2);

\*/

String confirm = new String(txtXacNhanMK.getPassword()); //cách chuyển char[] sang String

//so sánh mật khẩu và MK xác nhận

txtXacNhanMK.setBackground(*white*);

if (confirm.equals(model.getMatKhau())) {

try {

dao.insert(model);

this.load(); //cập nhật lại bảng nv

this.clear(); // xóa trắng form và vẫn để insertable

dialogHelper.*alert*(this, "Thêm mới thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Thêm mới thất bại!");

}

} else {

txtXacNhanMK.setBackground(*pink*);

dialogHelper.*alert*(this, "Xác nhận mật khẩu không đúng!");

}

}

//lấy thông tin trên form để

//cập nhật nhanVien theo maNV

void **update**() {

nhanVien model = getModel();

String confirm = new String(txtXacNhanMK.getPassword());

txtXacNhanMK.setBackground(*white*);

if (!confirm.equals(model.getMatKhau())) {

txtXacNhanMK.setBackground(*pink*);

dialogHelper.*alert*(this, "Xác nhận mật khẩu không đúng!");

} else {

try {

dao.update(model); //cập nhật nhân viên theo maNV

this.load(); //điền tt mới vào bảng

dialogHelper.*alert*(this, "Cập nhật thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Cập nhật thất bại!");

}

}

}

//lấy maNV trên form, xóa nv theo maNV

//xóa trắng form chuyển sang insertable

void **delete**() {

if (dialogHelper.*confirm*(this, "Bạn thực sự muốn xóa nhân viên này?")) {

String manv = txtMaNV.getText();

try {

dao.delete(manv); //xóa nhân viên theo maNV

this.load(); //điền tt mới vào bảng

this.clear(); //xóa trắng form và chỉnh lại status

dialogHelper.*alert*(this, "Xóa thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Xóa thất bại!");

}

}

}

//điền thông tin đt nhanVien lên form (theo index)

//chuyển sang chế độ editable

void **edit**() {

setTrang();

try {

String manv = (String) tblGridView.getValueAt(this.index, 0); //lấy maNV từ bảng theo index

nhanVien model = dao.findById(manv); //dùng maNV tìm ra đối tượng nhanVien

if (model != null) {

this.setModel(model); //điền thông tin đt nhanVien lên form

this.setStatus(false); //chuyển sang editable

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

public void **setTrang**(){

txtMaNV.setBackground(*white*);

txtMatKhau.setBackground(*white*);

txtXacNhanMK.setBackground(*white*);

txtHoTen.setBackground(*white*);

}

//xóa trắng form và chuyển sang chế độ insertable

void **clear**() {

setTrang();

this.setModel(new nhanVien()); //xóa trắng form

this.setStatus(true);

}

//param nhanVien

//đưa thông tin nhanVien lên form theo model

void **setModel**(nhanVien model) {

txtMaNV.setText(model.getMaNV());

txtHoTen.setText(model.getHoTen());

txtMatKhau.setText(model.getMatKhau());

txtXacNhanMK.setText(model.getMatKhau());

rdoTruongPhong.setSelected(model.isVaiTro());

rdoNhanVien.setSelected(!model.isVaiTro());

}

//return nhanVien

//lấy thông tin trên form cho vào đt nhanVien

nhanVien **getModel**() {

nhanVien model = new nhanVien();

model.setMaNV(txtMaNV.getText());

model.setHoTen(txtHoTen.getText());

model.setMatKhau(new String(txtMatKhau.getPassword())); //chuyển char[] thành String

model.setVaiTro(rdoTruongPhong.isSelected());

return model;

}

/\*

param: boolean

insertable: (true) THÊM NHÂN VIÊN MỚI

enable: txtMaNV, btnInsert

disable: btnUpdate, btnDelete, 4 nút điều hướng

editable: (false) ngược lại XEM VÀ SỬA

btnClear lúc nào cũng enable

4 nút điều hướng: disble 2 nút tương ứng khi ở đầu và cuối danh sách

\*/

void **setStatus**(boolean insertable) {

txtMaNV.setEditable(insertable); //enable txtMaNV

btnInsert.setEnabled(insertable); //enable btnInsert

btnUpdate.setEnabled(!insertable); //disable btnUpdate

btnDelete.setEnabled(!insertable); //disable brnDelete

boolean first = this.index > 0;

boolean last = this.index < tblGridView.getRowCount() - 1;

btnFirst.setEnabled(!insertable && first);//enable 4 nút này khi ở editable

btnPrev.setEnabled(!insertable && first); //disable khi First, Prev khi ở bản ghi đầu (index = 0)

btnNext.setEnabled(!insertable && last); //disable khi Next, Last khi ở bản ghi cuối

btnLast.setEnabled(!insertable && last); //index = tblGridView.getRowCount() - 1

}

public boolean **checkChinhMinh**(JTextField txt){

nhanVien nv=dao.findById(txt.getText());

if (nv.getMaNV().equals(*USER*.getMaNV())) {

dialogHelper.*alert*(this, "bạn không được xóa chính mình.");

return false;

} else {

return true;

}

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

txt.setBackground(*pink*);

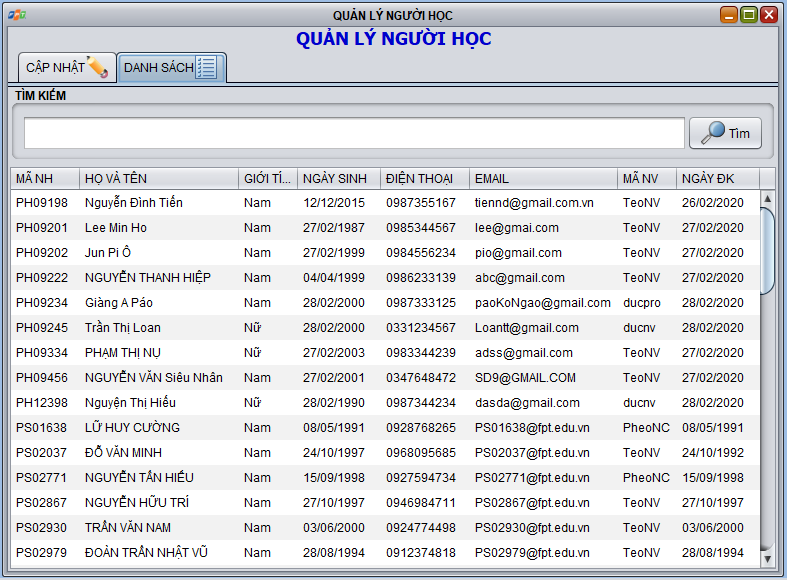
dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

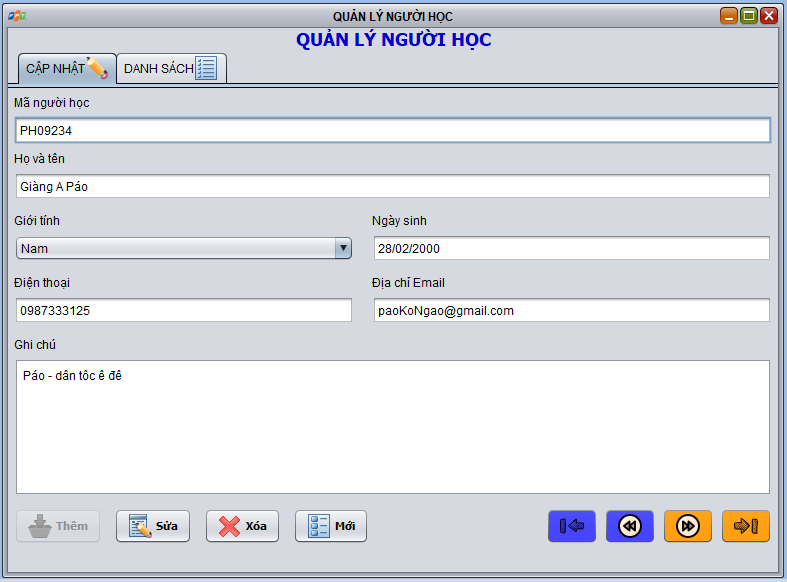
return false;

}

}

### Form người học (nguoiHocJInternalFrame)





|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init() |
| JinternalFrameOpened | this.load();  this.setStatus(true); |
| tblGridView.mouseClicked | if(evt.getClickCount() == 2){  this.index = tblGridView.rowAtPoint(evt.getPoint());  if (this.index >= 0) {  this.edit();  tabs.setSelectedIndex(0);  }  } |
| btnInsert | if (utilityHelper.*checkNullText*(txtMaNH)  && utilityHelper.*checkNullText*(txtHoTen)  && utilityHelper.*checkNullText*(txtNgaySinh)  && utilityHelper.*checkNullText*(txtDienThoai)  && utilityHelper.*checkNullText*(txtEmail)) {  if (utilityHelper.*checkMaNH*(txtMaNH)  && utilityHelper.*checkName*(txtHoTen)  && utilityHelper.*checkDate*(txtNgaySinh)  && utilityHelper.*checkSDT*(txtDienThoai)  && utilityHelper.*checkEmail*(txtEmail)) {  if (checkTrungMa(txtMaNH)) {  if (check16Nam(txtNgaySinh)) {  insert();  }  }  }  } |
| btnUpdate | if (utilityHelper.*checkNullText*(txtHoTen)  && utilityHelper.*checkNullText*(txtNgaySinh)  && utilityHelper.*checkNullText*(txtDienThoai)  && utilityHelper.*checkNullText*(txtEmail)) {  if (utilityHelper.*checkName*(txtHoTen)  && utilityHelper.*checkDate*(txtNgaySinh)  && utilityHelper.*checkSDT*(txtDienThoai)  && utilityHelper.*checkEmail*(txtEmail)) {  if (check16Nam(txtNgaySinh)) {  update();  }  }  } |
| btnDelete | delete() |
| btnClear | clear() |
| btnFirst | this.index=0;  this.edit(); |
| btnPrev | this.index--;  this.edit(); |
| btnNext | this.index++;  this.edit(); |
| btnLast | this.index = tblGridView.getRowCount() -1;  this.edit(); |
| btnTimKiem | this.load();  this.clear(); |

int index = 0; //vị trí nguoiHoc đang được chọn

nguoiHocDAO dao = new nguoiHocDAO();

void **init**() {

setFrameIcon(shareHelper.*APP\_ICON\_1*); //set Icon cho JInternalFrame

tabs.setSelectedIndex(1); //chuyển tabPanel sang tab thứ 2

}

//tìm kiếm người học theo keyword rồi đổ list người học vào bảng

//nếu không có keywork trong ô tìm kiếm thì đổ toàn bộ người học trong CSDL vào bảng

void **load**() {

//kết nối nội dung bảng (model) với thùng chứa bảng (tblGridView)

DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

model.setRowCount(0); //đưa số dòng bảng về 0 (xóa bảng)

try {

//tìm người học theo keywork

//nếu ko có keywork thì sẽ là tất cả người học

String keyword = txtTimKiem.getText();

List<nguoiHoc> list = dao.selectByKeyword(keyword);

//đưa list tìm được lên bảng

for (nguoiHoc nh : list) {

Object[] row = {

nh.getMaNH(),

nh.getHoTen(),

nh.isGioiTinh() ? "Nam" : "Nữ",

dateHelper.*toString*(nh.getNgaySinh()),

nh.getDienThoai(),

nh.getEmail(),

nh.getMaNV(),

dateHelper.*toString*(nh.getNgayDK())

};

model.addRow(row);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

//lấy thông tin trên form để

//thêm 1 nguoiHoc mới vào CSDL (vẫn insertable)

void **insert**() {

nguoiHoc model = getModel(); //lấy thông tin trên form gán cho đt nguoiHoc

try {

dao.insert(model); //thêm bản ghi mới vào CSDL theo tt từ nguoiHoc

this.load(); //đổ thông tin mới vào bảng

this.clear(); //xóa trằng form và vẫn để ở chế độ insertable

dialogHelper.*alert*(this, "Thêm mới thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Thêm mới thất bại!");

}

}

//lấy thông tin trên form

//để chỉnh sửa nguoiHoc trong CSDL theo maNH

void **update**() {

nguoiHoc model = getModel(); //lấy thông tin form gán cho đt nguoiHoc

try {

dao.update(model); //chỉnh sửa bản ghi theo tt từ nguoiHoc

this.load(); //đổ tt mới từ CSDL vào bảng

dialogHelper.*alert*(this, "Cập nhật thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Cập nhật thất bại!");

e.printStackTrace();

}

}

//xóa bản ghi khỏi CSDL theo maNH lấy trên form

//xóa trằng form và chuyển sang insertable

void **delete**() {

if (dialogHelper.*confirm*(this, "Bạn thực sự muốn xóa người học này?")) {

String manh = txtMaNH.getText();

try {

dao.delete(manh);

this.load();

this.clear();

dialogHelper.*alert*(this, "Xóa thành công!");

} catch (HeadlessException e) {

dialogHelper.*alert*(this, "Xóa thất bại!");

}

}

}

public void **setTrang**() {

txtMaNH.setBackground(*white*);

txtHoTen.setBackground(*white*);

txtNgaySinh.setBackground(*white*);

txtDienThoai.setBackground(*white*);

txtEmail.setBackground(*white*);

}

//xóa trằng form, ngayDK lấy ngày hôm nay, maNV tạo lấy maNV đang đăng nhập

//chuyển sang insertable

void **clear**() {

setTrang();

nguoiHoc model = new nguoiHoc();

model.setMaNV(shareHelper.*USER*.getMaNV());

model.setNgayDK(dateHelper.*now*());

this.setModel(model);

setStatus(true);

}

//điền thông tin lên form theo index

//chuyển sang trạng thái editable

void **edit**() {

setTrang();

try {

String manh = (String) tblGridView.getValueAt(this.index, 0); //lấy maNH theo index

nguoiHoc model = dao.findById(manh); //lấy nguoiHoc theo maNH

if (model != null) {

this.setModel(model); //điền thông tin lên form theo nguoiHoc

this.setStatus(false);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

//điền thông tin từ đối tượng nguoiHoc lên form

void **setModel**(nguoiHoc model) {

txtMaNH.setText(model.getMaNH());

txtHoTen.setText(model.getHoTen());

cboGioiTinh.setSelectedIndex(model.isGioiTinh() ? 0 : 1);

txtNgaySinh.setText(dateHelper.*toString*(model.getNgaySinh()));

txtDienThoai.setText(model.getDienThoai());

txtEmail.setText(model.getEmail());

txtGhiChu.setText(model.getGhiChu());

}

//lấy thông trên form cho vào đối tượng nguoiHoc

//return nguoiHoc

nguoiHoc **getModel**() {

nguoiHoc model = new nguoiHoc();

model.setMaNH(txtMaNH.getText());

model.setHoTen(txtHoTen.getText());

model.setGioiTinh(cboGioiTinh.getSelectedIndex() == 0);

model.setNgaySinh(dateHelper.*toDate*(txtNgaySinh.getText()));

model.setDienThoai(txtDienThoai.getText());

model.setEmail(txtEmail.getText());

model.setGhiChu(txtGhiChu.getText());

model.setMaNV(shareHelper.*USER*.getMaNV());

model.setNgayDK(dateHelper.*now*()); //ngày đăng kí luôn là ngày hôm nay dù có sửa trên form

return model;

}

/\*

insertable (true)

enable: txtMaNH, btnInsert

disable: btnDelete, btnUpdate, 4 nút điều hướng

editable (false) ngược lại

btnClear lúc nào cũng enable

ở đầu tiên disable first, prev, ở cuối cùng disable last, next

\*/

void **setStatus**(boolean insertable) {

txtMaNH.setEditable(insertable);

btnInsert.setEnabled(insertable);

btnUpdate.setEnabled(!insertable);

btnDelete.setEnabled(!insertable);

boolean first = this.index > 0;

boolean last = this.index < tblGridView.getRowCount() - 1;

btnFirst.setEnabled(!insertable && first);

btnPrev.setEnabled(!insertable && first);

btnLast.setEnabled(!insertable && last);

btnNext.setEnabled(!insertable && last);

}

public boolean **check16Nam**(JTextField txt) {

txt.setBackground(*white*);

Date date = dateHelper.*toDate*(txt.getText());

Calendar c1 = Calendar.*getInstance*();

Calendar c2 = Calendar.*getInstance*();

c1.setTime(date);

c2.setTime(new Date());

long a = (c2.getTime().getTime() - c1.getTime().getTime()) / (24 \* 3600 \* 1000);

if (a >= 5844) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " phải cách đây ít nhất 16 năm.");

return false;

}

}

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

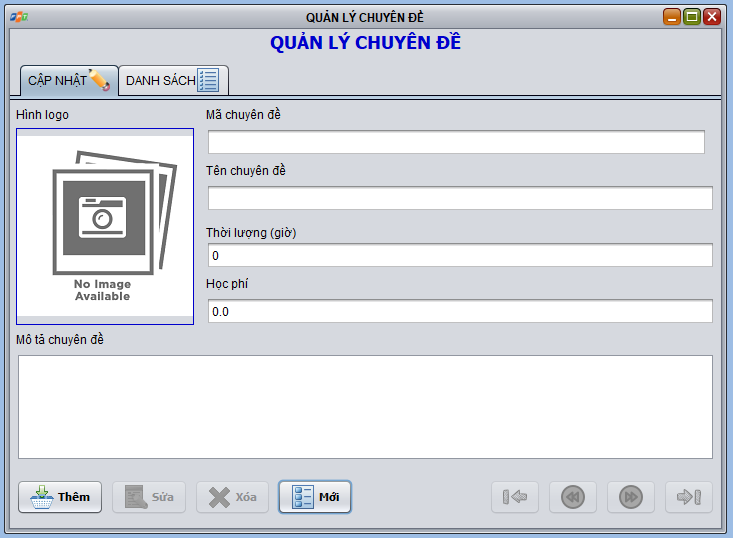
txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

return false;

}}

### Form chuyên đề (chuyenDeJInternalFrame)



|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init() |
| JinternalFrameOpened | this.load();  this.setStatus(true); |
| tblGridView.mouseClicked | if(evt.getClickCount() == 2){  this.index = tblGridView.rowAtPoint(evt.getPoint());  if (this.index >= 0) {  this.edit();  tabs.setSelectedIndex(0);  }  } |
| btnInsert | if(utilityHelper.*checkNullText*(txtMaCD)&&  utilityHelper.*checkNullText*(txtTenCD)&&  utilityHelper.*checkNullText*(txtThoiLuong)&&  utilityHelper.*checkNullText*(txtHocPhi)&&  utilityHelper.*checkNullText*(txtMoTa)&&  checkNullHinh()){  if(utilityHelper.*checkMaCD*(txtMaCD)&&  utilityHelper.*checkTenCD*(txtTenCD)&&  utilityHelper.*checkThoiLuong*(txtThoiLuong)&&  utilityHelper.*checkHocPhi*(txtHocPhi)&&  utilityHelper.*checkMoTaCD*(txtMoTa)){  if(checkTrungMa(txtMaCD)){  insert();  }  }  } |
| btnUpdate | if(utilityHelper.*checkNullText*(txtMaCD)&&  utilityHelper.*checkNullText*(txtTenCD)&&  utilityHelper.*checkNullText*(txtThoiLuong)&&  utilityHelper.*checkNullText*(txtHocPhi)&&  utilityHelper.*checkNullText*(txtMoTa)&&  checkNullHinh()){  if(utilityHelper.*checkMaCD*(txtMaCD)&&  utilityHelper.*checkTenCD*(txtTenCD)&&  utilityHelper.*checkThoiLuong*(txtThoiLuong)&&  utilityHelper.*checkHocPhi*(txtHocPhi)&&  utilityHelper.*checkMoTaCD*(txtMoTa)){  update();  }  } |
| btnDelete | delete() |
| btnClear | clear() |
| btnFirst | this.index=0;  this.edit(); |
| btnPrev | this.index--;  this.edit(); |
| btnNext | this.index++;  this.edit(); |
| btnLast | this.index = tblGridView.getRowCount() -1;  this.edit(); |
| lblHinh | this.selectImage(); |

int index = 0; //vị trí chuyenDe đang được chọn

chuyenDeDAO dao = new chuyenDeDAO();

JFileChooser fileChooser = new JFileChooser();

//hoặc JFileChooser c = new JFileChooser(String duongDan); //đường dẫn đến folder mở mặc định

void **init**() {

setFrameIcon(shareHelper.*APP\_ICON\_1*); //set Icon cho JInternalFrame

tabs.setSelectedIndex(1); //tabPanel chuyển sang tab 2

fileChooser.setDialogTitle("Chọn Logo cho chuyên đề"); //set title cho dialog

//dialog được nhận logo từ form cha của nó showOpenDialog(this) ko phải null

}

//xóa bảng, đổ toàn bộ chuyenDe từ CSDL vào bảng

void **load**() {

DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

model.setRowCount(0);

try {

List<chuyenDe> list = dao.select();

for (chuyenDe cd : list) {

Object[] row = {

cd.getMaCD(),

cd.getTenCD(),

cd.getHocPhi(),

cd.getThoiLuong(),

cd.getHinh()

};

model.addRow(row);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

/\*

lấy thông tin trên form, thêm bản ghi chuyenDe mới vào CSDL

load lại bảng

xóa trắng form, để ở trạng thái insertable

\*/

void **insert**() {

chuyenDe model = getModel();

try {

dao.insert(model);

this.load();

this.clear();

dialogHelper.*alert*(this, "Thêm mới thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Thêm mới thất bại!");

}

}

//lấy thông tin trên form, cập nhật vào bản ghi CSDL theo maCD

//load lại bảng

void **update**() {

chuyenDe model = getModel();

try {

dao.update(model);

this.load();

dialogHelper.*alert*(this, "Cập nhật thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Cập nhật thất bại!");

}

}

//xóa bản ghi trong CSDL theo maCD lấy trên form

//load lại bảng

//xóa trắng form, chuyển sang insertable

void **delete**() {

if (dialogHelper.*confirm*(this, "Bạn có muốn xóa hay không?")) {

String macd = txtMaCD.getText();

try {

dao.delete(macd);

this.load();

this.clear();

dialogHelper.*alert*(this, "Xóa thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Xóa thất bại!");

}

}

}

//xóa trắng form, chuyển sang insertable

void **clear**() {

this.setModel(new chuyenDe());

this.setStatus(true);

}

//lấy maCD theo index, lấy đt chuyenDe từ CSDL theo maCD

//hiển thị thông tin từ đt chuyenDe lên form, chuyển sang editable

void **edit**() {

try {

String macd = (String) tblGridView.getValueAt(this.index, 0);

chuyenDe model = dao.findById(macd);

if (model != null) {

this.setModel(model);

this.setStatus(false);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

//lấy thông tin từ đt chuyenDe đưa lên form

//lưu ý lấy hình ảnh từ thư mục logos đưa lên form theo tenFile lấy từ chuyenDe

void **setModel**(chuyenDe model) {

txtMaCD.setText(model.getMaCD());

txtTenCD.setText(model.getTenCD());

txtThoiLuong.setText(String.*valueOf*(model.getThoiLuong()));

txtHocPhi.setText(String.*valueOf*(model.getHocPhi()));

txtMoTa.setText(model.getMoTa());

lblHinh.setToolTipText(model.getHinh());

if (model.getHinh() != null) {

lblHinh.setIcon(shareHelper.*readLogo*(model.getHinh()));

/\*

ImageIcon readLogo(String tenFile) đọc file trong thư mục logos theo tên file trả về ImageIcon

void setIcon(ImageIcon icon) set Icon cho lbl

\*/

}else{

lblHinh.setIcon(shareHelper.*readLogo*("noImage.png"));

}

}

//lấy thông tin trên form điền vào đt chuyenDe

//return chuyenDe

chuyenDe **getModel**() {

chuyenDe model = new chuyenDe();

model.setMaCD(txtMaCD.getText());

model.setTenCD(txtTenCD.getText());

model.setThoiLuong(Integer.*valueOf*(txtThoiLuong.getText()));

model.setHocPhi(Double.*valueOf*(txtHocPhi.getText()));

model.setHinh(lblHinh.getToolTipText()); //lấy tên hình

model.setMoTa(txtMoTa.getText());

return model;

}

/\*

insertable:

enable: txtMaCD, btnInsert, btnClear

disable: btnUpdate, btnExit, 4 nút điều hướng

editable:

ngược lại

btnClear lúc nào cũng enable

\*/

void **setStatus**(boolean insertable) {

txtMaCD.setEditable(insertable);

btnInsert.setEnabled(insertable);

btnUpdate.setEnabled(!insertable);

btnDelete.setEnabled(!insertable);

boolean first = this.index > 0;

boolean last = this.index < tblGridView.getRowCount() - 1;

btnFirst.setEnabled(!insertable && first);

btnPrev.setEnabled(!insertable && first);

btnLast.setEnabled(!insertable && last);

btnNext.setEnabled(!insertable && last);

}

/\*

hiển thị hộp thoại cho người dùng chọn file

copy file đó vào thư mục logos

đọc file trong thư mục logos và hiển thị lên form, đặt tenFile là toolTipText của lbl

----------------------------------------------------------------------------

khởi tạo: JFileChooser c = new JFileChooser();

hoặc JFileChooser c = new JFileChooser(String duongDan); //đường dẫn đến folder mặc định

hiển thị hộp thoại mở file: c.showOpenDialog(this);

hiển thị hộp thoại lưu file: c.showSaveDialog(this);

2 câu lệnh trên trả về 1 giá trị int là:

JFileChooser.APPROVE\_OPTION người dùng đã chọn (lưu) đc file

JFileChooser.CANCEL\_OPTION người dùng chưa chọn (lưu) đc file

nếu người dùng đã chọn (lưu) đc file

File file = c.getSelectedFile(); lấy file chọn (lưu) đc

String path = c.getSelectedFile().toString(); lấy đường dẫn file chọn (lưu) đc

\*/

void **selectImage**() {

if (fileChooser.showOpenDialog(this) == JFileChooser.*APPROVE\_OPTION*) { //nếu người dùng đã chọn đc file

File file = fileChooser.getSelectedFile(); //lấy file người dùng chọn

if (shareHelper.*saveLogo*(file)) { //sao chép file đã chọn thư mục logos

// Hiển thị hình lên form

lblHinh.setIcon(shareHelper.*readLogo*(file.getName())); //file.getName(); lấy tên của file

//ImageIcon readLogo(String tenFile) đọc file trong thư mục logos theo tên file trả về ImageIcon

//void setIcon(ImageIcon icon) set Icon cho lbl

lblHinh.setToolTipText(file.getName());

}

}

}

public boolean **checkNullHinh**(){

if(lblHinh.getToolTipText()!=null){

return true;

}else{

dialogHelper.*alert*(this, "Không được để trống hình.");

return false;

}

}

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

txt.setBackground(*pink*);

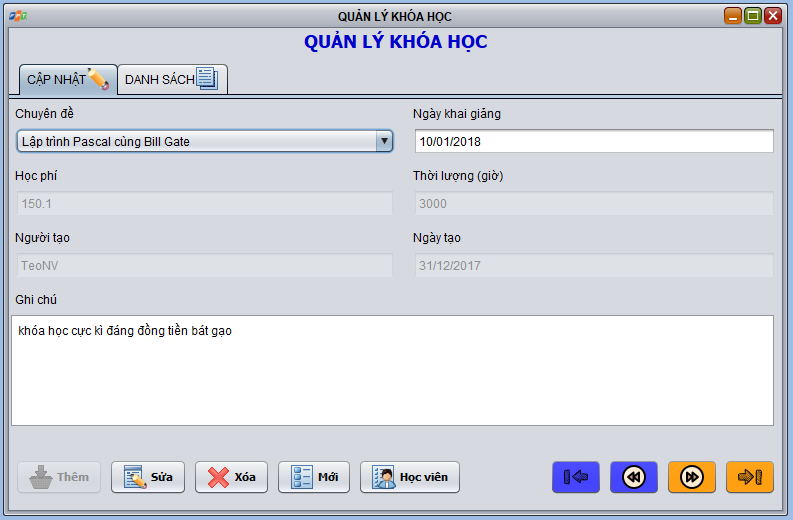
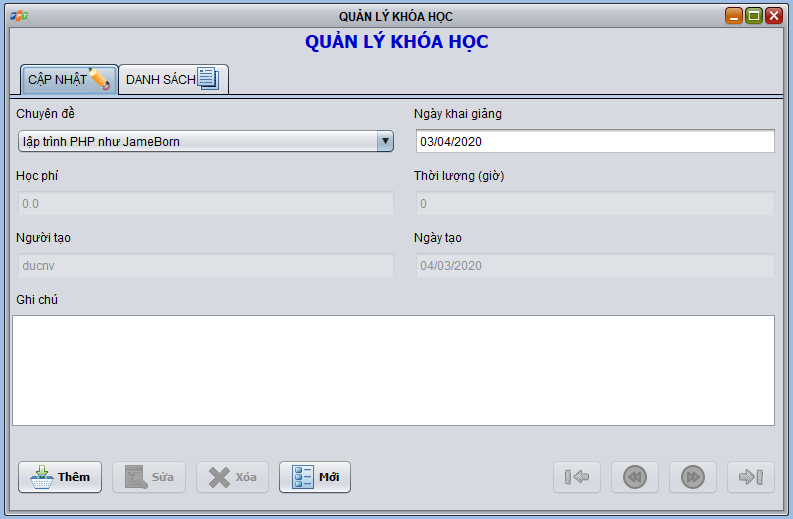
dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

return false;

}

}

### Form khóa học (khoaHocJInternalFrame)



|  |  |
| --- | --- |
| Sự kiện | Công việc phải làm |
| Constructor | init() |
| JinternalFrameOpened | this.fillComboBox();  this.load();  this.clear();  this.setStatus(true); |
| tblGridView.mouseClicked | if(evt.getClickCount() == 2){  this.index = tblGridView.rowAtPoint(evt.getPoint());  if (this.index >= 0) {  this.edit();  tabs.setSelectedIndex(0);  }  } |
| btnInsert | if(utilityHelper.*checkNullText*(txtNgayKG)){  if(utilityHelper.*checkDate*(txtNgayKG)){  if(check5Ngay(txtNgayKG,txtNgayTao)){  insert();  }  }  } |
| btnUpdate | if(utilityHelper.*checkNullText*(txtNgayKG)){  if(utilityHelper.*checkDate*(txtNgayKG)){  if(check5Ngay(txtNgayKG,txtNgayTao)){  update();  }  }  } |
| btnDelete | delete() |
| btnClear | clear() |
| btnFirst | this.index=0;  this.edit(); |
| btnPrev | this.index--;  this.edit(); |
| btnNext | this.index++;  this.edit(); |
| btnLast | this.index = tblGridView.getRowCount() -1;  this.edit(); |
| cboChuyenDe | selectComboBox(); |
| btnStudents | this.openHocVien(); |

int index = 0; //vị trí khoaHoc đang được chọn

khoaHocDAO dao = new khoaHocDAO();

chuyenDeDAO cddao = new chuyenDeDAO();

void **init**() {

setFrameIcon(shareHelper.*APP\_ICON\_1*); //set Icon JInternalFrame

tabs.setSelectedIndex(1); //chuyển tab panel sang tab 2

}

//xóa bảng, đổ toàn bộ khoaHoc từ CSDL vào bảng

void **load**() {

DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

model.setRowCount(0);

try {

List<khoaHoc> list = dao.select();

for (khoaHoc kh : list) {

Object[] row = {

kh.getMaKH(),

kh.getMaCD(),

kh.getThoiLuong(),

kh.getHocPhi(),

dateHelper.*toString*(kh.getNgayKG()),

kh.getMaNV(),

dateHelper.*toString*(kh.getNgayTao())

};

model.addRow(row);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

/\*

lấy thông tin trên form cho vào đt khoaHoc

thêm đt khoaHoc vào CSDL, load lại bảng

xóa trắng form, để ở insertable

MaKH là tự sinh từ 1 đến ... trong CSDL ko cần nhập

mã tự sinh này là cố định không đổi kể cả khi bản ghi phía trước bị xóa làm stt thay đổi

\*/

void **insert**() {

khoaHoc model = getModel();

model.setNgayTao(dateHelper.*now*());

try {

dao.insert(model);

this.load();

this.clear();

dialogHelper.*alert*(this, "Thêm mới thành công!");

} catch (HeadlessException e) {

dialogHelper.*alert*(this, "Thêm mới thất bại!");

}

}

//lấy thông tin trên form cho vào đt khoaHoc

//cập nhật bản ghi trong CSDL theo maKH và thông tin khác từ đt khoaHoc

//load lại bảng

void **update**() {

khoaHoc model = getModel();

try {

dao.update(model);

this.load();

dialogHelper.*alert*(this, "Cập nhật thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Cập nhật thất bại!");

}

}

//xóa bản ghi trong CSDL theo maKH lấu trên form

//load lại bảng, xóa trắng form, chuyển sang insertable

void **delete**() {

if (dialogHelper.*confirm*(this, "Bạn thực sự muốn xóa khóa học này?")) {

Integer makh = Integer.*valueOf*(cboChuyenDe.getToolTipText()); //maKH để nhờ ở toolTipText

try {

dao.delete(makh);

this.load();

this.clear();

dialogHelper.*alert*(this, "Xóa thành công!");

} catch (Exception e) {

dialogHelper.*alert*(this, "Xóa thất bại!");

}

}

}

public void **setTrang**(){

txtNgayKG.setBackground(*white*);

}

//xóa trắng form, chuyển sang insertable

void **clear**() {

setTrang();

khoaHoc model = new khoaHoc();

chuyenDe chuyenDe = (chuyenDe) cboChuyenDe.getSelectedItem();//lấy chuyenDe đang đc chọn ở combobox

cboChuyenDe.setToolTipText(""); //tự viết thêm, ko cần thiết

model.setMaCD(chuyenDe.getMaCD());

model.setMaNV(shareHelper.*USER*.getMaNV()); //người tạo là nhanVien đang đăng nhập

model.setNgayKG(dateHelper.*add*(30)); //ngày khai giảng sau ngày tạo 30 ngày

model.setNgayTao(dateHelper.*now*()); //ngày tạo là ngày hiện tại

this.setModel(model);

setStatus(true); //tự thêm

}

//lấy maKH từ bảng theo index, lấy đt khoaHoc từ CSDL theo maKH

// đưa thông tin từ đt khoaHoc lên form, chuyển sang editable

void **edit**() {

setTrang();

try {

Integer makh = (Integer) tblGridView.getValueAt(this.index, 0);

khoaHoc model = dao.findById(makh);

if (model != null) {

this.setModel(model);

this.setStatus(false);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

//điền thông tin từ đt khoaHoc vào form

void **setModel**(khoaHoc model) {

cboChuyenDe.setToolTipText(String.*valueOf*(model.getMaKH()));

//để nhờ maKH ở tooltipText vì ko có chỗ để trên form

//vì maKH này trong CSDL là tự sinh từ 1 đến hết ko cần nhập trên form

cboChuyenDe.getModel().setSelectedItem(cddao.findById(model.getMaCD()));

//lưu ý thêm getModel() khi áp dụng với đối tượng, ko cần thêm khi dùng với String VD: cbo.setSelectedItem("Item A");

//tìm đt chuyenDe theo maCD rồi setSelectedItem cho combobox

txtNgayKG.setText(dateHelper.*toString*(model.getNgayKG()));

txtHocPhi.setText(String.*valueOf*(model.getHocPhi()));

txtThoiLuong.setText(String.*valueOf*(model.getThoiLuong()));

txtMaNV.setText(model.getMaNV());

txtNgayTao.setText(dateHelper.*toString*(model.getNgayTao()));

txtGhiChu.setText(model.getGhiChu());

}

//lấy đt khoaHoc từ form, return khoaHoc

khoaHoc **getModel**() {

khoaHoc model = new khoaHoc();

chuyenDe chuyenDe = (chuyenDe) cboChuyenDe.getSelectedItem();

model.setMaCD(chuyenDe.getMaCD());

model.setNgayKG(dateHelper.*toDate*(txtNgayKG.getText()));

model.setHocPhi(Double.*valueOf*(txtHocPhi.getText()));

model.setThoiLuong(Integer.*valueOf*(txtThoiLuong.getText()));

model.setGhiChu(txtGhiChu.getText());

model.setMaNV(shareHelper.*USER*.getMaNV());

model.setNgayTao(dateHelper.*toDate*(txtNgayTao.getText()));

model.setMaKH(Integer.*valueOf*(cboChuyenDe.getToolTipText()));

return model;

}

//2 chế độ như các form trước

//nút btnStudents chỉ hiển thị ở chế độ editable

void **setStatus**(boolean insertable) {

btnInsert.setEnabled(insertable);

btnUpdate.setEnabled(!insertable);

btnDelete.setEnabled(!insertable);

boolean first = this.index > 0;

boolean last = this.index < tblGridView.getRowCount() - 1;

btnFirst.setEnabled(!insertable && first);

btnPrev.setEnabled(!insertable && first);

btnLast.setEnabled(!insertable && last);

btnNext.setEnabled(!insertable && last);

btnStudents.setVisible(!insertable); //chỉ nhìn thấy nút này ở chế độ Editable

}

//thay đổi thời lượng vào học phí trên form theo chuyên đề đc chọn ở combobox

void **selectComboBox**() {

chuyenDe chuyenDe = (chuyenDe) cboChuyenDe.getSelectedItem();

//lấy 1 Object được chọn từ combobox

//có thể điền và lấy 1 Object từ combobox

txtThoiLuong.setText(String.*valueOf*(chuyenDe.getThoiLuong()));

txtHocPhi.setText(String.*valueOf*(chuyenDe.getHocPhi()));

}

//mở hocVienJFrame với tham số là maKH

void **openHocVien**() {

Integer id = Integer.*valueOf*(cboChuyenDe.getToolTipText());

new hocVienJFrame(id).setVisible(true);

}

//

void **fillComboBox**() {

DefaultComboBoxModel model = (DefaultComboBoxModel) cboChuyenDe.getModel(); //kết nối model với cbo

model.removeAllElements(); //xóa toàn bộ item của cbo

try {

List<chuyenDe> list = cddao.select();

for (chuyenDe cd : list) {

model.addElement(cd); //thêm đối tượng (Object) vào model

//chỉ thêm đc đối tượng đối với model, cbo chỉ được cbo.addItem(String);

//lấy đối tượng thì từ cbo cũng được: cbo.getSelectedItem();

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn dữ liệu!");

}

}

public boolean **check5Ngay**(JTextField txt, JTextField txt2) {

txt.setBackground(*white*);

Date date = dateHelper.*toDate*(txt.getText());

Date date2 = dateHelper.*toDate*(txt2.getText());

Calendar c1 = Calendar.*getInstance*();

Calendar c2 = Calendar.*getInstance*();

c1.setTime(date);

c2.setTime(date2);

long a = (c1.getTime().getTime() - c2.getTime().getTime()) / (24 \* 3600 \* 1000);

if (a >= 5) {

return true;

} else {

txt.setBackground(*pink*);

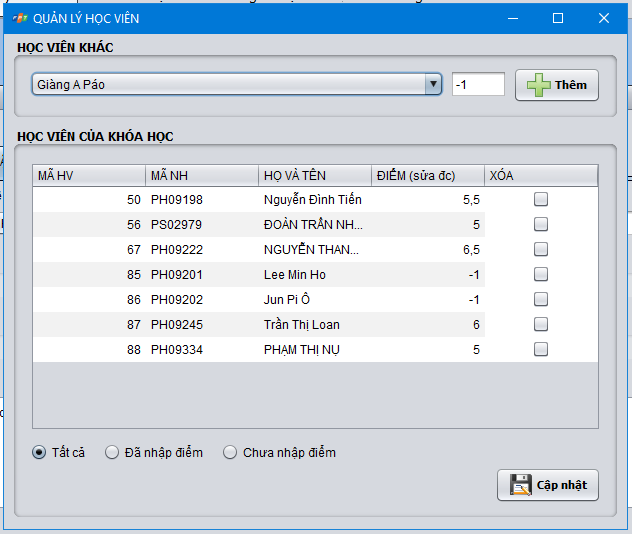
dialogHelper.*alert*(this, txt.getName() + " phải sau cách ngày tạo ít nhất 5 ngày.");

return false;

}

}

### Form học viên (hocVienJInternalFrame)



|  |  |
| --- | --- |
| Sự kiện | Công việc |
| Constructor | init()  this.MaKH = MaKH; |
| JinternalFrameOpened | this.fillComboBox();  this.fillGridView(); |
| btnThem | insert() |
| cboTatCa  cboDaNhap  cboChuaNhap | this.fillGridView() |
| btnCapNhat | Update(); |

public Integer MaKH; //maKH được chọn nhập từ constructor

hocVienDAO dao = new hocVienDAO();

nguoiHocDAO nhdao = new nguoiHocDAO();

void **init**() {

setIconImage(shareHelper.*APP\_ICON*);

setLocationRelativeTo(null);

}

//lấy tất cả đối tượng nguoiHoc không thuộc khoaHoc từ CSDL (theo maKH)

//rồi thêm vào combobox

void **fillComboBox**() {

DefaultComboBoxModel model = (DefaultComboBoxModel) cboNguoiHoc.getModel(); //kết nối cbo với model

model.removeAllElements(); //xóa toàn bộ item

try {

//lấy tất cả đối tượng nguoiHoc không thuộc khoaHoc từ CSDL

//rồi thêm vào combobox

List<nguoiHoc> list = nhdao.selectByCourse(MaKH);

for (nguoiHoc nh : list) {

model.addElement(nh);

}

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi truy vấn học viên!");

}

}

//lấy về MaHV, MaKH, MaNH, Diem, HoTen từ các bảng trong CSDL của các học viên thuộc khóa học (theo maKH)

//điền các bản ghi tương ứng vào bảng theo: tất cả, chưa nhập điểm, đã nhập điểm

void **fillGridView**() {

DefaultTableModel model = (DefaultTableModel) tblGridView.getModel();

model.setRowCount(0);

ResultSet rs = null;

try {

//lấy về MaHV, MaKH, MaNH, Diem, HoTen từ các bảng trong CSDL của các học viên thuộc

//khóa học (theo maKH)

String sql = "SELECT hv.\*, nh.HoTen FROM hocVien hv "

+ " JOIN nguoiHoc nh ON nh.MaNH=hv.MaNH WHERE MaKH=?";

rs = jdbcHelper.*executeQuery*(sql, MaKH);

while (rs.next()) {

double diem = rs.getDouble("Diem");

Object[] row = {

rs.getInt("MaHV"), rs.getString("MaNH"),

rs.getString("HoTen"), diem, false

};

if (rdoTatCa.isSelected()) { //tất cả thì add tất cả bản ghi vào

model.addRow(row);

} else if (rdoDaNhap.isSelected() && diem >= 0) {//đã nhập thì chỉ add bản ghi điểm 0-10

model.addRow(row);

} else if (rdoChuaNhap.isSelected() && diem < 0) {//chưa nhập thì chỉ nhập bản ghi điểm -1

model.addRow(row);

}

}

} catch (SQLException e) {

dialogHelper.*alert*(this, "Lỗi truy vấn học viên!");

} finally {

try {

rs.getStatement().getConnection().close();

} catch (SQLException ex) {

throw new RuntimeException();

}

}

}

/\*

thêm đt hocVien vào CSDL bảng hocVien

load lại combobox, bảng

MaHV là tự sinh từ 1 đến ... trong CSDL ko cần nhập

mã tự sinh này là cố định không đổi kể cả khi bản ghi phía trước bị xóa làm stt thay đổi

\*/

void **insert**() {

nguoiHoc nguoiHoc = (nguoiHoc) cboNguoiHoc.getSelectedItem(); //lấy đt nguoiHoc từ combobox

hocVien model = new hocVien(); //tạo đt hocVien

model.setMaKH(MaKH);

model.setMaNH(nguoiHoc.getMaNH());

model.setDiem(Double.*valueOf*(txtDiem.getText()));

try {

dao.insert(model); //thêm đt hocVien vào CSDL bảng hocVien

this.fillComboBox(); //load lại combobox

this.fillGridView(); //load lại bảng

txtDiem.setText("-1");

} catch (Exception e) {

dialogHelper.*alert*(this, "Lỗi thêm học viên vào khóa học!");

}

}

//chỉ sửa điểm hocVien, xóa hocVien bị tích

//cập nhật vào CSDL, load lại bảng, load lại cbo

void **update**() {

txtDiem.setBackground(*white*);

int a = 0,b=0;

for (int i = 0; i < tblGridView.getRowCount(); i++) {

Integer mahv = (Integer) tblGridView.getValueAt(i, 0); //lấy maHV từ bảng(ko sửa đc)

String manh = (String) tblGridView.getValueAt(i, 1); //lấy maNH từ bảng(ko sửa đc)

Double diem = (Double) tblGridView.getValueAt(i, 3); //lấy điểm (sửa đc)

Boolean isDelete = (Boolean) tblGridView.getValueAt(i, 4);

if (isDelete) {

a++;

}

if (isDelete && shareHelper.*USER*.isVaiTro()) { //nếu có tích thì xóa bản ghi đó đi

dao.delete(mahv);

} else { //còn ko tích thì cập

if(shareHelper.*USER*.isVaiTro()==false)tblGridView.setValueAt(false, i, 3);

if ((diem >= 0 && diem <= 10) || diem == -1) {

hocVien model = new hocVien();

model.setMaHV(mahv);

model.setMaKH(MaKH);

model.setMaNH(manh);

model.setDiem(diem);

dao.update(model);

}else{

b++;

}

}

}

this.fillComboBox();

this.fillGridView();

if (a > 0 && shareHelper.*USER*.isVaiTro() == false) {

dialogHelper.*alert*(this, "Chỉ trưởng phòng mới được xóa học viên**\n**bạn chỉ được thêm học viên và điểm");

return;

}

if(b>0){

dialogHelper.*alert*(this, "Điểm phải là số thực từ 0-10 hoặc chưa nhập (-1)");

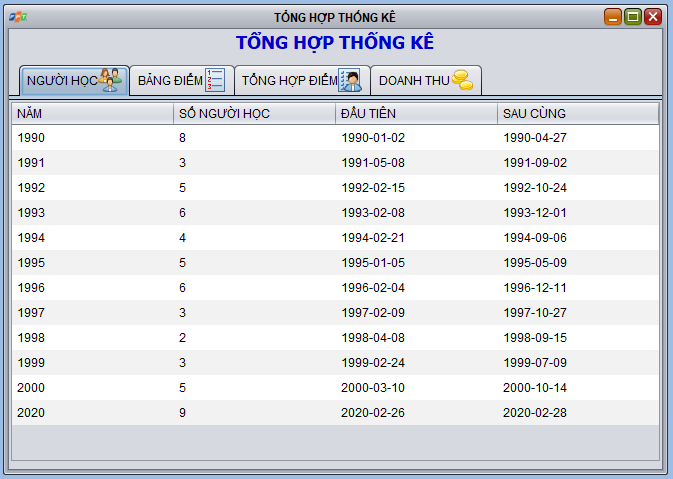
return;

}

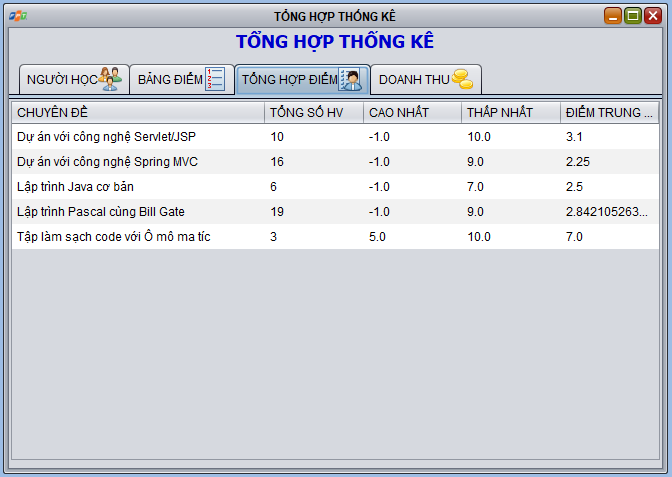
dialogHelper.*alert*(this, "Cập nhật thành công!");

}

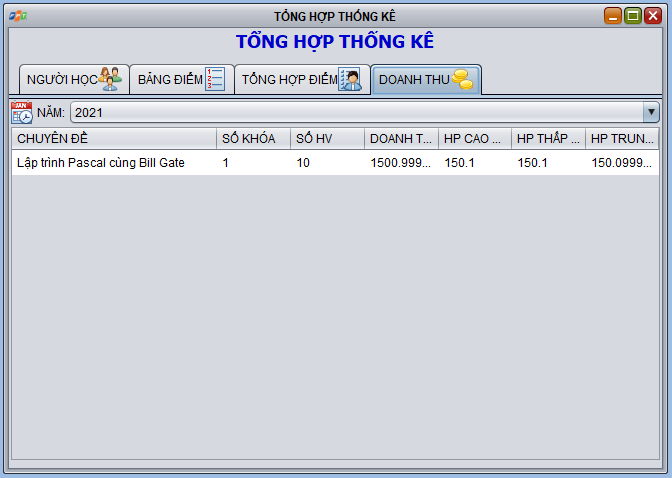
### Form thống kê (thongKeJInternalFrame)











|  |  |
| --- | --- |
| Sự kiện | Công việc |
| Constructor | tabs.setSelectedIndex(index);  init(); |
| JinternalFrameOpened | fillComboBoxKhoaHoc();  fillTableBangDiem();  fillTableNguoiHoc();  fillTableKhoaHoc();  fillComboBoxNam();  fillTableDoanhThu(); |
| cboNam | fillTableDoanhThu(); |
| cboKhoaHoc | fillTableBangDiem(); |

thongKeDAO dao = new thongKeDAO();

khoaHocDAO khdao = new khoaHocDAO();

void **init**() {

setFrameIcon(shareHelper.*APP\_ICON\_1*);

CardLayout card= (CardLayout) pnlDoanhThu.getLayout();

if(shareHelper.*USER*.isVaiTro()){

card.show(pnlDoanhThu, "card1");

}else{

card.show(pnlDoanhThu, "card2");

}

}

//xóa cbo, lấy tất cả đt khoaHoc từ CSDL thêm mới vào cbo

//chọn sẵn Item thứ nhất

void **fillComboBoxkhoaHoc**() {

DefaultComboBoxModel model = (DefaultComboBoxModel) cboKhoaHoc.getModel(); //kết nối cbo với model

model.removeAllElements(); //xóa tất cả item

List<khoaHoc> list = khdao.select();

for (khoaHoc kh : list) {

model.addElement(kh);

}

cboKhoaHoc.setSelectedIndex(0);

}

//lấy tất cả năm của khóa học (int) điền vào cbo (điền đt int), ko điền trùng

//chọn sẵn Item thứ nhất

void **fillComboBoxNam**() {

DefaultComboBoxModel model = (DefaultComboBoxModel) cboNam.getModel();

model.removeAllElements();

// List<khoaHoc> list = khdao.select();

// for (khoaHoc kh : list) {

// int nam = kh.getNgayKG().getYear() + 1900; //date.getYear() trả về int năm của date - 1900

// if (model.getIndexOf(nam) < 0) { //kiểm tra xem trong cbo đã có năm này chưa, nếu chưa có thì mới thêm năm

// //model.getIndexOf(Object) trả về vị trí của object trong cbo, nêú chưa có trả về -1

// model.addElement(nam);

// //model.getElementAt(2); //chẳng để làm gì

// }

// }

List<Integer> list=dao.getNamKhaiGiang();

for(Integer nam: list){

model.addElement(nam);

}

cboNam.setSelectedIndex(0);

}

//xóa bảng điểm, điền dữ liệu vào bảng điểm theo MaKH

void **fillTableBangDiem**() {

DefaultTableModel model = (DefaultTableModel) tblBangDiem.getModel();

model.setRowCount(0);

khoaHoc kh = (khoaHoc) cboKhoaHoc.getSelectedItem();

List<Object[]> list = dao.getBangDiem(kh.getMaKH()); //lấy về 1 <Object[]> list theo MaKH

for (Object[] row : list) {

model.addRow(row);

}

}

//xóa bảng người học, đièm dữ liệu vào bảng người học

void **fillTableNguoiHoc**() {

DefaultTableModel model = (DefaultTableModel) tblNguoiHoc.getModel();

model.setRowCount(0);

List<Object[]> list = dao.getNguoiHoc();

for (Object[] row : list) {

model.addRow(row);

}

}

//xóa bảng tổng hợp điểm, điền dữ liệu vào bảng tổng hợp điểm

void **fillTablekhoaHoc**() {

DefaultTableModel model = (DefaultTableModel) tblKhoaHoc.getModel();

model.setRowCount(0);

List<Object[]> list = dao.getDiemTheoChuyenDe();

for (Object[] row : list) {

model.addRow(row);

}

}

//xóa bảng doanh thu, điền dữ liệu vào bảng doanh thu theo năm tương ứng

void **fillTableDoanhThu**() {

DefaultTableModel model = (DefaultTableModel) tblDoanhThu.getModel();

model.setRowCount(0);

int nam = Integer.*parseInt*(cboNam.getSelectedItem().toString());

List<Object[]> list = dao.getDoanhThu(nam);

for (Object[] row : list) {

model.addRow(row);

}

}

# Kiểm thử

## UtilityHelper

Xem lại phần [3.2.1.5 Lớp tiện ích utilityHelper](#_Lớp_tiện_ích)

## Kiểm thử form nhân viên

* check trống: mã nhân viên, mật khẩu, xác nhận mật khẩu, họ và tên
* check mã nhân viên: 1-15 kí tự, a-z, A-Z, 0-9
* check mật khẩu: từ 3-16 kí tự
* check xác nhận mật khẩu: trùng pass, từ 3-16 kí tự
* check họ và tên: 3-25 kí tự, chữ tiếng anh, việt và dấu cách
* không được xóa chính mình
* chỉ trưởng phòng mới đc xóa
* các hàm gọi từ utilityHelper và viết thêm hàm sau:

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

return false;

}

}

public boolean **checkChinhMinh**(JTextField txt){

nhanVien nv=dao.findById(txt.getText());

if (nv.getMaNV().equals(*USER*.getMaNV())) {

dialogHelper.*alert*(this, "bạn không được xóa chính mình.");

return false;

} else {

return true;

}

}

## KIỂM THỬ FORM NGƯỜI HỌC

* check trống: Mã người học, họ và tên, ngày sinh, điện thoại, email
* check mã người học: đúng 7 kí tự, a-z, A-Z, 0-9
* check họ tên: 3-25 kí tự, chữ tiếng anh, việt và dấu cách
* check ngày sinh: dd/MM/yyy, cách ngày hiện tại 16 năm
* check điện thoại: 10 số, đầu số chính xác của nhà mạng
* check email: đúng định dạng email
* chỉ trưởng phòng mới được xóa
* các hàm gọi từ utilityHelper và viết thêm hàm sau:

public boolean **check16Nam**(JTextField txt) {

txt.setBackground(*white*);

Date date = dateHelper.*toDate*(txt.getText());

Calendar c1 = Calendar.*getInstance*();

Calendar c2 = Calendar.*getInstance*();

c1.setTime(date);

c2.setTime(new Date());

long a = (c2.getTime().getTime() - c1.getTime().getTime()) / (24 \* 3600 \* 1000);

if (a >= 5844) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " phải cách đây ít nhất 16 năm.");

return false;

}

}

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

return false;

}

}

## KIỂM THỬ FORM CHUYÊN ĐỀ

* check trống: mã chuyên đề, tên chuyên đề, thời lượng, học phí, ghi chú, hình ảnh
* check mã chuyên đề: đúng 5 kí tự, a-z, A-Z, 0-9
* check tên chuyên đề: từ 3-50 kí tự
* check thời lượng: int >=0
* check học phí: float >=0
* chỉ trưởng phòng mới được xóa
* các hàm gọi từ utilityHelper và viết thêm hàm sau:

public boolean **checkNullHinh**(){

if(lblHinh.getToolTipText()!=null){

return true;

}else{

dialogHelper.*alert*(this, "Không được để trống hình.");

return false;

}

}

public boolean **checkTrungMa**(JTextField txt) {

txt.setBackground(*white*);

if (dao.findById(txt.getText()) == null) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " đã bị tồn tại.");

return false;

}

}

## KIỂM THỬ FORM KHÓA HỌC

* check trống: ngày khai giảng
* check ngày khai giảng định dạng dd/MM/yy
* ngày khai giảng cách ngày tạo ít nhất 5 ngày
* chỉ trưởng phòng mới được xóa
* các hàm gọi từ utilityHelper và viết thêm hàm sau:

public boolean **check5Ngay**(JTextField txt, JTextField txt2) {

txt.setBackground(*white*);

Date date = dateHelper.*toDate*(txt.getText());

Date date2 = dateHelper.*toDate*(txt2.getText());

Calendar c1 = Calendar.*getInstance*();

Calendar c2 = Calendar.*getInstance*();

c1.setTime(date);

c2.setTime(date2);

long a = (c1.getTime().getTime() - c2.getTime().getTime()) / (24 \* 3600 \* 1000);

if (a >= 5) {

return true;

} else {

txt.setBackground(*pink*);

dialogHelper.*alert*(this, txt.getName() + " phải sau cách ngày tạo ít nhất 5 ngày.");

return false;

}

}

## KIỂM THỬ FORM HỌC VIÊN

* điểm không được trống
* điểm từ 1-10 hoặc -1 nếu chưa nhập
* chỉ trưởng phòng mới được xóa

## KIỂM THỬ FORM THỐNG KÊ

không có gì cả

## KIỂM THỬ FORM ĐĂNG NHẬP

* tên đăng nhập, mật khẩu phải không đươc trống
* tên đăng nhập: 1-15 kí tự, a-z, A-Z, 0-9
* mật khẩu: 3-16 kí tự

## KIỂM THỬ FORM ĐỔI MẬT KHẨU

* mật khẩu, xác nhận mật khẩu không được trống
* mật khẩu 3-16 kí tự

# Đóng gói và triển khai

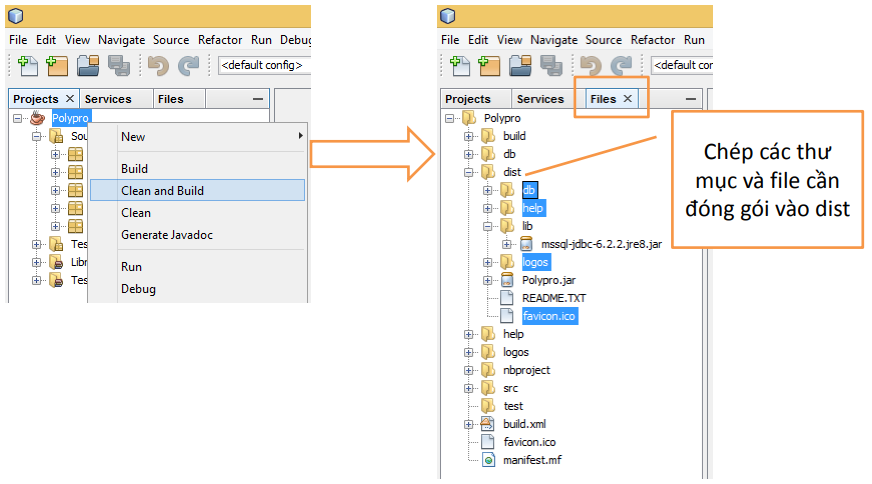
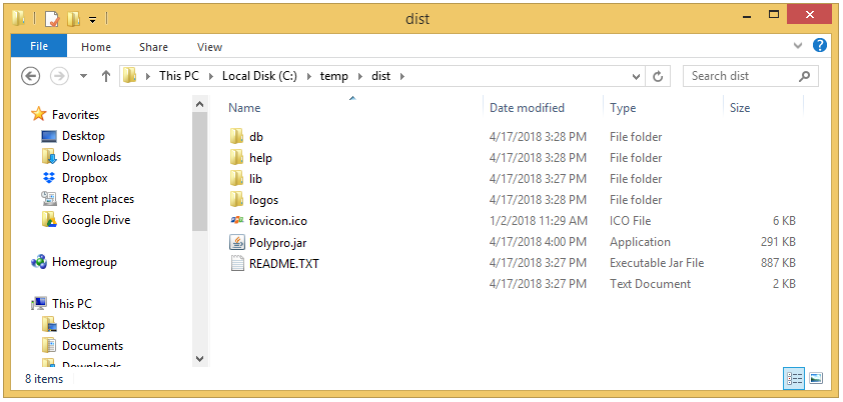
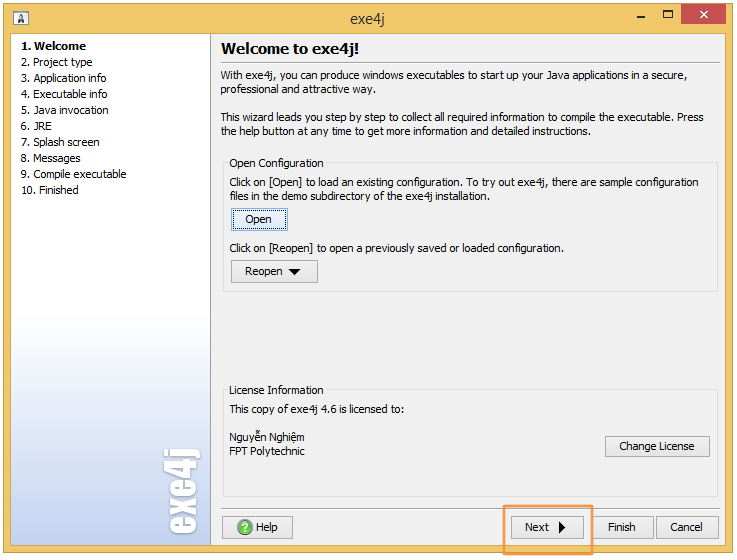
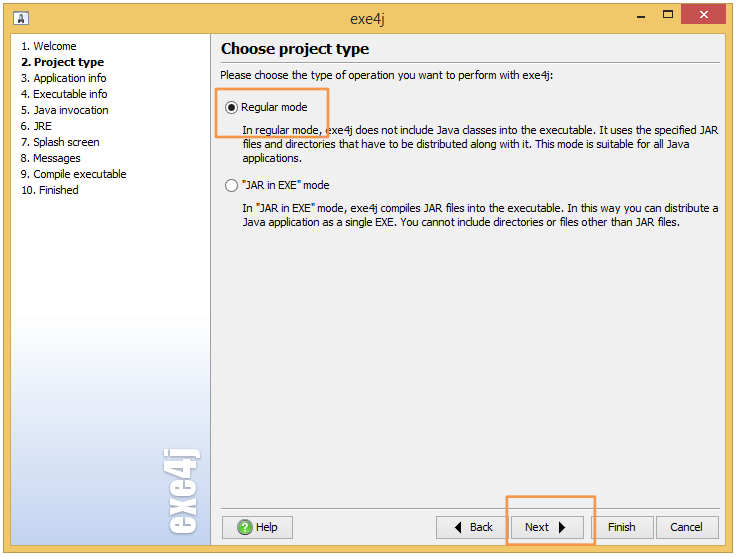
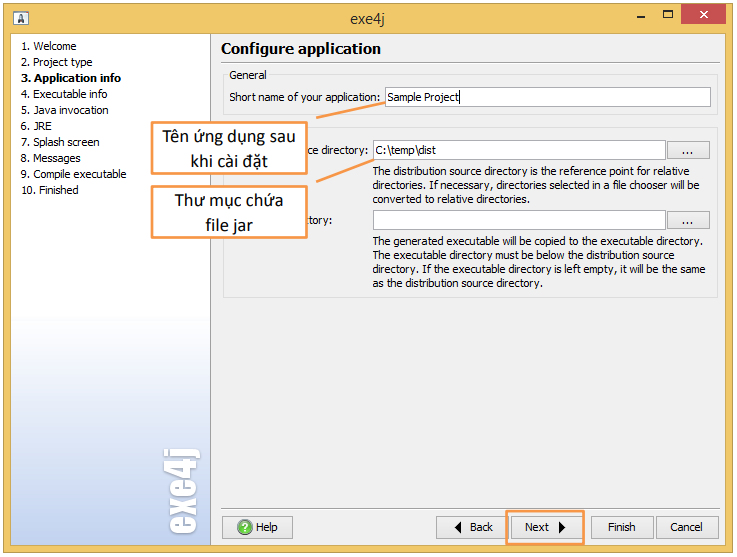
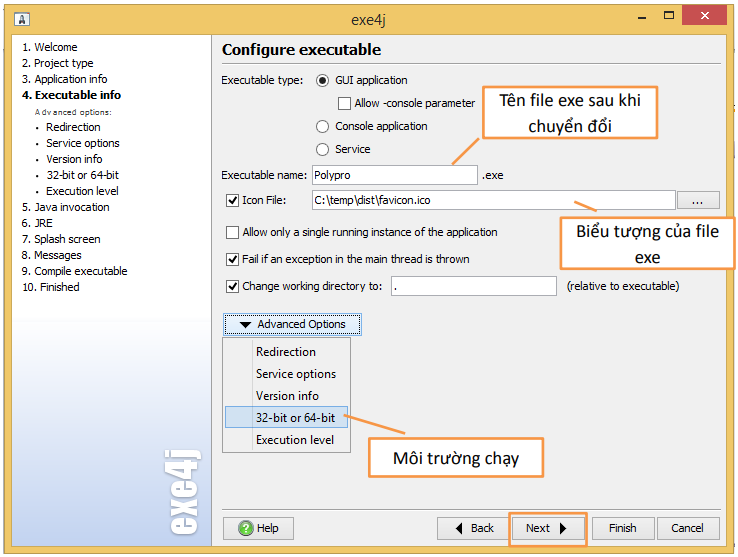
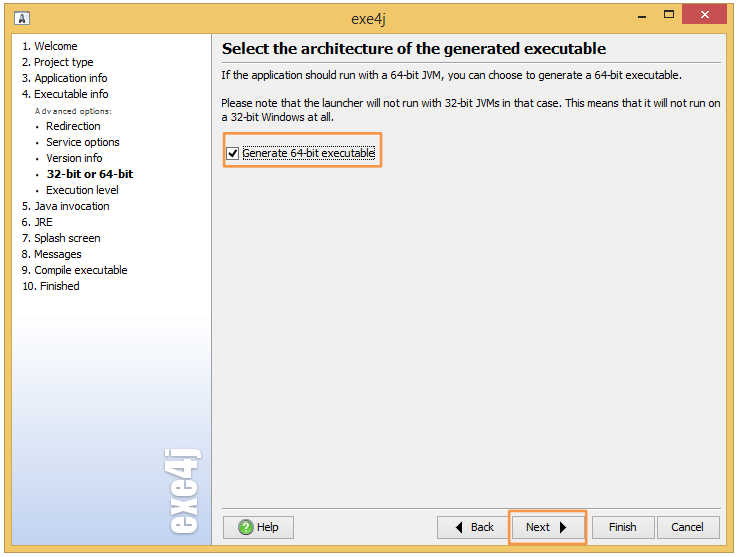
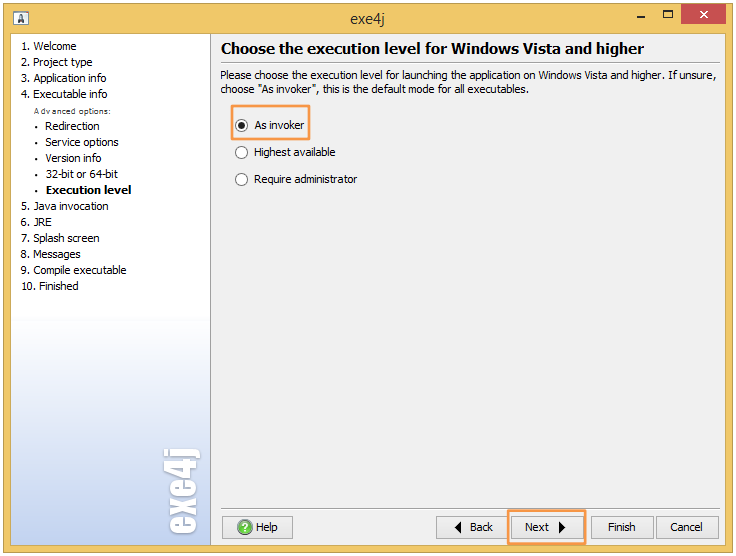
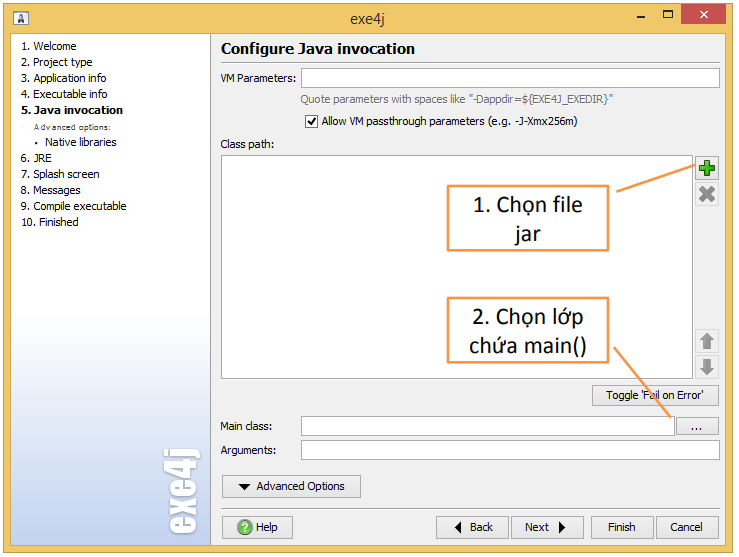
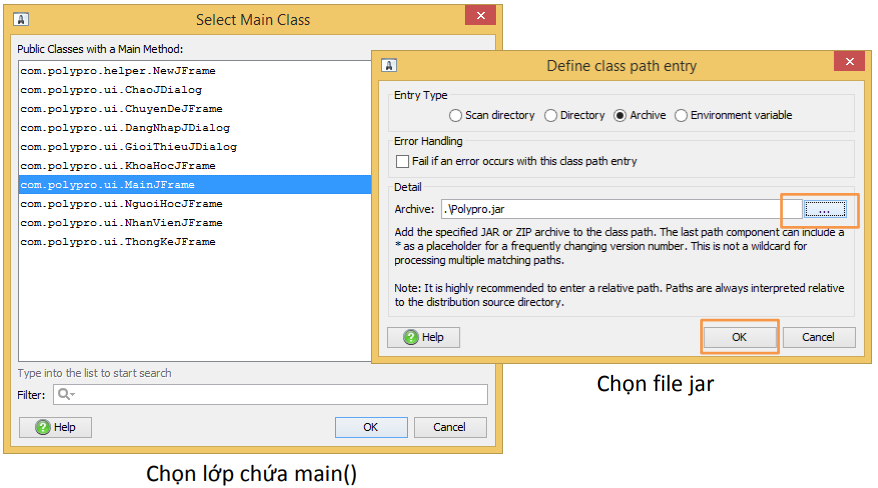
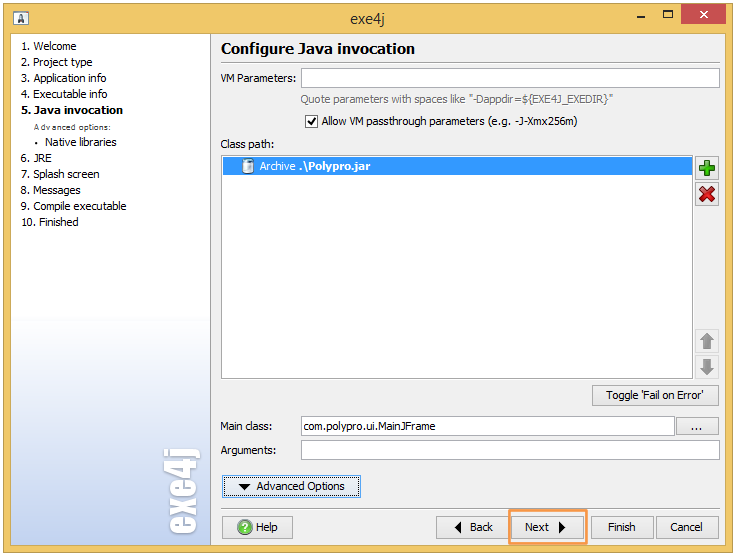
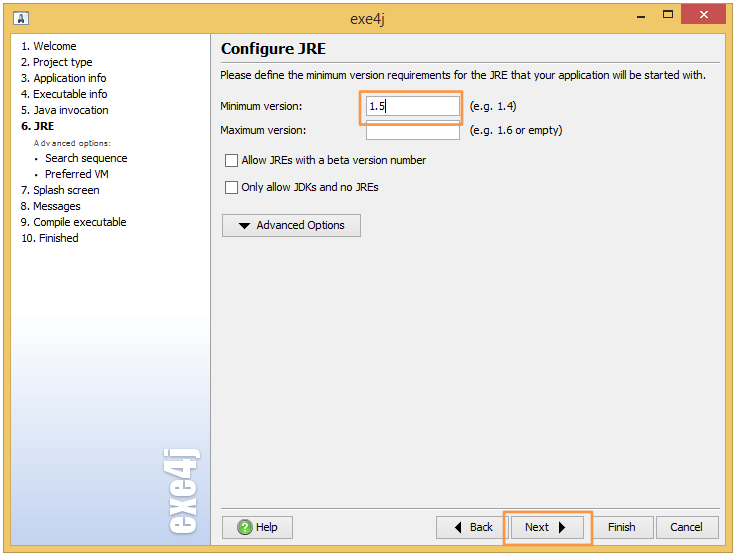
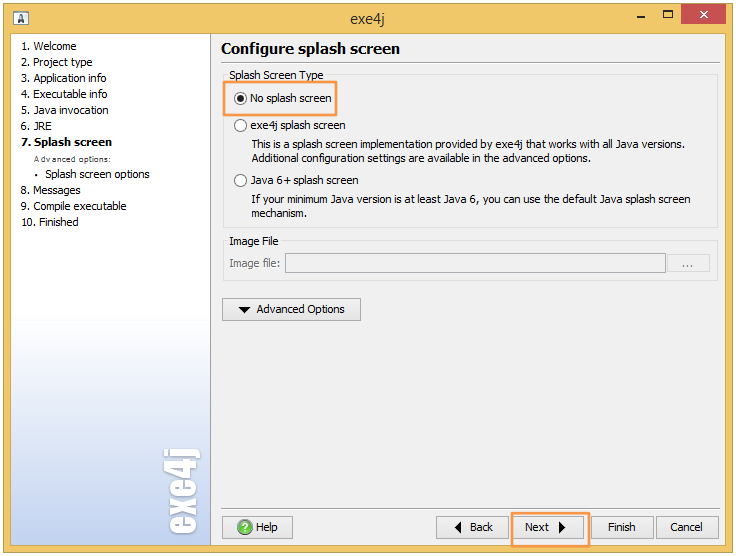
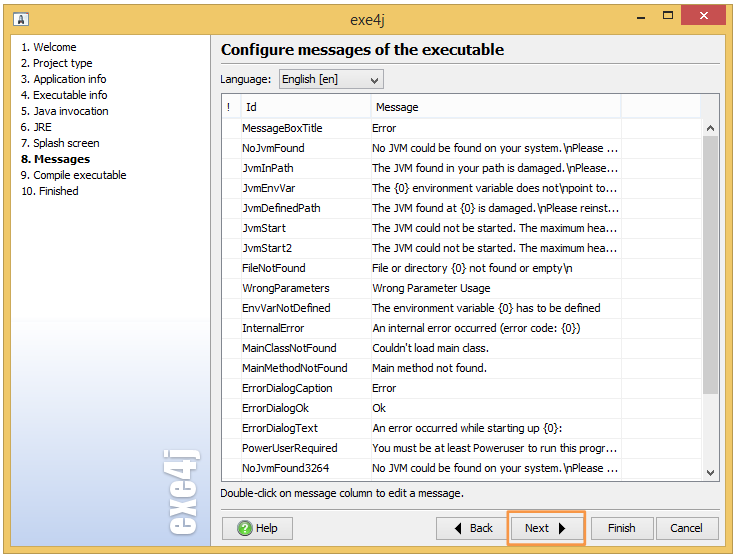
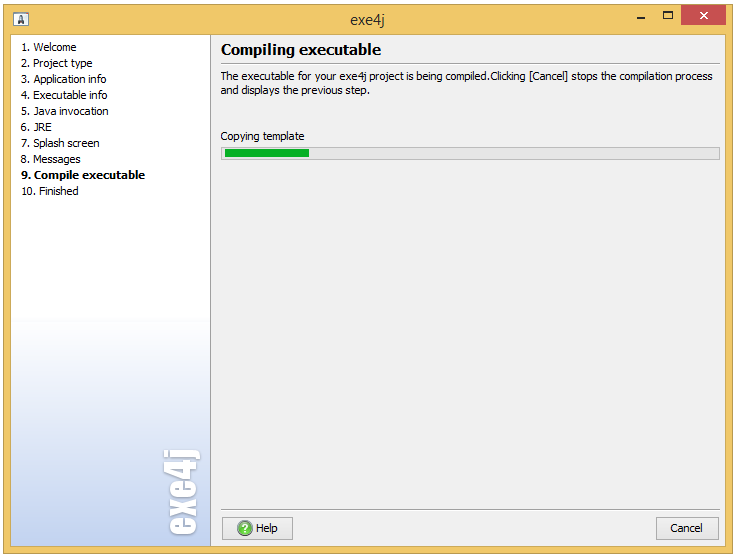
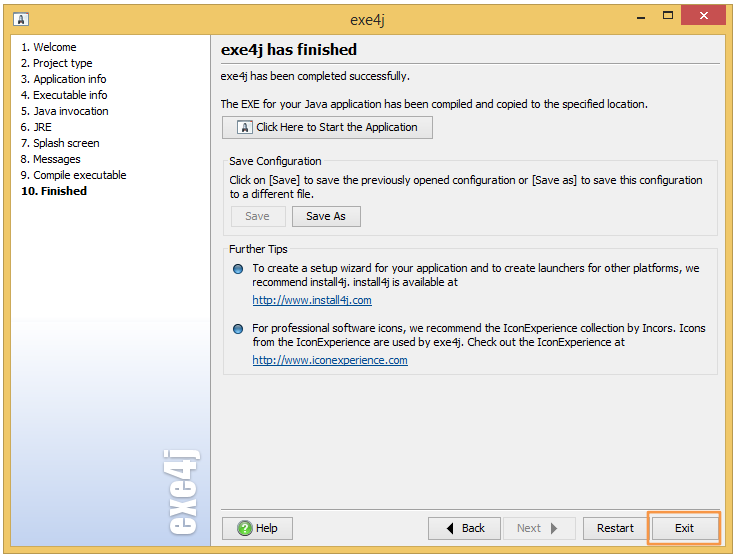
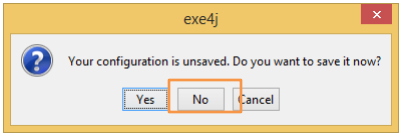
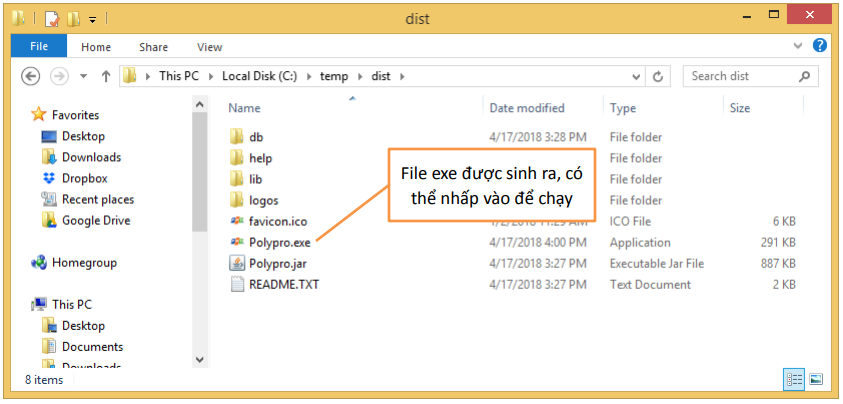
Tải phần mềm tại Exe4J:

<https://download-keycdn.ej-technologies.com/exe4j/exe4j_windows-x64_4_6.exe>

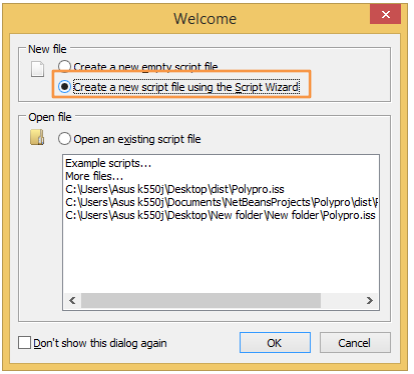
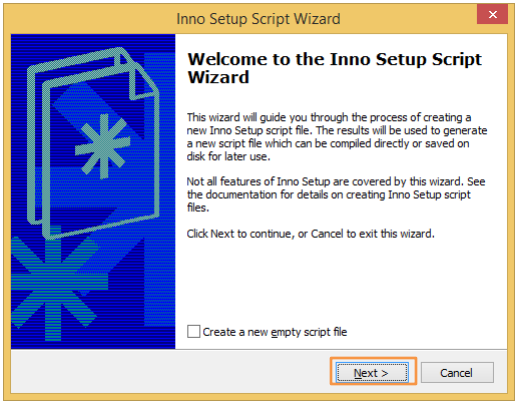
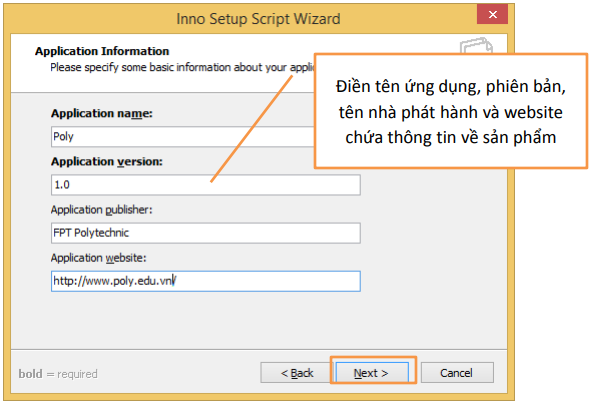
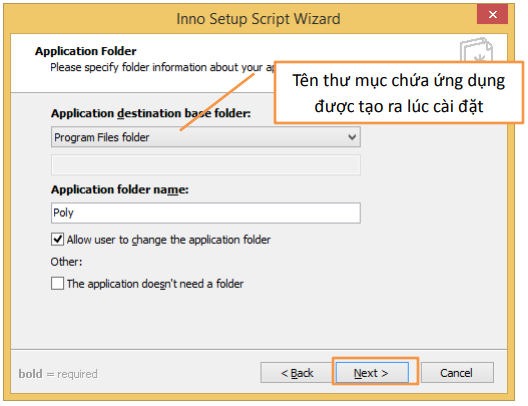
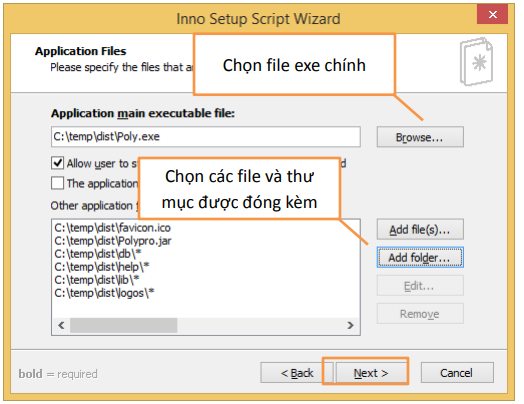
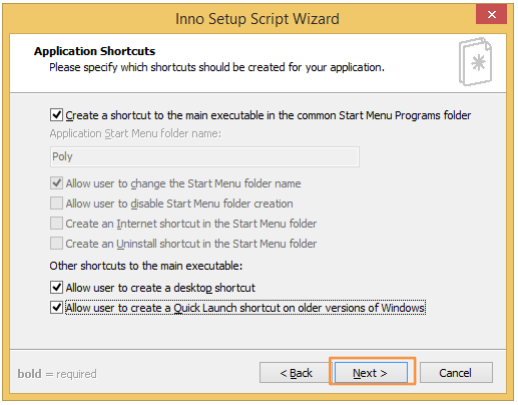
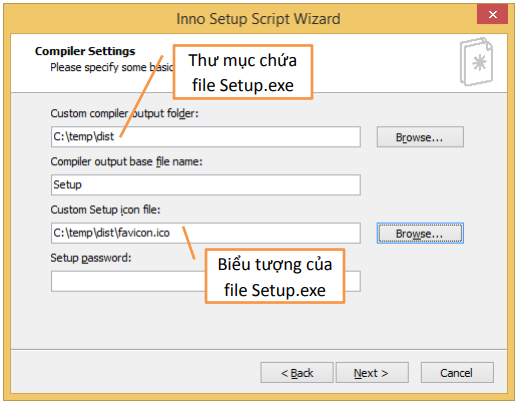
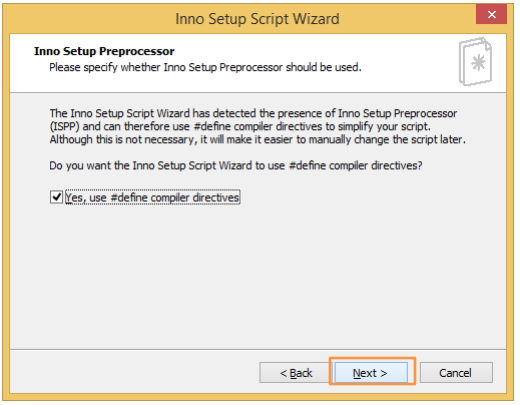
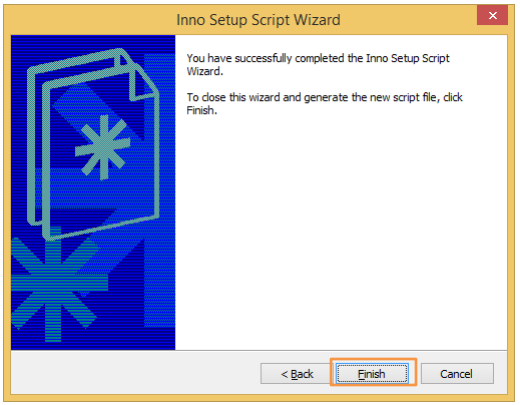
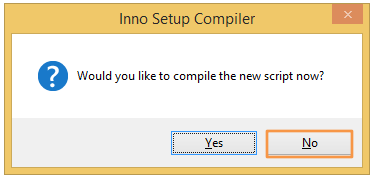
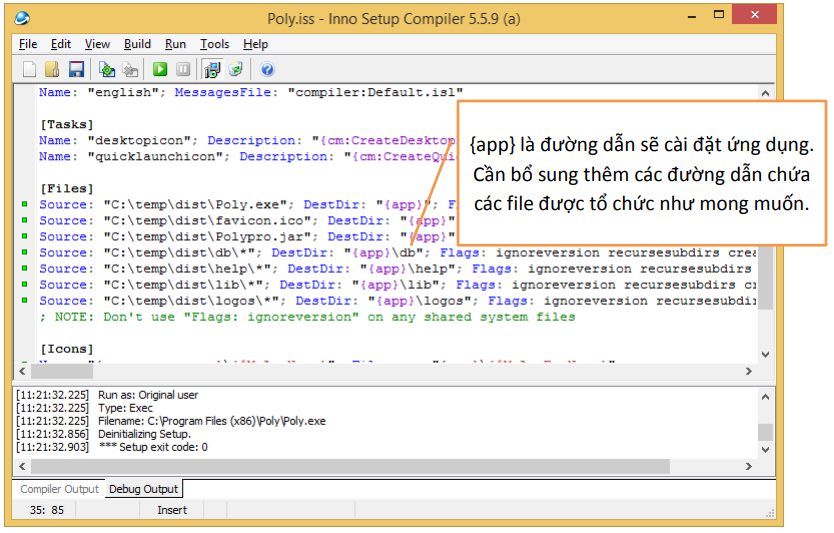
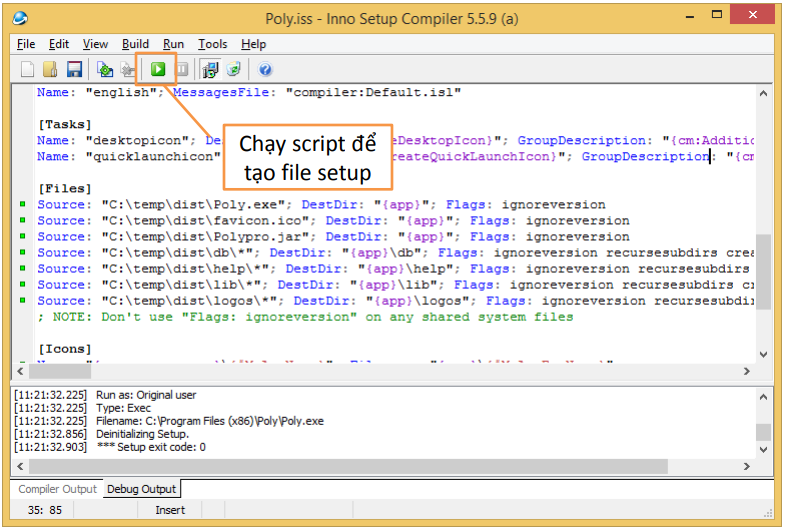
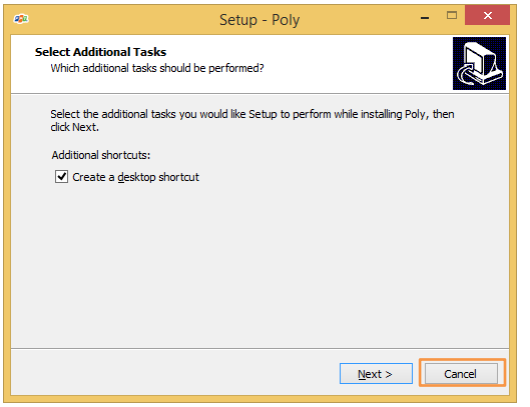
Tải phần mềm InnoSetup:

<http://files.jrsoftware.org/is/5/innosetup-5.5.9.exe>

## Hướng dẫn chuyển đổi jar thành exe

## DÙNG INNOSETUP ĐÓNG GÓI PHẦN MỀM

## Hướng dẫn cài đặt triển khai

1. Chạy Polypro.sql (hoặc minhducpro.sql) để tạo CSDL Polypro

2. Đặt mật khẩu sa của SQL Server là songlong

3. Chạy file Setup mật khẩu là: minhducpro (chú ý môi trường JDK1.8+)

4. Chạy ứng dụng: Run as Administrator

Các bạn lưu ý không để setup.exe trong trong thư mục có dấu tiếng việt (chứa trực tiếp hoặc gián tiếp) nhé, để tránh xảy ra lỗi trong quá trình cài đặt.

Link video minh họa: https://www.youtube.com/watch?v=eVKRmgzwZSc

## Hướng dẫn sử dụng phần mềm

Chạy phần mềm dưới quyền người quản trị (Run as adminstrator) để tránh bị lỗ hình ảnh form chuyên đề.

Link video minh họa: <https://www.youtube.com/watch?v=eVKRmgzwZSc>