ĐẠI HỌC QUỐC GIA THNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA HỆ THỐNG THÔNG TIN

ĐỒ ÁN

MÔN HỌC: HỆ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Đề tài: Quản lý học sinh

Giảng viên hướng dẫn: **ThS. Đỗ Thị Minh Phụng**

Sinh viên thực hiện:

Lê Phạm Huy Cường	–	**17520306**
Trần Minh Đức	–	**17520369**
Đỗ Quang Thiện	–	**17521073**
Dương Quốc Cường	–	**17520304**

Tp. Hồ Chí Minh tháng năm 2019

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI 7

I. Đặt vấn đề: 7

II. Mục tiêu: 8

III. Công cụ sử dụng: 9

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ DỮ LIỆU 10

I. Đặc tả bài toán: 10

II. Phân tích yêu cầu: 11

1. Yêu cầu phi chức năng: 11

2. Thiết kế mô hình quan hệ: 12

a. Mô hình thực thể mối kết hợp: 12

b. Thiết kế dữ liệu: 12

i. Sơ đồ logic: 13

ii. Mô hình dữ liệu quan hệ: 13

c. Mô tả thành phần dữ liệu: 14

d. Mô tả ràng buộc toàn vẹn: 23

i. Ràng buộc khóa chánh: 23

ii. Ràng buộc khóa ngoại: 23

iii. Ràng buộc miền dữ liệu: 25

CHƯƠNG 3: CÁC CÔNG NGHỆ SỬ DỤNG 27

I. Oracle: 27

II. JDBC (Java Database Connectivity): 27

III. JavaSwing: 28

IV. Git: 29
V. GitKraken: 29
VI. NetBean: 29
CHƯƠNG 4: XÂY DỰNG VÀ QUẢN LÍ GIAO TÁC 31
I. Trigger: 31
1. Trigger trong Oracle: 31
2. Danh sách các trigger: 33
3. Mô tả một số trigger : 33
a. Cập nhật điểm cho sinh viên: 33
b. Cập nhật ngày hết hạn đóng học phí khi thêm record vào bảng Fee: 35
II. Stored Procedure: 36
1. Stored Procedure trong Oracle: 36
2. Transaction trong Oracle: 37
a. Định nghĩa: 37
b. Cấu trúc transaction: 38
c. Transaction control: 39
3. Danh sách các Procedure: 48
4. Mô tả một số Stored Procedure : 50
a. Đăng kí học phần: 50
b. Lấy ra điểm trung bình học kỳ của sinh viên: 55
c. Lấy ra danh sách sinh viên dự kiến nhận học bổng trong học kỳ: 57
CHƯƠNG 5: XỬ LÝ TRUY XUẤT ĐỒNG THỜI 60
I. Các mức cô lập trong Oracle: 60

1. Read Committed : 60
2. Serializable: 60
3. Read-only: 61
II. Cơ chế khóa: 61
III. Deadlock: 63
IV. Mô tả đồ án môn học: 68
1. Lost Update: 68
2. Non-repeatable read: 76
3. Phantom read: 78
CHƯƠNG 6: THIẾT KẾ GIAO DIỆN 83
I. Danh sách các màn hình: 83
1. Đăng nhập: 83
2. IT-department: 83
3. Student: 83
4. Lecturer: 83
5. Phòng đào tạo: 84
6. Phòng công tác sinh viên: 85
II. Mô tả các màn hình: 85
1. Phòng công tác sinh viên: 85
a. Danh sách dự kiến: 85
b. Danh sách chính thức: 87
c. Học bổng: 88
2. Đăng nhập: 90

3. IT-department: 91
a. Quản lí phòng IT: 91
b. Tạo phòng: 92
4. Student: 93
a. Quản lí đăng kí học phần: 93
b. Quản lí bảng điểm: 95
c. Quản lí học phí: 96
5. Lecturer: 97
a. Quản lí điểm: 97
i. Môn học: 97
ii. Danh sách sinh viên: 99
6. Phòng đào tạo: 100
a. Quản lí môn học: 100
i. Thêm môn học: 101
b. Quản lí học phần: 103
i. Chỉnh sửa học phần: 104
ii. Mở học phần: 104
c. Quản lí khoa: 105
i. Thêm khoa: 106
d. Quản lí giảng viên: 108
i. Thêm vào khoa: 109
e. Quản lí sinh viên: 109
i. Thêm vào lớp: 110

f. Quản lí lớp: 111

i. Mở lớp: 112

CHƯƠNG 7: KẾT LUẬN 116

I. Kết quả đạt được: 116

II. Hạn chế: 117

III. Hướng phát triển: 117

Phụ lục 1: Bảng phân công công việc: 118

Phụ lục 2: Tài liệu tham khảo: 118

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

I. Đặt vấn đề:

Ngày nay, công nghệ thông tin được xem là một ngành mũi nhọn của các quốc gia, đặc biệt là các quốc gia đang phát triển, tiến hành công nghiệp hóa và hiện đại hoá như nước ta. Sự bùng nổ thông tin và sự phát triển mạnh mẽ của công nghệ kỹ thuật số, yêu cầu muốn phát triển thì phải tin học hoá vào tất cảcác ngành các lĩnh vực. Cùng với sự phát triển nhanh chóng về phần cứng máy tính, các phần mềm ngày càng trở nên đa dạng, phong phú, hoàn thiện hơn và hỗ trợ hiệu quả cho con người. Các phần mềm hiện nay ngày càng mô phỏng được rất nhiều nghiệp vụ khó khăn, hỗ trợ cho người dùng thuận tiện sử dụng, thời gian xử lý nhanh chóng, và một số nghiệp vụ được tự động hoá cao.

Do vậy mà trong việc phát triển phần mềm, đòi hỏi không chỉ là sự chính xác, xử lý được nhiều nghiệp vụ thực tế mà còn phải đáp ứng các yêu cầu khác như về tốc độ, giao diện thân thiện, mô hình hoá được thực tế vào máy tính để người sử dụng tiện lợi, quen thuộc, tính tương thích cao, bảo mật cao (đối với các dữ liệu nhạy cảm), ... Các phần mềm giúp tiết kiệm một lượng lớn thời gian, công sức của con người, và tăng độ chính xác và hiệu quả trong công việc (nhất là việc sửa lỗi và tự động đồng bộ hoá).Một ví dụ cụ thể, việc quản lý học vụ trong trường trung học phổ thông. Nếu không có sự hỗ trợ của tin học ứng dụng, việc quản lý này phải cần khá nhiều người, chia thành nhiều khâu, mới có thể quản lý được toàn bộ hồ sơ, học sinh (thông tin, điểm số, học phí ...) cũng như các nghiệp vụ sắp thời khoá biểu, tính điểm trung bình, xếp loại học lực cho học sinh toàn trường (số lượng học sinh có thể lên đến hai ba nghìn học sinh). Các công việc này đòi hỏi nhiều thời gian và công sức, mà sự chính xác và hiệu quả không cao, vì đa số đều làm bằng thủ công rất ít tự động. Một số nghiệp vụ như tra cứu, thống kê, và hiệu chỉnh thông tin khá vất vả. Ngoài ra còn có một số khó khăn về vấn đề lưu trữ khá đồ sộ, dễ bị thất lạc, tốn kém, ... Trong khi đó, các nghiệp vụ này hoàn toàn có thể tin học hoá một cách dễ dàng. Với sự giúp đỡ của tin học, việc quản lý học vụ sẽ trở nên đơn giản, thuận tiện, nhanh chóng và hiệu quả hơn rất nhiều.

Ứng dụng phần mềm quản lý học sinh thì rất phổ biến ở những trường Đại Học, ở những trường Trung Học PhổThông thì ít được sử dụng. Vấn đề này đã được nhiều người nghiên cứu . Song, vẫn còn nhiều hạn chế, để khắc phục những mặt hạn chế đó , tôi đề ra một sốphương pháp ứng dụng phần mềm quản lý học sinh trở nên hoàn thiện, hiệu quả hơn khi được ứng dụng vào thực tiễn.

Hệ thống cần thiết phải giải quyết được các vấn đề trong việc cập nhật đồng thời và thao tác đồng thời giữa các học sinh. Bên cạnh đó, hệ thống được xây dựng phải là hệ thống quản lý tích hợp, lưu trữ tập trung dễ tương tác với học sinh và nhân viên.

II. Mục tiêu:

Xây dựng được một hệ thống ứng dụng hoàn thiện để học sinh và tất cả những nhân viên trong phòng đào tạo có một hệ thống trao đổi một cách dễ dàng và thuận tiện. Ứng dụng sẽ giới thiệu những chức năng tối ưu nhất và dễ dàng sữ dụng đối với tất cả mọi người. Cung cấp những ứng dụng có chức năng dành riêng cho một bộ phận nào đó như : Học sinh có chức năng xem điểm rèn luyện , xem bảng điểm , đăng kí học phần và thông tin học phí … Phòng đào tạo có chức năng quản lý học sinh, quản lý môn học và quản lý điểm … Phòng kế hoạch tài chính có chức năng quản lý học phí ..v..v... Ngoài ra ứng dụng còn có mục đích giới thiệu rộng cho nhiều học sinh được biết hơn về thông tin trường cũng như thu hẹp khoảng cách xa gần khi đăng kí học phần qua ứng dụng giải quyết được vấn đề đường xa việc đi lại khó khăn.

Trong đó hệ thống sẽ đáp ứng được các yêu cầu sau:

- Quản lý thông tin học sinh, admin, phòng công tác sinh viên, phòng đào tạo, phòng kế hoạch tài chính .
- Quản lý các hoạt động của học sinh: đăng kí học phần online và có thể hủy bỏ những môn khi đăng kí, xem điểm bằng ứng dụng, xem thông tin học phí và những thông tin liên quan khác .
- Đảm bảo việc tìm kiếm và xem thông tin của học sinh, thông tin của những văn phòng: các thông tin về tình trạng học phí, trạng thái, thông tin chi tiết học sinh .

- Quản lý việc thu học phí của học sinh.
- Quản lý các hoạt động về báo cáo thống kê về học phí , điểm rèn luyện , học bổng.
- Thông tin lưu trữ sao cho việc kết xuất dữ liệu được nhanh chóng, hiệu quả.
- Hệ thống hỗ trợ phân quyền người dùng, đảm bảo tính bảo mật, an toàn của hệ thống.

III.Công cụ sử dụng:

Trong quá trình thực hiện, nhóm đã sử dụng một số phần mềm phục vụ cho việc tìm và xây dựng đề tài:

- Oracle developer
- Microsoft Visio
- Netbean
- Git
- Ireport

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ DỮ LIỆU

I. Đặc tả bài toán:

Sinh viên, phòng đào tạo, phòng kế hoạch tài chính, phòng cộng tác sinh viên, và admin cần phải đăng nhập thành công vào hệ thống với tài khoản và mật khẩu đã được cung cấp trước. Ứng với mỗi quyền khi đăng nhập sẽ được truy cập vào những chức năng khác nhau trong hệ thống.

Hoạt động của người quản lý (bao gồm phòng IT, phòng công tác sinh viên, phòng kế hoạch tài chính và phòng đào tạo) :

Phòng IT: sau khi đăng nhập bằng tài khoản đã được cấp trước đó mỗi nhân viên phòng IT có thể quản lý các thông tin sau đây: chỉnh sữa thông tin học sinh, giảng viên và quản lý tài khoản và quản lý hệ thống đăng ký học phần.

Phòng công tác sinh viên: xem thông tin học sinh và xét học bổng cho học sinh nào có kết quả tốt nhất trong học kỳ.

Phòng kế hoạch tài chính: sẽ cập nhật thông tin về các sinh viên kiểm tra và sử lý về thông tin học phí của các sinh viên đã hoàn thành học phí và những sinh viên chưa hoàn thành học phí . Sau đó sẽ thống kê tất cả sinh viên đã hoàn thành học phí và chưa hoàn thành học phí và sẽ tiến hành liên lạc cũng như nhắc nhở đối với những sinh viên chưa hoàn thành học phí và sẽ cập nhật thông tin trên lên cơ sở dữ liệu.

Phòng đào tạo: sau khi học sinh có kết quả lên lớp hay đậu vào trường đại học hoặc trường nào đó thì phòng đào tạo sẽ cung cấp kế hoạch đào tạo và bảng thống kê chi tiết về từng môn học và dự định từng kỳ sẽ học những môn tiên quyết nào cho học sinh có thể dễ nắm bắc hiểu được và thống kê điểm của sinh viên qua tất cả các kỳ để xem sinh viên có vượt qua tất cả các môn được đề ra nếu không sẽ cho sinh viên đăng ký học lại để có thể ra trường.

Hoạt động của học sinh:

Học sinh: mỗi học sinh có thể xem thông tin trên website mà nhà trường đã cung cấp hoặc có những thắc mắc có thể đến phòng cộng tác sinh viên để được hướng dẫn cụ thể, khi kết thúc kỳ thi giữa kỳ hoặc cuối kỳ thì học sinh có thể xem bảng điểm của mình trên hệ thống website. Khi bắt đầu một học kỳ mới thì học sinh có thể đăng ký học phần và hệ thống sẽ kiểm tra thông tin có hợp lệ hay không như là có hoàn thành các môn tuyên quyết hay chưa nếu chưa thì không được đăng kí những môn học sau đó, và học sinh có thể xem thông tin học phí, xem thông tin về chính bản thân mình.

II. Phân tích yêu cầu:

1. Yêu cầu phi chức năng:

Yêu cầu về giao diện: giao diện ứng dụng gần gũi, thân thiện. Người dùng có thể sử dụng dễ dàng, thuận tiện và có tính tương tác cao. Thao tác trên ứng dụng nhanh gọn và dễ sử dụng.

Yêu cầu về chất lượng:

Tính tiến hóa: Dễ dàng trong việc nâng cấp hay thêm các modules tiện ích khác.

Tính tiện dụng:

- Ứng dụng có giao diện trực quan, thân thiện và dễ sử dụng.
- Các đặc tả và hướng dẫn sử dụng rõ ràng.
- Thao tác đơn giản và nhanh gọn, đáp ứng nhu cầu của học sinh.

Tính tương thích: Hoạt động tốt trên mọi nền tảng, trên ứng dụng, mobile, tablet.

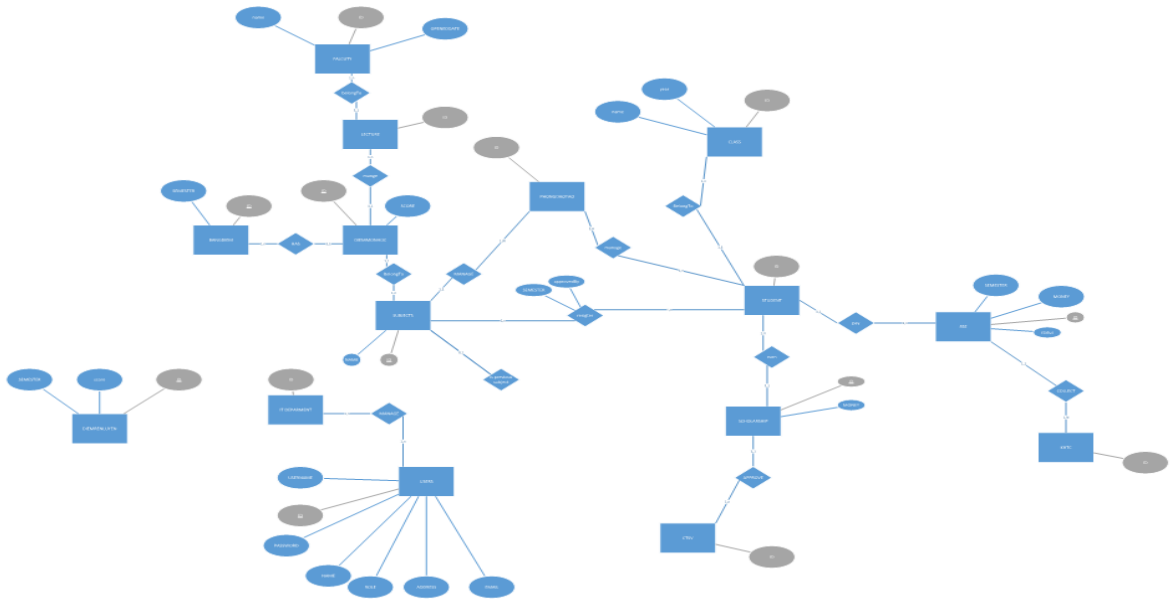
Tính hiệu quả:

- Ứng dụng thể hiện tốt thông tin học sinh, phòng đào tạo , phòng công tác sinh viên, admin,phòng kế hoạch tài chính, chi tiết về học phí , học bổng, và thanh toán.

Trang ứng dụng hoạt động ổn định, và đáng tin cậy, có thể thể truy cập và xử lý đồng thời nhiều hành động.

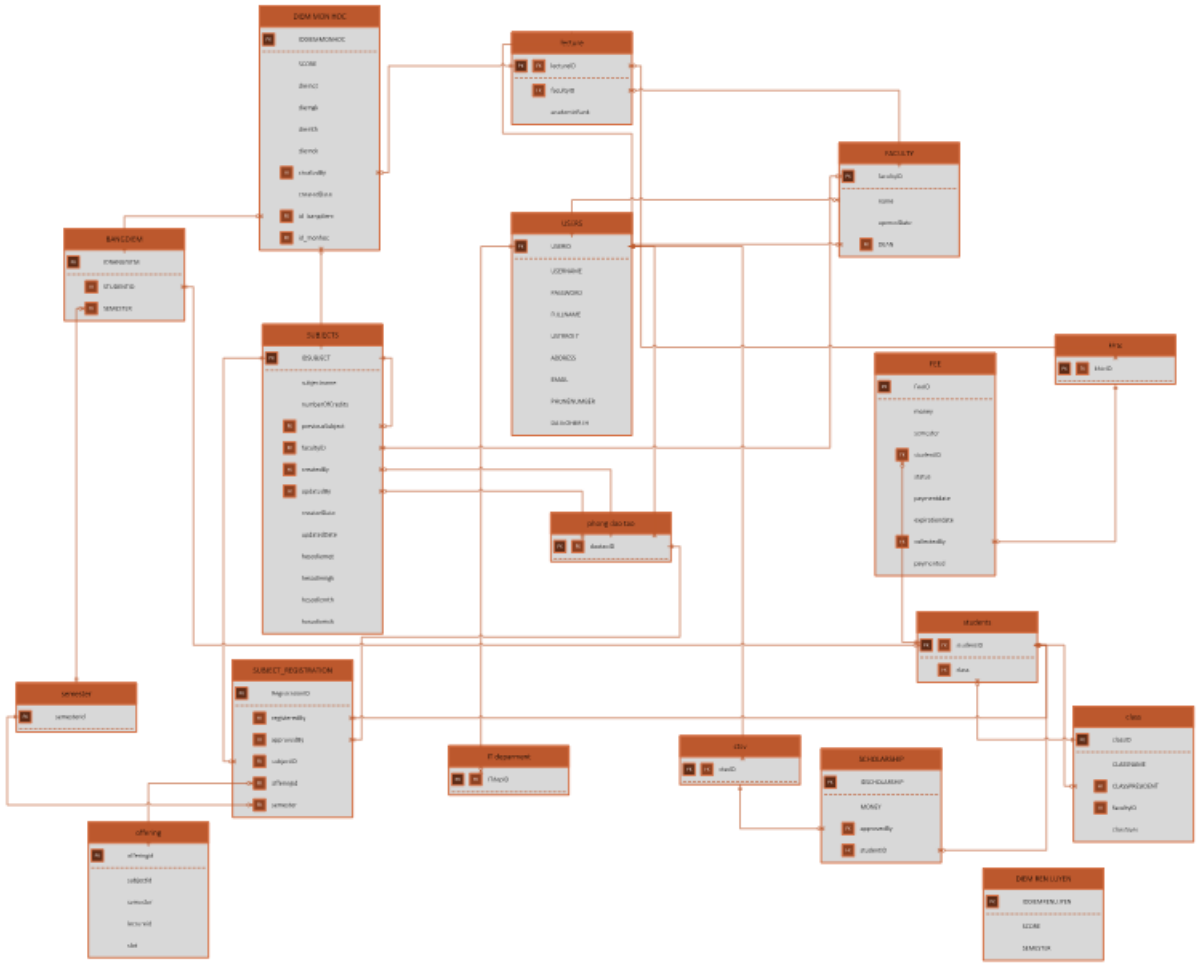
2. Thiết kế mô hình quan hệ:

a. Mô hình thực thể mối kết hợp:



b. Thiết kế dữ liệu:

i. Sơ đồ logic:



ii. Mô hình dữ liệu quan hệ:

Users (<u>userID</u>, userName , userPassword , fullName , gender , userRole , phoneNumber , dateOfBirth , email , address)

Role (<u>roleID</u>, roleName)

Student (<u>studentID</u>, classID)

Class (<u>classID</u> ,className ,classPresident ,facultyID ,classSize)

Lecturer (<u>LectureID</u>, facultyID ,academicRank)

Faculty (<u>facultyID</u> ,facultyName ,dean ,openedDate)

PhongDaoTao (<u>PDT_id</u>)

Subject (subjectID, subjectName, numberOfCredits, createdBy ,updatedBy,createdDate, updatedDate, facultyID , previousSubject, hesodiemQT, hesodiemTH, hesodiemGK hesodiemCK)

Offering(offeringID, subjectID, semester, lectureID, slot)

Semester(semesterID)

Subject_Registration(registrationID, registeredBy, approvedBy, offeringID, semester)

BangDiem (ID_bangdiem ,studentID, semester)

DiemMonHoc (ID_DiemMonHoc ,score, createdBy, createdDate, ID_bangdiem, subjectID, diemQT, diemTH, diemCK, diemGK)

IT_Department (ITdept_ID)

Ctsv (ID_ctsv)

Scholarship (ID_scholarship ,money ,approvedBy ,studentID, semester)

Fee (feeID ,money ,semester ,studentID ,status ,collectedBy, paymentDate, expirationDate, paymented)

KHTC (KHTC_id)

c. Mô tả thành phần dữ liệu:

Bảng Users

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
userID	varchar2(5)	id của user	khóa chánh
userName	varchar2(32)	tên của user	
userPassword	varchar2(64)	mật khẩu user	
fullName	varchar2(62)	tên đầy đủ của user	Ràng buộc
gender	varchar2(5)	giới tính của user	khóa chánh
userRole	varchar2(10)	vai trò của user	

phoneNumber	varchar2(10)	số điện thoại của user	
dateOfBirth	date	ngày sinh của user	Ràng buộc
email	varchar2(128)	email của user	khóa chánh
address	varchar2(256)	địa chỉ nhà của user	

Bảng Student

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
studentID	varchar2(5)	id của sinh viên	khóa chánh, khóa ngoại tham chiếu đến khóa chánh userID của bảng User
classID	varchar2(5)	mã lớp của sinh viên	khóa ngoại tham chiếu đến khóa chánh classID của bảng Class

Bảng Class

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
classID	varchar2(5)	mã lớp	khóa chánh
className	varchar2(32)	tên lớp	
classPresident	varchar2(5)	mã sinh viên	khóa ngoại tham chiếu đến khóa chánh studentID của bảng Student
facultyID	varchar2(5)	mã khoa	khóa ngoại tham chiếu đến khóa chánh facultyID của bảng Faculty

classSize	number	sỉ số lớp	

Bảng Lecture

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
LectureID	varchar2(5)	mã giảng viên	khóa chánh, khóa ngoại tham chiếu đến khóa chánh userID của bảng User
facultyID	varchar2(5)	mã khoa	khóa ngoại tham chiếu đến khóa chánh facultyID của bảng Faculty
academicRank	varchar2(32)	học vị, học hàm	

Bảng Faculty

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
facultyID	varchar2(5)	mã khoa	khóa chánh
facultyName	varchar2(32)	tên khoa	
dean	varchar2(5)	mã giảng viên	khóa ngoại tham chiếu đến khóa chánh LecturerID của bảng Lecturer
openedDate	date	ngày thành lập	

Bảng PhongDaoTao

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc

PDT_id	varchar2(5)	mã nhân viên phòng đào tạo	khóa chánh, khóa ngoại tham chiếu đến khóa chánh userID của bảng User

Bảng subject

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
subjectID	varchar2(5)	mã môn học	khóa chánh
subjectName	varchar2(32)	tên môn học	
numberOfCredits	number	số tín chỉ	
createdBy	varchar2(5)	mã nhân viên phòng đào tạo	khóa ngoại tham chiếu đến khóa chánh PDT_id của bảng PhongDaoTao
updatedBy	varchar2(5)	mã nhân viên phòng đào tạo	khóa ngoại tham chiếu đến khóa chánh PDT_id của bảng PhongDaoTao
createdDate	date	ngày tạo môn học	
updatedDate	date	ngày cập nhạt	
facultyID	varchar2(5)	mã khoa	khóa ngoại tham chiếu đến khóa chánh facultyID của bảng faculty

previousSubject	varchar2(5)	mã môn học trước	khóa ngoại tham chiếu đến khóa chánh subjectID của bảng Subject
hesodiemGK	number	hệ số điểm giữa kì	
hesodiemCK	number	hệ số điểm cuối kì	
hesodiemTH	number	hệ số điểm thực hành	
hesodiemQT	number	hệ số điểm quá trình	

Bảng Offering

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
offeringID	varchar2(10)	mã học phần	khóa chánh
subjectID	varchar2(10)	mã môn học	
semester	varchar2(32)	học kì	
lectureID	varchar2(10)	mã giảng viên	
slot	number	số chỗ tối đa của học phần đó	

Bảng Semester

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
semesterID	varchar2(32)	mã học kì	khóa chánh

Bảng Subject_Registration

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
registrationID	varchar2(5)	mã đăng kí học phần	khóa chánh

registeredBy	varchar2(5)	mã sinh viên	khóa ngoại tham chiếu đến khóa chánh studentIDcủa bảng Student
approvedBy	varchar2(5)	mã nhân viên phòng đào tạo	khóa ngoại tham chiếu đến khóa chánh PDT_id của bảng PhongDaoTao
offeringID	varchar2(10)	mã học phần	khóa ngoại tham chiếu đến khóa chánh subjectID của bảng Subject
semester	varchar2(32)	học kì	

Bảng BangDiem

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
ID_bangdiem	varchar2(5)	mã bảng điểm	khóa chánh
studentID	varchar2(5)	mã sinh viên	khóa ngoại tham chiếu đến khóa chánh studentIDcủa bảng Student
semester	varchar2(32)	học kì	

Bảng DiemMonHoc

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc

ID_DiemMonHoc	varchar2(5)	mã điểm môn học	khóa chánh
score	number	điểm	
createdBy	varchar2(5)	mã giảng viên	khóa ngoại tham chiếu đến khóa chánh LecturerID của bảng Lecturer
createdDate	date	ngày nhập	
ID_bangdiem	varchar2(5)	mã bảng điểm	khóa ngoại tham chiếu đến khóa chánh ID_bangdiem của bảng BangDiem
subjectID	varchar2(5)	mã môn học	khóa ngoại tham chiếu đến khóa chánh subjectID của bảng Subject
diemQT	number	điểm quá trình	
diemGK	number	điểm giữa kì	
diemCK	number	điểm cuối kì	
diemTH	number	điểm thực hành	

Bảng IT_department

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
ITdept_ID	varchar2(5)	mã nhân viên phòng IT	khóa chánh,khóa ngoại tham chiếu đến khóa chánh userID của bảng User

Bảng Ctsv

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
ID_ctsv	varchar2(5)	mã nhân viên phòng công tác sinh viên	khóa chánh,khóa ngoại tham chiếu đến khóa chánh userID của bảng User

Bảng Scholarship

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
ID_scholarship	varchar2(5)	mã học bổng	khóa chánh
money	number	số tiền	
approvedBy	varchar2(5)	mã nhân viên phòng công tác sinh viên	khóa ngoại tham chiếu đến khóa chánh ID_ctsv của bảng Ctsv
studentID	varchar2(5)	mã sinh viên	khóa ngoại tham chiếu đến khóa chánh studentIDcủa bảng Student
semester	varchar2(32)	học kì	

Bảng Fee

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
feeID	varchar2(5)	mã phí	khóa chánh

money	number	số tiền	
semester	varchar2(32)	học kì	
studentID	varchar2(5)	mã sinh viên	khóa ngoại tham chiếu đến khóa chánh studentIDcủa bảng Student
status	varchar2(256)	trạng thái	
collectedBy	varchar2(5)	mã nhân viên phòng tài chánh	khóa ngoại tham chiếu đến khóa chánh KHTC_id của bảng KHTC
paymentDate	date	ngày đóng tiền	
expirationDate	date	ngày hết hạn đóng tiền	
paymented	number	số tiền đã đóng	

Bảng KHTC

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
KHTC_id	varchar2(5)	mã nhân viên phòng tài chánh	khóa chánh,khóa ngoại tham chiếu đến khóa chánh userID của bảng User

Bảng Role

Thuộc tính	Kiểu dữ liệu	Ý nghĩa	Ràng buộc
roleID	varchar2(10)	mã vai trò	khóa chánh

rolename	varchar2(30)	tên vai trò	

d. Mô tả ràng buộc toàn vẹn:

i. Ràng buộc khóa chánh:

RB1: Thuộc tính userID trong bảng Users là duy nhất.

RB2: Thuộc tính studentID trong bảng Student là duy nhất.

RB3: Thuộc tính classID trong bảng Class là duy nhất.

RB4: Thuộc tính LecturerID trong bảng Lecturer là duy nhất.

RB5: Thuộc tính facultyID trong bảng Faculty là duy nhất.

RB6:Thuộc tính PDT_id trong bảng PhongDaoTao là duy nhất.

RB7:Thuộc tính subjectID trong bảng subject là duy nhất.

RB8:Thuộc tính registrationID trong bảng Subject_Registration là duy nhất.

RB9:Thuộc tính ID_bangdiem trong bảng BangDiem là duy nhất.

RB10:Thuộc tính ID_DiemMonHoc trong bảng DiemMonHoc là duy nhất.

RB11:Thuộc tính ITdep_ID trong bảng IT_department là duy nhất.

RB12:Thuộc tính ID_ctsv trong bảng Ctsv là duy nhất.

RB13:Thuộc tính ID_scholarship trong bảng Scholarship là duy nhất.

RB14:Thuộc tính feeID trong bảng Fee là duy nhất.

RB15:Thuộc tính KHTC_id trong bảng KHTC là duy nhất.

RB16: Thuộc tính offeringID trong bảng Offering là duy nhất.

RB17:Thuộc tính semesterID trong bảng Semester là duy nhất.

ii. Ràng buộc khóa ngoại:

Bảng Student:

RB1: học sinh cũng là user của hệ thống.

RB2: học sinh học một lớp được quản lí qua mã lớp.

Bảng Class:

RB1: trưởng lớp (classPresident) là học sinh học lớp đó.

RB2: một lớp được quản lí bởi một khoa.

Bảng Lecturer

RB1: một giảng viên thuộc một khoa.

RB2: giảng viên cũng là user của hệ thống.

Bảng Faculty

RB1: trưởng khoa (dean) cũng là giảng viên của khoa đó.

Bảng PhongDaoTao

RB1: nhân viên phòng đào tạo cũng là user của hệ thống.

Bảng subject

RB1: một môn học được quản lí bởi một khoa (facultyID).

RB2: môn học tiên quyết (previousSubject) cũng là một môn học.

RB3: môn học chỉ được tạo (createdBy) và cập nhật (updatedBy) bởi nhân viên phòng đào tạo.

Bảng Offering

RB1: một học phần thuộc một môn học (subjectID).

RB2: một học phần được giảng dạy bởi một giảng viên (lecturerID).

Bảng Subject_Registration

RB1: đăng ký học phần (registeredBy) được thực hiện bởi sinh viên.

RB2: người phê duyệt (approvedBy) đăng ký học phần là nhân viên phòng đào tạo.

RB3: môn học thuộc học phần nào (offeringID).

Bảng BangDiem

RB1: bảng điểm thuộc về một sinh viên cụ thể.

Bảng DiemMonHoc

RB1: điểm được nhập bởi giảng viên (createdBy) dạy lớp đó.

RB2: điểm môn học nằm trong một bảng điểm môn học.

RB3: điểm môn học là của một môn học (subjectID) cụ thể.

Bảng IT_department

RB1: nhân viên phòng IT cũng là user của hệ thống.

Bảng Ctsv

RB1: nhân viên phòng ctsv cũng là user của hệ thống.

Bảng Scholarship

RB1: học bổng được phê duyệt bởi nhân viên phòng công tác sinh viên (approvedBy).

RB2: học bổng được cấp cho sinh viên cụ thể.

Bảng Fee

RB1: phí được đóng bởi sinh viên nào.

RB2: phí được thu bởi nhân viên phòng kế hoạch tài chánh.

Bảng KHCT (collectedBy)

RB1: nhân viên phòng kế hoạch tài chánh là user của hệ thống.

iii. Ràng buộc miền dữ liệu:

Bảng users

RB1: Thuộc tính userrole chỉ có các giá trị : student, it_deparment,ctsv,khtc,phongdaotao.

RB2: Thuộc tính gender chỉ có 1 trong 2 giá trị: male, female.

CHƯƠNG 3: CÁC CÔNG NGHỆ SỬ DỤNG

I. Oracle:

Cơ sở dữ liệu Oracle (Oracle DB) là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) từ Oracle Corporation. Được phát triển vào năm 1977 bởi Lawrence Ellison và các nhà phát triển khác, Oracle DB là một trong những công cụ cơ sở dữ liệu quan hệ được sử dụng rộng rãi và đáng tin nhất.

Hệ thống được xây dựng xung quanh bởi một framework cơ sở dữ liệu quan hệ, trong đó các đối tượng dữ liệu có thể được truy cập trực tiếp bởi người dùng (hoặc giao diện người dùng ứng dụng) thông qua ngôn ngữ truy vấn có cấu trúc (SQL). Oracle là một kiến trúc cơ sở dữ liệu quan hệ có thể mở rộng hoàn toàn và thường được sử dụng bởi các doanh nghiệp toàn cầu, họ quản lý và xử lý dữ liệu trên các mạng diện rộng và cục bộ. Cơ sở dữ liệu Oracle có thành phần mạng riêng để cho phép liên lạc qua các mạng.

Oracle DB còn được gọi là Oracle RDBMS hay gọi tắt, chỉ là Oracle.

II. JDBC (Java Database Connectivity):

JDBC, là viết tắt của Java Database Connectivity, là một Java API chuẩn để kết nối giữa ngôn ngữ lập trình Java và các cơ sở dữ liệu đa dạng. Khi sử dụng JDBC chúng ta có thể thực hiện nhiều tác vụ đa dạng khi làm việc với cơ sở dữ liệu như tạo, xóa cơ sở dữ liệu; tạo và thực thi các lệnh SQL hoặc MySQL; tạo, xóa các bản ghi; …

Trước khi có JDBC, ODBC API được sử dụng là API để kết nối và thực thi truy vấn với cơ sở dữ liệu. Nhưng API này có một hạn chế rõ ràng là nó sử dụng ODBC driver được viết bằng ngôn ngữ C (là phụ thuộc nền tảng và không an toàn).

Do đó, Java đã định nghĩa riêng API (là JDBC API) để thực hiện các thao tác khi làm việc với cơ sở dữ liệu, và tất nhiên các JDBC driver này được viết bằng ngôn ngữ Java để cung cấp các lợi thế rõ rệt của ngôn ngữ này.

JDBC API cung cấp các Class và Interface sau:

- DriverManager: Lớp này quản lý các Database Driver. Ánh xạ các yêu cầu kết nối từ ứng dụng Java với Data driver thích hợp bởi sử dụng giao thức kết nối phụ.
- Driver: Interface này xử lý các kết nối với Database Server. Hiếm khi, chúng ta tương tác trực tiếp với các đối tượng Driver này. Thay vào đó, chúng ta sử dụng các đối tượng DriverManager để quản lý các đối tượng kiểu này.
- Connection: Đối tượng Connection biểu diễn ngữ cảnh giao tiếp. Interface này chứa nhiều phương thức đa dạng để tạo kết nối với một Database.
- Statement: Chúng ta sử dụng các đối tượng được tạo từ Interface này để đệ trình các lệnh SQL tới Database. Ngoài ra, một số Interface kết thừa từ nó cung chấp nhận thêm các tham số để thực thi các thủ tục đã được lưu trữ.
- ResultSet: Các đối tượng này giữ dữ liệu được thu nhận từ một Database sau khi chúng ta thực thi một truy vấn SQL. Nó nóng vai trò như một Iterator để cho phép chúng ta vọc qua dữ liệu của nó.
- SQLException: Lớp này xử lý bất cứ lỗi nào xuất hiện trong khi làm việc với Database.

III. JavaSwing:

JavaSwing là một phần của JFC (Java Foundation Class), gồm một tập các API dùng để thiết kế giao diện cho các chương trình của Java. Được phát triển từ AWT, Swing về sau có những nhiều chức năng và linh hoạt hơn.

Ưu điểm của Swing so với AWT:

- Cung cấp thêm các đối tượng mới để xây dựng giao diện đồ họa.
- Look-and-feel: tùy biến để các thành phần giao diện của Swing nhìn giống như các thành phần giao diện của hệ điều hành.
- Hỗ trợ các thao tác sử dụng bàn phím thay chuột.

- Sử dụng tài nguyên hiệu quả hơn.

IV. Git:

Git là một hệ thống quản lý phiên bản (Version Control System) dùng để quản lý và kiểm tra các phiên bản mã nguồn khác nhau trong quá trình phát triển mã nguồn. Git sử dụng mô hình phân tán, ngược lại so với SVN hoặc CSV. Mỗi nơi lưu source sẽ được gọi là Repositories, không cần lưu trữ tập trung một nơi, mà mỗi thành viên trong team sẽ có một repository ở máy của riêng mình. Có 2 dạng là Local repository và Remote repositories. Lập trình viên có thể thực hiện mọi công việc trên bản sao (local) repository trên máy tính. Sau khi hoàn thiện công việc rồi có thể đẩy lên Remote repository đểchia sẻ cho người khác.

Git có nhiều ưu điểm, đặc biệt là giải quyết xung đột (conflict) khi chia tách và nhập các nhánh phần mềm. Trong khi với SVN, việc này luôn là ác mộng vì các xung đột trong quá trình tách/nhập, còn với Git việc này quá dễ dàng.

V. GitKraken:

GitKraken là giao diện hướng về phía client của Git cho các nhà phát triển phần mềm sử dụng Windows, Mac và Linux.

GitKraken đơn giản hóa các lệnh phức tạp thành các hành động kéo và thả. Nó làm việc với các kho lưu trữ từ xa dễ dàng hơn thông qua tích hợp với GitHub, Bitbucket và GitLab. Nó cho phép bạn giải quyết xung đột hợp nhất mà không cần rời khỏi ứng dụng. Và nó hỗ trợ Gitflow, Git Hook, LFS, v.v.

Trên hết, nó miễn phí cho mục đích phi thương mại, giáo dục và khởi nghiệp!

VI. NetBean:

NetBean là một IDE dành cho lập trình bằng ngôn ngữ Java, nó có thể chạy trên một số nền tảng như Windows, Linux, Solaris, OS X và các nền tảng khác hỗ trợ JVM hoặc tương thích với máy ảo Java.

Có một số mô-đun trong NetBeans IDE. Trình lược tả Netbeans để giám sát các ứng dụng Java chủ yếu được sử dụng để tìm rò rỉ bộ nhớ, tối ưu hóa tốc độ và làm được nhiều hơn thế. Công cụ GUI hoặc Graphical User Interface là một trong những công cụ thiết kế và phát triển tốt nhất để phát triển các prototype và thiết kế Swing GUI bằng cách kéo, thả các GUI Components.NetBean cũng đã tích hợp NetBeans JavaScript Editor, mở rộng hỗ trợ cho Ajax, CSS và JavaScript cùng với những thứ khác

CHƯƠNG 4: XÂY DỰNG VÀ QUẢN LÍ GIAO TÁC

I. Trigger:

1. Trigger trong Oracle:

Trigger là một chương trình lưu trữ nó sẽ tự động thực thi khi một vài sự kiện xảy ra. Thực tế Trigger được viết để phản hồi lại những sự kiện xảy ra khi thực hiện các lệnh dưới đây:

- Các câu lệnh thuộc nhóm lệnh kiểm soát dữ liệu (DELETE, INSERT, or UPDATE).
- Các câu lệnh thuộc nhóm lệnh định nghĩa dữ liệu DML (CREATE, ALTER, or DROP).
- Các câu lệnh hoạt động dữ liệu (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).
- Lợi ích khi sử dụng Trigger:
- Ngăn ngừa những giao tác không hợp lệ.
- Tự động tạo ra những cột dữ liệu phụ.
- Nhân rộng bảng đồng bộ.

Cú pháp:

```
CREATE [OR REPLACE ] TRIGGER trigger_name

{BEFORE | AFTER | INSTEAD OF }

{INSERT [OR] | UPDATE [OR] | DELETE}

[OF col_name]

ON table_name

[REFERENCING OLD AS o NEW AS n]
```

```
[FOR EACH ROW]
WHEN (condition)
DECLARE
 Declaration-statements
BEGIN
 Executable-statements
EXCEPTION
 Exception-handling-statements
END;
```

Giải thích:

- CREATE [OR REPLACE] TRIGGER trigger_name: tạo hoặc thay thế một trigger đã tồn tại thành trigger_name.
- {BEFORE | AFTER | INSTEAD OF }: cái này chỉ định khi nào trigger được thực thi. INSTEAD OF được sử dụng để tạo trigger trên view.
- {INSERT [OR] | UPDATE [OR] | DELETE}: chỉ định cụ thể lệnh nào thuộc loại lệnh DML được thực thi.
- [OF col_name]: chỉ định cụ thể cột nào sẽ được cập nhật
- ON table_name: chỉ định trigger sẽ được thực thi trên bảng nào.
- [REFERENCING OLD AS o NEW AS n]: cho phép dùng giá trị new hay old cho các lệnh DML, như là Insert, Update, Delete.
- [FOR EACH ROW]: cho biết cụ thể trigger sẽ thực thi trên từng dòng.
- WHEN (condition): cho biết dòng đó có điều kiện gì thì trigger mới thực thi được.

2. Danh sách các trigger:

STT	Tên	Thao tác	Bảng	Nội dung	Ghi chú
1.	increase_classSize	INSERT OR UPDATE	Student	cập nhật sỉ số lớp	
2.	decrease_classSize	DELETE	Student	cập nhật sỉ số lớp	
3.	update_score_DiemMonHoc	UPDATE	DIEMMONHOC	cập nhật điểm môn học	
4.	Check_heso_subject	INSERT OR UPDATE	Subject	kiểm tra hệ số khi thêm môn	
5.	update_expirationDateForNewInsert	INSERT	FEE	Cập nhật ngày hết hạn đóng học phí khi thêm mới vào bảng FEE.	

3. Mô tả một số trigger :

a. Cập nhật điểm cho sinh viên:

Tên: update_score_DiemMonHoc

Thao tác: update

Trên bảng: DIEMMONHOC

Mã PL/SQL:

```
CREATE OR REPLACE TRIGGER update_score_DiemMonHoc BEFORE
UPDATE ON DIEMMONHOC
```

```
FOR EACH ROW

DECLARE

    v_HeSoGK number;

    v_HeSoQT number;

    v_HeSoTH number;

    v_HeSoCK number;

BEGIN

    SELECT s.HESODIEMQT,s.HESODIEMGK,s.HESODIEMTH,s.HESODIEMCK

    INTO v_HeSoQT,v_HeSoGK,v_HeSoTH,v_HeSoCK

    FROM SUBJECT s

    WHERE s.SUBJECTID = :NEW.ID_monHoc;

    IF(v_HeSoQT = 0)

    THEN

        :NEW.diemQT := 0;

    END IF;

    IF(v_HeSoGK = 0)

    THEN

        :NEW.diemGK := 0;

    END IF;

    IF(v_HeSoTH = 0)

    THEN

        :NEW.diemTH := 0;

    END IF;

    IF(v_HeSoCK = 0)

    THEN
```

```
        :NEW.diemCK := 0;
    END IF;
    :NEW.score := :NEW.diemQT * v_HeSoQT + :NEW.diemGK *
v_HeSoGK + :NEW.diemTH * v_HeSoTH + :NEW.diemCK * v_HeSoCK;
END;
```

Bước thực hiện:

[1]: Lấy ra các hệ số của môn học đó.

[2]: Nếu hệ số nào bằng 0 thì cập nhật điểm tương ứng với hệ số đó cũng bằng 0.

[3]: Cập nhật điểm trung bình môn = điểm QT * hệ số QT + điểm GK * hệ số GK + điểm TH * hệ số TH + điểm CK * hệ số CK.

b. Cập nhật ngày hết hạn đóng học phí khi thêm record vào bảng Fee:

Tên: update_expirationDateForNewInsert

Thao tác: INSERT

Trên bảng: Fee

Mã PL/SQL:

```
CREATE OR REPLACE TRIGGER update_expirationDateForNewInsert
FOR INSERT ON fee
COMPOUND TRIGGER
--declare create value
    v_expirationDate date;
BEFORE EACH ROW IS
BEGIN
```

```
    LOCK TABLE fee IN SHARE MODE;

    SELECT expirationdate

    INTO v_expirationDate

    FROM fee

    WHERE fee.SEMESTER = :NEW.semester AND fee.expirationdate
IS NOT NULL

    GROUP BY expirationdate;


    :NEW.expirationdate := v_expirationDate;


    EXCEPTION WHEN no_data_found

    THEN

        dbms_output.put_line('Khong co ngay het han');

END BEFORE EACH ROW;

END;
```

Bước thực hiện:

[1]: Kiểm tra học phí học kỳ đó đã có ngày hết hạn đóng học phí hay chưa

[1.1]: Nếu có thì cập nhật ngày hết hạn vào record trước khi thêm vào bảng.

[1.2]: Nếu không thì không làm gì hết.

II. Stored Procedure:

1. Stored Procedure trong Oracle:

Stored procedure là một hoặc một tập các lệnh SQL được biên dịch để thực hiện cùng lúc mỗi khi được gọi đến. Về bản chất giống với hàm, thủ tục trong các ngôn ngữ lập trình khác.

Stored procedure giúp giảm thời gian giao tiếp giữa các ứng dụng với hệ quản trị, bởi vì thay vì gửi nhiều câu lệnh dài thì ta chỉ cần gọi tới một thủ tục và trong thủ tục này sẽ thực hiện nhiều câu lệnh SQL.

Cú pháp:

```
CREATE [OR REPLACE] PROCEDURE procedure_name

[(parameter_name [IN | OUT | IN OUT] type [, ...])]

{IS | AS}

BEGIN

 < procedure_body >

END procedure_name;
```

Giải thích:

- procedure_name : tên của procedure
- [OR REPLACE]: thay thế những procedure đang tồn tại.
- IN: cho phép nhập tham số. OUT: giá trị trả về cụ thể của procedure.
- < procedure_body >: chứa những câu lệnh của procedure.
- AS được sử dụng thay vì IS để tạo ra những procedure độc lập.

2. Transaction trong Oracle:

a. Định nghĩa:

Transaction là một đơn vị làm việc có tính chất logic. Chứa một hoặc nhiều câu lệnh SQL. Ảnh hưởng của tất cả các câu lệnh SQL trong một transaction có thể là commit tất cả (áp dụng cho cơ sở dữ liệu) hoặc rollback tất cả(hoàn tác từ cơ sở dữ liệu).

b. Cấu trúc transaction:

Gồm 1 hay nhiều câu truy vấn, có thể bao gồm DDL,DML.

Có bắt đầu và kết thúc.

Bắt đầu transaction:

- Transaction bắt đầu khi câu lệnh SQL đầu tiên được thực thi, bao gồm DDL,DML hoặc lệnh SET TRANSACTION.
- TRANSACTION NAME: Đặt tên cho transaction, bắt đầu transasction.
- SET TRANSACTION NAME <Tên transaction> (trước lệnh này nên có 1 lệnh COMMIT).

```
SET TRANSACTION NAME 'ten_transaction';
```

- Khi 1 transaction mới bắt đầu , hệ quản trị oracle sẽ gán nó vào **undo data segment** (ghi nhận lại các thao tác của transaction trước khi commit, để có thể rollback khi có lỗi).

Kết thúc transaction:

- Transaction có thể kết thúc trong nhiều trường hợp:
  - Gặp lệnh COMMIT hoặc ROLLBACK mà không có savepoint.
  - Gặp các câu lệnh DDL như create, drop, rename, alter.
  - User ngắt kết nối đến hệ quản trị đột ngột, transaction sẽ tự động commit.
  - Các ứng dụng đang kết nối đến hệ quản trị bị dừng đột ngột, transaction sẽ tự động rollback.

```
BEGIN
  COMMIT;
  SET TRANSACTION NAME 'update_salary';
  UPDATE EMPLOYEE
  SET salary = salary + 500000
```

```
    where EmpNo = 1;

    COMMIT; -- hoặc ROLLBACK

    END;
```

- Một transaction đang thực thi là 1 transaction đã bắt đầu nhưng chưa được COMMIT hoặc ROLLBACK.
- Như ví dụ trên câu lệnh đầu tiên trong transaction **'update_salary'** là cập nhật lương của nhân viên có mã nhân viên là **1** . Từ lúc thực hiện câu lệnh **update** này cho đến câu lệnh **COMMIT** kết thúc transaction (khoảng này được gọi là phạm vi Transaction) , transaction **'update_salary'** đã được kích hoạt.

c. Transaction control:

Gồm các lệnh để quản lý sự thay đổi của DML lên database, gồm 1 số lệnh chính:

- SAVEPOINT: Xác định 1 điểm trong transaction để rollback về khi có sự cố.
- COMMIT: Kết thúc transaction, **lưu thay đổi vĩnh viễn**, xóa tất cả SAVEPOINT, mở transaction locks.
- ROLLBACK: phục hồi lại dữ liệu trước khi thay đổi.

ROLLBACK:

- Hoàn tác mọi thay đổi.
- Mở tất cả khóa.
- Xóa toàn bộ savepoints.
- Kết thúc transaction.

```
BEGIN

    COMMIT;

    set TRANSACTION NAME 'tadda';
```

```
    INSERT INTO Employee VALUES
(10,'Name10',TO_DATE('1/1/1998','dd/mm/yyyy'),8,
3,TO_DATE('1/1/2000','dd/mm/yyyy'),2000000,1,1,'Note1
','mail10@com.vn');

    INSERT INTO Employee VALUES
(12,'Name12',TO_DATE('1/1/1998','dd/mm/yyyy'),8,
3,TO_DATE('1/1/2000','dd/mm/yyyy'),2500000,1,1,'Note1
','mail12@com.vn');

    EXCEPTION WHEN DUP_VAL_ON_INDEX

    THEN

        ROLLBACK;

        dbms_output.put_line('abc');

END;
```

- COMMIT:

```
BEGIN

    UPDATE EMPLOYEE

    set LEVEL_EMPLOYEE = 3

    WHERE EmpNo = 3;

    COMMIT;

    ROLLBACK;

END;
```

COMMIT lưu giá trị vĩnh viễn ?

- Trong ví dụ trên nếu **không** có lệnh COMMIT thì khi chạy đến lệnh ROLLBACK , level_employee của nhân viên có mã nhân viên = 3 sẽ bị hoàn tác lại lúc chưa cập nhật lên level_employee = 3.

- Còn khi có lệnh COMMIT ở trước lệnh ROLLBACK thì khi chạy đến lệnh ROLLBACK thì dữ liệu vẫn không thể hoàn tác lại . vì vậy sau khi chạy xong thì level_employee = 3.

SAVEPOINTS:

- Savepoints là một điểm được người dùng khai báo trong phạm vi transaction.
- Savepoints chia một Transaction thành các phần nhỏ hơn.

ROLLBACK TO SAVEPOINT

- Việc ROLLBACK lại Savepoint trong transaction chưa được COMMIT có nghĩa là hoàn tác lại mọi thay đổi được thực hiện sau Savepoint , điều này không có nghĩa là sẽ Rollback lại toàn bộ Transaction.

```
BEGIN

    UPDATE EMPLOYEE

    set Salary = Salary + 500000

    WHERE EmpNo = 1;

    SAVEPOINT diem_1;

    UPDATE EMPLOYEE

    set Salary = 0

    WHERE EmpNo = 1;

    SAVEPOINT diem_2;

    COMMIT;

END;
```

- Khi một transaction Rollback lại Savepoint như ROLLBACK TO diem_1 ở ví dụ trên:

- Dữ liệu chỉ được phục hồi lại các câu lệnh ở sau savepoint diem_1, vì vậy mức lương của nhân viên 1 sẽ được hoàn tác lại **trước** khi cập nhật mức lương của nhân viên này = 0 .
- Oracle sẽ lưu trữ lại savepoint được chỉ định trong câu lệnh Rollback to savepoint trong trường hợp này là savepoint 'diem_1', nhưng tất cả savepoint tiếp theo đều bị mất (savepoint diem_2) .

HÀNG ĐỢI TRANSACTIONS: Transaction đang đợi tài nguyên bị khóa sẽ bị block, nó sẽ xếp vào hàng đợi của transaction đang giữ tài nguyên đó. Transaction đang giữ tài nguyên phải commit hoặc rollback để transaction bị khóa tiếp tục thực thi.

T	Session 1	Session 2	Session 3	Explanation
T0	`UPDATE employees SET salary=7000 WHERE last_name='Banda';`			Session 1 bắt đầu 1 giao tác. Session đặt 1 khóa riêng biệt lên dòng Banda (TX)và khóa bán riêng biệt lên bảng (SX)
T1	`SAVEPOINT after_banda_sal;`			Session 1 tạo một samepoints tên là after_banda_sal;
T2	`UPDATE employees SET salary=12000 WHERE last_name='Greene';`			Session 1 khóa dòng Greene

T3		UPDATE employees SET salary=14000 WHERE last_name='Greene';		Session 2 muốn update dòng Greene nhưng không thành công vì Session 1 đang giữ khóa. Do đó không có giao tác nào bắt đầu tại Session 2
T4	ROLLBACK TO SAVEPOINT after_band a_sal;			Session 1 rollback việc cập nhật lương cho dòng Greene tại dòng đã nhả khóa cho dòng Greene, khóa bảng nhận được tại t0 không đươc nhả Session 2 vẫn còn bị khóa bởi Session 1 bởi vì Session 2 được xếp vào trong hàng đợi trên giao tác của Session 1, giao tác đó vẫn chưa được hoàn tất
T5			UPDATE employees SET salary=11000	Dòng Greene đã được mở khóa vì vậy Session 3 nhận khóa giúp cho việc cập nhật tại dòng Greene .Dòng lệnh này bắt

			WHERE last_name= 'Greene';	đầu giao tác của Session 3
t6	Commit;			Session 1 được commit, kết thúc giao tác của Session 1. Session 2 được thêm vào hàng đợi để cập nhật cho dòng Greene sau giao tác của Session 3

AUTONOMOUS TRANSACTION:

- AUTONOMOUS TRANSACTION là một Transaction độc lập có thể được gọi từ Transaction chính. Có thể tạm dừng Transaction chính, thực hiện các câu lệnh SQL và COMMIT hoặc ROLLBACK trong Autonomouse Transaction và rồi sau đó tiếp tục Transaction chính.
- Autonomouse Transaction có những đặc điểm:
  - Autonomouse Transaction không thể thấy được những thay đổi chưa được COMMIT của Transaction chính và không chia sẻ các khóa hoặc tài nguyên với Transaction chính .

```
CREATE OR REPLACE PROCEDURE test_autonomous

AS

    PRAGMA AUTONOMOUS_TRANSACTION;

BEGIN
```

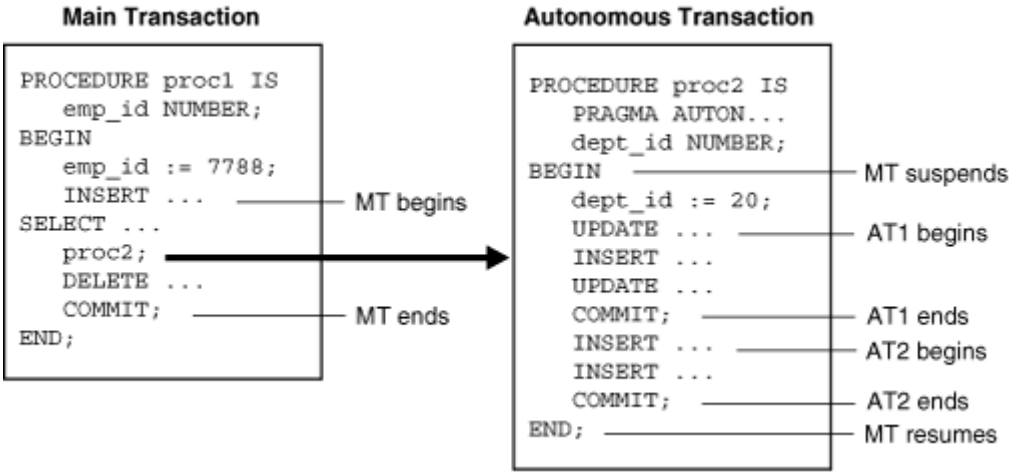
```
    UPDATE EMPLOYEE
    set EMPLOYEE.SALARY = EMPLOYEE.SALARY + 700000
    WHERE EMPLOYEE.EMPNO = 1;
    ROLLBACK;
END;

BEGIN
    UPDATE EMPLOYEE
    set EMPLOYEE.SALARY = EMPLOYEE.SALARY + 500000
    WHERE EMPLOYEE.EMPNO = 1;
    test_autonomous;
    COMMIT;
END;
```

T	Session 1	Session 2	Explanation
T0	`UPDATE EMPLOYEE` `set SALARY = SALARY + 500000`		Session 1 cập nhật lương cho nhân viên 1, Session 1 đã lock hàng nhân viên 1 này.
T1	`test autonomous;`		Session 1(Transaction Chính) gọi procedure **`test_autonomous`** (procedure

			này là 1 Autonomous Transaction)
T2		`UPDATE EMPLOYEE set SALARY = SALARY + 700000`	Session 2 **t_autonomous**) cập nhật lương cho nhân viên 1, hiện hàng nhân viên 1 này đang bị khóa bởi Session 1.
T3	`Error at line 1` `ORA-00060: deadlock detected while waiting for resource`		Kết quả là 2 Session này bị deadlock, vì khi Session 1 gọi **test_autonomous** thì Session 1 sẽ bị tạm dừng để cho **test_autonomous** thực thi, nhưng Transaction chính(S1) không chia sẻ khóa, tài nguyên với Autonomous

			Transaction(S2) nên S2 không có khóa để thực hiện . Cứ như vậy S1 chờ S2 thực hiện xong nhưng S2 lại chờ khóa từ S1 dẫn đến deadlock.

- Những thay đổi trong Autonomouse Transaction khi đã COMMIT thì sẽ được hiển thị cho các Transaction khác . Do đó người dùng có thể truy cập thông tin đã cập nhật trong Autonomouse Transaction mà không cần Transaction Chính (Transaction gọi Autonomouse Transaction)được COMMIT .
- Autonomose Transaction có thể gọi các Autonomose Transaction khác và không giới hạn số lần gọi .
- Autonomouse Transaction được khai báo bởi "*pragma AUTONOMOUS_TRANSACTION" ,*
- "*pragma AUTONOMOUS_TRANSACTION"* chỉ thị cho cơ sở dữ liệu rằng Transaction này khi được thực hiện sẽ được thực hiện như một Transaction mới độc lập với Transaction chính



3. Danh sách các Procedure:

STT	Tên	Tham số đầu vào	Tham số đầu ra	Ý nghĩa	Ghi chú
1.	INSERT_SUBJECT_REGISTRATION	in_studentID varchar2(10) , in_offeringID varchar2(10) , in_semester varchar(32), in_PDT_ID varchar2(10).		thao tác thêm vào bảng SUBJECT_REGISTRATION làm đăng ký học phần.	
2.	DELETE_SUBJECT_REGISTRATION	in_studentID varchar2(10) , in_offeringID varchar2(10) , in_semester varchar2(32).		xóa đăng kí học phần	
3.	Update_diem	in_studentID varchar2(10), in_semester varchar2(32), in_offeringID varchar2(10), in_lectureID varchar2(10), in_diemQT number, in_diemGK		cập nhật điểm cho sinh viên	

		number , in_diemTH number, in_diemCK number.			
4.	print_Fee	in_StudentID varchar2(10).		in ra số học phí	
5.	sleep	in_time number.		ngày nghỉ	
6.	getUserInfo	query_userID varchar2(10).		lấy thông tin user	
7.	Update_ExpirationDate_FEE	in_semester varchar2(32), in_date varchar2.		tạo ngày hết hạn cho học phí	
8.	collectFee	feeid_in varchar2(10), khtcid varchar2(10), money_in number.		thu học phí sinh viên, cập nhật thuộc tính payment của bảng fee	
9.	createOffering	subjectID varchar2(10), semester varchar2(32), lecture varchar2(32), slot number.		tạo mới một học phần ở bảng offering, tự động tạo id để insert	

4. Mô tả một số Stored Procedure :

a. Đăng kí học phần:

Tên: INSERT_SUBJECT_REGISTRATION.

Nội dung: Đăng ký học phần.

Tham số đầu vào: in_studentID, in_PDT_ID, in_offeringID, in_semester.

Tham số đầu ra : không có.

Mã PL/SQL:

```
CREATE OR REPLACE PROCEDURE INSERT_SUBJECT_REGISTRATION
(in_studentID STUDENT.STUDENTID%TYPE
  , in_PDT_ID PHONGDAOTAO.PDT_ID%TYPE
  , in_offeringID OFFERING.OFFERINGID%TYPE, in_semester
OFFERING.SEMESTER%TYPE)
AS
    v_Score DIEMMONHOC.SCORE%TYPE;
    v_preSubject SUBJECT.PREVIOUSSUBJECT%TYPE;
    v_countSubject number;
    v_SubjectID SUBJECT.SUBJECTID%TYPE;
    cur_BangDiemID BANGDIEM.ID_BANGDIEM%TYPE;
    v_status number := 0;
    v_status2 number := 0;
    v_status3 number := 0;
    v_idBangDiem BANGDIEM.ID_BANGDIEM%TYPE;
    v_lectureID LECTURE.LECTUREID%TYPE;
    v_date varchar2(255);
    v_registered number;
    v_slot number;
    v_hocphi number;
    v_tinchi number;
    v_studentID STUDENT.STUDENTID%TYPE;
    CURSOR cur IS SELECT ID_BANGDIEM
                  FROM BANGDIEM
```

```
                  WHERE BANGDIEM.STUDENTID = in_studentID;
BEGIN
       SELECT PREVIOUSSUBJECT
       INTO v_preSubject
       FROM SUBJECT S, OFFERING O
       WHERE S.SUBJECTID = O.SUBJECTID AND O.OFFERINGID =
in_offeringID;

       IF(v_preSubject IS NOT NULL)
       THEN
            OPEN cur;
            LOOP
                FETCH cur INTO cur_BangDiemID;
                EXIT WHEN cur%NOTFOUND;
                BEGIN
                    SELECT score
                    INTO v_Score
                    FROM BANGDIEM,DIEMMONHOC
                    WHERE BANGDIEM.ID_BANGDIEM =
DIEMMONHOC.ID_BANGDIEM
                    AND BANGDIEM.ID_BANGDIEM =
cur_BangDiemID
                    AND BANGDIEM.STUDENTID = in_studentID
                    AND DIEMMONHOC.ID_MONHOC = v_preSubject;
                EXCEPTION
                    WHEN NO_DATA_FOUND
                    THEN
                        v_Score:= NULL;
                END;
                IF(v_Score > 5)
                THEN
                    v_status := 1;
                    --RAISE_APPLICATION_ERROR(-20000, 'Dieu
Kien Tien Quyet Khong Duoc');
                END IF;
```

```
            END LOOP;
            IF(v_status = 0)
            THEN
            RAISE_APPLICATION_ERROR(-20000, 'Dieu Kien Tien
Quyet Khong Duoc');
            END IF;
       END IF;

       SELECT subjectID,LectureID,slot
       INTO v_SubjectID,v_lectureID,v_slot
       FROM OFFERING
       WHERE OFFERING.OFFERINGID = in_offeringID;

       SELECT COUNT(*)
       INTO v_countSubject
       FROM SUBJECT_REGISTRATION sr, OFFERING o, SUBJECT s
       WHERE sr.REGISTEREDBY= in_studentID
       AND sr.OFFERINGID = o.OFFERINGID
       AND o.SUBJECTID = s.SUBJECTID
       AND o.SUBJECTID = v_SubjectID ;

       IF(v_countSubject > 0)
       THEN
            RAISE_APPLICATION_ERROR(-20001, 'Mon nay da duoc
dang ky');
       END IF;

       SELECT COUNT(offeringID)
       INTO v_registered
       FROM SUBJECT_REGISTRATION
       WHERE offeringID = in_offeringID;

       IF(v_registered >= v_slot)
       THEN
            RAISE_APPLICATION_ERROR(-20002, 'Lop Da Full');
```

```
        END IF;

        SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
        BEGIN
             SELECT ID_BangDiem
             INTO v_idBangDiem
             FROM BangDiem
             WHERE BangDiem.STUDENTID = in_studentID AND
semester = in_semester;
             EXCEPTION
             WHEN NO_data_found
             THEN
                 v_status2 := 1;
        END;
        IF(v_status2 = 1)
        THEN
             v_idBangDiem := GETBANGDIEMID();
             INSERT INTO BANGDIEM VALUES
(v_idBangDiem,in_studentID,in_semester);
        END IF;

        SELECT SUBJECT.NUMBEROFCREDITS
         INTO v_tinchi
         FROM SUBJECT
         WHERE SUBJECT.SUBJECTID = v_SubjectID;

         v_hocphi := v_tinchi * 500000;

         BEGIN
             SELECT FEE.STUDENTID
             INTO v_studentID
             FROM FEE
             WHERE FEE.STUDENTID = in_studentID AND
FEE.SEMESTER = in_semester;
             EXCEPTION
```

```
            WHEN NO_data_found
            THEN
                v_status3 := 1;
        END;
        IF(v_status3 = 1)
        THEN
            INSERT INTO FEE VALUES
(FEE_ID_SEQ.NEXTVAL,v_hocphi,in_semester,in_studentID,0,NULL
,NULL,NULL,NULL);
        ELSE
            UPDATE fee
            set MONEY = MONEY + v_hocphi
            WHERE fee.STUDENTID = in_studentID
            AND fee.SEMESTER = in_semester;
        END IF;

       INSERT INTO SUBJECT_REGISTRATION VALUES
(registrationID_SEQ.NEXTVAL,in_studentID,in_PDT_ID,in_offeri
ngID,in_semester);
       SELECT TO_CHAR(SYSDATE, 'DD-MM-YYYY') INTO v_date
FROM dual;
       dbms_output.put_line(v_date);
       INSERT INTO DIEMMONHOC
VALUES(DiemMonHoc_ID_SEQ.NEXTVAL,0,0,0,0,0,v_lectureID,TO_DA
TE(v_date,'DD-MM-YYYY'),v_idBangDiem,v_SubjectID);
       COMMIT;
END;
```

Bước thực hiện:

[1]: Kiểm tra học phần này có môn học tiên quyết hay không.

[1.1]: Nếu có, thì kiểm tra trong các học kỳ trước, môn này đã được học và qua môn chưa, nếu chưa thì sẽ báo lỗi “Không thỏa điều kiện tiên quyết”.

[1.2]: Nếu không thì tiếp tục thực hiên.

[2]: Kiểm tra trong học kỳ này đã được đăng ký hay chưa.

[2.1]: Nếu đã đăng ký rồi thì sẽ báo lỗi “Môn học này đã được đăng ký”

[2.2]: Nếu không có thì tiếp tục thực hiên.

[3]: Kiểm tra học phần này đã đầy hay chưa.

[3.1]: Nếu đã đầy thì sẽ báo lỗi “Lớp đã đầy”.

[3.2]: Nếu chưa thì tiếp tục thực hiên.

[4]: Kiểm tra sinh viên trong học kỳ này đã được tạo bảng điểm hay chưa.

[4.1]: Nếu chưa thì insert vào bảng BangDiem dữ liệu tương ứng với sinh viên và học kỳ.

[4.2]: Nếu có rồi thì tiếp tục thực hiên.

[5]: Kiểm tra sinh viên trong học kỳ này đã có dữ liệu về học phí hay chưa.

[5.1]: Nếu chưa thì insert vào bảng Fee với dữ liệu học phí = số tín chỉ của môn học đó * 500.000.

[5.2]: Nếu rồi thì cập nhật dữ liệu học phí = học phí cũ + số tín chỉ của môn học đó * 500.000.

[6]: Thực hiện insert dữ liệu vào bảng SUBJECT_REGISTRATION.

[7]: Thực hiện insert dữ liệu vào bảng DIEMMONHOC tương ứng với bảng điểm đã tạo ở trên hoặc có sẵn để hoàn tất thủ tục.

b. Lấy ra điểm trung bình học kỳ của sinh viên:

Tên: get_diemHP.

Nội dung: Lấy ra điểm trung bình học kỳ của sinh viên.

Tham số đầu vào: in_studentID, in_semester.

Tham số đầu ra : v_avengeScore number.

Mã PL/SQL:

```
CREATE OR REPLACE FUNCTION get_diemHP(in_studentID
STUDENT.STUDENTID%TYPE, in_semester BANGDIEM.SEMESTER%TYPE)
RETURN number
AS
    v_numberOfCredits number;
    v_score number;
    v_tinchi number;
    v_avengeScore number := 0;
    cur_dmhID DIEMMONHOC.ID_DIEMMONHOC%TYPE;
    CURSOR cur IS SELECT ID_DIEMMONHOC
                  FROM bangdiem bd, diemmonhoc dmh
                  WHERE bd.STUDENTID = in_studentID AND
bd.SEMESTER = in_semester AND dmh.ID_BANGDIEM =
bd.ID_BANGDIEM;
BEGIN
    SELECT SUM(NUMBEROFCREDITS)
    INTO v_numberOfCredits
    FROM BANGDIEM bd, SUBJECT sub, DIEMMONHOC dmh
    WHERE bd.STUDENTID = in_studentID AND bd.SEMESTER =
in_semester AND bd.ID_BANGDIEM = dmh.ID_BANGDIEM
    AND dmh.ID_MONHOC = sub.SUBJECTID;

    OPEN cur;
    LOOP
        FETCH cur INTO cur_dmhID;
        EXIT WHEN cur%NOTFOUND;

        SELECT dmh.SCORE, s.NUMBEROFCREDITS
        INTO v_score,v_tinchi
        FROM DIEMMONHOC dmh, subject s
        WHERE dmh.ID_DIEMMONHOC = cur_dmhID AND
dmh.ID_MONHOC = s.SUBJECTID;

        v_avengeScore := v_avengeScore + (v_score*v_tinchi);
```

```
    END LOOP;

    v_avengeScore := v_avengeScore/v_numberOfCredits;

    RETURN v_avengeScore;
END;
```

Bước thực hiện:

[1]: Lấy ra điểm trung bình môn và số tín chỉ của từng môn học trong bảng điểm của sinh viên tương ứng với học kỳ.

[2]: Thực hiện tính tổng điểm += điểm trung bình * tín chỉ.

[3]: Thực hiện tính điểm trung bình học kỳ = tổng điểm / tổng số tín chỉ.

c. Lấy ra danh sách sinh viên dự kiến nhận học bổng trong học kỳ:

Tên: getListScholarship

Nội dung: Lấy ra danh sách sinh viên dự kiến nhận học bổng trong học kỳ.

Tham số đầu vào: in_semester.

Tham số đầu ra : v_tableTemp table_Scholarship.

Mục tiêu:

- Những sinh viên có khả năng nhận được học bổng khuyến khích học tập, phải đạt tối thiểu các chỉ tiêu sau:
- Sinh viên có điểm trung bình học kỳ trên 7.0.
- Sinh viên không có rớt môn học nào trong học kỳ.
- Sinh viên phải đăng ký tối thiểu 14 tín chỉ trong học kỳ.
- Mỗi lớp chỉ có tối đa 3 sinh viên được nhận học bổng, thứ tự ưu tiên sắp xếp theo điểm trung bình học kỳ từ cao xuống thấp.

Mã SQL:

```
CREATE OR REPLACE FUNCTION getListScholarship(in_semester
SEMESTER.SEMESTERID%TYPE)
```

```
RETURN table_Scholarship

AS

    v_tableTemp table_Scholarship;

BEGIN


    SELECT row_Scholarship(sub2.FULLNAME, sub2.STUDENTID,
sub2.CLASSNAME, sub2.diem)

    BULK COLLECT INTO v_tableTemp

    FROM

        (SELECT sub.FULLNAME, sub.STUDENTID,
sub.CLASSNAME, get_diemHP(sub.STUDENTID,sub.SEMESTER) AS
diem, ROW_NUMBER() OVER(PARTITION BY sub.CLASSNAME ORDER
BY get_diemHP(sub.STUDENTID,sub.SEMESTER) desc) AS rn

            FROM

                (SELECT us.FULLNAME, st.STUDENTID,
cl.CLASSNAME, bd.SEMESTER


                FROM users us, student st, class cl,
SUBJECT sub, bangdiem bd, diemmonhoc dmh


                WHERE st.STUDENTID = us.USERID AND
st.STUDENTID = bd.STUDENTID AND st.CLASSID = cl.CLASSID

                AND bd.SEMESTER = in_semester AND
bd.ID_BANGDIEM = dmh.ID_BANGDIEM AND dmh.ID_MONHOC =
sub.SUBJECTID AND

                NOT EXISTS (SELECT
diemmonhoc.ID_DIEMMONHOC

                            FROM diemmonhoc
```

```
                                WHERE diemmonhoc.ID_BANGDIEM
= dmh.ID_BANGDIEM
                                AND diemmonhoc.SCORE < 5 )
                GROUP BY us.FULLNAME, st.STUDENTID,
cl.CLASSNAME, bd.SEMESTER
                HAVING SUM(NUMBEROFCREDITS) >= 14) sub
            WHERE get_diemHP(sub.STUDENTID,sub.SEMESTER)
> 7) sub2
    WHERE sub2.RN <= 3;
RETURN v_tableTemp;
END;
```

Bước thực hiện:

Việc sử dụng function `get_diemHP` để lấy điểm trung bình tương ứng với từng sinh viên và học kỳ đó ngay tại function `getListScholarship` giúp tránh được việc truy vấn nhiều lần.

[1]: ở query con đầu tiền: ta sẽ thực hiện truy vấn ra danh sách sinh viên không bị rớt môn và đăng ký hơn 14 chỉ trong học kỳ.

[2]: ở query con thứ 2: ta sẽ thực hiện truy vấn lồng trên query đầu tiên lấy ra danh sách sinh viên có điểm trung bình học kỳ > 7 (dùng function get_diemHP để lấy điểm trung bình học kỳ của từng sinh viên tương ứng với học kỳ).

[3]: ở query thứ 3 : ta thực hiện truy vấn lồng trên query thứ 2, lấy ra những sinh viên có điểm cao nhất trong mỗi lớp (mỗi lớp tối đa có 3 sinh viên) và không lấy trùng với những sinh viên đã được nằm trong diện nhận học bổng (sinh viên có trong bảng Scholarship).

CHƯƠNG 5: XỬ LÝ TRUY XUẤT ĐỒNG THỜI

I. Các mức cô lập trong Oracle:

1. Read Committed :

a. Trong Read Committed , mọi truy vấn được thực hiện bởi một transaction chỉ nhìn thấy dữ liệu đã được commit. Mức cô lập này là mặc định. Nó phù hợp với môi trường cơ sở dữ liệu trong đó ít transaction có khả năng xung đột.

b. Xung đột ghi trong Read Committed:

i. Trong Read Committed, một xung đột ghi xảy ra khi transaction cố gắng thay đổi một dòng đã được cập nhật bởi một transaction khác mà chưa được commit hay rollback.

ii. Transaction ngăn chặn sửa đổi dòng được gọi là blocking transaction. Transaction Read Committed chờ blocking transaction kết thúc và giải phóng khóa rồi transaction Read Committed mới lấy được khóa và thực hiện thay đổi.

2. Serializable:

a. Ở mức cô lập tuần tự (serializable isolation level), transaction chỉ thấy những thay đổi đã được commit trước thời điểm transaction có mức cô lập tuần tự này bắt đầu và các thay đổi được thực hiện bởi chính transaction này.

b. Một transaction tuần tự hoạt động trong một môi trường chỉ một mình nó như thể không có người dùng nào khác đang sửa đổi dữ liệu trong cơ sở dữ liệu. mức cô lập tuần tự phù hợp với các môi trường:

i. Với cơ sở dữ liệu lớn và các transaction ngắn chỉ cập nhật một vài dòng.

ii. Trường hợp có 2 transaction đồng thời sửa đổi cùng 1 dòng là tương đối thấp.

iii. Trường hợp các transaction tương đối dài chủ yếu chỉ đọc.

c. Ở mức cô lập tuần tự, Bất kỳ dòng nào được đọc bởi transaction tuần tự được đảm bảo giống nhau khi đọc lại. Bất kỳ truy vấn nào cũng được đảm bảo trả về cùng kết quả trong suất thời gian transaction đó hoạt động, do đó, những thay đổi được thực hiện bởi transaction khác sẽ không hiển thị đối với truy vấn của transaction tuần tự bất kể những thay đổi của transaction khác đã chạy được bao lâu. Transaction tuần tự (serializable) ngăn ngừa được Dirty reads, Nonrepeatable reads, Phantom reads.

d. Trong cơ sở dữ liệu Oracle, nếu transaction Tuần tự thực hiện thay đổi 1 dòng nào đó mà dòng này đã được thay đổi bởi một transaction khác đã commit sau khi transaction tuần bắt đầu thực thi thì sẽ xuất hiện 1 lỗi đó là :

```
ORA-08177: Cannot serialize access for this
transaction
```

3. Read-only:

Mức cô lập Read-only tương tự như mức cô lập Serializable, nhưng các transaction có mức cô lập Read-only chỉ được đọc dữ liệu, không được sửa đổi dữ liệu trong transaction (trừ trường hợp là người dùng là SYS).

II. Cơ chế khóa:

1. "Khóa" là một cơ chế giúp ngăn chặn các tương tác "có hại" đến hệ thống.Vậy, thế nào gọi là tương tác "có hại" ?

2. Tương tác gọi là "có hại" khi nó cập nhật dữ liệu sai hoặc thay đổi sai cấu trúc dữ liệu, khiến dữ liệu không nhất quán giữa, thường xảy ra khi nhiều

transaction cùng thao tác lên 1 đơn vị dữ liệu. Vì vậy, cơ chế khóa đóng vai trò cực kì quan trọng để đảm bảo cơ sở dữ liệu đồng thời và nhất quán.

3. Điều gì xảy ra bên trong cơ chế khóa ?

a. Hệ quản trị có thể thực thi một vài cơ chế khóa, tùy vào các thao tác khác nhau lên dữ liệu.Nhìn chung, hệ quản trị sử dụng 2 loại chính: *exclusive locks* (khóa độc quyền) và *share locks* (khóa chia sẻ). Tại một thời điểm, chỉ có 1 *excusive lock* được thực thi trên 1 đơn vị dữ liệu, nhưng nhiều *share locks* có thể cùng thực thi trên 1 đơn vị dữ liệu.

b. Cơ chế khóa tác động đến sự tương tác giữa việc đọc và ghi dữ liệu:

i. Một dòng dữ liệu chỉ bị khóa khi nó đang được thay đổi bởi việc hành động ghi. Khi có 1 câu truy vấn cập nhật dữ liệu của 1 dòng, hệ quản trị sẽ khóa dòng đó lại.

ii. Một hành động ghi lên 1 dòng dữ liệu sẽ ngăn chặn các hành động khác cùng đồng thời ghi lên dòng dữ liệu đó.

iii. Hành động đọc dữ liệu không ngăn cản hành động ghi: Vì hành động đọc không khóa đơn vị dữ liệu nó đang đọc, nên hành động ghi có thể thao tác lên đơn vị dữ liệu này. Chỉ có một trường hợp ngoại lệ là câu lệnh **`Select … for update`** sẽ lock đơn vị dữ liệu nó đang đọc.

iv. Hành động ghi dữ liệu không ngăn cản hành động đọc: Khi một đơn vị dữ liệu đang được thao tác bởi hành động ghi, hệ quản trị sẽ sử dụng phiên bản trước đó của đơn vị dữ liệu để trả về cho hành động đọc.

c. Cơ chế khóa làm được gì ?

i. Cơ chế khóa phải thỏa các yêu cầu quan trọng của cơ sở dữ liệu: tính nhất quán và tính toàn vẹn

ii. Hệ quản trị Oracle đảm bảo dữ liệu đồng thời, nhất quán và toàn vẹn giữa các transactions thông qua cơ chế khóa. Cơ chế khóa diễn ra tự động và không cần sự can thiệp của người dùng.

iii. Các chế độ khóa:

a) Hệ quản trị Oracle tự động sử dụng chế độ khóa ở cấp cần thiết thấp nhất để đảm bảo khả năng truy cập đồng thời của nhiều tiến trình lên cùng đơn vị dữ liệu.

b) Sử dụng chế độ khóa cấp càng thấp, dữ liệu có thể được truy cập bởi càng nhiều user. Ngược lại, nếu sử dụng mức khóa càng cao, dữ liệu càng hạn chế người truy cập.

c) Hệ quản trị Oracle sử dụng 2 chế độ khóa chính cho cơ sở dữ liệu nhiều người dùng :

- Exclusive lock mode: Transaction nhận được exclusive lock khi nó có nhu cầu thay đổi dữ liệu. Các transaction khác sẽ không thể thao tác lên đơn vị dữ liệu này cho đến khi transaction đang giữ exclusive lock nhả khóa.

- Share lock mode: Chế độ này cho phép chia sẻ cùng 1 đơn vị dữ liệu. Nhiều giao tác đọc có thể chia sẻ dữ liệu cho nhau, mỗi giao tác giữ một share lock để ngăn hành động ghi vào cùng thời điểm của các giao tác ghi. Nhiều giao tác có thể được cấp share lock trên cùng đơn vị dữ liệu.

III. Deadlock:

1. Deadlock là tình huống khi một hay nhiều người dùng đang chờ một đơn vị dữ liệu mà đang bị các người dùng khác cũng đang chờ đơn vị dữ liệu khác block.

2. Hệ quản trị Oracle tự động phát hiện và xử lý deadlock bằng cách rollback một transaction gây deadlock, giải phóng 1 dòng đang bị khóa. Cơ sở dữ liệu sẽ trả về thông báo cho transaction bị rollback.

IV. Transaction và thiết lập mức cố lập trong JAVA

1. Transaction:

a. Transaction ở Oracle database: Khi thực hiện lệnh DML hoặc DDL sẽ bắt đầu transaction và kết thúc bằng lệnh Commit hoặc Rollback.

b. Transaction ở Java thì khác: Transaction trên java bắt đầu mỗi khi bạn tạo Connection. Trong java có nhiều cách để kết thúc transaction.

i. Nếu Connection đang ở chế độ tự động commit (Khi khởi tạo Connection thì mặc định ở chế độ tự động commit) khi thực hiện lệnh `executeUpdate()` (tương ứng với thực hiện lệnh DML) sẽ kết thúc Transaction.

```
public class ConnectionOracle {
    private static String user = "[redacted]";
    private static String password = "[redacted]";
    private static String port ="1521";
    private static String dbName ="orcl";
    private static String connectionObject = "jdbc:oracle:thin:@localhost:"+ port + ":" + dbName;
    private static Connection con;

    public static Connection getConnection(){
        try {
            //Class.forName("oracle.jdbc.driver.OracleDriver");
            con = DriverManager.getConnection(connectionObject,user,password);

        } catch ( SQLException e) {

            System.out.println(e);
        }
        return con;
    }
}
```

Figure 1 Hàm Tạo Connection

```java
public void actionPerformed(ActionEvent e) {
    try {
        conn = ConnectionOracle.getConnection();
        //bắt đầu Transaction 1.
        String stringSql = "Insert into KhachHang values(?,?,?,?)";
        PreparedStatement ps = conn.prepareStatement(stringSql);
        ps.setString(1, txtMaKhachHang.getText());
        ps.setString(2, txtHoTen.getText());
        ps.setString(3, txtDiaChi.getText());
        ps.setString(4, txtDienThoai.getText());
        ps.executeUpdate(); // Commit và kết thúc Transaction 1.
        //Tuy Transaction 1 kết thúc nhưng biến "conn" vẫn còn duy trì kết nối.
        String stringSql2 = "Select * from KhachHang";
        Statement stm = conn.createStatement();
        //tiếp tục dùng biến "conn" sẽ tạo ra Transaction 2.
        ResultSet rs = stm.executeQuery(stringSql2);
        // lệnh executeQuery không làm kết thúc Transaction 2.
        while(rs.next()){
            System.out.println(rs.getString(1));
        }
        conn.close(); // kết thúc transaction 2.
```

Figure 2 Kết thúc Transaction bằng lệnh executeUpdate() hoặc conn.close()

ii. Khi tạo Connection sau đó bạn thực hiện lệnh thiết lập tắt chế độ tự động commit trên java bằng lệnh "`conn.setAutoCommit(false)`" (conn ở đây là 1 biến Connection) thì lúc này muốn kết thúc Transaction phải thực thi lệnh `conn.commit()` hoặc `conn.close()` hoặc tạo Connection mới. Nhưng nếu không dùng lệnh `conn.commit()` thì dữ liệu sẽ không được lưu xuống database.

```java
try {
    conn = ConnectionOracle.getConnection();
    //bắt đầu Transaction 1.
    conn.setAutoCommit(false);
    //tắt chế độ tự động commit.
    String stringSql = "Insert into KhachHang values(?,?,?,?)";
    PreparedStatement ps = conn.prepareStatement(stringSql);
    ps.setString(1, txtMaKhachHang.getText());
    ps.setString(2, txtHoTen.getText());
    ps.setString(3, txtDiaChi.getText());
    ps.setString(4, txtDienThoai.getText());
    ps.executeUpdate();
    //Lúc này không được commit nên Transaction 1 vẫn tiếp tục thực hiện.
    String stringSql2 = "Select * from KhachHang";
    Statement stm = conn.createStatement();
    ResultSet rs = stm.executeQuery(stringSql2);
    while(rs.next()){
        System.out.println(rs.getString(1));
    }
    conn.commit();// kết thúc Transaction 1.
    conn.close();
```

Figure 3 Tắt chế độ tự động commit

iii. Trường hợp 3 là kể cả khi chế độ tự động commit bật hay tắt thì khi bạn thực hiện lệnh conn.close() hoặc tạo Connection mới cũng gây ra kết thúc Transaction. **Việc thường xuyên tạo mới Connection không được khuyến khích vì như vậy sẽ khiến chương trình chạy chậm.**

```java
try {
    conn = ConnectionOracle.getConnection();
    //bắt đầu Transaction 1.
    String stringSql = "Select * from XE";
    Statement stm = conn.createStatement();
    ResultSet rs = stm.executeQuery(stringSql);
    while(rs.next()){
        System.out.println(rs.getString(1));
    }
    conn = ConnectionOracle.getConnection();
    //kết thúc Transaction 1 và Tạo ra Transaction 2
    String stringSql2 = "Select * from KhachHang";
    Statement stm2 = conn.createStatement();
    ResultSet rs2 = stm2.executeQuery(stringSql2);
    while(rs2.next()){
        System.out.println(rs2.getString(1));
    }
    conn.close();
    //kết thúc Transaction 2
```

Figure 4 Kết thúc Transaction bằng cách tạo Connection mới

2. Thiết lập mức cô lập

a. Trong Oracle chỉ hỗ trợ 2 mức cô lập là Read Committed và Serializable. Vì vậy trên Java ta chỉ được thiết lập 2 mức cô lập đó, nếu thiết lập mức cô lập khác sẽ bị báo lỗi.

```java
try {
    conn = ConnectionOracle.getConnection();
    //bắt đầu Transaction 1.
    String stringSql = "Select * from XE";
    conn.setTransactionIsolation(Connection.TRANSACTION_READ_COMMITTED);
    conn.setTransactionIsolation(Connection.TRANSACTION_SERIALIZABLE);
    Statement stm = conn.createStatement();
    ResultSet rs = stm.executeQuery(stringSql);
    while(rs.next()){
        System.out.println(rs.getString(1));
    }
    conn = ConnectionOracle.getConnection();
    //kết thúc Transaction 1 và Tạo ra Transaction 2
    String stringSql2 = "Select * from KhachHang";
    Statement stm2 = conn.createStatement();
    ResultSet rs2 = stm2.executeQuery(stringSql2);
    while(rs2.next()){
        System.out.println(rs2.getString(1));
    }
    conn.close();
```

Figure 5 Thiết lập mức cô lập trên Java

Lưu ý: Các trường hợp vi phạm truy xuất đồng thời Non-repeatable Read, Phantom Read, Deadlock chỉ xảy ra khi Transaction chưa kết thúc.

V. Mô tả đồ án môn học:

1. Lost Update:

a. Mô tả tình huống: Khi sinh viên đăng ký học phần, nếu trong học kỳ đó chưa đăng ký học phần nào thì thủ tục `INSERT_SUBJECT_REGISTRATION` sẽ tự động insert dòng mới vào bảng Fee, khi insert vào bảng Fee trigger `update_expirationDateForNewInsert` sẽ kiểm tra học phí học kỳ này đã có ngày hết hạn đóng học phí hay chưa, nếu có thì nó sẽ cập nhật vào dòng mới insert nhưng ngay lúc đó người ở phòng kế hoạch tài chính thực hiện cập nhật lại ngày hết hạn đóng học phí. Dòng vừa insert vào bảng Fee sẽ cập nhật thời gian hết hạn cũ (lost update).

b. Mã PL/SQL:

i. Procedure `sleep` (dùng để giả lập hệ thống bị trì trệ)

```
CREATE OR REPLACE PROCEDURE sleep (in_time number)
AS
v_now date;
BEGIN
SELECT SYSDATE
INTO v_now
FROM DUAL;
LOOP
EXIT WHEN v_now + (in_time * (1/86400)) <= SYSDATE;
END LOOP;
end;
```

ii. Procedure `INSERT_SUBJECT_REGISTRATION`

```
CREATE OR REPLACE PROCEDURE INSERT_SUBJECT_REGISTRATION
(in_studentID STUDENT.STUDENTID%TYPE
  , in_PDT_ID PHONGDAOTAO.PDT_ID%TYPE
  , in_offeringID OFFERING.OFFERINGID%TYPE, in_semester
OFFERING.SEMESTER%TYPE)
AS
    v_Score DIEMMONHOC.SCORE%TYPE;
    v_preSubject SUBJECT.PREVIOUSSUBJECT%TYPE;
    v_countSubject number;
    v_SubjectID SUBJECT.SUBJECTID%TYPE;
    cur_BangDiemID BANGDIEM.ID_BANGDIEM%TYPE;
    v_status number := 0;
    v_status2 number := 0;
    v_status3 number := 0;
    v_idBangDiem BANGDIEM.ID_BANGDIEM%TYPE;
    v_lectureID LECTURE.LECTUREID%TYPE;
    v_date varchar2(255);
    v_registered number;
    v_slot number;
    v_hocphi number;
    v_tinchi number;
    v_studentID STUDENT.STUDENTID%TYPE;
    CURSOR cur IS SELECT ID_BANGDIEM
```

```
                  FROM BANGDIEM
                  WHERE BANGDIEM.STUDENTID =
in_studentID;
BEGIN
       SELECT PREVIOUSSUBJECT
       INTO v_preSubject
       FROM SUBJECT S, OFFERING O
       WHERE S.SUBJECTID = O.SUBJECTID AND O.OFFERINGID
= in_offeringID;

       IF(v_preSubject IS NOT NULL)
       THEN
            OPEN cur;
            LOOP
                FETCH cur INTO cur_BangDiemID;
                EXIT WHEN cur%NOTFOUND;
                BEGIN
                    SELECT score
                    INTO v_Score
                    FROM BANGDIEM,DIEMMONHOC
                    WHERE BANGDIEM.ID_BANGDIEM =
DIEMMONHOC.ID_BANGDIEM
                    AND BANGDIEM.ID_BANGDIEM =
cur_BangDiemID
                    AND BANGDIEM.STUDENTID =
in_studentID
                    AND DIEMMONHOC.ID_MONHOC =
v_preSubject;
                EXCEPTION
                    WHEN NO_DATA_FOUND
                    THEN
                        v_Score:= NULL;
                END;
                IF(v_Score > 5)
                THEN
```

```
                    v_status := 1;
                    --RAISE_APPLICATION_ERROR(-20000,
'Dieu Kien Tien Quyet Khong Duoc');
                END IF;
            END LOOP;
            IF(v_status = 0)
            THEN
            RAISE_APPLICATION_ERROR(-20000, 'Dieu Kien
Tien Quyet Khong Duoc');
            END IF;
        END IF;

        SELECT subjectID,LectureID,slot
        INTO v_SubjectID,v_lectureID,v_slot
        FROM OFFERING
        WHERE OFFERING.OFFERINGID = in_offeringID;

        SELECT COUNT(*)
        INTO v_countSubject
        FROM SUBJECT_REGISTRATION sr, OFFERING o, SUBJECT
s
        WHERE sr.REGISTEREDBY= in_studentID
        AND sr.OFFERINGID = o.OFFERINGID
        AND o.SUBJECTID = s.SUBJECTID
        AND o.SUBJECTID = v_SubjectID ;

        IF(v_countSubject > 0)
        THEN
            RAISE_APPLICATION_ERROR(-20001, 'Mon nay da
duoc dang ky');
        END IF;

        SELECT COUNT(offeringID)
        INTO v_registered
        FROM SUBJECT_REGISTRATION
```

```
        WHERE offeringID = in_offeringID;

        IF(v_registered >= v_slot)
        THEN
             RAISE_APPLICATION_ERROR(-20002, 'Lop Da
Full');
        END IF;

        SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
        BEGIN
             SELECT ID_BangDiem
             INTO v_idBangDiem
             FROM BangDiem
             WHERE BangDiem.STUDENTID = in_studentID AND
semester = in_semester;
             EXCEPTION
             WHEN NO_data_found
             THEN
                 v_status2 := 1;
        END;
        IF(v_status2 = 1)
        THEN
             v_idBangDiem := GETBANGDIEMID();
             INSERT INTO BANGDIEM VALUES
(v_idBangDiem,in_studentID,in_semester);
        END IF;

        SELECT SUBJECT.NUMBEROFCREDITS
         INTO v_tinchi
         FROM SUBJECT
         WHERE SUBJECT.SUBJECTID = v_SubjectID;

         v_hocphi := v_tinchi * 500000;

         BEGIN
```

```
            SELECT FEE.STUDENTID
            INTO v_studentID
            FROM FEE
            WHERE FEE.STUDENTID = in_studentID AND
FEE.SEMESTER = in_semester;
            EXCEPTION
            WHEN NO_data_found
            THEN
                v_status3 := 1;
        END;
        IF(v_status3 = 1)
        THEN
            INSERT INTO FEE VALUES
(FEE_ID_SEQ.NEXTVAL,v_hocphi,in_semester,in_studentID,0,
NULL,NULL,NULL,NULL);
        ELSE
            UPDATE fee
            set MONEY = MONEY + v_hocphi
            WHERE fee.STUDENTID = in_studentID
            AND fee.SEMESTER = in_semester;
        END IF;

       INSERT INTO SUBJECT_REGISTRATION VALUES
(registrationID_SEQ.NEXTVAL,in_studentID,in_PDT_ID,in_of
feringID,in_semester);
       SELECT TO_CHAR(SYSDATE, 'DD-MM-YYYY') INTO v_date
FROM dual;
       dbms_output.put_line(v_date);
       INSERT INTO DIEMMONHOC
VALUES(DiemMonHoc_ID_SEQ.NEXTVAL,0,0,0,0,0,v_lectureID,T
O_DATE(v_date,'DD-MM-YYYY'),v_idBangDiem,v_SubjectID);
       COMMIT;
END;
```

iii. Trigger `update_expirationDateForNewInsert`:

```
CREATE OR REPLACE TRIGGER
update_expirationDateForNewInsert
FOR INSERT ON fee
COMPOUND TRIGGER
    v_expirationDate date;
BEFORE EACH ROW IS
BEGIN
    --LOCK TABLE fee IN SHARE MODE;

    SELECT expirationdate
    INTO v_expirationDate
    FROM fee
    WHERE fee.SEMESTER = :NEW.semester AND
fee.expirationdate IS NOT NULL
    GROUP BY expirationdate;

    :NEW.expirationdate := v_expirationDate;

    EXCEPTION WHEN no_data_found
    THEN
        dbms_output.put_line('Khong co ngay het han');
END BEFORE EACH ROW;
END;
```

Bảng Mô tả Lost Update

Session 1	Session 2	Explanation
INSERT_SUBJECT_REGISTRATION ('SV001',NULL,'MH001.201', '2018-2019');		Session 1 thực hiện đăng ký học phần bằng INSERT_SUBJECT_REGISTRATION
INSERT INTO FEE VALUES (FEE_ID_SEQ.NEXTVAL,v_hocphi ,in_semester,in_studentID, 0,NULL,NULL,NULL,NULL);	No action.	Thủ tục đăng ký học phần chạy tới lệnh insert Fee.

SELECT expirationdate INTO v_expirationDate FROM fee WHERE fee.SEMESTER = :NEW.semester AND fee.expirationdate IS NOT NULL GROUP BY expirationdate; :NEW.expirationdate := v_expirationDate; --Ví dụ v_expirationDate = '22/06/2019'	No action.	Khi thực hiện insert vào bảng Fee, trigger Update_ExpirationDate_FEE sẽ kiểm tra học phí học kỳ đó đã có ngày hết hạn đóng học phí hay chưa, nếu có thì sẽ cập nhật vào dòng insert.
	Update_ExpirationDate_FEE ('2018-2019', '05/07/2019')	Sesson2 thực hiện cập nhật ngày hết hạn đóng học phí của học kỳ 2018-2019 thành ngày '05/07/2019'
No action.	Commit;	Transaction 2 thực hiện thành công và đã kết thúc.
Commit;		Transaction 1 thực hiện thành công và đã kết thúc.
	SELECT fee.studentID,TO_CHAR(ExpirationDate, 'DD-MM-YYYY'),fee.semester FROM fee WHERE fee.SEMESTER = '2018-2019' AND fee.STUDENTID = sr.REGISTEREDBY GROUP BY fee.studentID, FEE.EXPIRATIONDATE, fee.semester"; studentID ExpirationDate semester --- --- --- SV001 22/06/2019 2018-2019 Sv002 05/07/2019 2018-2019	Session 2 thực hiện truy vấn danh sách học phí sinh viên trong học kỳ 2018-2019, nhận thấy ngày hết hạn đóng học phí mới không được cập nhật cho sinh viên SV001.

c. Nguyên nhân và giải pháp:

i. Vấn đề xảy ra: khi trigger của transaction 1 cập nhật ngày hết hạn đóng học phí cho sinh viên SV001 thì đã lấy dữ liệu cũ.

ii. Nguyên nhân: Trigger của transaction 1 lấy ra dữ liệu thời gian hết hạn đóng học phí trước khi transaction 2 cập nhật thời gian hết hạn đóng học phí mới, dẫn đến trigger trong transaction 1 đã cập nhật sai thời gian hết hạn đóng học phí.

iii. Giải pháp: sử dụng câu lệnh "`LOCK TABLE fee IN SHARE MODE`" ở trong trigger `Update_ExpirationDate_FEE`, câu lệnh này giúp khi trigger truy vấn ngày hết hạn đóng học phí trong FEE sẽ đồng thời khóa bảng chỉ cho phép transaction khác đọc và không được phép ghi.

2. Non-repeatable read:

a. Mô tả tình huống: Trong khi nhân viên thứ nhất thuộc phòng kế hoạch tài chính đang xem bao nhiêu sinh viên đã đóng học phí trong hôm nay, thì nhân viên phòng KHTC khác lại thu tiền đóng học phí của sinh viên và cập nhật vào trong hệ thống, lúc này nhân thứ nhất lập báo cáo tổng học phí thu được trong ngày thì phát hiện dữ liệu bị sai sót

b. Mã PL/SQL:

Bảng Mô tả Non-repeatable read

Session 1	Session 2	Explanation
`Set transaction isolation level read committed;`	No action.	Session 1 thiết lập mức cô lập là read committed.

`Select studentID, paymented from fee`
`Where paymenteDate = '25/06/2019' and semester = '2018-2019';`
`studentID  paymented`
`---------  ---------`
`Sv001  2500000`
	No action.	Session 1 thực hiện truy vấn sinh viên, và số tiền mà sinh viên đã đóng trong ngày '25/06/2019' trong học kỳ 2018-2019

Sv002 3500000		
No action.	Update fee Set paymented = 1000000, paymentDate = '25/06/2019' Where studentID = 'SV003' and semester = '2018-2019';	Session 2 Cập nhật học phí cho sinh viên sv003 cũng vào ngày '25/06/2019'.
No action.	Commit;	Transaction 2 thực hiện xong và kết thúc.
Select studentID, paymented from fee Where paymenteDate = '25/06/2019' and semester = '2018-2019'; studentID paymented --- --- Sv001 2500000 Sv002 3500000 Sv003 1000000	No action.	Session 1 thực hiện lại truy vấn sinh viên, và số tiền mà sinh viên đã đóng trong ngày '25/06/2019' trong học kỳ 2018-2019 để làm báo cáo, thì thấy dữ liệu bị thay đổi

c. Nguyên nhân và giải pháp:

i. Vấn đề xảy ra: T1 thực hiện truy vấn số sinh viên và tiền học phí đã đóng của từng sinh viên trong ngày, ngay lúc này T2 cập nhật học phí cho Sv003, sau đó T1 thực lại truy vấn để làm báo cáo thì phát hiện dữ liệu đã bị thay đổi.

ii. Nguyên nhân: T1 thiết lập mức cô lập read committed nên mỗi lần SELECT trong cùng 1 thao tác dữ liệu, nó sẽ đọc lại dữ liệu từ cơ sở dữ liệu(cơ sở dữ liệu lúc này có thể bị thay đổi) mắc dù những câu lệnh này đọc trên những đơn vị dữ liệu giống nhau.

iii. Giải pháp: sử dụng câu lệnh "`set transaction isolation level serializable`" thay cho câu lệnh "`set transaction isolation level read committed`".

3. Phantom read:

a. Mô tả tình huống: Trong lúc nhân viên phòng CTSV xem danh sách dự kiến sinh viên nhận học bổng theo tiêu chí sinh viên có điểm trung bình học kỳ > 7, không bị rớt môn, đăng ký tối thiểu 14 tín chỉ, mỗi lớp lấy tối đa 3 sinh viên có điểm trung bình học kỳ cao nhất, ngay lúc đó giảng viên thực hiện cập nhật tăng điểm cho 1 sinh viên, sinh viên này từ không thuộc diện nhận học bổng trở thành thuộc danh sách dự kiến nhận học bổng, Sau khi nhân viên phòng CTSV xem xét kĩ càng về danh sách dự kiến nhân học bổng thì nhân viên này thực hiện thêm danh sách dự kiến này vào danh sách chính thức nhận học bổng và thấy số lượng sinh viên nhận được học bổng bị thay đổi.

b. Mã PL/SQL:

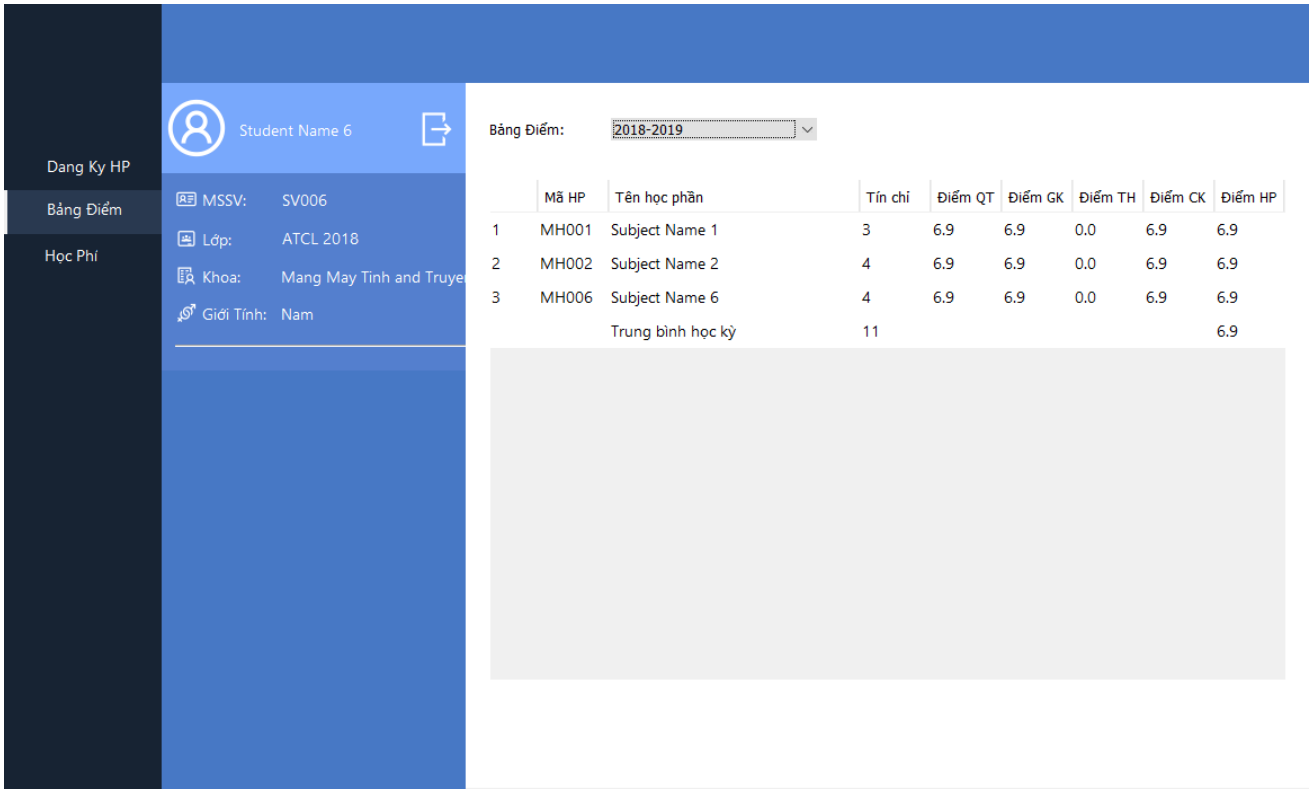
i. Funtion lấy ra danh sách dự kiến nhận học bổng.

```
CREATE OR REPLACE FUNCTION getListScholarship(in_semester
SEMESTER.SEMESTERID%TYPE)
RETURN table_Scholarship
AS
    v_tableTemp table_Scholarship;
BEGIN

    SELECT row_Scholarship(sub2.STUDENTID, sub2.FULLNAME,
sub2.CLASSNAME, sub2.diem)
    BULK COLLECT INTO v_tableTemp
    FROM
        (SELECT sub.FULLNAME, sub.STUDENTID, sub.CLASSNAME,
sub.SEMESTER, get_diemHP(sub.STUDENTID,sub.SEMESTER) AS
diem, ROW_NUMBER() OVER(PARTITION BY sub.CLASSNAME ORDER BY
get_diemHP(sub.STUDENTID,sub.SEMESTER) desc) AS rn
```

```
            FROM
                (SELECT us.FULLNAME, st.STUDENTID,
cl.CLASSNAME, bd.SEMESTER

                FROM users us, student st, class cl,
SUBJECT sub, bangdiem bd, diemmonhoc dmh

                WHERE st.STUDENTID = us.USERID AND
st.STUDENTID = bd.STUDENTID AND st.CLASSID = cl.CLASSID
                AND bd.SEMESTER = in_semester AND
bd.ID_BANGDIEM = dmh.ID_BANGDIEM
                AND dmh.ID_MONHOC = sub.SUBJECTID
                AND NOT EXISTS (SELECT
diemmonhoc.ID_DIEMMONHOC
                            FROM diemmonhoc
                            WHERE diemmonhoc.ID_BANGDIEM =
dmh.ID_BANGDIEM
                            AND diemmonhoc.SCORE < 5 )
                GROUP BY us.FULLNAME, st.STUDENTID,
cl.CLASSNAME, bd.SEMESTER
                HAVING SUM(NUMBEROFCREDITS) >= 11) sub
            WHERE get_diemHP(sub.STUDENTID,sub.SEMESTER) >
7) sub2
    WHERE sub2.RN <= 3 AND NOT EXISTS (SELECT hb.STUDENTID
                        FROM SCHOLARSHIP hb
                        WHERE sub2.SEMESTER = hb.SEMESTER
                        AND sub2.STUDENTID = hb.STUDENTID);

    RETURN v_tableTemp;
END;
```

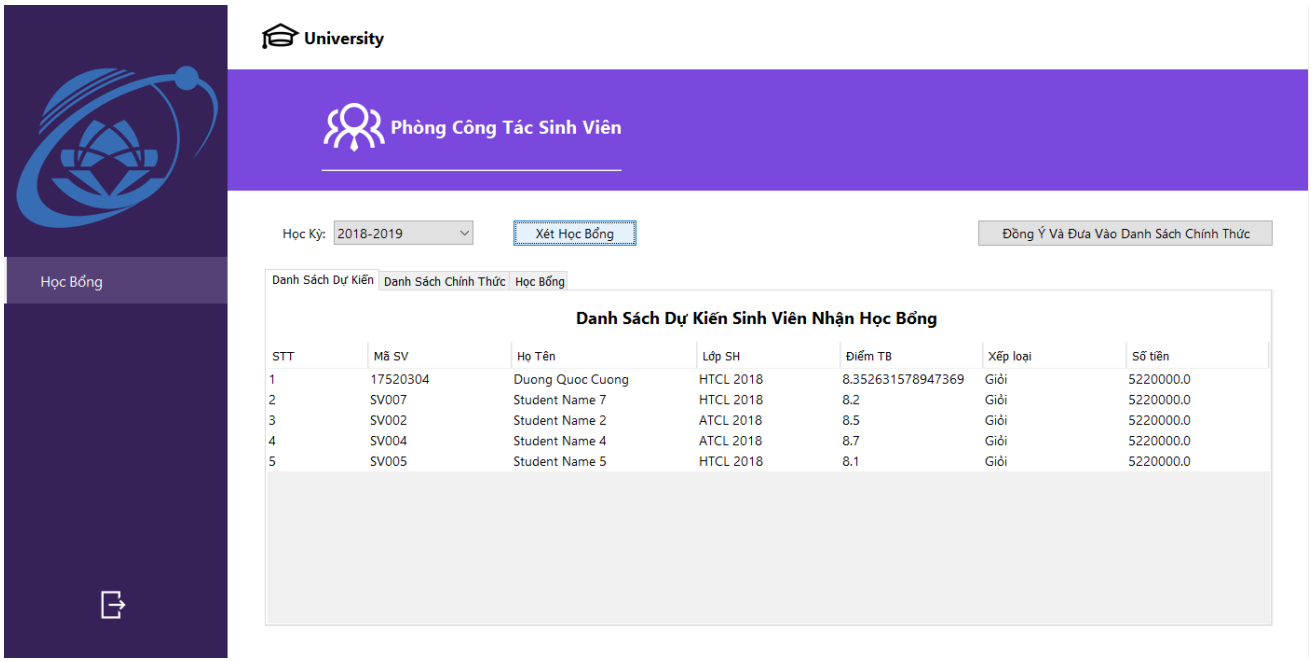
c. Mô tả:

Ban đầu sinh viên SV006 không đạt tiêu chuẩn để xét học bổng



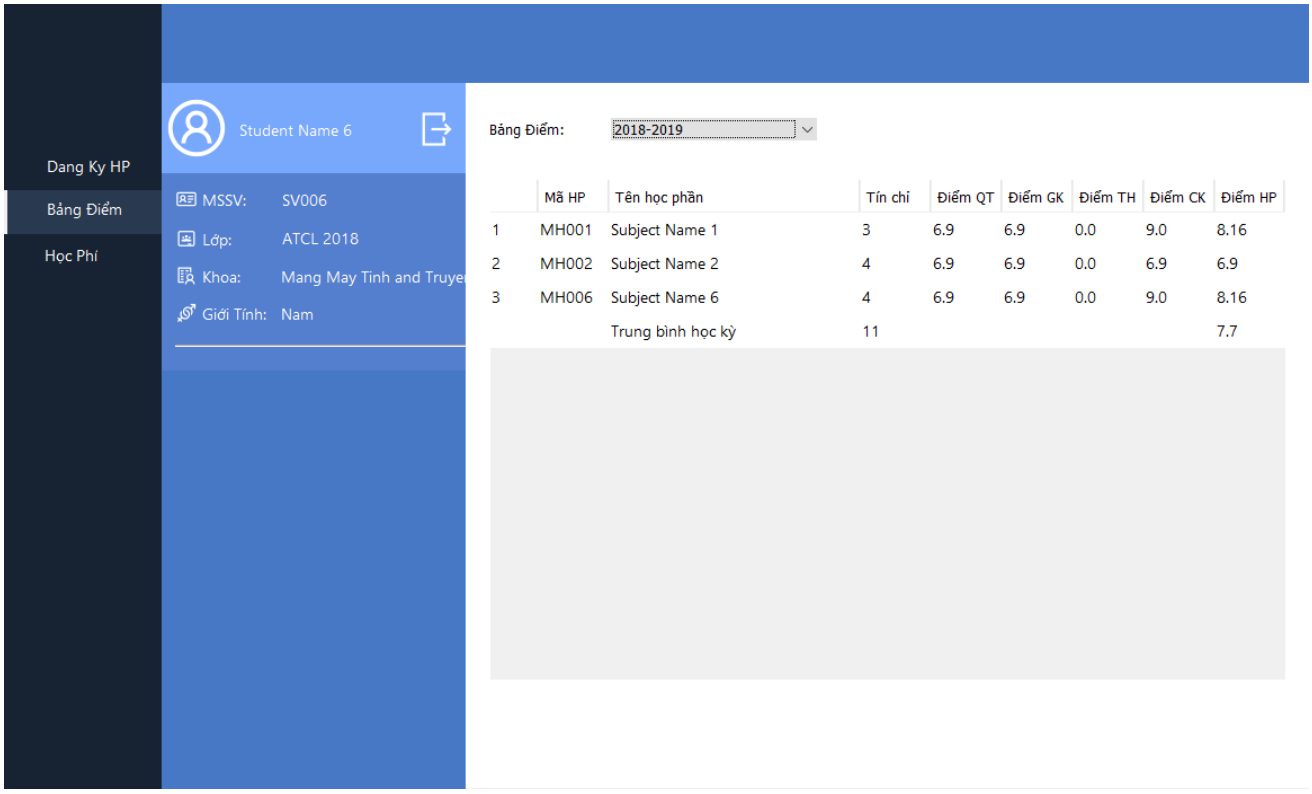
Hình 1 Điểm ban đầu của SV006

Nên ban đầu phòng CTSV xem danh sách dự kiến sẽ không có sinh viên SV006 trong danh sách dự kiến.



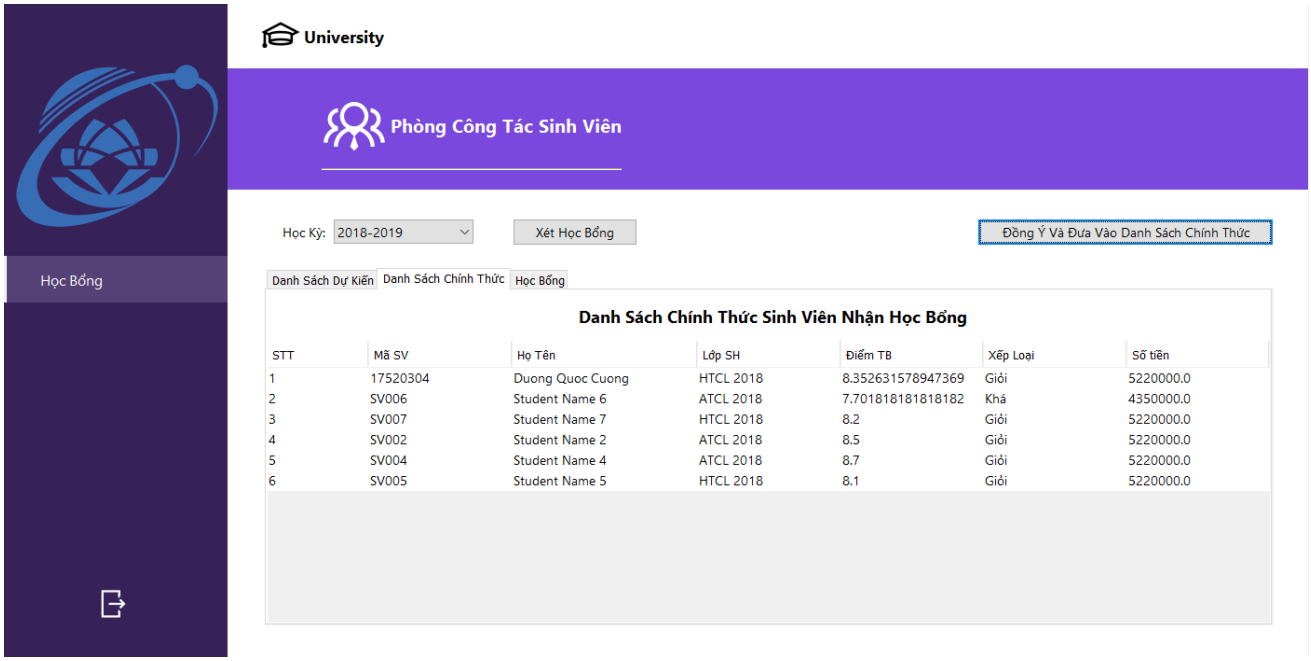
Hình 2 Danh sách sinh viên dự kiến nhận học bổng ban đầu

Nhưng ngay lúc này sinh viên SV006 được cập nhật điểm, khiến cho sinh viên SV006 đạt tiêu chuẩn nhận học bổng.



Hình 3 Điểm của Sinh viên SV006 sau khi được cập nhật

Sau khi xem xét kĩ càng phòng CTSV quyết định đưa danh sách dự kiến thành danh sách chính thức, thực hiện thêm vào bảng Scholarship. Nhưng lúc này thì phát hiện danh sách chính thức bị thay đổi so với danh sách dự kiến.



Hình 4 Đưa danh sách dự kiến thành danh sách chính thức.

d. Nguyên nhân và giải pháp:

i. Vấn đề xảy ra: Khi T1 truy vấn danh sách dự kiến sinh viên nhận được học bổng, lúc này T2 thực hiện cập nhật điểm cho SV001, ban đầu SV006 không có trong danh sách dự kiến được nhận học bổng, nhưng sau khi được cập nhật điểm thì SV006 được nằm trong danh sách dự kiến nhận học bổng.Sau đó T1 thực hiện thêm danh sách dự kiến thành danh sách chính thức và thêm dữ liệu vào bảng Scholarship thì thấy số dòng dữ liệu đã bị thay đổi so với danh sách dự kiến nhận học bổng.

ii. Nguyên nhân: T1 thiết lập mức cô lập read committed nên lần đọc dữ liệu thứ 2 sẽ tính luôn cả sinh viên mới được nhật lên danh sách dự kiến nhận học bổng.

iii. Giải pháp: sử dụng câu lệnh "`conn.setTransactionIsolation(Connection.TRANSACTION_READ_COMMITTED)`" thay cho câu lệnh "`conn.setTransactionIsolation(Connection.TRANSACTION_SERIALIZABLE)`".

CHƯƠNG 6: THIẾT KẾ GIAO DIỆN

I. Danh sách các màn hình:

1. Đăng nhập:

STT	Màn hình	Chức năng
1.	Đăng nhập	Đăng nhập

2. IT-department:

STT	Màn hình	Chức năng
1.	Quản lý phòng IT	Tra cứu phòng
2.	Tạo phòng	Thêm mới hoặc cập nhật thông tin phòng

3. Student:

STT	Màn hình	Chức năng
1.	Quản lý đăng kí học phần	Tra cứu thông tin đăng kí học phần
2.	Quản lý điểm	Tra cứu thông tin điểm cho từng năm học
3.	Quản lý học phí	Tra cứu và cập nhật thông tin học phí
4.	Quản lý đăng kí học phần	Tra cứu thông tin đăng kí học phần

4. Lecturer:

a. Quản lí điểm

STT	Màn hình	Chức năng
1.	Môn học	Hiển thị học kì đó mở bao nhiêu học phần
2.	Danh sách sinh viên	Danh sách sinh viên và điểm từng sinh viên học học phần đã chọn

5. Phòng đào tạo:

STT	Màn hình	Chức năng
1.	Quản lí giảng viên	Hiển thị những thông tin đã có trong hệ thống
2.	Quản lí sinh viên	Hiển thị những thông tin đã có trong hệ thống
3.	Quản lí lớp	Hiển thị những thông tin đã có trong hệ thống
4.	Quản lí học phần	Hiển thị những thông tin đã có trong hệ thống
5.	Quản lí khoa	Hiển thị những thông tin đã có trong hệ thống
6.	Quản lí môn	Hiển thị những thông tin đã có trong hệ thống
7.	Thêm môn học	Nhập thông tin để thêm môn học mới
8.	Chỉnh sửa học phần	Nhập thông tin để chỉnh sửa học phần đã có
9.	Mở học phần	Nhập thông tin để mở học phần mới

10.	Thêm khoa	Nhập thông tin để thêm khoa mới
11.	Thêm vào khoa	Chọn khoa để thêm giảng viên mới, vào khoa
12.	Thêm vào lớp	Chọn lớp để thêm sinh viên mới vào
13.	Mở lớp	Nhập thông tin để mở lớp mới

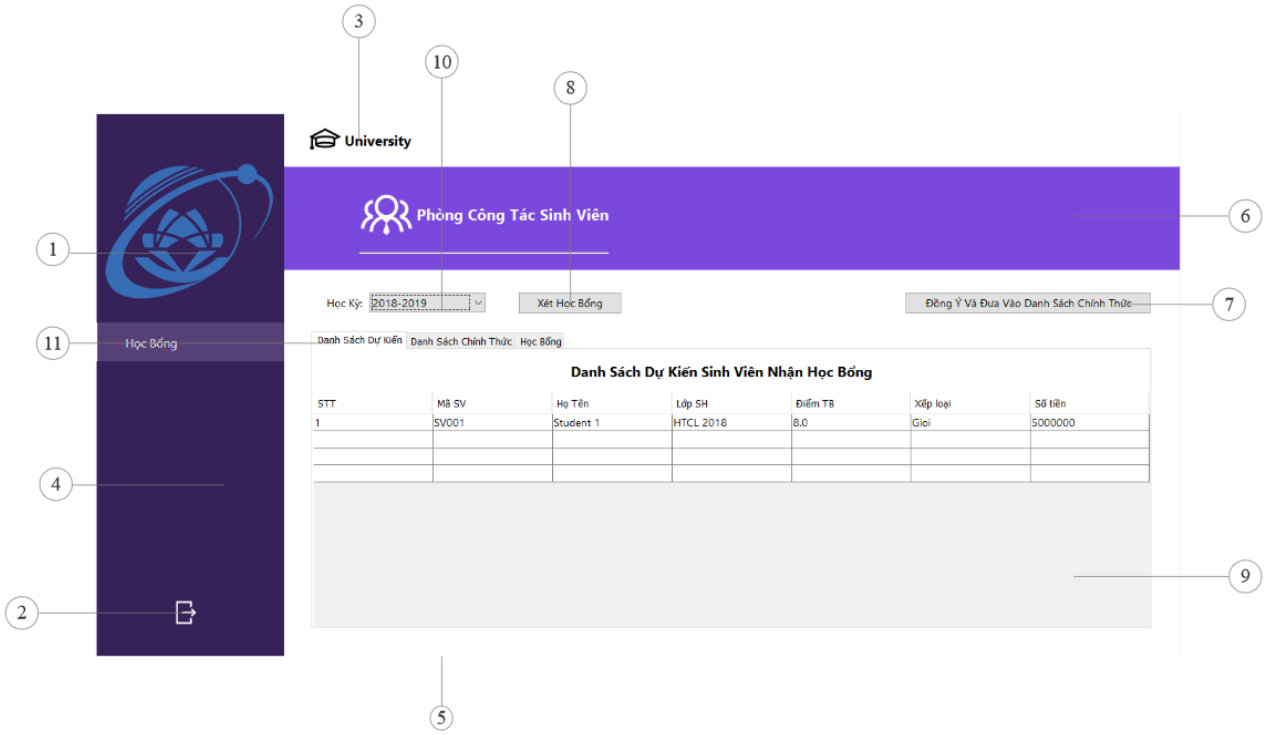
6. Phòng công tác sinh viên:

STT	Màn hình	Chức năng
1.	Danh sách dự kiến	Xem thông tin những sinh viên dự kiến sẽ được xét học bỗng
2.	Danh sách chính thức	Xem thông tin những sinh viên chính thức sẽ được xét học bỗng
3.	Học bỗng	Xem thông tin những sinh viên chính thức sẽ được xét học bỗng

II. Mô tả các màn hình:

1. Phòng công tác sinh viên:

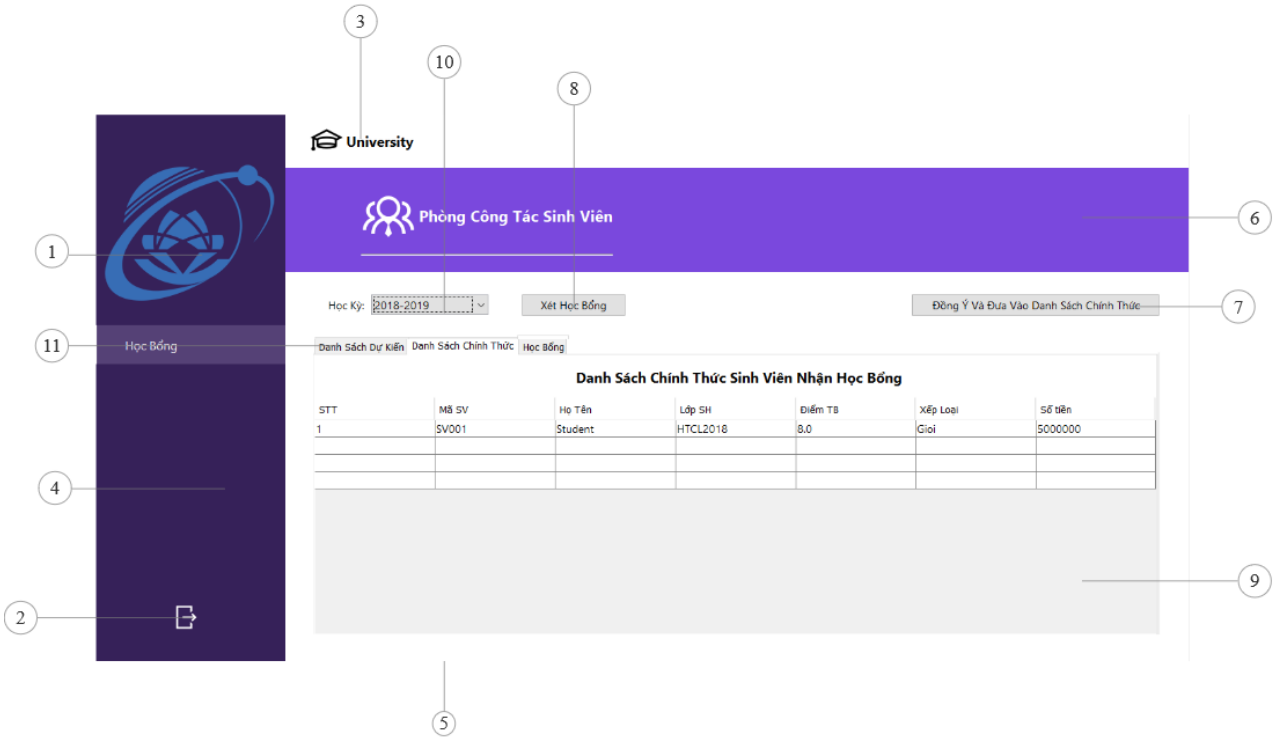
a. Danh sách dự kiến:



STT	Tên	Kiểu	Chức năng
1.	Logo uit	Label	Hiển thị thông tin
2.	Logo đăng xuất	Label	Cho biết đây là nơi đăng xuất
3.	Logo university	Label	Hiển thị thông tin
4.	Khung hình	Panel	Chứa những Components khác
5.	Khung hình	Panel	Chứa những Components khác
6.	Khung hình	Panel	Chứa những Components khác
7.	Đồng ý và đưa vào danh sách chính thức	Button	Xác nhận những sinh viên ở bảng dự kiến đã đủ điều kiện để nhận học bỗng
8.	Xét học bỗng	Button	Kiểm tra xem sinh viên đó đã đủ điều kiện nhận học bỗng hay chưa

9.	Bảng	Table	Hiển thị thông tin sinh viên dự kiến nhận học bỗng
10.	2018-2019	Combobox	Chọn học kì
11.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem

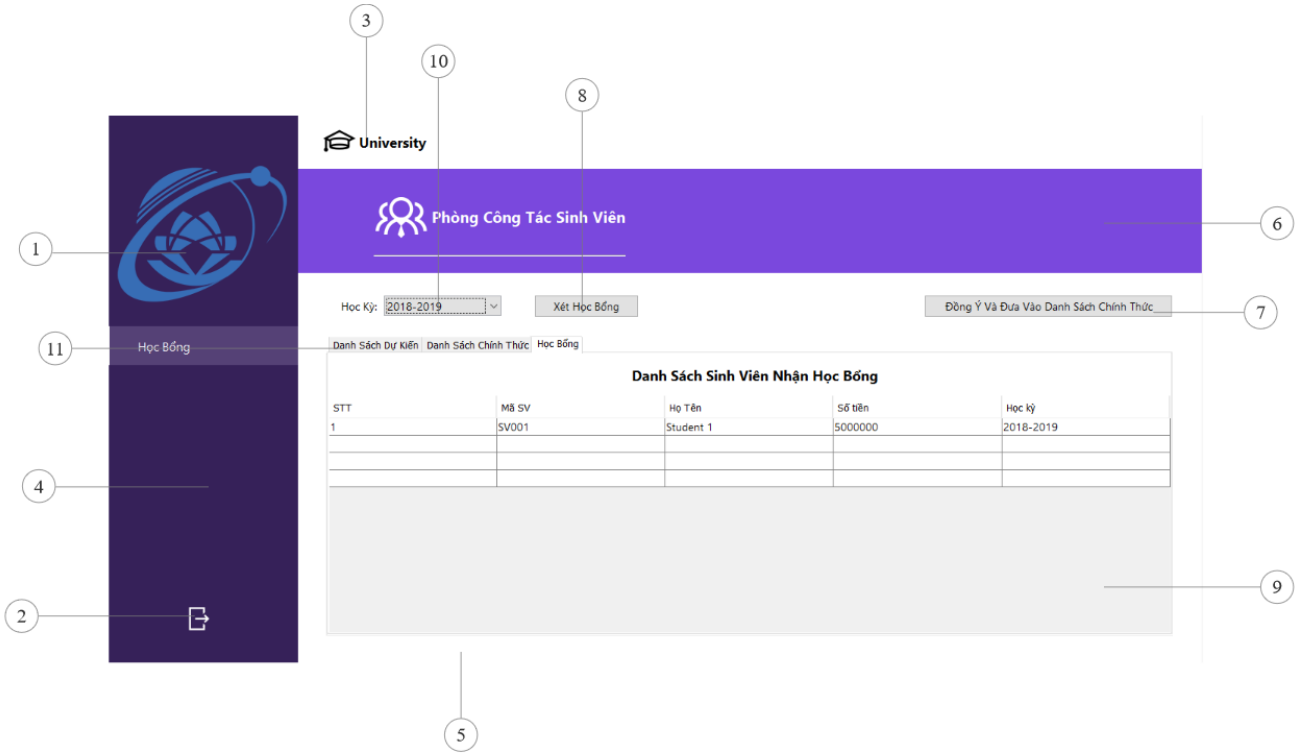
b. Danh sách chính thức:



STT	Tên	Kiểu	Chức năng
1.	Logo uit	Label	Hiển thị thông tin
2.	Logo đăng xuất	Label	Cho biết đây là nơi đăng xuất
3.	Logo university	Label	Hiển thị thông tin
4.	Khung hình	Panel	Chứa những Components khác

5.	Khung hình	Panel	Chứa những Components khác
6.	Khung hình	Panel	Chứa những Components khác
7.	Đồng ý và đưa vào danh sách chính thức	Button	Xác nhận những sinh viên ở bảng dự kiến đã đủ điều kiện để nhận học bỗng
8.	Xét học bỗng	Button	Kiểm tra xem sinh viên đó đã đủ điều kiện nhận học bỗng hay chưa
9.	Bảng	Table	Hiển thị thông tin sinh viên chính thức nhận học bỗng
10.	2018-2019	Combobox	Chọn học kì
11.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem

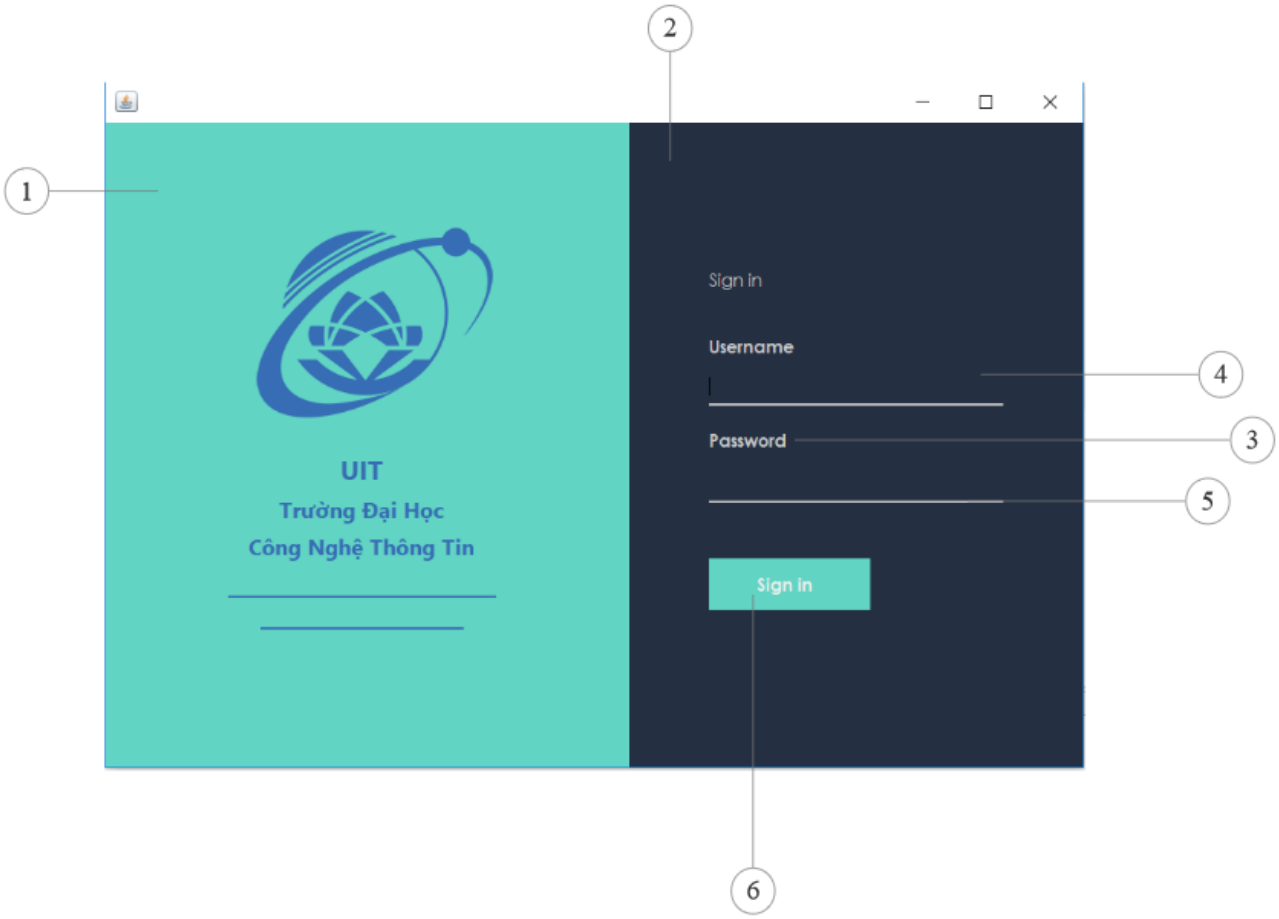
c. Học bổng:



STT	Tên	Kiểu	Chức năng
1.	Logo uit	Label	Hiển thị thông tin
2.	Logo đăng xuất	Label	Cho biết đây là nơi đăng xuất
3.	Logo university	Label	Hiển thị thông tin
4.	Khung hình	Panel	Chứa những Components khác
5.	Khung hình	Panel	Chứa những Components khác
6.	Khung hình	Panel	Chứa những Components khác
7.	Đồng ý và đưa vào danh sách chính thức	Button	Xác nhận những sinh viên ở bảng dự kiến đã đủ điều kiện để nhận học bổng

8.	Xét học bỗng	Button	Kiểm tra xem sinh viên đó đã đủ điều kiện nhận học bỗng hay chưa
9.	Bảng	Table	Hiển thị thông tin sinh viên sẽ nhận học bỗng
10.	2018-2019	Combobox	Chọn học kì
11.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem

2. Đăng nhập:

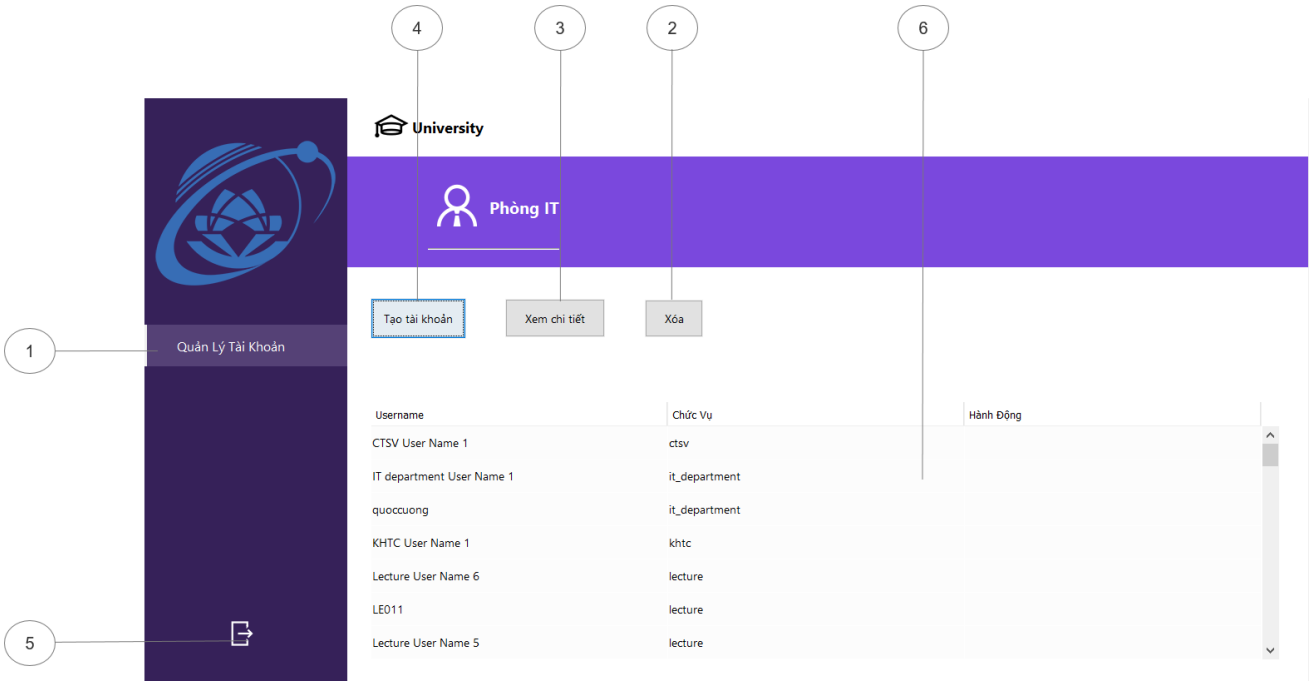


STT	Tên	Kiểu	Chức năng
1.	Khung hình	Panel	Chứa các thành phần khác
2.	Khung hình	Panel	Chứa các thành phần khác

3.	Username Password	Label	Hiển thị thông tin cho biết khung nhập bên dưới là nhập thông tin gì
4.	Username Password	Label	Hiển thị thông tin cho biết khung nhập bên dưới là nhập thông tin gì
5.	Thanh phân cách	Separator	Phân cách các component với nhau
6.	Sign in	Label	Thực hiện đăng nhập vào hệ thống

3. IT-department:

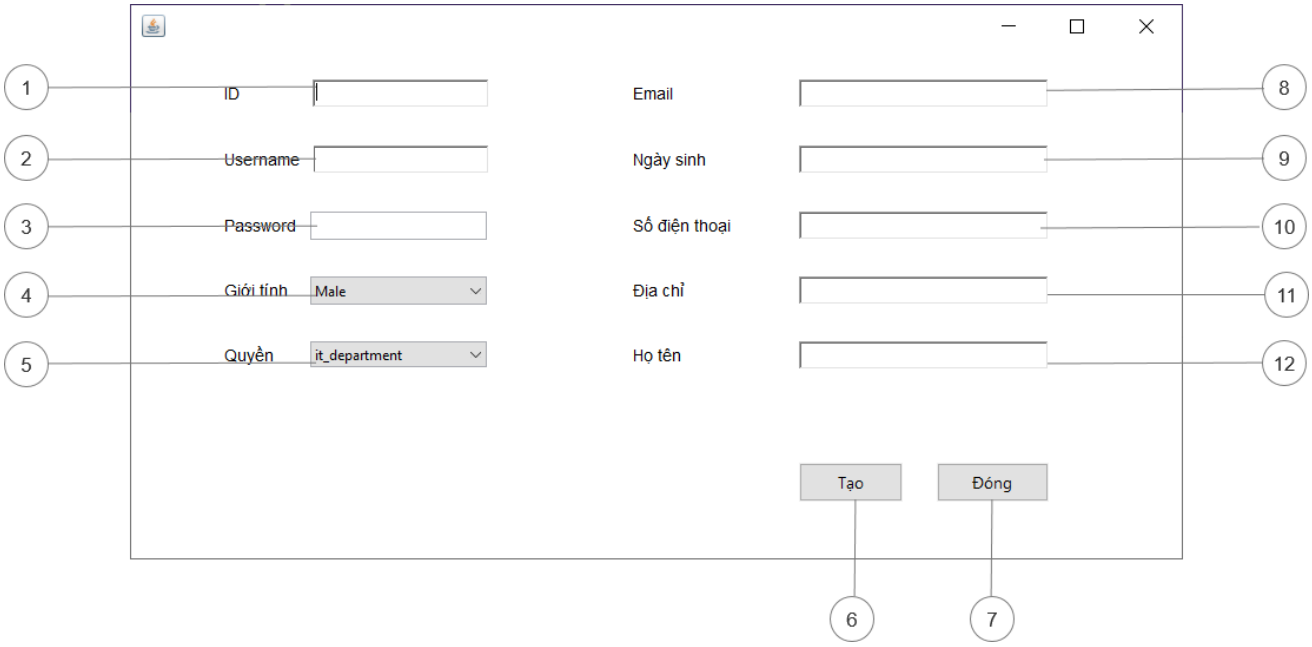
a. Quản lí phòng IT:



STT	Tên	Kiểu	Chức năng
1	Quản lý tài khoản	Lable	Hiển thị chức năng quản lý tài khoản

2	Xóa	Button	Thực hiện thao tác xóa trên một dòng dữ liệu
3	Xem chi tiết	Button	Hiển thị chi tiết danh sách từng phòng
4	Tạo tài khoản	Button	Thực hiện thao tác tạo tài khoản mới
5	Đăng xuất	Button	Đăng xuất khỏi hệ thống
6	Danh sách các phòng	Table Viewer	Hiển thị danh sách thông tin đăng ký học phần

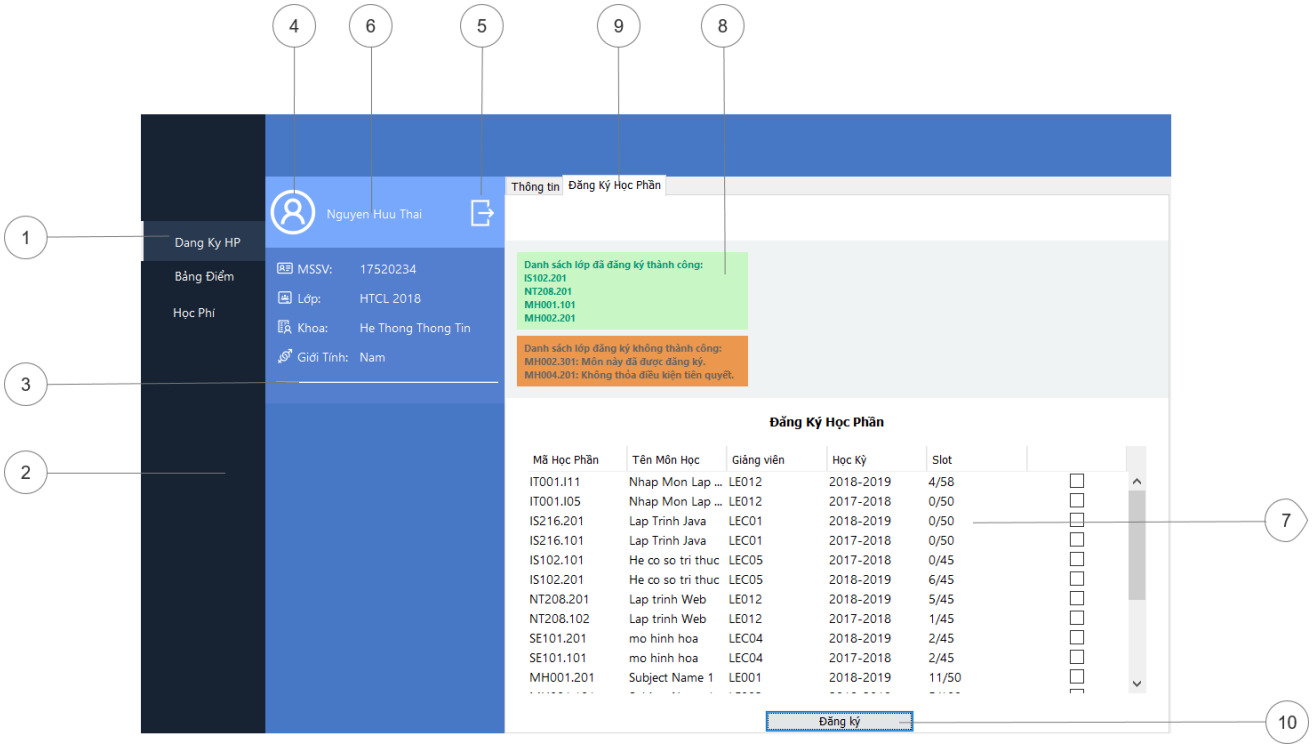
b. Tạo phòng:



STT	Tên	Kiểu	Chức năng
1	ID	Text	Nhập vào ID
2	Username	Text	Nhập vào Username
3	Password	Text	Nhập vào Password
4	Giới tính	Combobox	Nhập vào giới tính
5	Quyền	Combobox	Nhập vào quyền cho từng phòng ban
6	Tạo	Button	Thực hiện thao tác tạo cho phòng ban khi đã nhập vào các bước trên
7	Đóng	Button	Thực hiện thao tác đóng các màn hình
8	Email	Text	Nhập vào email
9	Ngày sinh	Date Time	Nhập vào ngày tháng năm sinh
10	Số điện thoại	Text	Nhập vào số điện thoại
11	Họ tên	Text	Nhập vào họ và tên
12	Địa chỉ	Text	Nhập vào địa chỉ

4. Student:

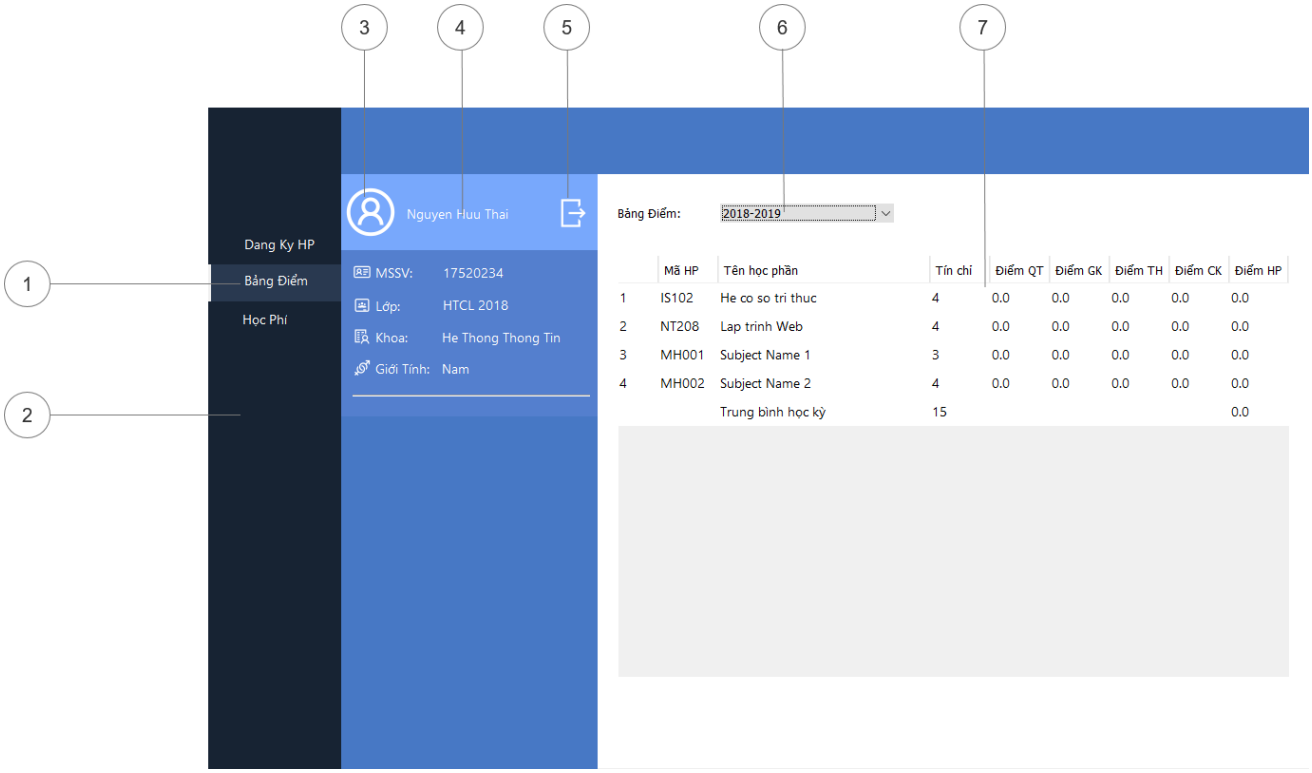
a. Quản lí đăng kí học phần:



STT	Tên	Kiểu	Chức năng
1	Quản lý đăng ký học phần	Lable	Hiển thị chức năng quản lý học phần
2	Khung hình	Panel	Chứa các thành phần khác
3	Thanh phân cách	Separator	Phân cách các Panel với nhau
4	Logo Studentname	Label	Hiển thị thông tin
5	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
6	Student name 1	Label	Hiển thị tên lecture đăng nhập
7	Danh sách đăng ký học phần	Table Viewer	Hiển thị danh sách thông tin đăng ký học phần
8	Thông tin đăng ký học phần	Textfield	Hiển thị thông tin thành công hay thất bại

9	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem

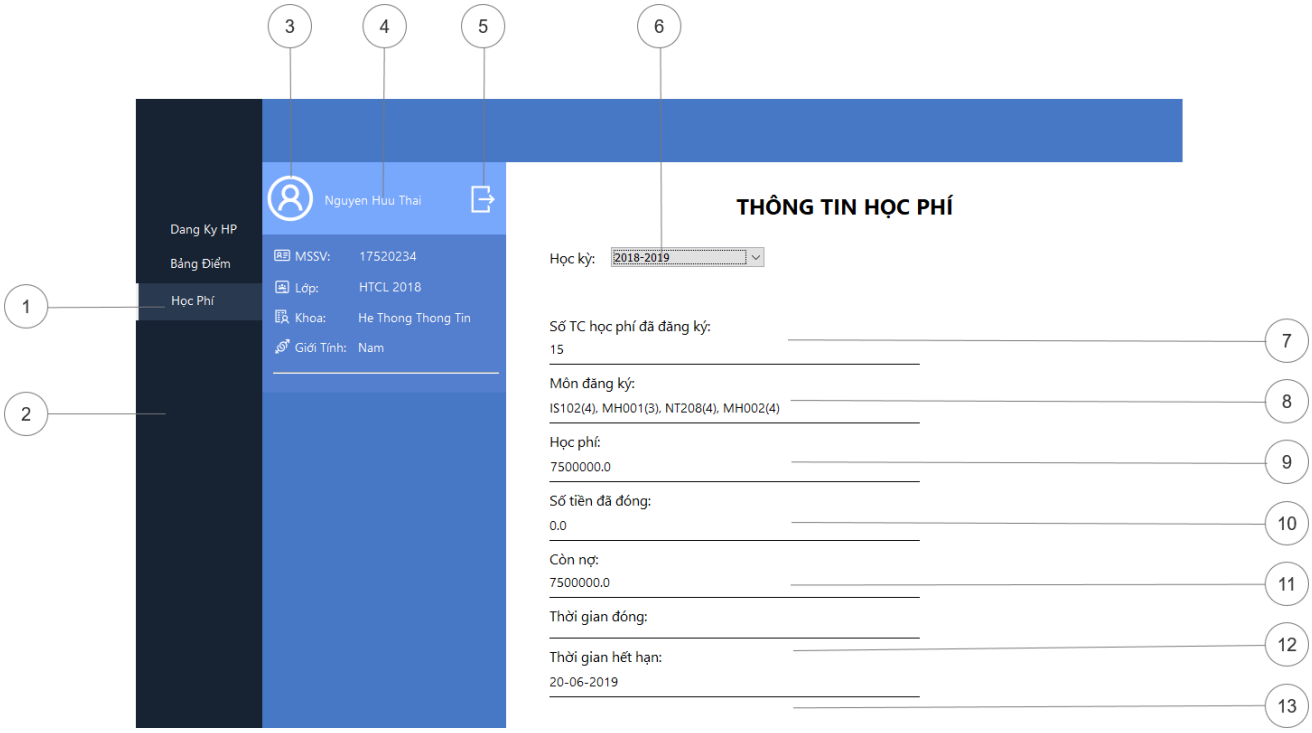
b. Quản lí bảng điểm:



STT	Tên	Kiểu	Chức năng
1	Quản lý bảng điểm	Lable	Hiển thị chức năng quản lý học phí
2	Khung hình	Panel	Chứa các thành phần khác
3	Logo Studentname	Label	Hiển thị thông tin
4	Student name 1	Label	Hiển thị tên lecture đăng nhập
5	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống

6	Học kì năm học	Combobox	Chọn học kì, cho biết điểm trong học kì đó
7	Danh dách bảng điểm	Table Viewer	Hiển thị điểm trong học kì đó

c. Quản lí học phí:



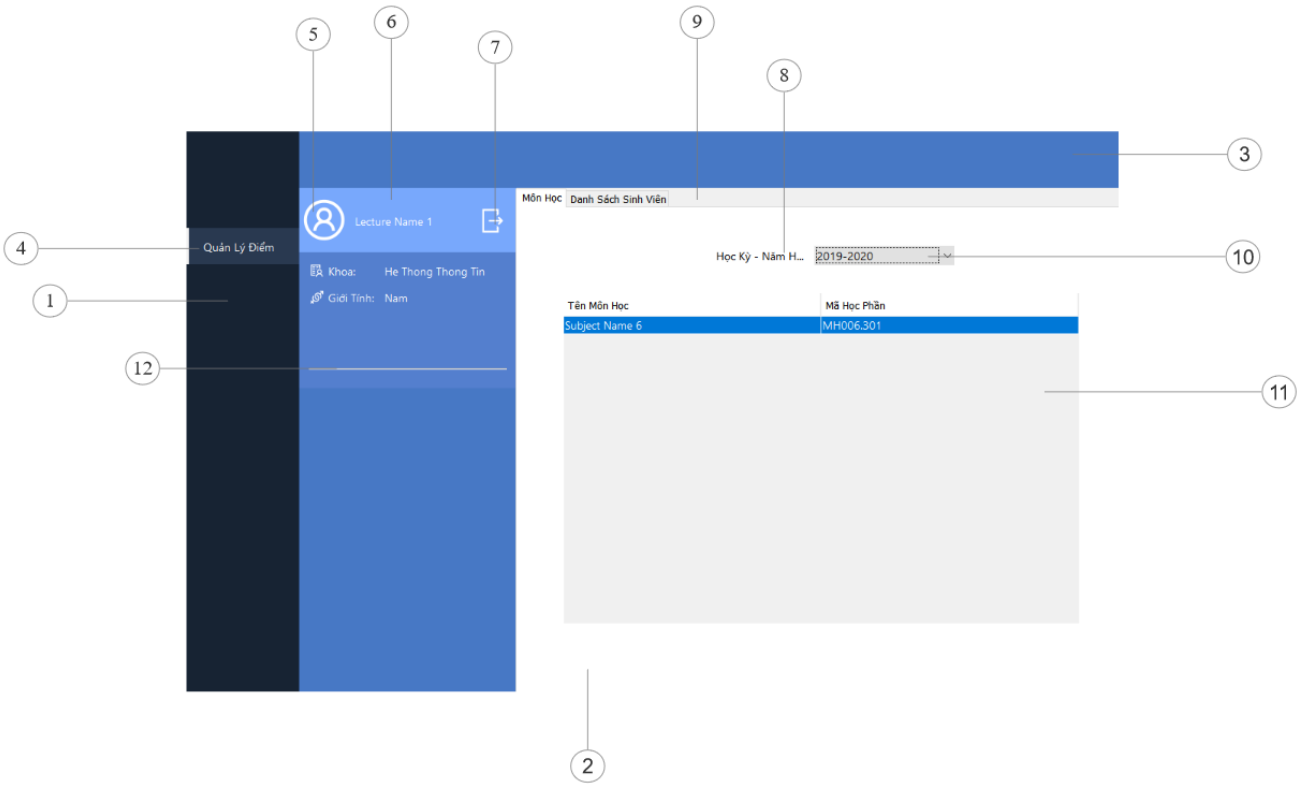
STT	Tên	Kiểu	Chức năng
1	Quản lý học phí	Lable	Hiển thị chức năng quản lý học phí
2	Khung hình	Panel	Chứa các thành phần khác
3	Logo Studentname	Label	Hiển thị thông tin
4	Student name 1	Label	Hiển thị tên lecture đăng nhập
5	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống

6	Học kì năm học	Combobox	Chọn học kì, cho biết trong học kì đã đóng học phí
7	Số tín chỉ đã đăng ký	Text	Hiển thị số tín chỉ đã đăng ký
8	Môn đăng ký	Text	Hiển thị môn đăng ký
9	Học phí	Text	Hiển thị học phí
10	Số tiền đã đóng	Text	Hiển thị số tiền đã đóng
11	Số tiền nợ	Text	Hiển thị số tiền nợ
12	Thời gian đóng	DateTime	Hiển thị thời gian đóng
13	Thời gian hết hạn	DateTime	Hiển thị thời gian hết hạn

5. Lecturer:

a. Quản lí điểm:

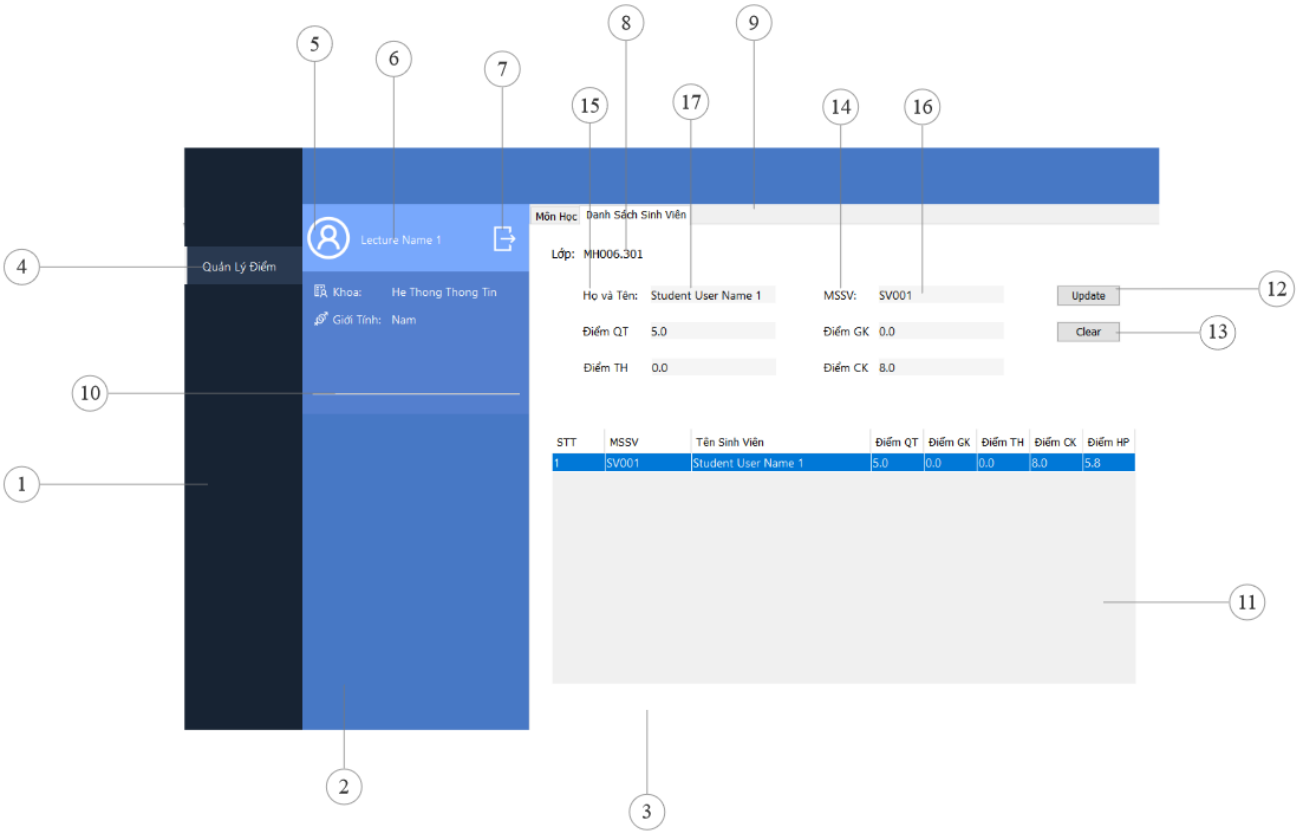
i. Môn học:



STT	Tên	Kiểu	Chức năng
1.	Khung hình	Panel	Chứa các thành phần khác
2.	Khung hình	Panel	Chứa các thành phần khác
3.	Khung hình	Panel	Chứa các thành phần khác
4.	Quản lí điểm	Label	Hiển thị chức năng là quản lí điểm
5.	Logo lecturename	Label	Hiển thị thông tin
6.	Lecturenam 1	Label	Hiển thị tên lecture đăng nhập
7.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
8.	Học kì năm học	Label	Hiển thị thông tin
9.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
10.	2019-2020	Combobox	Chọn học kì, cho biết trong học kì đó có bao nhiêu học phần được mở
11.	Bảng	Table	Hiển thị thông tin mà mình muốn, được lấy từ database. Khi chọn một dòng trong bảng thì sẽ cho biết thông tin học phần đó

12.	Thanh phân cách	Separator	Phân cách các Panel với nhau

ii. Danh sách sinh viên:

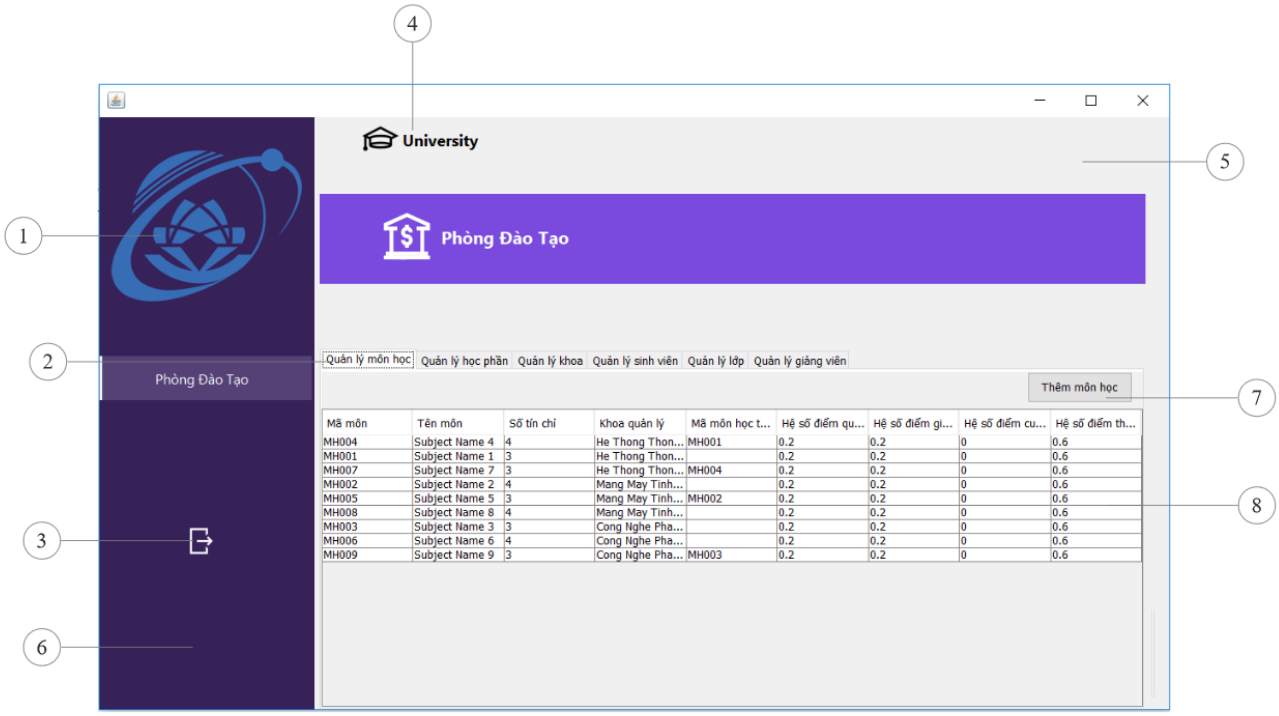


STT	Tên	Kiểu	Chức năng
1.	Khung hình	Panel	Chứa các thành phần khác
2.	Khung hình	Panel	Chứa các thành phần khác
3.	Khung hình	Panel	Chứa các thành phần khác
4.	Quản lí điểm	Label	Hiển thị chức năng là quản lí điểm
5.	Logo lecturename	Label	Hiển thị thông tin
6.	Lecturenam 1	Label	Hiển thị tên lecture đăng nhập
7.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
8.	MH006.301	Label	Hiển thị mã học phần được chọn từ bảng môn học
9.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem

10.	Thanh phân cách	Separator	Phân cách các component với nhau
11.	Bảng	Table	Hiển thị số lượng và điểm sinh viên học học phần đó
12.	Update	Button	Cập nhật lại điểm sinh viên sau khi đã điền các thông tin muốn sửa đổi, không được nhập điểm < 0 hoặc >10
13.	Clear	Button	Xóa những thông tin đã nhập trong khung nhập
14.	MSSV Điểm GK Điểm CK	Label	Hiển thị thông tin cho biết khung nhập kế bên là nhập thông tin gì
15.	Họ và tên Điểm QT Điểm TK	Label	Hiển thị thông tin cho biết khung nhập kế bên là nhập thông tin gì
16.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
17.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết

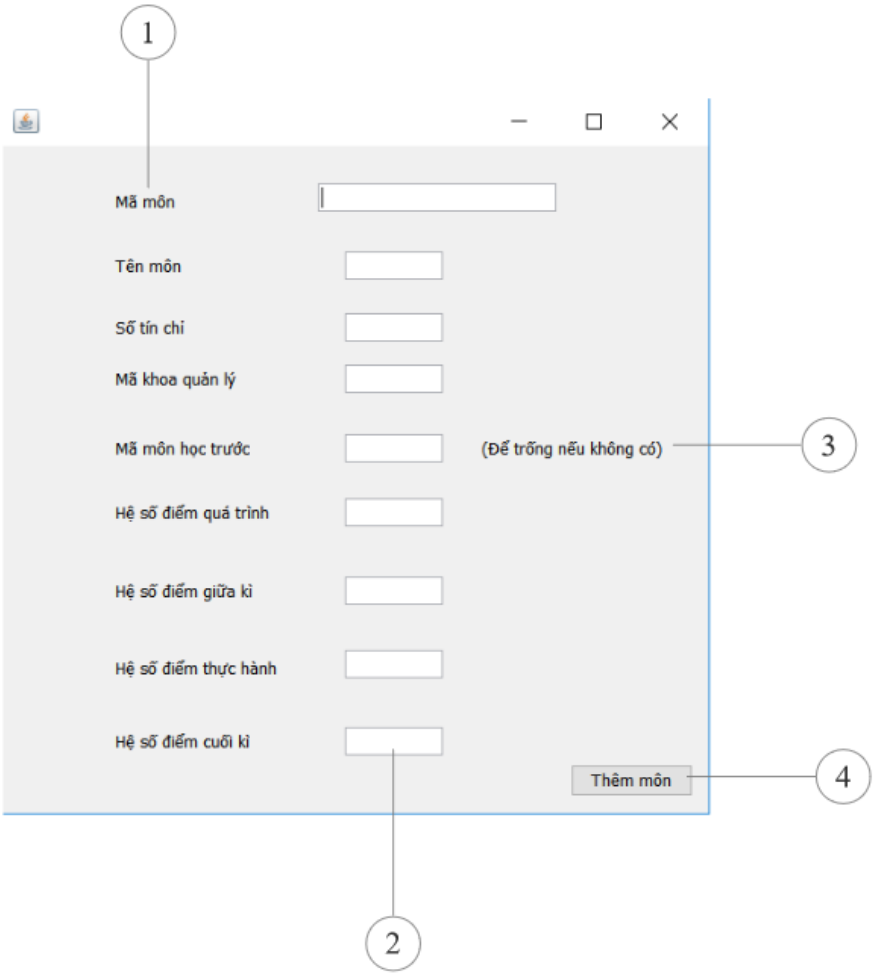
6. Phòng đào tạo:

a. Quản lí môn học:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác
6.	Khung hình	Panel	Chứa những component khác
7.	Thêm môn học	Button	Bấm vào để thêm môn học
8.	Bảng	Table	Hiển thị thông tin về quản lí môn học

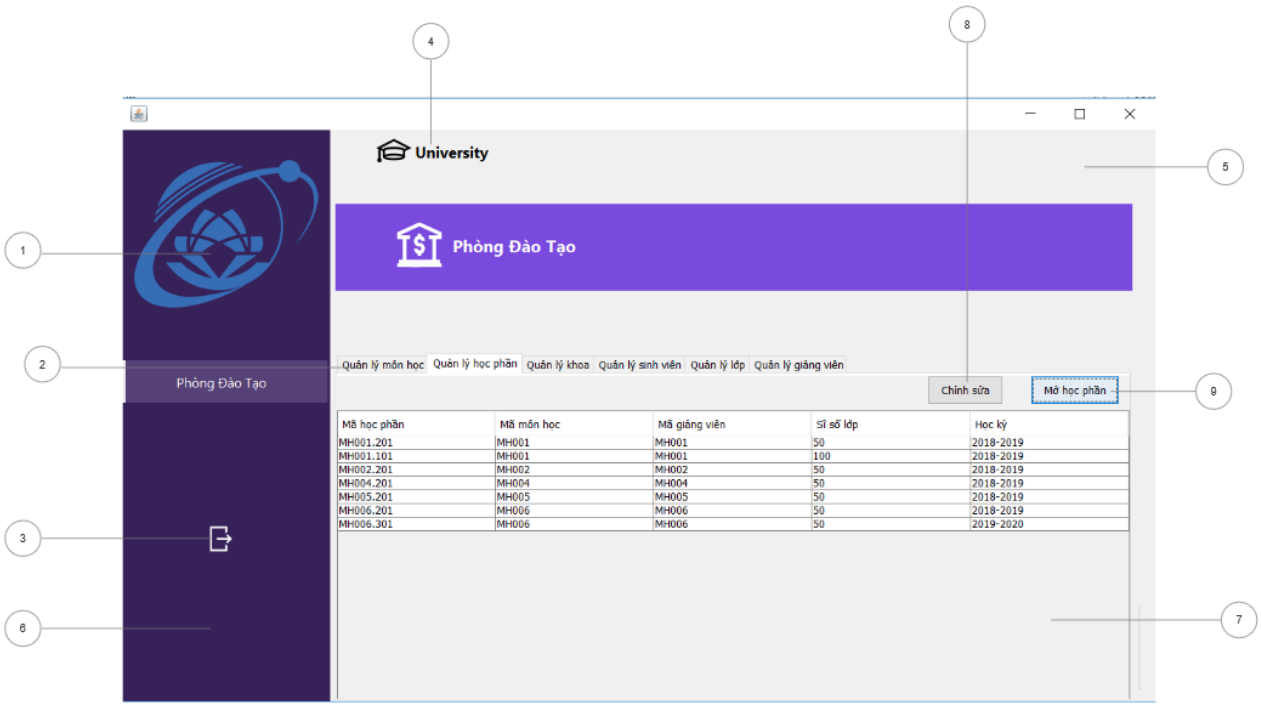
i. Thêm môn học:



STT	Tên	Kiểu	Chức năng
1.	Mã môn Tên môn Số tín chỉ Mã khoa quản lí Mã môn học trước Hệ số điểm quá trình Hệ số điểm giữa kì Hệ số điểm thực hành Hệ số điểm cuối kì	Label	Hiển thị thông tin

2.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
3.	(Để trống nếu không có)	Label	Hiển thị thông tin
4.	Thêm môn	Button	Thêm môn học vào bảng

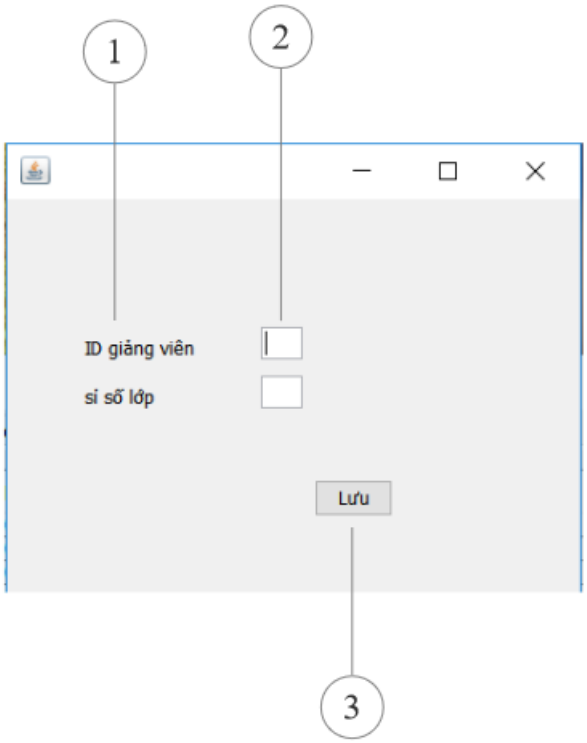
b. Quản lí học phần:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác
6.	Khung hình	Panel	Chứa những component khác
7.	Bảng	Table	Hiển thị thông tin về quản lí học phần
8.	Chỉnh sửa	Button	Chỉnh sửa học phần đã có

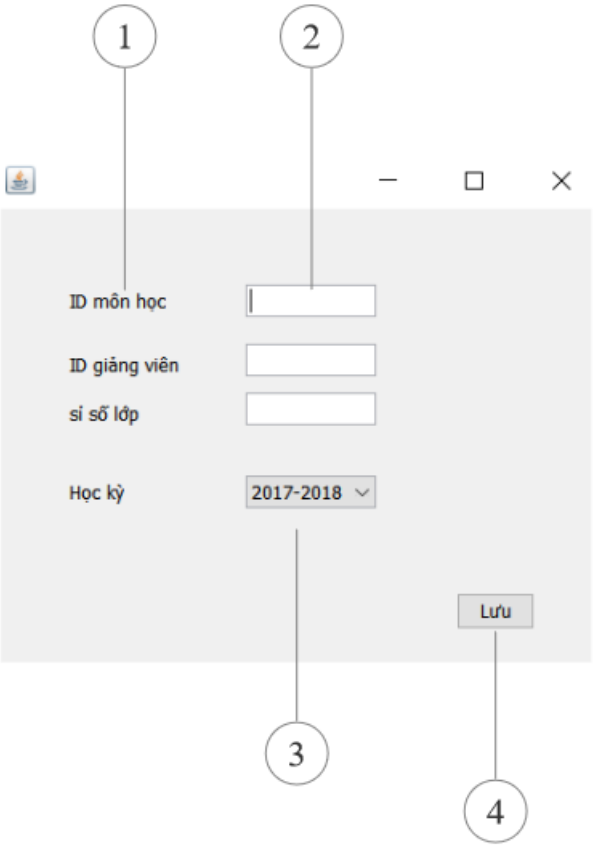
9.	Mở học phần	Button	Mở học phần mới

i. Chỉnh sửa học phần:



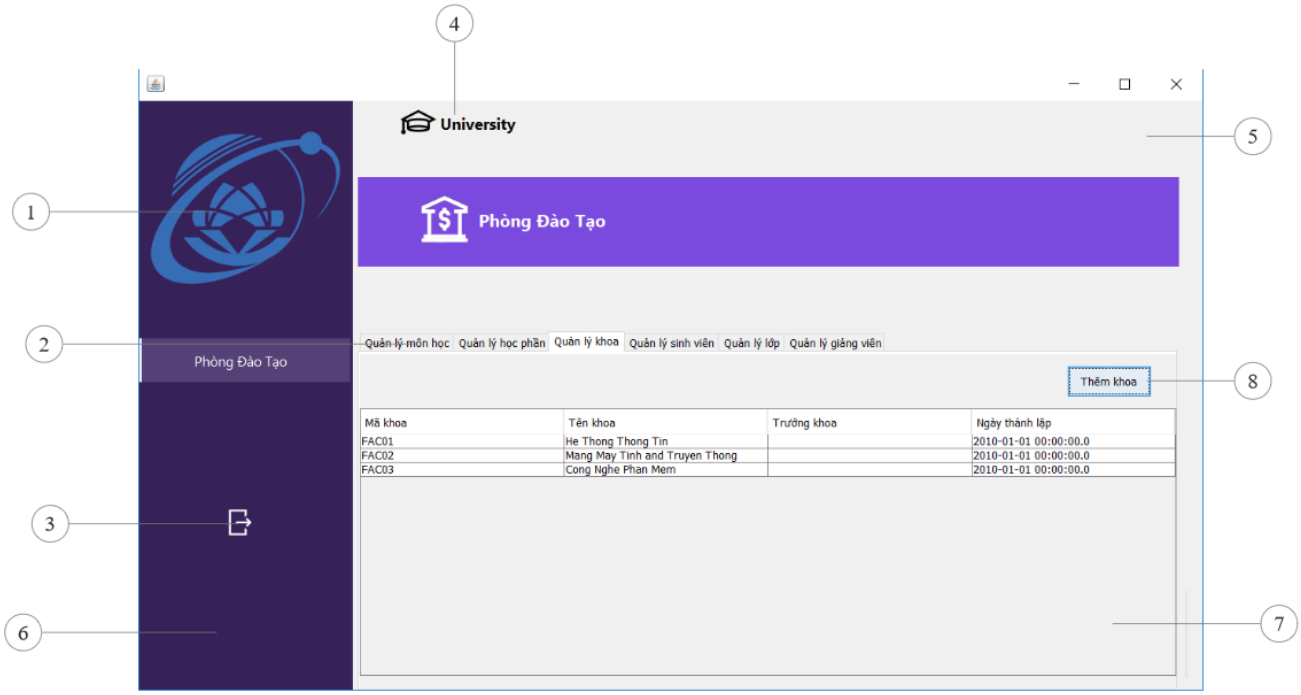
STT	Tên	Kiểu	Chức năng
1.	ID giảng viên Sỉ số lớp	Label	Hiển thị thông tin
2.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
3.	Lưu	Button	Lưu những thay đổi

ii. Mở học phần:



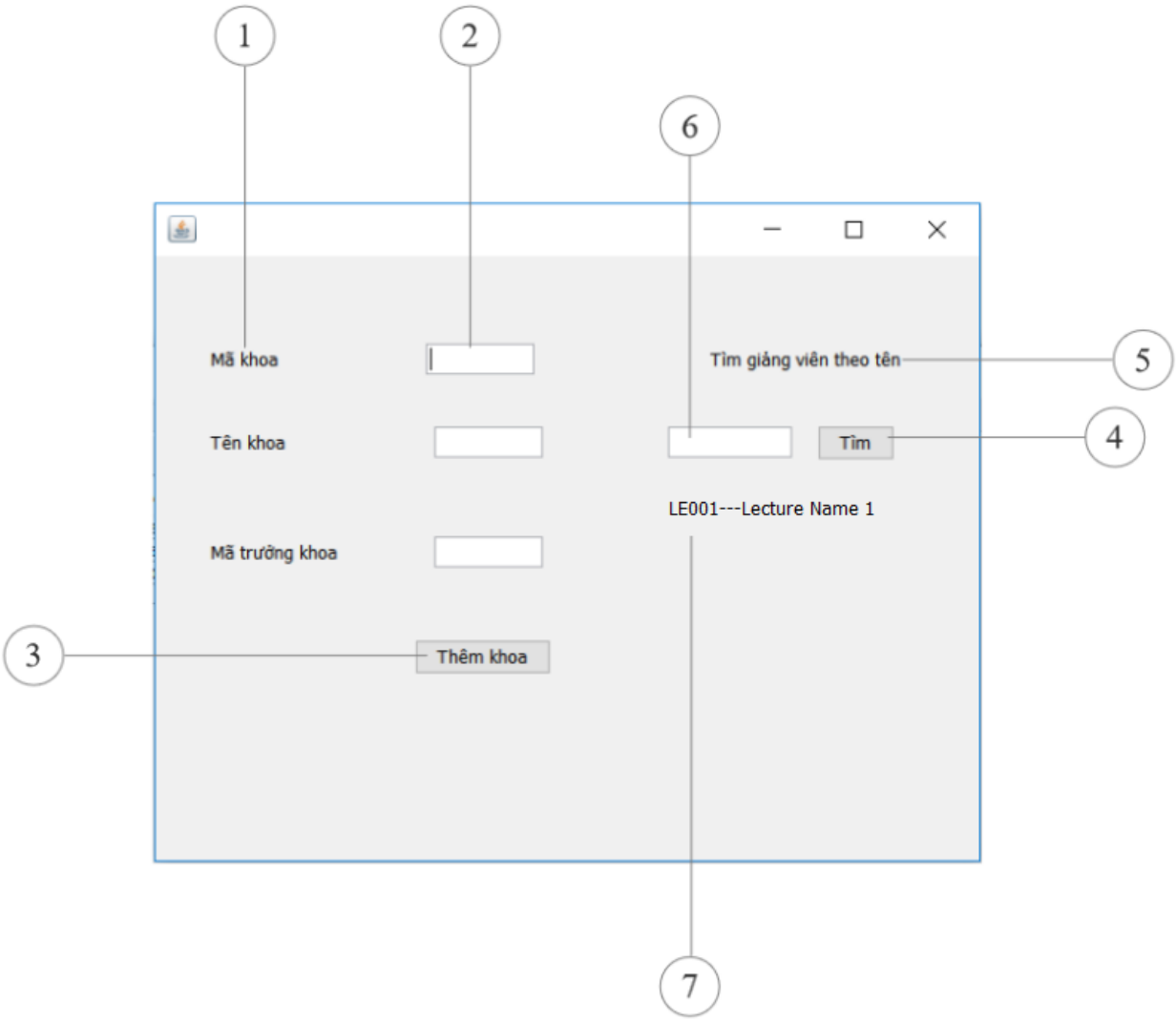
STT	Tên	Kiểu	Chức năng
1.	ID môn học ID giảng viên Sỉ số lớp Học kì	Label	Hiển thị thông tin
2.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
3.	2017-2018	Combobox	Chọn học kì để học phần
4.	Lưu	Button	Lưu những thay đổi

c. Quản lí khoa:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác
6.	Khung hình	Panel	Chứa những component khác
7.	Bảng	Table	Hiển thị thông tin về quản lí khoa
8.	Thêm khoa	Button	Thêm khoa mới

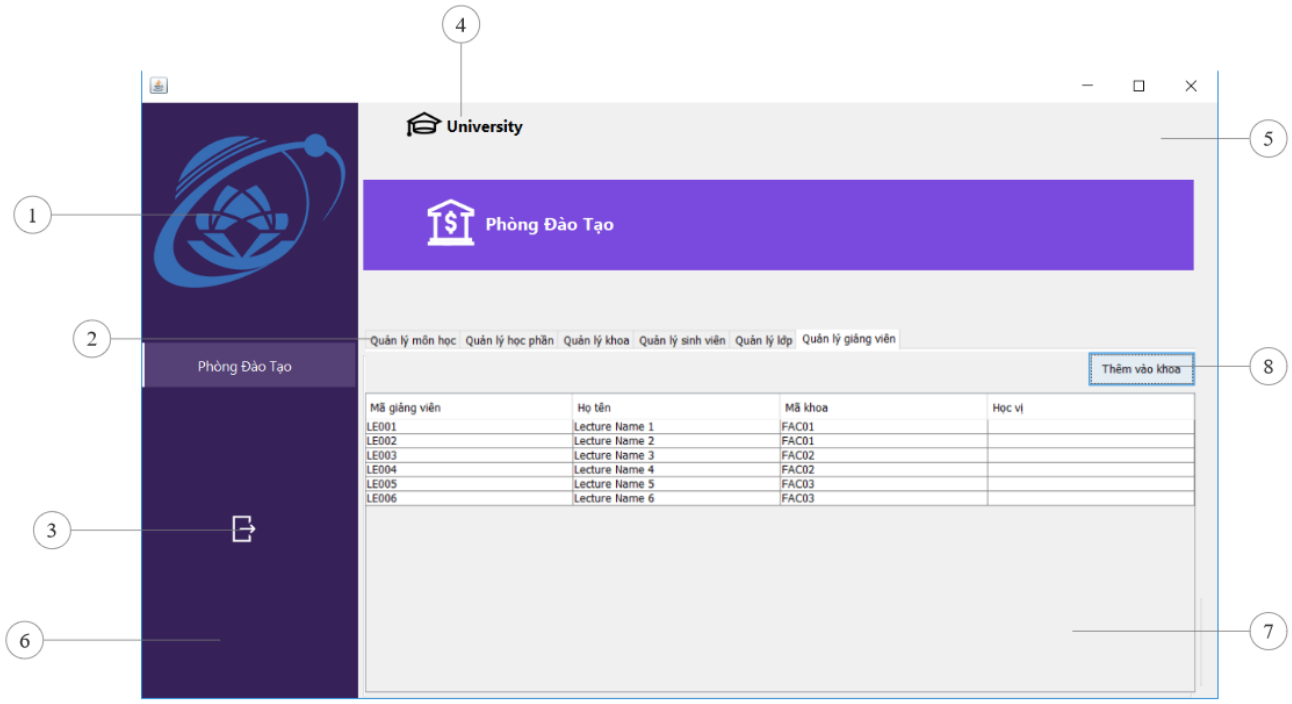
i. Thêm khoa:



STT	Tên	Kiểu	Chức năng
1.	Mã khoa Tên khoa Trưởng khoa	Label	Hiển thị thông tin
2.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
3.	Thêm khoa	Button	Thêm khoa vào hệ thống
4.	Tìm	Button	Tìm giảng viên có trong hệ thống

5.	Tìm giảng viên theo tên	Label	Hiển thị thông tin
6.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
7.	LE001- Lecture Name 1	Label	Tìm và hiển thị mã, tên giảng viên được tìm ở 6.

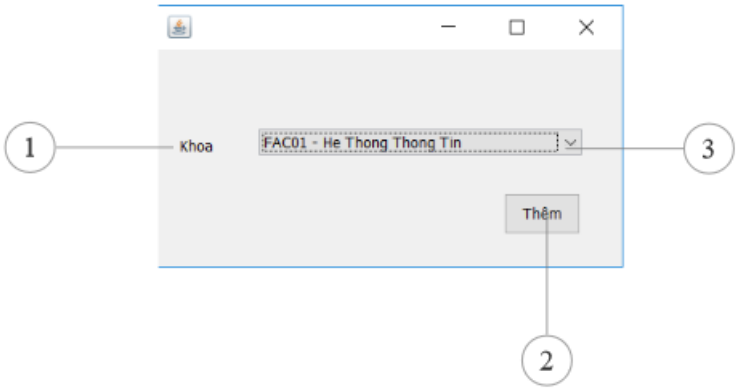
d. Quản lí giảng viên:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác

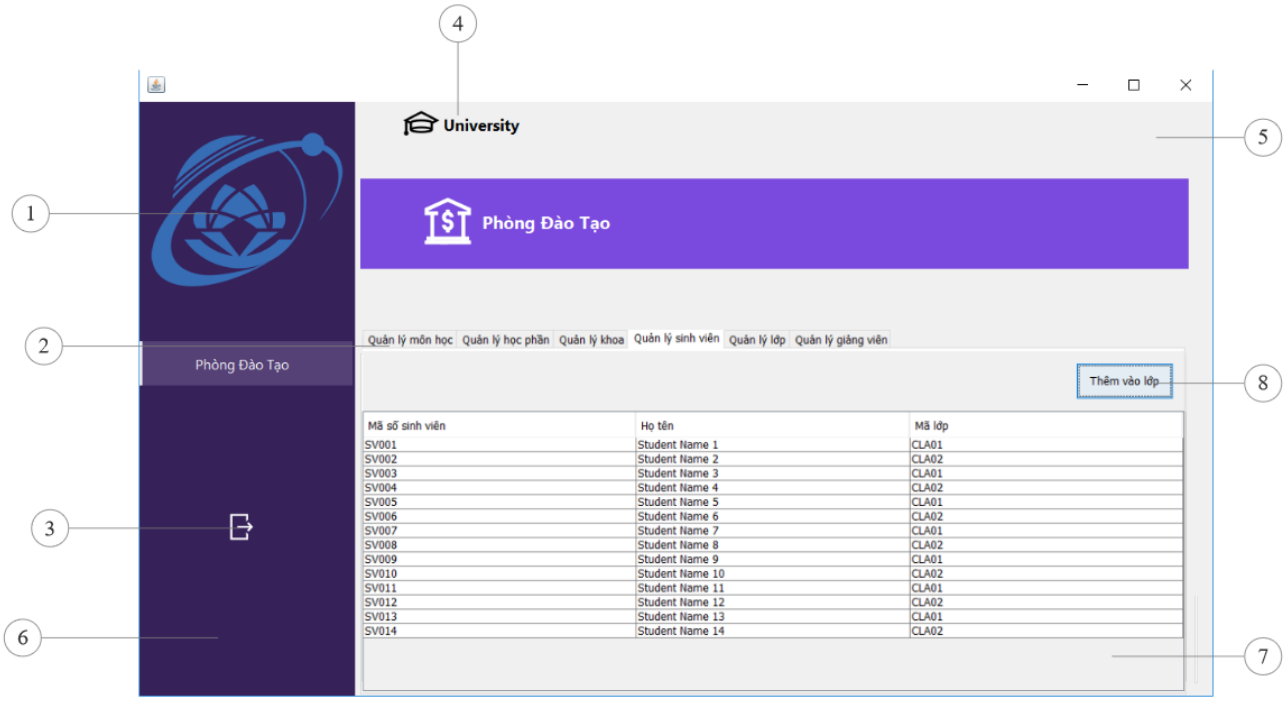
6.	Khung hình	Panel	Chứa những component khác
7.	Bảng	Table	Hiển thị thông tin về quản lí giảng viên
8.	Thêm vào khoa	Button	Thêm giảng viên mới vào khoa

i. Thêm vào khoa:



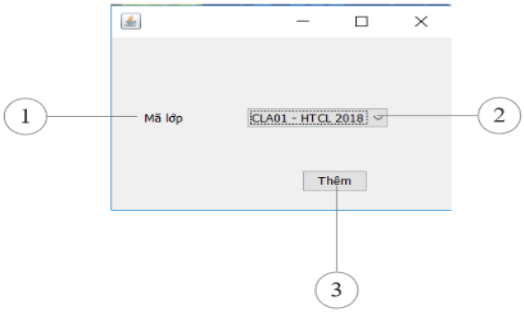
STT	Tên	Kiểu	Chức năng
1.	Khoa	Label	Hiển thị thông tin
2.	FA01-He Thong Thong Tin	Combobox	Lựa chọn khoa mới để thêm vào
3.	Thêm	Button	Thêm khoa mới

e. Quản lí sinh viên:



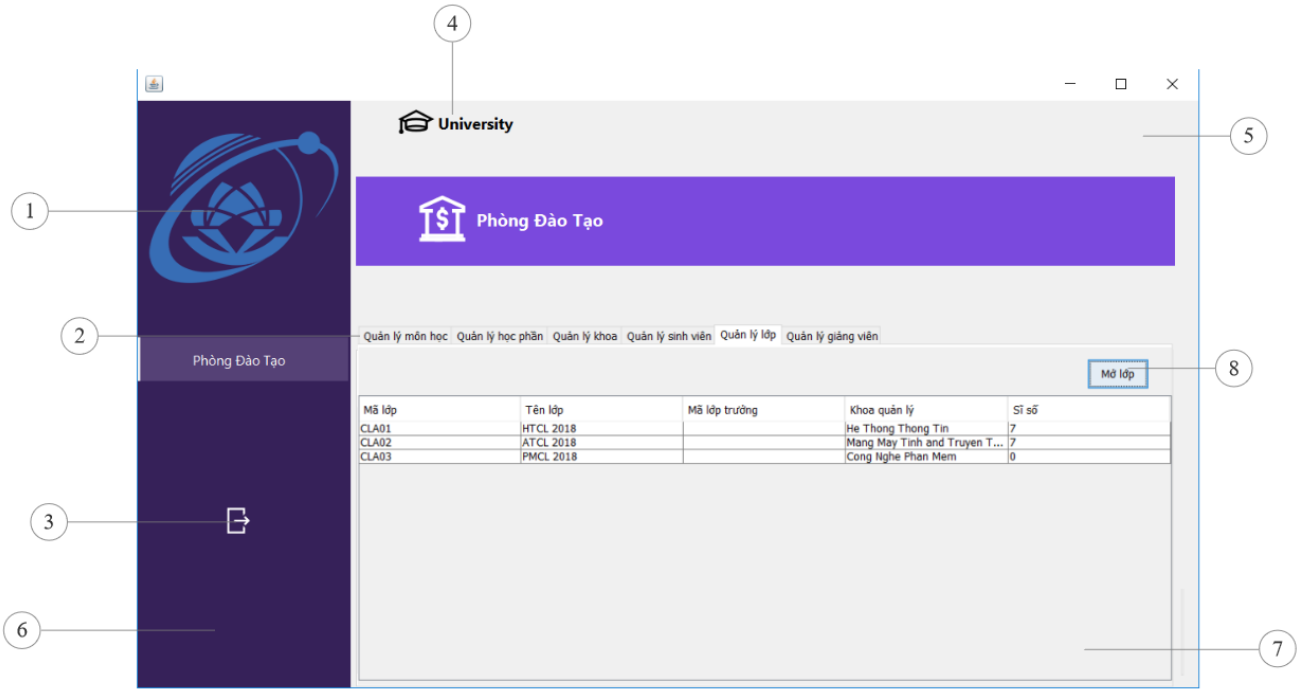
STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống
4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác
6.	Khung hình	Panel	Chứa những component khác
7.	Bảng	Table	Hiển thị thông tin về quản lí sinh viên
8.	Thêm vào lớp	Button	Thêm sinh viên mới vào lớp

i. Thêm vào lớp:



STT	Tên	Kiểu	Chức năng
1.	Mã lớp	Label	Hiển thị thông tin
2.	CLA01-HTCL 2018	Combobox	Lựa chọn lớp để thêm sinh viên vào
3.	Thêm	Button	Thêm sinh viên vào lớp

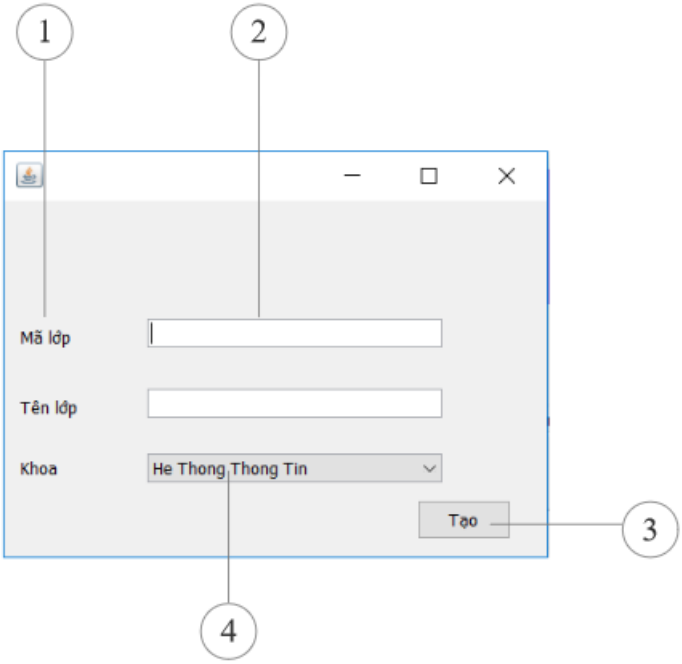
f. Quản lí lớp:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Thanh lựa chọn	TabbedPane	Lựa chọn những bảng muốn xem
3.	Logo đăng xuất	Label	Đăng xuất khỏi hệ thống

4.	Logo university	Label	Hiển thị chức năng là quản lí điểm
5.	Khung hình	Frame	Chứa những component khác
6.	Khung hình	Panel	Chứa những component khác
7.	Bảng	Table	Hiển thị thông tin về quản lí lớp
8.	Mở lớp	Button	Mở lớp mới

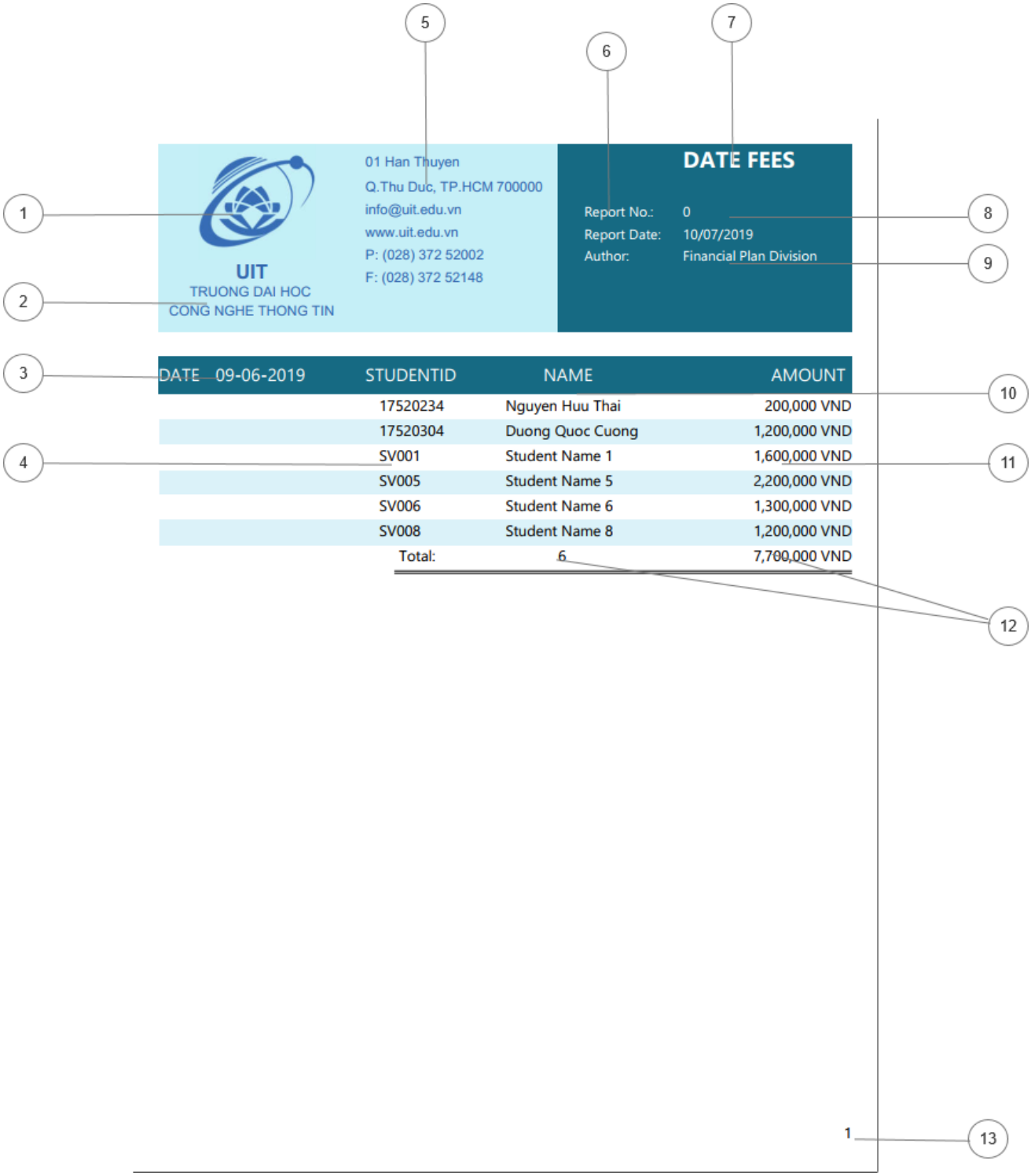
i. Mở lớp:



STT	Tên	Kiểu	Chức năng
1.	Logo UIT	Label	Hiển thị thông tin
2.	Khung trống	Textfield	Nhập thông tin mà label kế bên đã cho biết
3.	Tạo	Button	Tạo lớp mới
4.	He Thong Thong Tin	Combobox	Chọn khoa để mở lớp trong khoa đó

7. Report

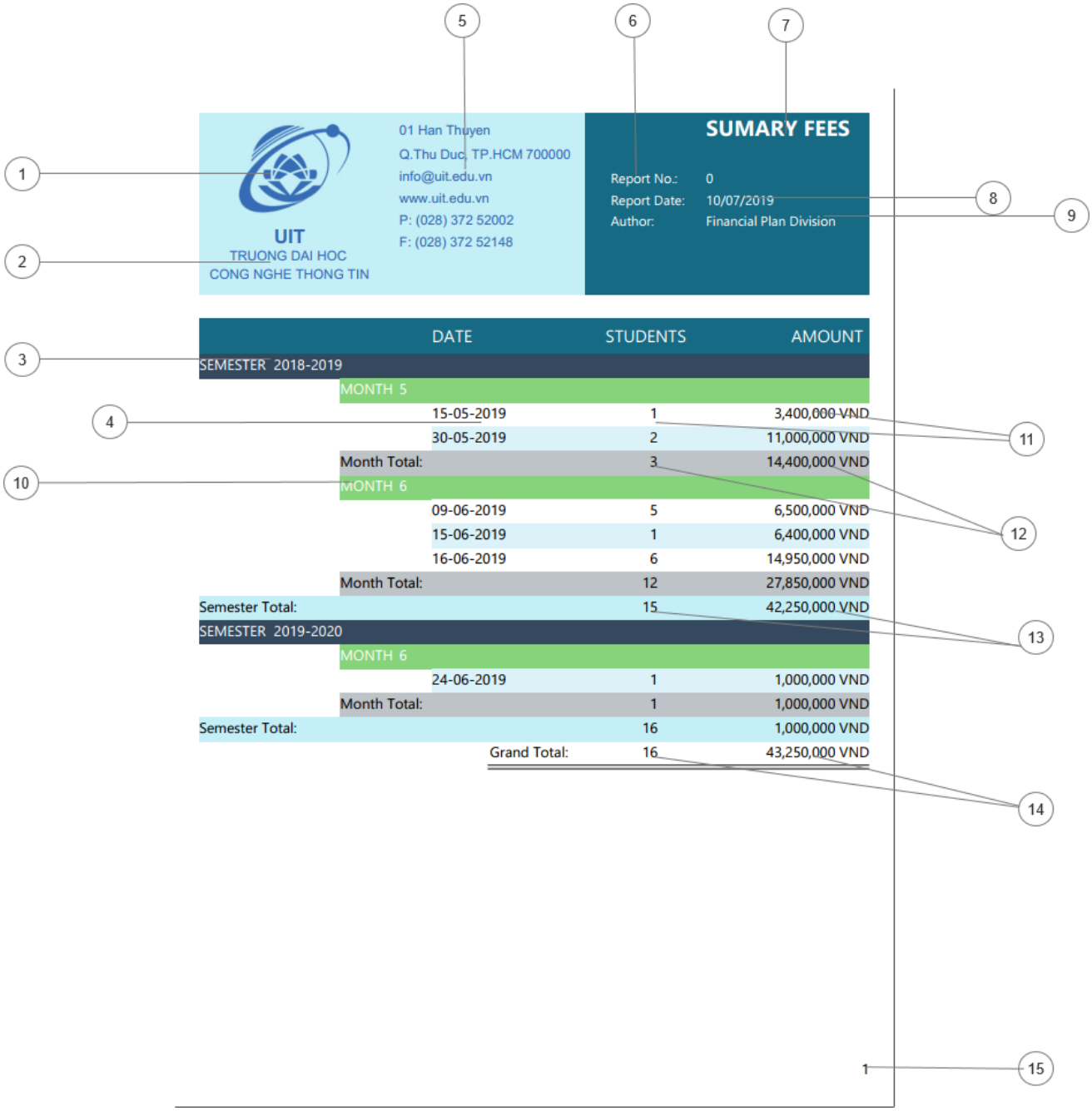
a. Date Report



STT	Tên	Kiểu	Chức năng
1.	Logo		Hiển thị Logo của trường
2.	Tên trường		Hiển thị tên trường
3.	Ngày báo cáo		Hiển thị ngày mà sinh viên đóng học phí dùng để lập báo cáo

4.	Mã số sinh viên		Hiển thị mã số sinh viên
5.	Địa chỉ		Hiển thị thông tin liên lạc của trường
6.	Số Report		Hiển thị số Report được tạo
7.	Tên Report		Hiển thị tên loại Report
8.	Thời gian		Thời gian khi lập báo cáo
9.	Tác giả		Nhân viên lập báo cáo
10.	Tên sinh viên		Hiển thị tên sinh viên đã đóng học phí trong ngày
11.	Số tiền		Hiển thị số tiền sinh viên đã đóng học phí trong ngày
12.	Tổng tiền và tổng số sinh viên		Hiển thị tổng số tiền và tổng số sinh viên đã đóng học phí trong ngày
13.	Số thứ tự		Hiển thị số trang của report

b. Sumary Report



STT	Tên	Kiểu	Chức năng
1.	Logo		Hiển thị Logo của trường
2.	Tên trường		Hiển thị tên trường
3.	Học kỳ		Hiển thị tên học kỳ
4.	Ngày		Hiển thị ngày mà sinh viên đóng học phí ở trong mỗi tháng
5.	Địa chỉ		Hiển thị thông tin liên lạc của trường

6.	Số Report		Hiển thị số Report được tạo
7.	Tên Report		Hiển thị tên loại Report
8.	Thời gian		Thời gian khi lập báo cáo
9.	Tác giả		Nhân viên lập báo cáo
10.	Tháng		Hiển thị tháng sinh viên đóng học phí trong mỗi học kỳ
11.	Tổng số sinh viên và số tiền trong ngày		Hiển thị tổng số tiền sinh viên và tổng số tiền đã đóng trong ngày
12.	Tổng tiền và tổng số sinh viên trong tháng		Hiển thị tổng số tiền và tổng số sinh viên đã đóng học phí trong tháng
13.	Tổng tiền và tổng số sinh viên trong học kỳ		Hiển thị tổng số tiền và tổng số sinh viên đã đóng học phí trong học kỳ.
14.	Tổng tất cả tiền và tổng tất cả số sinh viên đã đóng học phí		Hiển thị tổng tất cả số tiền và tổng tất cả số sinh viên đã đóng học phí.
15.	Số thứ tự		Hiển thị số trang của report

CHƯƠNG 7: KẾT LUẬN

I. Kết quả đạt được:

Trong suốt quá trình làm đồ án vừa qua, nhóm đã đạt được những điều sau:

- Biết cách thiết kế, hoàn thiện một dự án nhỏ.
- Nắm được kĩ năng phân chia công việc, sắp xếp thời gian, làm việc nhóm.
- Hiểu và vận dụng được kiến thức của môn hệ quản trị cơ sở dữ liệu, cụ thể là làm việc trên Oracle, như:
    - Các cú pháp và cách suy nghĩ của ngôn ngữ PL/SQL.

- Cách thực hiện một Trigger, Function, Stored Procedure.
- Trang bị thêm kiến thức về khóa và các mức cô lập.
- Giải quyết được các trường hợp truy xuất đồng thời.

- Xây dựng được phần mềm "Quản lý học vụ" với các chức năng sau:

  - Quản lý IT-department.
  - Quản lý Student.
  - Quản lý Lecture.
  - Quản lý Phòng đào tạo.
  - Bảng đăng nhập.

II. Hạn chế:

Do thời gian ngắn cộng với yêu cầu công việc ban đầu khá lớn nên trong quá trình thực hiện đồ án nhóm còn gặp phải một số hạn chế :

- Giao diện không được chăm chút.
- Các tình huống truy xuất đồng thời có thể không hợp lí.
- Phân chia công việc có thể không đều.
- Không thực hiện đầy đủ những yêu cầu đề ra ban đầu.

III. Hướng phát triển:

Để khắc phục các hạn chế còn tồn tại cũng như phát triển phần mềm nhóm có một số đề xuất như sau:

- Dùng những công nghệ hiện đại hơn.
- Phát triển thành một app được thật nhiều người sử dụng
- Phần mềm hỗ trợ đa ngôn ngữ.

Phụ lục 1: Bảng phân công công việc:

Công việc	Huy Cường	Quốc Cường	Quang Thiện	Minh Đức
Phân tích bài toán, xác định chức năng	x	x	x	x
Thiết kế CSDL	x	x	x	x
Thiết kế giao diện		x		x
Xử lí giao diện	x	x		
Xử lí các vấn đề liên quan đến truy xuất đồng thời	x	x		
Test chức năng	x	x		
Soạn và chỉnh sửa báo cáo		x	x	x

Phụ lục 2: Tài liệu tham khảo:

1. Các slide môn Hệ quản trị cơ sở dữ liệu của khoa Hệ thống thông tin
2. https://docs.oracle.com/en/database/oracle/oracle-database/index.html : tài liệu Oracle của hãng Oracle
3. https://docs.oracle.com/javase/tutorial/:tài liệu về Java, Java Swing