**TRƯỜNG ĐẠI HỌC PHENIKAA**

**------------oOo-----------**



**ĐỀ TÀI: PHẦN MỀM QUẢN LÝ SINH VIÊN**

**Course/Học phần**: Lập trình hướng đối tượng

**Code/Mã học phần**: CSE703029

**Lớp/Class :** N01

**Instructor/Giảng viên**: TS. Nguyễn Lệ Thu

**Group/Nhóm : 07**

**HÀ NỘI – 2024**

**Danh sách thành viên trong nhóm**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TT** | **Họ và tên** | **Lớp -khóa- ngành** | **Email** | **Điện thoại** | **Mã sinh viên** | **Mức độ đóng góp (theo thang điểm 10)** |
| 1 | Trương Văn Diệu | K17\_KHMT\_1 | 23017208@st.phenikaa-uni.edu.vn | 0983078205 | 23017208 | 10 |
| 2 | Đinh Đức Mạnh | K17\_KHMT\_1 | 23010029@st.phenikaa- uni.edu.vn | 0368413630 | 23015270 | 10 |

**Mục lục**

1. Mô tạ nghiệp vụ

2. Thiết kế chức năng

3. Thiết kế cơ sở dữ liệu

4. Giao diện chương trình

5. Lời kết

6. Tài liệu tham khảo

# Mô tả nghiệp vụ

Hệ thống bao gồm ba tác nhân: Để sử dụng hệ thống tất cả các tác nhân cần thực hiện bước đăng nhập bằng username và password.

Bất kỳ ai cũng có thể xem danh sách học phần của các Viện.

Bất kỳ ai cũng có thể xem danh sách các lớp học tín chỉ mà một Sinh viên đã đăng ký trong một kỳ. Vào cuối mỗi kỳ học thông tin của hai trường: Nhóm và Mã lớp thi sẽ được Giáo vụ cập nhật.

## Sinh viên:

Được người quản trị cấp cho một tài khoản với username là mã số sinh viên, password mặc định trong lần đầu tiên sử dụng hệ thống. Sau khi đăng nhập thành công vào hệ thống, sinh viên có quyền đổi mật khẩu.

Vào giữa mỗi học kỳ Sinh viên thực hiện quá trình đăng ký học phần các môn học cho kỳ sắp tới bằng cách nhập mã học phần.

Sinh viên có thể xem thông tin cá nhân của mình bao gồm: Mã sinh viên, họ tên, ngày sinh, lớp, hệ học, trạng thái học. Hai thông tin: Email, số điện thoại do Sinh viên tự nhập.

Sinh viên có thể xem bảng điểm cá nhân nhưng không được phép sửa đổi. Bảng điểm cá nhân bao gồm các trường: Học kỳ, Mã HP, Tên HP, TC, Lớp học, Điểm QT, Điểm thi, Điểm chữ. Trong đó Điểm chữ được suy ra từ Điểm QT và Điểm thi phụ thuộc vào trọng số được quy định trong học phần. Cuối mỗi kỳ học, sau khi đã có đủ điểm của các môn học đã đăng ký, kết quả học tập của Sinh viên kỳ đó sẽ được tự động tính.

Sinh viên có thể xem Bảng điểm học phần: Học kỳ, Mã HP, Tên HP, TC, Điểm học phần, Thay thế HP

## Giáo vụ:

Được phép xem thông tin cá nhân, bảng điểm sinh viên, bảng điểm học phần.

Sau mỗi kỳ học giáo vụ có chức năng nhập điểm Quá trình và điểm Cuối kỳ của từng môn học cho từng Sinh viên theo lớp Tín chỉ.

Tạo báo cáo: Bảng điểm sinh viên, thống kê số lượng sinh viên mỗi lớp tín chỉ,..

## Người quản trị hệ thống:

Có thể xem thông tin về các tác nhân sử dụng hệ thống: Sinh viên, giáo vụ.

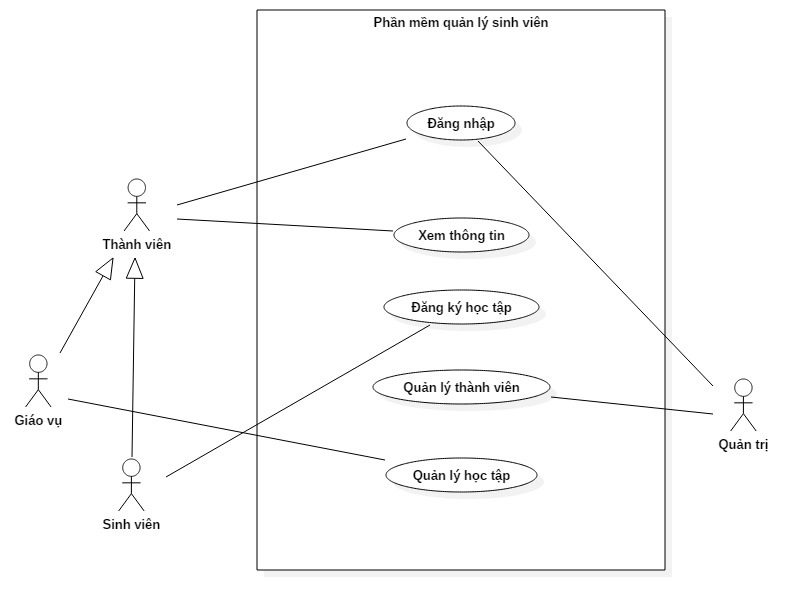
Có khả năng tạo, xóa, sửa đổi thông tin tài khoản Sinh viên, giáo vụ:

Bắt đầu mỗi năm học người quản trị có chức năng tạo mới tài khoản cho các sinh viên mới nhập học. Trong quá trình học nếu sinh viên quên mật khẩu, người quản trị có chức năng reset tài khoản sinh viên về mật khẩu mặc định. Sau khi Sinh viên tốt nghiệp, người quản trị thực hiện việc xóa tài khoản Sinh viên trên hệ thống.

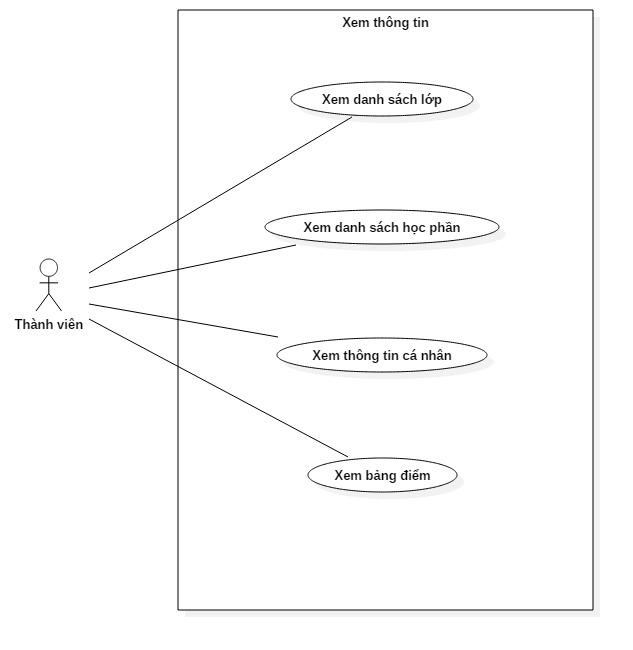
Người quản trị có khả năng cấp phát tài khoản giáo vụ cho giáo vụ mới và thu hồi lại tài khoản khi giáo vụ cũ chuyển đi.

Bảo trì hệ thống.

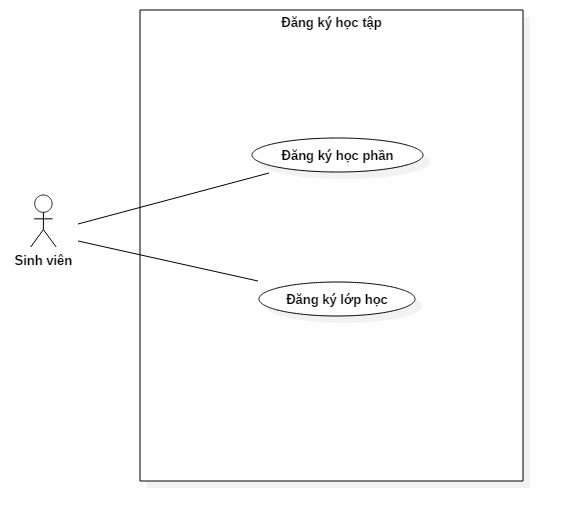
# Thiết kế chức năng

Biểu đồ usecase tổng quát

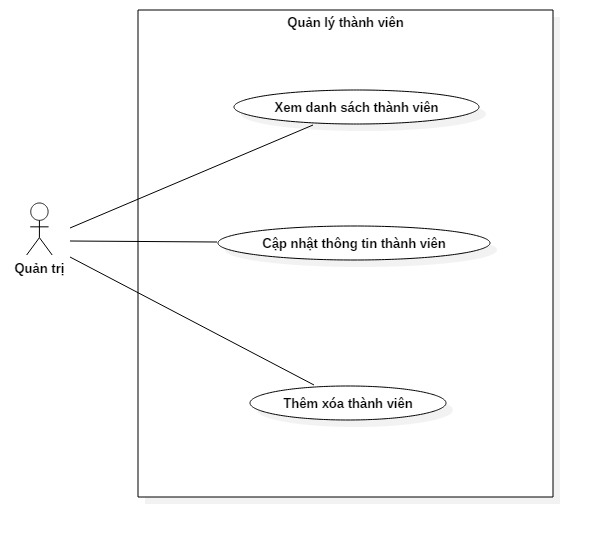
Package usecase “Xem thông tin”



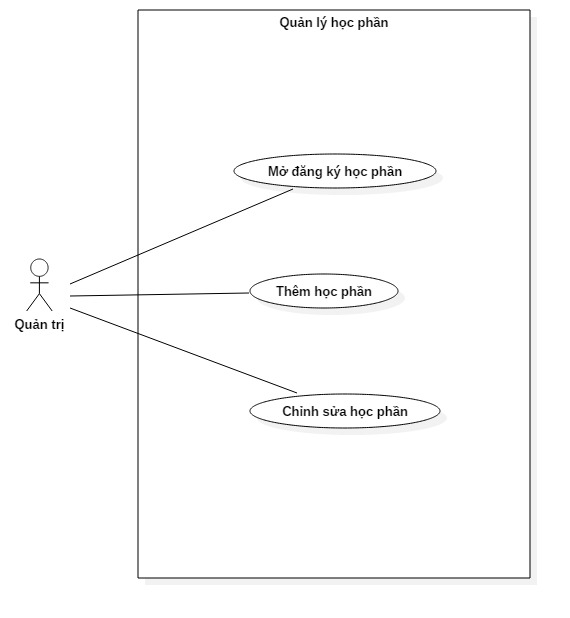
Package usecase “Đăng ký học tập”



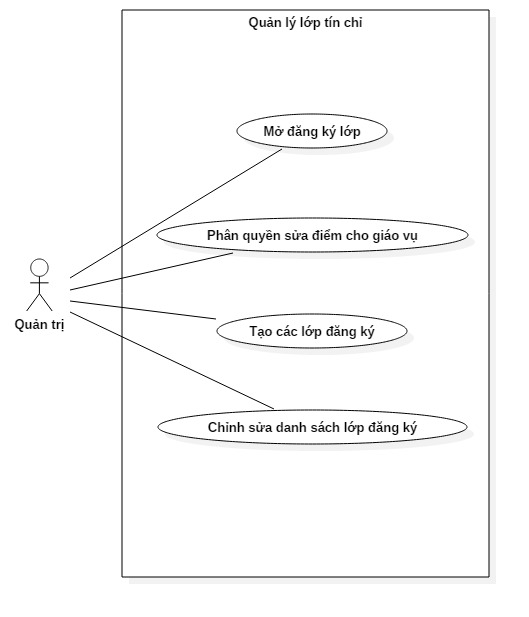
Package usecase “Quản lý thành viên”

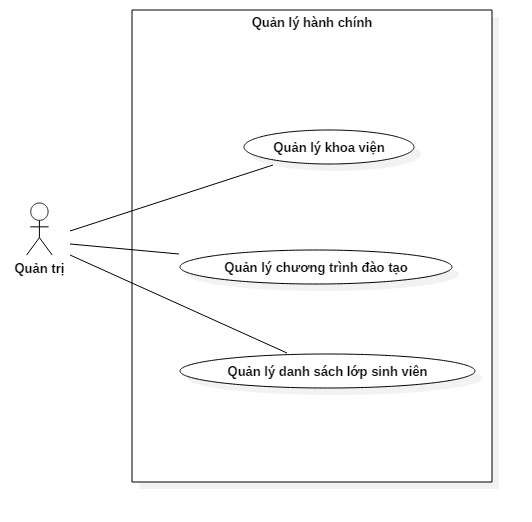


Package usecase “Quản lý học phần”



Package usecase “Quản lý lớp tín chỉ”

Package usecase “Quản lý hành chính”



# Thiết kế Cơ sở dữ liệu

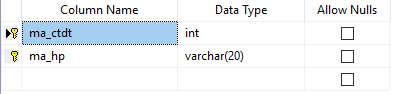
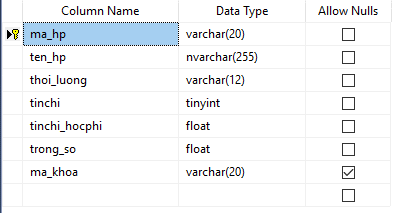
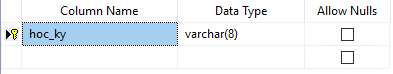
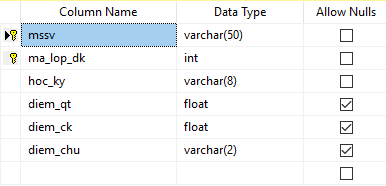
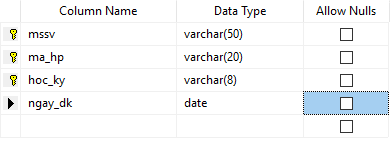
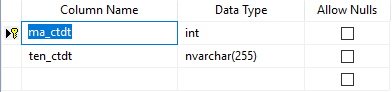
Cơ chế kết nối

Sử dụng Hibernate để kết nối ứng dụng Java tới cơ sở dữ liệu SQL Server.

**Các bảng trong cơ sở dữ liệu**

**Hình 2: DangKyLop**

**Hình 1: ChuongTrinhDT**

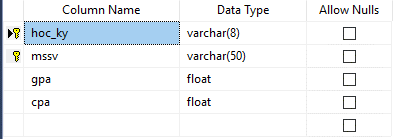


**Hình 3: HocPhan**

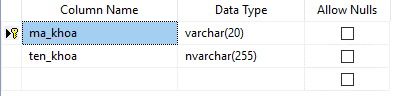
**Hình 4: DangKyHP**

**Hình 5: HocKyMoDK**

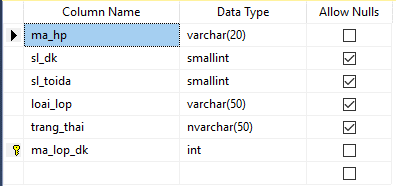
**Hình 6: HocPhanCTDT**



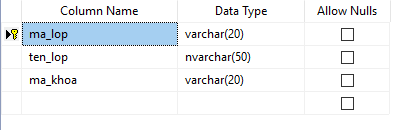
**Hình 7: KetQuaHT**



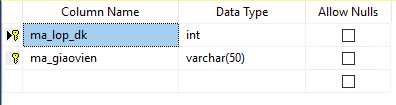
**Hình 1: KhoaVien**



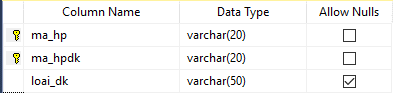
**Hình 2: LopDK**



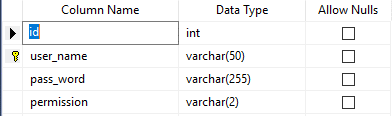
**Hình 3: LopSV**



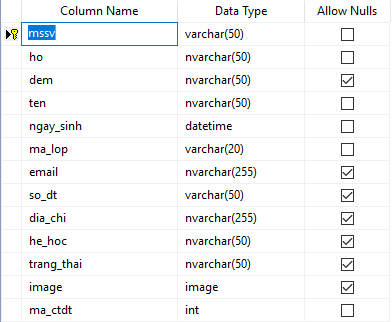
**Hình 4: PhanQuyenLopDK**



**Hình 5: RangBuocHP**



**Hình 6: Users**



**Hình 1: SinhVien**

## View

View BangdiemSV

SELECT dbo.DangKyLop.mssv, dbo.DangKyLop.ma\_lop\_dk, dbo.DangKyLop.hoc\_ky,

dbo.DangKyLop.diem\_qt, dbo.DangKyLop.diem\_ck, dbo.DangKyLop.diem\_chu,

dbo.HocPhan.ten\_hp, dbo.HocPhan.ma\_hp, dbo.HocPhan.tinchi

FROM dbo.DangKyLop INNER JOIN dbo.LopDK

ON dbo.DangKyLop.ma\_lop\_dk = dbo.LopDK.ma\_lop\_dk INNER JOIN dbo.HocPhan

ON dbo.LopDK.ma\_hp = dbo.HocPhan.ma\_hp

WHERE (dbo.DangKyLop.diem\_chu IS NOT NULL)

View DsHp

SELECT dbo.HocPhan.ma\_hp, dbo.HocPhan.ten\_hp, dbo.HocPhan.thoi\_luong, dbo.HocPhan.tinchi, dbo.HocPhan.tinchi\_hocphi, dbo.HocPhan.trong\_so, dbo.KhoaVien.ten\_khoa

FROM dbo.HocPhan INNER JOIN dbo.KhoaVien

ON dbo.HocPhan.ma\_khoa = dbo.KhoaVien.ma\_khoa

View DsLopSV

SELECT dbo.SinhVien.mssv, dbo.SinhVien.ho, dbo.SinhVien.dem, dbo.SinhVien.ten, dbo.SinhVien.ngay\_sinh, dbo.LopSV.ten\_lop

FROM dbo.LopSV INNER JOIN dbo.SinhVien

ON dbo.LopSV.ma\_lop = dbo.SinhVien.ma\_lop

View ThongTinSV

SELECT dbo.SinhVien.mssv, dbo.SinhVien.ngay\_sinh, dbo.SinhVien.ma\_lop, dbo.SinhVien.email, dbo.SinhVien.so\_dt, dbo.SinhVien.dia\_chi, dbo.SinhVien.he\_hoc, dbo.SinhVien.trang\_thai, dbo.LopSV.ten\_lop, CONCAT\_WS(' ',

dbo.SinhVien.ho, dbo.SinhVien.dem, dbo.SinhVien.ten) AS ho\_ten, dbo.SinhVien.image

FROM dbo.LopSV INNER JOIN dbo.SinhVien

ON dbo.LopSV.ma\_lop = dbo.SinhVien.ma\_lop

## Stored Procedure

### Stored procedure insertAccount

Thêm một tài khoản, nếu có lỗi xảy ra thì trả về error code trong biến @@ERROR.

Ví dụ: @@ERROR = 2627, báo cho người dùng biết tài khoản thêm vào đã tồn tại.

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[insertAccount]

@userName varchar(50), @passWord varchar(255), @permission varchar(2)

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRY

INSERT INTO dbo.Users(user\_name, pass\_word, permission)

VALUES (@userName, @passWord, @permission);

END TRY

BEGIN CATCH

SELECT @@ERROR AS errorCode;

END CATCH

SELECT @@ERROR AS errorCode;

END

### Stored procedure insertGroupStudentAccount

Dùng để tạo nhanh tài khoản sinh viên trong một khoảng được chỉ định

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[insertGroupStudentAccount]

@startMssv int, @endMssv int

AS

BEGIN

SET NOCOUNT ON;

DECLARE @errorCode int;

DECLARE @passWord varchar(255);

BEGIN TRAN

BEGIN TRY

WHILE @startMssv <= @endMssv

BEGIN

SET @passWord = CONVERT(VARCHAR(255), @startMssv);

SET @passWord = CONVERT(VARCHAR(255), HashBytes('MD5', @passWord),2);

SET @passWord = LOWER(@passWord);

SET @passWord = CONVERT(VARCHAR(255), HashBytes('SHA2\_256', @passWord),2);

SET @passWord = LOWER(@passWord);

INSERT INTO dbo.Users(user\_name, pass\_word, permission)

VALUES(CONVERT(varchar(50), @startMssv), @passWord, 's');

SET @startMssv = @startMssv + 1;

END

END TRY

BEGIN CATCH

SET @errorCode = @@ERROR;

IF @errorCode <> 0

BEGIN

SELECT @errorCode AS errorCode;

ROLLBACK TRAN

RETURN;

END

END CATCH

SET @errorCode = @@ERROR;

COMMIT TRAN

SELECT @errorCode AS errorCode;

END

### Stored procedure changePassWord

Kiểm tra xem người dùng có nhập đúng mật khẩu cũ không. Nếu đúng thì cho phép cập nhật lại mật khẩu mới.

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[changePassWord]

@old\_pass varchar(255), @new\_pass varchar(255), @user\_name varchar(50)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @retval INT;

IF

(

SELECT COUNT(\*)

FROM Users

WHERE Users.user\_name = @user\_name AND Users.pass\_word = @old\_pass

) > 0

BEGIN

UPDATE Users SET pass\_word = @new\_pass WHERE user\_name = @user\_name;

SET @retval = 0;

END

ELSE SET @retval = 1;

SELECT @retval AS RetVal;

END

### Stored procedure checkDkhp

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[checkDkhp]

@maHp varchar(20), @mssv varchar(50)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @retVal nvarchar(255);

SET @retVal = '';

IF (SELECT COUNT(\*)

FROM(

(SELECT ma\_ctdt FROM SinhVien WHERE @mssv = mssv)t1

INNER JOIN

(SELECT ma\_ctdt FROM HocPhanCTDT WHERE @maHp = ma\_hp)t2

ON t1.ma\_ctdt = t2.ma\_ctdt

)

) > 0

BEGIN

DECLARE @hocTruoc nvarchar(255);

DECLARE @tienQuyet nvarchar(255);

DECLARE @songHanh nvarchar(255);

DECLARE @BangDiemSinhVien table

(diem\_chu varchar(2), ma\_hp varchar(20));

INSERT INTO @BangDiemSinhVien(diem\_chu, ma\_hp)

SELECT diem\_chu, ma\_hp

FROM dbo.BangdiemSV WHERE mssv=@mssv;

DECLARE curModuleConstraint CURSOR

FOR SELECT ma\_hpdk, loai\_dk FROM dbo.RangBuocHP WHERE ma\_hp=@maHp;

DECLARE @maHpDk varchar(20), @loaiDk varchar(50);

OPEN curModuleConstraint;

WHILE(0=0)

BEGIN

FETCH NEXT FROM curModuleConstraint into @maHpDk, @loaiDk;

IF(@@FETCH\_STATUS <> 0) break;

IF @loaiDk = 'h'

BEGIN

IF((SELECT COUNT(\*) FROM @BangDiemSinhVien WHERE ma\_hp = @maHpDk) <= 0)

SET @hocTruoc = CONCAT\_WS(', ', @hocTruoc, @maHpDk);

END

ElSE IF @loaiDk = 't'

BEGIN

IF((SELECT COUNT(\*) FROM @BangDiemSinhVien

WHERE diem\_chu IN('A','A+','B','B+','C', 'C+','D','D+') AND ma\_hp=@maHpDk)<=0)

SET @tienQuyet = CONCAT\_WS(', ', @tienQuyet, @maHpDk);

END

ELSE IF @loaiDk = 's'

BEGIN

;

IF ((SELECT COUNT(\*) FROM @BangDiemSinhVien WHERE ma\_hp = @maHpDk) <= 0)

BEGIN

IF ((SELECT COUNT(\*) FROM DangKyHP WHERE ma\_hp = @maHpDk) <= 0)

SET @songHanh = CONCAT\_WS(', ', @songHanh, @maHpDk);

END

END

END

CLOSE curModuleConstraint;

DEALLOCATE curModuleConstraint;

IF (@tienQuyet <> '')

SET @retVal = CONCAT(@retVal, '; ', N'thiếu học phần tiên quyết ', @tienQuyet);

IF (@hocTruoc <> '')

SET @retVal = CONCAT(@retVal, '; ', N'thiếu học phần học trước ', @hocTruoc);

IF (@songHanh <> '')

SET @retVal = CONCAT(@retVal, '; ', N'thiếu học phần song hành ', @songHanh);

END

ELSE

BEGIN

SET @retVal = CONCAT(@maHp,N' Không nằm trong chương trình đào tạo.');

END

SELECT @retVal AS retVal;

END

### Stored procedure checkLogin

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[checkLogin]

@user\_name varchar(50), @pass\_word varchar(255)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @retVal INT;

DECLARE @type VARCHAR(2);

SELECT @type = Users.permission FROM [dbo].[Users]

WHERE Users.user\_name = @user\_name AND Users.pass\_word = @pass\_word;

IF (@@ROWCOUNT > 0)

BEGIN

SET @retVal =

CASE @type

WHEN 'a' THEN 0

WHEN 'g' THEN 1

WHEN 's' THEN 2

END; --Login success

END

ELSE

SET @retVal = -1; --Access denied

SELECT @retVal AS RetVal;

END

**Stored procedure getKetQuaHocTapStored procedure getDsTenLop**

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[getKetQuaHocTap]

@mssv varchar(50)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @BangDiemSinhVien table

(ma\_lop\_dk int, hoc\_ky varchar(8), diem\_qt float, diem\_ck float, diem\_chu varchar(2),

ma\_hp varchar(20), tinchi int);

INSERT INTO

@BangDiemSinhVien(ma\_lop\_dk, hoc\_ky, diem\_qt, diem\_ck, diem\_chu, ma\_hp, tinchi)

SELECT ma\_lop\_dk, hoc\_ky, diem\_qt, diem\_ck, diem\_chu, ma\_hp, tinchi

FROM BangdiemSV WHERE mssv=@mssv;

DECLARE @KetQuaHocTap table

(hoc\_ky varchar(8), gpa float, cpa float, tc\_qua int, tc\_tich\_luy int, tc\_no\_dk int,

tc\_dk int, thieu\_diem varchar(255), trinh\_do nvarchar(255));

INSERT INTO @KetQuaHocTap(hoc\_ky, gpa, cpa)

SELECT hoc\_ky, gpa, cpa FROM KetQuaHT WHERE mssv=@mssv;

DECLARE curHocKy CURSOR

FOR SELECT DISTINCT hoc\_ky FROM KetQuaHT WHERE mssv=@mssv;

OPEN curHocKy;

DECLARE @hocKy varchar(8);

DECLARE @tinChiTichLuy int;

DECLARE @tinChiQua int;

DECLARE @tinChiDK int;

DECLARE @thieuDiem varchar(255);

WHILE(0 = 0)

BEGIN

SET @thieuDiem = '';

FETCH NEXT FROM curHocKy into @hocKy;

IF(@@FETCH\_STATUS <> 0) break;

SELECT @tinChiTichLuy = SUM(tinchi) FROM HocPhan WHERE ma\_hp IN

(

SELECT DISTINCT ma\_hp FROM @BangDiemSinhVien

WHERE diem\_chu IN ('A', 'A+','B', 'B+', 'C', 'C+', 'D', 'D+')

AND hoc\_ky <= @hocKy

);

UPDATE @KetQuaHocTap SET tc\_tich\_luy = @tinChiTichLuy WHERE hoc\_ky = @hocKy;

SELECT @tinChiDK = SUM(tinchi) FROM HocPhan WHERE ma\_hp IN

(

SELECT DISTINCT HocPhan.ma\_hp

FROM DangKyLop

INNER JOIN LopDK ON DangKyLop.ma\_lop\_dk = LopDK.ma\_lop\_dk

INNER JOIN HocPhan ON LopDK.ma\_hp = HocPhan.ma\_hp

WHERE mssv = @mssv AND hoc\_ky <= @hocKy

);

UPDATE @KetQuaHocTap SET tc\_dk = @tinChiDK WHERE hoc\_ky = @hocKy;

UPDATE @KetQuaHocTap

SET tc\_no\_dk = @tinChiDK - @tinChiTichLuy WHERE hoc\_ky = @hocKy;

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[getDsTenLop]

@khoa\_hoc varchar(255), @ten\_khoa nvarchar(255)

AS

BEGIN

SET NOCOUNT ON;

SELECT ten\_lop

FROM LopSV INNER JOIN KhoaVien ON LopSV.ma\_khoa = KhoaVien.ma\_khoa

WHERE ten\_khoa = @ten\_khoa

AND REVERSE(SUBSTRING(REVERSE(ma\_lop), 1,

CHARINDEX('K', REVERSE(ma\_lop)) - 1)) = @khoa\_hoc;

END

SELECT @tinChiQua = SUM(tinchi) FROM @BangDiemSinhVien

WHERE diem\_chu IN ('A', 'A+','B', 'B+', 'C', 'C+', 'D', 'D+')

AND hoc\_ky = @hocKy;

UPDATE @KetQuaHocTap SET tc\_qua = @tinChiQua WHERE hoc\_ky = @hocKy;

SELECT @thieuDiem = CONCAT(@thieuDiem, ma\_hp, ';') FROM

(

SELECT ma\_hp

FROM DangKyLop INNER JOIN LopDK ON DangKyLop.ma\_lop\_dk = LopDK.ma\_lop\_dk

WHERE hoc\_ky <= @hocKy

EXCEPT

SELECT ma\_hp FROM @BangDiemSinhVien

WHERE hoc\_ky <= @hocKy

) table1;

UPDATE @KetQuaHocTap SET thieu\_diem = @thieuDiem WHERE hoc\_ky = @hocKy;

UPDATE @KetQuaHocTap SET trinh\_do =

(

CASE

WHEN @tinChiTichLuy < 32 THEN N'Năm thứ nhât'

WHEN @tinChiTichLuy < 64 THEN N'Năm thứ hai'

WHEN @tinChiTichLuy < 96 THEN N'Năm thứ ba'

WHEN @tinChiTichLuy < 128 THEN N'Năm thứ tư'

ELSE N'Năm thứ năm'

END

) WHERE hoc\_ky = @hocKy;

END

CLOSE curHocKy;

DEALLOCATE curHocKy;

SELECT \* FROM @KetQuaHocTap;

END

## Trigger

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER TRIGGER [dbo].[deleteAccountTrigger] ON [dbo].[Users]

INSTEAD OF DELETE

AS

SET NOCOUNT ON;

DECLARE @type varchar(2);

DECLARE @userName varchar(50);

SELECT @type=deleted.permission, @userName=deleted.user\_name FROM deleted;

if @type='s'

DELETE FROM SinhVien WHERE mssv=@userName;

DELETE FROM Users WHERE user\_name=@userName;

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER TRIGGER [dbo].[deleteSinhVienTrigger]

ON [dbo].[SinhVien]

INSTEAD OF DELETE

AS

BEGIN

SET NOCOUNT ON;

DECLARE @mssv varchar(50);

SELECT @mssv=deleted.mssv FROM deleted;

DELETE FROM KetQuaHT WHERE mssv=@mssv;

DELETE FROM DangKyLop WHERE mssv=@mssv;

DELETE FROM DangKyHP WHERE mssv=@mssv;

DELETE FROM SinhVien WHERE mssv=@mssv;

END

USE [StudentManager]

GO

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER TRIGGER [dbo].[updateDangKyLopTrigger]

ON [dbo].[DangKyLop]

FOR UPDATE

AS

BEGIN

SET NOCOUNT ON;

DECLARE @mssv varchar(50);

DECLARE @maLopDk int;

DECLARE @diemTK float;

DECLARE @diemChu varchar(2);

SELECT @maLopDk=ma\_lop\_dk, @mssv=mssv FROM inserted;

SELECT @diemTK=(diem\_ck\*trong\_so)+diem\_qt\*(1-trong\_so)

FROM DangKyLop

JOIN LopDK ON DangKyLop.ma\_lop\_dk=LopDK.ma\_lop\_dk

JOIN HocPhan ON LopDK.ma\_hp = HocPhan.ma\_hp

WHERE mssv=@mssv AND DangKyLop.ma\_lop\_dk=@maLopDk;

IF @diemTK IS NULL

RETURN;

SET @diemChu =

CASE

WHEN @diemTK >= 9.5 THEN 'A+'

WHEN @diemTK>=8.5 THEN 'A'

WHEN @diemTK>=8.0 THEN 'B+'

WHEN @diemTK>=7.0 THEN 'B'

WHEN @diemTK>=6.5 THEN 'C+'

WHEN @diemTK>=5.5 THEN 'C'

WHEN @diemTK>=5.0 THEN 'D+'

WHEN @diemTK>=4.0 THEN 'D'

WHEN @diemTK<4.0 THEN 'F'

END

UPDATE DangKyLop

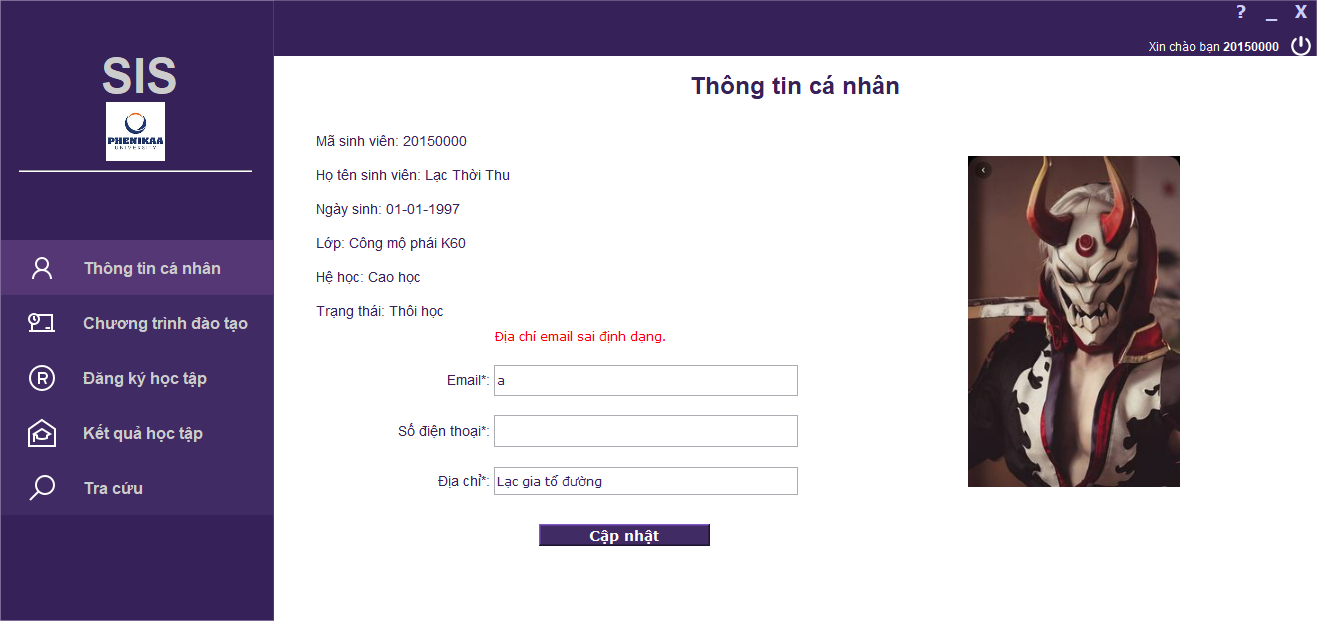
SET diem\_chu=@diemChu

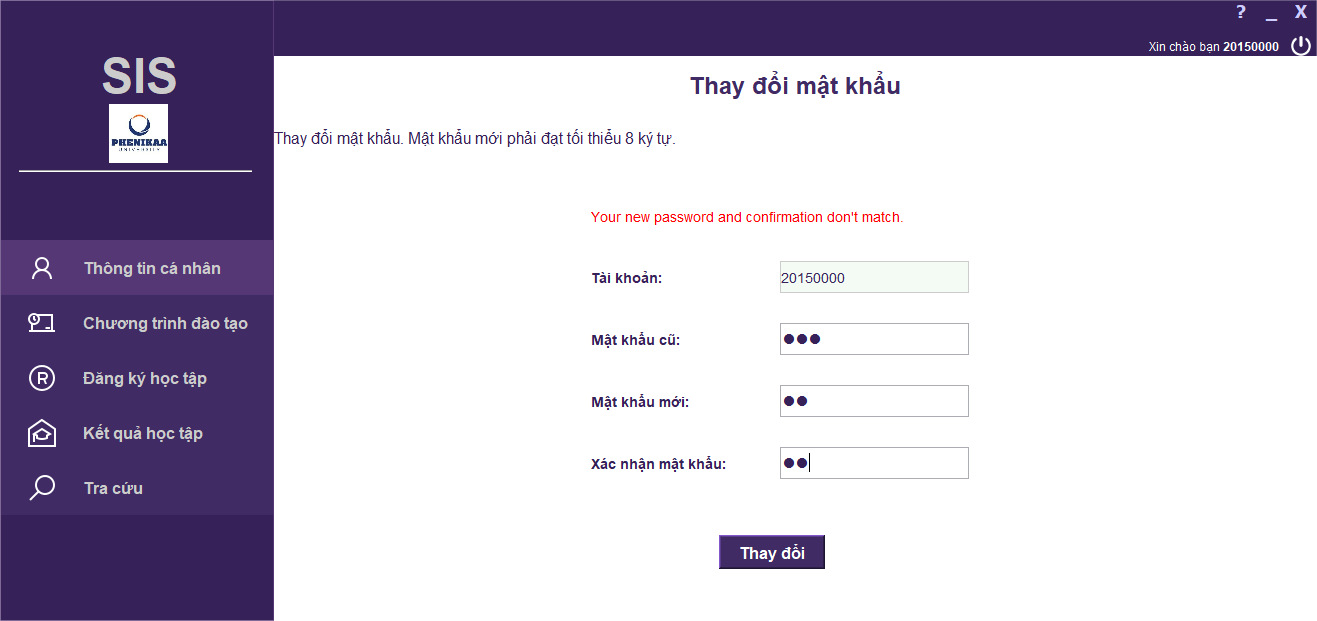
WHERE ma\_lop\_dk=@maLopDk AND mssv=@mssv;

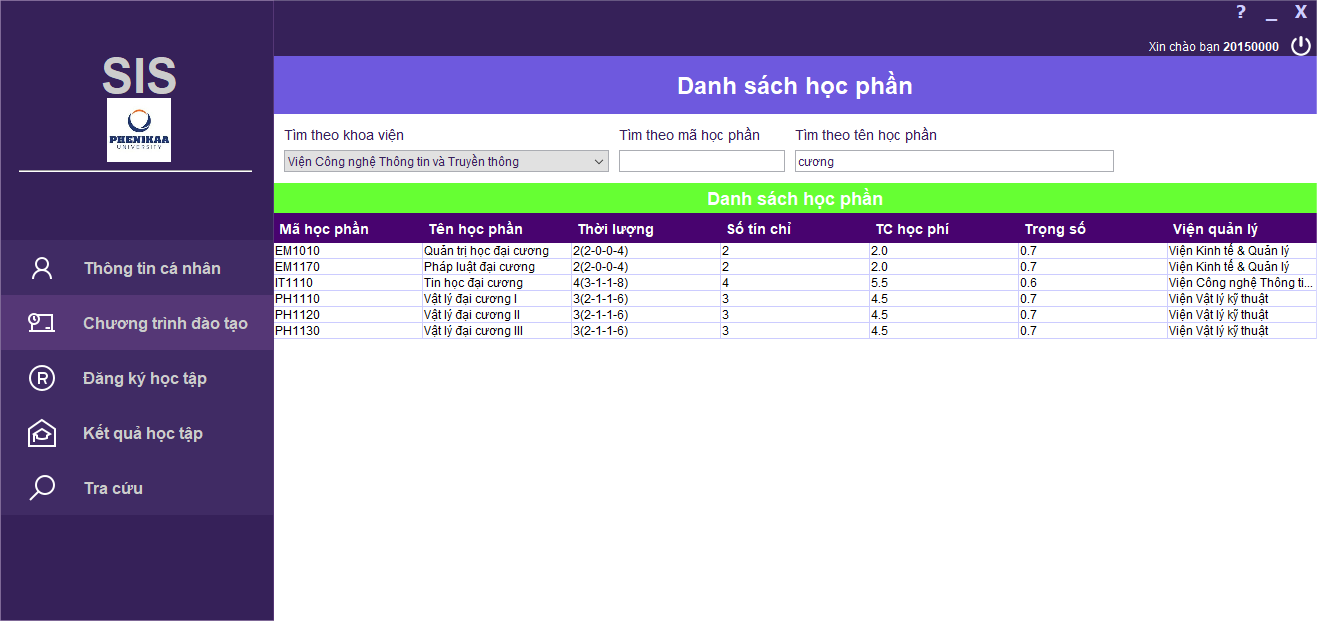
END

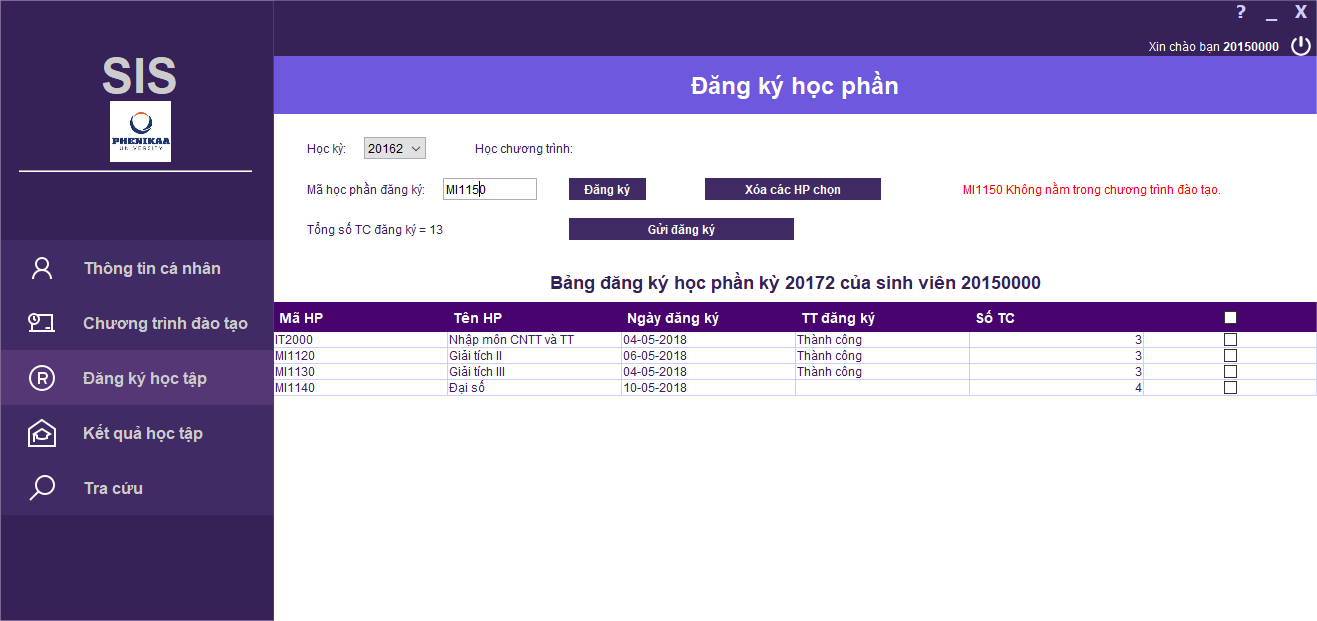
# Giao diện chương trình

Sinh viên

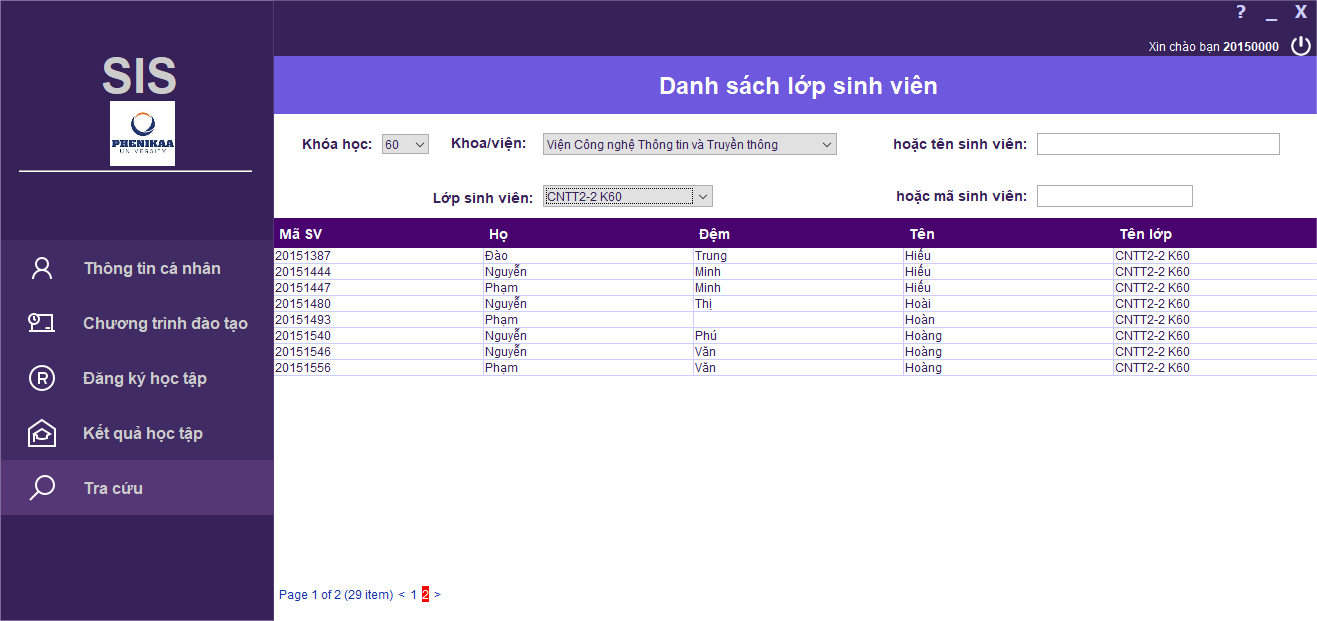
Thông tin cá nhân

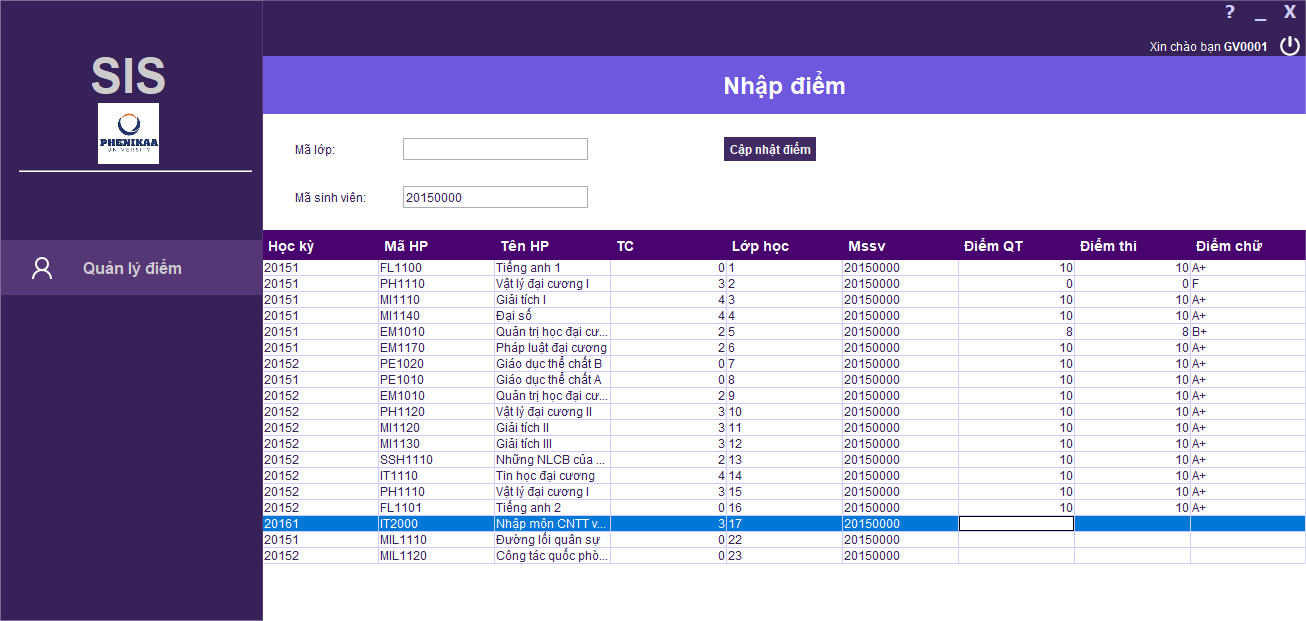
Đổi mật khẩu

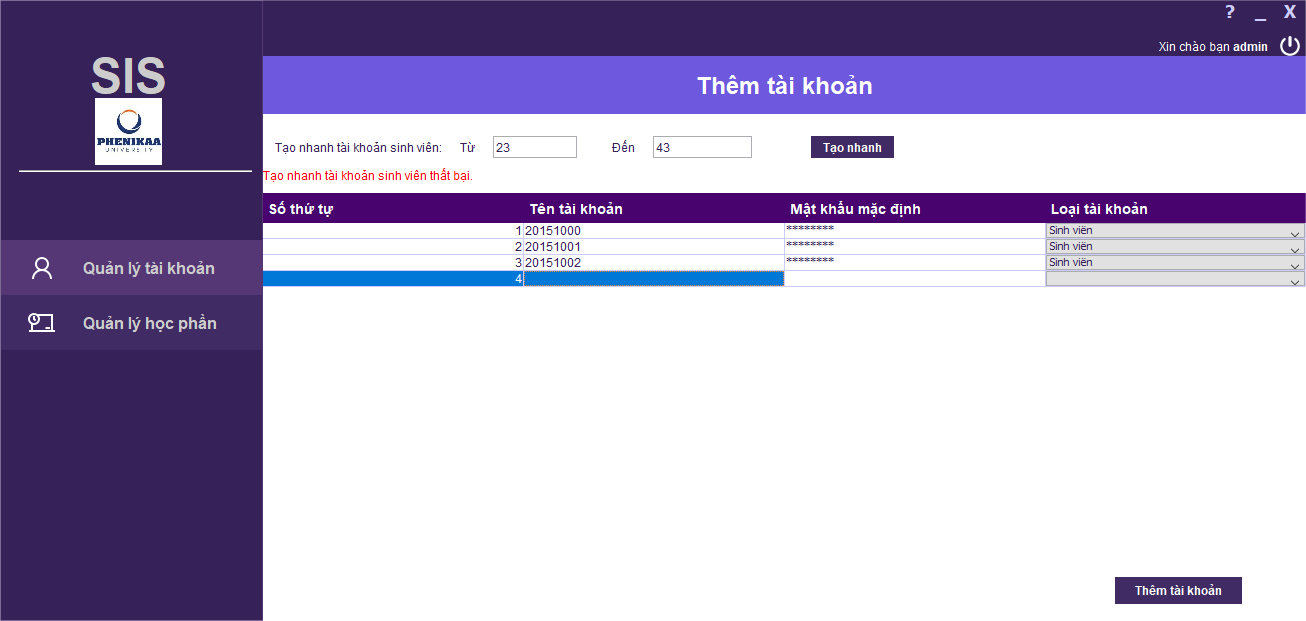
Danh sách học phần

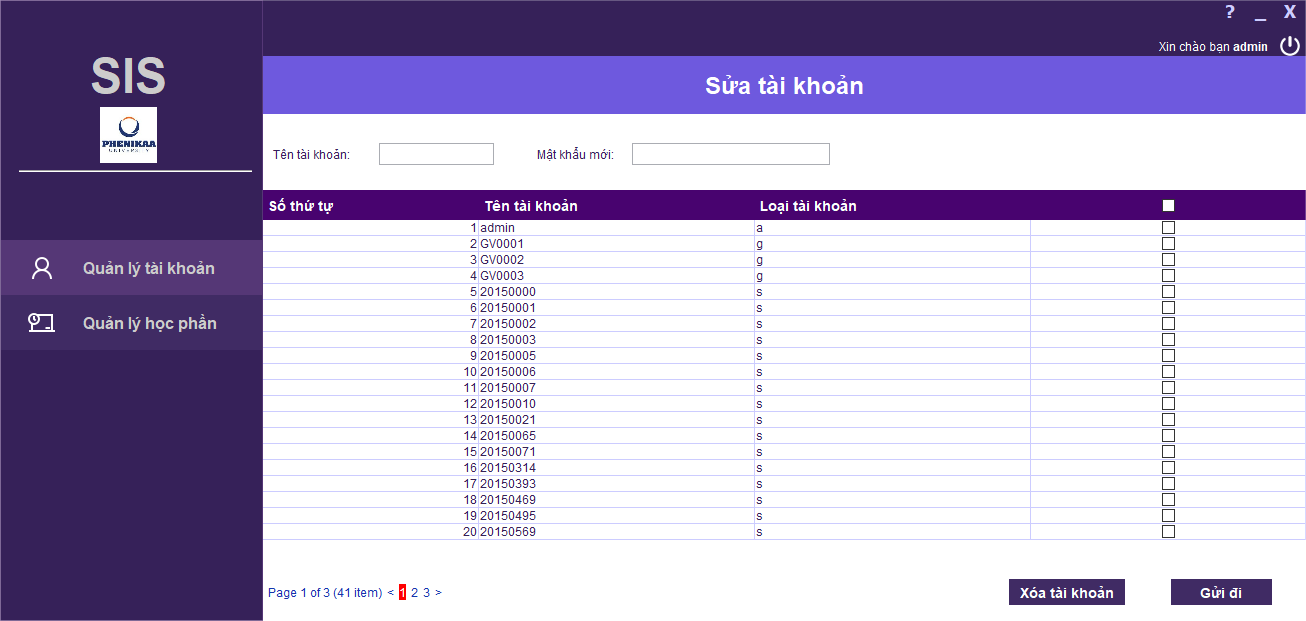
Đăng ký học phần

Bảng điểm cá nhân

Danh sách lớp sinh viên

Nhập điểm

Thêm tài khoản

Sửa tài khoản

# Lời kết

# Qua quá trình nghiên cứu và thực hành, chúng em đã có cơ hội tìm hiểu sâu hơn về các khái niệm và ứng dụng của lập trình hướng đối tượng (OOP). Từ việc phân tích vấn đề đến thiết kế và triển khai các lớp, đối tượng, chúng em nhận thấy rằng lập trình hướng đối tượng không chỉ giúp tổ chức mã nguồn một cách khoa học, dễ bảo trì mà còn tạo điều kiện thuận lợi cho việc phát triển các ứng dụng phức tạp. Chúng em hy vọng những kiến thức này sẽ là nền tảng vững chắc để chúng em tiếp tục phát triển trong các dự án lập trình sắp tới. Tuy còn một số hạn chế trong việc áp dụng và thực hành, chúng em sẽ nỗ lực học hỏi thêm để hoàn thiện kỹ năng của mình.

Chúng em xin chân thành cảm ơn cô Nguyễn Lệ Thu đã tận tình hướng dẫn và truyền đạt những kiến thức quý báu trong suốt quá trình học lập trình hướng đối tượng. Sự nhiệt tình và chu đáo của cô đã giúp chúng em nắm vững những kiến thức quan trọng và có được cái nhìn toàn diện hơn về lập trình OOP. Những kiến thức và kinh nghiệm mà cô chia sẻ không chỉ hữu ích trong phạm vi môn học mà còn là hành trang quan trọng cho con đường học tập và làm việc của chúng em trong tương lai.

# Tài liệu tham khảo

## Quy chế đào tạo đại học hệ chính quy ĐH Bách Khoa

1. Java Documentation: <https://docs.oracle.com/en/java/>
2. **Java OOP Tutorials**: <https://www.baeldung.com/java-oop>
3. Github Java OOP projects
4. SQL Server documentation: <https://learn.microsoft.com/en-us/sql/?view=sql-server-ver15>
5. MySQL Documentation: <https://dev.mysql.com/doc/>