HỌC PHẦN THAY THẾ TỐT NGHIỆP 2

CHUYÊN NGÀNH CÔNG NGHỆ PHẦN MỀM

PHÂN TÍCH THIẾT KẾ
ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM

NGUYỄN MẠNH HÙNG

ĐỖ THỊ BÍCH NGỌC

GIỚI THIỆU [1]

TÀI LIỆU THAM KHẢO [2]

[1] Object-Oriented and Classical Software Engineering, Stephen R. Schach, Eigtth Edition, Mc [3]
Graw Hill, 2010.
[2] . *Mastering Software Quality Assurance: Best Practices, Tools and Techniques for Software Developers*, Murali Chemuturi, J. Ross Publication Inc., 2011.

MỤC LỤC 1

MỤC LỤC.....2 2
CHƯƠNG 1: QUY TRÌNH VÀ VÒNG ĐỜI PHÁT TRIỂN PHẦN MỀM.....4 3
1.1 PHẦN MỀM VÀ PHÁT TRIỂN PHẦN MỀM.....4 4
1.1.1. Phần mềm.....4 5
1.1.2. Phát triển phần mềm.....4
1.2 VÒNG ĐỜI PHÁT TRIỂN PHẦN MỀM.....6
1.2.1 Quy trình phát triển phần mềm hướng đối tượng.....6
1.2.2 Một số mô hình vòng đời phát triển phần mềm.....10
1.3 UML - CÔNG CỤ HỖ TRỢ PHÂN TÍCH THIẾT KẾ HƯỚNG ĐỐI TƯỢNG.....18
1.3.1 Lịch sử ra đời của UML.....18
1.3.2 UML – Ngôn ngữ mô hình hoá hướng đối tượng.....19
1.3.3 Các khái niệm cơ bản trong UML.....19
1.3.4 Các biểu đồ UML.....21
1.4 CÂU HỎI ÔN TẬP.....34
CHƯƠNG 2: ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM.....35 6
2.1 TỔNG QUAN VỀ ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM 35 7
2.1.1 Một số khái niệm 35
2.1.2 Các tiêu chí chất lượng 36
2.2 CÁC HOẠT ĐỘNG ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM 41
2.2.1. Đảm bảo chất lượng đặc tả 41
2.2.2. Đảm bảo chất lượng phân tích thiết kế 42
2.2.3. Đảm bảo chất lượng phát triển phần mềm (lập trình) 43
2.2.4. Kiểm thử phần mềm 44
2.3. CÁC KỸ THUẬT RÀ SOÁT 46
2.3.1. Mục tiêu của rà soát 46
2.3.2. Các hình thức rà soát 46
2.4. CÁC KỸ THUẬT KIỂM THỬ 52
2.4.1 Kỹ thuật kiểm thử hộp đen 52
2.4.2 Kỹ thuật kiểm thử hộp trắng 54
2.5 CÂU HỎI ÔN TẬP.....55
CHƯƠNG 3: THU THẬP VÀ PHÂN TÍCH YÊU CẦU.....56 8
3.1 THU THẬP YÊU CẦU.....56 9
3.1.1 Tìm hiểu lĩnh vực chuyên môn.....56 10
3.1.2 Mô tả hệ thống bằng ngôn ngữ tự nhiên.....58
3.1.3 Mô tả hệ thống bằng ngôn ngữ UML - use case.....62
3.2 PHÂN TÍCH YÊU CẦU.....68 11
3.2.1 Viết kịch bản.....68 12
3.2.2 Trích lớp thực thể.....72
3.2.3 Trích các lớp biên và điều khiển.....76
3.2.4 Phân tích hoạt động.....83
3.3 BÀI TẬP.....91
CHƯƠNG 4: THIẾT KẾ.....92 13
4.1 THIẾT KẾ LỚP THỰC THỂ.....92 14
4.1.1 Quy trình thực hiện.....92 15

4.1.2 Áp dụng.....92 1
4.2 THIẾT KẾ CSDL.....94 2
4.2.1 Quy trình thực hiện.....94 3
4.2.2 Áp dụng.....95 4
4.3 THIẾT KẾ CHI TIẾT CHO MODUL.....96 5
4.3.1 Thiết kế tĩnh.....96 6
4.3.2 Thiết kế hoạt động.....103
4.3.3 Thiết kế triển khai.....113
4.4 CÂU HỎI ÔN TẬP.....114 7
CHƯƠNG 5: CÀI ĐẶT HỆ THỐNG.....115 8
5.1 TỔ CHỨC DỰ ÁN.....115 9
5.2 CÀI ĐẶT CÁC MODUL.....116
5.2.1 Chức năng đăng kí học.....116 10
5.2.2 Chức năng nhập điểm.....142
5.2.3 Chức năng xem thống kê.....149
5.3 KIỂM THỬ ĐƠN VỊ 157 11
5.3.Xây dựng bộ test case cho kiểm thử đơn vị.....158 12
5.3.2 Cài đặt kiểm thử đơn vị.....159 13
5.4 CÂU HỎI ÔN TẬP.....169 14
CHƯƠNG 6: RÀ SOÁT VÀ KIỂM THỬ HỆ THỐNG.....170 15
6.1 THỰC HIỆN CÁC HOẠT ĐỘNG RÀ SOÁT 170 16
6.1.1 Rà soát đặc tả.....170 17
6.1.2 Rà soát phân tích.....172
6.1.3 Rà soát thiết kế.....177
6.1.4 Rà soát code.....182
6.2. THỰC HIỆN TEST CHỨC NĂNG 184
6.3 CÂU HỎI ÔN TẬP.....233
BÀI TẬP DỰ ÁN.....234 18

CHƯƠNG 1: QUY TRÌNH VÀ VÒNG ĐỜI PHÁT TRIỂN PHẦN MỀM 1

1.1 PHẦN MỀM VÀ PHÁT TRIỂN PHẦN MỀM 2

1.1.1. Phần mềm 3

Đặc trưng của phần mềm 4

- Phần mềm không mòn 5
- Phần mềm được phát triển mà không được sản xuất theo nghĩa thông thường
- Mặc dù công nghiệp phần mềm đang hướng đến phát triển dựa trên thành phần nhưng phần lớn phần mềm phát triển dựa theo yêu cầu của khách hàng.
- Cho đến nay những đặc trưng của phần mềm vẫn còn là vấn đề tranh cãi. Chính điều này thể hiện sự chưa trưởng thành của ngành học CÔNG NGHỆ PHẦN MỀM.

Các kiểu phần mềm 6

- **Phần mềm hệ thống:** Tập hợn các Chương trình được viết để phục vụ các chương trình khác, tương tác với phần cứng (ví dụ, biên dịch, trình soạn thảo, HĐH…) 7
- **Phần mềm thời gian thực:** Phần mềm điều phối/phân tích kiểm soát, đáp ứng thời gian
- **Phần mềm nghiệp vụ:** Các phần mềm tính lương, kế toán, quản lý kho…
- **Phần mềm khoa học và công nghệ:** Các ứng dụng trong thiên văn, sinh học phân tử, điều khiển tàu con thoi,…
- **Phần mềm nhúng:** Nằm trong bộ nhớ chỉ đọc điều khiển các sản phẩm và hệ thống
- **Phần mềm máy tính cá nhân:** Xử lý văn bản, đồ họa máy tính…
- **Phần mềm trên Web**
- **Phần mềm trí tuệ nhân tạo:** Dựa trên những kỹ thuật của Trí tuệ nhân tạo như hệ chuyên gia.

Nhận xét: Hiện nay web được xem là môi trường phổ biến để xây dựng giao diện với người sử dụng cho nhiều hệ thống phần mềm trên mạng. 8

1.1.2. Phát triển phần mềm 9

Khía cạnh lịch sử 10

- Năm 1967 nhóm NATO đưa ra thuật ngữ Công nghệ phần mềm (Software Engineering). 11
- Năm 1968 Hội nghị Software Engineering ở Garmisch, Đức đưa ra mục đích là giải quyết "Cuộc khủng hoảng phần mềm": 12
    - Phần mềm hoàn thành không đúng thời hạn 13
    - Chi phí vượt dự toán ban đầu
    - Phần mềm còn nhiều lỗi

Tại sao không thể sử dụng kỹ nghệ xây cất như xây dựng cầu để xây dựng các hệ điều hành? 1

- Thái độ đối với việc sập cầu/hệ điều hành 2
- Thông tin về CNPM thường không đầy đủ, không chắc chắn
- Độ phức tạp
- Bảo trì

Công nghệ phần mềm không thể xem giống như các kỹ nghệ thông thường khác. 3

Khía cạnh kinh tế 4

- CNPM và khoa học máy tính (tương tự như hóa học và kỹ nghệ hóa) 5
  - Khoa học có phần thực nghiệm và lý thuyết: Phần thực nghiệm của Hóa học là thí nghiệm còn của khoa học máy tính là lập trình. 6
  - Khoa học máy tính nghiên cứu những sách khác nhau để tạo ra phần mềm. Nhưng kỹ sư phần mềm chỉ quan tâm kỹ thuật có ý nghĩa kinh tế.
  - Ví dụ: Phương pháp mã hóa mới KTmới (lập trình hướng thành phần) nhanh hơn phương pháp đang sử dụng hiện thời KTcũ (lập trình hướng đối tượng) là 10%. Chúng ta nên sử dụng phương pháp mới hay không?
  - Câu trả lời thông thường là: Tất nhiên! Câu trả lời Công nghệ phần mềm: xét hiệu quả của KTmới.

Khía cạnh bảo trì 7

- Vòng đời phần mềm: Một loạt các pha mà phần mềm phải trải qua từ khám phá các khái niệm đến loại bỏ hoàn toàn. 8
- Mô hình vòng đời 9
  - Pha yêu cầu 10
  - Pha đặc tả
  - Pha thiết kế
  - Pha cài đặt
  - Pha tích hợp
  - Pha bảo trì
  - Loại bỏ
- **Bảo trì:** Mọi thay đổi đối với sản phẩm một khách hàng đã đồng ý sản phẩm thỏa mãn tài liệu đặc tả. 11
- Phần mềm tốt được bảo trì khác với phần mềm tồi bị loại bỏ 12
- Các dạng bảo trì" 13
  - Bảo trì sửa lỗi [17,5%]: sửa chữa lỗi nhưng đặc tả không đổi 14
  - Bảo trì nâng cao: sửa chữa theo thay đổi của đặc tả. Bảo trì hoàn thiện [60,5%]: thêm chức năng để cải tiến sản phẩm. Bảo trì thích nghi [18%]: thay đổi phần

mềm để đáp ứng thay đổi của môi trường như quy định chính phủ, CPU, công nghệ mới… 1

Kết luận: Bảo trì là pha tốn kém nhiều thời gian và chi phí 2

Khía cạnh phân tích và thiết kế 3

- 60 đến 70% lỗi của phần mềm là những lỗi do đặc tả và thiết kế 4
- Dữ liệu của Kelly, Sherif and Hops [1992] về chương trình không gian liên hành tinh của Nasa
- 1,9 lỗi trên một trang đặc tả
- 0,9 lỗi trên một trang thiết kế
- 0,3 lỗi trên một trang chương trình nguồn
- Dữ liệu của Bhandari et al. [1994]: Lỗi trong cuối pha thiết kế của phiên bản mới của sản phẩm
    - 13% lỗi từ phiên bản trước 5
    - 16% lỗi trong pha đặc tả mới
    - 71% lỗi trong pha thiết kế mới

Kết luận: Xây dựng những kỹ thuật sinh ra thiết kế và đặc tả tốt hơn là một vấn đề quan trọng trong công nghệ phần mềm. 6

Khía cạnh nhóm phát triển 7

- Phần cứng rẻ, khả năng tăng, kích thước giảm 8
- Phần mềm lớn, đắt. Nhiều phần mềm quá lớn nên một người không thể phát triển được trong thời gian có hạn.

1.2 VÒNG ĐỜI PHÁT TRIỂN PHẦN MỀM 9

1.2.1 Quy trình phát triển phần mềm hướng đối tượng 10

- Tiến trình phần mềm là "phương cách" sản xuất ra phần mềm. Tiến trình phần mềm nghiên cứu các cách kết hợp: 11
    - Mô hình vòng đời 12
    - Các công cụ CASE
    - Các cá nhân xây dựng phần mềm
    - Các công nghệ

Tiến trình phần mềm = Khía cạnh kỹ thuật + Khía cạnh quản lý 13

- Các tổ chức khác nhau có những tiến trình phần mềm khác nhau. 14
- Khâu viết tài liệu (quan trọng/không quan trọng)
- Chi phí dành cho kiểm thử (1/2 chi phí/không quan trọng)
- Tập trung khaua nghiên cứu, phát triển phần mềm. Khâu bảo trì dành cho người khác.

- Vì sao như vậy? 1
- Thiếu kỹ năng về kỹ nghệ phần mềm
- Nhiều người quản lý phần mềm thiếu kiến thức về bảo trì và phát triển phần mềm (do đó trễ hạn!)
- Quan điểm về quản lý (giao đúng hạn hay kiểm tra kỹ trước khi giao)
- Phụ thuộc vào cá nhân

Có pha kiểm thử không? 2

- KHÔNG có pha kiểm thử. Vì sao? Kiểm thử là hoạt động thực hiện trong mọi pha của sản xuất phần mềm. 3
- Check = test
- Testing: Thương được hiểu sau khi coding
- Kiểm tra (Varification): Thực hiện vào cuối mỗi pha
- Kiểm chứng (Validation): Thực hiện trước khi giao sản phẩm cho khách hàng

Có pha viết tài liệu không? 4

- KHÔNG có pha viết tài liệu 5
- Mọi pha phải được viết tài liệu trước khi khởi đầu một pha mới.
- Một số lý do: 6
  - Tài liệu bị hoãn lại thì sẽ không bao giờ hoàn thành 7
  - Cá nhân chịu trách nhiệm trong pha trước có thể đã chuyển sang bộ phận khác.
  - Sản phẩm thường xuyên thay đổi trong khi phát triển vì thế ta cần tài liệu để ghi lại điều này. Ví dụ, thiết kế thường phải sửa đổi trong khi cài đặt. Việc sửa đổi như vậy chỉ có thể thực hiện được khi có tài liệu của nhóm thiết kế.

  Kiểm thử và viết tài liệu được tiến hành trong mọi pha của tiến trình phần mềm. 8

SQA 9

- Nhóm đảm bảo chất lượng phần mềm (SQA: Software quality assurance): 10
  - Có trách nhiệm đảm bảo sản phẩm được xây dựng đúng (theo đặc tả) và theo đơn đặt hàng.
  - Nhóm SQA phải đóng đúng vai trò ngay từ đầu của tiến trình và hoạt động trong mọi pha của tiến trình phần mềm.
  - SQA kiểm tra với khách hàng xem phiên bản cuối cùng thỏa mãn hoàn toàn chưa.

Pha yêu cầu 11

- Tiến trình phát triển phần mềm bắt đầu khi khách hàng tiếp xúc với công ty phần mềm và cho rằng: Phần mềm có khả năng thích hợp với khách hàng và giá thành hợp lý. 12
- Khám phá khái niệm: 13

1. Trong lần gặp đầu tiên, khách hàng phát họa sản phẩm mà họ hình dung. Theo quan điểm của người phát triển, mô tả này không rõ ràng, không hợp lý, mâu thuẫn hay không thể xây dựng phần mềm như thế được. 1
2. Người phát triển xác định nhu cầu và ràng buộc của khách hàng 2

Kiểm thử pha yêu cầu:

- Làm bản mẫu nhanh: là mẫu phần mềm kết hợp nhiều chức năng của sản phẩm cuối cùng nhưng bỏ qua những khía cạnh mà khách hàng không thấy được như cập nhật file hay xử lý lỗi. 3
- Bản mẫu nhanh phải được kiểm tra bởi khách hàng và người sử dụng.
- Bản mẫu nhanh có thể thay đổi cho đến khi khách hàng và người sử dụng cho rằng nó có những chức năng mà họ mong muốn.

Viết tài liệu pha yêu cầu: 4

- Tài liệu pha yêu cầu: 5
    - Có bản mẫu: Bản mẫu nhanh. Bản ghi thỏa thuận với khách hàng và người sử dụng về cơ sở xây dựng và sửa đổi bản mẫu.
    - Không có bản mẫu (nhóm quyết định không xây dựng bản mẫu): Mô tả nhu cầu khách hàng. Tài liệu này phải được khách hàng, người sử dụng và nhóm phát triển kiểm tra trước khi nhóm SQA xem xét.

Pha đặc tả/phân tích 6

- Khi khách hàng cho rằng nhóm phát triển hiểu được yêu cầu, nhóm đặc tả viết tài liệu đặc tả để mô tả chức năng của sản phẩm (những gì sản phẩm cần có + ràng buộc). 7
- Đặc tả bao gồm những input của sản phẩm và output được yêu cầu
- Ví dụ: Khách hàng cần tính bảng lương thì input bao gồm mức trả cho mỗi nhân viên, thông tin từ hồ sơ cá nhân để tính thuế… và output là số lương thuế, chi bảo hiểm,…
- Yêu cầu của đặc tả
- Không nhập nhằng: không sử dụng thuật ngữ như tiện lợi, đầy đủ chức năng, nhanh, 98%...
- Đầy đủ: Thể hiện mọi yêu cầu của khách hàng
- Phi mâu thuẫn: không chứa mâu thuẫn
- Theo dõi được (Traceability): có thể lần theo phán đoán trong đặc tả trở lại phán đoán đưa ra bởi khách hàng trong pha yêu cầu. Nếu đặc tả được trình bày đúng phương pháp, có đánh chỉ số,… thì nhóm SQA sẽ ít gặp khó khăn. Nếu có bản mẫu thì phán đoán liên quan của đặc tả phải theo dõi được đến bản mẫu.
- Môt khi đặc tả được hoàn thành và đã thông qua thì hình thành kế hoạch quản lý quá trình sản xuất phần mềm (SPMP : The software product management plan).
- Yêu cầu kế hoạch:

- SPMP cần nêu lên thời gian thực hiện, chi phí cho từng pha, gán trách nhiệm cá nhân cho từng pha, thời hạn hoàn thành cho mỗi pha. 1
- Mô hình vòng đời nào sẽ sử dụng, cấu trúc tổ chức, kỹ thuật và CASE sử dụng, lịch tình, chi phí…

Kiểm thử pha đặc tả: 2

- Nguồn gốc chính của lỗi trong các phần mềm đã phân phối đến nay là những lỗi trong tài liệu đặc tả và những lỗi này chỉ được phát hiện khi tổ chức khách hàng sử dụng nó. 3
- Duyệt xét lại (Review): là cách tốt nhất để kiểm tra đặc tả. Mục đích là xác định đặc tả có đùng không. Nhóm SQA chủ trì cuộc họp với đại diện nhóm đặc tả và khách hàng.

Pha thiết kế 4

- Đặc tả - What 5
- Thiết kế - How
- Giữ lại quyết định thiết kế
    - Thời điểm kết thúc
    - Cho nhóm bảo trì
    - Thiết kế nên mở (open-ended)
- Thiết kế kiến trúc : Phân rã sản phẩm thành mô đun
- Thiết kế chi tiết: Thiết kế các mô đun: cấu trúc dữ liệu, thuật toán

Kiểm thử pha thiết kế: 6

- Tài liệu viết sao cho dễ theo dõi 7
- Duyệt tài liệu

Pha cài đặt 8

- Cài đặt thiết kế chi tiết thành chương trình 9

Kiểm thử pha cài đặt:

- Rà soát 10
- Các test case
    - Test không hình thức (desk checking)
    - Test hình thức (Formal testing) do nhóm SQA

Tích hợp 11

- Kết hợp các mô đun và kiểm thử toàn bộ sản phẩm 12

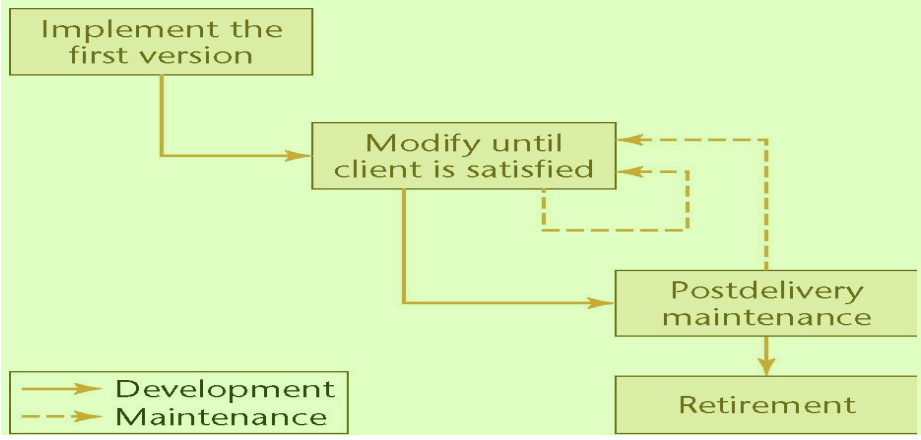
Tài liệu pha tích hợp: 13

- Mã nguồn có chú thích 14
- Các test cases

1.2.2 Một số mô hình vòng đời phát triển phần mềm 1

a. Mô hình xây sửa (code and fixe) 2

- Không thiết kế 3
- Không đặc tả
  - Bảo trì là ác mộng
- Là cách dễ dàng nhất để phát triển phần mềm
- Là cách đắt nhất

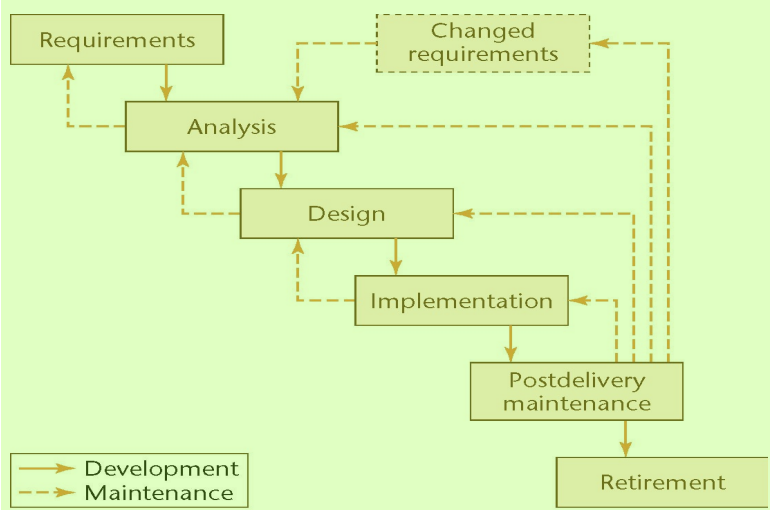


4

Hình 1.1: Mô hình xây và sửa 5

b. Mô hình thác nước (waterfall) 6

- Đặc trưng bởi 7
  - Các vòng lặp phản hồi
  - Hướng tài liệu
  - Mô hình thác nước không biểu hiện thứ tự các sự kiện
- Thuận lợi
  - Tài liệu
  - Bảo trì dễ dàng
- Bất lợi
  - Tài liệu đặc tả
    - Joe và Jane Johnson
    - Mark Marberry

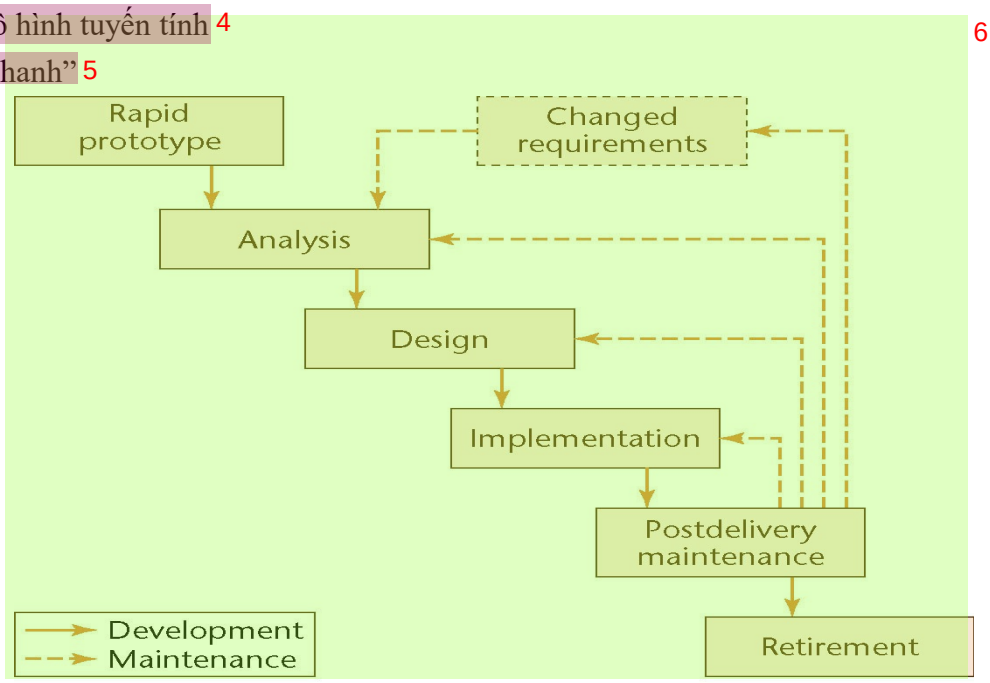


1

Hình 1.2: Mô hình thác nước 2

c. *Mô hình bản mẫu nhanh (Rapid prototype)* 3

- Mô hình tuyến tính 4
- "Nhanh" 5



6

Hình 1.3: Mô hình bản mẫu nhanh 7

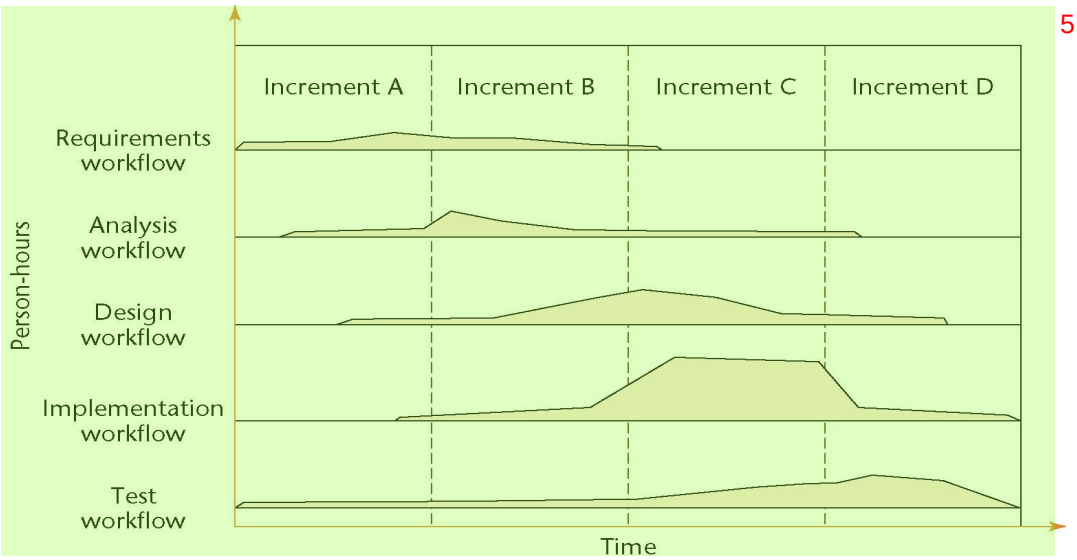
d. *Mô hình lặp và tăng trưởng (Iteration and Increasement)* 8

- Trong thực tế, chúng ta không thể nói về "pha phân tích" 9
  - Mặc dù, các thao tác của pha phân tích trải rộng xuyên suốt vòng đời phần mềm.
- Tiến trình phát triển phần mềm là lặp
  - Mỗi phiên bản kế tiếp tiến gần với hệ thống đích hơn phiên bản trước đó.

Luật Miller: 10

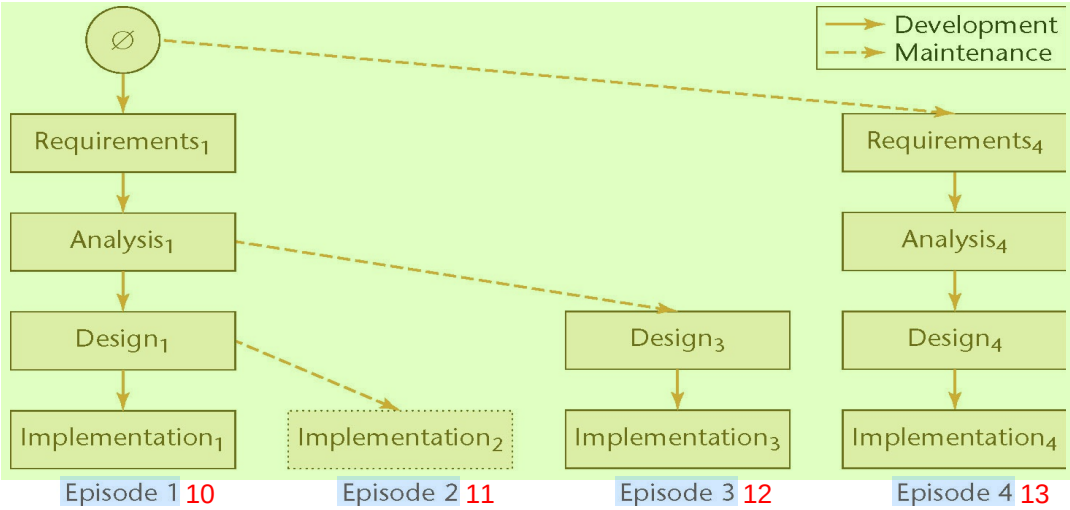
- Ở bất cứ thời điểm nào, chúng ta chỉ có thể tập trung vào khoảng 7 chunks (tương ứng 7đơn vị thông tin) 1
- Để xử lý lượng thông tin lớn hơn yêu cầu sử dụng bước làm mịn theo kiểu bậc thang 2
  - Tập trung vào các khía cạnh quan trọng nhất hiện thời 3
  - Các khía cạnh trì hoãn thường ít quan trọng hơn
  - Cuối cùng mọi khía cạnh đều được xử lý nhưng theo thứ tự mức độ quan trọng hiện thời

Đây là tiến trình tăng 4



Hình 1.4: Các bước tăng trưởng 6

- Lặp và tăng được sử dụng chung với các mô hình khác. 7
  - Không có Episode nào mà chỉ có pha "xác định yêu cầu" hoặc "pha thiết kế" 8
  - Có nhiều thể hiện ở mỗi pha



Hình 1.5: Các bước lặp và tăng trưởng 14

- Số lượng của sự gia tăng sẽ thay đổi – không phải là 4 15
- Và số lượng vòng lặp cũng thay đổi không phải luôn bằng ba.

Các pha cổ điển với các workflow 1

- Các pha tuần tự không có trong thế giới thực 2
- Mặc dù có 5 workflow (luồng công việc) chính được thực hiện ở trên toàn vòng đời phát triển phần mềm 3
  - Luồng công việc xác định yêu cầu - Requirements workflow 4
  - Luồng công việc phân tích - Analysis workflow
  - Luồng công việc thiết kế - Design workflow
  - Luồng công việc cài đặt - Implementation workflow
  - Luồng công việc kiểm thử - Test workflow

Các luồng công việc 5

- Cả năm luồng công việc chính được thực hiện trên toàn bộ vòng đời phần mềm 6
- Tuy nhiên, ở mỗi một thời điểm có một luồng công việc chiếm ưu thế.
- Chẳng hạn: 7
  - Ở đầu mỗi vòng đời phát triển phần mềm 8
    - Luồng công việc đặc tả yêu cầu chiếm ưu thế
  - Ở cuối mỗi vòng đời phát triển phần mềm
    - Luồng công việc cài đặt và kiểm thử chiếm ưu thế
- Các hoạt động lập kế hoạch và viết tài liệu được thực hiện xuyết suốt vòng đời phát triển phần mềm 9

Xem xét lại bài toán Winburg Mini 10

- Mô hình cây tiến hóa đã được thêm vào mô hình vòng đời lặp và tăng 11
- Luồng kiểm thử đã bị bỏ quên – mô hình cây tiến hóa giả sử rằng kiểm thử liên tục 12
- Đối với quá trình tăng: 13
  - Mỗi Episode tương ứng với một sự gia tăng 14
  - Không phải mỗi sự gia tăng đều bao gồm mọi luông công việc
  - Sự gia tăng B không được hoàn thiện
  - Những đường nét đứt biểu diễn sự bảo trì
    - Episodes 2, 3: Bảo trì sửa lỗi
    - Episode 4: Bảo trì hoàn thiện chức năng

Rủi ro và những khía cạnh khác của mô hình lặp và tăng 15

- Chúng ta có thể xem xét toàn bộ dự án như là một tập các dự án nhỏ (quá trình tăng) 16
- Mỗi dự án nhỏ gồm
  - Tài liệu yêu cầu 17
  - Tài liệu phân tích
  - Tài liệu thiết kế
  - Tài liệu cài đặt

- Tài liệu kiểm thử 1
- Tập tài liệu cuối cùng là sản phẩm phần mềm hoàn thiện 2
- Trong suốt dự án nhỏ:During each mini project we 3
  - Mở rộng các tài liệu (Sự gia tăng); 4
  - Kiểm tra tài liệu (luồng công việc kiểm thử); và
  - Nếu cần, thay đổi các tài liệu liên quan (vòng lặp)
- Mỗi vòng lặp có thể được xem xét như là một mô hình vòng đời thác nước nhỏ nhưng hoàn thiện 5
- Trong suốt mỗi vòng lặp chúng ta lựa chọn một phần của hệ thống phần mềm 6
- Trong mỗi phần đó cần thực hiện: 7
  - Pha xác định yêu cầu cổ điển 8
  - Pha phân tích cổ điển
  - Pha thiết kế cổ điển
  - Pha cài đặt cổ điển

Điểm mạnh của mô hình vòng đời lặp và tăng 9

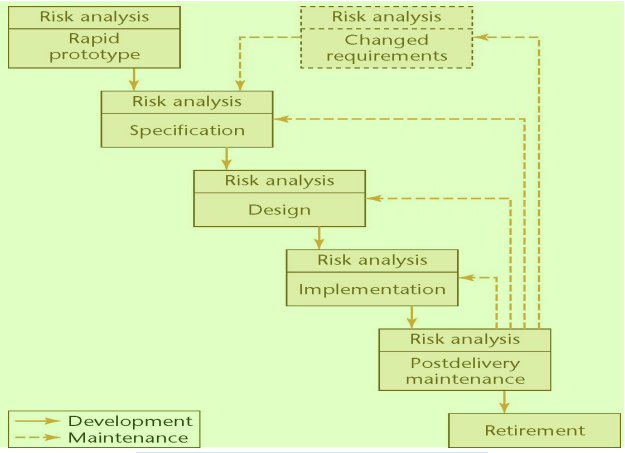
- Có nhiều sự tối ưu cho việc kiểm tra tính đúng đắn của sản phẩm 10
  - Mỗi vòng lặp có tích hợp luồng công việc kiểm thử
  - Các lỗi có thể được phát hiện và sửa chữa sớm
- Sự mạnh mẽ của kiến trúc có thể được xác định sớm trong vòng đời 11
  - Kiến trúc – các mô đun thành phần khác nhau và cách chúng kết hợp với nhau 12
  - *Sự mạnh mẽ - có khả năng xử lý việc mở rộng và thay đổi mà hệ thống không bị tách thành từng mảnh*
- Chúng ta có thể giảm bớt rủi ro sớm hơn 13
  - Lúc nào cũng vậy rủi ro luôn liên quan tới quá trình phát triển phần mềm và bảo trì 14
- Chúng ta có một phiên bản hệ thống phần mềm đang làm việc 15
  - Khách hàng và người dùng có thể thử nghiệm với phiên bản này để xác định cái họ cần thay đổi 16
  - Mức độ thay đổi: các phiên bản từng phần đưa ra để làm cho sự giới thiệu sản phẩm mới tới tổ chức khách hàng dễ dàng hơn.
- Có bằng chứng thực nghiệm chứng tỏ mô hình vòng đời làm việc 17
- Các bản tường trình CHAOS của nhóm Standish đã chỉ ra phần trăm phần mềm thành công tăng 18
- Lý do của sự suy giảm của các dự án thành công năm 2004 bao gồm: 19
  - Nhiều dự án lớn trong năm 2004 hơn năm 2002 20
  - Sử dụng mô hình thác nước
  - Thiếu sự quan tâm của người dùng
  - Thiếu sự hỗ trợ từ những người điều hành lâu năm

Việc quản lý lặp và tăng 21

- Mô hình vòng đời lặp và tăng là tập hợp các mô hình thách nước… 1
- … bởi vì mô hình vòng đời lặp và tăng là mô hình thác nước, được áp dụng liên tiếp
- Mỗi sự gia tăng là mộ dự án nhỏ theo mô hình vòng đời thác nước

e. Mô hình xoắn ốc (Spiral) 2

Là mô hình bản mẫu nhanh kết hợp với phân tích rủi ro đi kèm ở mỗi pha 3
Đặc điểm chính của mô hình xoán ốc: Nếu các rủi ro không được làm giảm bớt, thì dự án bị kết thúc ngay lập tức.

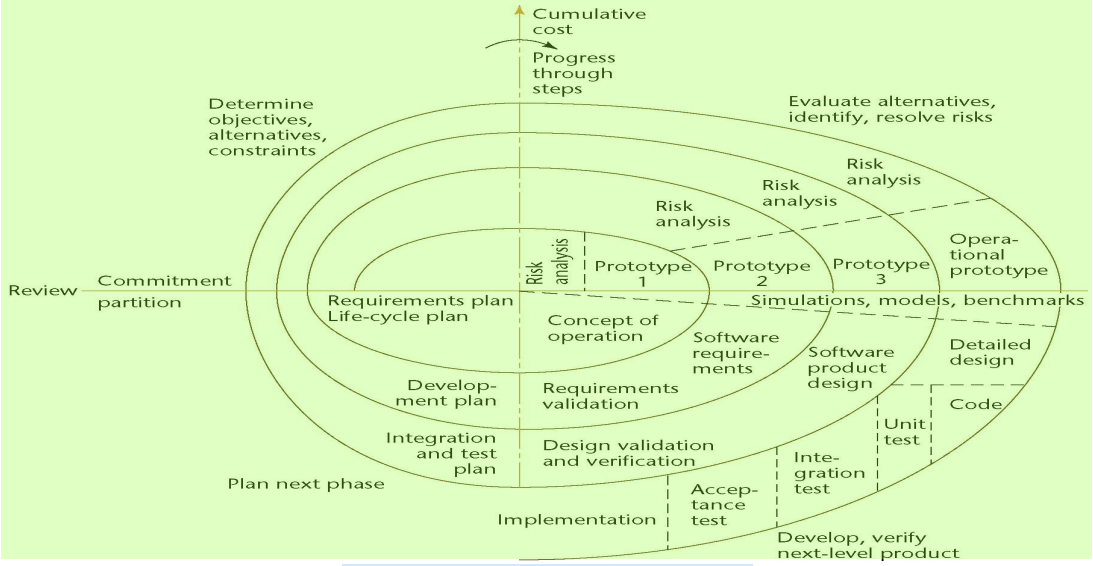


4

Hình 1.6: Mô hình xoắn ốc 5

Mô hình xoắn ốc đầy đủ: 6

- Ở trước mỗi pha là các giải pháp và phân tích rủi ro 7
- Theo sau mỗi pha là: Đánh giá và lập kế hoạch cho pha tiếp theo
- Chiều bán kính: Chi phí tích lũy cho đến hiện tại
- Chiều góc: sự đi lên theo vòng ốc



8

Hình 1.7: Các bước lặp xoắn ốc 9

- Điểm mạnh của mô hình xoắn ốc: 10

- Dễ dàng thay đổi mức độ kiểm thử 1
- Không phân biệt giữa phát triển và bảo trì
- Điểm yếu của mô hình xoắn ốc:
- Chỉ phù hợp với những phần mềm lớn
- Chỉ phù hợp với những phần mềm nội bộ

f. Mô hình mã nguồn mở (Open source) 2

- Hai pha không hình thức 3
- Đầu tiên, một các nhân xây dựng một phiên bản đầu tiên
  - Đưa phiên bản này lên Internet (ví dụ: SourceForge.net)
- Sau đó, nếu có sự quan tâm về dự án này 4
  - Phiên bản đầu tiên sẽ được tải về một cách rộng rãi
  - Người dùng trở thành người đồng phát triển
  - Sản phẩm được mở rộng
- Điểm chính: Các cá nhân nhìn chung làm việc tình nguyện đối với dự án mã nguồn mở trong thời gian rảnh rỗi của họ 5
- Các hoạt động trong pha phi hình thức thứ hai 6
  - Báo cáo và sửa lỗi 7
  - Bảo trì sửa lỗi
  - Thêm các chức năng mới
  - Bảo trì hoàn thiện chức năng
  - Điều chỉnh phần mềm ở môi trường mới
  - Bảo trì thích hợp
  - Pha phi hình thức thứ hai chỉ bao gồm bảo trì sau khi đã chuyển giao sản phẩm
  - Từ "người đồng phát triển" trong slide trước có thể thay thế bằng từ "đồng bảo trì"
- Mô hình vòng đời bảo trì sau khi chuyển giao sản phẩm 8

9

Hình 1.8: Mô hình mã nguồn mở 10

- Sản phẩm mã nguồn đóng được bảo trì và kiểm thử bởi nhân viên 11
  - Người dùng có thể đưa ra bản tường trình sự thất bại của phần mềm (failure) nhưng không bao giờ đưa ra được lỗi mã nguồn (fault) (vì mã nguồn không sẵn có) 12

- Phần mềm mã nguồn mở nhìn chung được bảo trì bởi người tình nguyện viên không lương 1
  - Người dùng được khuyến khích đưa ra bản tường trình về sự thất bại của phần mềm (failure) cũng như lỗi mã nguồn (fault) 2
- Nhóm chính 3
  - Số lượng nhỏ những người bảo trì tận tụy với sở thích, thời gian và những kỹ năng cần thiết để đưa ra những báo cáo lỗi mã nguồn đã được cố định("fixes") 4
  - Họ chịu trách nhiệm quản lý dữ án
  - Họ có quyền cài đặt lại những lỗi đã cố định
- Nhóm ngoại vi 5
  - Users who choose to submit defect reports from time to time
- Các phiên bản mới của phần mềm mã nguồn đóng được phát hành thường xuyên một năm một lần 6
  - Sau khi đã được kiểm thử cẩn thận bởi nhóm quản lý chất lượng phần mềm
- Các phát hành của nhóm chính của một phiên bản mới của sản phẩm mã nguồn mở được đưa ra ngay khi sẵn sàng 7
  - Có thể là một tháng hoặc thậm chí là một ngày so với với phiên bản trước đó 8
  - Nhóm chính thực hiện kiểm thử tối thiểu
  - Kiểm thử mở rộng được thực hiện bởi các thành viên của nhóm ngoại vi trong suốt thời gian sử dụng phần mềm
  - "Phát hành sớm và thường xuyên"
- Phiên bản làm việc đầu tiên được đưa ra khi sử dụng: 9
  - Mô hình bản mẫu nhanh; 10
  - Mô hình xây và sửa; và
  - Mô hình mã nguồn mở
- Sau đó: 11
  - Mô hình bản mẫu nhanh 12
    - Phiên bản đầu tiên bị bỏ qua
  - Mô hình xây và sửa và mô hình vòng đời mã nguồn mở
    - Phiên bản ban đầu trở thành phiên bản đích
- Do đó, trong dự án mã nguồn mở, nhình chung không có đặc tả và không có thiết kế 13
- Một số dự án mã nguồn mở đã thành công như thế nào khi không có đặc tả hoặc thiết kế? 14
- Sản phẩm mã nguồn mở đã thu hút được một số chuyên gia phần mềm tốt nhất trên thế giới 15
  - Họ có thể xây dựng các chức năng hiệu quả mà không cần đặc tả hoặc thiết kế
- Tuy nhiên, cuối cùng thì sự phát triển sản phẩm mã nguồn mở cũng dừng lại khi mà nó không có sự bảo trì nữa 16
- Mô hình vòng đời mã nguồn mở bị hạn chế trong khả năng ứng dụng của nó
- Nó có thể thành công cực độ đối với những dự án cơ sở hạ tầng như 17

- Hệ điều hành (Linux, OpenBSD, Mach, Darwin) 1
- Trình duyệt Web (Firefox, Netscape)
- Trình biên dịch (gcc)
- Máy chủ Web (Apache)
- Hệ thống quản lý dữ liệu (MySQL)

- Không thể phát triển mã nguồn mở một sản phẩm phần mềm mà chỉ được sử dụng trong tổ chức thương mại 2
  - Các thành viên của cả nhóm chính và nhóm ngoại vi đều luôn luôn là những người dùng của phần mềm đang được phát triển
- Mô hình vòng đời mã nguồn mở không không thể áp dụng được trừ khi sản phẩm đích được xem xét bởi một số lượng lớn người dùng vì sản phẩm đó có ích cho họ.
- Khoảng một nửa dự án mã nguồn mở trên Web không thu hút được các nhóm phát triển vào làm việc cho dự án
- Even where work has started, the overwhelming preponderance will never be completed
- Nhưng khi mô hình mã nguồn mở đã làm việc thì đối khi nó cũng đem lại thành công đáng kinh ngạc
  - Những sản phẩm mã nguồn mở được kể ở slide trước đã đước sử dụng bởi hàng triệu người dùng

1.3 UML - CÔNG CỤ HỖ TRỢ PHÂN TÍCH THIẾT KẾ HƯỚNG ĐỐI TƯỢNG 3

1.3.1 Lịch sử ra đời của UML 4

Việc áp dụng rộng rãi phương pháp hướng đối tượng đã đặt ra yêu cầu cần phải xây dựng một phương pháp mô hình hóa để có thể sử dụng như một chuẩn chung cho những người phát triển phần mềm hướng đối tượng trên khắp thế giới. Trong khi các ngôn ngữ hướng đối tượng ra đời khá sớm, ví dụ như Simula-67 (năm 1967), Smalltalk (đầu những năm 1980), C++, CLOS (giữa những năm 1980)…thì những phương pháp luận cho phát triển hướng đối tượng lại ra đời khá muộn. Cuối những năm 80, đầu những năm 1990, một loạt các phương pháp luận và ngôn ngữ mô hình hóa hướng đối tượng mới ra đời, như Booch của Grady Booch, OMT của James Rambaugh, OOSE của Ivar Jacobson, hay OOA and OOD của Coad và Yordon. 5

Mỗi phương pháp luận và ngôn ngữ trên đều có hệ thống ký hiệu riêng, phương pháp xử lý riêng và công cụ hỗ trợ riêng. Chính điều này đã thúc đẩy những người tiên phong trong lĩnh vực mô hình hoá hướng đối tượng ngồi lại cùng nhau để tích hợp những điểm mạnh của mỗi phương pháp và đưa ra một mô hình thống nhất chung. Nỗ lực thống nhất đầu tiên bắt đầu khi Rumbaugh gia nhập nhóm nghiên cứu của Booch tại tập đoàn Rational năm 1994 và sau đó Jacobson cũng gia nhập nhóm này vào năm 1995. 6

James Rumbaugh, Grady Booch và Ivar Jacobson đã cùng cố gắng xây dựng được một Ngôn Ngữ Mô Hình Hoá Thống Nhất và đặt tên là UML (Unifield Modeling Language). UML 7

đầu tiên được đưa ra năm 1997 và sau đó được chuẩn hoá để trở thành phiên bản 1.0. Chương này trình bày ngôn ngữ UML phiên bản 2.0. 1

1.3.2 UML – Ngôn ngữ mô hình hoá hướng đối tượng 2

UML (Unified Modelling Language) là ngôn ngữ mô hình hoá tổng quát được xây dựng để đặc tả, phát triển và viết tài liệu cho các khía cạnh trong phát triển phần mềm hướng đối tượng. UML giúp người phát triển hiểu rõ và ra quyết định liên quan đến phần mềm cần xây dựng. UML bao gồm một tập các khái niệm, các ký hiệu, các biểu đồ và hướng dẫn. UML hỗ trợ xây dựng hệ thống hướng đối tượng dựa trên việc nắm bắt khía cạnh cấu trúc tĩnh và các hành vi động của hệ thống. 3

- Các cấu trúc tĩnh định nghĩa các kiểu đối tượng quan trọng của hệ thống, nhằm cài đặt và chỉ ra mối quan hệ giữa các đối tượng đó. 4
- Các hành vi động (dynamic behavior) định nghĩa các hoạt động của các đối tượng theo thời gian và tương tác giữa các đối tượng hướng tới đích.

Các mục đích của ngôn ngữ mô hình hoá thống nhất UML: 5

- Mô hình hoá các hệ thống sử dụng các khái niệm hướng đối tượng. 6
- Thiết lập sự liên hệ từ nhận thức của con người đến các sự kiện cần mô hình hoá.
- Giải quyết vấn đề về mức độ thừa kế trong các hệ thống phức tạp với nhiều ràng buộc khác nhau.
- Tạo một ngôn ngữ mô hình hoá có thể sử dụng được bởi người và máy.

UML quy định một loạt các ký hiệu và quy tắc để mô hình hoá các pha trong quá trình phát triển phần mềm hướng đối tượng dưới dạng các biểu đồ. 7

1.3.3 Các khái niệm cơ bản trong UML 8

- **Khái niệm mô hình** 9

Mô hình là một biểu diễn của sự vật hay một tập các sự vật trong một lĩnh vực áp dụng nào đó theo một cách khác. Mô hình nhằm nắm bắt các khía cạnh quan trọng của sự vật, bỏ qua các khía cạnh không quan trọng và biểu diễn theo một tập ký hiệu và quy tắc nào đó. Các mô hình thường được xây dựng sao cho có thể vẽ được thành các biểu đồ dựa trên tập ký hiệu và quy tắc đã cho. 10

Khi xây dựng các hệ thống, mô hình được sử dụng nhằm thoả mãn các mục đích sau: 11

- Nắm bắt chính xác yêu cầu và tri thức miền mà hệ thống cần phát triển 12
- Thể hiện tư duy về thiết kế hệ thống
- Trợ giúp ra quyết định thiết kế dựa trên việc phân tích yêu cầu
- Tổ chức, tìm kiếm, lọc, kiểm tra và sửa đổi thông tin về các hệ thống lớn.

- Làm chủ được các hệ thống phức tạp 1

Các thành phần trong một mô hình bao gồm: 2

- Ngữ nghĩa và biểu diễn: Ngữ nghĩa là nhằm đưa ra ý nghĩa, bản chất và các tính chất của tập các ký hiệu. Biểu diễn là phương pháp thể hiện mô hình theo cách sao cho có thể nhìn thấy được. 3
- Ngữ cảnh: mô tả tổ chức bên trong, cách sử dụng mô hình trong tiến trình phần mềm …

- **Các hướng nhìn (View) trong UML** 4

Các mô hình trong UML nhằm mục đích hỗ trợ phát triển các hệ thống phần mềm hướng đối tượng. Trong phương pháp luận hướng đối tượng không có sự phân biệt rạch ròi giữa các pha hay các bước. Tuy nhiên, thông thường UML vẫn được chia thành một số hướng nhìn và nhiều loại biểu đồ. 5

Một hướng nhìn trong UML là một tập con các biểu đồ UML được xây dựng để biểu diễn một khía cạnh nào đó của hệ thống. 6

Sự phân biệt giữa các hướng nhìn là rất linh hoạt. Có thể có những biểu đồ UML có mặt trong cả hai hướng nhìn. Các hướng nhìn cùng các biểu đồ tương ứng được mô tả trong bảng sau: 7

8

Khía cạnh chính	***Hướng nhìn***	***Các biểu đồ***	***Các khái niệm chính***
Khía cạnh cấu trúc hệ thống	Hướng nhìn tĩnh (static view)	Biểu đồ lớp	lớp, liên hệ, kế thừa, phụ thuộc, giao diện
	Hướng nhìn use case (Use case view)	Biểu đồ use case	Use case, tác nhân, liên hệ, extend, include …
	Hướng nhìn cài đặt (implementation view)	Biểu đồ thành phần	Thành phần, giao diện, quan hệ phụ thuộc …
	Hướng nhìn triển khai (deployment view)	Biểu đồ triển khai	Node, thành phần, quan hệ phụ thuộc, vị trí (location)
Khía cạnh động	Hướng nhìn máy trạng thái (state machine view)	Biểu đồ trạng thái	Trạng thái, sự kiện, chuyển tiếp, hành động
	Hướng nhìn hoạt động (activity view)	Biểu đồ động	Trạng thái, sự kiện, chuyển tiếp, kết hợp, đồng bộ …
	Hướng nhìn tương tác (interaction view)	Biểu đồ tuần tự	Tương tác, đối tượng, thông điệp, kích hoạt …

		Biểu đồ cộng tác	Cộng tác, vai trò cộng tác, thông điệp …
Khía cạnh quản lý mô hình	Hướng nhìn quản lý mô hình	Biểu đồ lớp	Gói, hệ thống con, mô hình
Khía cạnh khả năng mở rộng	Tất cả	Tất cả	Các ràng buộc, stereotype, …

Bảng 1.1: Các hướng nhìn trong UML

1.3.4 Các biểu đồ UML

Thành phần mô hình chính trong UML là các biểu đồ:

- ***Biểu đồ use case*** biểu diễn sơ đồ chức năng của hệ thống. Từ tập yêu cầu của hệ thống, biểu đồ use case sẽ phải chỉ ra hệ thống cần thực hiện điều gì để thoả mãn các yêu cầu của người dùng hệ thống đó. Đi kèm với biểu đồ use case là các kịch bản.
- ***Biểu đồ lớp*** chỉ ra các lớp đối tượng trong hệ thống, các thuộc tính và phương thức của từng lớp và các mối quan hệ giữa những lớp đó.
- ***Biểu đồ trạng thái*** tương ứng với mỗi lớp sẽ chỉ ra các trạng thái mà đối tượng của lớp đó có thể có và sự chuyển tiếp giữa những trạng thái đó.
- ***Các biểu đồ tương tác*** biểu diễn mối liên hệ giữa các đối tượng trong hệ thống và giữa các đối tượng với các tác nhân bên ngoài. Có hai loại biểu đồ tương tác:
  - ***Biểu đồ tuần tự:*** Biểu diễn mối quan hệ giữa các đối tượng và giữa các đối tượng và tác nhân theo thứ tự thời gian.
  - ***Biểu đồ cộng tác:*** Biểu diễn mối quan hệ giữa các đối tượng và giữa các đối tượng và tác nhân nhưng nhấn mạnh đến vai trò của các đối tượng trong tương tác.
- ***Biểu đồ hoạt động*** biểu diễn các hoạt động và sự đồng bộ, chuyển tiếp các hoạt động, thường được sử dụng để biểu diễn các phương thức phức tạp của các lớp.
- ***Biểu đồ thành phần*** định nghĩa các thành phần của hệ thống và mối liên hệ giữa các thành phần đó.
- ***Biểu đồ triển khai hệ thống*** mô tả hệ thống sẽ được triển khai như thế nào, thành phần nào được cài đặt ở đâu, các liên kết vật lý hoặc giao thức truyền thông nào được sử dụng.

Dựa trên tính chất của các biểu đồ, UML chia các biểu đồ thành hai lớp mô hình[1]:

- **Biểu đồ mô hình cấu trúc (Structural Modeling Diagrams):** biểu diễn các cấu trúc tĩnh của hệ thống phần mềm được mô hình hoá. Các biểu đồ trong mô hình tĩnh tập trung biểu diễn khía cạnh tĩnh của hệ thống, liên quan đến cấu trúc cơ bản cũng như

[1] Tham khảo http://www.sparxsystems.com.au/resources/uml2_tutorial/

các phần tử chính trong miền quan tâm của bài toán. Các biểu đồ trong mô hình tĩnh bao gồm:

1. Biểu đồ gói
2. Biểu đồ đối tượng và lớp
3. Biểu đồ thành phần
4. Biểu đồ triển khai

- **Biểu đồ mô hình hành vi** (Behavioral Modeling Diagrams)**:** Nắm bắt đến các hoạt động và hành vi của hệ thống, cũng như tương tác giữa các phần tử bên trong và bên ngoài hệ thống. Các dạng biểu đồ trong mô hình động bao gồm:
  - Biểu đồ use case
  - Biểu đồ tương tác dạng tuần tự
  - Biểu đồ tương tác dạng cộng tác
  - Biểu đồ trạng thái
  - Biểu đồ động

Chúng ta sẽ lần lượt xem xét chi tiết các biểu đồ UML, mỗi biểu đồ sẽ được trình bày ý nghĩa của nó, tập kí hiệu UML cho biểu đồ đó và một ví dụ.

a. Biểu đồ use case

Ý nghĩa

Biểu đồ use case biểu diễn sơ đồ chức năng của hệ thống. Từ tập yêu cầu của hệ thống, biểu đồ use case sẽ phải chỉ ra hệ thống cần thực hiện điều gì để thoả mãn các yêu cầu của người dùng hệ thống đó. Đi kèm với biểu đồ use case là các kịch bản (scenario). Có thể nói, biểu đồ use case chỉ ra sự tương tác giữa các *tác nhân* và hệ thống thông qua các use case.

Mỗi *use case* mô tả một chức năng mà hệ thống cần phải có xét từ quan điểm người sử dụng. *Tác nhân* là con người hay hệ thống thực khác cung cấp thông tin hay tác động tới hệ thống.

Một *biểu đồ use case* là một tập hợp các tác nhân, các use case và các mối quan hệ giữa chúng. Các use case trong biểu đồ use case có thể được phân rã theo nhiều mức khác nhau.

Tập ký hiệu UML cho biểu đồ use case

Một biểu đồ Use Case chứa các phần tử mô hình biểu thị hệ thống, tác nhân cũng như các trường hợp sử dụng và các mối quan hệ giữa các Use Case. Chúng ta sẽ lần lượt xem xét các phần tử mô hình này:

- *Hệ thống*: Với vai trò là thành phần của biểu đồ use case, hệ thống biểu diễn ranh giới giữa bên trong và bên ngoài của một chủ thể trong phần mềm chúng ta đang xây dựng. Chú ý rằng một hệ thống ở trong biểu đồ use case không phải bao giờ cũng nhất thiết là

một hệ thống phần mềm; nó có thể là một chiếc máy, hoặc là một hệ thống thực (như một doanh nghiệp, một trường đại học, …).

- *Tác nhân (actor):* là người dùng của hệ thống, một tác nhân có thể là một người dùng thực hoặc các hệ thống máy tính khác có vai trò nào đó trong hoạt động của hệ thống. Như vậy, tác nhân thực hiện các use case. Một tác nhân có thể thực hiện nhiều use case và ngược lại một use case cũng có thể được thực hiện bởi nhiều tác nhân.
- *Các use case:* Đây là thành phần cơ bản của biểu đồ use case. Các use case được biểu diễn bởi các hình elip. Tên các use case thể hiện một chức năng xác định của hệ thống.
- *Mối quan hệ giữa các use case:* giữa các use case có thể có các mối quan hệ như sau:
- *Include:* use case này sử dụng lại chức năng của use case kia.
- *Extend:* use case này mở rộng từ use case kia bằng cách thêm vào một chức năng cụ thể.
- *Generalization:* use case này được kế thừa các chức năng từ use case kia.

Các phần tử mô hình use case cùng với ý nghĩa và cách biểu diễn của nó được tổng kết trong Bảng 1.2.

Phần tử mô hình	Ý nghĩa	Cách biểu diễn	Ký hiệu trong biểu đồ
Use case (Usecase name)	Biểu diễn một chức năng xác định của hệ thống	Hình ellip chứa tên của use case	
Tác nhân	Là một đối tượng bên ngoài hệ thống tương tác trực tiếp với các use case	Biểu diễn bởi một lớp kiểu actor (hình người tượng trưng)	Actor
Mối quan hệ giữa các use case (<<extend>>, <<include>>)	Tùy từng dạng quan hệ	Extend và include có dạng các mũi tên đứt nét Generalization có dạng mũi tên tam giác.	
Biên thống	Phân biệt phần bên trong và bên ngoài hệ thống	Được biểu diễn bởi một hình chữ nhật rỗng.	

Bảng 1.2: Các phần tử mô hình trong biểu đồ use case

b. Biểu đồ lớp 1
Ý nghĩa

Trong phương pháp hướng đối tượng, *một nhóm đối tượng có chung một số thuộc tính và phương thức tạo thành một lớp.* Mối tương tác giữa các đối tượng trong hệ thống sẽ được biểu diễn thông qua mối quan hệ giữa các lớp. 2

Các lớp (bao gồm cả các thuộc tính và phương thức) cùng với các mối quan hệ sẽ tạo thành biểu đồ lớp. Biểu đồ lớp là một biểu đồ dạng mô hình tĩnh nhằm mô tả hướng nhìn tĩnh về một hệ thống bằng các khái niệm lớp, các thuộc tính, phương thức của lớp và mối quan hệ giữa chúng với nhau. 3

Tập ký hiệu UML cho biểu đồ lớp 4

Trong phần này, tài liệu sẽ xem xét các vấn đề liên quan đến biểu diễn sơ đồ lớp trong UML. Cuối phần này sẽ là một bảng tổng kết các ký hiệu UML sử dụng trong sơ đồ lớp. 5

- ***Kí hiệu lớp:*** trong UML, mỗi lớp được biểu diễn bởi hình chữ nhật gồm 3 phần: tên lớp, các thuộc tính và các phương thức. 6
- ***Thuộc tính***: các thuộc tính trong biểu đồ lớp được biểu diễn theo cấu trúc chung như sau: 7

phạm_vi tên : kiểu số_đối_tượng = mặc_định (Giá_ trị_giới_hạn) 8

Trong đó: 9

phạm_vi cho biết phạm vi truy nhập của thuộc tính. Có ba kiểu xác định thuộc tính phổ biến là: 10

+: thuộc tính kiểu public 11

#: thuộc tính kiểu protected

-: thuộc tính kiểu private.

~: thuộc tính được phép truy nhập tới từ các lớp trong cùng package

Các phạm vi của thuộc tính có thể được biểu diễn dưới dạng ký hiệu (+, #, -, ~) hoặc biểu diễn dưới dạng các từ khoá (public, protected, private). 12

Tên: là xâu ký tự biểu diễn tên thuộc tính. 13

kiểu: là kiểu dữ liệu của thuộc tính.

số_đối_tượng: chỉ ra số đối tượng khai báo cho thuộc tính ứng với một *mặc_định:* là giá trị khởi đầu mặc định (nếu có) của thuộc tính.

Giá_ trị_giới_hạn: là giới hạn các giá trị cho thuộc tính (thông tin này không bắt buộc). 14

Ví dụ một khai báo thuộc tính đầy đủ:

purchaseDate:Date[1] =”01-01-2000” (Saturday) 15

- ***Phương thức (method):*** các phương thức trong UML được biểu diễn theo cấu trúc chung như sau [UNG]: 1

phạm_vi tên(danh_s ách_tham_số): kiểu_trả_lại { ki ểu_ph ương_thức} 2

Trong đó: 3

visibility biểu diễn phạm vi cho phương thức. Giống như đối với thuộc tính, có ba dạng kiểu xác định cơ bản cho phương thức là: 4

- +: phương thức kiểu public
- #: phương thức kiểu protected
- -: phương thức kiểu private
- ~: phương thức được phép truy nhập tới từ các lớp trong cùng package 5

tên là xâu ký tự xác định tên của phương thức.

kiểu_trả_lại: chỉ ra kiểu giá trị trả về của phương thức.

danh_sách_tham_số: biểu diễn danh sách các tham số trong khai báo của phương thức. Mỗi tham số được biểu diễn dưới dạng chung: tên tham số: *kiểu giá trị = giá trị mặc định.* 6

ki ểu_ph ương_thức: không bắt buộc, cho biết kiểu phương thức. Phương thức có thể nhận một trong các kiểu đặc biệt sau: 7

abstract: phương thức kiểu trừu tượng 8

query: phương thức kiểu truy vấn.

Ví dụ một khai báo phương thức cho một lớp: 9

generatePurchaseList(prodID:int): String 10

- ***Các kiểu lớp trong UML*** 11

UML định nghĩa một số kiểu lớp đăc biệt dựa trên vai trò của nó trong biểu đồ lớp. Ngoài kiểu lớp thông thường đã trình bày ở trên, UML còn định nghĩa một số kiểu lớp bổ sung gồm: 12

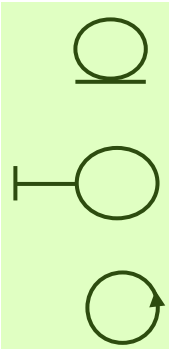
- *Lớp thực thể:* là lớp đại diện cho các thực thể chứa thông tin về các đối tượng xác định nào đó. 13
- *Lớp biên (lớp giao diện):* là lớp nằm ở ranh giới giữa hệ thống với môi trường bên ngoài, thực hiện vai trò nhận yêu cầu trực tiếp từ các tác nhân và chuyển các yêu cầu đó cho các lớp bên trong hệ thống.
- *Lớp điều khiển:* thực hiện các chức năng điều khiển hoạt động của hệ thống ứng với các chức năng cụ thể nào đó với một nhóm các lớp biên hoặc lớp thực thể xác định.

STT	Kiểu lớp	Kí hiệu UML

1 *Lớp thực thể* 1

2 *Lớp biên (lớp giao diện)* 2

3 *Lớp điều khiển* 3



4

Bảng 1.3: Các kiểu lớp trong UML 5

- ***Các mối quan hệ trong biểu đồ lớp*** 6

Giữa các lớp có các dạng quan hệ cơ bản như sau: 7

- *Quan hệ kết hợp (Association)*: Một kết hợp (association) là một sự nối kết giữa các lớp, cũng có nghĩa là sự nối kết giữa các đối tượng của các lớp này. Trong UML, một quan hệ đượcấnc định nhằm mô tả một tập hợp các liên kết (links), tức là một sự liên quan về ngữ nghĩa (semantic connection) giữa một nhóm các đối tượng được biểu diễn bởi các lớp tương ứng. Mặc định, quan hệ kết hợp được biểu diễn bởi đoạn thẳng 2 chiều nối 2 đối tượng và có thể kèm theo ngữ nghĩa của quan hệ tại hai đầu của đoạn thẳng. 8
- *Khái quát hóa (Generalization)*: Khái quát hóa là mối quan hệ giữa một lớp có các đặc trưng mang tính khái quát cao hơn và một lớp có tính chất đặc biệt hơn. Trong sơ đồ lớp, mối quan hệ khái quát hóa chính là sự kế thừa của một lớp từ lớp khác. Quan hệ khái quát hoá được biểu diễn bằng một mũi tên có tam giác rỗng gắn ở đầu. 9
- *Quan hệ cộng hợp (Aggregation)*: là dạng quan hệ mô tả một lớp A là một phần của lớp B và lớp A có thể tồn tại độc lập. Quan hệ cộng hợp được biểu diễn bằng một mũi tên gắn hình thoi rỗng ở đầu hướng về lớp bao hàm. 10
- *Quan hệ gộp (Composition):* Một quan hệ gộp biểu diễn một quan hệ kiểu tổng thể-bộ phận. Lớp A có quan hệ gộp với lớp B nếu lớp A là một phần của lớp B và sự tồn tại của đối tượng lớp B điều khiển sự tồn tại của đối tượng lớp A. Quan hệ này được biểu diễn bởi một mũi tên gắn hình thoi đặc ở đầu. 11
- *Quan hệ phụ thuộc (Dependency)*: Phụ thuộc là mối quan hệ giữa hai lớp đối tượng: một lớp đối tượng A có tính độc lập và một lớp đối tượng B phụ thuộc vào A; một sự thay đổi của A sẽ ảnh hưởng đến lớp phụ thuộc B. 12
- *Quan hệ thực thi (Realization):* biểu diễn mối quan hệ ngữ nghĩa giữa các thành phần của biểu đồ lớp, trong đó một thành phần mô tả một công việc dạng hợp đồng và thành phần còn lại thực hiện hợp đồng đó. Thông thường lớp *thực hiện hợp đồng* có thể là các giao diện. 13

Bảng 1.4 tổng kết các phần tử mô hình UML được sử dụng trong mô hình lớp, ý nghĩa và ký hiệu tương ứng trong các biểu đồ. 1

Phần tử mô hình	Ý nghĩa	Cách biểu diễn	Ký hiệu trong biểu đồ
Lớp (class)	Biểu diễn tên lớp, các thuộc tính và phương thức của lớp đó.	Một hình chữ nhật gồm 3 phần tách biệt.	Tên lớp Các thuộc tính Các phương thức
Quan hệ kiểu [illegible] *ợp* Tên	Biểu diễn quan hệ [illegible] hai lớp độc [illegible] có liên quan đến nhau.	Một đường kẻ liền nét (có tên xác định) nối giữa hai lớp.	
hệ gộp	[illegible] diễn quan hệ [illegible] bộ phận – tổng thể.	Đường kẻ liền nét có hình thoi ở đầu.	
Quan hệ khái quát hoá (kế [illegible]	Lớp này thừa hưởng các thuộc [illegible] phương thức của lớp kia	Mũi tên tam giác.	
[illegible] *hệ phụ* [illegible]	[illegible] ớp phụ thuộc [illegible] au trong hoạt động của hệ thống.	Mũi tên đứt nét.	

Bảng 1.4: Tóm tắt các phần tử mô hình UML trong biểu đồ lớp 2

3

c. Biểu đồ trạng thái 4

Ý nghĩa

Biểu đồ trạng thái được sử dụng để biểu diễn các trạng thái và sự chuyển tiếp giữa các trạng thái của các đối tượng trong một lớp xác định. Thông thường, mỗi lớp sẽ có một biểu đồ trạng thái (trừ lớp trừu tượng là lớp không có đối tượng). Biểu đồ trạng thái được biểu diễn dưới dạng máy trạng thái hữu hạn với các trạng thái và sự chuyển tiếp giữa các trạng thái đó. Có hai dạng biểu đồ trạng thái: 5

- Biểu đồ trạng thái cho một use case: mô tả các trạng thái và chuyển tiếp trạng thái của một đối tượng thuộc một lớp nào đó trong hoạt động của một use case cụ thể. 6

- Biểu đồ trạng thái hệ thống mô tả tất cả các trạng thái của một đối tượng trong toàn bộ hoạt động của cả hệ thống. 1

Tập ký hiệu UML cho biểu đồ trạng thái 2

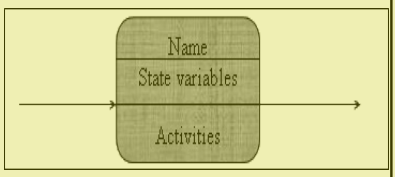
Các thành phần trong một biểu đồ trạng thái bao gồm: 3

- ***Trạng thái (state).*** Bên trong các trạng thái có thể miêu tả các biến trạng thái hoặc các hành động (action) tương ứng với trạng thái đó. 4
- ***Trạng thái con (substate):*** là một trạng thái chứa bên trong một trạng thái khác. Trạng thái có nhiều trạng thái con gọi là trạng thái tổ hợp. 5
- ***Trạng thái khởi đầu (initial state):*** trạng thái đầu tiên khi kích hoạt đối tượng. 6
- ***Trạng thái kết thúc (final state):*** kết thúc vòng đời đối tượng. 7
- ***Các chuyển tiếp (transition):*** biểu diễn các chuyển đổi giữa các trạng thái. 8
- ***Sự kiện (event):*** sự kiện tác động gây ra sự chuyển đổi trạng thái. Mỗi sự kiện được đi kèm với các điều kiện (guard) và các hành động (action). 9

Trong biểu đồ trạng thái của UML, một số loại sự kiện sau đây sẽ được xác định: 10

- *Sự kiện gọi (call event)*: Yêu cầu thực hiện một hành động (một phương thức) 11
- *Sự kiện tín hiệu (signal event):* Gửi thông điệp (chứa các giá trị thuộc tính tham số liên quan) giữa các trạng thái.
- *Sự kiện thời gian (time event):* Biểu diễn quá trình chuyển tiếp theo thời gian, thường kèm theo từ mô tả thời gian cụ thể.

Các phần tử mô hình UML và ký hiệu tương ứng cho biểu đồ trạng thái được tổng kết như trong Bảng 1.5. 12

Phần tử mô hình	**Ý nghĩa**	**Biểu diễn**	**Ký hiệu trong biểu đồ**
Trạng thái	Biểu diễn một trạng thái của đối tượng trong vòng đời của đối tượng đó.	Hình chữ nhật vòng ở góc, gồm 3 phần: tên, các biến, và các hoạt động.	
Trạng thái khởi đầu	Khởi đầu vòng đời của đối tượng.	Hình tròn đặc	
Trạng thái kết thúc	Kết thúc vòng đời của đối tượng.	Hai hình tròn lồng nhau	

13

Chuyển tiếp (transition) Tên_chuyển_tiếp	Chuyển từ trạng thái này sang trạng thái khác	Mũi tên liền nét với tên gọi là biểu diễn của chuyển tiếp đó.	

2

Bảng 1.5: Các phần tử mô hình UML trong biểu đồ trạng thái 1

d. Biểu đồ tương tác dạng tuần tự 3

Các biểu đồ tương tác biểu diễn mối liên hệ giữa các đối tượng trong hệ thống và giữa các đối tượng với các tác nhân bên ngoài. Có hai loại biểu đồ tương tác: Biểu đồ tuần tự và biểu đồ cộng tác. 4

Ý nghĩa 5

Biểu đồ tuần tự: Biểu diễn mối quan hệ giữa các đối tượng, giữa các đối tượng và tác nhân theo thứ tự thời gian. Biểu đồ tuần tự nhấn mạnh thứ tự thực hiện của các tương tác. 6

Tập ký hiệu UML cho biểu đồ tuần tự 7

Các thành phần cơ bản của một biểu đồ tuần tự là: 8

- *Các đối tượng (object):* được biểu diễn bởi các hình chữ nhật, bên trong là tên của đối tượng. Cách viết chung của đối tượng là: *tên đối tượng: tên lớp*. Nếu chỉ viết*: tên_lớp* thì có nghĩa là bất cứ đối tượng nào của lớp tương ứng đó. Trong biểu đồ tuần tự, không phải các đối tượng đều xuất hiện ở trên cùng của biểu đồ mà chúng chỉ xuất hiện (về mặt thời gian) khi thực sự tham gia vào tương tác. 9
- *Các message:* được biểu diễn bằng các mũi tên hướng từ đối tượng gửi sang đối tượng nhận. Tên các message có thể biểu diễn dưới dạng phi hình thức (như các thông tin trong kịch bản) hoặc dưới dạng hình thức (với dạng giống như các phương thức). Biểu đồ tuần tự cho phép có các message từ một đối tượng tới chính bản thân nó. 10
- Trong biểu đồ tuần tự có thể có nhiều loại message khác nhau tuỳ theo mục đích sử dụng và tác động của message đến đối tượng. Các dạng message được tổng kết trong Bảng 1.6 dưới đây: 11

STT	***Loại message***	***Mô tả***	***Biểu diễn***
1	*Gọi (call)*	Mô tả một lời gọi từ đối tượng này đến đối tượng kia.	Method()
2	*Trả về (return)*	Trả về giá trị ứng với lời gọi	Giá trị trả về

12

3	*Gửi (send)*	Gửi một tín hiệu tới một đối tượng	Send()
4	*Tạo (create)*	Tạo một đối tượng	<<create>>
5	*Huỷ (destroy)*	Huỷ một đối tượng	<<destroy>>

1

Bảng 1.6: Các dạng message trong biểu đồ tuần tự

- *Đường lifeline:* là một đường kẻ nối dài phía dưới đối tượng, mô tả quá trình của đối tượng trong tương tác thuộc biểu đồ. 2
- *Chú thích:* biểu đồ tuần tự cũng có thể có chú thích để người đọc dễ dàng hiểu được nội dung của biểu đồ đó.

e. Biểu đồ tương tác dạng cộng tác 3
Ý nghĩa

Biểu đồ cộng tác: Là biểu đồ tương tác biểu diễn mối quan hệ giữa các đối tượng; giữa các đối tượng và tác nhân nhấn mạnh đến vai trò của các đối tượng trong tương tác. 4

Biểu đồ cộng tác cũng có các messgage với nội dung tương tự như trong biểu đồ tuần tự. Tuy nhiên, các đối tượng được đặt một cách tự do trong không gian của biểu đồ và không có đường life line cho mỗi đối tượng. Các message được đánh số thể hiện thứ tự thời gian. 5

Tập ký hiệu UML cho biểu đồ cộng tác 6

Các thành phần cơ bản của một biểu đồ cộng tác là: 7

- *Các đối tượng:* được biểu diễn bởi các hình chữ nhật, bên trong là tên của đối tượng. Cách viết chung của đối tượng là: *tên đối tượng: tên lớp*. Trong biểu đồ cộng tác, các đối tượng tham gia tương tác luôn xuất hiện tại một vị trí xác định. 8
- *Các liên kết*: giữa hai đối tượng có tương tác sẽ có một liên kết nối 2 đối tượng đó. Liên kết này không có chiều.
- *Các message:* được biểu diễn bằng các mũi tên hướng từ đối tượng gửi sang đối tượng nhận bên cạnh liên kết giữa 2 đối tượng đó. Trong biểu đồ cộng tác, các message được đánh số thứ tự theo thứ tự xuất hiện trong kịch bản mô tả use case tương ứng.

f. Biểu đồ hoạt động 9
Ý nghĩa

Biểu đồ hoạt động biểu diễn các hoạt động và sự đồng bộ, chuyển tiếp các hoạt động của hệ thống trong một lớp hoặc kết hợp giữa các lớp với nhau trong một chức năng cụ thể. 1

Biểu đồ hoạt động có thể được sử dụng cho nhiều mục đích khác nhau, ví dụ như: 2

- Để xác định các hành động phải thực hiện trong phạm vi một phương thức. 3
- Để xác định công việc cụ thể của một đối tượng.
- Để chỉ ra một nhóm hành động liên quan của các đối tượng được thực hiện như thế nào và chúng sẽ ảnh hưởng đến những đối tượng nằm xung quanh.

Tập ký hiệu UML 4

Các phần tử mô hình UML cho biểu đồ hoạt động bao gồm: 5

- *Hoạt động* (*Activity*): là một quy trình được định nghĩa rõ ràng, có thể được thực hiện bởi một hàm hoặc một nhóm đối tượng. Hoạt động được thể hiện bằng hình chữ nhật tròn cạnh. 6
- *Thanh đồng bộ hóa (Synchronisation bar):* cho phép ta mở ra hoặc là đóng lại các nhánh chạy song song trong tiến trình.
- *Điều kiện (Guard Condition)*: các biểu thức logic có giá trị hoặc đúng hoặc sai. Điều kiện được thể hiện trong ngoặc vuông, ví dụ: [Customer existing].
- *Các luồng (swimlane):* Mỗi biểu đồ động có thể biểu diễn sự phối hợp hoạt động trong nhiều lớp khác nhau. Khi đó mỗi lớp được phân tách bởi một luồng (swimlane) riêng biệt. Các luồng này được biểu diễn đơn giản là các ô khác nhau trong biểu đồ.

Các ký hiệu UML cho biểu đồ hoạt động được tổng kết trong Bảng sau: 7

8

Phần tử mô hình	Ý nghĩa	Ký hiệu trong biểu đồ
Hoạt động	Mô tả một hoạt động gồm tên hoạt động và đặc tả của nó.	NewActivity
Trạng thái khởi đầu		
Trạng thái kết thúc		
Thanh đồng bộ ngang	Mô tả thanh đồng bộ nằm ngang	

Thanh đồng bộ hoá dọc	Mô tả thanh đồng bộ theo chiều thẳng đứng	
Chuyển tiếp		
Quyết định	Mô tả một lựa chọn điều kiện.	
Các luồng (swimlane)	Phân tách các lớp đối tượng khác nhau tồn tại trong biểu đồ hoạt động	Phân cách nhau bởi một đường kẻ dọc từ trên xuống dưới biểu đồ

2

Bảng 1.7: Các phần tử của biểu đồ hoạt động 1

g. Biểu đồ thành phần 3
Ý nghĩa

Biểu đồ thành phần được sử dụng để biểu diễn các thành phần phần mềm cấu thành nên hệ thống. 4 Một hệ phần mềm có thể được xây dựng từ đầu bằng cách sử dụng mô hình lớp như đã trình bày trong các phần trước của tài liệu, hoặc cũng có thể được tạo nên từ các thành phần sẵn có.

Mỗi thành phần có thể coi như một phần mềm nhỏ hơn, cung cấp một khối dạng hộp đen 5 trong quá trình xây dựng phần mềm lớn. Nói cách khác, các thành phần là các gói được xây dựng cho quá trình triển khai hệ thống. Các thành phần có thể là các gói ở mức cao như JavaBean, các gói thư viện liên kết động dll, hoặc các phần mềm nhỏ được tạo ra từ các thành phần nhỏ hơn như các lớp và các thư viện chức năng.

Tập ký hiệu UML 6

Tập ký hiệu UML cho biểu đồ thành phần được tổng kết trong bảng sau: 7

Phần tử mô hình	**Ý nghĩa**	**Ký hiệu trong biểu đồ**

8

Thành phần	Mô tả một thành phần của biểu đồ, mỗi thành phần có thể chứa nhiều lớp hoặc nhiều chương trình con.	Component
Giao tiếp	Mô tả giao tiếp gắn với mỗi thành phần. Các thành phần trao đổi thông tin qua các giao tiếp.	
Mối quan hệ phụ thuộc giữa các thành phần	Mối quan hệ giữa các thành phần (nếu có).	
Gói (package)	Được sử dụng để nhóm một số thành phần lại với nhau.	NewPackage

2

Bảng 1.8: Các ký hiệu của biểu đồ thành phần 1

h. Biểu đồ triển khai hệ thống 3

Ý nghĩa

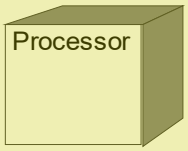
Biểu đồ triển khai biểu diễn kiến trúc cài đặt và triển khai hệ thống dưới dạng các nodes và các mối quan hệ giữa các node đó. Thông thường, các nodes được kết nối với nhau thông qua các liên kết truyền thông như các kết nối mạng, liên kết TCP-IP, microwave… và được đánh số theo thứ tự thời gian tương tự như trong biểu đồ cộng tác. 4

Tập ký hiệu UML cho biểu đồ triển khai 5

Tập ký hiệu UML cho biểu đồ triển khai hệ thống được biểu diễn trong Bảng sau: 6

Phần tử mô hình	**Ý nghĩa**	**Ký hiệu trong biểu đồ**
Các nodes (hay các thiết bị)	Biểu diễn các thành phần không có bộ vi xử lý trong biểu đồ triển khai hệ thống	Device

7

Các bộ xử lý	Biểu diễn các thành phần có bộ vi xử lý trong biểu đồ triển khai hệ thống	
Các liên kết truyền thông	Nối các thành phần của biểu đồ triển khai hệ thống. Thường mô tả một giao thức truyền thông cụ thể.	

Bảng 1.9: Các ký hiệu của biểu đồ triển khai hệ thống

1.4 CÂU HỎI ÔN TẬP

CHƯƠNG 2: ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM [1]

2.1 TỔNG QUAN VỀ ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM [2]

2.1.1 Một số khái niệm [3]

Khái niệm đảm bảo chất lượng phần mềm được định nghĩa như sau : [4]

Đảm bảo chất lượng phần mềm là một tập các hoạt động đã được lập kế hoạch và có hệ thống, cần thiết để cung cấp đầy đủ sự tin cậy vào quy trình phát triển phần mềm hay quy trình bảo trì phần mềm của sản phẩm hệ thống phần mềm phù hợp với các yêu cầu chức năng kỹ thuật cũng như với các yêu cầu quản lý mà giữ cho lịch biểu và hoạt động trong phạm vi ngân sách. [5]

Mục tiêu đảm bảo chất lượng phần mềm [6]

Những mục tiêu đảm bảo chất lượng phần mềm tương ứng với giai đoạn phát triển và bảo trì được xác định cụ thể như sau : [7]

- Phát triển phần mềm (hướng tiến trình) [8]

1. Đảm bảo một mức độ chấp nhận được rằng phần mềm sẽ thực hiện được các yêu cầu chức năng. [9]
2. Đảm bảo một mức đọ cấp nhận được rằng phần mềm sẽ đáp ứng được các yêu cầu về lịch biểu và ngân sách [10]
3. Thiết lập và quản lý các hoạt động để cải thiện và nâng cao hiệu quả của phát triển phần mềm và các hoạt động SQA. [11]

- Bảo trì phần mềm (hướng sản phẩm) [12]

1. Đảm bảo một mức độ chấp nhận được rằng các hoạt động bảo trì phần mềm sẽ đáp ứng được các yêu cầu chức năng. [13]
2. Đảm bảo một mức đọ cấp nhận được rằng các hoạt động bảo trì phần mềm sẽ đáp ứng được các yêu cầu về lịch biểu và ngân sách [14]
3. Thiết lập và quản lý các hoạt động để cải thiện và nâng cao hiệu quả của bảo trì phần mềm. [15]

Xác minh, thẩm định và đánh giá chất lượng [16]

Ba khía cạnh đảm bảo chất lượng của sản phẩm phần mềm gồm Verification, Validation, và Qualification. [17]

- ***Verification***: quá trình đánh giá một hệ thống hay một thành phần để xác định xem những sản phẩm của một pha phát triển xác định có thỏa mãn những điều kiện được đặt gia khi bắt đầu pha đó hay không. 1

- ***Validation:*** quá trình đánh giá một hệ thống hay một thành phần trong suốt hoặc khi kết thúc quá trình phát triển để xác định xem nó có thỏa mãn những yêu cầu đã đặc tả hay không. 2

- ***Qualification:*** quá trình xác định một hệ thống hoặc một thành phần có phù hợp với việc sử dụng hay không. 3

Theo những định nghĩa của IEEE, verification kiểm tra tính nhất quán của sản phẩm đang phát triển với những sản phẩm đã được phát triển ở pha trước. Khi thực hiện, người thẩm tra đi sau quy trình phát triển và giả sử rằng tất cả những pha phát triển đằng trước đã được hoàn thành một cách chính xác hoặc là như kế hoặc gốc hoặc là sau khi đã sửa chữa những sai sót được phát hiện. 4

Validation miêu tả sự quan tâm của khách hàng, bằng cách thẩm tra những yêu cầu gốc của họ. 5

Qualification tập trung vào những khía cạnh hoạt động, ở đó việc bảo trì là vấn đề chính. Một thành phần phần mềm đã được phát triển và tài liệu hóa theo những chuẩn, kiểu, và cấu trúc chuyên nghiệp sẽ dễ dàng để bảo trì hơn những thành phần có code lạ. 6

Người lên kế hoạch cần phải xác định xem những khía cạnh nào nên được sát hạnh trong mỗi hoạt động đảm bảo chất lượng phần mềm. 7

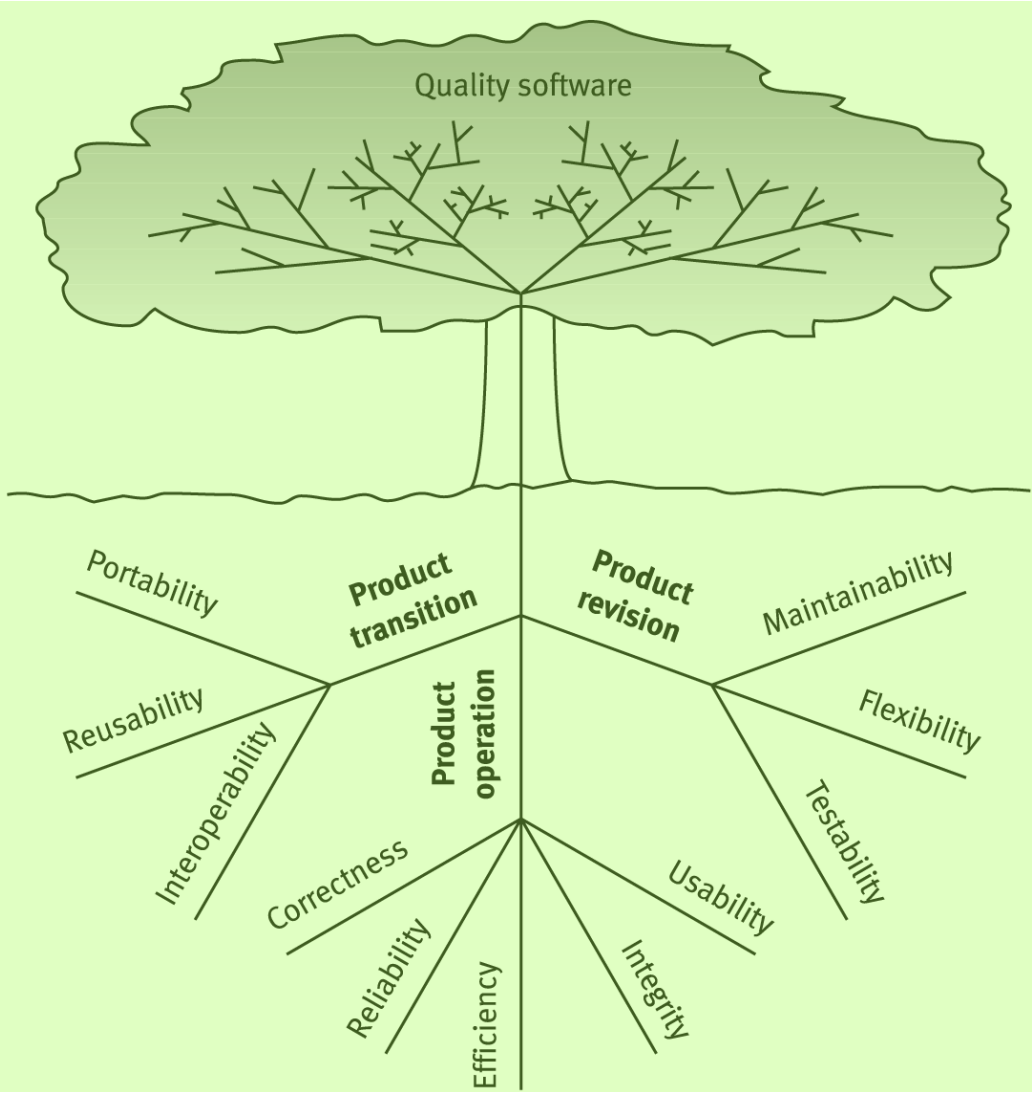
Xác minh thường là hoạt động có tính kỹ thuật cao hơn, sử dụng những tri thức về các yêu cầu, đặc tả phần mềm. Xác nhận thường phụ thuộc vào tri thức về lĩnh vực tương ứng. Cụ thể là, tri thức về ứng dụng của phần mềm được viết. Ví dụ, xác nhận của phần mềm về máy bay yêu cầu tri thức từ kỹ sư hàng không và phi công. 8

Cụm từ IV & V - independent verification and validation- nghĩa là xác nhận và xác minh độc lập. IV & V yêu cầu việc đánh giá phải được thực hiện bởi người không phải là lập trình viên. Một số trường hợp đội IV & V thuộc dự án, hoặc thuộc cùng công ty, cũng có thể thuộc một tổ chức độc lập. Vì sự độc lập của IV & V, tiến trình thường chưa bắt đầu cho tới khi dự án kết thúc và thường được thực hiện bởi chuyên gia trong lĩnh vực hơn là người phát triển phần mềm. 9

2.1.2 Các tiêu chí chất lượng 10

Đã có nhiều tác giả nghiên cứu về các tiêu chí chất lượng phần mềm từ các yêu cầu. Theo thời gian có thể quan niệm về việc đảm bảo chất lượng phần mềm có phần thay đổi, tuy nhiên mô hình các yếu tố đảm bảo chất lượng phần mềm của McCall ra đời vào những năm 70 của thế kỷ trước vẫn còn được nhiều người nhắc đến như là cơ sở tham chiếu các yêu cầu phần mềm. Sau McCall cũng có một số mô hình được quan tâm như mô hình do Evans, Marciniak do hay mô hình của Deutsch và Willis, tuy nhiên những mô hình này chỉ bổ sung hay sửa đổi một vài yếu tố chất lượng. Theo McCall, các yếu tố chất lượng phần mềm được chia làm ba loại : 11

- Các yếu tố hoạt động của sản phẩm bao gồm tính chính xác, tin cậy, hiệu quả, tính toàn vẹn, sử dụng được 1
- Các yếu tố rà soát bao gồm tính bảo trì, linh hoạt, có thể test được
- Các yếu tố chuyển giao bao gồm tính khả chuyển, có khả năng sử dụng lại, có khả năng giao tác.



2

Hình 2.1: Cây mô hình tiêu chí chất lượng phần mềm theo MCCall 3

Chi tiết các thuộc tính được phân tích như sau : 4

(1) **Các yếu tố vận hành sản phẩm :** Sự chính xác, độ tin cậy, tính hiệu quả, tính toàn vẹn và khả năng sử dụng được : 5

- **Sự chính xác :** Các yêu cầu về độ chính xác được xác định trong một danh sách các đầu ra cần thiết của hệ thống phần mềm, như màn hình hiển thị truy vấn số dư của khách hàng 6

trong một hệ thống thông tin kế toán bán hàng…Các đặc tả đầu ra thường là đa chiều, một số chiều thông dụng là : 1

- Nhiệm vụ đầu ra (ví dụ : bản in hóa đơn bán hàng, hay đèn báo động đỏ khi nhiệt độ tăng lên trên 250 độ F) 2
- Độ chính xác yêu cầu của các đầu ra này; chúng có thể bị ảnh hưởng bất lợi bởi các tính toán không chính xác hay các dữ liệu không chính xác.
- Tính đầy đủ của thông tin đầu ra; chúng có thể bị ảnh hưởng bất lợi bởi dữ liệu không đầy đủ.
- Up-to-dateness của thông tin (xác định bằng thời gian giữa sự kiện và việc xem xét hệ thống phần mềm.
- Độ sẵn sàng của thông tin (thời gian đáp ứng : được định nghĩa là thời gian cần thiết để có được các thông tin yêu cầu)
- Các chuẩn cho việc code và viết tài liệu cho hệ thống phần mềm.

- **Độ tin cậy :** Các yêu cầu về độ tin cậy giải quyết các lỗi để cung cấp dịch vụ. Chúng xác định tỷ lệ lỗi hệ thống phần mềm tối đa cho phép, các lỗi này có thể là lỗi toàn bộ hệ thống hoặc một hay nhiều chức năng riêng biệt của nó. 3
- **Tính hiệu quả :** Các yêu cầu về tính hiệu quả giải quyết vấn đề về các tài nguyên phần cứng cần thiết để thực hiện tất cả các chức năng của hệ thống phần mềm với sự phù hợp của tất cả các yêu cầu khác. Các tài nguyên phần cứng chính được xem xét ở đây là khả năng xử lý của máy tính (được đo bằng MIPS – triệu lệnh/giây; MHz – triệu chu kỳ/giây…); khả năng lưu trữ dữ liệu (dung lượng bộ nhớ, dung lượng đĩa – được đo bằng MBs, GBs, TBs…) và khả năng truyền dữ liệu (thường được đo bằng MBPS, GBPS). Các yêu cầu này có thể bao gồm cả các giá trị tối đa tài nguyên phần cứng được sử dụng trong hệ thống phần mềm. Một yêu cầu khác về tính hiệu quả đó là thời gian giữa các lần phải sạc điện đối với các hệ thống nằm trên các máy tính xách tay hay các thiết bị di động. 4
- **Tính toàn vẹn:** Các yêu cầu về tính toàn vẹn giải quyết các vấn đề về bảo mật hệ thống phần mềm, các yêu cầu này để ngăn chặn sự truy cập trái phép, để phân biệt giữa phần lớn nhân viên chỉ được phép xem thông tin với một nhóm hạn chế những người được phép thêm và thay đổi dữ liệu… 5
- **Tính khả dụng:** Các yêu cầu về khả năng sử dụng được sẽ đưa ra phạm vi của tài nguyên nhân lực cần thiết để đào tạo một nhân viên mới và để vận hành hệ thống phần mềm. 6

(2) **Các yếu tố về rà soát sản phẩm :** bảo trì được, linh động và kiểm tra được : 7

- **Khả năng bảo trì được** : Các yêu cầu về khả năng bảo trì được sẽ xác định người dùng và nhân viên bảo trì phải nỗ lực thế nào để xác định được nguyên nhân của các lỗi phần mềm, để sửa lỗi và để xác nhận việc sửa lỗi thành công. Các yêu cầu của yếu tố này nói tới cấu trúc modul của phần mềm, tài liệu chương trình nội bộ và hướng dẫn sử dụng của lập trình viên… 1

- **Tính linh động** : Các yêu cầu về tính linh động cũng bao gồm cả các khả năng và nỗ lực cần thiết để hỗ trợ các hoạt động bảo trì. Chúng gồm các nguồn lực (man-day) cần thiết để thích nghi với một gói phần mềm, với các khách hàng trong cùng nghề, với các mức độ hoạt động khác nhau, với các loại sản phẩm khác nhau…Các yêu cầu về yếu tố này cũng hỗ trợ các hoạt động bảo trì trở nên hoàn hảo, như thay đổi và bổ sung vào phần mềm để tăng dịch vụ của nó và để thích nghi với các thay đổi trong môi trường thương mại và kỹ thuật của công ty. 2

- **Khả năng test được** : Các yêu cầu về khả năng kiểm tra được nói tới việc kiểm tra sự vận hành có tốt hay không của các hệ thống thông tin. Các yêu cầu về khả năng kiểm tra được liên quan tới các tính năng đặc biệt trong chương trình giúp người tester dễ dàng thực hiện công việc của mình hơn, ví dụ như đưa ra các kết quả trung gian. Các yêu cầu về khả năng kiểm tra được liên quan tới vận hành phần mềm bao gồm các chuẩn đoán tự động được thực hiện bởi hệ thống phần mềm trước khi bắt đầu hệ thống, để tìm hiểu xem có phải tất cả các thành phần của hệ thống phần mềm đều làm việc tốt hay không, và để có một bản báo cáo về các lỗi đã được phát hiện. Một loại khác của yêu cầu này là việc check các dự đoán tự động, được các kỹ thuật viên bảo trì sử dụng để phát hiện nguyên nhân gây lỗi phần mềm. 3

(3) **Các yếu tố về chuyển giao sản phẩm :** tính lưu động (khả năng thích nghi với môi trường), khả năng tái sử dụng và khả năng cộng tác được : 4

- **Tính lưu động :** các yêu cầu về tính lưu động nói tới khả năng thích nghi của hệ thống phần mềm với các môi trường khác, bao gồm phần cứng khác, các hệ điều hành khác… Các yêu cầu này đòi hỏi các phần mềm cơ bản có thể tiếp tục sử dụng độc lập hoặc đồng thời trong các trường hợp đa dạng. 5

- **Khả năng tái sử dụng** : Các yêu cầu về khả năng tái sử dụng nói tới việc sử dụng các modul phần mềm trong một dự án mới đang được phát triển mà các modul này ban đầu được thiết kế cho một dự án khác. Các yêu cầu này cũng cho phép các dự án tương lai có thể sử dụng một modul đã có hoặc một nhóm các modul hiện đang được phát triển. Tái sử dụng phần mềm sẽ tiết kiệm tài nguyên phát triển, rút ngắn thời gian phát triển và tạo ra các moduls chất lượng cao hơn. Chất lượng modul cao hơn là dựa trên giả định rằng hầu hết các lỗi phần mềm đều được phát hiện bởi các hoạt động đảm bảo chất lượng phần mềm thực hiện trên phần mềm ban đầu, bởi những người sử dụng phần mềm ban đầu và trong suốt những lần tái sử dụng trước của nó. Các vấn đề về tái sử dụng phần mềm đã trở thành một phần trong chuẩn công nghiệp phần mềm (IEEE,1999). 6

- **Khả năng cộng tác** : Các yêu cầu về khả năng cộng tác tập trung vào việc tạo ra các giao diện với các hệ thống phần mềm khác. Các yêu cầu về khả năng cộng tác có thể xác định tên của phần mềm với giao diện bắt buộc. Chúng cũng có thể xác định cấu trúc đầu ra được chấp nhận như một tiêu chuẩn trong một ngành công nghiệp cụ thể hoặc một lĩnh vực ứng dụng. 1

Các tiêu chí chất lượng ảnh hưởng tới hoạt động đảm bảo chất lượng phần mềm như nào 2

Những hoạt động đảm bảo chất lượng là hướng quy trình, nói cách khác, có liên kết tới sự hoàn thành của một pha dự án, sự hoàn tất một mốc dự án, và nhiều hơn nữa. Những hoạt động đảm bảo chất lượng được tích hợp vào trong kế hoạch phát triển. 3

Những người lên kế hoặc đảm bảo chất lượng cho một dự án cần xác định: 4

- Danh sách những hoạt động đảm bảo chất lượng cần thiết cho dự án. 5
- Với mỗi hoạt động đảm bảo chất lượng:
  - Thời gian.
  - Loại hoạt động đảm bảo chất lượng áp dụng.
  - Người thực hiện hoạt động và tài nguyên yêu cầu.
  - Những tài nguyên yêu cầu cho việc khắc phục những nhược sai sót và những thay đổi.

Cường độ của những hoạt động đảm bảo chất lượng đã được lên kế hoạch được chỉ ra bởi số những hoạt động yêu cầu. Những yếu tố dự án và nhóm ảnh hưởng tới cường độ như sau: 6

Yếu tố dự án: 7

- Độ lớn của dự án. 8
- Sự phức tạp và khó của kỹ thuật.
- Phạm vi của những thành phần sử dụng lại.
- Hậu quả nếu dự án bị lỗi.

Yếu tố nhóm: 9

- Trình độ chuyên môn của những thành viên trong nhóm. 10
- Sự quen thuộc của nhóm với dự án và Kinh nghiệm của nhóm trong lĩnh vực.
- Tính sẵn sàng của những thành viên có thể trợ giúp nhóm.
- Sự hiểu biết giữa những thành viên trong nhóm, hay nói cách khác là số thành viên mới trong nhóm.

2.2 CÁC HOẠT ĐỘNG ĐẢM BẢO CHẤT LƯỢNG PHẦN MỀM 1

2.2.1. Đảm bảo chất lượng đặc tả 2

Chất lượng đặc tả 3

Đặc tả không có các hoạt động trước nó, và tất cả các hoạt động khác đều theo sau đặc tả. Vì vậy, nếu đặc tả không tốt thì phân tích, thiết kế cũng sẽ không tốt, và dẫn tới phát triển, bảo trì của 1 phần mềm kém hơn, hoặc thậm chí một phần mềm lỗi, và công sức bỏ ra để đảm bảo chất lượng thành lãng phí. Vì vậy, điều tối quan trọng là đặc tả phải đầy đủ và được định nghĩa tốt. Đặc tả thường gồm 6 nội dung sau: 4

- Chức năng: đặc tả các chức năng nào sẽ được phát triển 5
- Phi chức năng:
  (a) Khả năng chịu tải: đặc tả khả năng chịu tải của hệ thống (ví dụ: 100 người cùng thao tác)
  (b) Intended use: đặc tả yêu cầu hoặc các yêu cầu mà sản phẩm cần thoả mãn.
  (c) Tính tin cậy: đặc tả thời gian mà sản phẩm có thể hoạt động tốt trước khi cần bảo trì và phù hợp với yêu cầu của người dùng.
  (d) Tính an toàn: đặc tả ngưỡng an toàn cho con người và đặc tính khi sử dụng phần mềm
  (e) Tính bảo mật: đặc tả các mối đe doạ mà sản phẩm cần chuẩn bị

Làm thế nào để đảm bảo đặc tả là đầy đủ và chính xác? Với đặc tả chức năng, người phân tích nghiệp vụ, hoặc người phân tích hệ thống có thể đảm đương nhiệm vụ này. Với đặc tả phi chức năng, cần xây dựng các chuẩn nội bộ hoặc áp dụng các chuẩn chuyên nghiệp. 6

Đảm bảo chất lượng đặc tả 7

Trong công nghiệp phần mềm, đặc tả được xem như là đặc tả của người dùng. Nghĩa là, người dùng cuối coi đây như các yêu cầu của phần mềm mong muốn. Các tình huống sau có thể được dùng để lấy đặc tả người dùng: 8

- Người phân tích nghiệp vụ tìm hiểu, viết báo cáo, xây dựng đặc tả. Họ cần: 9
  - Gặp tất cả người dùng cuối để lấy các yêu cầu và lưu ý của họ 10
  - Gặp tất cả người đứng đầu các bộ phận để lấy yêu cầu và lưu ý của họ
  - Gặp người quản lý để lấy yêu cầu và lưu ý của họ
  - Tổng hợp các yêu cầu và trình bày để người dùng cuối, người đứng đầu các bộ phận và người quản lý phản hồi, nhận xét
  - Chỉnh sửa theo phản hồi, nhận xét và xây dựng bản đặc tả chuẩn
- Có sẵn một bản yêu cầu người dùng như một phần của bản đề xuất 11
- Yêu cầu tham khảo một sản phẩm tương tự và cung cấp các tuỳ chỉnh riêng cho khách hàng. 12

Căn cứ vào các tình huống trên, khi một đặc tả đã sẵn sàng, các hoạt động đảm bảo chất lượng sẽ được thực hiện. Nhiệm vụ của bước đảm bảo chất lượng trong giai đoạn này là 13

đảm bảo các đặc tả đã đầy đủ và phủ hết cả các yêu cầu chức năng và yêu cầu phi chức năng. 1

Các công cụ có thể sử dụng để đảm bảo chất lượng cho đặc tả là: 2

- Tài liệu quy trình: chi tiết phương pháp để lấy, phát triển, phân tích và tổng hợp đặc tả 3
- Các chuẩn (standard), hướng dẫn (guideline), các biểu mẫu (template): xác định tập tối thiểu các đặc tả cần phải xây dựng
- Danh sách kiểm tra (checklist): giúp phân tích để đảm bảo tính đầy đủ của đặc tả

Sử dụng các công cụ này, người phân tích có thể phát triển đặc tả đầy đủ và rõ ràng để sẵn sàng cho hoạt động tiếp theo (phân tích, thiết kế) và đảm bảo chất lượng cho đặc tả. 4

Các công cụ cũng có thể được dùng để đảm bảo chất lượng đặc tả là rà soát chính thức và rà soát ngang hàng. 5

2.2.2. Đảm bảo chất lượng phân tích thiết kế 6

Chất lượng phân tích, thiết kế 7

Nếu phân tích, thiết kế không tốt, sản phẩm sẽ lỗi thậm chí cả khi đặc tả được làm tốt. Phân tích gồm xác định kiến trúc, chức năng, …, cách tiếp cận cho các yêu cầu về tính linh hoạt, tính di động, tính bảo trì được,… Thiết kế gồm thiết kế cơ sở dữ liệu, thiết kế giao diện, báo cáo… Phân tích, thiết kế gồm các thành phần sau: 8

1. Thiết kế chức năng 9
2. Kiến trúc phần mềm
3. Điều hướng (navigation)
4. Thiết kế cơ sở dữ liệu
5. Nền tảng phát triển
6. Nền tảng triển khai
7. Thiết kế giao diện người dùng
8. Thiết kế báo cáo
9. Bảo mật
10. Khả năng chịu lỗi
11. Khả năng chịu tải
12. Tính tin cậy
13. Tính bảo trì được
14. Tính hiệu quả và tính đồng thời
15. Kết nối
16. Đặc tả chương trình
17. Thiết kế kiểm thử

Ta cần người thiết kế có trình độ để đảm bảo thiết kế đúng đã được chọn và triển khai. Các chuẩn thiết kế phần mềm là cần thiết, có thể là được xây dựng nội bộ, hoặc lấy từ một tổ chức chuyên nghiệp. Các chuẩn này hỗ trợ người thiết kế để đạt được bản thiết kế tốt nhất có thể. 10

Thông thường, đầu pha phân tích cần lựa chọn cách thức thiết kế và quyết định các yếu tố chung như số lượng tầng, nền tảng kỹ thuật sử dụng, kết nối trong phần mềm… Bằng cách này, người thiết kế có thể đưa ra giải pháp tốt nhất có thể cho dự án. Một bản mẫu thiết kế có thể được xây dựng và đánh giá. 1

Ở cuối pha phân tích, cần bước đánh giá dựa trên các chuẩn sẵn có để đảm bảo phân tích đạt yêu cầu của dự án. Bản phân tích cần được rà soát ngang hàng, rà soát từ chuyên gia, từ nhà quản lý trước khi chuyển sang thiết kế chi tiết. 2

Đảm bảo chất lượng phân tích, thiết kế 3

Pha phân tích, thiết kế nhận đầu vào là sản phẩm của pha đặc tả, từ đó xây dựng tài liệu phân tích thiết kế. Tài liệu này sẽ được lập trình viên sử dụng để lập trình ra phần mềm đặt yêu cầu. 4

Phân tích – còn gọi là thiết kế mức cao: đặc tả chức năng phần mềm, đặc tả yêu cầu phần mềm, kiến trúc phần mềm. Ở bước này, kiến trúc tổng thể của phần mềm, bao gồm số tầng, số module, cách tiếp cận để đạt được chức năng, thiết kế cơ sở dữ liệu, tính tin cậy, tính bảo mật, … sẽ được xác định và tài liệu hoá. Tài liệu này được sử dụng cho pha thiết kế chi tiết. 5

Thiết kế - còn gọi là thiết kế chi tiết: chi tiết của từng đơn vị chương trình, màn hình, báo cáo, bảng,… được xây dựng và lập trình viên có thể sử dụng nó để lập trình cho phần mềm. 6

Các công cụ để đảm bảo chất lượng phân tích, thiết kế gồm: 7

- Tài liệu quy trình: chi tiết phương thức để thiết kế được xem xét, tiêu chí để lựa chọn các giải pháp cho dự án, cho pha phân tích 8
- Chuẩn, hướng dẫn, các biểu mẫu: xác định kiến trúc cùng với ưu, nhược của nó, phương thức để có danh sách các lựa chọn thay thế,…
- Danh sách kiểm tra (Checklist): giúp người thiết kế đảm bảo thiết kế là đầy đủ và chính xác

2.2.3. Đảm bảo chất lượng phát triển phần mềm (lập trình) 9

Chất lượng lập trình 10

Thông thường, pha phát triển thường có các hoạt động sau: 11

- Tạo cơ sở dữ liệu và các bảng (table) 12
- Phát triển các thư viện kết nối động cho các hoạt động chung
- Phát triển giao diện
- Phát triển báo cáo
- Phát triển kế hoạch test đơn vị
- Phát triển các thành phần với các khía cạnh khác như bảo mật, tính hiệu quả, khả năng chịu lỗi,..

Chất lượng phát triển tốt đạt được bằng cách cung cấp hướng dẫn lập trình cho ngôn ngôn ngữ sử dụng. Hướng dẫn này cần chứa cách đặt tên, cấu trúc code, cách lập trình hiệu quả, cách ngăn ngừa lỗi và giúp lập trình viên viết code đáng tin cậy và không mắc lỗi. Tất nhiên, cần có các lập trình viên tốt. 13

Đảm bảo chất lượng lập trình 1

Phát triển là bước xây dựng phần mềm tuân theo đặc tả. Trong pha này, mã nguồn được xây dựng và kết nối với các thư viện có sẵn, để hoàn thành các yêu cầu lập trình cho sản phẩm. Mã nguồn sẽ được dịch sang chương trình chạy được trên các phần cứng đã xác định trước. Đây cũng là bước cơ sở dữ liệu được xây dựng, từ đó dữ liệu có thể được nạp và sử dụng trong phần mềm. 2

Làm thế nào để đảm bảo chất lượng pha phát triển? Ta có thể sử dụng các chuẩn nội bộ sẵn có về chất lượng code, cũng như các hướng dẫn lập trình cho ngôn ngữ lựa chọn. Ngoài ra, những thay đổi không được kiểm soát có thể dẫn tới lỗi. Vì vậy, quản lý thay đổi và quản lý cấu hình rất quan trọng trong đảm bảo chất lượng code. 3

2.2.4. Kiểm thử phần mềm 4

a. Khái niệm 5

Có thể định nghĩa một cách dễ hiểu như sau: *Kiểm thử phần mềm là quá trình thực thi một hệ thống phần mềm để xác định xem phần mềm có đúng với đặc tả không và môi trường hoạt động có đúng yêu cầu không.* 6

Software testing objectives 7

Các mục tiêu trực tiếp 8

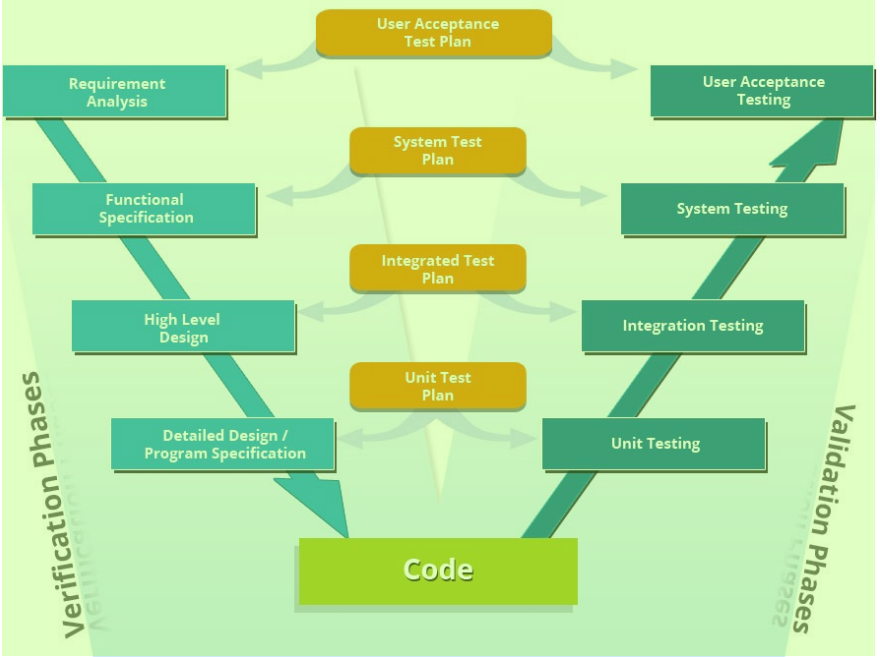
- Xác định và phát hiện nhiều lỗi nhất có thể trong phần mềm được kiểm thử 9
- Sau khi sửa chữa các lỗi đã xác định và kiểm tra lại, làm cho phần mềm đã được kiểm thử đến một mức độ chấp nhận được về chất lượng.
- Thực hiện các yêu cầu kiểm thử cần thiết một cách hiệu quả và có hiệu quả, trong phạm vi ngân sách và thời gian cho phép.

Các mục tiêu gián tiếp 10

Biên dịch một bản ghi về các lỗi phần mềm để sử dụng trong công tác phòng chống lỗi (bằng các hành động khắc phục và ngăn ngừa). 11

b. Các mức độ kiểm thử 12

Các mức độ kiểm thử được xây dựng tương ứng với từng pha trong vòng đời phát triển phần mềm. 13



1

Hình 2.2: Các mức độ kiểm thử 2

Kiểm thử đơn vị là hoạt động kiểm thử nhỏ nhất. Kiểm thử thực hiện trên các hàm hay thành phần riêng lẻ, cần hiểu biết về thiết kế chương trình và code. Kiểm thử đơn vị được thực hiện bởi Lập trình viên (không phải kiểm thử viên) 3

Mục đích: Đảm bảo thông tin được xử lý đúng và có đầu ra chính xác trong mối tương quan giữa dữ liệu nhập và chức năng của đơn vị. 4

Kiểm thử tích hợp: nhằm phát hiện lỗi giao tiếp xảy ra giữa các thành phần cũng như lỗi của bản thân từng thành phần (nếu có). *Thành phần* có thể là: các module, các ứng dụng riêng lẻ, các ứng dụng client/server trên một mạng 5

Kiểm thử hệ thống: là một mức của tiến trình kiểm thử phần mềm khi các module và tích hợp các module đã được test. 6

Mục tiêu của kiểm thử hệ thống là để đánh giá phần mềm có tuân thủ theo các yêu cầu đã đưa ra không, gồm cả yêu cầu chức năng và yêu cầu phi chức năng. Vì vậy, System Test lại gồm nhiều loại kiểm thử khác nhau, cả kiểm thử chức năng và kiểm thử phi chức năng. 7

Kiểm thử chấp nhận: là một cấp độ trong tiến trình kiểm thử phần mềm nhằm kiểm thử hệ thống về khả năng chấp nhận được. Mục tiêu của kiểm thử này là để đánh giá sự tuân thủ của hệ thống với các yêu cầu nghiệp vụ và thẩm định xem đã có thể chấp nhận để bàn giao chưa. Kiểm thử chấp nhận được khách hàng thực hiện (hoặc ủy quyền cho một nhóm thứ ba thực hiện). 8

2.3. CÁC KỸ THUẬT RÀ SOÁT 1

2.3.1. Mục tiêu của rà soát 2

Mục tiêu của rà soát
Định nghĩa

Định nghĩa của IEEE (1990), *quá trình rà soát* là: 3

"Một quá trình hoặc một cuộc họp mà trong đó một sản phẩm công việc hoặc một tập các sản phẩm công việc được đưa ra tới toàn thể cá nhân tham gia vào dự án, các giám đốc, người dùng, khách hàng và các bên quan tâm đến dự án nhằm lấy ý kiến phê bình và phê chuẩn" 4

Một số phương pháp dùng để xem xét lại tài liệu: 5

- Xem xét lại thiết kế hình thức (Formal design reviews) 6
- Xem xét lại ngang hàng (Peer reviews)
- Ý kiến chuyên gia

Mục tiêu 7

Mục đích được chia làm 2 loại: mục đích trực tiếp và gián tiếp. 8

Mục đích trực tiếp: 9

- Phát hiện lỗi phân tích và thiết kế. 10
- Xác định các rủi ro mới.
- Xác định sự sai lệch so với mẫu, các kiểu thủ tục và qui ước.
- Để phê chuẩn sản phẩm của phân tích hoặc thiết kế.

Mục đích gián tiếp: 11

- Nơi họp mặt không chính thức để trao đổi về những kiến thức chuyên môn. 12
- Ghi lại những lỗi phân tích và thiết kế sẽ hỗ trợ một cơ sở cho những hoạt động sửa chữa lỗi trong tương lai.

2.3.2. Các hình thức rà soát 13

a. Rà soát chính thức 14

Ví dụ một số loại rà soát chính thức : 15

- DPR – Development Plan Review : Review kế hoạch phát triển 16
- SRSR – Software Requirement Specification Review : Review đặc tả yêu cầu phần mềm

- PDR – Preliminary Design Review : Review thiết kế sơ bộ 1
- DBDR- Detailed Design Review : Review thiết kế chi tiết
- TPR – Test Plan Review : Review kế hoạch kiểm thử
- STPR – Software Test Procedure Review : Review thủ tục kiểm thử phần mềm
- VDR- Version Description Review : Review mô tả phiên bản
- OMR- Operator Manual Review : Review vận hành thủ công
- SMR- Support Manual Review :Review trợ giúp thủ công
- TRR- Test Readiness Review : Review sự sẵn sàng kiểm thử
- PRR- Product Release Review : Review bản phát hành sản phẩm
- IPR-Installation Plan Review : Review kế hoạch cài đặt

Các nhân tố ảnh hưởng tới DRs 2

- Những người tham gia 3
- Sự chuẩn bị trước
- Phiên DR
- Các hoạt động sau DR được đề xuất

Những người tham gia rà soát thiết kế 4

Gồm có Review leader và review team. 5

Review Leader : 6

- Có kiến thức và kinh nghiệm trong việc phát triển kiểu dự án được review 7
- Có thâm niên ở mức độ bằng với hoặc cao hơn của project leader
- Có mối quan hệ tốt với project leader và đội dự an
- Có vị trí bên ngoài đội dự án

Review team: 8

- Phần lớn không thuộc đội dự án 9
- Kích thước từ 3-5 người
- Đa dạng về kinh nghiệm và phương pháp

Sự chuẩn bị cho một phiên làm việc DR 10

Được hoàn thành bởi 3 thành viên: review leader, review team và development team.. 11

Chuẩn bị của Review leader 1

- Bổ nhiệm các thành viên nhóm 2
- Lập lịch các phiên review
- Phân chia tài liệu thiết kế cho các thành viên của nhóm

Chuẩn bị của Review team 3

- Xem lại tài liệu thiết kế 4
- Danh sách bình luận

Chuẩn bị của Development team 5

- Trình diễn ngắn tài liệu thiết kế 6
- Checklist các công việc review

Phiên DR 7

Một cuộc họp phiên DR thông thường gồm có : 8

1. Trình diễn ngắn gọn về tài liệu thiết kế 9
2. Các bình luận của các thành viên review team
3. Kiểm tra và xác nhận thảo luận mỗi bình luận
4. Các quyết định về tài liệu thiết kế để xác định tiến trình dự án. Các quyết định có thể có 3 loại :

- Phê duyệt đầy đủ 10
- Phê duyệt từng phần
- Từ chối phê duyệt

Các hoạt động hậu review 11

- Báo cáo Review 12
- Review leader thực hiện sau phiên review 13
- Bao gồm : 14
    - Tổng kết các thảo luận review 15
    - Quyết định về sự tiếp tục của dự án
    - Danh sách các hoạt động cần thiết phải làm
    - Tên thành viên chịu trách nhiệm theo sát việc hiệu chỉnh

- Tiến trình theo dõi 1
    - Review leader thực hiện
    - Chắc rằng việc hiệu chỉnh được thực hiện đúng đắn

Việc theo dõi cần được ghi lại

b. Rà soát ngang hàng 2

Mục đích chính của peer review là xác định lỗi và độ lệch dựa vào các chuẩn. 3

Có hai phương pháp peer reviews: 4

- xét duyệt (inspection) 5
- kiểm tra từng bước (walkthrough).

Walkthrough phát hiện sai sót và ghi chú lên tài liệu. 6

Inspection phát hiện sai sót và kết hợp với nỗ lực để cải tiến. 7

Những người tham gia vào peer reviews 8

Một đội peer review tối ưu 3-5 người tham gia. 9

Tất cả những người tham gia nên là những người cùng địa vị của nhà thiết kế hệ thống phần mềm. 10

Một đội peer review đề cử bao gồm: 11

- Một leader review. 12
- Một người thực thi (author).
- Các chuyên gia đặc biệt (specialized professionals).

Phân công trách nhiệm trong đội (team assignments) 13

Hai trong số các thành viên sẽ là: 14

- một người dẫn chương trình 15
- một người viết tài liệu trong cuộc thảo luận.

Chuẩn bị cho phiên peer review 16

- ***Leader:*** 17
    - Xác định những đoạn trong tài liệu thiết kế sẽ đuợc review 18
    - Lựa chọn thành viên nhóm
    - Lập lịch cho những phiên review

- Đưa tài liệu cho các thành viên trong đội trước phiên review 1

- ***Đội peer review**: yêu cầu của inspection khá tỉ mỉ, còn walkthrough chỉ yêu cầu đơn giản.* 2
  - Inspection: Đọc & liệt kê chú thích của họ 3
  - Walkth-rough : Đọc các đoạn sẽ được review

Phiên peer review 4

Inspection : 5

- Presenter đọc một đoạn tài liệu và thêm vào nếu cần thiết. 6
- Những người liên quan hoặc đưa ra chú thích, hoặc phản ứng với những lời chú thích trong tài liệu.

Walkthrough: 7

- Bắt đầu bằng sự trình bày ngắn của author (thường ko phải là presenter) hoặc tổng quan về dự án và những đoạn thiết kế sẽ được review. 8
- Ghi lại vị trí, mô tả, kiểu, đặc điểm (sai sót, những phần thiếu, những phần thêm vào) của mỗi lỗi được chấp nhận.

Quy tắc thời gian: phiên không nên vượt quá 2 giờ, hoặc lập lịch không nhiều hơn 2 ngày. 9

Tài liệu sau mỗi phiên review: 10

- Báo cáo những phát hiện trong phiên inspection 11
- Báo cáo tóm tắt của phiên inspection

Các hoạt động sau peer review (post-peer review activities) 12

Inspection: 13

- Nhắc nhở, sửa chữa hiệu quả, làm lại tất cả các lỗi 14
- Chuyển giao các bản báo cáo inspection tới CAB để phân tích.

Hiệu quả của peer review (the efficency of peer reviews) 15

Một vài độ đo phổ biến ước lượng hiệu suất của peer review: 16

- Số giờ trung bình trên một lỗi. 17
- Mật độ phát hiện thiếu sót (Số thiếu sót trung bình trên một trang tài liệu thiết kế).
- Hiệu năng peer review bên trong.
- Peer review coverage: Là tỉ lệ nhỏ của tài liệu và toàn bộ code đã từng trải qua peer review

c. Ý kiến chuyên gia 18

Ý kiến của chuyên ra rất hữu ích trong những trường hợp sau: 1

- Thiếu sự hiểu biết đầy đủ về lĩnh vực nào đó. 2
- Tạm thời thiếu những người chuyên nghiệp để tham gia vào đội xem xét lại
- Các thành viên chuyên nghiệp cao cấp trong tổ chức không thống nhất được với nhau.
- Trong các tổ chức nhỏ số lượng ứng viên phù hợp cho đội xem xét lại là không đủ.

d. So sánh rà soát chính thức và rà soát ngang hàng 3

Để rõ hơn những đặc điểm của các phương pháp rà soát có thể tham khảo bảng so sánh giữa ba phương pháp rà soát sau đây: 4

Bảng 2.1: So sánh các hình thức rà soát 5

6

Thuộc tính	Formal design reviews	Inspections	Walkthroughs
Mục đích chính trực tiếp	Phát hiện lỗi Xác định rủi ro mới Phê chuẩn tài liệu thiết kế	Phát hiện lỗi Xác định sự sai lệch so với tiêu chuẩn	Phát hiện lỗi
Mục đích chính gián tiếp	Trao đổi kiến thức	Trao đổi kiến thức Hỗ trợ hoạt động sửa chữa	Trao đổi kiến thức
Lãnh đạo việc xem xét lại	Người đứng đầu kỹ nghệ phần mềm hoặc là thành viên cao cấp	Người điều hành chỉ đạo (ngang hàng)	Người tổ chức
Người tham gia	Các thành viên cấp cao và đại diện khách hàng	Những người ngang hàng	Những người ngang hàng
Sự tham gia của trưởng dự án	Có	Có	Có, thông thường là khi bắt đầu xem xét lại
Thành viên chuyên gia trong đội		Người thiết kế Lập trình viên - Người kiểm thử	Người giám sát tiêu chuẩn - Chuyên gia bảo trì - Đại diện người dùng

Quá trình xem xét lại			
Họp tổng quan	Không	Có	Có
Sự chuẩn bị của những người tham gia	Có – chuẩn bị kỹ lưỡng	Có – chuẩn bị kỹ lưỡng	Có – chuẩn bị tóm lược
Phiên xem xét lại	Có	Có	Có
Bám sát việc sửa chữa	Có	Có	Có
Cơ sở hạ tầng			
Đào tạo chính thức cho người tham gia	Không	Có	Không
Sử dụng checklist	Không	Có	Không
Lỗi liên quan đế thu thập dữ liệu	Không yêu cầu hình thức	Yêu cầu hình thức	Không yêu cầu hình thức
Tài liệu hóa việc xem xét lại	Báo cáo xem xét lại thiết kế hình thức	Báo cáo đánh giá phiên thanh tra Báo cáo tổng kết phiên thanh tra	Báo cáo đánh giá phiên walkthrough

1

2.4. CÁC KỸ THUẬT KIỂM THỬ 2

2.4.1 Kỹ thuật kiểm thử hộp đen 3

Khái niệm: 4

Black-box testing là phương pháp kiểm thử mà không cần biết cài đặt của chương trình. Kiểm thử hộp đen quan trọng trong công nghiệp 5

Chúng ta cần có một bản chương trình chạy được và đặc tả. 6

Test cases được công thức hoá là một cặp. Ví dụ, (input, output mong muốn) 7

Một số kỹ thuật thiết kế test: phân lớp tương đương, test biên, phân loại, kiểm thử theo cặp… 8

Kỹ thuật Phân lớp tương đương: 9

Phân lớp tương đương là một phương pháp kiểm thử hộp đen chia miền đầu vào của chương trình thành các lớp dữ liệu, từ đó suy dẫn ra các ca kiểm thử. Lớp tương đương biểu thị cho tập các trạng thái hợp lệ hay không hợp lệ đối với điều kiện vào. 10

Thiết kế Test-case bằng phân lớp tương đương tiến hành theo 2 bước: 11

1. Xác định các lớp tương đương. 1
2. Xác định các ca kiểm thử. 2

Kỹ thuật Phân tích giá trị biên: 3

Kinh nghiệm cho thấy các ca kiểm thử mà khảo sát tỷ mỷ các điều kiện biên có tỷ lệ phần trăm cao hơn các ca kiểm thử khác. Các điều kiện biên là những điều kiện mà các tình huống ngay tại, trên và dưới các cạnh của các lớp tương đương đầu vào và các lớp tương đương đầu ra. Phân tích các giá trị biên là phương pháp thiết kế ca kiểm thử bổ sung thêm cho phân lớp tương đương, nhưng khác với phân lớp tương đương ở 2 khía cạnh: 4

- Phân tích giá trị biên không lựa chọn phần tử bất kỳ nào trong 1 lớp tương đương là điển hình, mà nó yêu cầu là 1 hay nhiều phần tử được lựa chọn như vậy mà mỗi cạnh của lớp tương đương đó chính là đối tượng kiểm tra. 5
- Ngoài việc chỉ tập trung chú ý vào các trạng thái đầu vào (không gian đầu vào), các ca kiểm thử cũng nhận được bằng việc xem xét không gian kết quả (các lớp tương đương đầu ra).

Kỹ thuật Bảng quyết định 6

- Miêu tả các qui tắc nghiệp vụ phức tạp mà phần mềm phải thực hiện dưới dạng dễ đọc và dễ kiểm soát 7
- Ví dụ 1 chức năng nhỏ của công ty bảo hiểm : khuyến mãi cho những chủ xe nếu họ thỏa ít nhất 1 trong 2 điều kiện: đã lập gia đình / là sinh viên giỏi. Mỗi dữ liệu nhập là 1 giá trị luận lý, nên bảng quyết định chỉ cần có 4 cột, miêu tả 4 luật khác nhau : 8

	Rule 1	**Rule 2**	**Rule 3**	**Rule 4**
Conditions				
Married?	Yes	Yes	No	No
Good Student?	Yes	No	Yes	No
Actions Discount ($)	60	25	50	0

9

Từ bảng quyết định chuyển thành bảng các testcase trong đó mỗi cột miêu tả 1 luật được chuyển thành 1 đến n cột miêu tả các testcase tương ứng với luật đó. 10

Lược đồ chuyển trạng 11

- Là 1 công cụ rất hữu ích để đặc tả các yêu cầu phần mềm hoặc để đặc tả bảng thiết kế hệ thống phần mềm. 12

- Lược đồ chuyển trạng thái ghi nhận các sự kiện xảy ra, rồi được hệ thống xử lý cũng như những đáp ứng của hệ thống. 1
- Khi hệ thống phải nhớ trạng thái trước đó của mình, hay phải biết trình tự các hoạt động nào là hợp lệ, trình tự nào là không hợp lệ thì lược đồ chuyển trạng thái là rất thích hợp.

Kiểm thử theo cặp 2

Thực tế cho thấy hầu hết các lỗi đều được sinh ra từ sự kết hợp giá trị của các cặp tham số đầu vào. 3

Để thực hiện kiểm thử theo cặp, ta cần: 4

- Lựa chọn tham số đầu vào và các giá trị tương ứng 5
- Lấy tổ hợp (pairwise) của các giá trị giữa 2 tham số
- Xây dựng bộ test sao cho bao phủ được tất cả các cặp xác định ở trên

2.4.2 Kỹ thuật kiểm thử hộp trắng 6

Khái niệm: 7

- Kiểm thử hộp trắng dựa vào thuật giải cụ thể, vào cấu trúc dữ liệu bên trong của module cần kiểm thử để xác định module đó có thực hiện đúng không. 8
- Do đó người Kiểm thử hộp trắng phải có kỹ năng, kiến thức để có thể thông hiểu chi tiết về đoạn code cần kiểm thử.
- Thường tốn rất nhiều thời gian và công sức

Kiểm thử luồng điều khiển 9

Đường thi hành (Execution path) : là 1 kịch bản thi hành đơn vị phần mềm tương ứng : danh sách có thứ tự các lệnh được thi hành ứng với 1 lần chạy cụ thể của đơn vị phần mềm, bắt đầu từ điểm nhập của đơn vị phần mềm đến điểm kết thúc của đơn vị phần mềm. 10

Mục tiêu của phương pháp kiểm thử luồng điều khiển là đảm bảo mọi đường thi hành của đơn vị phần mềm cần kiểm thử đều chạy đúng. Rất tiếc trong thực tế, công sức và thời gian để đạt mục tiêu trên đây là rất lớn, ngay cả trên những đơn vị phần mềm nhỏ. 11
Ta nên kiểm thử số test case tối thiểu mà kết quả độ tin cậy tối đa. Nhưng làm sao xác định được số test case tối thiểu nào có thể đem lại kết quả có độ tin cậy tối đa ?
Phủ kiểm thử (Coverage) : là tỉ lệ các thành phần thực sự được kiểm thử so với tổng thể sau khi đã kiểm thử các test case được chọn. Phủ càng lớn thì độ tin cậy càng cao.
Thà nh phần liên quan có thể là lệnh, điểm quyết định, điều kiện con, đường thi hành hay là sự kết hợp của chúng.

Các mức phủ kiểm thử: 12

- Phủ cấp 0 : kiểm thử những gì có thể kiểm thử được, phần còn lại để người dùng phát hiện và báo lại sau. Đây là mức độ kiểm thử không thực sự có trách nhiệm. 1
- Phủ cấp 1 : kiểm thử sao cho mỗi lệnh được thực thi ít nhất 1 lần.
- Phủ cấp 2 : kiểm thử sao cho mỗi điểm quyết định đều được thực hiện ít nhất 1 lần cho trường hợp TRUE lẫn FALSE. Ta gọi mức kiểm thử này là phủ các nhánh (Branch coverage). Phủ các nhánh đảm bảo phủ các lệnh.
- Phủ cấp 3 : kiểm thử sao cho mỗi điều kiện luận lý con (subcondition) của từng điểm quyết định đều được thực hiện ít nhất 1 lần cho trường hợp TRUE lẫn FALSE. Ta gọi mức kiểm thử này là phủ các điều kiện con (subcondition coverage). Phủ các điều kiện con chưa chắc đảm bảo phủ các nhánh.
- Phủ cấp 4 : kiểm thử sao cho mỗi điều kiện luận lý con (subcondition) của từng điểm quyết định đều được thực hiện ít nhất 1 lần cho trường hợp TRUE lẫn FALSE & điểm quyết định cũng được kiểm thử cho cả 2 nhánh. Ta gọi mức kiểm thử này là phủ các nhánh & điều kiện con (branch & subcondition coverage).

2.5 CÂU HỎI ÔN TẬP 2

1. Khái niệm Đảm bảo chất lượng phần mềm? 3
2. Khái niệm V&V và IV&V? 4
3. Xác định đặc tả theo bộ tiêu chí MCCall? Lấy ví dụ minh hoạ? 5
4. Phân biệt các loại hoạt động review? 6
5. Nêu các level kiểm thử 7
6. Trình bày các kỹ thuật kiểm thử? 8

CHƯƠNG 3: THU THẬP VÀ PHÂN TÍCH YÊU CẦU 1

3.1 THU THẬP YÊU CẦU 2

3.1.1 Tìm hiểu lĩnh vực chuyên môn 3

Tóm tắt các bước thực hiện: 4

- Bước 1: Làm việc nhóm (có thể dùng brainstorming) để liệt kê tất cả các từ khóa chuyên môn trong lĩnh vực của ứng dụng 5
- Bước 2: Phân các từ tìm ra thành nhóm cho dễ theo dõi
- Bước 3: Giải thích nghĩa của từng từ khóa và lập bảng kết quả các từ khóa (glosary list).

Áp dụng các bước trên vào bài toán quản lí đăng kí học tín chỉ của một trường đại học, gộp bước 1 và 2, có thể gom các từ khóa trong lĩnh vực quản lí đăng kí tín chỉ thành ba nhóm như sau: 6

Con người 7	*Hoạt động của con người* 8	*Vật, đối tượng* 9
Thành viên 10	Đăng nhập 11	Trường 12
Nhân viên	Đăng xuất	Khoa
Sinh viên	Đổi mật khẩu	Ngành học
Nhân viên quản lí	Cập nhật thông tin cá nhân	Bộ môn
Nhân viên khảo thí	Sinh viên đăng kí học	Năm học
Nhân viên giáo vụ	Sinh viên xem lịch học	Khóa học
Giảng viên	Sinh viên sửa đăng kí	Học kì
Người dùng	Sinh viên xem điểm	Môn học
....	Giảng viên đăng kí dạy	Lớp học phần
	Giảng viên nhập điểm	Tuần học
	Giảng viên xem lịch dạy	Ngày học trong tuần
	Giảng viên xem điểm	Kíp/ca học trong ngày
	Giáo vụ quản lí môn học	Phòng học
	Giáo vụ quản lí lớp học phần	Tòa nhà
	Giáo vụ quản lí thông tin sinh viên	Đầu điểm thành phần
		Điểm thi
	Giáo vụ quản lí thông tin học kì	Điểm trung bình môn
	Quản lí thông tin trường	Điểm chữ
	Quản lí thông tin Khoa	Tín chỉ
	Quản lí thông tin bô môn	Tổng tín chỉ tích lũy
	Quản lí thông tin giảng viên	Điểm trung bình tích lũy
	Xem thống kê giảng viên	Học lực
	Xem thống kê sinh viên	Loại học lực
	Xem thống kê học lực	....
	Xem thống kê phòng học	

Xem thống kê theo khoa 1
....

Bước 3, lập bảng giải thích ngữ nghĩa chi tiết cho từng khái niệm trong lĩnh vực của ứng dụng: 2

TT	Tên Tiếng Việt	Tiếng Anh	Giải thích
Nhóm các khái niệm liên quan đến con người			
1	Thành viên	Member	Người có tài khoản và thẩm quyền để đăng nhập vào hệ thống, thực hiện các chức năng được cho phép. Ví dụ sinh viên có thể đăng nhập vào hệ thống để đăng kí học, xem lịch học, xem điểm. Thông tin sinh viên được quản lí trong hệ thống. Nên sinh viên là một thành viên của hệ thống
...			
Nhóm các khái niệm liên quan đến hoạt động của người			
22	Đăng kí học	Registration	Là hoạt động của sinh viên trước mỗi kì học. Theo đó, trước mỗi kì học, trường sẽ mở chế độ đăng kí cho sinh viên vào đăng kí trong một thời gian nhất định. Trong thời gian đó, sinh viên phải đăng nhập vào hệ thống, thực hiện chức năng đăng kí học. Chức năng này cho phép sinh viên chọn môn học được phép đăng kí trong kì học, chọn lớp học phần phù hợp với bản thân. Thường có quy định mỗi học kì phải học tối thiểu bao nhiêu tín chỉ, tối đa bao nhiêu tín chỉ. Sinh viên phải chọn đủ số tín chỉ trong ngưỡng này mới hợp lệ. Trong thời gian trường còn mở đăng kí, sinh viên có quyền vào lại hệ thống thay đổi thông tin đăng kí của mình.
...			
Nhóm các khái niệm liên quan đến vật, đối tượng xử lí			
33	Điểm trung bình môn	Final grade	Điểm tổng kết môn học. Tính trung bình theo các đầu điểm thành phần của môn học. Ví dụ, môn Toán có 3 đầu điểm thành phần: chuyên cần 10%, trung bình kiểm tra 10%, trung bình bài tập lớn 20%. Và điểm bài thi cuối kì 60%. Nếu sinh viên A đạt các đầu điểm lần lượt là 5, 6, 7, 8 thì điểm trung bình môn được tính là

			5*0.1 + 6*0.1 + 7*0.2 + 8*0.6 = 7.3 hệ 10. Tương ứng điểm chữ B. Tương ứng điểm hệ 4 là 3.0
...			

1

3.1.2 Mô tả hệ thống bằng ngôn ngữ tự nhiên 2

Tóm tắt các bước thực hiện: 3

- Bước 1: Giới thiệu mục đích hệ thống
- Bước 2: Phạm vi hệ thống: ai được dùng phần mềm/hệ thống? Mỗi người vào vào hệ thống được phép thực hiện các chức năng nào?
- Bước 3: Với mỗi chức năng mà người dùng được phép thực hiện ở bước 2, mô tả chi tiết hoạt động nghiệp vụ của chức năng đấy diễn ra như thế nào?
- Bước 4: Các đối tượng nào được quản lí, xử lí trong hệ thống? Mỗi đối tượng cần dùng/quản lí các thuộc tính nào?
- Bước 5: Quan hệ (số lượng) giữa các đối tượng đã nêu ở bước 4?

4

Áp dụng vào hệ thống quản lí đăng kí tín chỉ, các bước được tiến hành như sau: 5

Bước 1: Mục đích của hệ thống: Hệ thống trang web phục vụ công tác quản lí đăng kí tín chỉ của sinh viên, đăng kí giảng dạy của giảng viên, quản lí điểm các môn học của một trường đại học. 6

Bước 2: Phạm vi hệ thống: Những người được vào hệ thống và chức năng mỗi người được thực hiện khi vào hệ thống này được quy định như sau: 7

- Thành viên hệ thống: 8
  - Đăng nhập
  - Đăng xuất
  - Đổi mật khẩu cá nhân
- Sinh viên: 9
  - Được thực hiện các chức năng như thành viên 10
  - Đăng kí học, sửa thông tin đăng kí của mình
  - Xem lịch học của mình
  - Xem điểm của mình
- Giảng viên: 11
  - Đăng kí dạy, sửa thông tin đăng kí dạy 12
  - Nhập, sửa điểm các lớp mình dạy
  - Xem lịch dạy của mình
  - Xem thống kê liên quan đến các lớp mình dạy

- Nhân viên giáo vụ: 1
  - Quản lí thông tin sinh viên: thêm, xóa, sửa theo yêu cầu từ sinh viên 2
  - Quản lí thông tin môn học
  - Quản lí thông tin lớp học phần
- Nhân viên quản lí: 3
  - Quản lí thông tin chung: thông tin trường, thông tin khoa, thông tin bộ môn, thông tin các toàn nhà, phòng học, thông tin năm học, kì học. 4
  - Quản lí thông tin giảng viên theo yêu cầu từ giảng viên
  - Xem các loại thống kê
- Nhân viên khảo thí: 5
  - Xuất bảng điểm theo yêu cầu của sinh viên

Những chức năng không đề cập đến thì mặc định là không thuộc phạm vi của hệ thống. 6

Bước 3: Hoạt động nghiệp vụ của các chức năng: Theo nguyên tắc, mỗi chức năng liệt kê trong bước 2 đều phải mô tả chi tiết. Tuy nhiên, trong phạm vi tài liệu này, chỉ có ba chức năng được mô tả chi tiết vì đây là các chức năng được dùng để minh họa cho các bước phân tích, thiết kế từ đầu đến cuối. Các chức năng còn lại coi như bài tập cho người học. 7

- *Sinh viên đăng kí học*: Sinh viên đăng nhập vào hệ thống -> chọn chức năng đăng kí tín chỉ (đang trong thời gian mở đăng kí mới được chọn) -> chọn kì đăng kí + ngành học (có thể có sinh viên học đồng thời hai chuyên ngành) -> hệ thống hiện danh sách các môn học có thể đăng kí (mã, tên môn học, số tín chỉ, mô tả), các lớp học phần đã đăng kí rồi, nếu có -> Sinh viên chọn môn học -> hệ thống hiện danh sách các lớp học phần của môn học đấy (mã, tên, sĩ số tối đa, sĩ số hiện tại, phòng học, giảng viên, lịch học hàng tuần vào các ngày nào trong tuần, kíp nào trong ngày): chỉ active các nhóm mà không bị trùng lịch học với các môn đã chon trước, các nhóm bị trùng lịch thì chỉ xem, không chọn được -> Sinh viên chọn lớp học phần mình thích -> hệ thống quay lại trang bắt đầu đăng kí với lớp học phần vừa chọn được bổ sung vào danh sách các lớp học phần đã chọn. Sinh viên lặp lại các bước trên cho đến khi chọn đủ số tín chỉ trong ngưỡng cho phép -> nút lưu được active -> Sinh viên click lưu thì thông tin đăng kí mới chính thức được lưu vào hệ thống, hệ thống quay về giao diện chính của sinh viên. 8
- *Giảng viên nhập điểm*: Giảng viên đăng nhập vào hệ thống -> chọn chức năng nhập điểm -> Chọn học kì đang active -> hệ thống hiện danh sách các môn học do giảng viên dạy của kì đã chọn (mã, tên, số tín chỉ, mô tả) -> Giảng viên click chọn môn học muốn nhập điểm -> Hệ thống hiện danh sách các lớp học phần do giảng viên dạy (mã, tên, sĩ số thực, phòng học, ngày học, kíp học) -> Giảng viên chọn 1 lớp học phần muốn nhập -> Hệ thống hiện danh sách các sinh viên đăng kí lớp học phần được chọn với điểm thành phần, nếu có: thứ tự, mã sinh viên, họ tên, các đầu điểm thành phần, điểm thi, cột trung bình môn và điểm chữ được tự tính sau khi nhập -> Giảng viên nhập đầu điểm muốn nhập cho tất cả 9

sinh viên trong danh sách và click lưu -> hệ thống lưu điểm vào và quay về giao diện chính của giảng viên. 1

- *Quản lí xem thống kê theo loại học lực*: Nhân viên quản lí đăng nhập vào hệ thống -> chọn chức năng xem thống kê -> hệ thống hiện giao diện chọn thông tin thống kê -> chọn thống kê loại học lực -> Hệ thống hiện giao diện thống kê loại học lực -> Quản lí chọn học kì muốn thống kê -> Kết quả thống kê hiện lên, mỗi loại học lực trên một dòng, xếp theo thứ tự cao nhất đến thấp nhất của loại học lực trong bảng đánh giá (Ưu tú, xuất sắc, giỏi, khá, trung bình, yếu kém): thứ tự, loại học lực, tổng số sinh viên đạt loại đó, điểm trung bình sinh viên trong nhóm đạt loại đó trong học kì đã chọn -> Quản lí click vào một loại học lực -> Hệ thống hiện danh sách các sinh viên đạt loại học lực đấy lên, xếp theo thứ tự các ngành học, đến thứ tự abc của tên sinh viên: thứ tự, mã sinh viên, họ và tên, ngành học, khóa học, tổng số tín chỉ của học kì, điểm trung bình của học kì -> Quản lí click vào một sinh viên danh sách -> Hệ thống hiện lên danh sách các môn và kết quả của sinh viên đã học trong học kì đó, xếp theo thứ tự abc của tên môn học: thứ tự, tên môn học, số tín chỉ, điểm trung bình môn đó của sinh viên. Dòng cuối là tổng số tín chỉ, điểm trung bình cả học kì của sinh viên -> Quản lí click vào một môn học trong danh sách -> Hệ thống hiện lên điểm chi tiết của môn học của sinh viên: mã môn, tên môn, tổng tín chỉ, các đầu điểm thành phần dạng bảng: tên đầu điểm thành phần, tỉ lệ % tính của đầu điểm thành phần, điểm của sinh viên. Dòng cuối là điểm trung bình môn của sinh viên trong môn học đó. 2

Bước 4: Thông tin các đối tượng cần xử lí, quản lí: 3

Nhóm các thông tin liên quan đến con người: 4

- Thành viên: tên đăng nhập, mật khẩu, họ tên, địa chỉ, ngày sinh, email, số điện thoại 5
- Sinh viên: giống thành viên, có thêm: mã sinh viên. Theo mỗi ngành học còn có khóa học, ngành học
- Nhân viên: giống thành viên, có thêm: vị trí công việc.
- Nhân viên quản lí: giống nhân viên
- Nhân viên giáo vụ: giống nhân viên
- Nhân viên khảo thí: giống nhân viên
- Giảng viên: giống nhân viên

Nhóm các thông tin liên quan đến cơ sở vật chất: 6

- Tòa nhà: tên, mô tả 7
- Phòng học: tên, sức chứa tối đa, mô tả

Nhóm các thông tin liên quan đến đơn vị, tổ chức: 8

- Trường: tên, địa chỉ, mô tả 9
- Khoa: tên, mô tả
- Ngành học: tên, mô tả
- Bộ môn: tên, mô tả

Nhóm các thông tin liên quan đến chuyên môn, vận hành: 1

- Năm học: tên, mô tả
- Kì học: tên, mô tả
- Tuần học: tên, mô tả
- Ngày trong tuần: tên, mô tả
- Kíp học trong ngày: tên, mô tả
- Môn học: tên, số tín chỉ, mô tả
- Lớp học phần: tên, mô tả, sĩ số tối đa, sĩ số hiện tại, giảng viên dạy, phòng học, tuần nào học ngày nào, kíp nào.

2

Nhóm thông tin liên quan đến thống kê: 3

- Thống kê theo loại học lực
- Thống kê sinh viên theo kết quả học
- Thống kê các môn học theo kết quả học
- Thống kê giảng viên theo: số giờ dạy, kết quả học
- Thống kê học kì theo số sinh viên

4

Bước 5: Quan hệ giữa các đối tượng, thông tin: 5

- Một trường có nhiều khoa
- Một khoa có nhiều bộ môn
- Một khoa có nhiều ngành học
- Một bộ môn quản lí chuyên môn nhiều môn học
- Một bộ môn có nhiều giảng viên
- Một năm học có nhiều học kì
- Một học kì liên quan đến nhiều năm học. Một năm học + một học kì tạo ra một kì học (kì học # học kì).
- Một kì học có nhiều môn học
- Một môn học, vào một kì học, có nhiều lớp học phần
- Một lớp học phần có thể học vào nhiều buổi, mỗi buổi có thể liên quan đến 1 tuần khác nhau, 1 ngày khác nhau, 1 kíp khác nhau, 1 phòng học khác nhau, 1 giảng viên khác nhau.
- Một giảng viên có thể dạy nhiều môn học trong mỗi kì học
- Một môn học, trong một kì học, giảng viên có thể dạy nhiều lớp học phần khác nhau, miễn sao không trùng lịch buổi nào.
- Một lớp học phần, có thể có nhiều giảng viên dạy. Nhưng mỗi buổi học chỉ có một giảng viên dạy.
- Một môn học có nhiều đầu điểm thành phần.
- Mỗi đầu điểm thành phần, đối với mỗi môn học, có tỉ lệ % tính điểm nhất định.
- Một tuần có thể có nhiều buổi dạy/học
- Một ngày có thể có nhiều buổi học/dạy

6

- Một kíp có thể có nhiều buổi học/dạy của nhiều lớp học phần khác nhau 1
- Một phòng học có thể có nhiều lớp học phần vào học ở những buổi khác nhau.
- Một sinh viên có thể đăng kí học nhiều ngành khác nhau (tối đa 2 ngành đồng thời).
- Với mỗi ngành, sinh viên phải học một số môn nhất định, và điểm tính theo từng ngành. Các môn trùng nhau giữa các ngành thì sinh viên chỉ phải học 1 lần, qua là được.
- Mỗi sinh viên, mỗi môn học, có một diểm trung bình môn.

3.1.3 Mô tả hệ thống bằng ngôn ngữ UML - use case 2

Tóm tắt các bước thực hiện để vẽ biểu đồ use case tổng quan: 3

- Bước 1: Đề xuất các actor. Với mỗi người dùng xác định ở bước 2 của mục 3.1.2, đề xuất thành một actor tương ứng. Nếu các actor có đặc điểm gì chung, có thể đề xuất actor trừu tượng thành actor cha của các actor tương ứng. Ngoài ra, cần xem xét cần có các actor gián tiếp tác động vào để thực hiện các chức năng hay không. 4
- Bước 2: Đề xuất use case. Với mỗi chức năng xác định của mục 3.1.2, đề xuất thành một use case tương ứng
- Bước 3: Mịn hóa các use case. Nếu có ít nhất 2 use case trùng nhau, cần xem xét gộp lại thành 1. Nếu gộp lại gây hiểu nhầm về số các actor tác động vào, thì có thể dùng use case trừu tượng cho các use case giống nhau, mỗi use case con liên quan đến nhóm các actor tương ứng mà thôi.

Tóm tắt các bước thực hiện để vẽ biểu đồ use case chi tiết: 5

- Bước 1: Trích phần use case của chức năng tương ứng từ biểu đồ use case tổng quan. 6
- Bước 2: Phân rã use case chính thành các use case con: mỗi giao diện (hoặc một số giao diện) tương tác với người dùng có thể đề xuất thành một use case con.
- Bước 3: Xác định quan hệ của use case con với use case chính: generalization, include, hay extend.
- Bước 4: Gộp các use case con tương tự nhau bằng cách dùng các use case trừu tượng tổng quát hơn.

Lưu ý tránh sai sót trong biểu đồ use case: 7

- Tên use case phải là động từ chỉ hành động của actor. Không nên là động từ chỉ hành động của hệ thống. Cũng không nên là danh từ, tính từ... 8
- Mỗi use case phải có tương tác với ít nhất một actor, có thể là trực tiếp hoặc gián tiếp: phải tồn tại ít nhất một đường đi từ một actor nào đó đến use case theo hướng: quan hệ include thì theo chiều mũi tên, quan hệ extend thì ngược chiều mũi tên, quan hệ kế thừa thì gộp lại.

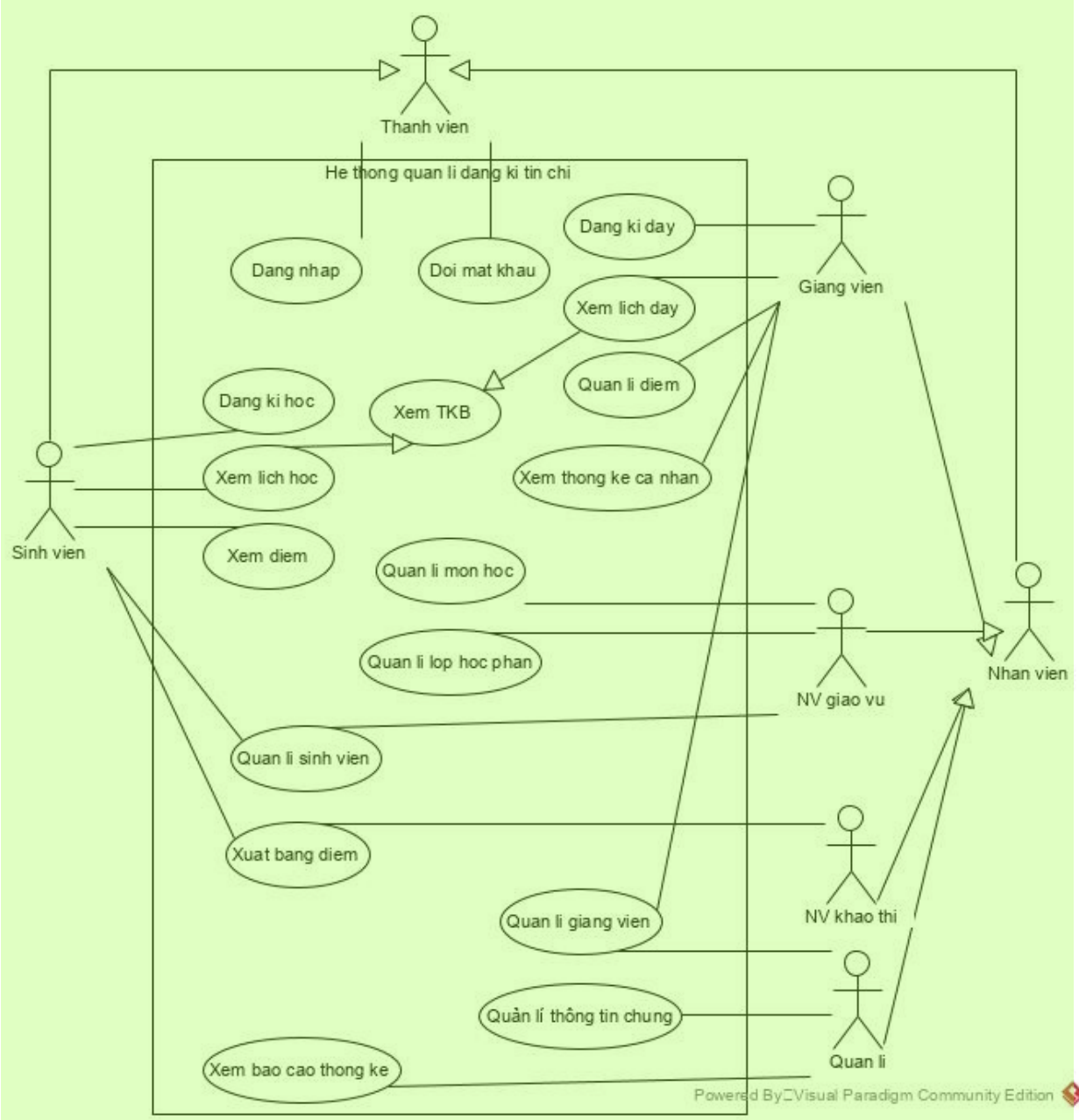
a. Biểu đồ use case tổng quan 9

Từ bước 2 trong mục 3.1.2, ta có thể đề xuất được các actor của hệ thống: sinh viên, giảng viên, quản lí, giáo vụ, và khảo thí. Tất cả đều có chức năng giống thành viên nên kế thừa từ thành viên. 10

Riêng giảng viên, quản lí, giáo vụ, khảo thí còn kế thừa từ actor nhân viên của trường. Nhân viên kế thừa trực tiếp từ thành viên. 1

Các chức năng tương ứng với từng actor: 2

- Thành viên: đăng nhập, đổi mật khẩu 3
- Sinh viên: đăng kí học, xem lịch học, xem điểm. Ngoài ra có thể tham gia gián tiếp vào các chức năng: quản lí thông tin sinh viên, xuất bảng điểm cho sinh viên.
- Giảng viên: đăng kí dạy, xem lịch dạy, nhập điểm, xem thống kê cá nhân. Ngoài ra có thể tham gia gián tiếp vào chức năng quản lí thông tin giảng viên.
- Giáo vụ: quản lí thông tin sinh viên theo yêu cầu sinh viên, quản lí môn học, lớp học phần.
- Khảo thí: xuất bảng điểm theo yêu cầu sinh viên.
- Quản lí: quản lí thông tin chung, quản lí thông tin giảng viên theo yêu cầu giảng viên, xem các loại báo cáo thống kê.



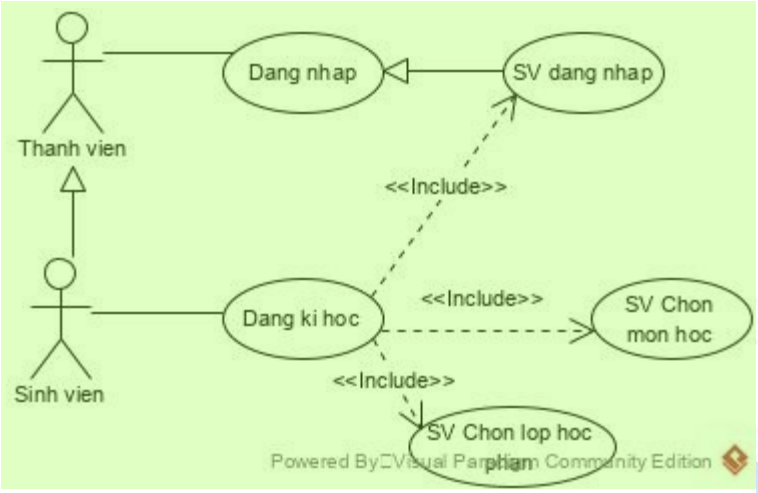
Hình 3.1: Biểu đồ use case tổng quan

Trong số các chức năng này, việc xem lịch học của sinh viên là tương tự chức năng xem lịch dạy của giảng viên. Nên hai use case này được cho kế thừa từ use case xem TKB. Như vậy, biểu đồ use case tổng quan của hệ thống được trình bày như Hình 3.1, với mô tả các use case như sau:

- Đăng kí học: UC này cho phép sinh viên vào hệ thống đăng kí các môn học theo nguyện vọng cá nhân.
- Xem lịch học: UC này cho phép sinh viên vào hệ thống xem lịch học cá nhân
- Xem điểm: UC này cho phép sinh viên vào hệ thống xem kết quả các môn học của mình.

- Đăng kí dạy: UC này cho phép giảng viên vào hệ thống để đăng kí lịch dạy của mình vào đầu mỗi kì học. 1
- Xem lịch dạy: UC này cho phép giảng viên xem lịch dạy cá nhân
- Nhập điểm: UC này cho phép giảng viên nhập điểm các lớp học phần do mình dạy
- Xem thống kê cá nhân: UC này cho phép giảng viên xem các thống kê cá nhân
- Quản lí thông tin giảng viên: UC này cho phép giáo vụ quản lí thông tin giảng viên theo yêu cầu của giảng viên tương ứng.
- Quản lí thông tin sinh viên: UC này cho phép giáo vụ quản lí thông tin sinh viên theo yêu cầu từ sinh viên tương ứng.
- Quản lí thông tin môn học: UC này cho phép giáo vụ quản lí thông tin các môn học
- Quản lí thông tin lớp học phần: UC này cho phép giáo vụ quản lí thông tin các lớp học phần.
- Xuất bảng điểm: UC này cho phép khảo thí xuất bảng điểm cho sinh viên theo yêu cầu từ sinh viên.
- Xem thống kê: UC này cho phép nhân viên quản lí xem các loại báo cáo thống kê.

b. Use chi tiết của đăng kí học 2



3

Hình 3.2: Biểu đồ use case chi tiết cho modul đăng kí học 4 5

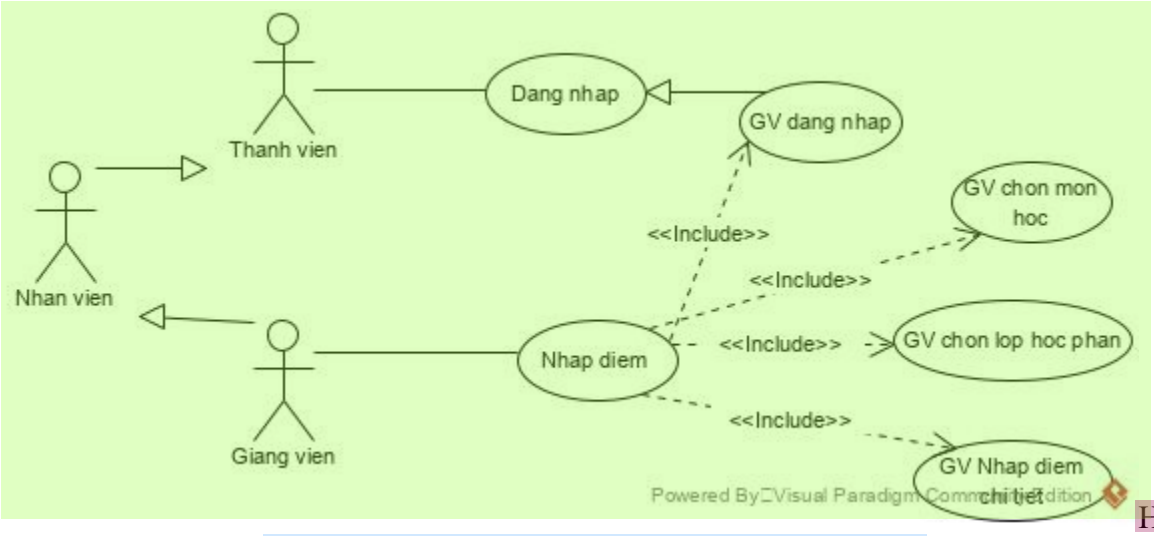
Chức năng đăng kí học có các giao diện tương tác với sinh viên: 6

- Đăng nhập -> đề xuất UC đăng nhập 7
- Đăng kí -> đề xuất UC đăng kí
- Chọn môn học -> đề xuất UC chọn môn học
- Chọn lớp học phần -> đề xuất UC chọn lớp học phần
- Đăng nhập, chọn môn học, chọn lớp học phần là bắt buộc mới hoàn thành được việc đăng kí, do đó UC đăng kí include các UC này.

Như vậy, biểu đồ UC chi tiết cho modul đăng kí học được trình bày trong Hình 3.2. Các UC được mô tả như sau:

- Chọn môn học: UC này cho phép sinh viên chọn môn học để đăng kí học
- Chọn lớp học phần: UC này cho phép sinh viên chọn lớp học phần để đăng kí học.

c. Use case chi tiết modul nhập điểm



Hình 3.3: Biểu đồ use case chi tiết cho modul nhập điểm

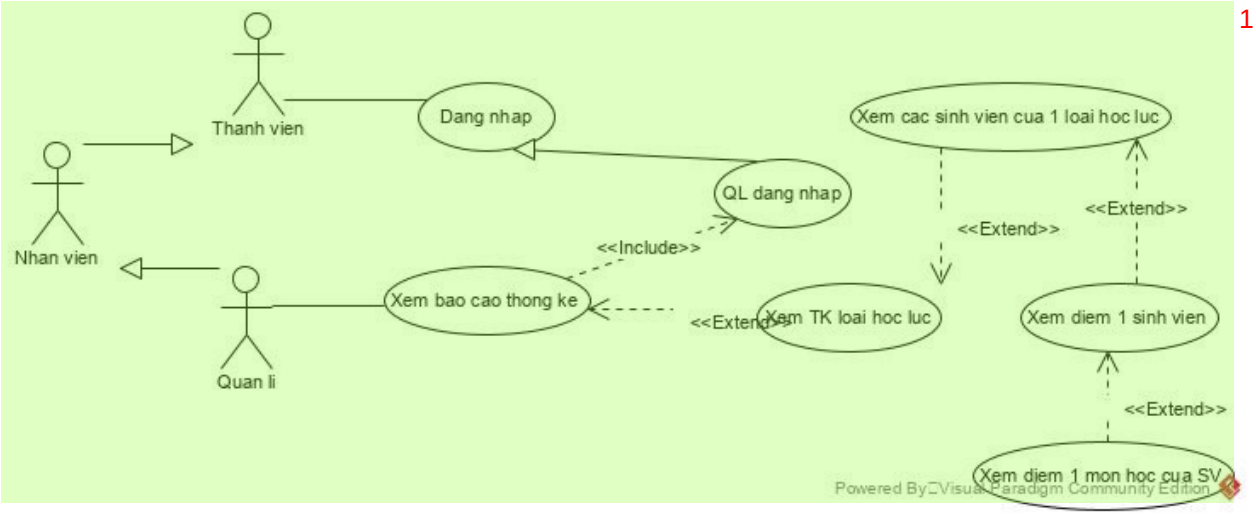
Trong chức năng nhập điểm, giảng viên phải tương tác với các giao diện:

- Đăng nhập -> thống nhất với UC đăng nhập
- Chọn kì học + môn học -> đề xuất UC chọn môn học
- Chọn lớp học phần -> đề xuất UC chọn lớp học phần
- Nhập điểm chi tiết -> đề xuất UC nhập điểm chi tiết
- Các UC trên đều bắt buộc thực hiện mới hoàn thành việc nhập điểm -> chúng đều bị chứa trong UC nhập điểm

Như vậy, biểu đồ UC chi tiết cho modul nhập điểm được trình bày trong Hình 3.3. Trong đó các UC được mô tả như sau:

- Chọn môn học: UC này cho phép giảng viên chọn môn học để nhập điểm
- Chọn lớp học phần: UC này cho phép giảng viên chọn lớp học phần để nhập điểm
- Nhập điểm chi tiết: UC cho phép giảng viên nhập/sửa điểm chi tiết từng đầu điểm thành phần của từng sinh viên trong một lớp học phần do mình dạy

d. Use case chi tiết cho modul xem thống kê loại học lực



Hình 3.4: Biểu đồ use case chi tiết cho modul xem thống kê loại học lực

Trong chức năng này, nhân viên quản lí có thể phải tương tác với các giao diện:

- Đăng nhập -> thống nhất với UC đăng nhập
- Xem thống kê loại học lực -> đề xuất UC xem TK loại học lực
- Xem thống kê các sinh viên của 1 loại học lực -> đề xuất UC xem TK sinh viên của loại học lực.
- Xem kết quả của một sinh viên -> đề xuất UC xem điểm các môn học của sinh viên
- Xêm chi tiết một môn học của sinh viên -> đề xuất UC xem điểm chi tiết 1 môn học của sinh viên.
- Các giao diện thống kê lần lượt theo dạng: giao diện sau là tùy chọn từ giao diện trước. Do đó, chúng có quan hệ mở rộng lần lượt cái sau từ cái trước.

Như vậy, biểu đồ UC chi tiết cho chức năng thống kê loại học lực được trình bày như trong Hình 3.4. Trong đó, các UC chi tiết được mô tả như sau:

- Xem TK loại học lực: UC này cho phép NVQL xem thống kê số lượng sinh viên trong mỗi loại học lực của một kì học
- Xem TK sinh viên của một loại học lực: UC này cho phép NVQL xem TK kết quả các sinh viên của một loại học lực
- Xem TK điểm của một sinh viên: UC này cho phép NVQL xem kết quả các môn học của một sinh viên trong một kì học
- Xem kết quả một một học của một sinh viên: UC này cho phép NVQL xem kết quả chi tiết một môn học của một sinh viên.

3.2 PHÂN TÍCH YÊU CẦU 1

3.2.1 Viết kịch bản 2

Lưu ý tránh sai sót trong kịch bản: 3

- Mỗi use case chính trong biểu đồ use case tổng quan thường có một kịch bản chuẩn và có thể có nhiều kịch bản ngoại lệ. 4
- Use case có bao nhiêu actor thì kịch bản cũng có bấy nhiêu actor
- Kịch bản càng chi tiết càng tốt: thông tin hệ thống hiện lên, thông tin người dùng nhập vào...

a. Kịch bản cho modul đăng kí học 5

Use case	Đăng kí học 6
Actor	Sinh viên
Tiền điều kiện	Sinh viên đã đăng nhập thành công, đang trong thời gian sinh viên được phép đăng kí môn học đầu học kì
Hậu điều kiện	Sinh viên đăng kí xong môn học cho học kì
Kịch bản chính	1. Sau khi đăng nhập, từ giao diện chính, sinh viên A chọn chức năng đăng kí học cho học kì tới. 2. Giao diện chọn kì học đăng kí, chọn ngành học hiện lên. Có danh sách các kì học đang mở đăng kí. Danh sách các ngành học mà sinh viên đang học. Nút vào đăng kí. 3. Sinh viên chọn ngành học, chọn kì học muốn đăng kí và click vào đăng kí. 4. Giao diện đăng kí học hiện lên, có ô chọn kì học muốn đăng kí; bảng danh sách các môn học/lớp học phần đã đăng kí đang rỗng; nút tiếp tục và nút lưu chưa được active. 5. Sinh viên chọn học kì tới trong danh sách và click vào nút tiếp tục. 6. Giao diện hiện lên danh sách các môn học mà sinh viên có thể đăng kí trong học kì:

7

TT	*Mã*	*Tên môn học*	*số tín chỉ*	*mô tả*	*Chọn*
1	INT17	Toán rời rạc 1	3		chọn
2	INT55	Cơ sở dữ liệu	3		chọn
3	INT68	Lập trình hướng đối tượng	3		chọn

7. Sinh viên click chọn môn Lập trình hướng đối tượng 8
8. Giao diện các lớp học phần hiện ra: 9

<table>
<tr><td></td><td>
<table>
<tr><th>TT</th><th>Mã</th><th>Tên</th><th>sĩ số tối đa</th><th>sĩ số hiện tại</th><th>Lịch học</th><th>Chọn</th></tr>
<tr><td>1</td><td>hdt1</td><td>HDT-N1</td><td>60</td><td>11</td><td>xem</td><td>chọn</td></tr>
<tr><td>2</td><td>hdt2</td><td>HDT-N2</td><td>60</td><td>19</td><td>xem</td><td>chọn</td></tr>
<tr><td>3</td><td>hdt3</td><td>HDT-N3</td><td>60</td><td>35</td><td>xem</td><td>chọn</td></tr>
</table>
9. Sinh viên lick nhóm số 1
10. Hệ thống quay lại giao diện bước 2, bảng các môn đã chọn cập nhật lại:
<table>
<tr><th>TT</th><th>Mã</th><th>Tên</th><th>sĩ số tối đa</th><th>sĩ số hiện tại</th><th>Lịch học</th><th>sửa</th><th>xóa</th></tr>
<tr><td>1</td><td>hdt1</td><td>HDT-N1</td><td>60</td><td>11</td><td>xem</td><td>sửa</td><td>xóa</td></tr>
</table>
(Lặp lại các bước 4-10 cho đến khi đạt số tín chỉ tối thiểu, nút lưu sẽ active)
11. Sinh viên click vào nút lưu.
12. Hệ thống báo thành công và quay về giao diện chính của sinh viên.
</td></tr>
<tr><td>Ngoại lệ</td><td>6. Không có môn nào để đăng kí
8. Các lớp học phần đều hết slot hoặc trùng giờ học với các môn đã chọn trước nên không chọn được.</td></tr>
</table>

b. Kịch bản cho modul nhập điểm

Use case	Nhập điểm
Actor	Giảng viên
Tiền điều kiện	Giảng viên đã đăng nhập thành công, đã có sinh viên đăng kí vào các lớp do giảng viên dạy
Hậu điều kiện	Giảng viên nhập xong điểm cho lớp học phần
Kịch bản chính	1. Giảng viên chọn chức năng nhập điểm từu giao diện chính sau khi đăng nhập 2. Giao diện chọn kì học đang học hiện lên, có danh sách các kì học đang học (active) để chọn. Nút tiếp tục. 3. Giảng viên chọn 1 kì học muốn nhập điểm và click tiếp tục. 4. Giao diện chọn môn học hiện lên: danh sách các môn học do giảng viên dạy kì tương ứng: <table><tr><th>TT</th><th>Mã</th><th>Tên môn học</th><th>số tín chỉ</th><th>mô tả</th></tr><tr><td>1</td><td>INT68</td><td>Lập trình hướng đối tượng</td><td>3</td><td></td></tr></table>5. Giảng viên click chọn môn Lập trình hướng đối tượng.

6. Giao diện các lớp học phần do giảng viên dạy hiện lên: 1

TT	*Mã*	*Tên*	*sĩ số hiện tại*
1	hdt1	HDT-N1	5
2	hdt2	HDT-N2	19
3	hdt3	HDT-N3	35

2

7. Giảng viên click vào nhóm số 1 3

8. Giao diện nhập điểm chi tiết cho nhóm số 1 hiện lên: có thông tin nhóm số 1, môn học lập trình hướng đối tượng, và danh sách điểm sinh viên, dưới cùng là nút lưu: 4

TT	*Mã sv*	*Họ và tên*	*CC*	*TBKT*	*BTL*	*Thi*	*TB chung*	*TB chữ*
1	cn11	Cao văn A	5	5	5			
2	cn22	Lê Thị C	6	6	6			
3	cn33	Trần Minh D	7	7	7			
4	cn44	Sầm Ngọc S	8	8	8			
5	cn55	Vũ Tiến V	9	9	9			

5

9. Giảng viên nhập đầu điểm thi: 6

A: 5, C:6, D:7, S:8, V:9 và click lưu 7

10. Hệ thống báo thành công và quay về giao diện chính của giảng viên. 8

Ngoại lệ

4. Không có môn học nào hiện lên
6. Chọn môn học xong, không có lớp học phần nào hiện lên
8. Chọn lớp xong, không có sinh viên nào hiện lên.

c. Kịch bản cho modul xem thống kê loại học lực 9

Use case	Xem thống kê loại học lực
Actor	Nhân viên quản lí
Tiền điều kiện	Nhân viên quản lí đã đăng nhập thành công, kì học đã có điểm của tất cả các môn học
Hậu điều kiện	

10

Kịch bản chính 1

1. Quản lí A chọn chức năng xem thống kê từ giao diện chính sau khi đăng nhập 2
2. Giao diện chọn loại thống kê và kì học hiện lên. Có danh sách các loại thống kê: loại học lực, sinh viên, môn học... và anh sách các kì học đã kết thúc.
3. Quản lí chọn chức năng thống kê loại học lực, thống kê học kì 1 năm học 2019-2020
4. Giao diện thống kê loại học lực, danh sách thống kê hiện lên:

TT	***Loại học lực***	***Tổng sinh viên***	***Điểm trung bình***
1	Ưu tú	0	
2	Xuất sắc	2	3.6
3	Giỏi	20	3.4
4	Khá	500	3.0
5	Trung bình	1200	2.67
6	Yếu kém	200	1.88

3

5. Quản lí click vào dòng Xuất sắc 4
6. Giao diện thống kê các sinh viên xuất sắc trong kì kiện lên: 5

TT	***Họ tên***	***Ngành học***	***Tổng tín chỉ***	***Điểm trung bình***
1	Đoàn Văn H	CNTT	18	3.6
2	Cao Ngọc X	CNTT	18	3.6

6

7. Quản lí click vào sinh viên Đoàn văn H 7
8. Giao diện kết quả các môn học của sinh viên Đoàn văn H hiện lên: 8

TT	Mã	Môn học	Số TC	Điểm
1	INT17	Toán rời rạc 1	3	A+
2	INT55	Cơ sở dữ liệu	3	A+
3	INT68	Lập trình hướng đối tượng	3	A+
4	INT25	Kỹ thuật đồ họa	3	A+
5	BAS15	Kinh tế chính trị Mác Lê Nin	3	A+
6	BAS44	Phương pháp nghiên cứu khoa học	3	A+

9

9. Quản lí click vào môn Lập trình hướng đối tượng 10
10. Giao diện kết quả môn Lập trình hướng đối tượng của Đoàn Văn H hiện lên:

	Môn học	Lập trình hướng đối tượng
	Điểm chuyên cần (10%)	10
	Điểm TBKT (10%)	9
	Điểm BTL (20%)	9.5
	Điểm thi (60%)	9
	Trung bình môn	9.2
	Điểm chữ	A+
	Điểm hệ 4	3.6
Ngoại lệ	4. Các loại học lực đều có 0 sinh viên	

1

3.2.2 Trích lớp thực thể 2

Tóm tắt các bước trong phương pháp trích danh từ để trích lớp thực thể cho hệ thống: 3

- Bước 1: Mô tả ngắn gọn nhưng phải đầy đủ hệ thống trong một đoạn văn. Có thể mô tả lần lượt các chức năng của hệ thống theo từng người dùng để tránh bị sót. Có thể thay thế bước này bằng cách tổng hợp tất cả các kịch bản (bao gồm kịch bản chuẩn và kịch bản ngoại lệ) của tất cả các use case của hệ thống. 4
- Bước 2: Trích các danh từ xuất hiện trong đoạn văn của bước 1. Mỗi danh từ xuất hiện chỉ cần tính một lần. 5
- Bước 3: Đánh giá các danh từ. 6
  - Một là đề xuất danh từ thành một lớp thực thể. 7
  - Hai là đề xuất danh từ thành thuộc tính của lớp nào đó.
  - Ba là loại bỏ danh từ đó vì quá chung chung, trừu tượng, hoặc ngoài pham vi quản lí của hệ thống.
- Bước 4: Xác định quan hệ số lượng giữa các thực thể. 8 9
  - Nếu quan hệ 1-1 thì có thể giữ nguyên hoặc gộp lại.
  - Nếu quan hệ 1-n thì giữ nguyên.
  - Nếu quan hệ là n-n thì phải đề xuất các lớp thực thể trung gian để tách thành ít nhất 2 quan hệ 1-n.
- Bước 5: Xác định quan hệ đối tượng giữa các thực thể: quan hệ kế thừa, hợp thành (aggregation), gắn chặt (composition), liên kết (association), phụ thuộc (dependence)... 10

Lưu ý khi vẽ biểu đồ lớp thực thể pha phân tích: 11

- Tên lớp, tên thuộc tính có thể dùng ngôn ngữ tự nhiên vẫn được, tuy nhiên, nên đặt tên theo chuẩn code convention để vào pha thiết kế đỡ phải thay đổi. 12
- Thuộc tính chưa cần kiểu dữ liệu

- Các lớp thực thể chưa cần thuộc tính id trong pha phân tích. 1

Áp dụng cho hệ thống quản lí đăng kí tín chỉ, các bước được thực hiện như sau:

Bước 1: Mô tả hệ thống trong một đoạn văn 2

Hệ thống là một trang web hỗ trợ quản lí đăng kí tín chỉ và điểm của sinh viên trong một trường đại học. Trong đó, sinh viên được phép đăng kí các môn học, lớp học phần theo lịch học cá nhân vào đầu mỗi học kì; xem lịch học hàng ngày với đầy đủ thông tin về kíp học, ngày học, tuần học, phòng học, tòa nhà; xem kết quả học chi tiết với từng môn học, từng đầu điểm thành phần của môn học, cũng như điểm trung bình tích lũy các kì học của từng năm học hay từ đầu khóa học theo ngành học của sinh viên. Giảng viên được phép đăng kí dạy các môn học, các lớp học phần theo nguyện vọng cá nhân, sau đó sẽ được bộ môn quản lí duyệt và phân công chính thức; nhập hay sửa điểm của các sinh viên trong các lớp học phần do mình dạy; xem lịch dạy hàng tuần; và có thể xem các thống kê cá nhân về tỉ lệ sinh viên qua môn, điểm trung bình của sinh viên theo các môn học hay lớp học phần do mình dạy. Nhân viên giáo vụ có thể quản lí thông tin liên quan đến các môn học; lên lịch hay điều chỉnh lịch học cho các lớp học phần; quản lí thông tin sinh viên theo yêu cầu của sinh viên. Nhân viên khảo thí có thể xuất bảng điểm toàn khóa theo yêu cầu của sinh viên. Nhân viên quản lí có thể quản lí thông tin giảng viên theo yêu cầu của giảng viên; xem các loại thống kê: thống kê loại học lực, thống kê sinh viên theo kết quả học, theo ngành học, theo khoa, theo năm học, hay học kì; thống kê giảng viên theo khối lượng giảng dạy, theo kết quả sinh viên, theo khoa hay bộ môn. 3

Bước 2: Trích các danh từ xuất hiện trong đoạn văn 4

- Các danh từ liên quan đến người: sinh viên, nhân viên, giảng viên, nhân viên giáo vụ, nhân viên khảo thí, nhân viên quản lí 5
- Các danh từ liên quan đến vật: trường đại học, toà nhà, phòng học, bảng điểm
- Các danh từ liên quan đến thông tin: Khoa, ngành học, bộ môn, năm học, học kì, môn học, lớp học phần, điểm thành phần, điểm tích lũy, điểm trung bình môn, tín chỉ, học lực, khóa học, lịch học, thời khóa biểu, tuần học, ngày học, kíp học, báo cáo thống kê, thống kê học lực, thống kê sinh viên, thống kê giảng viên, thống kê môn học, thống kê học kì.

Bước 3: Đánh giá và lựa chọn các danh từ làm lớp thực thể hoặc thuộc tính 6

- Các danh từ trừu tượng: hệ thống, trang web, báo cáo thống kê, thông tin -> loại. 7
- Các danh từ liên quan đến người: 8
  - Thành viên hệ thống -> Thanhvien (trừu tượng): tên, tên đăng nhập, mật khẩu, ngày sinh, địa chỉ, email, điện thoại, vai trò, ghi chú. 9
  - Sinh viên -> lớp Sinhvien: kế thừa lớp Thành viên, thêm thuộc tính: mã sinh viên.
  - Nhân viên -> lớp Nhanvien (trừu tượng): kế thừa lớp Thành viên, thêm thuộc tính: vị trí.
  - Giảng viên -> lớp Giangvien: kế thừa Thanhvien

- Nhân viên giáo vụ -> lớp Giaovu: Kế thừa Thanhvien 1
- Nhân viên khảo thí -> lớp NVKhaothi: kế thừa Thanhvien
- Nhân viên quản lí -> lớp NVQuanli: Kế thừa Thanhvien

- Các danh từ liên quan đến vật: 2
  - Trường đại học -> lớp Truong: tên, địa chỉ, mô tả 3
  - Tòa nhà -> lớp Toanha: Tên, mô tả
  - Phòng học -> lớp Phonghoc: Tên, sức chứa, mô tả
  - Bảng điểm, là số nhiều của một điểm của một môn học nên không đề xuất lớp.
- Các danh từ thông tin: 4
  - Khoa + ngành -> lớp Khoa: Tên, mô tả 5
  - Bộ môn -> lớp Bomon: Tên, mô tả
  - Năm học -> lớp Namhoc: Tên, mô tả
  - Học kì -> lớp Hocki: Tên, mô tả
  - Môn học -> lớp Monhoc: Tên, số tín chỉ, mô tả
  - Lớp học phần -> lớp Lophocphan: Tên
  - Đầu điểm thành phần -> Daudiem: tên, mô tả
  - Điểm tích lũy, điểm trung mình môn là thông tin dẫn xuất nên đề xuất là thuộc tính của một lớp nào đó, quyết định sau.
  - Tín chỉ là thuộc tính của môn học
  - Học lực -> lớp Loaihocluc: tên, mô tả
  - Khóa học là thuộc tính của sinh viên học một ngành học.
  - Lịch học -> lớp Lichhoc: liên quan đến tuần học, ngày học, kíp học, phòng học, giảng viên.
  - Thời khóa biểu là số nhiều của lịch học.
  - Tuần học -> lớp Tuanhoc: tên, mô tả
  - Ngày học -> lớp Ngayhoc: tên thứ ngày trong tuần, mô tả
  - Kíp học -> lớp Kip: tên, mô tả
  - Thống kê loại học lực -> lớp TKLoaihocluc
  - Thống kê môn học -> lớp TKMonhoc
  - Thống kê giảng viên -> lớp TKGiangvien
  - Thống kê sinh viên -> TKSinhvien
  - Thống kê học kì -> TKHocki

Bước 4: Xác định quan hệ số lượng giữa các thực thể 6

- Trường có nhiều khoa/ngành -> quan hệ giữa Truong và Khoa là 1-n 7
- Khoa có nhiều bộ môn -> Khoa - Bomon là 1-n
- Bộ môn có nhiều giảng viên -> Bomon - Giangvien là 1-n

1

- Trường có nhiều tòa nhà -> Truong - Toanha là 1-n
- Tòa nhà có nhiều phòng học -> Toanha - Phonghoc là 1-n
- Một năm học có nhiều học kì, một học kì có thể nằm trong nhiều năm học -> Namhoc - Hocki là n-n -> đề xuất lớp Kihoc kết nối giữa 1 năm học và 1 học kì (Kihoc # Hocki).
- Một kì học có nhiều môn học, một môn học có thể học trong nhiều kì học -> Kihoc - Monhoc là n-n -> đề xuất lớp MonhocKihoc xác định duy nhất một môn học dạy trong một kì học.
- Một môn học có một số môn học tiên quyết.
- Một môn học, trong một kì học, có nhiều lớp học phần -> MonhocKihoc - Lophocphan là 1-n.
- Một môn học có nhiều đầu điểm thành phần, một đầu điểm thành phần có thể xuất hiện trong nhiều môn học -> Monhoc - Daudiem là n-n -> đề xuất lớp MonhocDaudiem xác định duy nhất một đầu điểm trong một môn học.
- Một sinh viên có thể học nhiều ngành học/khoa, một ngành/khoa có thể có nhiều sinh viên -> Sinhvien - Khoa là n-n -> đề xuất lớp SinhvienKhoa xác định duy nhất một sinh viên học một ngành/khoa nhất định.
- Một sinh viên, đối với một ngành học, có thể học nhiều lớp học phần của nhiều môn học (mỗi môn học 1 lớp), mỗi lớp học phần có thể có nhiều sinh viên -> SinhvienKhoa - Lophocphan là n-n -> đề xuất lớp Dangkihoc xác định duy nhất một sinh viên đăng kí vào một lớp học phần của một môn học nhất định.
- Một Dangkihoc có nhiều MonhocDaudiem, 1 MonhocDaudiem có nhiều Dangkihoc, nên đề xuất lớp Ketqua xác định duy nhất đầu điểm thành phần của sinh viên đã học môn học xác định.
- Một lớp học phần có nhiều buổi học (lịch học) khác nhau -> Lophocphan - Lichhoc là 1-n.
- Tương tự, các quan hệ Giangvien - Lichhoc, Phonghoc - Lichhoc, Tuanhoc - Lichhoc, Ngayhoc - Lichhoc, Kiphoc - Lichhoc đều là 1-n.
- Ngoài ra, thông tin Họ và tên có thể tách thành một lớp thực thể riêng. Thông tin địa chỉ cũng có thể tách thành một lớp thực thể riêng.

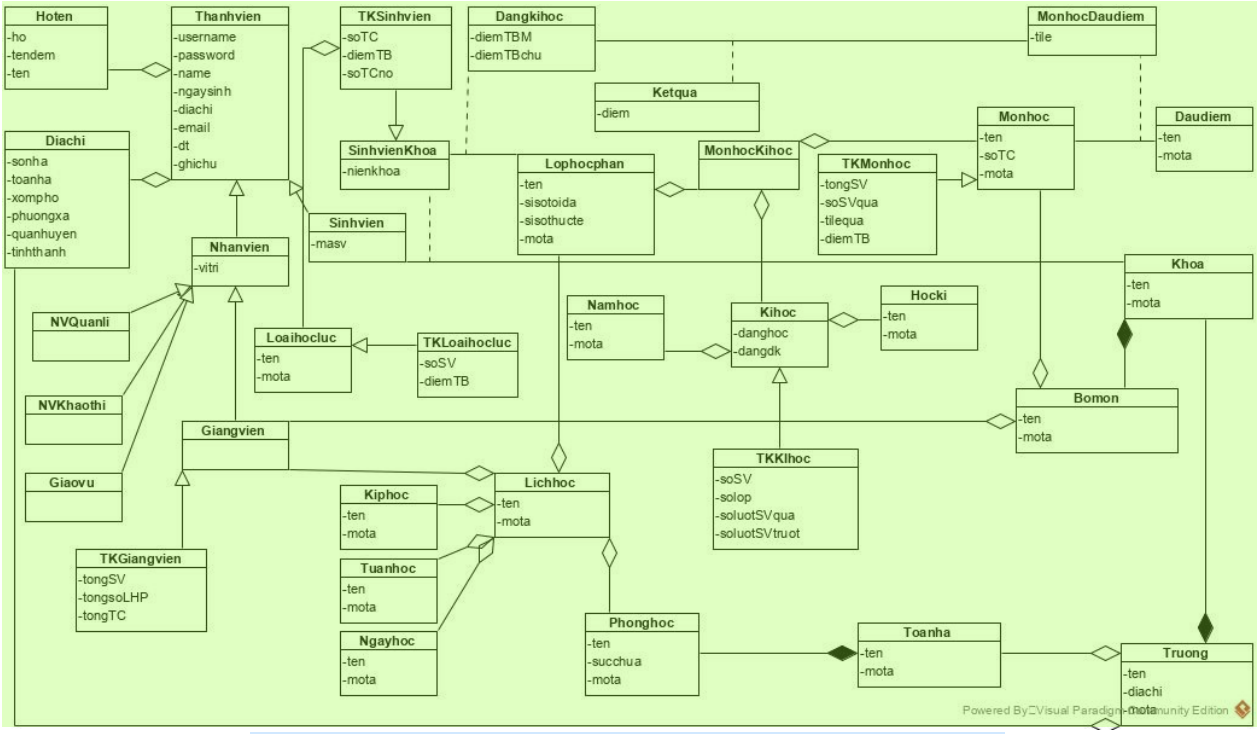
Bước 5: Xác định quan hệ đối tượng giữa các thực thể 2

3

- Khoa là thành phần của Trường
- Bộ môn là thành phần của Khoa.
- Giảng viên là thành phần của Bộ môn
- Môn học là thành phần của Bộ môn
- Tòa nhà là thành phần của Trường
- Phòng học là thành phần của tòa nhà
- Năm học và Học kì liên kết tạo ra Kì học duy nhất

- Sinh viên và ngành học liên kết tạo ra SinhvienKhoa duy nhất.
- Môn học và đầu điểm liên kết tạo ra MonhocDaudiem duy nhất
- Sinh viên và lớp học phần liên kết tạo ra Dangkihoc duy nhất
- Đăng kí học và MonhocDaudiem liên kết tạo ra điểm thành phần duy nhất cho sinh viên đối với môn học.
- Thông tin môn học nằm trong thông tin lớp học phần
- Thông tin lớp học phần, giảng viên, phòng học, tuần học, ngày học, kíp học nằm trong thông tin lịch học.
- Thông tin địa chỉ nằm trong thông tin về thành viên, trường
- Thông tin họ và tên nằm trong thông tin thành viên.
- Các lớp thống kê kế thừa từ các thực thể tương ứng.

Kết quả thu được biểu đồ lớp thực thể pha phân tích toàn hệ thống như trong Hình 3.5.



Hình 3.5: Biểu đồ lớp thực thể pha phân tích của hệ thống

3.2.3 Trích các lớp biên và điều khiển

Tóm tắt các bước thực hiện để vẽ biểu đồ lớp pha phân tích:

- Bước 1: Mỗi giao diện xuất hiện có thể đề xuất thành một lớp biên.
- Bước 2: Với mỗi giao diện, xem xét cần các thành phần nào để nhận dữ liệu vào, hiện dữ liệu lên, hay submit dữ liệu vào hệ thống. Mỗi thành phần như vậy đề xuất thành một thuộc tính của giao diện, đặt tên với các tiền tố tương ứng:
  - in: tiền tố của tên thuộc tính nhận dữ liệu vào (ô text, ô chọn....)

- out: tiền tố của tên thuộc tính hiện dữ liệu lên (bảng biểu, thông báo...) 1
- sub: tiền tố của tên thuộc tính để submit dữ liệu (nút nhấn, liên kết...)

- Bước 3: Với mỗi chức năng phải thực hiện tầng dưới tầng giao diện, thực hiện đề xuất phương thức: 2
  - Đề xuất tên phương thức (có thể đặt tên gợi nhớ, và tuân thủ code convention) 3
  - Xác định tham số đầu vào là gì, tham số đầu ra là gì.
  - Đề xuất gán hành động tương ứng với chức năng này cho lớp thực thể nào. Theo nguyên tắc:
    - Nếu tham số đầu ra liên quan lớp thực thể nào thì gán phương thức này cho lớp đó 4
    - Nếu không, xét tham số đầu vào. Nếu tham số đầu vào chỉ liên quan đến 1 thực thể thì gán phương thức này cho thực thể đó. Nếu tham số vào liên quan đến hai hay nhiều thực thể, thì tìm thực thể nào nhỏ nhất mà chứa được càng nhiều càng tốt các tham số đầu vào, gán phương thức này cho lớp thực thể đó.
- Vẽ biểu đồ lớp pha phân tích cho từng modul. 5

Lưu ý khi vẽ biểu đồ lớp pha phân tích: 6

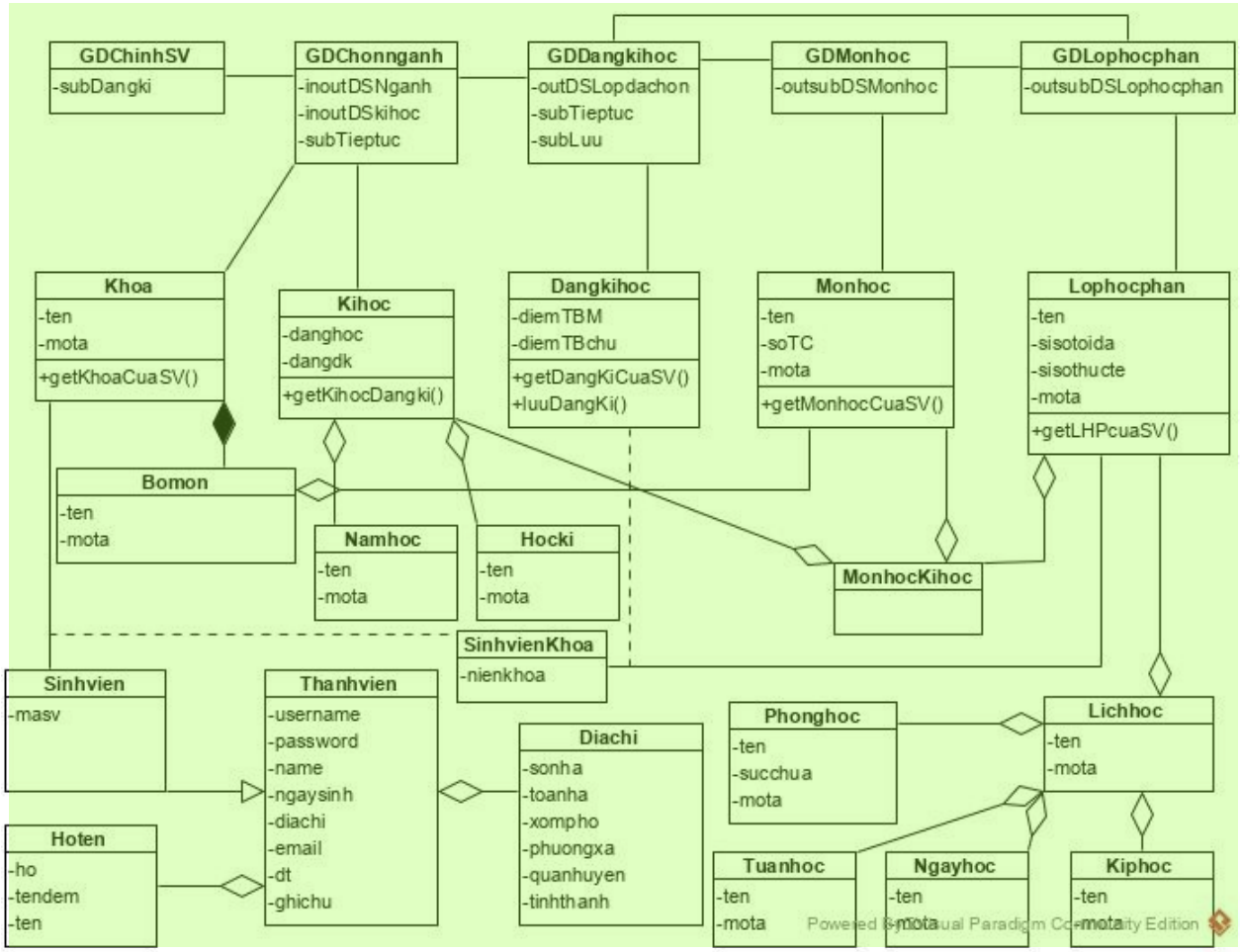
- Tên lớp, tên thuộc tính có thể dùng ngôn ngữ tự nhiên vẫn được, tuy nhiên, nên đặt tên theo chuẩn code convention để vào pha thiết kế đỡ phải thay đổi. 7
- Thuộc tính chưa cần kiểu dữ liệu
- Quan hệ giữa các lớp thực thể phải thống nhất, đồng bộ với quan hệ giữa chúng trong biểu đồ lớp thực thể đã vẽ ở bước trước.

a. Phân tích tĩnh modul đăng kí học 8

- Ban đầu, giao diện chính của sinh viên -> đề xuất lớp GDChinhSV, cần các thành phần: 9
  - chọn đăng kí học: kiểu submit
- Bước 2, giao diện chọn kì học, ngành học -> đề xuất lớp GDChonnganh, cần có các thành phần: 10
  - Chọn ngành học để đăng kí: vừa input, vừa output 11
  - Chọn kì học: vừa input, vừa output
  - Nút chọn tiếp tục: kiểu submit
- Để có danh sách ngành học để chọn ở bước 2, cần xử lí dưới hệ thống: 12
  - Tìm các ngành học của sinh viên 13
  - input: sinh viên
  - output: danh sách Khoa
  - đề xuất phương thức getKhoacuaSV(), gán cho lớp Khoa.
- Để có danh sách kì học đang mở đăng kí, cần xử lí bên dưới hệ thống: 14
  - Tìm các kì học đang mở đăng kí
  - input: không có

- output: danh sách Kihoc 1
- đề xuất phương thức getKihocDangki(), gán vào lớp Kihoc

- Bước 4, giao diện đăng kí -> đề xuất lớp GDDangki, cần có các thành phần: 2
  - Bảng danh sách các môn học: vừa output, vừa submit
  - Nút tiếp tục, nút lưu: kiểu submit
- Để có danh sách các đăng kí đã có, cần xử lí bên dưới hệ thống: 3
  - Tìm các môn học/lớp học phần đã đăng của sinh viên ở kì học đã chọn 4
  - input: học kì, sinh viên
  - output: danh sách Dangkihoc
  - đề xuất phương thức getDangkicuaSV(), gán vào lớp Dangkihoc



Hình 3.6: Biểu đồ lớp phân tích modul đăng kí hoc 6

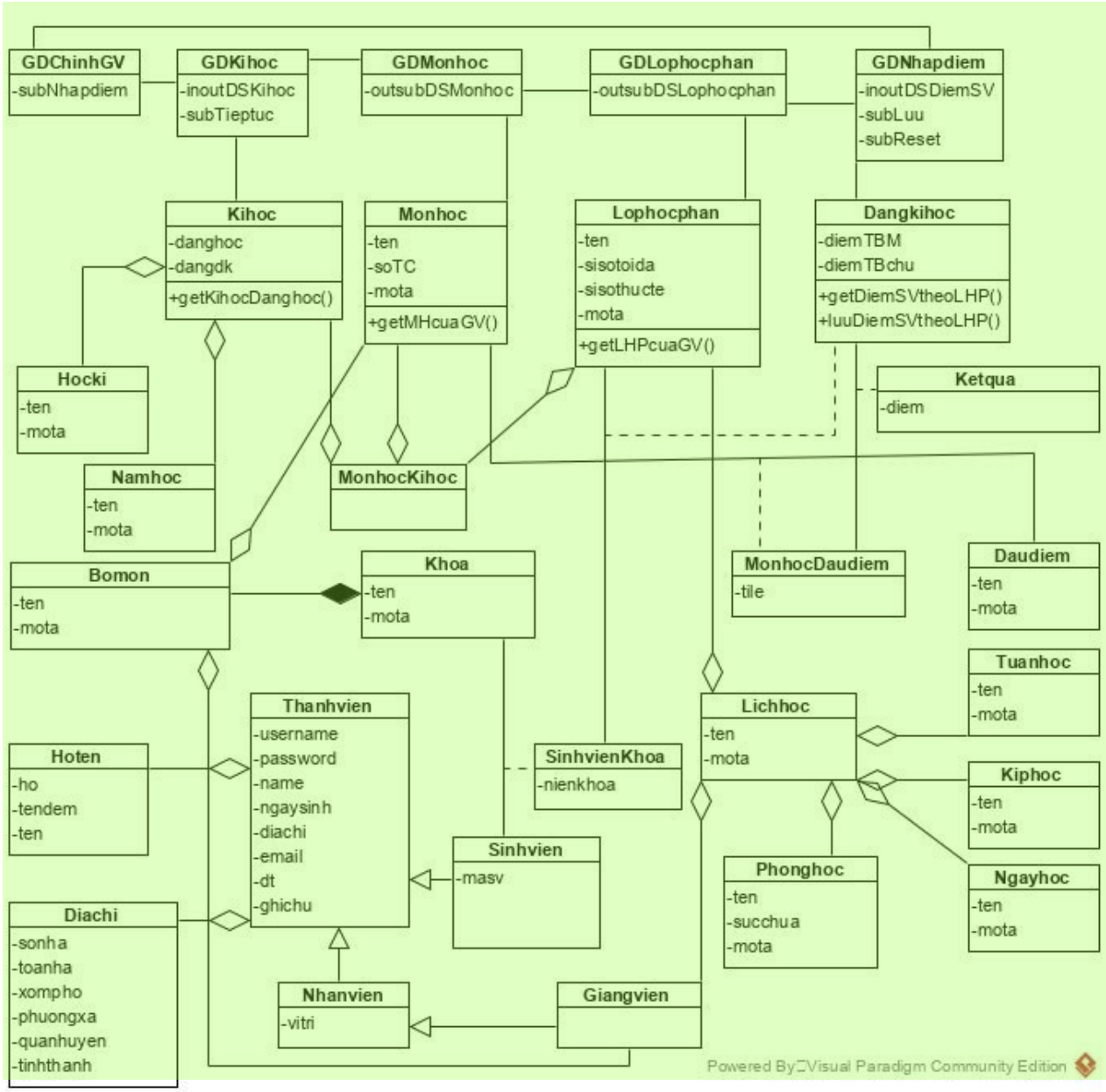
- Bước 6, giao diện chọn môn học -> đề xuất lớp GDMonhoc, cần các thành phần: 7
  - bảng danh sách các môn học có thể đăng kí: vừa output, vừa submit
- Để hiển thị được danh sách các môn học, cần xử lí trước dưới hệ thống:

- Tìm các môn học có thể dăng kí của sinh viên 1
- input: sinh viên, kì học
- output: danh sách Monhoc
- đề xuất phướng thức getMonhoccuaSV(), gán cho lớp Monhoc.

- Bước 8, giao diện lớp học phần -> đề xuất lớp GDLophocphan, cần có: 2
  - bảng danh sách lớp học phần: vừa output, vừa submit
- Để hiển thị danh sách lớp học phần, cần xử lí trước từ dưới hệ thống:
  - Tìm các lớp học phần của môn học
  - input: monhoc, kì học
  - output: danh sách Lophocphan
  - đề xuất phương thức getLHPcuaMH(), gán cho lớp Lophocphan.

Kết quả thu được biểu đồ lớp cho modul đăng kí học như trong Hình 3.6. 3

b. Phân tích tĩnh cho modul nhập điểm 4



Hình 3.7: Biểu đồ lớp phân tích modul nhập điểm

- Ban đầu, giao diện chính của giảng viên -> đề xuất lớp GDChinhGV, cần có:
  - chọn nhập/sửa điểm: kiểu submit
- Bước 2, giao diện chọn kì học -> đề xuất GDKihoc, cần có:
  - danh sách kì học: vừa input, vừa output
  - nút tiếp tục: kiểu submit
- Để có danh sách kì học đang học, cần xử lí dưới hệ thống:
  - Tìm các kì học đnag học
  - input: không có

- output: danh sách các Kihoc 1
- đề xuất phương thức getKihocDanghoc(), gán cho lớp Kihoc.

- Bước 4, giao diện chọn môn học -> đề xuất GDMonhoc, cần có: 2
  - danh sách các môn học: vừa output, vừa submit
- Sau khi chọn kì học, cần xử lí dưới hệ thống để lấy danh sách môn học: 3
  - Tìm các môn học của giảng viên đang dạy 4
  - input: giảng viên, kì học
  - output: danh sách các MonhocDaudiem
  - đề xuất phương thức getMHcuaGV(), gán cho lớp Monhoc.
- Bước 6, giao diện chọn lớp học phần -> đề xuất lớp GDLophocphan, cần có: 5
  - bảng danh sách lớp học phần: vừa output, vừa submit
- Để có danh sách lớp học phần, cần xử lí trước từ dưới hệ thống: 6
  - Lấy danh sách các lớp học phần của môn học do giảng viên dạy 7
  - input: môn học, giảng viên
  - output: danh sách Lophocphan
  - đề xuất phương thức getLHPcuaGV(), gán cho lớp Lophocphan
- Bước 8, giao diện nhập điểm -> đề xuất lớp GDNhapdiem, cần có: 8
  - Danh sách sinh viên + các đầu điểm: vừa output, vừa input
  - Nút lưu, nút reset: kiểu submit
- Để có danh sách sinh viên, cần xử lí trước từ dưới hệ thống: 9
  - lấy danh sách sinh viên + các đầu điểm đã có lên 10
  - input: lớp học phần
  - output: thông tin sinh viên + các đầu điểm thành phần của môn học (Dangkihoc)
  - đề xuất phương thức getDiemSVtheoLHP(), gán cho lớp Dangkihoc.
- Sau khi click lưu điểm, cần xử lí dưới hệ thống: 11
  - lưu danh sách sinh viên + các đầu điểm vào 12
  - input: thông tin sinh viên + các đầu điểm thành phần của môn học (Dangkihoc)
  - output: thành công hay không
  - đề xuất phương thức luuDiemSVtheoLHP(), gán cho lớp Dangkihoc.

Kết quả thu được biểu đồ lớp phân tích modul nhập điểm như trong Hình 3.7. 13

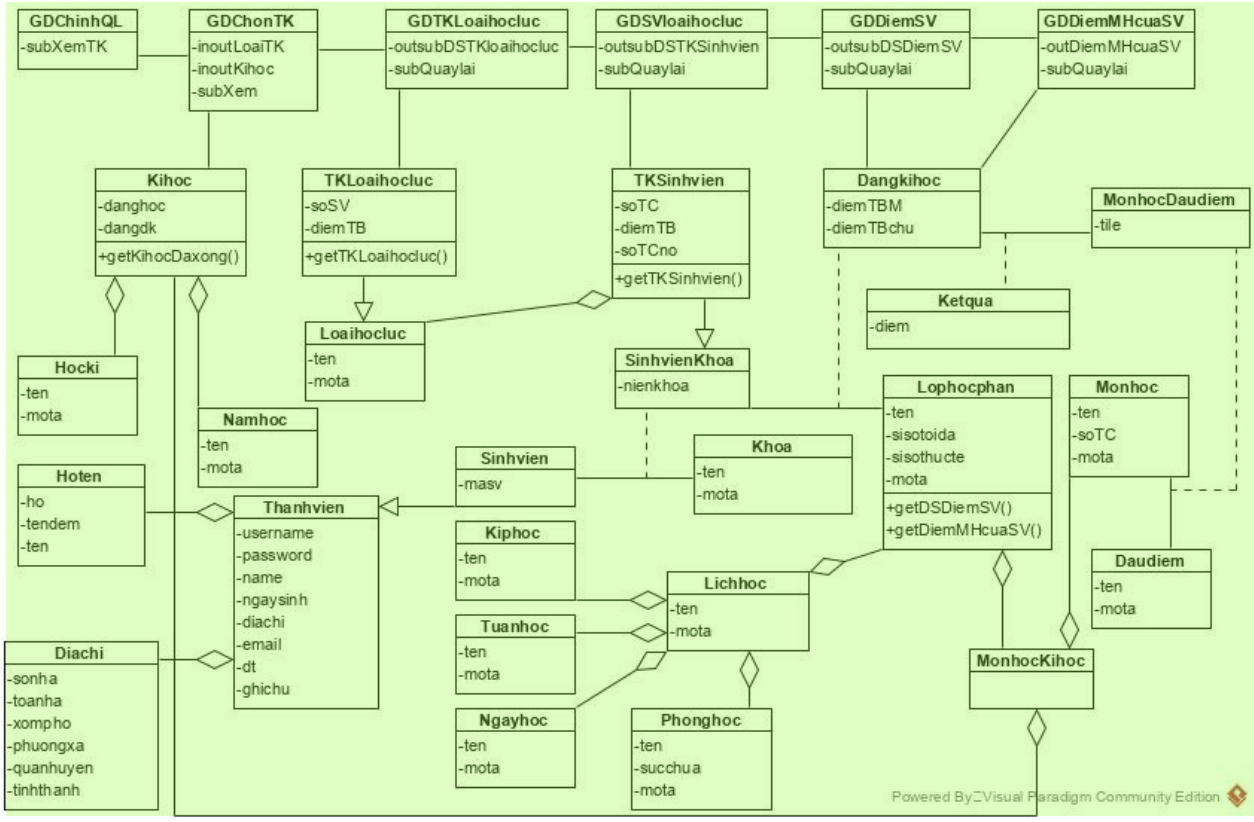
c. Phân tích tĩnh cho modul xem thống kê loại học lực 14

- Ban đầu, giao diện chính của quản lí -> đề xuất lớp GDChinhQL, cần có 15
  - chọn xem thống kê: kiểu submit
- Bước 2, giao diện chọn loại thống kê và kì học -> đề xuất lớp GDChonTK, cần có:
  - danh sách loại thống kê: vừa input, vừa output
  - danh sách kì học đã kết thúc: vừa input, vừa output

- nút xem: kiểu submit 1

- Đề có danh sách kì học đã kết thúc, cần xử lí trước từ dưới hệ thống: 2
  - lấy danh sách kì học đã kết thúc 3
  - input: không có
  - output: tất cả kì học đã kết thúc
  - đề xuất phương thức getKihocDaxong(), gán cho lớp Kihoc
- Bước 4+6, giao diện thống kê loại học lực -> đề xuất lớp GDTKLoaihocluc, cần có: 4
  - danh sách thống kê loại học lực: output, submit
- Để có dữ liệu thống kê, cần xử lí từ dưới hệ thống: 5
  - lấy dữ liệu thống kê loại học lực của học kì đã chọn 6
  - input: học kì
  - output: danh sách TKLoaihocluc
  - đề xuất phương thức getTKHocluc(), gán cho lớp TKLoaihocluc
- Bước 8, giao diện các sinh viên của loại học lực -> GDSVLoaihocluc, cần có: 7
  - Danh sách các sinh viên + điểm: kiểu output, submit
- Để có danh sách các sinh viên và điểm tổng kết, cần xử lí: 8
  - lấy danh sách sinh viên + kết quả theo học kì
  - input: loại học lực, học kì
  - output: danh sách TKSinhvien
  - đề xuất phương thức getTKSinhvien(), gán cho lớp TKSinhvien.
- Bước 10, giao diện danh sách điểm các môn học của sinh viên -> lớp GDDiemSV, cần: 9
  - danh sách điểm các môn học của sinh viên: output, submit
- Để có danh sách điểm các môn học của sinh viên, cần xử lí: 10
  - lấy danh sách điểm các môn học của sinh viên
  - input: sinh viên, học kì
  - output: danh sách các điểm sinh viên (Dangkihoc)
  - đề xuất phương thức getDSDiemSV(), gán cho lớp Dangkihoc
- Bước 12, giao diện điểm chi tiết môn học -> lớp GDDiemMHcuaSV, cần có: 11
  - thông tin môn học: output
  - danh sách các đầu điểm và điểm: output
- Để có thông tin điểm chi tiết, cần xử lí: 12
  - lấy điểm chi tiết môn học của sinh viên
  - input: môn học, sinh viên
  - output: điểm chi tiết của môn học (Dangkihoc)
  - đề xuất phương thức getDiemMHcuaSV(), gán cho lớp Dangkihoc.

Kết quả thu được biểu đồ lớp phân tích modul xem thống kê loại học lực như trong Hình 3.8. 13



Hình 3.8: Biểu đồ lớp phân tích modul xem thống kê theo loại học lực

3.2.4 Phân tích hoạt động

Tóm tắt các bước thực hiện để vẽ biểu đồ trạng thái của một modul/chức năng:

- Bước 1: Mỗi trạng thái chờ tương ứng với mỗi lần hệ thống hiển thị 1 giao diện để chờ tương tác của người dùng
- Bước 2: Điều kiện chuyển trạng thái tương ứng với hành động của người dùng trên giao diện tương ứng.

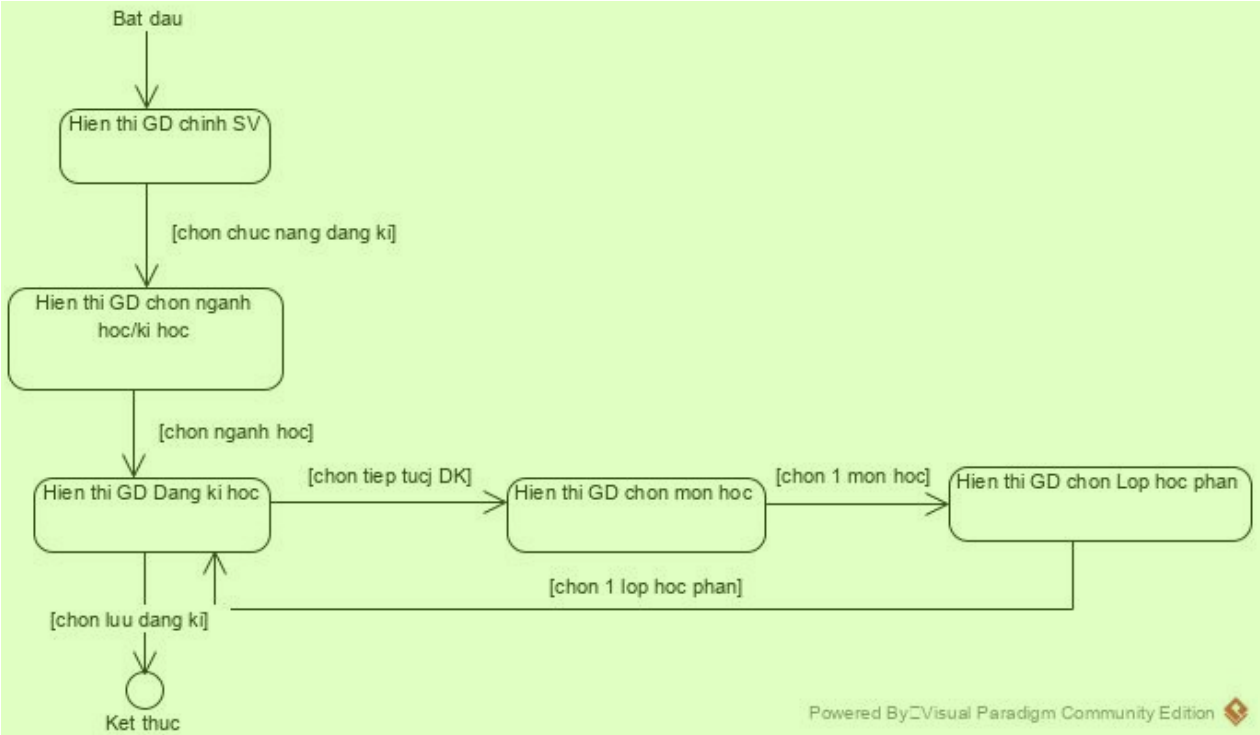
Tóm tắt các bước thực hiện để vẽ biểu đồ tuần tự/giao tiếp pha phân tích:

- Bước 1: Viết kịch bản phiên bản 2 từ biểu đồ lớp của modul: Mỗi bước là một hành động của một lớp nào đó trong biểu đồ lớp, hoặc là của tác nhân.
- Bước 2: Vẽ biểu đồ giao tiếp cho kịch bản chuẩn: các bước tương ứng với các bước trong kịch bản phiên bản 2. Quan hệ giữa các lớp phải thống nhất với quan hệ giữa chúng trong biểu đồ lớp của modul.
- Bước 3: chuyển hóa biểu đồ giao tiếp thành biểu đồ tuần tự cho kịch bản chuẩn. Kiểm tra lại các bước đúng tuần tự và logic nghiệp vụ.

a. Modul đăng kí học

Biểu đồ trạng thái mô tả hoạt động của modul như biểu diễn trong Hình 3.9: 1

- Từ giao diện chính của sinh viên, nếu chức năng đăng kí được chọn thì chuyển sang giao diện chọn ngành học, 2
- Từ giao diện chọn ngành học, khi ngành được chọn, hệ thống chuyển sang giao diện đăng kí học.
- Tại giao diện đăng kí, nếu click tiếp tục thì chuyển sang giao diện chọn môn học. Nếu chọn một lớn đã đăng kí thì chuyển sang giao diện chọn lớp học phần để chọn lại. Nếu chọn lưu thì lưu và kết thúc.
- Tại giao diện chọn môn học, nếu môn học được chọn thì chuyển sang giao diện chọn lớp học phần.
- Tại giao diện chọn lớp học phần, sau khi chọn lớp học phần thì quay về giao diện đăng kí.

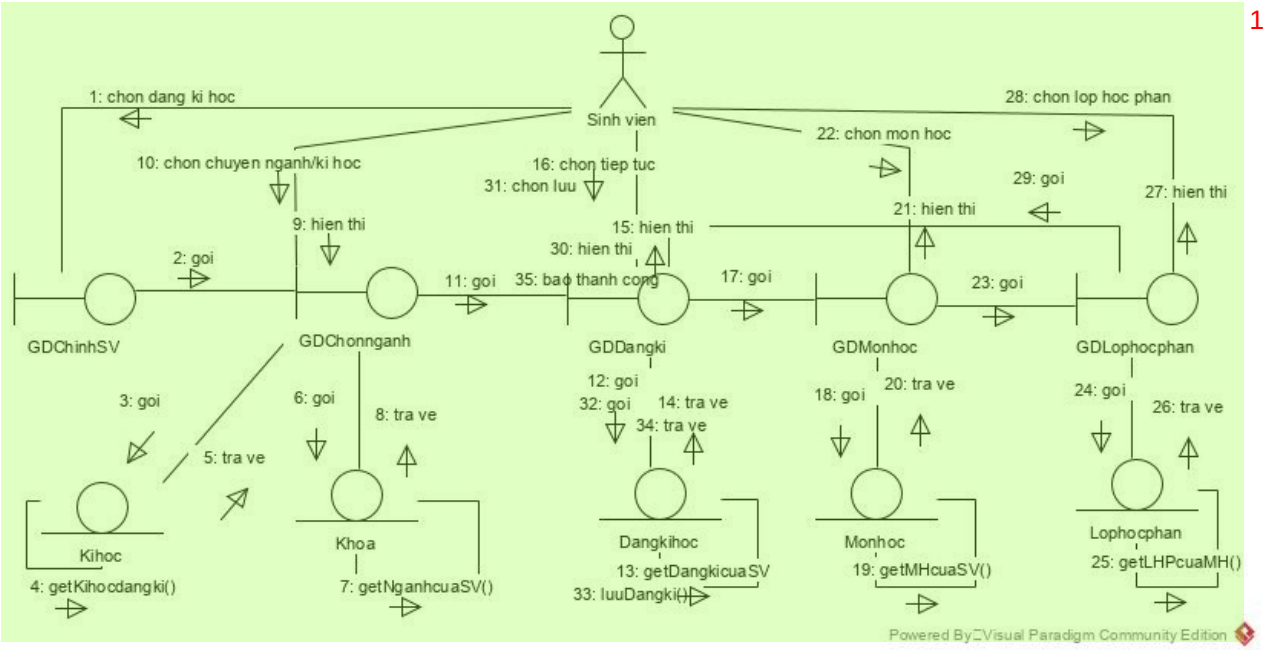


3

Hình 3.9: Biểu đồ chuyển trạng thái của modul đăng kí học 4

Kịch bản v.2 cho modul đăng kí môn học như sau (Hình 3.10): 5

1. Tại giao diện chính của sinh viên sau khi đăng nhập, sinh viên click chọn đăng kí. 6
2. Lớp GDChinhSV gọi lớp GDChonnganh
3. GDChonnganh gọi lớp Kihoc yêu cầu tìm danh sách các kì học đnag mở đăng kí
4. Lớp Kihoc tìm các kì học đang mở đăng kí
5. Lớp Kihoc trả kết quả lại cho lớp GDChonnganh
6. GDChonnganh gọi lớp Khoa yêu cầu tìm danh sách ngành



Hình 3.10: Biểu đồ giao tiếp của modul đăng kí học

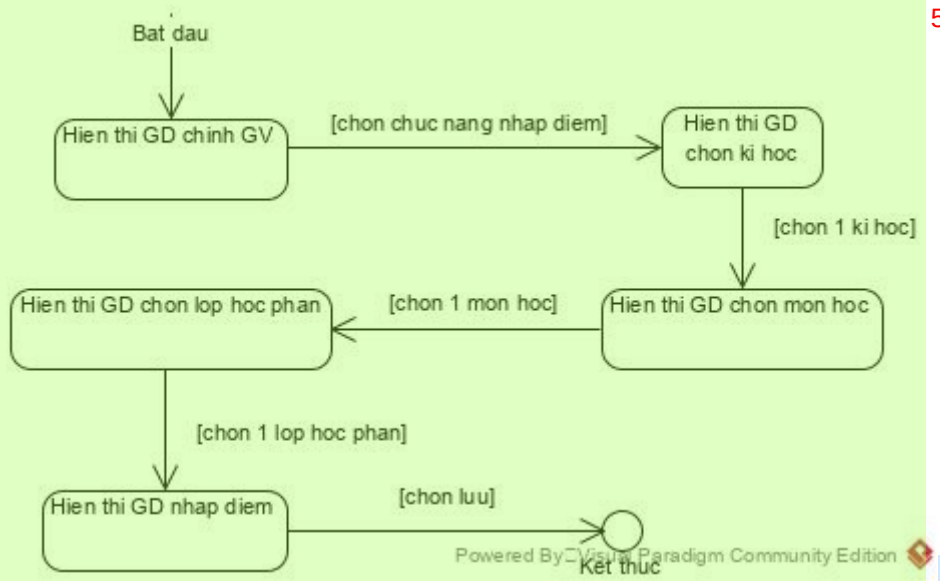
7. Lớp Khoa tìm các ngành học của sinh viên
8. Lớp Khoa trả kết quả lại cho lớp GDChonnganh
9. Lớp GDChonnganh hiển thị cho sinh viên
10. Sinh viên click 1 ngành của mình học
11. Lớp GDChonnganh gọi lớp GDDangki
12. Lớp GDDangki gọi lớp Dangkihoc yêu cầu tìm thông tin đã đăng kí
13. Lớp Dangkihoc lấy thông tin đã đăng kí của sinh viên
14. Lớp Dangkihoc trả kết quả lại cho lớp GDDangki.
15. Lớp GDDangki hiển thị cho sinh viên.
16. Sinh viên chọn tiếp tục đăng kí
17. Lớp GDDangki gọi GDMonhoc
18. Lớp GDMonhoc gọi lớp Monhoc yêu cầu tìm các môn học được đăng kí
19. Lớp Monhoc tìm các môn học mà sinh viên được đăng kí
20. Lớp Monhoc trả kết quả lại cho lớp GDMonhoc
21. Lớp GDMonhoc hiển thị cho sinh viên
22. Sinh viên click chọn 1 môn học
23. Lớp GDMonhoc gọi lớp GDLophocphan
24. Lớp GDLophocphan gọi lớp Lophocphan yếu cầu tìm các lớp học phần của môn học
25. Lớp Lophocphan tìm các lớp học phần của môn học
26. Lớp Lophocphan trả kết quả cho lớp GDLophocphan
27. Lớp GDLophocphan hiển thị cho sinh viên

28. Sinh viên click 1 lớp lớp học phần
29. GDLophocphan gọi lại lớp GDDangki
30. GDDangki lại hiển thị (Lặp các bước từ 12 đến 27 đến khi đủ số tín chỉ)
31. Sinh viên click lưu
32. Lớp GDDangki gọi lớp Dangkihoc yêu cầu lưu
33. Lớp Dangkihoc lưu thông tin đăng kí
34. Lớp Dangkihoc trả kết quả lại cho lớp GDDangki
35. Lớp GDDangki báo đăng kí thành công.

b. Modul nhập điểm

Biểu đồ trạng thái của modul được trình bày trong Hình 3.11:

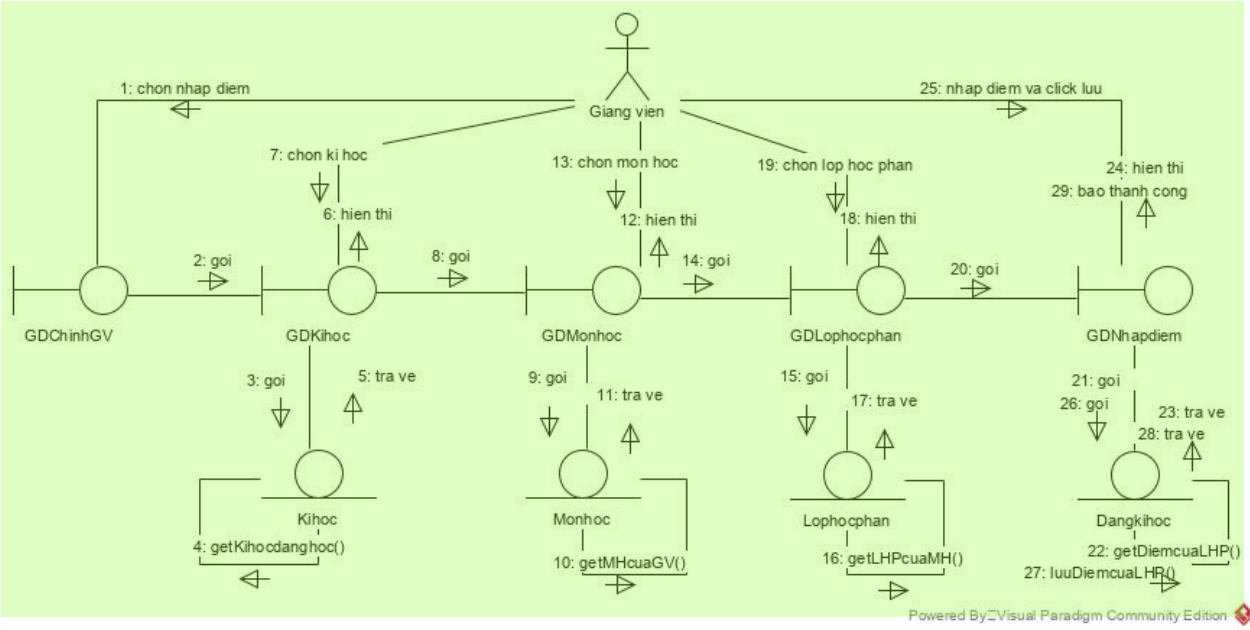
- Tại giao diện chính của giảng viên, hệ thống chuyển sang giao diện chọn môn học khi chức năng chọn môn học được chọn.
- Tại giao diện chọn môn học, khi giảng viên chọn 1 môn học, hệ thống sẽ chuyển sang giao diện chọn lớp học phần.
- Tại giao diện chọn lớp học phần, khi giảng viên chọn 1 lớp học phần, hệ thống sẽ chuyển sang giao diện nhập điểm.
- Tại giao diện nhập điểm, sau khi nhập nhập và chọn lưu, hệ thống lưu điểm vào và kết thúc.



Hình 3.11: Biểu đồ chuyển trạng thái của modul nhập điểm

Kịch bản v.2 của modul diễn ra như sau (Hình 3.12):

1. Tại giao diện chính của giảng viên, sau khi đăng nhập, giảng viên chọn chức năng nhập điểm.
2. Lớp GDChinhGV gọi GDKihoc
3. Lớp GDKihoc goi lớp Kihoc yêu cầu tìm kì học đang học
4. Lớp Kihoc tìm các kì học đang học
5. Lớp Kihoc trả kết quả cho lớp GDKihoc
6. Lớp GDKihoc hiển thị cho giảng viên
7. Giảng viên chọn 1 kì học.
8. Lớp GDKihoc gọi GDMonhoc.
9. Lớp GDMonhoc gọi lớp Monhoc yêu cầu tìm các môn do giảng viên dạy
10. Lớp Monhoc tìm các môn học do giảng viên dang dạy
11. Lớp Monhoc trả kết quả lại cho lớp GDMonhoc.
12. Lớp GDMonhoc hiển thị cho giảng viên
13. Giảng viên chọn 1 môn học trong danh sách
14. Lớp GDMonhoc gọi lớp GDLophocphan
15. Lớp GDLophocphan gọi lớp Lophocphan yêu cầu tìm lớp học phần của giảng viên
16. Lớp Lophocphan tìm các lớp học phần do giảng viên dạy, của môn học đã chọn
17. Lớp Lophocphan trả kết quả lại cho lớp GDLophocphan
18. Lớp GDLophocphan hiển thị cho giảng viên.
19. Giảng viên chọn 1 lớp học phần trong danh sách
20. Lớp GDLophocphan gọi lớp GDNhapdiem
21. Lớp GDNhapdiem gọi lớp Dangkihoc yêu cầu tìm thông tin điểm đã có
22. Lớp Dangkihoc tìm sinh viên và các đầu điểm đã có của lớp học phần
23. Lớp Dangkihoc trả kết quả lại cho lớp GDNhapdiem
24. Lớp GDNhapdiem hiển kết quả cho giảng viên.
25. Giảng viên nhập thêm đầu điểm và click lưu
26. Lớp GDNhapdiem gọi lớp Dangkihoc yêu cầu cập nhật
27. Lớp Dangkihoc lưu thông tin điểm của lớp học phần
28. Lớp Dangkihoc trả kết quả lại cho lớp GDNhapdiem
29. Lớp GDNhapdiem thông báo thành công.

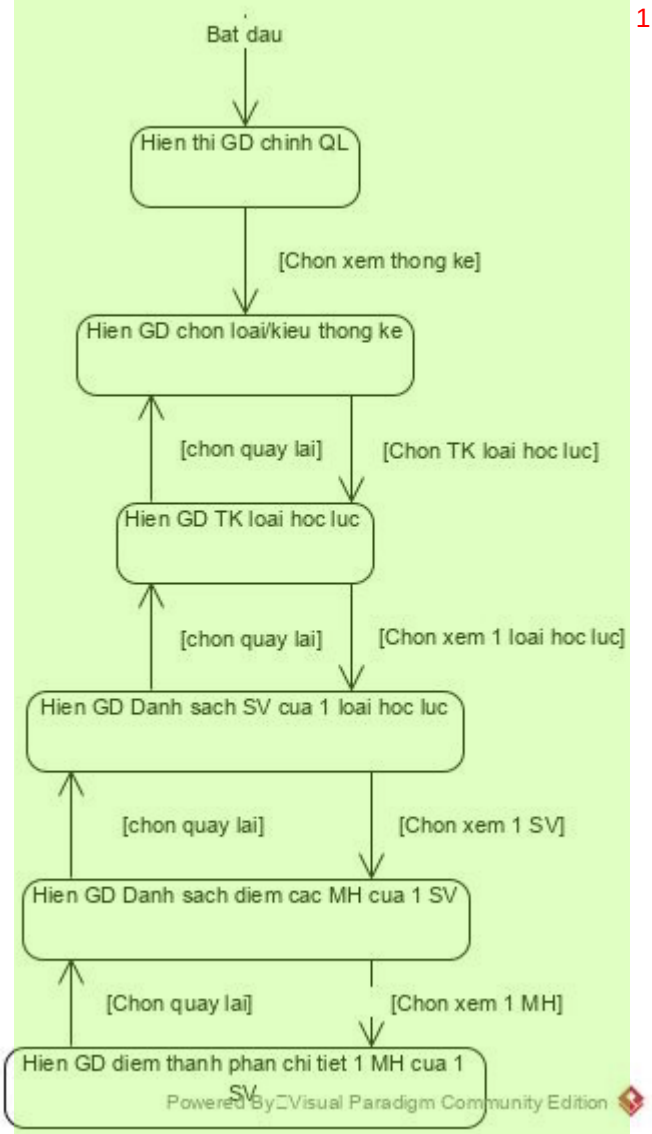


Hình 3.12: Biểu đồ giao tiếp của modul nhập điểm

c. Modul xem thống kê loại học lực

Hoạt động của biểu đồ trạng thái của modul được biểu diễn trong Hình 1.13:

- Tại giao diện chính của nhân viên quản lí, khi chọn chức năng xem thống kê, hệ thống sẽ chuyển sang giao diện chọn kiểu thống kê
- Tại giao diện chọn kiểu thống kê, chọn loại thống kê loại học lực, hệ thống chuyển snag giao diện thống kê loại học lực
- Tại giao diện thống kê loại học lực, khi chọn vào 1 loại học lực, hệ thống chuyển snag giao diện thống kê các sinh viên của loại học lực đã chọn
- Tại giao diện thống kê sinh viên, chọn một sinh viên, hệ thống chuyển sang giao diện liệt kê điểm các môn học của sinh viên đã chọn
- Tại giao diện điểm các môn học, chọn 1 môn học, hệ thống sẽ chuyển snag giao diện thông tin điểm chi tiết của môn học của sinh viên đã chọn.
- Tại tất cả các giao diện, nếu chọn quay lại, hệ thống sẽ quay lại giao diện trước đấy.

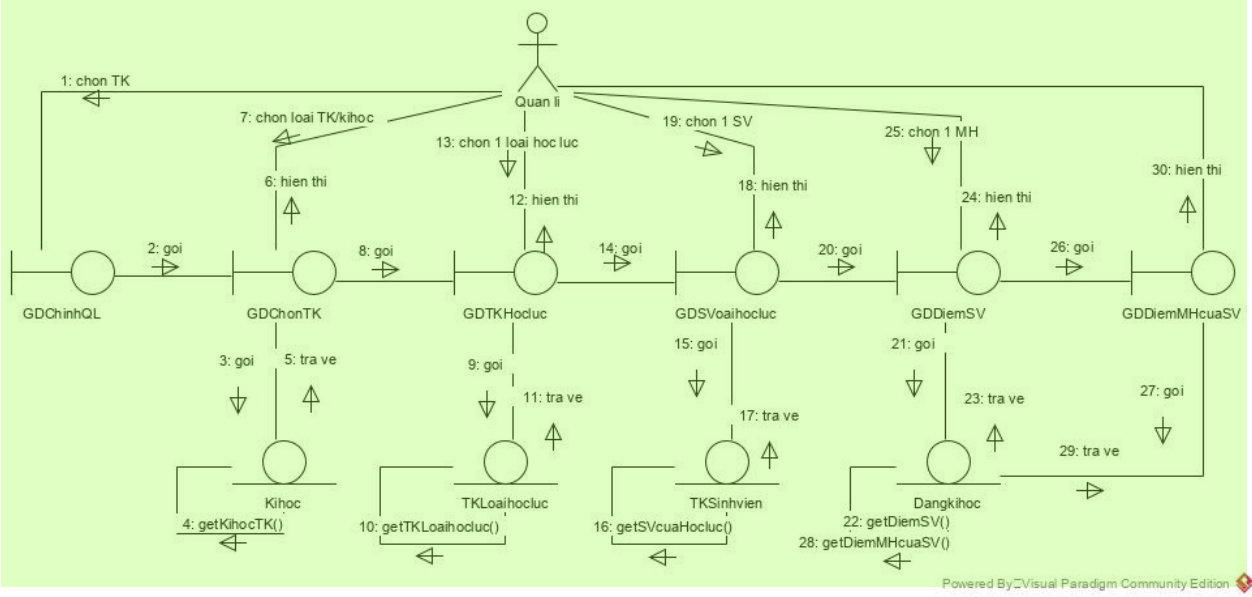


Hình 3.13: Biểu đồ chuyển trạng thái của modul xem thống kê loại học lực

Kịch bản v.2 của modul diễn ra như sau (Hình 3.14):

1. Tại giao diện chính của nhân viên quản lí, sau khi đăng nhập, quản lí chọn xem thống kê.
2. Lớp GDChinhQL gọi lớp GDChonTK.
3. Lớp GDChonTK gọi lớp Kihoc yêu cầu tìm các kì học đã kết thúc
4. Lớp Kihoc tìm các kì học đã kết thúc
5. Lớp Kihoc trả kết quả lại cho lớp GDChonTK.
6. Lớp GDChonTK hiển thị cho QL
7. QL chọn thống kê loại học lực
8. Lớp GDChonTK gọi lớp GDTKHocluc

9. Lớp GDTKHocluc hiển thị cho QL
10. QL chọn học kì muốn thống kê
11. Lớp GDTKHocluc gọi lớp TKLoaihocluc yêu cầu lấy dữ liệu
12. Lớp TKLoaihocluc lấy dữ liệu thống kê
13. Lớp TKLoaihocluc trả kết quả cho lớp GDTKHocluc
14. Lớp GDTKHocluc hiển thị cho QL
15. QL chọn 1 loại học lực
16. Lớp GDTKHocluc gọi lớp GDSVLoaihocluc
17. Lớp GDSVLoaihocluc gọi lớp TKSinhvien yêu cầu lấy dữ liệu
18. Lớp TKSinhvien lấy dữ liệu thống kê sinh viên của loại học lực
19. Lớp TKSinhvien trả kết quả lại cho lớp GDSVLoaihocluc
20. Lớp GDSVLoaihocluc hiển thị cho QL



Hình 3.14: Biểu đồ giao tiếp của modul thống kê theo loại học lực

21. QL click 1 sinh viên
22. Lớp GDSVLoaihocluc gọi lớp GDDiemSV
23. Lớp GDDiemSV gọi lớp Dangkihoc yêu cầu lấy dữ liệu điểm
24. Lớp Dangkihoc lấy điểm các môn học của sinh viên
25. Lớp Dangkihoc trả kết quả cho lớp GDDiemSV
26. Lớp GDDiemSV hiển thị cho QL
27. QL chọn 1 môn học
28. Lớp GDDiemSV gọi lớp GDDiemMHcuaSV
29. Lớp GDDiemMHcuaSV gọi lớp Dangkihoc yêu cầu lấy điểm chi tiết
30. Lớp Dangkihoc lấy thông tin điểm chi tiết của 1 môn học của sinh viên.

31. Lớp Dangkihoc trả kết quả ;ại cho lớp GDDiemMHcuaSV [1]
32. Lớp GDDiemMHcuaSV hiển thị cho QL

3.3 BÀI TẬP [2]

1. Lập danh sách glossary list cho ứng dụng của nhóm. [3]
2. Mô tả hệ thống bằng ngôn ngữ tự nhiên.
3. Vẽ biểu đồ use case toàn hệ thống
4. Vẽ biểu đồ use case cho modul cá nhân
5. Viết kịch bản cho modul cá nhân
6. Trích lớp thực thể và vẽ biểu đồ lớp thực thể cho toàn hệ thống
7. Trích và vẽ biểu đồ lớp cho modul
8. Vẽ biểu đồ trạng thái cho modul
9. Viết kịch bản v.2 và vẽ biểu đồ giao tiếp cho modul.

CHƯƠNG 4: THIẾT KẾ 1

4.1 THIẾT KẾ LỚP THỰC THỂ 2

4.1.1 Quy trình thực hiện 3

Đầu vào là biểu đồ lớp thực thể pha phân tích, tiến hành các bước như sau: 4

- Bước 1: Bổ sung thuộc tính id cho các lớp không kế thừa từ lớp khác 5
- Bước 2: Bổ sung kiểu dữ liệu cho các thuộc tính. Kiểu dữ liệu phải tương ứng với kiểu dữ liệu của ngôn ngữ lập trình đã chọn.
- Bước 3: Chuyển đổi các quan hệ association sang các dạng aggregation/composition, nếu có: Nếu hai lớp A và B liên kết (asociation) tạo ra lớp C thì có thể chuyển thành:
  - A chứa C, C chứa B;
  - hoặc B chứa C, C chứa A;
  - hoặc C chứa A và B.
- Bước 4: Bổ sung thuộc tính đối tượng của các lớp thực thể, nếu có
  - Giữa hai lớp có quan hệ aggregation/composition thì lớp chứa lớp còn lại nên khai báo tường minh thuộc tính có kiểu là lớp còn lại
  - Số lượng (số ít/số nhiều) của thuộc tính phụ thuộc vào quan hệ số lượng giữa hai thực thể.
- Vẽ biểu đồ thực thể pha thiết kế 6

Lưu ý khi vẽ biểu đồ lớp thực thể pha thiết kế: 7

- Tên lớp, tên thuộc tính nên đặt tên theo chuẩn code convention: Tên lớp nên bắt đầu bằng chữ hoa, tên thuộc tính nên bắt đầu bằng chữ thường. Có thể đặt tên gợi nhớ theo Tiếng Anh hoặc Tiếng Việt. Nếu Tiếng Việt thì không được có dấu, không được có dấu cách. Trong tên lớp/thuộc tính không được có các kí tự đặc biệt. 8
- Thuộc tính cần khai báo tường minh kiểu dữ liệu.

4.1.2 Áp dụng 9

Bước 1 10

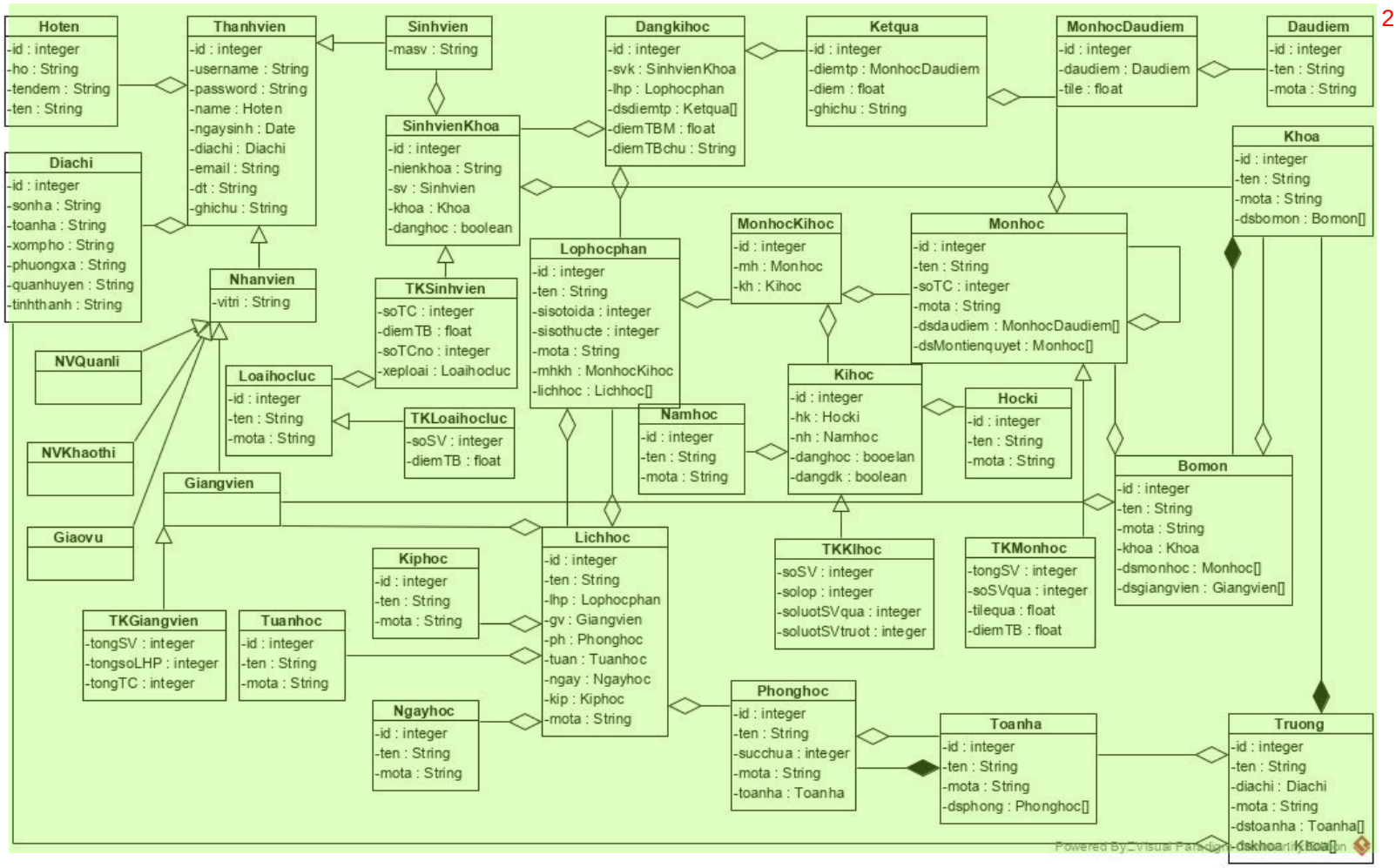
Các lớp được bổ sung thuộc tính id: trừ các lớp TK, Sinhvien, Thanhvien, Giaovu, 11 Giangvien, NVQuanli, NVKhaothi.

Bước 2 12

Thuộc tính của các lớp được bổ sung thuộc tính theo kiểu của ngôn ngữ lập trình Java. 13

Bước 3 14

- Quan hệ Sinhvien - Khoa -> SinhvienKhoa chuyển thành SinhvienKhoa chứa Sinhvien và Khoa. 1
- Quan hệ Monhoc - Daudiem -> MonhocDaudiem chuyển thành MonhocDaudiem chứa Monhoc và Daudiem.
- Quan hệ Namhoc - Hocki -> Kihoc chuyển thành Kihoc chứa Namhoc và Hocki
- Quan hệ Monhoc - Kihoc -> MonhocKihoc chuyển thành MonhocKihoc chứa Monhoc và Kihoc.
- Quan hệ SinhvienKhoa - Lophocphan -> Dangkihoc chuyển thành Dangkihoc chứa SinhvienKhoa và Lophocphan.
- Quan hệ Dangkihoc - MonhocDaudiem -> Ketqua chuyển thành Dangkihoc chứa Ketqua, Ketqua chứa MonhocDaudiem.



2

Hình 4.1: Biểu đồ thiết kế lớp thực thể 3

Bước 4

Bổ sung các thuộc tính thành phần của quan hệ hợp thành/gắn chặt, ta thu được biểu đồ lớp thực thể pha thiết kế như trong Hình 4.1.

4.2 THIẾT KẾ CSDL

4.2.1 Quy trình thực hiện

Đầu vào là biểu đồ lớp thực thể pha thiết kế, tiến hành các bước như sau:

- Bước 1: Với mỗi lớp thực thể, đề xuất một bảng dữ liệu tương ứng. Nên đặt tên bảng theo cách gợi nhớ tương ứng. Ví dụ, nếu lớp thực thể là Student, thì có thể đặt tên bảng tương ứng là tblStudent.
- Bước 2: Với mỗi lớp thực thể, bỏ qua các thuộc tính kiểu đối tượng, lấy các thuộc tính kiểu cơ bản đưa sang làm thuộc tính của bảng tương ứng, và chuyển kiểu dữ liệu theo hệ quản trị cơ sở dữ liệu tương ứng.
- Bước 3: Xét quan hệ số lượng giữa các lớp thực thể, quan hệ số lượng giữa hai lớp thực thể cũng là quan hệ số lượng giữa hai bảng tương ứng với hai lớp đấy. Khi đó:
  - Nếu hai bảng có quan hệ là 1-1 thì nên gộp lại, tuy nhiên, trong một số trường hợp đặc biệt thì có thể giữ nguyên.
  - Nếu hai bảng có quan hệ 1-n thì giữ nguyên.
  - Nếu hai bảng có quan hệ n-n thì nên bổ sung ít nhất một bảng trung gian để tách thành một số quan hệ 1-n. Thực tế, nếu đến thời điểm này vẫn còn quan hệ n-n giữa các bảng thì chứng tỏ trong biểu đồ lớp thực thể, vẫn còn quan hệ n-n, cho nên tốt nhất là nên quay lại sửa từ biểu đồ lớp thực thể pha phân tích.
- Bước 4: Bổ sung khóa cho các bảng:
  - Khóa chính (Primary key - PK): Bảng nào có thuộc tính id thì thiết lập thuộc tính đó làm khóa chính cho bảng.
  - Khóa ngoại (Foreinger key - FK): Nếu hai bảng tblA và tblB có quan hệ 1-n (1 tblA có n tblB), thì trong bảng tblB phải có khóa ngoài tham chiếu đến khóa chính của bảng tblA. Khó ngoại có thể đặt tên là aId hoặc idA.
- Bước 5: Loại bỏ các thuộc tính gây dư thừa dữ liệu. Có hai loại thuộc tính gây dư thừa dữ liệu:
  - Thuộc tính bị trùng lặp: thường là thuộc tính của cùng một đối tượng, không phải thuộc tính khóa, nhưng xuất hiện ở hai bảng khác nhau.
  - Thuộc tính dẫn xuất: là thuộc tính có thể tính toán cơ học từ các thuộc tính khác trong CSDL. Ví dụ một số thuộc tính xuất thân từ các lớp thực thể thống kê thì thường là thuộc tính dẫn xuất. Sau khi loại bỏ các thuộc tính dư thừa, nếu xuất hiện bảng nào không còn thuộc tính nào nữa ngoài 1 khóa ngoại thì có thể loại bỏ luôn bảng đấy.

4.2.2 Áp dụng 1

Bước 1 2

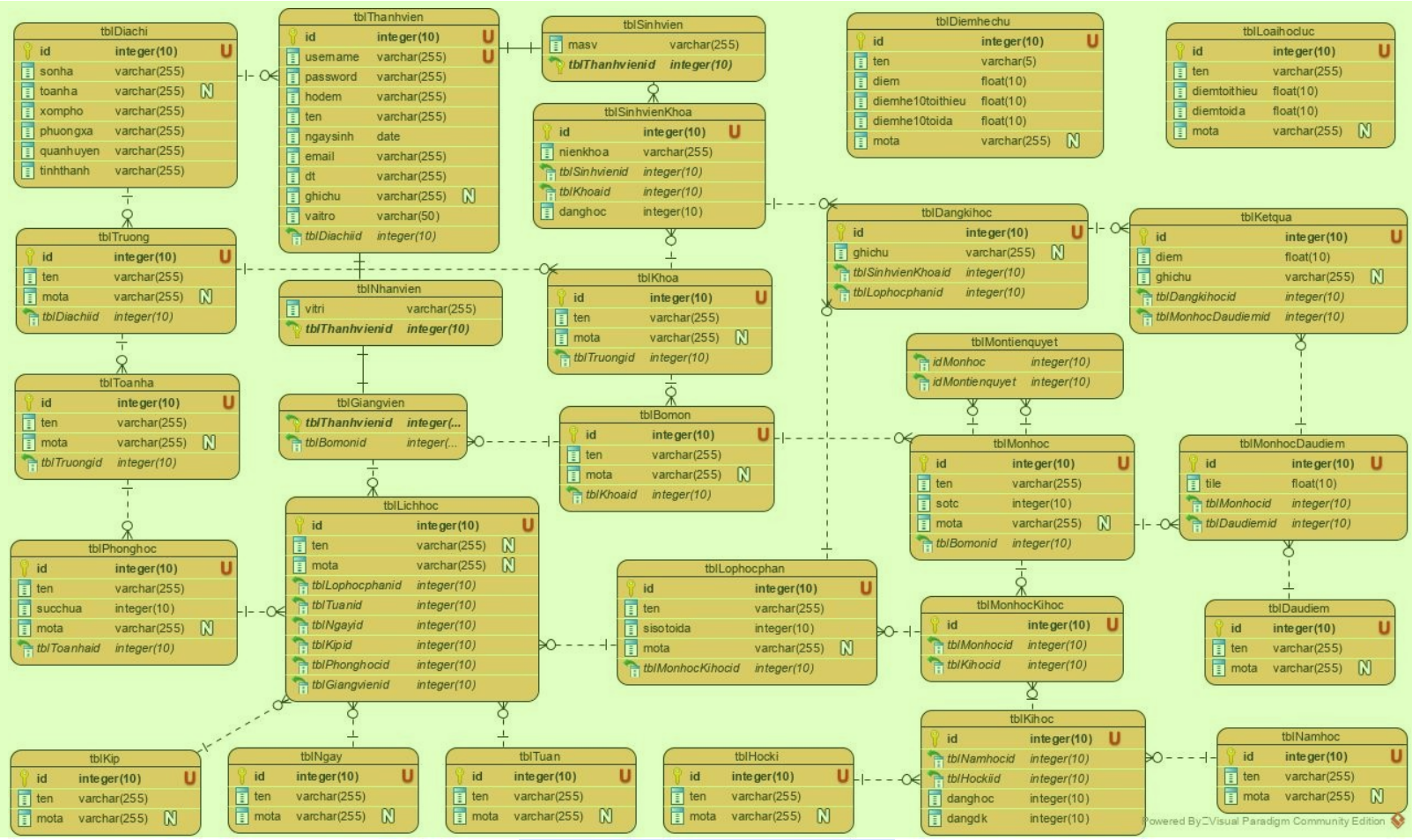
Mỗi lớp thực thể đề xuất bảng tương ứng: 3

- Lớp Truong -> bảng tblTruong 4
- Lớp Khoa -> bảng tblKhoa
- Lớp Toanha -> bảng tblToanha
- ....

Bước 2 5

Đưa thuộc tính không phải đối tượng của lớp thực thể thành thuộc tính của bảng tương ứng: 6

- tblTruong có các thuộc tính: id, tên, mô tả 7
- tblKhoa: id, tên, mô tả
- tblToanha: id, tên, mô tả
- ....



Hình 4.2: Biểu đồ thiết kế CSDL 9

Bước 3 1

Chuyển quan hệ số lượng giữa các lớp thực thể thành quan hệ số lượng giữa các bảng: 2

- 1 tblTruong - n tblKhoa 3
- 1 tblTruong - n tblToanha
- 1 tblToanha - n tblPhonghoc
- ....

Bước 4 4

Bổ sung các thuộc tính khóa. Khóa chính được thiết lập với thuộc tính id của các bảng tương ứng: trừ các bảng tblTK, tblSinhvien, tblThanhvien, tblGiangvien, tblNVQuanli, tblNVKhaothi, tblGiaovu. 5

Khóa ngoại được thiết lập cho các bảng: 6

- 1 tblTruong - n tblKhoa -> bảng tblKhoa có khóa ngoại tblTruongid 7
- 1 tblTruong - n tblToanha -> bảng tblToanha có khóa ngoại tblTruongid
- 1 tblToanha - n tblPhonghoc -> bảng tblPhonghoc có khóa ngoại tblToanhaid
- ....

Bước 5 8

Các thuộc tính dẫn xuất: 9

- điểm TBM, điểm TB chữ trong bảng tblDangkihoc 10
- Các thuộc tính của các lớp thống kê -> loại bỏ hết các bảng thống kê.

Kết quả thu được CSDL toàn hệ thống được trình bày trong Hình 4.2. 11

4.3 THIẾT KẾ CHI TIẾT CHO MODUL 12

4.3.1 Thiết kế tĩnh 13

Đầu vào là biểu đồ lớp pha phân tích, tiến hành các bước như sau: 14

- Bước 1: Thiết kế giao diện cho các giao diện xuất hiện trong biểu đồ. 15
- Bước 2: Đề xuất các lớp giao diện tương ứng với ngôn ngữ lập trình và nền tảng hệ thống. Ví dụ hệ thống chạy trên ứng dụng máy bàn, hệ thống chạy trên nền web, hệ thống chạy trên nền tảng thiết bị di động.
  - Đề xuất tên lớp tương ứng. Có thể tách, gộp các lớp giao diện so với pha phân tích
  - Thiết kế các thuộc tính đã liệt kê trong pha phân tích.
  - Bổ sung các *thuộc tính ẩn* để chứa thông tin cần thiết để xử lí và trao đổi qua lại giữa các lớp giao diện.
- Bước 3: Đề xuất các lớp truy xuất dữ liệu (DAO), nếu có.
  - Nên thiết kế dạng Interface hoặc kế thừa từ lớp trừu tượng để có thể dùng chung kết nối vào CSDL cho toàn hệ thống để tiết kiệm tài nguyên.

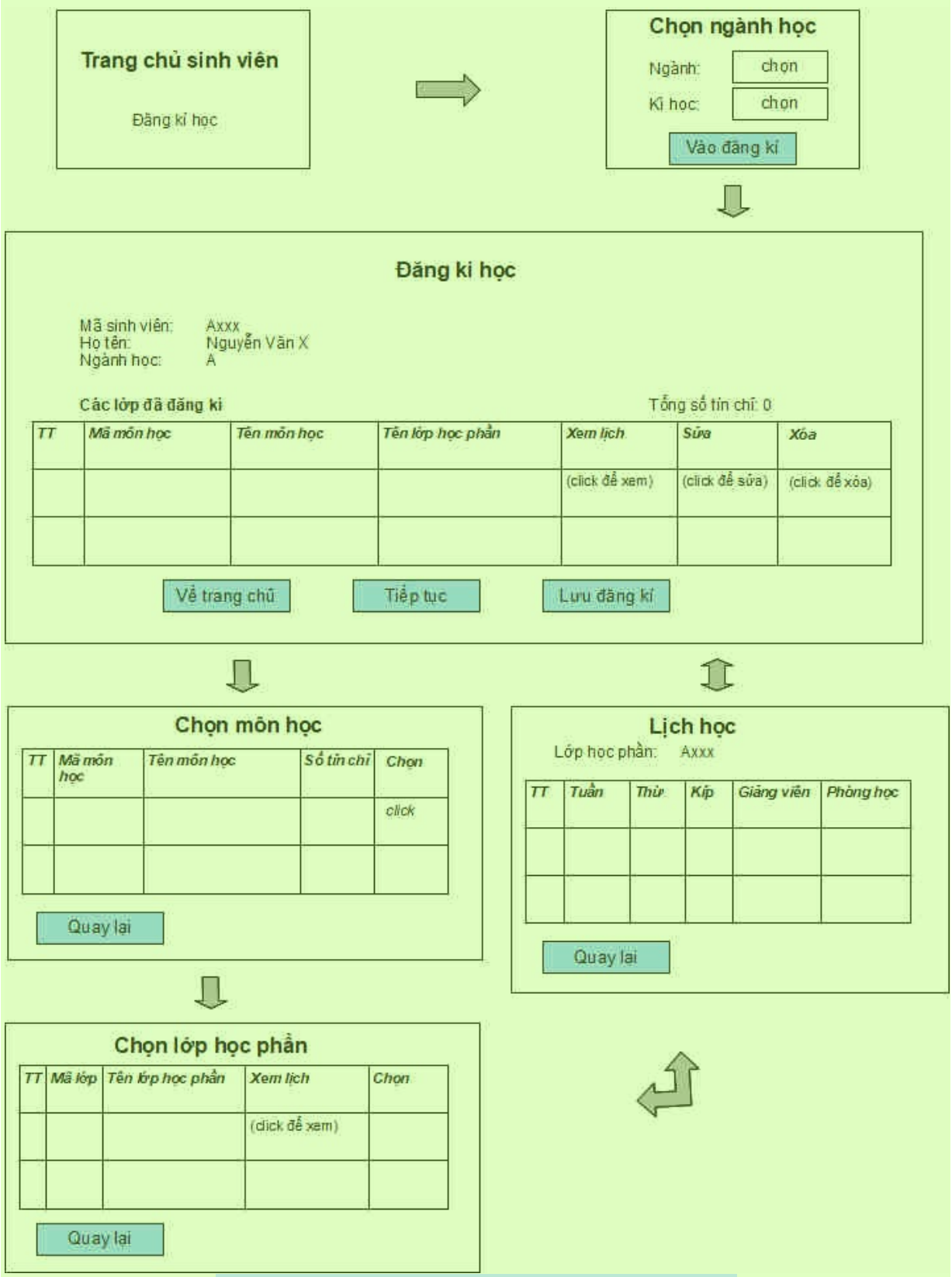
- Thường, nếu lớp thực thể trong pha phân tích cần phương thức, hành động nào thì nên đề xuất lớp DAO cho lớp thực thể tương ứng, và gán phương thức đang xem xét cho lớp vùa đề xuất. 1
- Với mỗi phương thức của lớp DAO, cần lưu ý thiết kế tên gợi nhớ, các tham số đầu vào, các tham số đầu ra.

- Bước 4: Bổ sung các lướp thực thể liên quan đến modul. Giữ nguyên quan hệ giữa các thực thể như các quan hệ giữa chúng trong biểu đồ lớp thực thể. 2

a. Thiết kế tĩnh modul đăng kí học 3

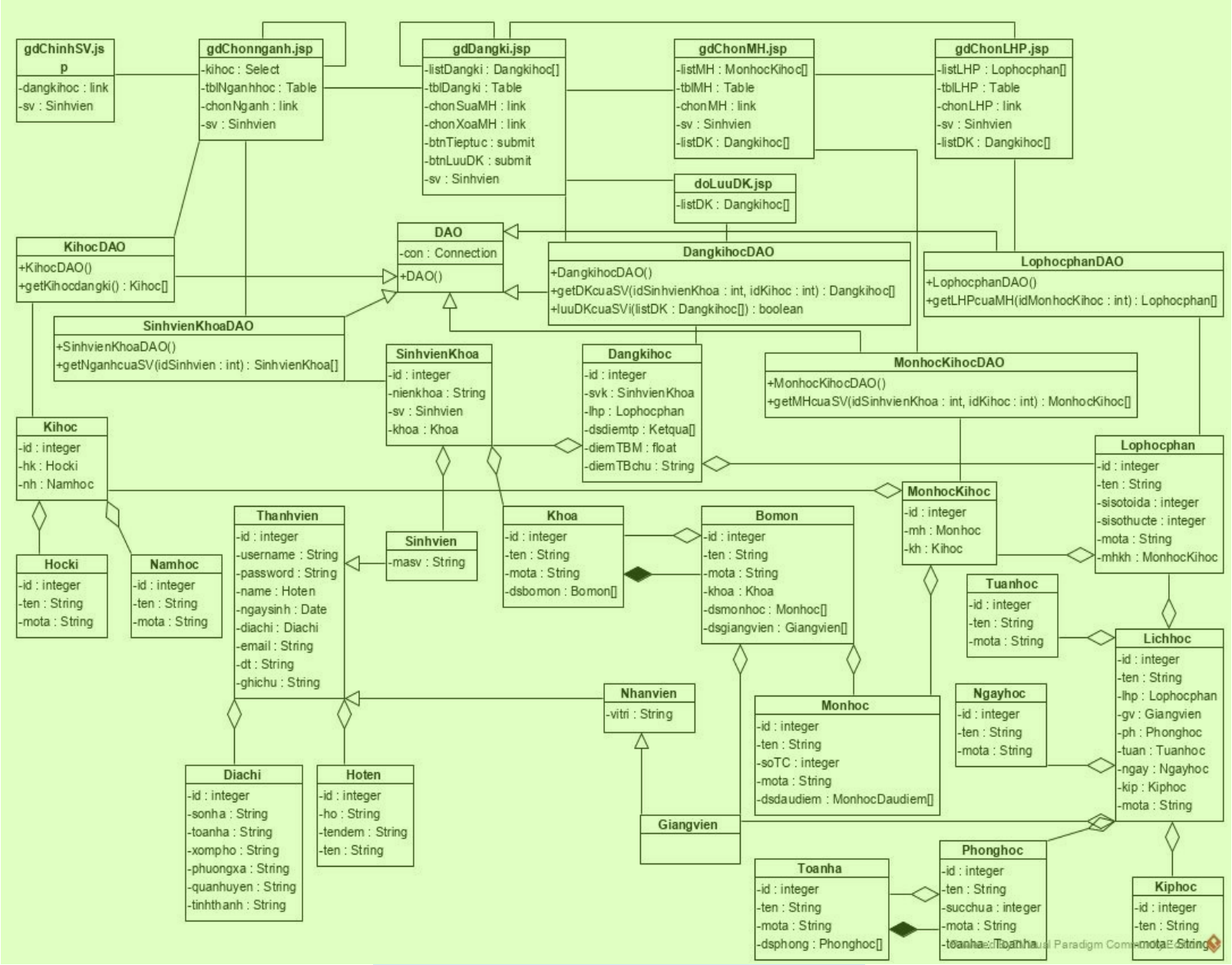
Các giao diện cho chức năng này được thiết kế như trong Hình 4.3. Biểu đồ lớp thiết kế tương ứng trong Hình 4.4: 4

- Tầng giao diện có các trang jsp: gdChinhSV, gdChonnganh, gdDangki, gdChonMH, gdChonLHP, và doLuuDK. 5
- Tầng thao tác với dữ liệu có các lớp DAO, KihocDAO, SinhvienKhoaDAO, MonhocKihocDAO, LophocphanDAO, DangkihocDAO.



Hình 4.3: Thiết kế giao diện cho modul đăng kí học

- Tầng thực thể gồm tất cả các thực thể liên quan.



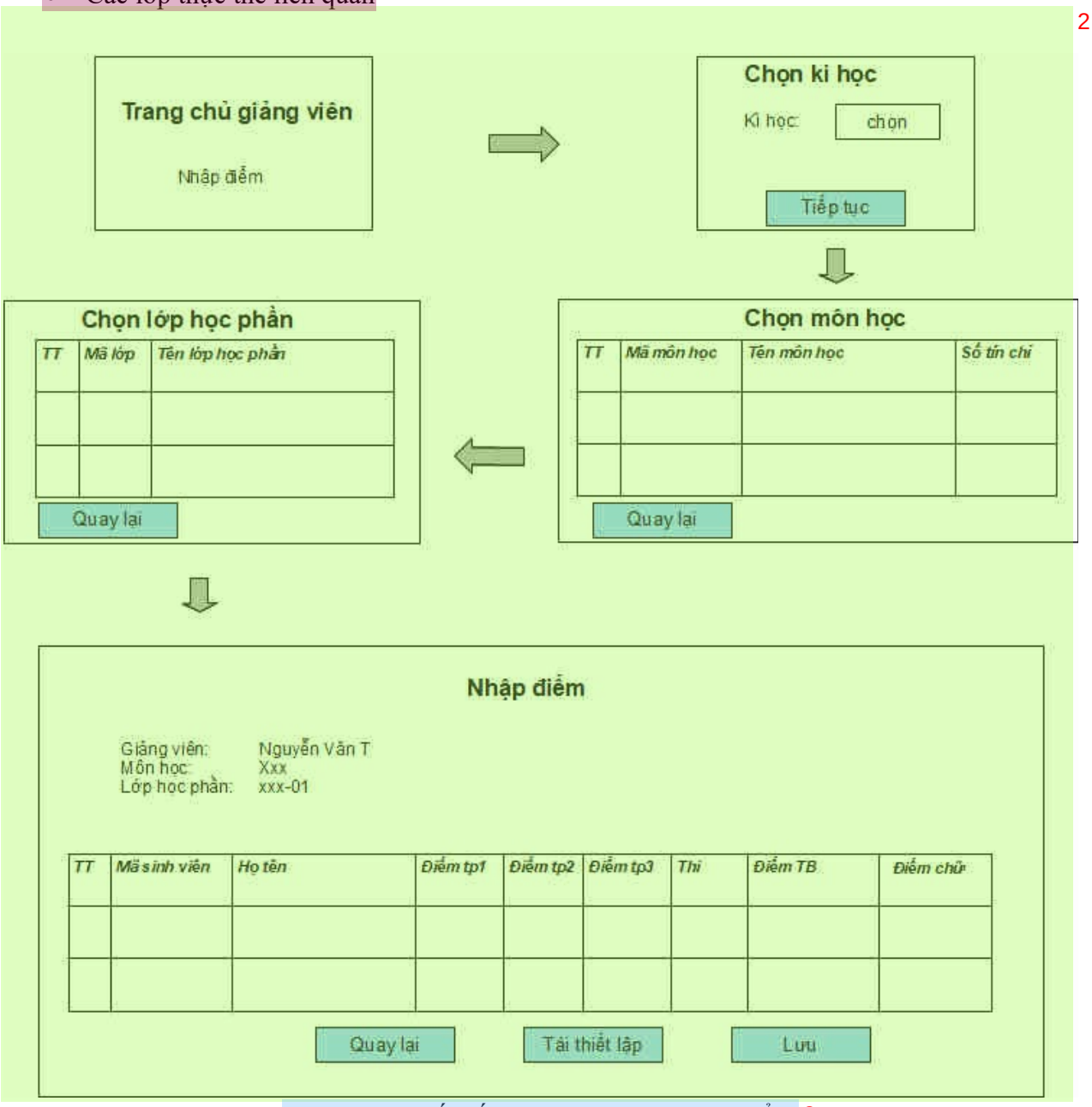
Hình 4.4: Biểu đồ lớp thiết kế modul đăng kí học

b. Thiết kế tĩnh modul nhập điểm

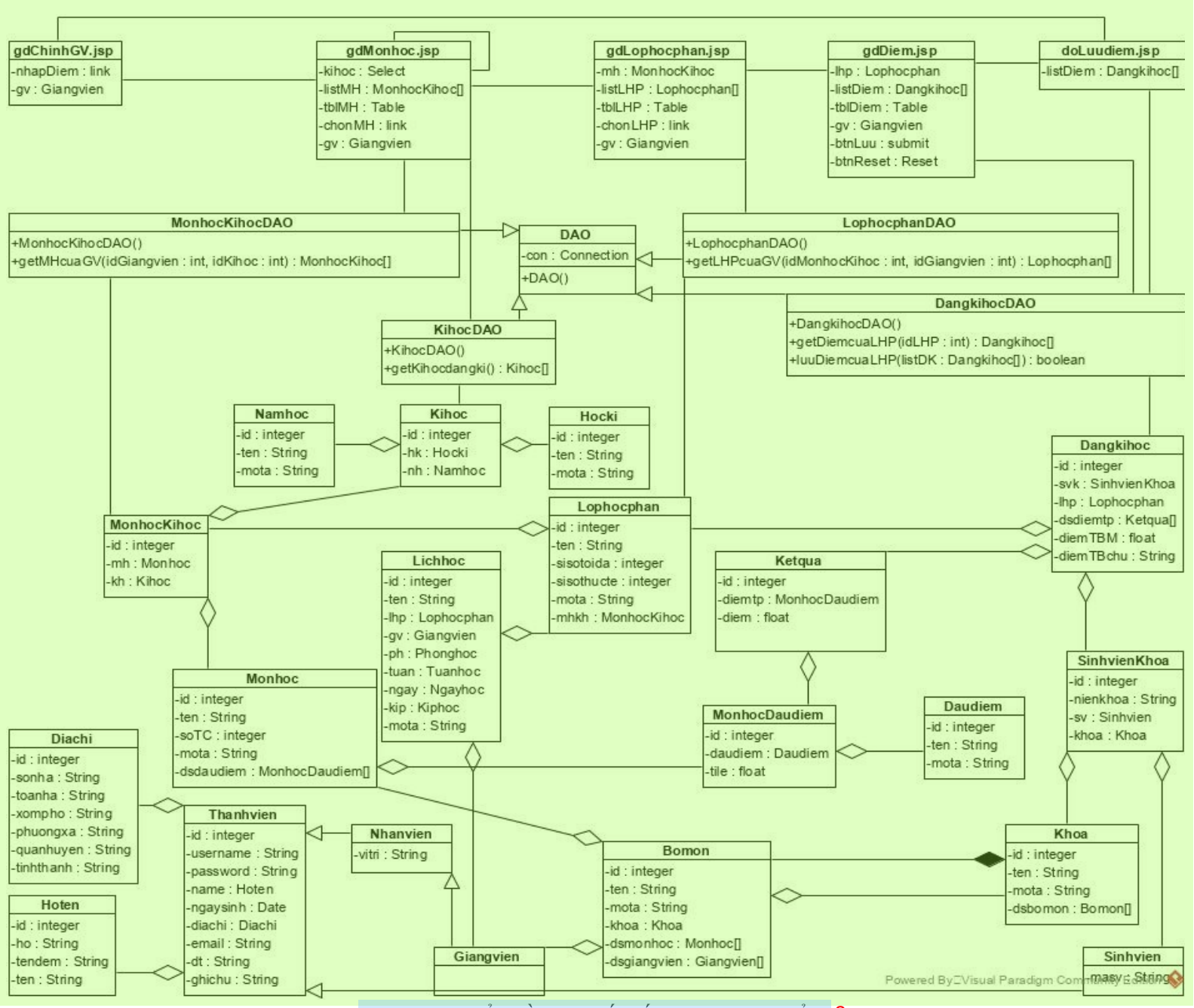
Các giao diện cho chức năng nhập điểm được thiết kế như trong Hình 4.5. Biểu đồ lớp của modul được thiết kế như trong Hình 4.6:

- Các lớp giao diện là các trang jsp: gdChinhGV, gdMonhoc, gdLophocphan, gdDiem, doLuudiem.
- Các lớp tầng truy cập dữ liệu DAO: DAO, KihocDAO, MonhocKihocDAO, LophocphanDAO, DangkihocDAO.

- Các lớp thực thể liên quan 1



Hình 4.5: Thiết kế giao diện modul nhập điểm 3



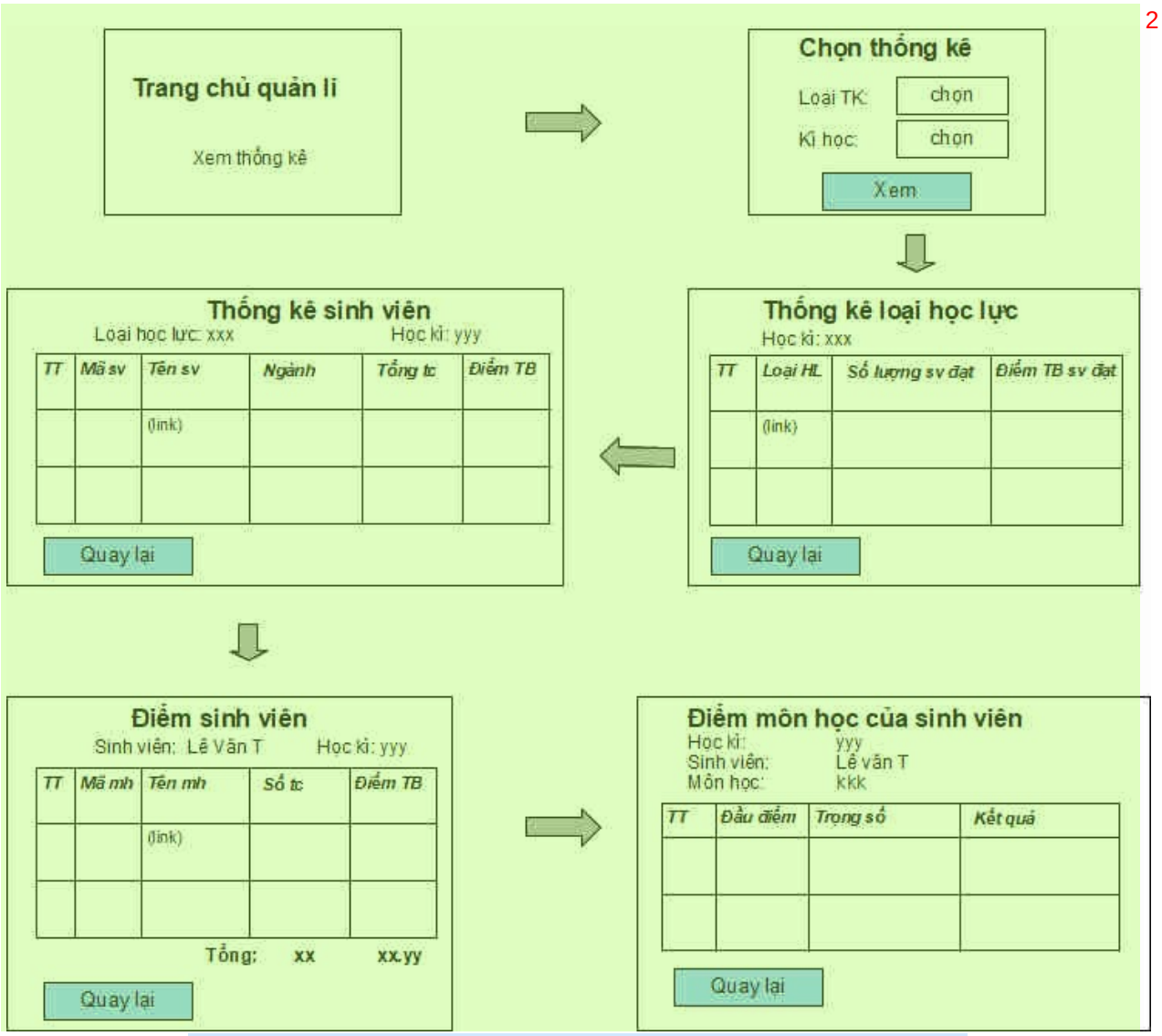
Hình 4.6: Biểu đồ lớp thiết kế modul nhập điểm

c. Thiết kế tĩnh modul xem thống kê loại học lực

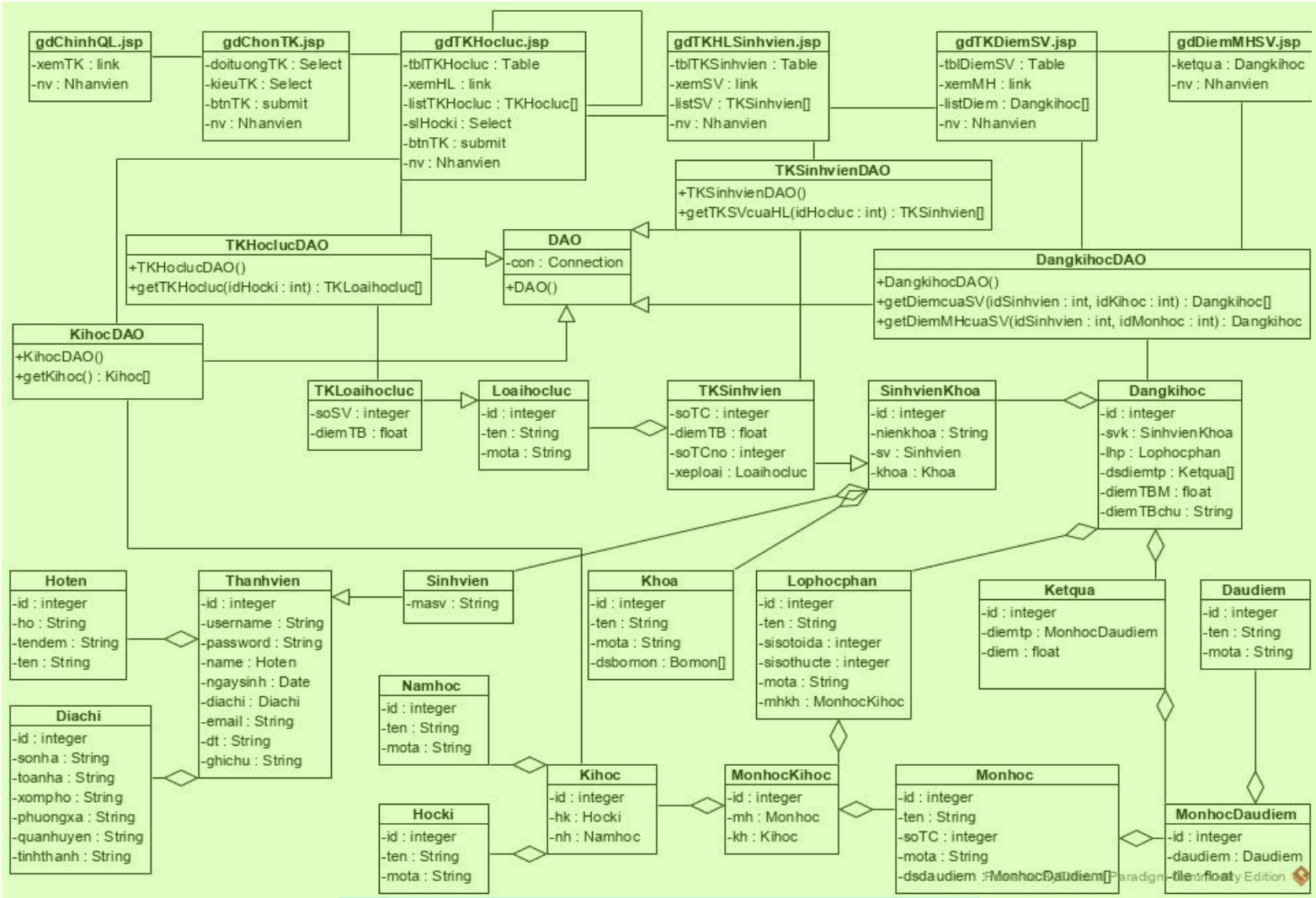
Giao diện cho modul được thiết kế như trong Hình 4.7. Thiết kế biểu đồ lớp cho modul được trình bày trong Hình 4.8:

- Tầng giao diện gồm các trang jsp: gdChinhQL, gdChonTK, gdTKHocluc, gdTKHLSinhvien, gdTKDiemSV, gdDiemMHSV.
- Tầng xử lí truy cập dữ liệu: DAO, KihocDAO, TKHoclucDAO, TKSinhvienDAO, DangkihocDAO.

- Các lớp thực thể liên quan. 1



Hình 4.7: Thiết kế giao diện cho modul xem thống kê loại học lực 3



Hình 4.8: Biểu đồ lớp thiết kế modul thống kê loại học lực

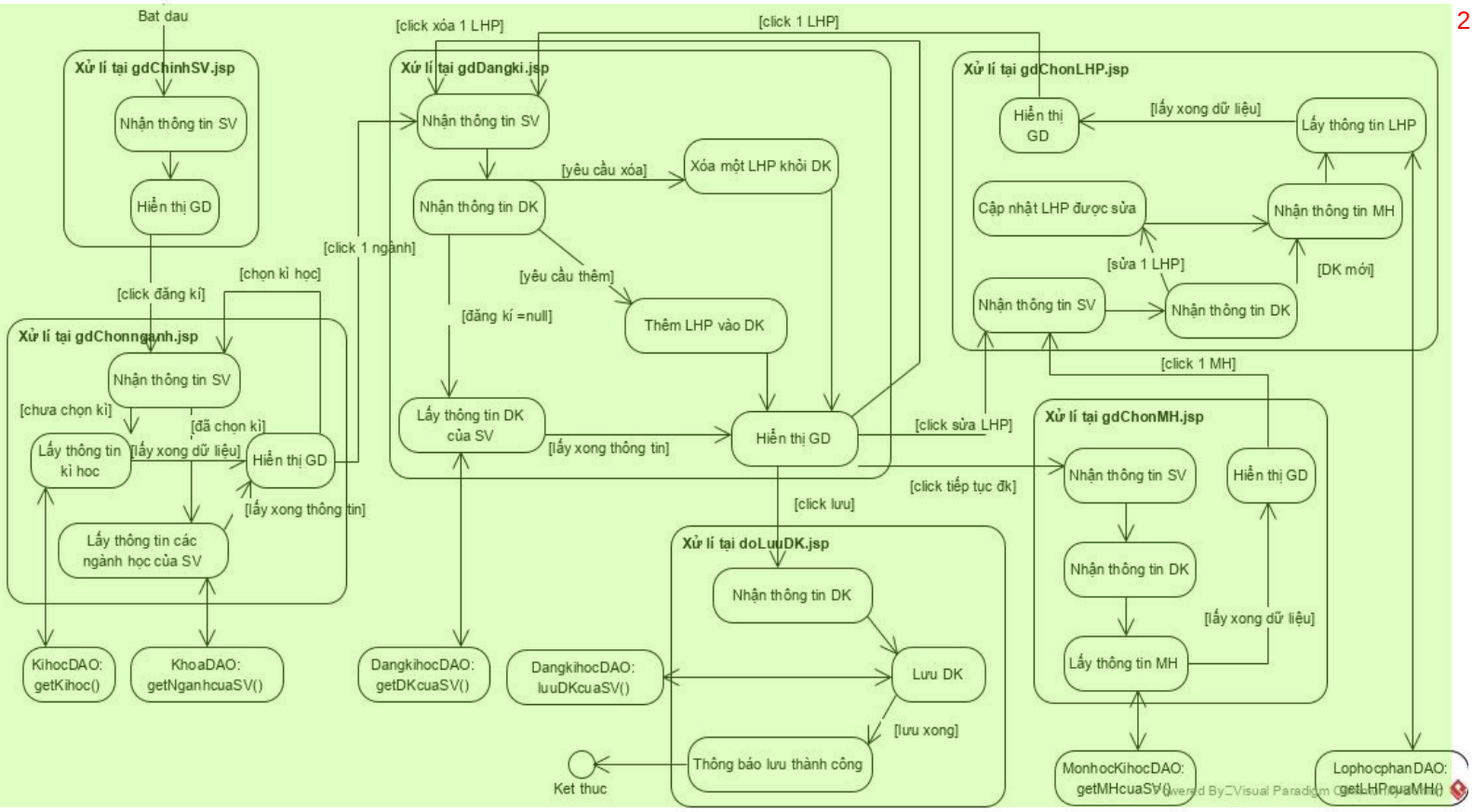
4.3.2 Thiết kế hoạt động

Quy trình tiến hành các bước như sau:

- Bước 1: Vẽ biểu đồ hoạt động của modul: Mỗi hành động tương ứng một phương thức đã thiết kế trong biểu đồ lớp.
- Bước 2: Viết kịch bản phiên bản 3 và vẽ biểu đồ tuần tự cho modul. Những lưu ý về yêu cầu khác biệt của biểu đồ tuần tự pha thiết kế so với biểu đồ tuần tự pha phân tích:
  - Tên các hành động của các lớp phải tương ứng với các phương thức đã thiết kế trong biểu đồ lớp thiết kế
  - Biểu diễn được phạm vi và khoảng thời gian hoạt động của từng phương thức trong biểu đồ.

Lưu ý là trong các kịch bản thiết kế cho tiết ở mục này, các bước đóng gói thuộc tính con của lớp thực thể đã được lược bỏ đi cho gọn.

a. Thiết kế động modul đăng kí học 1

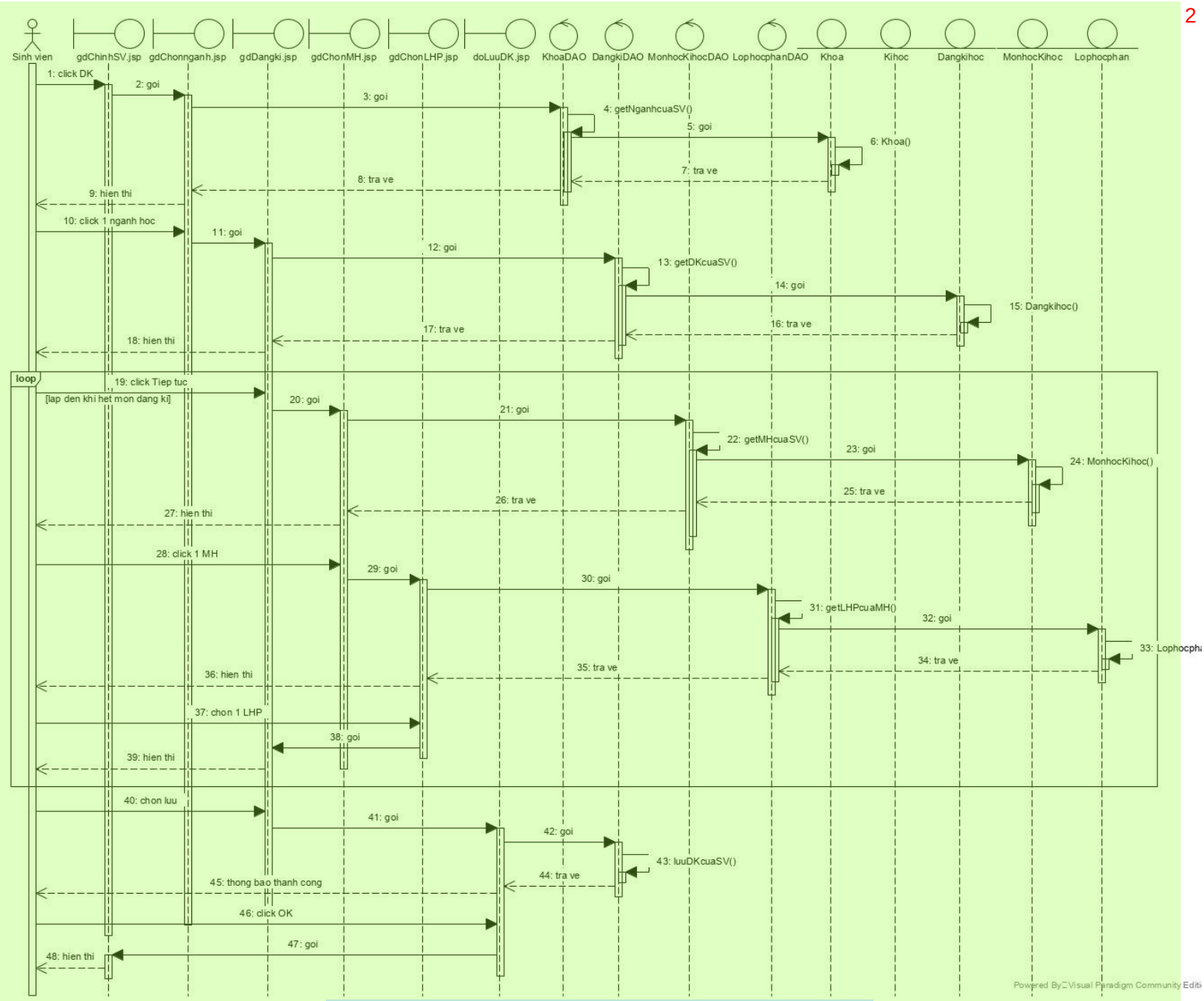


Hình 4.9: Biểu đồ thiết kế hoạt động modul đăng kí học 3

Biểu đồ hoạt động của modul được trình bày trong Hình 4.9. Biểu đồ tuần tự được trình bày trong Hình 4.10, với kịch bản cụ thể như sau: 4

1. Tại giao diện chính của sinh viên, sau khi đăng nhập, sinh viên click chọn chức năng đăng kí học. 5
2. Trang gdChinhSV.jsp gọi trang gdChonnganh.jsp.
3. Trang gdChonnganh.jsp gọi lớp KhoaDAO yêu cầu tìm danh sách ngành học của sinh viên.
4. Lớp KhoaDAO gọi hàm getNganhhoccuaSV()
5. Hàm getNganhhoccuaSV() gọi lớp Khoa để đóng gói thông tin
6. Lớp Khoa đóng gói thông tin thực thể
7. Lớp Khoa trả kết quả về cho hàm getNganhhoccuaSV()
8. Hàm getNganhhoccuaSV() trả kết quả cho trang gdChonnganh,jsp
9. Trang gdChonnganh.jsp hiển thị cho sinh viên
10. Sinh viên chọn 1 ngành học để đăng kí
11. Trang gdChonnganh.jsp gọi trang gdDangki.jsp

12. Trang gdDangki.jsp gọi lớp DangkihocDAO yêu cầu lấy thông tin đăng kí đã có của sinh viên 1
13. Lớp DangkihocDAO gọi hàm getDangkicuaSV()
14. Hàm getDangkicuaSV() gọi lớp Dangkihoc để đóng gói thông tin
15. Lớp Dangkihoc đóng gói thông tin thực thể
16. Lớp Dangkihoc trả kết quả lại cho hàm getDangkicuaSV()



Hình 4.10: Biểu đồ tuần tự thiết kế modul đăng kí học 3

17. Hàm getDangkicuaSV() trả kết quả lại cho trang gdDangki.jsp 4
18. Trang gdDangki.jsp hiển thị cho sinh viên.

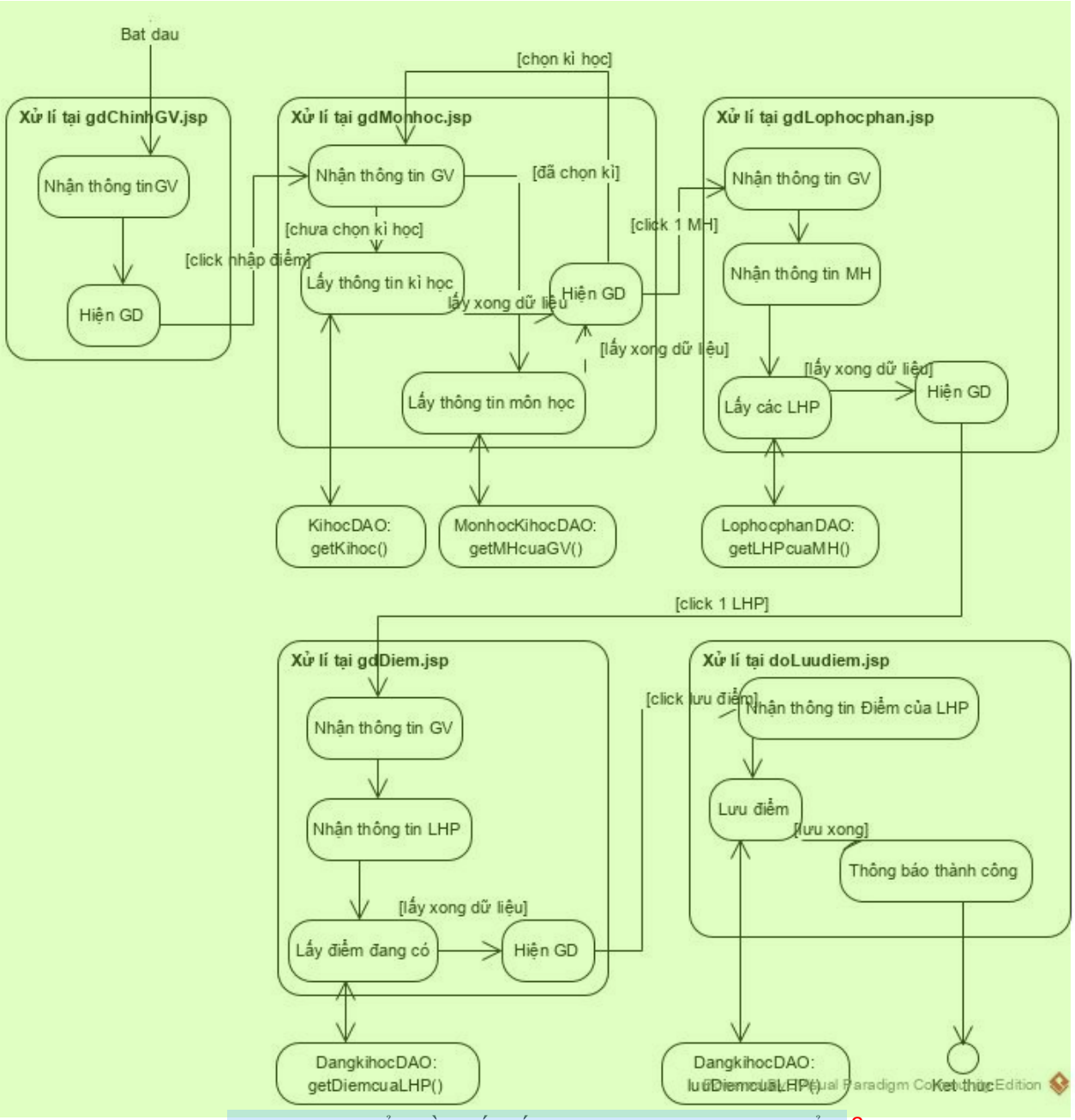
19. Sinh viên click tiếp tục để đăng kí.
20. Trang gdDangki.jsp gọi trang gdChonMH.jsp
21. Trang gdChonMH.jsp gọi lớp MonhocKihocDAO yêu cầu lấy danh sách môn học mà sinh viên được đăng kí
22. Lớp MonhocKihocDAO gọi hàm getMonhoccuaSV()
23. Hàm getMonhoccuaSV() thực hiện và gọi lớp MonhocKihoc để đóng gói
24. Lớp MonhocKihoc đóng gói thông tin thực thể
25. Lớp MonhocKihoc trả kết quả lại cho hàm getMonhoccuaSV()
26. Hàm getMonhoccuaSV() trả kết quả lại cho trang gdChonMH.jsp
27. Trang gdChonMH.jsp hiển thị cho sinh viên.
28. Sinh viên click chọn 1 môn học
29. Trang gdChonMH.jsp gọi trang gdChonLHP.jsp
30. Trang gdChonLHP.jsp gọi lớp LophocphanDAO yêu cầu tìm danh sách lớp học phần của môn học đã chọn.
31. Lớp LophocphanDAO gọi hàm getLHPcuaMH()
32. Hàm getLHPcuaMH() thực hiện và gọi lớp Lophocphan
33. Lớp Lophocphan đóng gói thông tin thực thể
34. Lớp Lophocphan trả lại kết quả cho hàm getLHPcuaMH()
35. Hàm getLHPcuaMH() trả kết quả cho trang gdChonLHP.jsp
36. Trang gdChonLHP.jsp hiển thị cho sinh viên
37. Sinh viên chọn 1 lớp học phần để đăng kí
38. Trang gdChonLHP.jsp gọi lại trang gdDangki.jsp
39. Trang gdDangki.jsp hiển thị cho sinh viên. (có thể lặp lại các bước 18 đến 38 cho đến khi đủ số tín chỉ tối thiểu)
40. Sinh viên click chọn lưu đăng kí
41. Trang gdDangki.jsp gọi trang doLuuDK.jsp
42. Trang doLuuDK.jsp gọi lớp DangkihocDAO yêu cầu lưu
43. Lớp DangkihocDAO gọi hàm luuDKcuaSV()
44. Lớp DangkihocDAO trả lại lớp doLuuDK.jsp
45. Lớp doLuuDK.jsp thông báo thành công
46. Sinh viên click OK của thông báo
47. Trang doLuuDK.jsp gọi lại trang gdChinhSV.jsp
48. Trang gdChinhSV.jsp hiển thị.

b. Thiết kế động modul nhập điểm

Biểu đồ hoạt động của modul nhập điểm được trình bày trong Hình 4.11. Tương ứng với biểu đồ tuần tự được trình bày trong Hình 4.12, kịch bản các hoạt động diễn ra như sau:

1. Tại trang gdChinhGV.jsp, sau khi đăng nhập, giảng viên chọn chức năng nhập điểm
2. Trang gdChinhGV.jsp gọi trang gdMonhoc.jsp
3. Trang gdMonhoc.jsp gọi lớp KihocDAO yêu cầu lấy thông tin các học kì đang học.

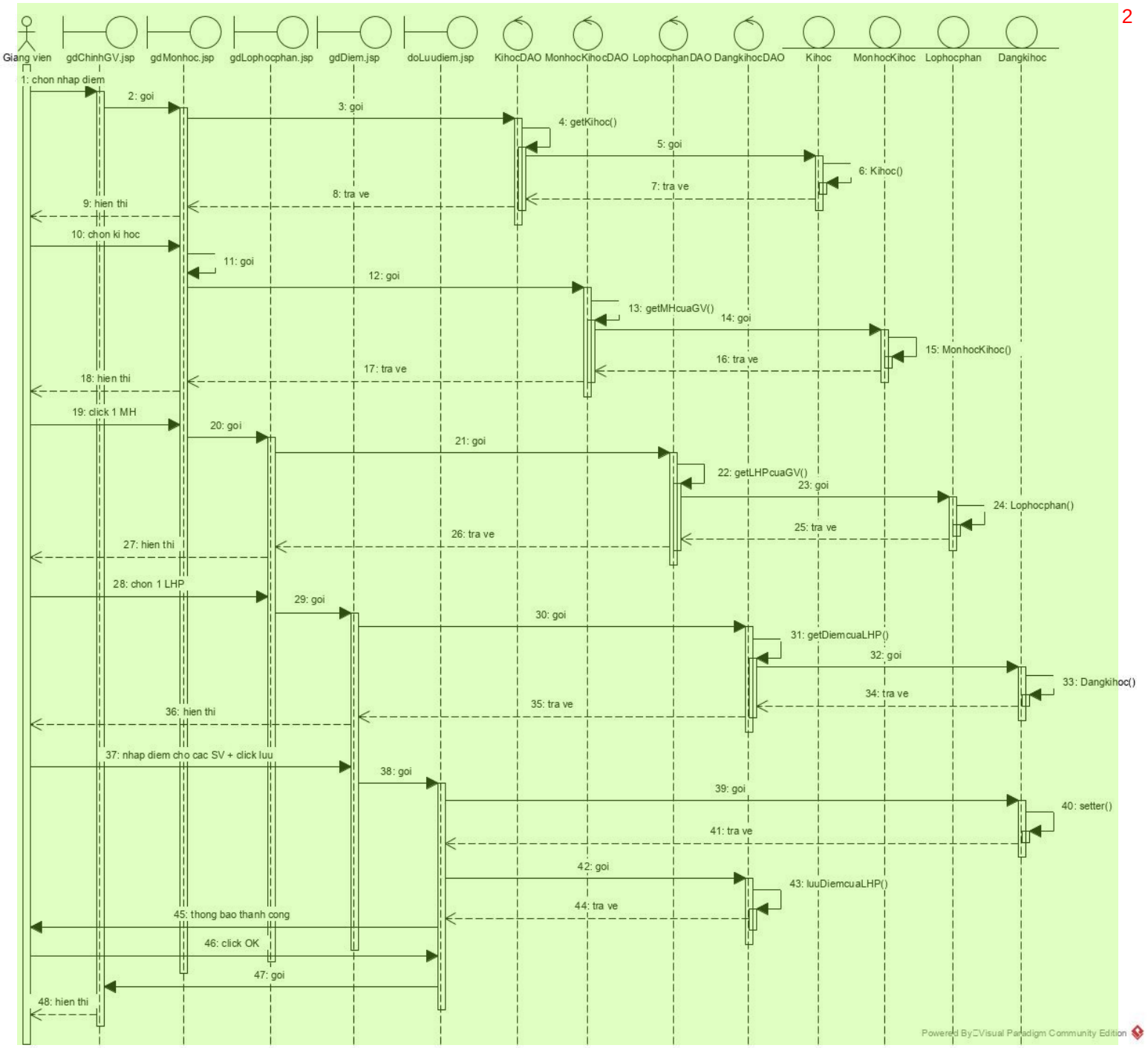
4. Lớp KihocDAO gọi hàm getKihoc() 1
5. Hàm getKihoc() thục hiện và gọi lớp Kihoc đóng gói thông tin
6. Lớp Kihoc đóng gói thông tin thực thể
7. Lớp Kihoc trả kết quả cho hàm getKihoc()
8. Hàm getKihoc() trả kết quả cho trang gdMonhoc.jsp
9. Trang gdMonhoc.jsp hiển thị cho giảng viên
10. Giảng viên chọn 1 kì học
11. Trang gdMonhoc.jsp submit gọi chính nó xử lí
12. Trang gdMonhoc.jsp gọi lớp MonhocKihocDAO yêu cầu lấy danh sách môn học do giảng viên dạy trong kì được chọn.
13. Lớp MonhocKihocDAO gọi hàm getMHcuaGV()
14. Hàm getMHcuaGV() thực hiện và gọi lớp MonhocKihoc để đóng gói thông tin
15. Lớp MonhocKihoc đóng gói thông tin thực thể
16. Lớp MonhocKihoc trả kết quả lại cho hàm getMHcuaGV()
17. Hàm getMHcuaGV() trả kết quả lại cho trang gdMonhoc.jsp
18. Trang gdMonhoc.jsp hiển thị cho giảng viên



Hình 4.11: Biểu đồ thiết kế hoạt động modul nhập điểm

19. Giảng viên click chọn 1 môn học
20. Trang gdMonhoc.jsp gọi trang gdLophocphan.jsp
21. Trang gdLophocphan.jsp gọi lớp LophocphanDAO yêu cầu lấy danh sách các lớp học phần của môn học đã chọn, do giảng viên đó giảng dạy
22. Lớp LophocphanDAO gọi hàm getLHPcuaGV()
23. Hàm getLHPcuaGV() thực hiện và gọi lớp Lophocphan để đóng gói thông tin
24. Lớp Lophocphan đóng gói thông tin thực thể
25. Lớp Lophocphan trả kết quả lại cho hàm getLHPcuaGV()
26. Hàm getLHPcuaGV() trả kết quả lại cho trang gdLophocphan.jsp

27. Trang gdLophocphan.jsp hiển thị cho giảng viên 1



Hình 4.12: Biểu đồ tuần tự thiết kế modul nhập điểm 3

28. Giảng viên click chọn 1 lớp học phần. 4

29. Trang gdLophocphan.jsp gọi trang gdDiem.jsp 5

30. Trang gdDiem.jsp gọi lớp DangkihocDAO yêu cầu lấy danh sách sinh viên đăng kí và kết quả đã có của lớp học phần đã chọn

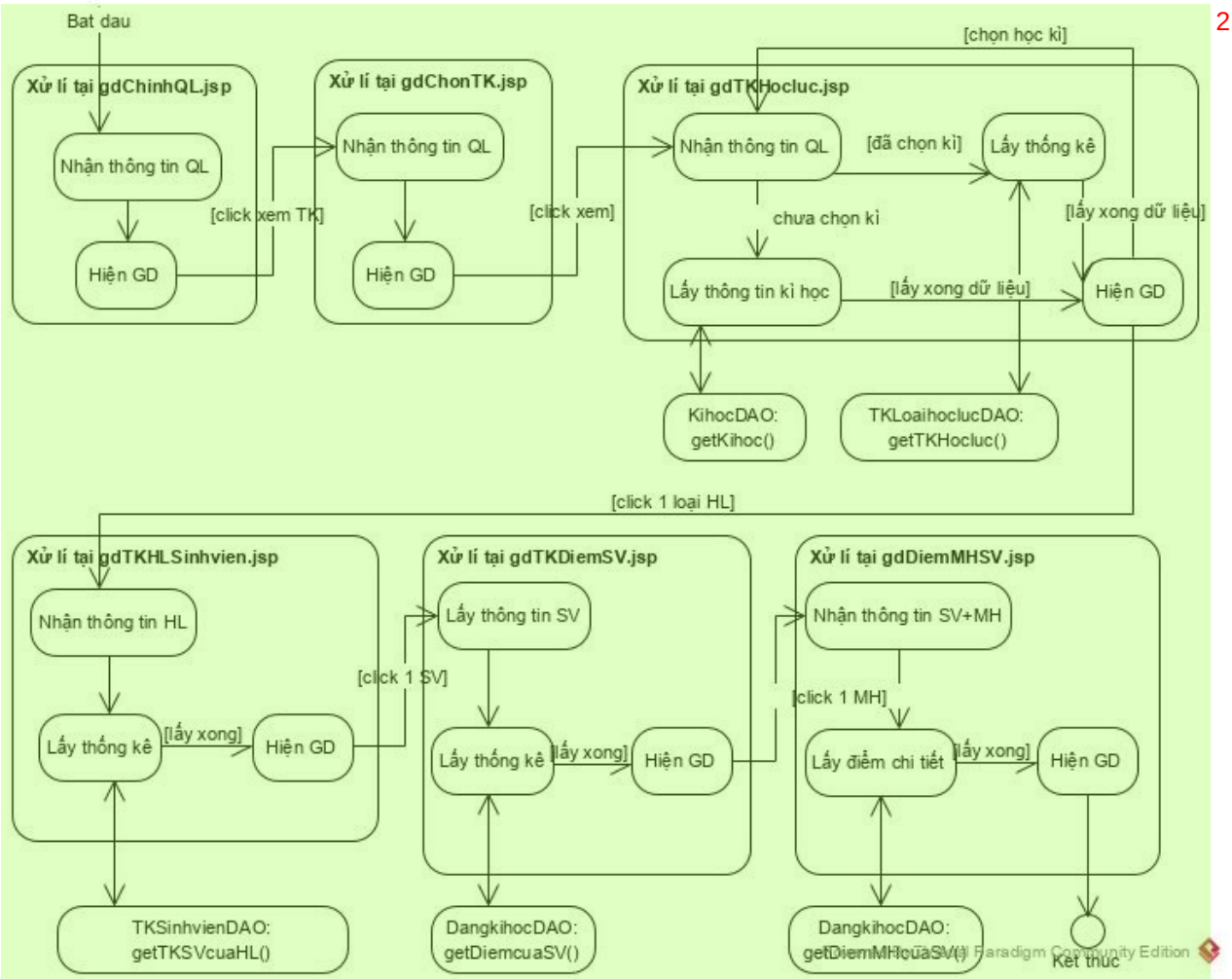
31. Lớp DangkihocDAO gọi hàm getDiemcuaLHP() 1
32. Hàm getDiemcuaLHP() thực hiện và gọi lớp Dangkihoc để đóng gói thông tin
33. Lớp Dangkihoc đóng gói thông tin thực thể
34. Lớp Dangkihoc trả kết quả cho hàm getDiemcuaLHP()
35. Hàm getDiemcuaLHP() trả kết quả cho trang gdDiem.jsp
36. Trang gdDiem.jsp hiển thị cho giảng viên
37. Giảng viên nhập các đầu điểm cho sinh viên và click lưu
38. Trang gdDiem.jsp gọi doLuudiem.jsp
39. Trang doLuudiem.jsp gọi lớp Dangkihoc yêu cầu set lại accs thuộc tính điểm mới nhập.
40. Lớp Dangkihoc gọi các hàm set thuộc tính điểm tương ứng
41. Lớp Dangkihoc trả đối tượng về cho trang doLuudiem.jsp
42. Trang doLuudiem.jsp gọi lớp DangkihocDAO yêu cầu lưu điểm cho lớp học phần
43. Lớp DangkihocDAO gọi hàm luuDiemcuaLHP()
44. Lớp DangkihocDAO trả kết quả lại cho trang doLuudiem.jsp
45. Trang doLuudiem.jsp hiện thống báo lưu thành công
46. Giảng viên click vào OK của thông báo
47. Trang doLuudiem.jsp gọi trang gdChinhGV.jsp
48. Trang gdChinhGV.jsp hiển thị

c. Thiết kế động cho modul xem thống kê loại học lực 2

Biểu đồ hoạt động của modul xem thống kê loại học lực được trình bày trong Hình 4.13. Tương ứng với biểu đồ tuần tự được trình bày trong Hình 4.14, kịch bản các hoạt động diễn ra như sau: 3

1. Tại trang gdChinhQL.jsp, sau khi đăng nhập, nhân viên quản lí chọn chức năng xem thống kê 4
2. Trang gdChinhQL.jsp gọi trang gdChonTK.jsp
3. Trang gdChonTK.jsp hiển thị cho NV quản lí
4. NV quản lí click xem thống kê loại học lực
5. Trang gdChonTK.jsp gọi trang gdTKHocluc.jsp
6. Trang gdTKHocluc.jsp gọi lớp KihocDAO yêu cầu lấy danh sách các kì đã học
7. Lớp KihocDAO gọi hàm getKihoc()
8. Hàm getKihoc() thực hiện và gọi lớp Kihoc đóng gói thông tin
9. Lớp Kihoc đóng gói thông tin thực thể.
10. Lớp Kihoc trả kết quả về cho hàm getKihoc()
11. Hàm getKihoc() trả kết quả về cho trang gdTKHocluc.jsp
12. Trang gdTKHocluc.jsp hiển thị cho NV quản lí
13. NV quản lí click chọn 1 kì học để xem thống kê
14. Trang gdTKHocluc.jsp submit vào chính nó để xử lí
15. Trang gdTKHocluc.jsp gọi lớp TKLoaihoclucDAO yêu cầu lấy thông tin thống kê
16. Lớp TKLoaihoclucDAO gọi hàm getTKHocluc()
17. Hàm getTKHocluc() thực hiện và gọi lớp TKLoaihocluc đóng gói thông tin

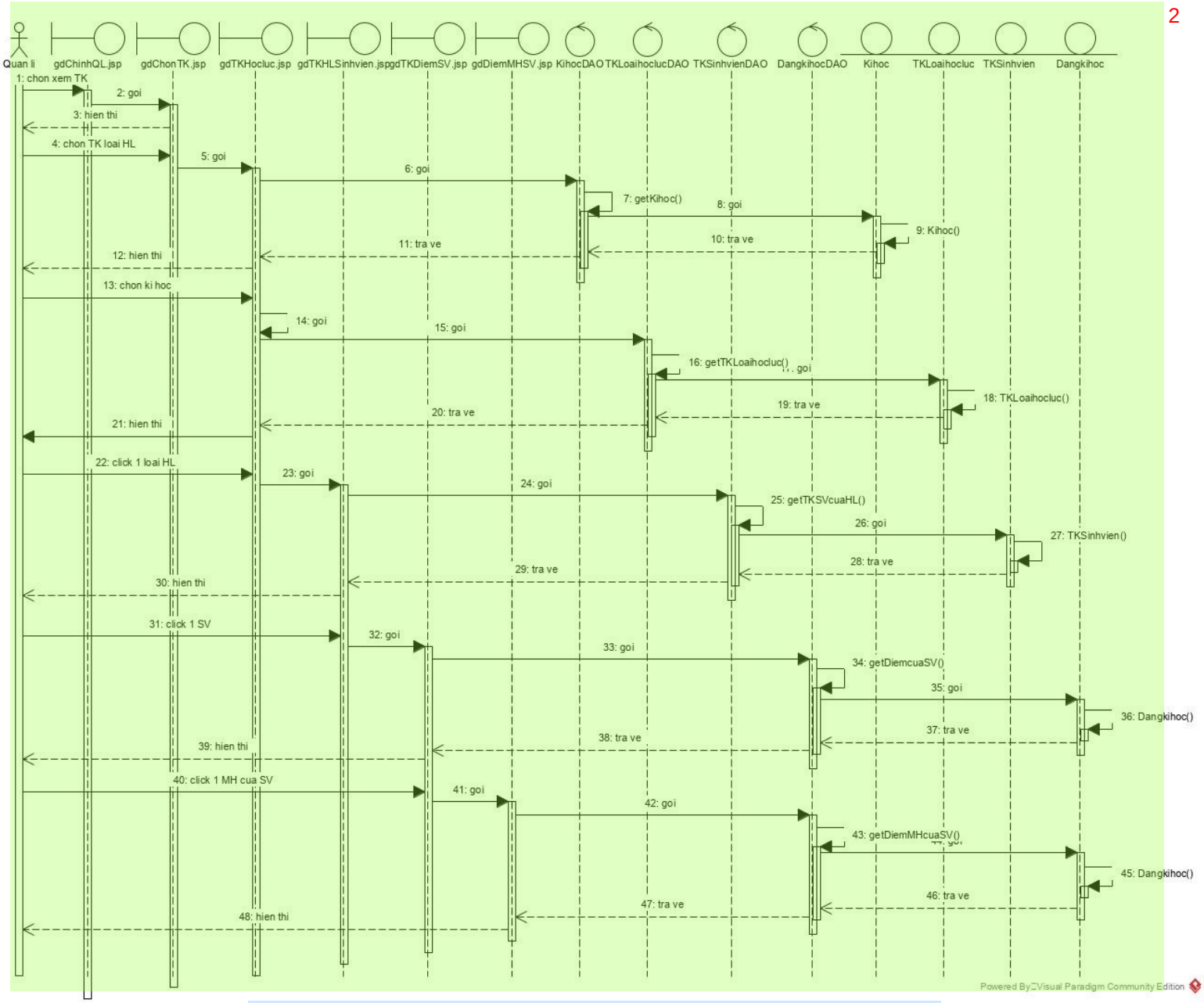
18. Lớp TKLoaihocluc đóng gói thông tin thực thể 1
19. Lớp TKLoaihocluc trả kết quả cho hàm getTKHocluc()
20. Hàm getTKHocluc() trả kết quả cho trang gdTKHocluc.jsp
21. Trang gdTKHocluc.jsp hiển thị cho NV quản lí
22. NV quản lí click chọn 1 loại học lực
23. Trang gdTKHocluc.jsp gọi trang gdTKHLSinhvien.jsp
24. Trang gdTKHLSinhvien.jsp gọi lớp TKSinhvienDAO yêu cầu lấy thông tin thống kê



Hình 4.13: Biểu đồ thiết kế hoạt động modul xem thống kê loại học lực 3

25. Lớp TKSinhvienDAO gọi hàm getTKSVcuaHL() 4
26. Hàm getTKSVcuaHL() thwucj hiện và gọi lớp TKSinhvien đóng gói thông tin
27. Lớp TKSinhvien đóng gói thông tin thực thể
28. Lớp TKSinhvien trả kết quả lại cho hàm getTKSVcuaHL()
29. Hàm getTKSVcuaHL() trả kết quả lại cho trang gdTKHLSinhvien.jsp
30. Trang gdTKHLSinhvien.jsp hiển thị cho NV quản lí
31. NV quản lí click 1 sinh viên

32. Trang gdTKHLSinhvien.jsp gọi trang gdTKDiemSV.jsp
33. Trang gdTKDiemSV.jsp gọi lớp DangkihocDAO yêu cầu lấy thông tin điểm các môn học của sinh viên
34. Lớp DangkihocDAO gọi hàm getDiemcuaSV()
35. Hàm getDiemcuaSV thực hiện và gọi lớp Dangkihoc đóng gói thông tin
36. Lớp Dangkihoc đóng gói thông tin thực thể
37. Lớp Dangkihoc trả kết quả cho hàm getDiemcuaSV()
38. Hàm getDiemcuaSV() trả kết quả lại cho trang gdTKDiemSV.jsp
39. Trang gdTKDiemSV.jsp hiển thị cho NV quản lí
40. NV quản lí click vào một môn học



Hình 4.14: Biểu đồ tuần tự thiết kế modul xem thống kê loại học lực

41. Trang gdTKDiemSV.jsp gọi trang gdDiemMHSV.jsp 1
42. Trang gdDiemMHSV.jsp gọi lớp DangkihocDAO yêu cầu lấy thông tin điểm chi tiết của môn học
43. Lớp DangkihocDAO gọi hàm getDiemMHcuaSV()
44. Hàm getDiemMHcuaSV() thực hiện và gọi lớp Dangkihoc đóng gói thông tin
45. Lớp Dangkihoc đóng gói thông tin thực thể
46. Lớp Dangkihoc trả lại cho hàm getDiemMHcuaSV()
47. Hàm getDiemMHcuaSV() trả kết quả cho trang gdDiemMHSV.jsp
48. Trang gdDiemMHSV.jsp hiển thị cho NV quản lí

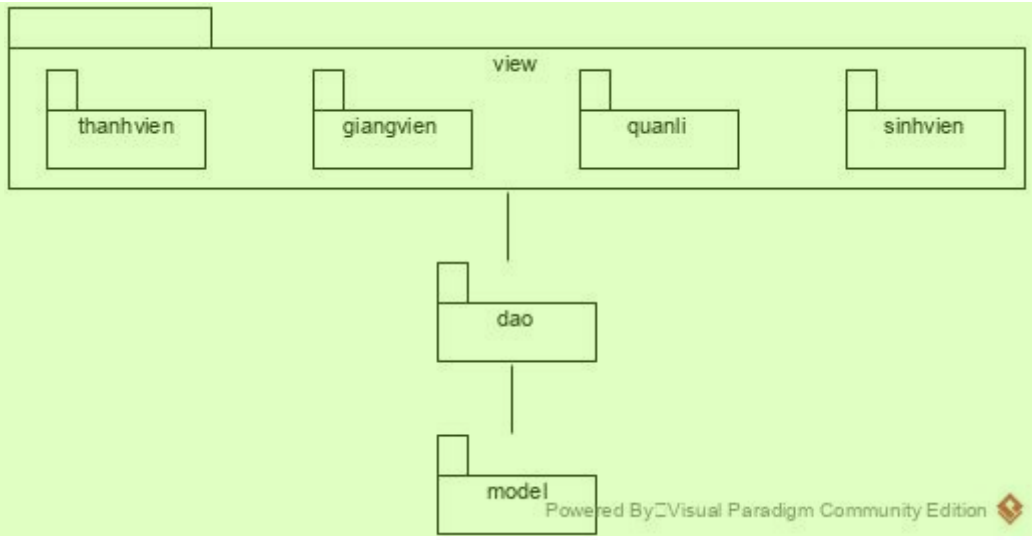
4.3.3 Thiết kế triển khai 2

Quy trình tiến hành các bước như sau: 3

- Bước 1: Vẽ biểu đồ triển khai của hệ thống: Các máy sẽ cài đặt hệ thống. Mỗi máy đặt các modul hay package nào. Các modul/package tương tác với nhau thế nào. 4

Áp dụng cho hệ thống, các package được thiết kế theo cấu trúc trong Hình 4.15: 5

- Các lớp thực thể đặt chung trong gói model 6
- Các lớp DAO đặt chung trong gói dao.
- Các trang jsp để trong gói view. Gói view được chia nhỏ thành các gói nhỏ hơn tương ứng với các giao diện cho các người dùng khác nhau:
  - Các trang thao tác cho đăng nhập, quản lí mật khẩu đặt trong gói thanhvien.
  - Các trang cho chức năng liên quan đến giảng viên đặt trong gói giangvien
  - Các trang cho chức năng liên quan đến nhân viên quản lí đặt trong gói quanli
  - Các trang cho chức năng liên quan đến sinh viên đặt trong gói sinhvien



1

Hình 4.15: Biểu đồ thiết kế gói của hệ thống 2

4.4 CÂU HỎI ÔN TẬP 3

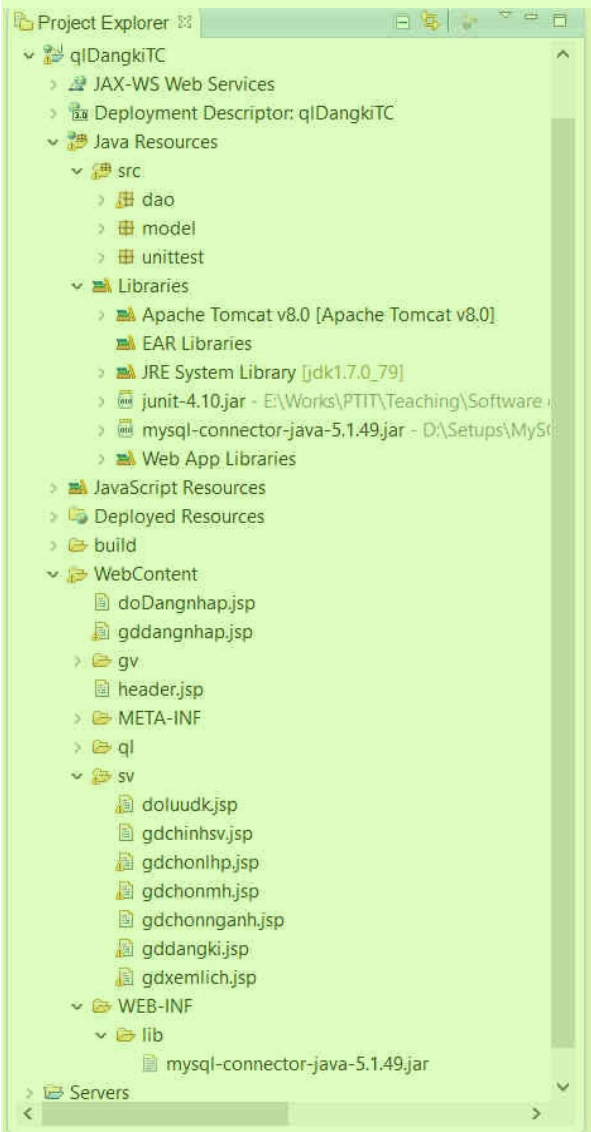
1. Thiết kế thực thể cho toàn hệ thống 4
2. Thiết kế CSDL cho toàn hệ thống
3. Thiết kế giao diện liên quan cho modul
4. Thiết kế biểu đồ lớp cho modul
5. Thiết kế biểu đồ hoạt động cho modul
6. Thiết kế biểu đồ tuần tự cho modul
7. Thiết kế biểu đồ gói cho toàn hệ thống

CHƯƠNG 5: CÀI ĐẶT HỆ THỐNG [1]

5.1 TỔ CHỨC DỰ ÁN [2]

Dự án trong Eclipse được tổ chức như Hình 5.1: [3]

- Các lớp thực thể và điều khiển để trong thư mục \java resources, chia nhỏ thành các package con tương ứng: model, dao. [4]
- Các trang jsp để trong thư mục \Web content, phần xử lí đăng nhập để ở thu mục chung này, các phần xử lí riêng có thể để vào các thư mục con. Ví dụ, phần xử lí các chức năng của sinh viên thì để vào thư mục con \sv
- Các thư viện (file .jar) của kết nối CSDL, Junit thì add vào phần libraries của java resources, đồng thời copy vào thư mục \Web content\WEB-INF\lib



Hình 5.1: Tổ chức thư mục dự án web trong Eclipse [6]

5.2 CÀI ĐẶT CÁC MODUL 1

Nội dung phần này sẽ trình bày cài đặt 3 chức năng: sinh viên đăng kí, giảng viên nhập điểm LHP, và quản lí xem thống kê loại học lực. Tuy nhiên: 2

- Phần thực thể không trình bày code ở đây và coi như bài tập đơn giản cho người đọc. 3
- Chức năng nhập điểm và xem thống kê chỉ được cài đặt phần DAO, phần JSP cũng coi như là bài tập cho người đọc.
- Riêng chức năng sinh viên đăng kí được trình bày đầy đủ DAO và các trang JSP.
- Phần JSP chỉ sử dụng JSP đơn giản nhất mà chưa áp dụng framwork nào vào với mục đích chính là minh họa nghiệp vụ. Cho nên cũng chưa có xử lí html/css. Người đọc có thể áp dụng thêm vào để tăng tính thẩm mỹ của phần giao diện.
- Người đọc cần chỉnh sửa thông tin kết nối vào CSDL (dbUrl và dbClass, username, password) trong lớp DAO cho phù hợp với cấu hình CSDL trên máy mình
- Nếu người đọc đổi tên bảng/tên cột trong CSDL thì phải cập nhật tương ứng trong các câu lệnh SQL.
- Các lớp DAO có sử dụng store procedure thì ngay sau code của lớp sẽ là code của các Store Procedure tương ứng được gọi trong lớp đó.

5.2.1 Chức năng đăng kí học 4

Lớp DAO.java 5

6

```
package dao;
import java.sql.Connection;
import java.sql.DriverManager;

public class DAO {
public static Connection con;
	public DAO(){
		if(con == null){
			String dbUrl = "jdbc:mysql://localhost:3307/qldktc?
			    autoReconnect=true&useSSL=false";
			String dbClass = "com.mysql.jdbc.Driver";

			try {
				Class.forName(dbClass);
				con = DriverManager.getConnection (dbUrl, "root",
				     "Cnpm@2020?");
			}catch(Exception e) {
				e.printStackTrace();
			}
		}
```

```
        }
}
```

1

Lớp ThanhvienDAO.java 2

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import model.Hoten;
import model.Thanhvien;

public class ThanhvienDAO extends DAO{
        public ThanhvienDAO() {
                super();
                // TODO Auto-generated constructor stub
        }

        public boolean kiemtraDangnhap(Thanhvien tv){
                boolean kq = false;
                if(tv.getUsername().contains("true") ||
                  tv.getUsername().contains("=")||
                  tv.getPassword().contains("true") ||
                  tv.getPassword().contains("=")) return false;
                String sql = "{call kiemtraDN(?,?)}"; //su dung stored procedure
                try{
                        CallableStatement cs = con.prepareCall(sql);
                        cs.setString(1,tv.getUsername());
                        cs.setString(2,tv.getPassword());
                        ResultSet rs = cs.executeQuery();

                        if(rs.next()){
                                tv.setId(rs.getInt("id"));
                                tv.setVaitro(rs.getString("vaitro"));
                                //hoten
                                Hoten ht = new Hoten();
                                ht.setHodem(rs.getString("hodem"));
                                ht.setTen(rs.getString("ten"));
                                tv.setHoten(ht);
                                kq = true;
                        }
                }catch(Exception e){
                        e.printStackTrace();
                        kq = false;
                }
                return kq;
        }
}
```

3

4

5

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `kiemtraDN`(IN usr VARCHAR(255), IN pwd
VARCHAR(255))
BEGIN
	SELECT * FROM tblthanhvien
		WHERE username = usr AND password = pwd;
END
```

1

Lớp KihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Hocki;
import model.Kihoc;
import model.Namhoc;

public class KihocDAO extends DAO{

	public KihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * Lay danh sach cac hoc ki dang mo cho sinh vien chon dang ki hoc
	 * @return danh sach ki hoc dang mo dang ki
	 */
	public ArrayList<Kihoc> getKihocdangki(){
		ArrayList<Kihoc> kq=null;
		String sql = "{call kiDangDK()}"; //su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Kihoc>();
				Kihoc kh = new Kihoc();
				kh.setId(rs.getInt("idkihoc"));
				//nam hoc
				Namhoc nh = new Namhoc();
				nh.setId(rs.getInt("idnamhoc"));
				nh.setTen(rs.getString("namhoc"));
				kh.setNamhoc(nh);
				//hoc ki
				Hocki hk = new Hocki();
```

1

```
                        hk.setId(rs.getInt("idhocki"));
                        hk.setTen(rs.getString("hocki"));
                        kh.setHocki(hk);
                        kq.add(kh);
                    }
                }catch(Exception e){
                    e.printStackTrace();
                    kq = null;
                }
                return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `kiDangDK`()
BEGIN
        SELECT a.id as idnamhoc, a.ten as namhoc, b.id as idhocki,
        b.ten as hocki, c.id as idkihoc
        FROM tblnamhoc a, tblhocki b, tblkihoc c
        WHERE c.idnamhoc = a.id AND c.idhocki = b.id AND c.dangdk = 1;
END
```

Lớp SinhvienKhoaDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Hoten;
import model.Khoa;
import model.Sinhvien;
import model.SinhvienKhoa;

public class SinhvienKhoaDAO extends DAO{
        public SinhvienKhoaDAO() {
                super();
                // TODO Auto-generated constructor stub
        }

        /**
         * lay danh sach cac khoa/nganh ma sinh vien dang hoc de chon dang ki hoc
         * @param idSV: id cua sinh vien
         * @return danh sach cacs doi tuong SinhvienKhoa
         */
        public ArrayList<SinhvienKhoa> getNganhcuaSV(int idSV){
                ArrayList<SinhvienKhoa> kq = null;
                String sql = "{call nganhcuaSV(?)}";// su dung stored procedure
                try{
                        CallableStatement cs = con.prepareCall(sql);
```

1

```
                    cs.setInt(1, idSV);
                    ResultSet rs = cs.executeQuery();

                    while(rs.next()){
                          if(kq == null) kq = new ArrayList<SinhvienKhoa>();
                          SinhvienKhoa svk = new SinhvienKhoa();
                          svk.setId(rs.getInt("idsvk"));
                          // sinh vien
                          Sinhvien sv = new Sinhvien();
                          sv.setId(rs.getInt("id"));
                          sv.setMasv(rs.getString("masv"));
                          //hoten
                          Hoten ht = new Hoten();
                          ht.setHodem(rs.getString("hodem"));
                          ht.setTen(rs.getString("ten"));
                          sv.setHoten(ht);
                          svk.setSinhvien(sv);
                          //khoa
                          Khoa k = new Khoa();
                          k.setId(rs.getInt("idkhoa"));
                          k.setTen(rs.getString("tenkhoa"));
                          svk.setKhoa(k);
                          kq.add(svk);
                    }
              }catch(Exception e){
                    e.printStackTrace();
                    kq = null;
              }
              return kq;
       }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `nganhcuaSV`(IN idSV INT)
BEGIN
       SELECT a.id as idkhoa, a.ten as tenkhoa, b.id as idsvk,
    c.masv, d.*
       FROM tblkhoa a, tblsinhvienkhoa b, tblsinhvien c, tblthanhvien d
       WHERE b.idsinhvien = idSV AND b.idkhoa = a.id
    AND b.danghoc = 1 AND c.idthanhvien =idSV
    AND d.id = idSV;
END
```

Lớp MonhocKihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
```

```
import model.Kihoc;
import model.Monhoc;
import model.MonhocKihoc;

public class MonhocKihocDAO extends DAO{
	public MonhocKihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * Lay danh sach cac mon hoc duoc day trong hoc ki @idKihoc
	 * @param idSVK: id cua sinh vien theo khoa/nganh hoc
	 * @param idKihoc: id cua ki hoc dang chon
	 * @return danh sach cac mon hoc duoc phep
	 */
	public ArrayList<MonhocKihoc> getMHcuaSV(int idSVK, int idKihoc){
		ArrayList<MonhocKihoc> kq = null;
		String sql = "{call MHmaSVduocDK(?,?)}";// su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1, idSVK);
			cs.setInt(2, idKihoc);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<MonhocKihoc>();
				MonhocKihoc mhkh = new MonhocKihoc();
				mhkh.setId(rs.getInt("idmhkh"));
				//mon hoc
				Monhoc mh = new Monhoc();
				mh.setId(rs.getInt("idmh"));
				mh.setTen(rs.getString("tenmh"));
				mh.setSoTC(rs.getInt("sotc"));
				mhkh.setMonhoc(mh);
				//ki hoc
				Kihoc kh = new Kihoc();
				kh.setId(idKihoc);
				mhkh.setKihoc(kh);
				kq.add(mhkh);
			}
		}catch(Exception e){
			e.printStackTrace();
			kq = null;
		}
		return kq;
	}
```

1

```
}
CREATE DEFINER=`root`@`localhost` PROCEDURE `MHmaSVduocDK`(IN idSVK INT, IN idKihoc
INT)
BEGIN
       DROP TEMPORARY TABLE IF EXISTS ketquamon;
       CREATE TEMPORARY TABLE ketquamon AS
       SELECT l.idmonhoc, SUM(k.diem*l.tile) as ketqua
    FROM tblmonhoc h, tbldangkihoc i, tblketqua k, tblmonhocdaudiem l
       WHERE i.idsinhvienkhoa =idSVK AND k.iddangkihoc = i.id
    AND l.id = k.idmonhocdaudiem
    GROUP BY l.idmonhoc;

    DROP TEMPORARY TABLE IF EXISTS idmondaqua;
       CREATE TEMPORARY TABLE idmondaqua AS
    SELECT idmonhoc FROM ketquamon WHERE ketqua >=4;

    DROP TEMPORARY TABLE IF EXISTS idmonchuaqua;
       CREATE TEMPORARY TABLE idmonchuaqua AS
    SELECT idmonhoc FROM ketquamon WHERE ketqua <4;

    SELECT a.id as idmhkh, b.id as idmh, b.ten as tenmh, b.sotc
    FROM tblmonhockihoc a, tblmonhoc b
    WHERE a.idkihoc = idKihoc AND b.id = a.idmonhoc
    AND b.id NOT IN (SELECT idmonhoc FROM ketquamon WHERE ketqua >=4)
    AND (b.id IN (SELECT idmonhoc FROM idmonchuaqua) OR
             ((SELECT g.idmontienquyet
        FROM tblmontienquyet g
        WHERE g.idmonhoc = b.id) IN (SELECT idmonhoc FROM idmondaqua))
        OR b.id NOT IN (SELECT idmonhoc FROM tblmontienquyet));
END
```

Lớp LophocphanDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Kihoc;
import model.Lophocphan;
import model.Monhoc;
import model.MonhocKihoc;

public class LophocphanDAO extends DAO{
       public LophocphanDAO() {
              super();
              // TODO Auto-generated constructor stub
       }
```

```
	/**
	 * lay danh sach cac lop hoc phan cua @idMHKH ma sinh vien @idSVK
	 * @param idMHKH: id cua mon hoc ki hoc
	 * @param idSVK: id cua sinh vien khoa
	 * @return: danh sach lop hoc phan con co the dang ki duoc
	 */
	public ArrayList<Lophocphan> getLHPchoSVdangki(int idSVK, int idMHKH){
		ArrayList<Lophocphan> kq = null;
		String sql = "{call LHPchoSVdangki(?,?)}";// su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1, idSVK);
			cs.setInt(2, idMHKH);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Lophocphan>();
				Lophocphan lhp = new Lophocphan();
				lhp.setId(rs.getInt("id"));
				lhp.setTen(rs.getString("ten"));
				lhp.setSisotoida(rs.getInt("sisotoida"));
				lhp.setSisothucte(rs.getInt("sisothuc"));
				//monhockihoc
				MonhocKihoc mhkh = new MonhocKihoc();
				mhkh.setId(idMHKH);
				//monhoc
				Monhoc mh = new Monhoc();
				mh.setId(rs.getInt("idmh"));
				mh.setTen(rs.getString("tenmh"));
				mh.setSoTC(rs.getInt("sotc"));
				mhkh.setMonhoc(mh);
				//kihoc
				Kihoc kh = new Kihoc();
				kh.setId(rs.getInt("idkihoc"));
				mhkh.setKihoc(kh);
				lhp.setMonhocKihoc(mhkh);
				kq.add(lhp);
			}
		}catch(Exception e){
			e.printStackTrace();
			kq = null;
		}
		return kq;
	}
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `LHPchoSVdangki`(IN idSVK int,IN idMHKH
```

```
int)
BEGIN
      DECLARE idKH INT;
    SELECT idkihoc INTO idKH
    FROM tblmonhockihoc WHERE id = idMHKH;

      DROP TEMPORARY TABLE IF EXISTS slotdadangki;
      CREATE TEMPORARY TABLE slotdadangki AS
    SELECT a.idtuan, a.idngay, a.idkip
    FROM tbllichhoc a, tbllophocphan b, tbldangkihoc c, tblmonhockihoc d
    WHERE d.idkihoc = idKH AND b.idmonhockihoc = d.id
    AND a.idlophocphan = b.id AND c.idlophocphan = b.id
    AND c.idsinhvienkhoa = idSVK;

    DROP TEMPORARY TABLE IF EXISTS lhpchuaday;
      CREATE TEMPORARY TABLE lhpchuaday AS
    SELECT a.id, a.ten, a.sisotoida, a.idmonhockihoc,
    COUNT(e.id) as sisothuc
    FROM tbllophocphan a LEFT JOIN tbldangkihoc e
    ON e.idlophocphan = a.id
    WHERE a.idmonhockihoc = idMHKH
   GROUP BY a.id
   HAVING a.sisotoida > sisothuc;

    SELECT a.*, b.idkihoc, d.id as idmh, d.ten as tenmh, d.sotc
    FROM lhpchuaday a, tblmonhockihoc b, tblmonhoc d
    WHERE a.id NOT IN
             (SELECT b.id FROM tbllichhoc b, slotdadangki c
             WHERE b.idtuan = c.idtuan AND
        b.idngay = c.idngay AND b.idkip = c.idkip)
      AND b.id = a.idmonhockihoc AND d.id = b.idmonhoc;
END
```

Lớp LichhocDAO.java

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Giangvien;
import model.Hoten;
import model.Kiphoc;
import model.Lichhoc;
import model.Lophocphan;
import model.Ngayhoc;
import model.Phonghoc;
import model.Tuanhoc;
```

```
public class LichhocDAO extends DAO{
	public LichhocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * lay danh sach cac kip hoc cua lop hoc phan @idLHP
	 * @param idLHP: id lop hoc phan
	 * @return danh sach cac kip hoc, moi kip hoc tuong ung mot  lich hoc
	 */
	public ArrayList<Lichhoc> getLHcuaLHP(int idLHP){
		ArrayList<Lichhoc> kq = null;
		String sql = "{call LichLHP(?)}";// su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1, idLHP);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Lichhoc>();
				Lichhoc lh = new Lichhoc();
				lh.setId(rs.getInt("id"));
				lh.setTen(rs.getString("ten"));
				//lop hoc phan
				Lophocphan lhp = new Lophocphan();
				lhp.setId(idLHP);
				lh.setLophocphan(lhp);
				//giang vien
				Giangvien gv = new Giangvien();
				gv.setId(rs.getInt("idgiangvien"));
				Hoten ht = new Hoten();
				ht.setHodem(rs.getString("hodemgv"));
				ht.setTen(rs.getString("tengv"));
				gv.setHoten(ht);
				lh.setGiangvien(gv);
				//Phong hoc
				Phonghoc ph = new Phonghoc();
				ph.setId(rs.getInt("idphonghoc"));
				ph.setTen(rs.getString("tenphong"));
				lh.setPhonghoc(ph);
				//Tuan hoc
				Tuanhoc tu = new Tuanhoc();
				tu.setId(rs.getInt("idtuan"));
				tu.setTen(rs.getString("tentuan"));
				lh.setTuanhoc(tu);
				//Ngay hoc
```

1

```
                        Ngayhoc ng = new Ngayhoc();
                        ng.setId(rs.getInt("idngay"));
                        ng.setTen(rs.getString("tenngay"));
                        lh.setNgayhoc(ng);
                        //Kip hoc
                        Kiphoc ki = new Kiphoc();
                        ki.setId(rs.getInt("idkip"));
                        ki.setTen(rs.getString("tenkip"));
                        lh.setKiphoc(ki);
                        kq.add(lh);
                    }
                }catch(Exception e){
                    e.printStackTrace();
                    kq = null;
                }
                return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `LichLHP`(IN idLHP int)
BEGIN
       SELECT a.*, b.ten as tengv, b.hodem as hodemgv, c.ten as tenphong,
    d.ten as tentuan, e.ten as tenngay, f.ten as tenkip
    FROM tbllichhoc a, tblthanhvien b, tblphonghoc c,
    tbltuan d, tblngay e, tblkip f
    WHERE a.idlophocphan = idLHP AND b.id = a.idgiangvien
    AND c.id = a.idphonghoc AND d.id = a.idtuan
    AND e.id = a.idngay AND f.id = a.idkip;
END
```

Lớp DangkihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Dangkihoc;
import model.Daudiem;
import model.Hoten;
import model.Ketqua;
import model.Khoa;
import model.Kihoc;
import model.Lophocphan;
import model.Monhoc;
import model.MonhocDaudiem;
import model.MonhocKihoc;
import model.Sinhvien;
```

```
import model.SinhvienKhoa;

public class DangkihocDAO extends DAO{
	public DangkihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * lay danh sach cac lop hoc phan ma @idSVK da dang ki trong hoc ki @idKihoc
	 * @param idSVK: id sinh vien khoa
	 * @param idKihoc: id ki hoc
	 * @return dang sach dang ki hoc
	 */
	public ArrayList<Dangkihoc> getDKcuaSV(int idSVK, int idKihoc){
		ArrayList<Dangkihoc> kq = null;
		String sql = "{call DKcuaSVtheoKi(?,?)}";// su dung stored procedure
		LichhocDAO ld = new LichhocDAO();
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1, idSVK);
			cs.setInt(2, idKihoc);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Dangkihoc>();
				Dangkihoc dk = new Dangkihoc();
				dk.setId(rs.getInt("id"));
				//lop hoc phan
				Lophocphan lhp = new Lophocphan();
				lhp.setId(rs.getInt("idlhp"));
				lhp.setTen(rs.getString("tenlhp"));
				lhp.setDsLichhoc(ld.getLHcuaLHP(lhp.getId()));//get lich
				//monhockihoc
				MonhocKihoc mhkh = new MonhocKihoc();
				mhkh.setId(rs.getInt("idmhkh"));
				//monhoc
				Monhoc mh = new Monhoc();
				mh.setId(rs.getInt("idmh"));
				mh.setTen(rs.getString("tenmh"));
				mh.setSoTC(rs.getInt("sotc"));
				mhkh.setMonhoc(mh);
				//kihoc
				Kihoc kh = new Kihoc();
				kh.setId(idKihoc);
				mhkh.setKihoc(kh);
				lhp.setMonhocKihoc(mhkh);
```

```
                        dk.setLophocphan(lhp);
                        //sinhvienkhoa
                        SinhvienKhoa svk = new SinhvienKhoa();
                        svk.setId(idSVK);
                        //sinh vien
                        Sinhvien sv = new Sinhvien();
                        sv.setId(rs.getInt("idsinhvien"));
                        svk.setSinhvien(sv);
                        //khoa
                        Khoa k = new Khoa();
                        k.setId(rs.getInt("idkhoa"));
                        svk.setKhoa(k);
                        dk.setSinhvienKhoa(svk);
                        kq.add(dk);
                }
        }catch(Exception e){
                e.printStackTrace();
                kq = null;
        }
        return kq;
}
```

1

```
public boolean luuDKcuaSV(ArrayList<Dangkihoc> listDK){
        if((listDK == null)|| (listDK.size() ==0)) return false;
        boolean kq = false;
        String sqlXoa = "DELETE FROM tbldangkihoc WHERE idsinhvienkhoa=?
                    AND idlophocphan IN "
                      + "(SELECT a.id FROM tbllophocphan a, tblmonhockihoc b
             WHERE b.idkihoc = ? AND a.idmonhockihoc=b.id)";
        String sqlThem = "INSERT INTO tbldangkihoc(idsinhvienkhoa,
           idlophocphan) VALUES(?,?)";
        try{
                this.con.setAutoCommit(false);
                //xoa het dang ki cu
                int idSVK = listDK.get(0).getSinhvienKhoa().getId();
                int idKihoc = listDK.get(0).getLophocphan().
                             getMonhocKihoc().getKihoc().getId();
                PreparedStatement psXoa = con.prepareStatement(sqlXoa);
                psXoa.setInt(1, idSVK);
                psXoa.setInt(2, idKihoc);
                psXoa.executeUpdate();
```

2

```
                //them lai dang ki nhu dang ki moi
                for(Dangkihoc dk : listDK){
                        PreparedStatement psThem = con.prepareStatement(sqlThem);
                        psThem.setInt(1, dk.getSinhvienKhoa().getId());
```

3

1

```
                        psThem.setInt(2, dk.getLophocphan().getId());
                        psThem.executeUpdate();
                    }
                    this.con.commit();//cmt dong nay ney chay che do JUnit test
                    kq=true;
                }catch(Exception e){
                    try{
                        this.con.rollback();//cmt dong nay ney chay che do JUnit
test
                    }catch(Exception ee){
                        kq=false;
                        ee.printStackTrace();
                    }
                    e.printStackTrace();
                }finally{
                    try{
                        this.con.setAutoCommit(true);//cmt dong nay neu chay che
do JUnit test
                    }catch(Exception e){
                        kq=false;
                        e.printStackTrace();
                    }
                }
                return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DKcuaSVtheoKi`(IN idSVK int, IN idKihoc
int)
BEGIN
        SELECT a.*, c.id as idmhkh, d.id as idmh, d.ten as tenmh, d.sotc,
    b.id as idlhp, b.ten as tenlhp, e.idkhoa,
    e.nienkhoa, e.idsinhvien, f.masv, g.hodem, g.ten as tensv
    FROM tbldangkihoc a, tbllophocphan b, tblmonhockihoc c,
    tblmonhoc d, tblsinhvienkhoa e, tblsinhvien f, tblthanhvien g,
    tblkhoa h
    WHERE a.idsinhvienkhoa = idSVK AND b.id = a.idlophocphan
    AND c.id = b.idmonhockihoc AND c.idkihoc = idKihoc
    AND d.id = c.idmonhoc AND e.id=a.idsinhvienkhoa
    AND f.idthanhvien = e.idsinhvien AND g.id=e.idsinhvien
    AND h.id = e.idkhoa;
END
```

Trang gddangnhap.jsp 2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.ArrayList,dao.*,model.*"%>
```

3

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Dang nhap</title>
</head>
<body>
 <%
      if(request.getParameter("err") !=null &&
request.getParameter("err").equalsIgnoreCase("timeout")){
        %> <h4>Hết phiên làm việc. Làm ơn đăng nhập lại!</h4><%
      }else if(request.getParameter("err") !=null &&
request.getParameter("err").equalsIgnoreCase("fail")){
        %> <h4 color="red">Sai tên đăng nhập/mật khẩu!</h4><%
      }
      %>
<h2> Đăng nhập </h2>
       <form name="dangnhap" action="doDangnhap.jsp" method="post">
             <table border="0">
                    <tr>
                          <td>Tên đăng nhập:</td>
                          <td><input type="text" name="username" id="username"
required /></td>
                    </tr>
                    <tr>
                          <td>Mật khẩu:</td>
                          <td><input type="password" name="password" id="password"
                                 required /></td>
                    </tr>
                    <tr>
                          <td></td>
                          <td><input type="submit" value="Đăng nhập" /></td>
                    </tr>
             </table>
       </form>
</body>
</html>
```

Trang doDangnhap.jsp

```
<%@page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8" import="java.util.ArrayList,dao.*,model.*"%>

<%
       String username = (String)request.getParameter("username");
       String password = (String)request.getParameter("password");
       Thanhvien tv = new Thanhvien();
```

```
    tv.setUsername(username);
    tv.setPassword(password);

    ThanhvienDAO dao = new ThanhvienDAO();
    boolean kq = dao.kiemtraDangnhap(tv);

    if(kq && (tv.getVaitro().equalsIgnoreCase("sinhvien"))){
          session.setAttribute("sinhvien", tv);
          response.sendRedirect("sv\\gdchinhsv.jsp");
    }else if(kq &&(tv.getVaitro().equalsIgnoreCase("giangvien"))){
          session.setAttribute("giangvien", tv);
          response.sendRedirect("gv\\gdchinhgv.jsp");
    }else{
          response.sendRedirect("gddangnhap.jsp?err=fail");
    }
%>
```

1

Trang gdChinhSV.jsp 2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<%@include file ="../header.jsp" %>
<title>Trang chu sinh vien</title>
</head>
<body>
<%
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
      if(sv==null){
         response.sendRedirect("dangnhap.jsp?err=timeout");
      }
      %>
<h2> Trang chủ sinh viên </h2>
<button onclick="openPage('gdchonnganh.jsp')">Đăng kí học</button>
</body>
</html>
```

3

Trang gdChonnganh.jsp 4

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
```

5

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Chon nganh hoc</title>
</head>
<%
//lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
       response.sendRedirect("dangnhap.jsp?err=timeout");
}
//lay danh sach nganh hoc cua sinh vien
ArrayList<SinhvienKhoa> listNganh = (new
SinhvienKhoaDAO()).getNganhcuaSV(sv.getId());
session.setAttribute("listSVK", listNganh);
//lay danh sach ki hoc dang mo dang ki
ArrayList<Kihoc> listKi = (new KihocDAO()).getKihocdangki();
%>
<body>
<h2> Chọn ngành </h2>
       <form name="chonnganh" action="gddangki.jsp" method="post">
              <table border="0">
                     <tr>
                            <td>Chọn ngành:</td>
                            <td><select name="nganh" size=1 >
                                   <option value="-1" selected>--chọn ngành--</option>
                                   <% for(SinhvienKhoa svk: listNganh){%>
                                          <option value="<%=svk.getId()%>"><
%=svk.getKhoa().getTen() %></option>
                                   <%}
                                   %>
                            </select>
                            </td>
                     </tr>
                     <tr>
                            <td>Chọn kì học:</td>
                            <td><select name="kihoc" size=1 >
                                   <option value="-1" selected>--chọn kì học--
</option>
                                   <% for(Kihoc k: listKi){
                                          String ten = k.getHocki().getTen()+" -
"+k.getNamhoc().getTen();%>
                                          <option value="<%=k.getId()%>"><%=ten
%></option>
                                   <%}
                                   %>
                            </select>
                            </td>
```

```
                    </tr>
                    <tr>
                          <td></td>
                          <td><input type="submit" value="Vào đăng kí" /></td>
                    </tr>
              </table>
        </form>
</body>
</html>
```

1

Trang gdDangki.jsp 2

3

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"  import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<%@include file ="../header.jsp" %>
<title>Dang ki hoc</title>
</head>
<%
       //lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
       response.sendRedirect("dangnhap.jsp?err=timeout");
}

ArrayList<Dangkihoc> listDK = null;
SinhvienKhoa svk = null;
//kiem tra cach trang nay bi goi
String action = request.getParameter("action");
if ((action == null)||(action.trim().length() ==0)) {//goi tu trang chon nganh
       //lay du lieu nganh hoc va ki hoc
       int idSVK = Integer.parseInt(request.getParameter("nganh"));
       int idKihoc = Integer.parseInt(request.getParameter("kihoc"));
       session.setAttribute("idSVK", idSVK);
       session.setAttribute("idKihoc", idKihoc);
       //lay thong tin sinh vien + nganh hoc
       ArrayList<SinhvienKhoa> listSVK =
(ArrayList<SinhvienKhoa>)session.getAttribute("listSVK");
       for(SinhvienKhoa tmp:listSVK)
              if(tmp.getId() == idSVK){
       svk = tmp;
              }
       //lay danh sach dang ki cua sinh vien, neu co
       listDK = (new DangkihocDAO()).getDKcuaSV(idSVK, idKihoc);
```

```
} else if (action.equalsIgnoreCase("them")) {// goi tu trang chon lop hoc phan
	listDK = (ArrayList<Dangkihoc>)session.getAttribute("dangkihoc");
	if(listDK == null) listDK = new ArrayList<Dangkihoc>();
	svk = (SinhvienKhoa)session.getAttribute("sinhvienkhoa");
	ArrayList<Lophocphan> listLHP =
(ArrayList<Lophocphan>)session.getAttribute("listLHP");
	int idLHP = Integer.parseInt(request.getParameter("idlhp"));
	boolean daTontaiLHP = false;
	for(Dangkihoc dk:listDK){
		if(dk.getLophocphan().getId() == idLHP){
	daTontaiLHP = true;
	break;
		}
	}

	if(!daTontaiLHP){
			//tao dang ki moi
			Dangkihoc dkMoi = null;
			for (Lophocphan lhp : listLHP)
				if (lhp.getId() == idLHP) {
					dkMoi = new Dangkihoc();
					dkMoi.setLophocphan(lhp);
					dkMoi.setSinhvienKhoa(svk);
				}
			//kiem tra co phai sua dang ki cu khong
			boolean daTontaiMHKH = false;
			int index = 0;
			for (Dangkihoc dk : listDK) {
				if (dk.getLophocphan().getMonhocKihoc().getId() == dkMoi
						.getLophocphan().getMonhocKihoc().getId()) {
					listDK.remove(dk);
					break;
				}
				index++;
			}
			//them dang ki moi vao danh sach
			listDK.add(index, dkMoi);
		}
	} else if (action.equalsIgnoreCase("xoa")) {// goi tu trang dang ki, de xoa
		listDK = (ArrayList<Dangkihoc>)session.getAttribute("dangkihoc");
		if (listDK == null)
			listDK = new ArrayList<Dangkihoc>();
		svk = (SinhvienKhoa)session.getAttribute("sinhvienkhoa");
		int idLHP = Integer.parseInt(request.getParameter("idlhp"));
		for (Dangkihoc dk : listDK)
			if (dk.getLophocphan().getId() == idLHP) {
				listDK.remove(dk);
```

```
                        break;
                    }
        }
session.setAttribute("sinhvienkhoa", svk);
session.setAttribute("dangkihoc", listDK);
%>
<body>
<h2> Đăng kí học </h2>
<table border="0">
<tr> <td>
<%if(svk != null){ %>
<p/>Họ tên: <%=svk.getSinhvien().getHoten().getHodem() %> <
%=svk.getSinhvien().getHoten().getTen() %>
<p/>Mã: <%=svk.getSinhvien().getMasv() %>
<p/>Ngành: <%=svk.getKhoa().getTen() %>
<%} %>
</td>
</tr>
<tr><td>
<h3> Các môn đã đăng kí: </h3>
</td>
</tr>
<tr> <td> <table style="border: 1px solid black;border-collapse: collapse;" >
<thead><td style="border: 1px solid black; padding:0 15px 0 15px;">T.T</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Mã môn học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Tên môn học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Tên lớp học phần</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Xem lịch</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Sửa</td>
<td style="border: 1px solid black;padding:0 15px 0 15px;">Xóa</td>
</thead>
<% int soTC =0;
if(listDK != null)
for(int i=0; i<listDK.size(); i++){
        soTC += listDK.get(i).getLophocphan().getMonhocKihoc().getMonhoc().getSoTC();
%>
<tr>
<td style="text-align:center; padding:0 15px 0 15px;"><%=(i+1) %></td>
<td style="text-align:center; padding:0 15px 0 15px;"><
%=listDK.get(i).getLophocphan().getMonhocKihoc().getMonhoc().getId() %></td>
<td style="padding:0 15px 0 15px;"><
%=listDK.get(i).getLophocphan().getMonhocKihoc().getMonhoc().getTen() %></td>
<td style="padding:0 15px 0 15px;"><%=listDK.get(i).getLophocphan().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
<a href="gdxemlich.jsp?idlhp=<%=listDK.get(i).getLophocphan().getId()%>">Xem
lịch</a></td>
<td style="text-align:center; padding:0 15px 0 15px;">
```

```
<a href="gdchonlhp.jsp?idmhkh=<
%=listDK.get(i).getLophocphan().getMonhocKihoc().getId()%>">Sửa</a></td>
<td style="text-align:center; padding:0 15px 0 15px;">
<a href="gddangki.jsp?action=xoa&&idlhp=<%=listDK.get(i).getLophocphan().getId()%>"
onclick="return confirm('Bạn muốn xóa lớp học phần khỏi danh sách đăng
kí?');">Xóa</a></td>
</tr>
<%} %>
</table>
<p/>Tổng số tín chỉ: <%=soTC%>
</td>
</tr>
<tr>
<td>
<button onclick="openPage('gdchinhsv.jsp')">Về trang chính</button>
<button onclick="openPage('gdchonmh.jsp')">Chọn tiếp</button>
<button onclick="openPage('doluudk.jsp')">Lưu đăng kí</button>
</td>
</tr>
</table>
</body>
</html>
```

1

Trang gdChonMH.jsp 2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Chon mon hoc</title>
</head>
<%
//lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
      response.sendRedirect("dangnhap.jsp?err=timeout");
}

//lay du lieu nganh hoc va ki hoc
int idSVK = (int)session.getAttribute("idSVK");
int idKihoc = (int)session.getAttribute("idKihoc");

//lay danh sach mon hoc duoc dang ki
ArrayList<MonhocKihoc> listMHKH = (new MonhocKihocDAO()).getMHcuaSV(idSVK, idKihoc);
//loai bo cac mon hoc da dang ki
```

3

```
ArrayList<Dangkihoc> listDK =
(ArrayList<Dangkihoc>)session.getAttribute("dangkihoc");
if(listDK != null)
for(int i=0; i<listMHKH.size(); i++){
	boolean daDK = false;
	for(Dangkihoc dk:listDK)
		if(dk.getLophocphan().getMonhocKihoc().getId() ==
listMHKH.get(i).getId()){
			daDK = true;
			break;
		}
	if(daDK){
		listMHKH.remove(i);
		//if(i>0)
			i--;
	}
}
session.setAttribute("listMHKH", listMHKH);
%>
<body>
<h2> Chọn môn học </h2>
<table style="border: 1px solid black;border-collapse: collapse;">
<thead><td style="border: 1px solid black; padding:0 15px 0 15px;">T.T</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Mã môn học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Tên môn học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Số tín chỉ</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Chọn</td>
</thead>
<%
if(listMHKH != null)
for(int i=0; i<listMHKH.size(); i++){
%>
<tr>
<td style="text-align:center; padding:0 15px 0 15px;"><%=(i+1) %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listMHKH.get(i).getMonhoc().getId() %></td>
<td style="padding:0 15px 0 15px;"><%=listMHKH.get(i).getMonhoc().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listMHKH.get(i).getMonhoc().getSoTC() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
<a href="gdchonlhp.jsp?idmhkh=<%=listMHKH.get(i).getId()%>">Chọn</a></td>
</tr>
<%} %>
</table>
<br/>
<button type="button" name="back" onclick="history.back()">Quay lại</button>
</body>
```

```
</html>
```

Trang gdChonLHP.jsp 1

2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"  import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Chon lop hoc phan</title>
</head>
<%
//lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
       response.sendRedirect("dangnhap.jsp?err=timeout");
}

//lay du lieu nganh hoc va ki hoc
int idSVK = (int)session.getAttribute("idSVK");
int idMHKH = Integer.parseInt(request.getParameter("idmhkh"));
//lay danh sach mon hoc duoc dang ki
ArrayList<Lophocphan> listLHP = (new LophocphanDAO()).getLHPchoSVdangki(idSVK,
idMHKH);
//loai cac LHP trung lich voi cac lop da hoc
ArrayList<Dangkihoc> listDK =
(ArrayList<Dangkihoc>)session.getAttribute("dangkihoc");
if((listDK !=null) && (listLHP !=null)){
       ArrayList<Lichhoc> lichdachon = new ArrayList<Lichhoc>();
       LichhocDAO lhdao = new LichhocDAO();
       for(Dangkihoc dk:listDK){
              lichdachon.addAll(lhdao.getLHcuaLHP(dk.getLophocphan().getId()));
       }
       for(int i=0; i<listLHP.size(); i++){
              ArrayList<Lichhoc> listLH = lhdao.getLHcuaLHP(listLHP.get(i).getId());
              boolean ok = true;
              for(Lichhoc lhmoi:listLH){
                     for(Lichhoc lhdk:lichdachon)
                            if(lhmoi.getTuanhoc().equals(lhdk.getTuanhoc()) &&
                                          lhmoi.getNgayhoc().equals(lhdk.getNgayhoc())
&&
                                          lhmoi.getKiphoc().equals(lhdk.getKiphoc())){
                                   ok = false;
                                   break;
                            }
                     if(!ok) break;
```

```
                }
                if(!ok){
                        listLHP.remove(i);
                        i--;
                }
        }
}
session.setAttribute("listLHP", listLHP);
%>
<body>
<%if(listLHP != null){
%>
<h2> Chọn lớp học phần </h2>
<table style="border: 1px solid black;border-collapse: collapse;">
<thead><td style="border: 1px solid black; padding:0 15px 0 15px;">T.T</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Mã lớp học phần</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Tên lớp học phần</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Sĩ số tối đa</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Sĩ số thực</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Lịch học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Chọn</td>
</thead>
<%

for(int i=0; i<listLHP.size(); i++){
%>
<tr>
<td style="text-align:center; padding:0 15px 0 15px;"><%=(i+1) %></td>
<td style="text-align:center; padding:0 15px 0 15px;"><%=listLHP.get(i).getId()
%></td>
<td style="padding:0 15px 0 15px;"><%=listLHP.get(i).getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listLHP.get(i).getSisotoida() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listLHP.get(i).getSisothucte() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
<a href="gdxemlich.jsp?idlhp=<%=listLHP.get(i).getId()%>">Xem lịch</a></td>
<td style="text-align:center; padding:0 15px 0 15px;">
<a href="gddangki.jsp?action=them&&idlhp=<%=listLHP.get(i).getId()%>">Chọn</a></td>
</tr>
<%}
%>
</table>
<%
}else{%>
<h4> Không có lớp học phần nào trống lịch và còn chỗ</h4>
<%} %>
```

```
<br/>
<button type="button" name="back" onclick="history.back()">Quay lại</button>
</body>
</html>
```
1

Trang gdxemlich.jsp 2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Xem lich hoc</title>
</head>
<%
//lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
      response.sendRedirect("dangnhap.jsp?err=timeout");
}

int idLHP = Integer.parseInt(request.getParameter("idlhp"));
//lay danh sach mon hoc duoc dang ki
ArrayList<Lichhoc> listLH = (new LichhocDAO()).getLHcuaLHP(idLHP);
%>
<body>
<h2> Lịch học </h2>
<table style="border: 1px solid black;border-collapse: collapse;">
<thead><td style="border: 1px solid black; padding:0 15px 0 15px;">T.T</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Tuần</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Thứ</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Kip</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Phòng học</td>
<td style="border: 1px solid black; padding:0 15px 0 15px;">Giảng viên</td>
</thead>
<%
if(listLH != null)
for(int i=0; i<listLH.size(); i++){
%>
<tr>
<td style="text-align:center; padding:0 15px 0 15px;"><%=(i+1) %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listLH.get(i).getTuanhoc().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
%=listLH.get(i).getNgayhoc().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;">
```
3

```
%=listLH.get(i).getKiphoc().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;"><
%=listLH.get(i).getPhonghoc().getTen() %></td>
<td style="text-align:center; padding:0 15px 0 15px;"><
%=(listLH.get(i).getGiangvien().getHoten().getHodem()+" "+
             listLH.get(i).getGiangvien().getHoten().getTen()) %></td>
</tr>
<%} %>
</table>
<br/>
<button type="button" name="back" onclick="history.back()">Quay lại</button>
</body>
</html>
```

1

Trang doluudk.jsp 2

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.*,dao.*,model.*"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<%
//lay id sinh vien
Thanhvien sv = (Thanhvien)session.getAttribute("sinhvien");
if(sv==null){
response.sendRedirect("dangnhap.jsp?err=timeout");
}
//lay thong tin dang ki va luu
ArrayList<Dangkihoc> listDK =
(ArrayList<Dangkihoc>)session.getAttribute("dangkihoc");
if((new DangkihocDAO()).luuDKcuaSV(listDK)){
%>
      <script type="text/javascript">
      alert("Đăng kí thành công!");
      </script>
<%
      response.sendRedirect("gdchinhsv.jsp");
}else{
%>
      <script type="text/javascript">
      alert("Lỗi đăng kí!");
      history.back();
      </script>
<%
}
%>
```

3

4

5.2.2 Chức năng nhập điểm 1

Lớp KihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Hocki;
import model.Kihoc;
import model.Namhoc;

public class KihocDAO extends DAO{
	public KihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * Lay danh sach cac hoc ki dang hoc
	 * @return danh sach ki hoc dang hoc
	 */
	public ArrayList<Kihoc> getKidanghoc(){
		ArrayList<Kihoc> kq=null;
		String sql = "{call kiDanghoc()}"; //su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Kihoc>();
				Kihoc kh = new Kihoc();
				kh.setId(rs.getInt("idkihoc"));
				//nam hoc
				Namhoc nh = new Namhoc();
				nh.setId(rs.getInt("idnamhoc"));
				nh.setTen(rs.getString("namhoc"));
				kh.setNamhoc(nh);
				//hoc ki
				Hocki hk = new Hocki();
				hk.setId(rs.getInt("idhocki"));
				hk.setTen(rs.getString("hocki"));
				kh.setHocki(hk);
				kq.add(kh);
			}
```

1

```
            }catch(Exception e){
                  e.printStackTrace();
                  kq = null;
            }
            return kq;
      }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `kiDanghoc`()
BEGIN
      SELECT a.id as idnamhoc, a.ten as namhoc, b.id as idhocki,
      b.ten as hocki, c.id as idkihoc
      FROM tblnamhoc a, tblhocki b, tblkihoc c
      WHERE c.idnamhoc = a.id AND c.idhocki = b.id AND c.danghoc = 1;
END
```

Lớp MonhocKihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Kihoc;
import model.Monhoc;
import model.MonhocKihoc;

public class MonhocKihocDAO extends DAO{
      public MonhocKihocDAO() {
            super();
            // TODO Auto-generated constructor stub
      }

      /**
       * lay danh sach cac mon hoc ma giang vien @idGV da dang ki day
       * @param idGV: id giang vien
       * @param idKihoc: id ki hoc
       * @return danh sach monhockihoc
       */
      public ArrayList<MonhocKihoc> getMHcuaGV(int idGV, int idKihoc){
            ArrayList<MonhocKihoc> kq = null;
            String sql = "{call MHmaGVday(?,?)}";// su dung stored procedure
            try{
                  CallableStatement cs = con.prepareCall(sql);
                  cs.setInt(1, idGV);
                  cs.setInt(2, idKihoc);
                  ResultSet rs = cs.executeQuery();

                  while(rs.next()){
```

1

```
                        if(kq == null) kq = new ArrayList<MonhocKihoc>();
                        MonhocKihoc mhkh = new MonhocKihoc();
                        mhkh.setId(rs.getInt("idmhkh"));
                        //mon hoc
                        Monhoc mh = new Monhoc();
                        mh.setId(rs.getInt("idmh"));
                        mh.setTen(rs.getString("tenmh"));
                        mh.setSoTC(rs.getInt("sotc"));
                        mhkh.setMonhoc(mh);
                        //ki hoc
                        Kihoc kh = new Kihoc();
                        kh.setId(idKihoc);
                        mhkh.setKihoc(kh);
                        kq.add(mhkh);
                }
            }catch(Exception e){
                e.printStackTrace();
                kq = null;
            }
            return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `MHmaGVday`(IN idGV int, IN idKihoc int)
BEGIN
       SELECT distinctrow a.id as idmh, a.ten as tenmh, a.sotc, b.id as idmhkh
    FROM tblmonhoc a, tblmonhockihoc b, tbllophocphan c, tbllichhoc d
    WHERE b.idKihoc = idKihoc AND a.id = b.idmonhoc
    AND c.idmonhockihoc = b.id AND d.idlophocphan = c.id
    AND d.idgiangvien = idGV;
END
```

Lớp LophocphanDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Kihoc;
import model.Lophocphan;
import model.Monhoc;
import model.MonhocKihoc;

public class LophocphanDAO extends DAO{
       public LophocphanDAO() {
              super();
              // TODO Auto-generated constructor stub
       }
```

1

```
	/**
	 * lay danh sach lop hoc phan ma giang vien @idGV day cua mon hoc @idMHKH
	 * @param idMHKH id cua monhoc kihoc
	 * @param idGV id cua giang vien
	 * @return dang sach lop hoc phan
	 */
	public ArrayList<Lophocphan> getLHPmaGVday(int idGV, int idMHKH){
		ArrayList<Lophocphan> kq = null;
		String sql = "{call LHPmaGVday(?,?)}";// su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1, idGV);
			cs.setInt(2, idMHKH);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Lophocphan>();
				Lophocphan lhp = new Lophocphan();
				lhp.setId(rs.getInt("id"));
				lhp.setTen(rs.getString("ten"));
				lhp.setSisotoida(rs.getInt("sisotoida"));
				//monhockihoc
				MonhocKihoc mhkh = new MonhocKihoc();
				mhkh.setId(idMHKH);
				lhp.setMonhocKihoc(mhkh);
				kq.add(lhp);
			}
		}catch(Exception e){
			e.printStackTrace();
			kq = null;
		}
		return kq;
	}
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `LHPmaGVday`(IN idGV int, IN idMHKH int)
BEGIN
	SELECT distinctrow a.*
    FROM tbllophocphan a, tbllichhoc b
    WHERE a.idmonhockihoc = idMHKH AND b.idlophocphan = a.id
    AND b.idgiangvien = idGV;
END
```

Lớp DangkihocDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Dangkihoc;
import model.Daudiem;
import model.Hoten;
import model.Ketqua;
import model.Khoa;
import model.Kihoc;
import model.Lophocphan;
import model.Monhoc;
import model.MonhocDaudiem;
import model.MonhocKihoc;
import model.Sinhvien;
import model.SinhvienKhoa;

public class DangkihocDAO extends DAO{
	public DangkihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * lay danh sach diem cua cac sinh vien da dang ki vao lop hoc phan @idLHP
	 * @param idLHP id cua lop hoc phan
	 * @return danh sach cac doi tuong dang ki hoc, trong co cac dau diem
	 */
	public ArrayList<Dangkihoc> getDiemcuaLHP(int idLHP){
		ArrayList<Dangkihoc> kq = null;
		String sqlDK = "{call DKcuaLHP(?)}";// su dung stored procedure
		String sqlDiem = "{call DiemcuaDK(?)}";// su dung stored procedure
		LichhocDAO ld = new LichhocDAO();
		try{
			CallableStatement cs = con.prepareCall(sqlDK);
			cs.setInt(1, idLHP);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<Dangkihoc>();
				Dangkihoc dk = new Dangkihoc();
				dk.setId(rs.getInt("id"));
				//lop hoc phan
				Lophocphan lhp = new Lophocphan();
				lhp.setId(idLHP);
				dk.setLophocphan(lhp);
				//sinhvien
				SinhvienKhoa sv = new SinhvienKhoa();
```

```
                    sv.setId(rs.getInt("idsinhvienkhoa"));
                    dk.setSinhvienKhoa(sv);
                    //diem
                    CallableStatement csDiem = con.prepareCall(sqlDiem);
                    cs.setInt(1, dk.getId());
                    ResultSet rsDiem = csDiem.executeQuery();
                    while(rsDiem.next()){
                          Ketqua diem = new Ketqua();
                          if(rsDiem.getObject("idkq")==null){//chua co diem
                                diem.setId(-1);
                                diem.setDiem(0);
                          }else{
                                diem.setId(rsDiem.getInt("idkq"));
                                diem.setDiem(rsDiem.getFloat("diem"));
                          }
                          //monhoc dau diem
                          MonhocDaudiem mhdd = new MonhocDaudiem();
                          mhdd.setId(rsDiem.getInt("id"));
                          mhdd.setTiLe(rsDiem.getFloat("tile"));
                          //Daudiem
                          Daudiem dd = new Daudiem();
                          dd.setId(rsDiem.getInt("iddd"));
                          dd.setTen(rsDiem.getString("tendd"));
                          mhdd.setDauDiem(dd);
                          diem.setDiemTP(mhdd);
                          dk.getDsDiemTP().add(diem);
                    }
                    kq.add(dk);
              }
        }catch(Exception e){
              e.printStackTrace();
              kq = null;
        }
        return kq;
}

/**
 * luu hoac cap nhat diem cua cac sinh vien trong lop hoc phan
 * @param listDiem danh sach diem cua sinh vien trong lop hoc phan
 * @return thanh cong hay co loi
 */
public boolean luuDiemcuaLHP(ArrayList<Dangkihoc> listDiem){
        boolean kq = false;
        String sqlKiemtra = "SELECT id FROM tblKetqua WHERE iddangkihoc = ?
           AND idmonhocdaudiem =?";
        String sqlThem = "INSERT INTO tblKetqua(diem, ghichu, iddangkihoc,
            idmonhocdaudiem) VALUES(?,?,?,?)";
```

```
String sqlSua = "UPDATE tblKetqua SET diem=?, ghichu=? WHERE id=?";
try{
	this.con.setAutoCommit(false);
	for(Dangkihoc dk:listDiem){
		for(Ketqua diem: dk.getDsDiemTP()){
			PreparedStatement ps =
con.prepareStatement(sqlKiemtra);
			ps.setInt(1, dk.getId());
			ps.setInt(2, diem.getDiemTP().getId());
			ResultSet rs = ps.executeQuery();
			if(rs.next()){// da ton tai, nen chi cap nhat
				PreparedStatement psSua =
				    con.prepareStatement(sqlSua);
				psSua.setFloat(1, diem.getDiem());
				psSua.setString(2, diem.getGhichu());
				psSua.setInt(3, rs.getInt("id"));
				psSua.executeUpdate();
			}else{// chua ton tai nen them moi
				PreparedStatement psThem =
				   con.prepareStatement(sqlThem);
				psThem.setFloat(1, diem.getDiem());
				psThem.setString(2, diem.getGhichu());
				psThem.setInt(3, dk.getId());
				psThem.setInt(4, diem.getDiemTP().getId());
				psThem.executeUpdate();
			}
		}
	}
	this.con.commit();
	kq=true;
}catch(Exception e){
	try{
		this.con.rollback();
	}catch(Exception ex){
		kq=false;
		ex.printStackTrace();
	}
	kq=false;
	e.printStackTrace();
}finally{
	try{
		this.con.setAutoCommit(true);
	}catch(Exception ex){
		kq=false;
		ex.printStackTrace();
	}
}
```

1

```
            return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DKcuaLHP`(IN idLHP int)
BEGIN
        SELECT * FROM tbldangkihoc
    WHERE idlophocphan = idLHP;
END
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DiemcuaDK`(IN idDK int)
BEGIN
        SELECT a.id, a.tile, b.id as iddd, b.ten,
    (SELECT c.id FROM tblKetqua c
    WHERE c.iddangkihoc = idDK AND c.idmonhocdaudiem = a.id) as idkq,
    (SELECT c.diem FROM tblKetqua c
    WHERE c.iddangkihoc = idDK AND c.idmonhocdaudiem = a.id) as diem
    FROM tblmonhocdaudiem a, tbldaudiem b,
    tblmonhockihoc e, tbllophocphan f, tbldangkihoc g
    WHERE g.id = idDK AND f.id = g.idlophocphan
    AND e.id = f.idmonhockihoc AND a.idmonhoc=e.idmonhoc
    AND  b.id = a.iddaudiem AND a.tile > 0;
END
```

5.2.3 Chức năng xem thống kê 2

Lớp KihocDAO.java 3

4

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Hocki;
import model.Kihoc;
import model.Namhoc;

public class KihocDAO extends DAO{
        public KihocDAO() {
                super();
                // TODO Auto-generated constructor stub
        }

        /**
         * lay tat ca cac ki da hoc va dang hoc de chon xem thong ke
         * @return danh sach cacs doi tuong kihoc
         */
        public ArrayList<Kihoc> getTatcaKihoc(){
                ArrayList<Kihoc> kq=null;
                String sql = "{call tatcaKihoc()}"; //su dung stored procedure
```

1

```
            try{
                CallableStatement cs = con.prepareCall(sql);
                ResultSet rs = cs.executeQuery();

                while(rs.next()){
                    if(kq == null) kq = new ArrayList<Kihoc>();
                    Kihoc kh = new Kihoc();
                    kh.setId(rs.getInt("idkihoc"));
                    //nam hoc
                    Namhoc nh = new Namhoc();
                    nh.setId(rs.getInt("idnamhoc"));
                    nh.setTen(rs.getString("namhoc"));
                    kh.setNamhoc(nh);
                    //hoc ki
                    Hocki hk = new Hocki();
                    hk.setId(rs.getInt("idhocki"));
                    hk.setTen(rs.getString("hocki"));
                    kh.setHocki(hk);
                    kq.add(kh);
                }
            }catch(Exception e){
                e.printStackTrace();
                kq = null;
            }
            return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `tatcaKihoc`()
BEGIN
    SELECT a.id as idnamhoc, a.ten as namhoc, b.id as idhocki,
    b.ten as hocki, c.id as idkihoc
    FROM tblnamhoc a, tblhocki b, tblkihoc c
    WHERE c.idnamhoc = a.id AND c.idhocki = b.id
  ORDER BY idnamhoc, idhocki;
END
```

Lớp TKHoclucDAO.java 2

3

```
package dao;
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.TKLoaihocluc;

public class TKHoclucDAO extends DAO{
    public TKHoclucDAO() {
        super();
```

1

```
            // TODO Auto-generated constructor stub
        }

        /**
         * thong ke moi loai hoc luc trong hoc ki @idKihoc
         * @param idKihoc id cua ki hoc duoc chon
         * @return danh sach doi tuong TKLoaihocluc
         */
        public ArrayList<TKLoaihocluc> getTKHocluc(int idKihoc){
            ArrayList<TKLoaihocluc> kq = null;
            String sql = "{call TKHocluc(?)}"; //su dung stored procedure
            try{
                CallableStatement cs = con.prepareCall(sql);
                cs.setInt(1,idKihoc);
                ResultSet rs = cs.executeQuery();

                while(rs.next()){
                    if(kq == null) kq = new ArrayList<TKLoaihocluc>();
                    TKLoaihocluc tkhl = new TKLoaihocluc();
                    tkhl.setId(rs.getInt("id"));
                    tkhl.setTen(rs.getString("ten"));
                    tkhl.setSoSV(rs.getInt("soluong"));
                    tkhl.setDiemTB(rs.getFloat("tb"));
                    kq.add(tkhl);
                }
            }catch(Exception e){
                e.printStackTrace();
                kq = null;
            }
            return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `TKHocluc`(IN idKihoc int)
BEGIN
    -- bang tam tinh diem tung mon hoc cua sinh vien
    DROP TEMPORARY TABLE IF EXISTS diemsvmh;
      CREATE TEMPORARY TABLE diemsvmh AS
    SELECT a.idsinhvienkhoa, e.id as idmonhoc, e.sotc,
    SUM(f.tile*b.diem) as tbmon
    FROM tbldangkihoc a, tblketqua b, tbllophocphan c,
    tblmonhockihoc d, tblmonhoc e, tblmonhocdaudiem f
    WHERE d.idkihoc = idKihoc AND c.idmonhockihoc = e.id
    AND a.idlophocphan = c.id AND b.iddangkihoc = a.id
    AND e.id = d.idmonhoc AND f.idmonhoc =e.id
    AND b.idmonhocdaudiem = f.id
    GROUP BY a.idsinhvienkhoa, idmonhoc, e.sotc;
```

```
-- chuyen diem tu he 10 sang he 4                    1
DROP TEMPORARY TABLE IF EXISTS diemsvmhhe4;
   CREATE TEMPORARY TABLE diemsvmhhe4 AS
SELECT *,
(SELECT diem FROM tblDiemhechu
WHERE tbmon >= diemhe10toithieu
AND tbmon < diemhe10toida) as tbmonhe4
FROM diemsvmh;

-- bang tam tinh diem trung binh ca hoc ki cua sinh vien 2
DROP TEMPORARY TABLE IF EXISTS diemsvhk;
   CREATE TEMPORARY TABLE diemsvhk AS                3
SELECT idsinhvienkhoa, SUM(sotc) as tongtc,
SUM(sotc*tbmonhe4)/SUM(sotc) as tbki
FROM diemsvmhhe4
GROUP BY idsinhvienkhoa;

-- tinh so luong                                     4
DROP TEMPORARY TABLE IF EXISTS tksoluong;
   CREATE TEMPORARY TABLE tksoluong AS
SELECT a.id, (SELECT COUNT(idsinhvienkhoa) FROM diemsvhk
WHERE tbki>=a.diemtoithieu AND tbki < a.diemtoida) as soluong
FROM tbllloaihocluc a;

-- tinh diem trung binh                              5
DROP TEMPORARY TABLE IF EXISTS tkdiemtb;
   CREATE TEMPORARY TABLE tkdiemtb AS                6
SELECT a.id, (SELECT AVG(tbki) FROM diemsvhk
WHERE tbki>=a.diemtoithieu AND tbki < a.diemtoida) as tb
FROM tbllloaihocluc a;

-- tlay ket qua day du                               7
   SELECT a.*, b.soluong, c.tb
FROM tbllloaihocluc a, tksoluong b, tkdiemtb c
WHERE b.id=a.id AND c.id=a.id;

END
```

Lớp TKSinhvienDAO.java 8

```
package dao;                                         9
import java.sql.CallableStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Hoten;
import model.Khoa;
import model.Sinhvien;
import model.TKSinhvien;
```

```
public class TKSinhvienDAO extends DAO{
	public TKSinhvienDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * thong ke ket qua cua cac sinh vien cos hoc luc @idLoaiHL, trong ki hoc
	 * @param idLoaiHL id loai hoc luc
	 * @param idKihoc id ki hoc
	 * @return danh sach doi tuong TKSinhvien
	 */
	public ArrayList<TKSinhvien> getTKSVcuaHL(int idLoaiHL, int idKihoc){
		ArrayList<TKSinhvien> kq = null;
		String sql = "{call TKSVtheoHL(?,?)}"; //su dung stored procedure
		try{
			CallableStatement cs = con.prepareCall(sql);
			cs.setInt(1,idLoaiHL);
			cs.setInt(2,idKihoc);
			ResultSet rs = cs.executeQuery();

			while(rs.next()){
				if(kq == null) kq = new ArrayList<TKSinhvien>();
				TKSinhvien tksv = new TKSinhvien();
				tksv.setId(rs.getInt("idsinhvienkhoa"));
				tksv.setDiemTB(rs.getFloat("tbki"));
				tksv.setSoTC(rs.getInt("tongtc"));
				//khoa-nganh
				Khoa k = new Khoa();
				k.setTen(rs.getString("tennganh"));
				tksv.setKhoa(k);
				//sinhvien
				Sinhvien sv = new Sinhvien();
				sv.setId(rs.getInt("id"));
				sv.setMasv(rs.getString("masv"));
				//hoten
				Hoten ht = new Hoten();
				ht.setHodem(rs.getString("hodem"));
				ht.setTen(rs.getString("ten"));
				sv.setHoten(ht);
				tksv.setSinhvien(sv);
				kq.add(tksv);
			}
		}catch(Exception e){
			e.printStackTrace();
			kq = null;
```

```
            }
            return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `TKSVtheoHL`(IN idLoaiHL int, IN idKihoc
int)
BEGIN
       DECLARE minHL float default 0;
    DECLARE maxHL float default 0;
    SELECT diemtoithieu, diemtoida INTO minHL, maxHL
    FROM tbllloaihocluc WHERE id = idLoaiHL;

    -- bang tam tinh diem tung mon hoc cua sinh vien
    DROP TEMPORARY TABLE IF EXISTS diemsvmh;
       CREATE TEMPORARY TABLE diemsvmh AS
    SELECT a.idsinhvienkhoa, e.id as idmonhoc, e.sotc,
    SUM(f.tile*b.diem) as tbmon
    FROM tbldangkihoc a, tblketqua b, tbllophocphan c,
    tblmonhockihoc d, tblmonhoc e, tblmonhocdaudiem f
    WHERE d.idkihoc = idKihoc AND c.idmonhockihoc = e.id
    AND a.idlophocphan = c.id AND b.iddangkihoc = a.id
    AND e.id = d.idmonhoc AND f.idmonhoc =e.id
    AND b.idmonhocdaudiem = f.id
    GROUP BY a.idsinhvienkhoa, idmonhoc, e.sotc;

    -- chuyen diem tu he 10 sang he 4
    DROP TEMPORARY TABLE IF EXISTS diemsvmhhe4;
       CREATE TEMPORARY TABLE diemsvmhhe4 AS
    SELECT *,
    (SELECT diem FROM tblDiemhechu
    WHERE tbmon >= diemhe10toithieu
    AND tbmon < diemhe10toida) as tbmonhe4
    FROM diemsvmh;

    -- bang tam tinh diem trung binh ca hoc ki cua sinh vien
    DROP TEMPORARY TABLE IF EXISTS diemsvhk;
       CREATE TEMPORARY TABLE diemsvhk AS
    SELECT idsinhvienkhoa, SUM(sotc) as tongtc,
    SUM(sotc*tbmonhe4)/SUM(sotc) as tbki
    FROM diemsvmhhe4
    GROUP BY idsinhvienkhoa
    HAVING ((tbki >= minHL) AND (tbki < maxHL));

    -- lay ket qua day du
    SELECT a.id, a.hodem, a.ten, b.masv, d.ten as tennganh,
    e.idsinhvienkhoa, e.tongtc, e.tbki
    FROM tblthanhvien a, tblsinhvien b, tblsinhvienkhoa c,
```

```
    tblKhoa d, diemsvhk e
    WHERE c.id = e.idsinhvienkhoa AND a.id = c.idsinhvien
    AND b.idthanhvien = a.id AND d.id= c.idkhoa;
END
```

1

Lớp DangkihocDAO.java 2

```
package dao;
import java.sql.CallableStatement;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import model.Dangkihoc;
import model.Daudiem;
import model.Hoten;
import model.Ketqua;
import model.Khoa;
import model.Kihoc;
import model.Lophocphan;
import model.Monhoc;
import model.MonhocDaudiem;
import model.MonhocKihoc;
import model.Sinhvien;
import model.SinhvienKhoa;

public class DangkihocDAO extends DAO{
	public DangkihocDAO() {
		super();
		// TODO Auto-generated constructor stub
	}

	/**
	 * lay diem trung binh cua tat ca cac mon cua sinh vien @idSVK trong hoc ki
	 * @param idSVK ma sinh vien khoa
	 * @param idKihoc ma ki hoc
	 * @return danh sach doi tuong Dangkihoc
	 */
	public ArrayList<Dangkihoc> getDiemSVtheoHK(int idSVK, int idKihoc){
		ArrayList<Dangkihoc> kq = null;
		String sqlDK = "{call DKcuaSVtheoKi(?,?)}";// su dung stored procedure
		String sqlDiem = "{call DiemcuaDK(?)}";// su dung stored procedure
		LichhocDAO ld = new LichhocDAO();
		try{
			CallableStatement cs = con.prepareCall(sqlDK);
			cs.setInt(1, idSVK);
			cs.setInt(2, idKihoc);
			ResultSet rs = cs.executeQuery();
```

3

```
while(rs.next()){                                                    1
    if(kq == null) kq = new ArrayList<Dangkihoc>();
    Dangkihoc dk = new Dangkihoc();
    dk.setId(rs.getInt("id"));
    //lop hoc phan
    Lophocphan lhp = new Lophocphan();
    lhp.setId(rs.getInt("idlhp"));
    dk.setLophocphan(lhp);
    //sinhvienkhoa
    SinhvienKhoa svk = new SinhvienKhoa();
    svk.setId(rs.getInt("idsinhvienkhoa"));
    svk.setNienkhoa(rs.getString("nienkhoa"));
    //sinhvien
    Sinhvien sv = new Sinhvien();
    sv.setId(rs.getInt("idsinhvien"));
    sv.setMasv(rs.getString("masv"));
    //Hoten
    Hoten hoten = new Hoten();
    hoten.setHodem(rs.getString("hodem"));
    hoten.setTen(rs.getString("tensv"));
    sv.setHoten(hoten);
    svk.setSinhvien(sv);
    dk.setSinhvienKhoa(svk);

    //diem 2
    CallableStatement csDiem = con.prepareCall(sqlDiem);             3
    cs.setInt(1, dk.getId());
    ResultSet rsDiem = csDiem.executeQuery();
    while(rsDiem.next()){
        Ketqua diem = new Ketqua();
        if(rsDiem.getObject("idkq")==null){//chua co diem
            diem.setId(-1);
            diem.setDiem(0);
        }else{
            diem.setId(rsDiem.getInt("idkq"));
            diem.setDiem(rsDiem.getFloat("diem"));
        }
        //monhoc dau diem
        MonhocDaudiem mhdd = new MonhocDaudiem();
        mhdd.setId(rsDiem.getInt("id"));
        mhdd.setTiLe(rsDiem.getFloat("tile"));
        //Daudiem
        Daudiem dd = new Daudiem();
        dd.setId(rsDiem.getInt("iddd"));
        dd.setTen(rsDiem.getString("tendd"));
        mhdd.setDauDiem(dd);
        diem.setDiemTP(mhdd);
```

```
                                dk.getDsDiemTP().add(diem);
                            }
                            kq.add(dk);
                        }
                }catch(Exception e){
                        e.printStackTrace();
                        kq = null;
                }
                return kq;
        }
}
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DKcuaSVtheoKi`(IN idSVK int, IN idKihoc
int)
BEGIN
       SELECT a.*, c.id as idmhkh, d.id as idmh, d.ten as tenmh, d.sotc,
    b.id as idlhp, b.ten as tenlhp, e.idkhoa,
    e.nienkhoa, e.idsinhvien, f.masv, g.hodem, g.ten as tensv
    FROM tbldangkihoc a, tbllophocphan b, tblmonhockihoc c,
    tblmonhoc d, tblsinhvienkhoa e, tblsinhvien f, tblthanhvien g,
    tblkhoa h
    WHERE a.idsinhvienkhoa = idSVK AND b.id = a.idlophocphan
    AND c.id = b.idmonhockihoc AND c.idkihoc = idKihoc
    AND d.id = c.idmonhoc AND e.id=a.idsinhvienkhoa
    AND f.idthanhvien = e.idsinhvien AND g.id=e.idsinhvien
    AND h.id = e.idkhoa;
END
```

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `DiemcuaDK`(IN idDK int)
BEGIN
       SELECT a.id, a.tile, b.id as iddd, b.ten,
    (SELECT c.id FROM tblKetqua c
    WHERE c.iddangkihoc = idDK AND c.idmonhocdaudiem = a.id) as idkq,
    (SELECT c.diem FROM tblKetqua c
    WHERE c.iddangkihoc = idDK AND c.idmonhocdaudiem = a.id) as diem
    FROM tblmonhocdaudiem a, tbldaudiem b,
    tblmonhockihoc e, tbllophocphan f, tbldangkihoc g
    WHERE g.id = idDK AND f.id = g.idlophocphan
    AND e.id = f.idmonhockihoc AND a.idmonhoc=e.idmonhoc
    AND  b.id = a.iddaudiem AND a.tile > 0;
END
```

5.3 KIỂM THỬ ĐƠN VỊ

Nội dung phần này sẽ trình bày kế hoạch kiểm thử đơn vị, các ca kiểm thử đơn vị cho lớp DangkihocDAO. Kế hoạch kiểm thử và các ca kiểm thử cho các lớp DAO còn lại coi như là bài tập cho người đọc.

5.3.Xây dựng bộ test case cho kiểm thử đơn vị 1

Bộ test case cho kiểm thử đơn vị được xây dựng theo bảng sau: 2

3

TT	Chức năng/ use case	Lớp điều khiển	Phương thức	Trường hợp test
1	Đăng kí học	DangkihocDAO	getDKcuaSV()	SV đã đăng kí 1 LHP
				SV đã đăng kí >1 LHP
				Kì học tồn tại, SV không tồn tại
				Kì học không tồn tại, SV tồn tại
				Kì học không tồn tại, SV không tồn tại.
				SV có đăng kí ở kì học khác, không có đăng kí ở kì học này
			luuDKcuaSV()	ĐK 1 LHP, chưa có ĐK cũ
				ĐK >1 LHP, chưa có ĐK cũ
				ĐK 1 LHP, cùng môn học với ĐK cũ, ĐK cũ cũng chỉ có 1 LHP
				ĐK 1 LHP, khác môn học với ĐK cũ, ĐK cũ cũng chỉ có 1 LHP
				ĐK >1 LHP, ĐK cũ có ít LHP hơn
				ĐK >1 LHP, ĐK cũ có nhiều LHP hơn
				ĐK mới trùng hoàn toàn với ĐK cũ
				ĐK mới không có LHP nào
		KihocDAO	...	...
		MonhocKihocDAO	...	...
		LophocphanDAO	...	...
		LichhocDAO	...	...
2	Nhập điểm	...	...	...
3	Thống kê học	...	...	...

	lực			
	...			

1

5.3.2 Cài đặt kiểm thử đơn vị 2

Việc thực hiện test đơn vị sẽ sử dụng Junit. Ngoại trừ các class interface, các class còn lại cần được thực hiện unit test. Mục tiêu của unit test là để đảm bảo từng unit (method, class) thực hiện đúng như trong thiết kế. 3

Quy trình tiến hành các bước như sau: 4

- Bước 1: Với mỗi method/class, xây dựng test cases với lưu ý lựa chọn các input sao cho đạt mức phủ cấp 2 (phủ hết các nhánh trong source code), expected output thì cần căn cứ vào thiết kế, đặc tả để xác định, không căn cứ vào source code. 5
- Bước 2: Viết script Junit
  - Với mỗi test cases, khai báo biến để lưu input/expected result
  - Gọi method cần test với test input ở trên.
  - Kiểm tra kết quả thực method với expected output
  - Kiểm tra DB xem đã thay đổi như mong muốn chưa?
- Bước 3: Báo cáo kết quả test, số lượng pass/false và mô tả chi tiết trường hợp false. 6

Lưu ý là cần thực hiện chuẩn bị môi trường (kết nối DB, dữ liệu mẫu) và cần Rollback sau khi thực hiện test để đảm bảo việc thực hiện các test cases khác, và sau này test lại sẽ không bi lỗi. . 7

Lớp TestDangkihocDAO.java 8

```
package unittest;
import java.util.ArrayList;
import junit.framework.Assert;
import model.Dangkihoc;
import model.Kihoc;
import model.Lophocphan;
import model.MonhocKihoc;
import model.SinhvienKhoa;
import org.junit.Test;
import dao.DAO;
import dao.DangkihocDAO;

public class TestDangkihocDAO {
	DangkihocDAO dkhdao = new DangkihocDAO();

	@Test
```

9

```
public void getDKcuaSV_testChuan1(){
        // sinh vien da dang ki dung 1 LHP
        int idSVK = 2;
        int idKihoc = 4;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNotNull(listDK);
        Assert.assertEquals(1, listDK.size());
        Assert.assertEquals(idSVK, listDK.get(0).getSinhvienKhoa().getId());
        Assert.assertEquals(idKihoc,
           listDK.get(0).getLophocphan().getMonhocKihoc().getKihoc().getId());
}

@Test
public void getDKcuaSV_testChuan2(){
        // sinh vien dang ki nhieu hon 1 LHP
        int idSVK = 1;
        int idKihoc = 4;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNotNull(listDK);
        Assert.assertEquals(3, listDK.size());
        for(int i=0; i<listDK.size(); i++){
                Assert.assertEquals(idSVK,
                        listDK.get(i).getSinhvienKhoa().getId());
                Assert.assertEquals(idKihoc,
          listDK.get(i).getLophocphan().getMonhocKihoc().getKihoc().getId());
        }
}

@Test
public void getDKcuaSV_testNgoaile1(){
        //ki hoc ton tai, SV khong ton tai
        int idSVK = 1000;
        int idKihoc = 4;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNull(listDK);
}

@Test
public void getDKcuaSV_testNgoaile2(){
        //ki hoc khong ton tai, SV ton tai
        int idSVK = 1;
        int idKihoc = 1000;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNull(listDK);
}

@Test
```

```java
public void getDKcuaSV_testNgoaile3(){
        //ki hoc khong ton tai, SV khong ton tai
        int idSVK = 1000;
        int idKihoc = 1000;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNull(listDK);
}
```

1

```java
@Test
public void getDKcuaSV_testNgoaile4(){
        //SV co dang ki o ki hoc khac, khong co dang ki o ki hoc nay
        int idSVK = 1;
        int idKihoc = 5;
        ArrayList<Dangkihoc> listDK = dkhdao.getDKcuaSV(idSVK, idKihoc);
        Assert.assertNull(listDK);
}
```

2

3

```java
@Test
public void luuDKcuaSV_testChuan1(){
        //dang ki 1LHP, chua co dang ki cu
        ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
        //tao dang ki hoc thu nhat
        SinhvienKhoa svk = new SinhvienKhoa();
        svk.setId(3);
        Kihoc kh = new Kihoc();
        kh.setId(4);
        MonhocKihoc mhkh = new MonhocKihoc();
        mhkh.setId(1);
        mhkh.setKihoc(kh);
        Lophocphan lhp = new Lophocphan();
        lhp.setId(3);
        lhp.setMonhocKihoc(mhkh);
        Dangkihoc dk = new Dangkihoc();
        dk.setSinhvienKhoa(svk);
        dk.setLophocphan(lhp);
        listDK.add(dk);

        try{
                DAO.con.setAutoCommit(false);
                boolean ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertTrue(ok);

                //lay ra kiem tra
                ArrayList<Dangkihoc> listDKnew =
                    dkhdao.getDKcuaSV(svk.getId(), kh.getId());
                Assert.assertNotNull(listDKnew);
                Assert.assertEquals(1, listDKnew.size());
```

```
                Assert.assertEquals(svk.getId(),
                  listDKnew.get(0).getSinhvienKhoa().getId());
                Assert.assertEquals(lhp.getId(),
                  listDKnew.get(0).getLophocphan().getId());
        }catch(Exception e){
                e.printStackTrace();
        }finally{
                try{
                        DAO.con.rollback();
                        DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                        ex.printStackTrace();
                }
        }
}
```

1

```
@Test
public void luuDKcuaSV_testChuan2(){
        //dang ki nhieu LHP, chua co dang ki cu
        ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
        //tao dang ki hoc thu nhat
        SinhvienKhoa svk = new SinhvienKhoa();
        svk.setId(3);
        Kihoc kh = new Kihoc();
        kh.setId(4);
        MonhocKihoc mhkh = new MonhocKihoc();
        mhkh.setId(1);
        mhkh.setKihoc(kh);
        Lophocphan lhp = new Lophocphan();
        lhp.setId(3);
        lhp.setMonhocKihoc(mhkh);
        Dangkihoc dk = new Dangkihoc();
        dk.setSinhvienKhoa(svk);
        dk.setLophocphan(lhp);
        listDK.add(dk);
```

2

```
        //tao dang ki hoc thu hai
        MonhocKihoc mhkh2 = new MonhocKihoc();
        mhkh2.setId(2);
        mhkh2.setKihoc(kh);
        Lophocphan lhp2 = new Lophocphan();
        lhp2.setId(9);
        lhp2.setMonhocKihoc(mhkh2);
        Dangkihoc dk2 = new Dangkihoc();
        dk2.setSinhvienKhoa(svk);
        dk2.setLophocphan(lhp2);
        listDK.add(dk2);
```

3

```
try{                                                    1
        DAO.con.setAutoCommit(false);
        boolean ok = dkhdao.luuDKcuaSV(listDK);
        Assert.assertTrue(ok);

        //lay ra kiem tra                               2
        ArrayList<Dangkihoc> listDKnew =
            dkhdao.getDKcuaSV(svk.getId(), kh.getId());
        Assert.assertNotNull(listDKnew);
        Assert.assertEquals(2, listDKnew.size());
        Assert.assertEquals(svk.getId(),
          listDKnew.get(0).getSinhvienKhoa().getId());
        Assert.assertEquals(lhp.getId(),
          listDKnew.get(0).getLophocphan().getId());
        Assert.assertEquals(svk.getId(),
          listDKnew.get(1).getSinhvienKhoa().getId());
        Assert.assertEquals(lhp2.getId(),
          listDKnew.get(1).getLophocphan().getId());
    }catch(Exception e){
        e.printStackTrace();
    }finally{
        try{
            DAO.con.rollback();
            DAO.con.setAutoCommit(true);
        }catch(Exception ex){
            ex.printStackTrace();
        }
    }
}

@Test                                                   3
public void luuDKcuaSV_testChuan3(){
    // dang ki 1LHP, cung MH voi dang ki cu, dang ki cu cung chi co 1LHP
    ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
    //tao dang ki hoc thu nhat
    SinhvienKhoa svk = new SinhvienKhoa();
    svk.setId(2);
    Kihoc kh = new Kihoc();
    kh.setId(4);
    MonhocKihoc mhkh = new MonhocKihoc();
    mhkh.setId(4);
    mhkh.setKihoc(kh);
    Lophocphan lhp = new Lophocphan();
    lhp.setId(27);
    lhp.setMonhocKihoc(mhkh);
    Dangkihoc dk = new Dangkihoc();
```

```
        dk.setSinhvienKhoa(svk);1
        dk.setLophocphan(lhp);
        listDK.add(dk);

        try{                                                    2
                DAO.con.setAutoCommit(false);
                boolean ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertTrue(ok);

                //lay ra kiem tra                               3
                ArrayList<Dangkihoc> listDKnew =
                        dkhdao.getDKcuaSV(svk.getId(), kh.getId());
                Assert.assertNotNull(listDKnew);
                Assert.assertEquals(1, listDKnew.size());
                Assert.assertEquals(svk.getId(),
                    listDKnew.get(0).getSinhvienKhoa().getId());
                Assert.assertEquals(lhp.getId(),
                    listDKnew.get(0).getLophocphan().getId());
        }catch(Exception e){
                e.printStackTrace();
        }finally{
                try{
                        DAO.con.rollback();
                        DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                        ex.printStackTrace();
                }
        }
}

@Test                                                           4
public void luuDKcuaSV_testChuan4(){
        // dang ki 1LHP, khac MH voi dang ki cu, dang ki cu cung chi co 1LHP
        ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
        //tao dang ki hoc thu nhat
        SinhvienKhoa svk = new SinhvienKhoa();
        svk.setId(2);
        Kihoc kh = new Kihoc();
        kh.setId(4);
        MonhocKihoc mhkh = new MonhocKihoc();
        mhkh.setId(2);
        mhkh.setKihoc(kh);
        Lophocphan lhp = new Lophocphan();
        lhp.setId(10);
        lhp.setMonhocKihoc(mhkh);
        Dangkihoc dk = new Dangkihoc();
        dk.setSinhvienKhoa(svk);
```

```
        dk.setLophocphan(lhp);1
        listDK.add(dk);

        try{2
                DAO.con.setAutoCommit(false);
                boolean ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertTrue(ok);

                //lay ra kiem tra                                        3
                ArrayList<Dangkihoc> listDKnew =
                      dkhdao.getDKcuaSV(svk.getId(), kh.getId());
                Assert.assertNotNull(listDKnew);
                Assert.assertEquals(1, listDKnew.size());
                Assert.assertEquals(svk.getId(),
                  listDKnew.get(0).getSinhvienKhoa().getId());
                Assert.assertEquals(lhp.getId(),
                  listDKnew.get(0).getLophocphan().getId());
        }catch(Exception e){
                e.printStackTrace();
        }finally{
                try{
                        DAO.con.rollback();
                        DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                        ex.printStackTrace();
                }
        }
}

@Test                                                                   4
public void luuDKcuaSV_testChuan5(){
        // dang ki > 1LHP, dang ki cu it LHP hon
        ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
        //tao dang ki hoc thu nhat
        SinhvienKhoa svk = new SinhvienKhoa();
        svk.setId(2);
        Kihoc kh = new Kihoc();
        kh.setId(4);
        MonhocKihoc mhkh = new MonhocKihoc();
        mhkh.setId(1);
        mhkh.setKihoc(kh);
        Lophocphan lhp = new Lophocphan();
        lhp.setId(3);
        lhp.setMonhocKihoc(mhkh);
        Dangkihoc dk = new Dangkihoc();
        dk.setSinhvienKhoa(svk);
        dk.setLophocphan(lhp);
```

```
        listDK.add(dk);1

        //tao dang ki hoc thu hai                    2
        MonhocKihoc mhkh2 = new MonhocKihoc();
        mhkh2.setId(2);
        mhkh2.setKihoc(kh);
        Lophocphan lhp2 = new Lophocphan();
        lhp2.setId(9);
        lhp2.setMonhocKihoc(mhkh2);
        Dangkihoc dk2 = new Dangkihoc();
        dk2.setSinhvienKhoa(svk);
        dk2.setLophocphan(lhp2);
        listDK.add(dk2);

        try{ 3
            DAO.con.setAutoCommit(false);
            boolean ok = dkhdao.luuDKcuaSV(listDK);
            Assert.assertTrue(ok);
                                                       4
            //lay ra kiem tra
            ArrayList<Dangkihoc> listDKnew =
              dkhdao.getDKcuaSV(svk.getId(), kh.getId());
            Assert.assertNotNull(listDKnew);
            Assert.assertEquals(2, listDKnew.size());
            Assert.assertEquals(svk.getId(),
              listDKnew.get(0).getSinhvienKhoa().getId());
            Assert.assertEquals(lhp.getId(),
              listDKnew.get(0).getLophocphan().getId());
            Assert.assertEquals(svk.getId(),
              listDKnew.get(1).getSinhvienKhoa().getId());
            Assert.assertEquals(lhp2.getId(),
              listDKnew.get(1).getLophocphan().getId());
        }catch(Exception e){
            e.printStackTrace();
        }finally{
            try{
                DAO.con.rollback();
                DAO.con.setAutoCommit(true);
            }catch(Exception ex){
                ex.printStackTrace();
            }
        }
    }

    @Test                                              5
    public void luuDKcuaSV_testChuan6(){
        // dang ki > 1LHP, dang ki cu nhieu LHP hon
```

```
ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();  1
//tao dang ki hoc thu nhat                                  2
SinhvienKhoa svk = new SinhvienKhoa();
svk.setId(1);
Kihoc kh = new Kihoc();
kh.setId(4);
MonhocKihoc mhkh = new MonhocKihoc();
mhkh.setId(1);
mhkh.setKihoc(kh);
Lophocphan lhp = new Lophocphan();
lhp.setId(3);
lhp.setMonhocKihoc(mhkh);
Dangkihoc dk = new Dangkihoc();
dk.setSinhvienKhoa(svk);
dk.setLophocphan(lhp);
listDK.add(dk);

//tao dang ki hoc thu hai                                   3
MonhocKihoc mhkh2 = new MonhocKihoc();
mhkh2.setId(2);
mhkh2.setKihoc(kh);
Lophocphan lhp2 = new Lophocphan();
lhp2.setId(9);
lhp2.setMonhocKihoc(mhkh2);
Dangkihoc dk2 = new Dangkihoc();
dk2.setSinhvienKhoa(svk);
dk2.setLophocphan(lhp2);
listDK.add(dk2);

try{ 4
    DAO.con.setAutoCommit(false);                           5
    boolean ok = dkhdao.luuDKcuaSV(listDK);
    Assert.assertTrue(ok);

    //lay ra kiem tra                                       6
    ArrayList<Dangkihoc> listDKnew =
      dkhdao.getDKcuaSV(svk.getId(), kh.getId());
    Assert.assertNotNull(listDKnew);
    Assert.assertEquals(2, listDKnew.size());
    Assert.assertEquals(svk.getId(),
      listDKnew.get(0).getSinhvienKhoa().getId());
    Assert.assertEquals(lhp.getId(),
      listDKnew.get(0).getLophocphan().getId());
    Assert.assertEquals(svk.getId(),
      listDKnew.get(1).getSinhvienKhoa().getId());
    Assert.assertEquals(lhp2.getId(),
      listDKnew.get(1).getLophocphan().getId());
```

```
        }catch(Exception e){                                    1
                e.printStackTrace();
        }finally{
                try{
                        DAO.con.rollback();
                        DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                        ex.printStackTrace();
                }
        }
}
```

```
@Test                                                           2
public void luuDKcuaSV_testNgoaile1(){
        //dang ki moi trung hoan toan voi dang ki cu
        ArrayList<Dangkihoc> listDK = new ArrayList<Dangkihoc>();
        //tao dang ki hoc thu nhat
        SinhvienKhoa svk = new SinhvienKhoa();
        svk.setId(2);
        Kihoc kh = new Kihoc();
        kh.setId(4);
        MonhocKihoc mhkh = new MonhocKihoc();
        mhkh.setId(4);
        mhkh.setKihoc(kh);
        Lophocphan lhp = new Lophocphan();
        lhp.setId(28);
        lhp.setMonhocKihoc(mhkh);
        Dangkihoc dk = new Dangkihoc();
        dk.setSinhvienKhoa(svk);
        dk.setLophocphan(lhp);
        listDK.add(dk);

        try{ 3
                DAO.con.setAutoCommit(false);                   4
                boolean ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertTrue(ok);

                //lay ra kiem tra                               5
                ArrayList<Dangkihoc> listDKnew =
                    dkhdao.getDKcuaSV(svk.getId(), kh.getId());
                Assert.assertNotNull(listDKnew);
                Assert.assertEquals(1, listDKnew.size());
                Assert.assertEquals(svk.getId(),
                    listDKnew.get(0).getSinhvienKhoa().getId());
                Assert.assertEquals(lhp.getId(),
                    listDKnew.get(0).getLophocphan().getId());
        }catch(Exception e){
```

```
                e.printStackTrace();
            }finally{
                try{
                    DAO.con.rollback();
                    DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                    ex.printStackTrace();
                }
            }
        }

        @Test
        public void luuDKcuaSV_testNgoaile2(){
            //dang ki moi khong co LHP nao
            ArrayList<Dangkihoc> listDK = null;

            try{
                DAO.con.setAutoCommit(false);
                boolean ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertFalse(ok);

                //test 2
                listDK = new ArrayList<Dangkihoc>();
                ok = dkhdao.luuDKcuaSV(listDK);
                Assert.assertFalse(ok);
            }catch(Exception e){
                e.printStackTrace();
            }finally{
                try{
                    DAO.con.rollback();
                    DAO.con.setAutoCommit(true);
                }catch(Exception ex){
                    ex.printStackTrace();
                }
            }
        }
}
```

5.4 CÂU HỎI ÔN TẬP

1. Hoàn thiện các phần giao diện cho 2 modul còn lại
2. Bổ sung các modul chức năng khác vào hệ thống

CHƯƠNG 6: RÀ SOÁT VÀ KIỂM THỬ HỆ THỐNG 1

6.1 THỰC HIỆN CÁC HOẠT ĐỘNG RÀ SOÁT 2

Việc thực hiện rà soát sẽ đảm bảo các tài liệu (đặc tả, phân tích, thiết kế…) không bị sai sót. 3
Thông thường, chúng ta sẽ rà soát dựa trên các checklist tương ứng.

Quy trình tiến hành các bước như sau: 4

- Bước 1: Lựa chọn và xây dựng checklist phù hợp với loại rà soát. 5
- Bước 2: Thực hiện rà soát theo các câu hỏi trong checklist, trường hợp có lỗi, có vấn đề cần mô tả rõ lỗi và yêu cầu chỉnh sửa.

Các phần tiếp theo giới thiệu các checklist tiêu biểu cho đặc tả, phân tích, thiết kế. 6

Một lưu ý khác là phần tài liệu đặc tả, phân tích, thiết kế về case study Hệ thống quản lí đăng kí tín chỉ tập trung vào các yêu cầu chức năng, kĩ thuật của hệ thống mà chưa đề cập tới các yêu cầu phi chức năng, thủ tục… vì vậy việc rà soát cũng sẽ tạm bỏ qua yêu cầu phi chức năng. Bên cạnh đó, nhằm cung cấp cho người học các checklist thông dụng và áp dụng được cho nhiều dự án, vì vậy có một số câu hỏi sẽ không cần thiết hoặc không liên quan tới Hệ thống quản lí đăng kí tín chỉ. Do vậy, một số câu hỏi trong checklist sẽ có trả lời là NA. 7

Lưu ý cột Y/N/NA là kết quả rà soát tương ứng với Y (không có lỗi), N (có lỗi), NA (không liên quan tới hệ thống hoặc không xác định). 8

6.1.1 Rà soát đặc tả 9

Để rà soát đặc tả, ta sử dụng checklist sau. 10

Bảng 6.2: Checklist để rà soát đặc tả yêu cầu của người dùng 11

Mã	Mục	Y/N/NA	Ghi chú
1	Các yêu cầu có đúng theo hợp đồng?		
2	Tất cả các yêu cầu đã được liệt kê chưa?		
3	Các yêu cầu đều rõ ràng?		
4	Mỗi yêu cầu đã được mô tả đầy đủ chưa?		
5	Các yêu cầu đã được xác định thống nhất xuyên suốt tài liệu chưa?		
6	Các yêu cầu đã được xác nhận chưa?		
7	Có các yêu cầu chức năng được thêm vào ngoài phạm vi hợp đồng hay không?		
8	Các yêu cầu về quản lý dự án đã được thêm vào chưa?		

12

9	Lý do cho các yêu cầu phát sinh có thỏa đáng hay không?		
10	Các interfaces với bên ngoài có tương thích không?		
11	Các yêu cầu giao diện người dùng đã hoàn thiện chưa?		
12	Các yêu cầu có thể được kiểm thử? Các yêu cầu có thể được sử dụng trực tiếp để thẩm định trong quá trình kiểm thử chấp nhận?		
13	Các yêu cầu hiệu năng đầy đủ và khả thi không?		
14	Các yêu cầu bảo mật đã được xác định chưa?		
15	Không có bất kỳ yêu cầu nào xung đột hoặc trùng lặp các yêu cầu khác?		
16	Mỗi yêu cầu đã được viết rõ ràng, súc tích, không mơ hồ?		
17	Mỗi yêu cầu đã được chỉnh sửa tất cả các lỗi nội dung và ngữ pháp?		
18	Các chức năng cần lưu ý về thời gian có được xác định và tiêu chí thời gian cho chúng đã được chỉ định?		
19	Các vấn đề quốc tế hóa đã được giải quyết thỏa đáng?		
20	Định dạng đã phù hợp định dạng của quy trình của tổ chức không?		
21	Tất cả các tham chiếu chéo nội bộ tới các yêu cầu khác có chính xác?		
22	Các yêu cầu có cung cấp một cơ sở đầy đủ cho phân tích phần mềm không?		
23	Các thuật toán nội tại cho các yêu cầu chức năng đã được định nghĩa?		
24	Mỗi yêu cầu đều trong phạm vi dự án?		
25	Tất cả các cảnh báo về an ninh và an toàn được xác định?		

1

Bên cạnh đó, một số vấn đề và câu hỏi chi tiết liên quan tới chức năng của hệ thống cần trao đổi như sau: 2

1. Tổng số tín chỉ được đăng kí có ngưỡng giá trị tối đa/tối thiểu là bao nhiêu? Có cần thiết lập không? 3

2. Các môn học đã học ở kỳ trước, các môn có thể cải thiện, các môn phải học lại...có được chú thích hay không? 4

3. Một sinh viên học 2 khoa, có một số môn học trùng thì bảng đăng kí hiển thị như nào? Đã xử lý chưa? 5

4. Việc thực hiện mở chế độ đăng ký cho sinh viên vào đăng ký như nào? Có cần chức năng riêng không? 6

5. Có yêu cầu về thiết kế giao diện (màu nền, bố cục, phông chữ, cỡ chữ...) không? 7

6.1.2 Rà soát phân tích 1

Phần này giới thiệu một checklist mẫu có thể dùng để rà soát tài liệu phân tích. 2

Bảng 6.3: Checklist rà soát phân tích phần mềm 3

Mã	Mục	Y/N/NA	Ghi chú
1	Các tài liệu phân tích phần mềm có tuân theo các tiêu chuẩn và hướng dẫn nêu trong kế hoạch dự án không?		
2	Kiến trúc phần mềm có tối ưu cho nền tảng được sử dụng cho việc triển khai?		
3	Trong trường hợp là sản phẩm, những mục sau đây đã được kiểm tra:		
	a. Hệ thống có tính khả chuyển đến các máy khác?		
	b. Interface với các files đã có.		
	c. Interface với phần mềm và phần cứng hiện có?		
4	Thiết kế của tệp hoặc cơ sở dữ liệu có tính đến các vấn đề sau đây (bất cứ nơi nào áp dụng):		
	a. Độ lớn và tổ chức?		
	b. Phương pháp truy cập (đối với hệ thống tập tin phẳng)?		
	c. Nếu được lập chỉ mục, chỉ mục là duy nhất / thay thế / thứ cấp?		
	d. Ghi lại bố cục?		
	e. Kiểm tra tính toàn vẹn?		
	f. Miền dữ liệu (loại, kích thước, phạm vi)?		
	g. Bảo mật?		
	h. Chuẩn hóa?		
5	Tài liệu có xác định các thành phần sau không:		
	a. Các báo cáo?		
	b. Các màn hình?		
	c. Các thành phần của chương trình và mã nguồn?		
6	Tài liệu có mô tả đầy đủ và chính xác về các phụ thuộc với bên ngoài?		
7	Thiết kế có đủ linh hoạt để đáp ứng yêu cầu trong tương lai?		
8	Thiết kế của giao diện và kết nối giữa các mô-đun có chính xác? Các kết nối dựa trên dữ liệu hay dựa trên thủ tục?		
9	Trong thiết kế màn hình, những điều sau đây đã được kiểm tra :		
10	a. Tất cả các trường đầu vào đã bố trí trên màn hình?		
	b. Dữ liệu xuất ra được ghi lại trên màn hình?		
	c. Bố cục có tương thích với các tài liệu đặc tả không?		
	d. Các trường thuộc tính được chỉ định cho màn hình có phù hợp với loại trường và chiều dài tương ứng được chỉ định trong		

4

	đặc tả?		
	e. Có sử dụng các tính năng đặc biệt của thiết kế màn hình phần mềm?		
	f. Có tooltip không?		
	g. Thiết kế màn hình có kết hợp xác thực dữ liệu cho trường đầu vào không?		
	h. Điều hướng màn hình có theo tiêu chuẩn giao diện đồ họa người dùng của tổ chức không?		
	Tất cả các xác thực được đặc tả trong yêu cầu người dùng đã được thêm vào?		
11	Thông báo lỗi, cảnh báo và thông báo thông tin đầy đủ?		
12	Tài liệu phân tích có bao gồm lựa chọn thiết kế hợp lý?		
13	Môi trường hoạt động tiêu chuẩn đã được đề cập?		
14	Đã bao gồm các quy trình hoạt động phần mềm hoặc tham chiếu đến chúng?		
15	Tài liệu phân tích đã gồm các phân tích rủi ro		
16	Đối với các báo cáo, những điều sau đây đã được kiểm tra:		
17	a. Các trường được chỉ định trong báo cáo tồn tại trong cơ sở dữ liệu, hoặc có thể đã được tính toán?		
	b. Chức năng được chỉ định trong các đặc tả yêu cầu người dùng đã được đề cập trong báo cáo?		
	c. Các tham số báo cáo đã được chỉ định?		
	d. Thứ tự sắp xếp báo cáo đã được chỉ định?		
	Tài liệu phân tích bao gồm các quy trình về an ninh không?		
18	Có bao gồm các thủ tục kiểm toán (audit)?		
19	Có bao gồm các thủ tục cho dự phòng (fallback)?		
20	Có bao gồm các thủ tục để sao lưu(backup)?		
21	Có bao gồm các thủ tục để khôi phục dữ liệu từ bản sao lưu?		
22	Có bao gồm các thủ tục thủ công cần thiết?		
23	Có bao gồm các chính sách lưu trữ?		
24	Đã bao gồm quy trình xử lý định kỳ (ví dụ: hàng ngày, hàng tháng)?		
25	Tất cả các giao tiếp giữa các thành phần đã được xác định?		
26	Các giao diện được cung cấp dễ sử dụng và nhất quán về định dạng?		
27	Tài liệu này có đủ để có thể tiến hành thiết kế phần mềm?		
28	Không có bất kỳ chức năng bổ sung nào được thêm vào (vượt quá phạm vi của hợp đồng)?		
29	Có phải tất cả các yêu cầu trong đặc tả yêu cầu được bao gồm trong tài liệu này?		
30	Không có bất kỳ thông tin cần thiết bị thiếu từ các yêu cầu?		

31	Hành vi dự kiến được ghi vào tài liệu cho tất cả các điều kiện lỗi có thể xảy ra?		

1

Việc rà soát cần thực hiện với toàn tài liệu phân tích, 2 ví dụ dưới đây minh hoạ cách rà soát và phát hiện các vấn đề trong pha phân tích. 2

Ví dụ 1: với kịch bản cho modul đăng ký học được viết như sau: 3

4

Use case	*Đăng kí học*
Actor	*Sinh viên*
Tiền điều kiện	*Sinh viên đã đăng nhập thành công, đang trong thời gian sinh viên được phép đăng kí môn học đầu học kì*
Hậu điều kiện	*Sinh viên đăng kí xong môn học cho học kì*
Kịch bản chính	*1. Sau khi đăng nhập, từ giao diện chính, sinh viên A chọn chức năng đăng kí học cho học kì tới.* *2. Giao diện đăng kí học hiện lên, có ô chọn kì học muốn đăng kí; bảng danh sách các môn học/lớp học phần đã đăng kí đang rỗng; nút tiếp tục và nút lưu chưa được active.* *3. Sinh viên chọn học kì tới trong danh sách và click vào nút tiếp tục.* *4. Giao diện hiện lên danh sách các môn học mà sinh viên có thể đăng kí trong học kì:*

TT	***Mã***	***Tên môn học***	***số tín chỉ***	***mô tả***
1	*INT17*	*Toán rời rạc 1*	*3*	
2	*INT55*	*Cơ sở dữ liệu*	*3*	
3	*INT68*	*Lập trình hướng đối tượng*	*3*	

5. Sinh viên click chọn môn Lập trình hướng đối tượng 5

6. Giao diện các lớp học phần hiện ra:

6

TT	***Mã***	***Tên***	***sĩ số tối đa***	***sĩ số hiện tại***	***Giảng viên***	***Phòng***	***ngày***	***kíp***
1	*hdt1*	*HDT-N1*	*60*	*11*	*Lê Thị C*	*201A3*	*thứ 2*	*1 (t1-15) + 3(t1-7)*
2	*hdt2*	*HDT-N2*	*60*	*19*	*Trần Văn X*	*201A3*	*thứ 3*	*4 (t1-15) + 6(t1-7)*

3	*hdt3*	*HDT-N3*	*60*	*35*	*Cao Văn K*	*201A3*	*thứ 5*	*2 (t1-15) + 3(t1-7)*

7. Sinh viên lick nhóm số 1
8. Hệ thống quay lại giao diện bước 2, bảng các môn đã chọn cập nhật lại:

TT	***Mã***	***Tên***	***sĩ số tối đa***	***sĩ số hiện tại***	***Giảng viên***	***Phòng***	***ngày***	***kíp***
1	*hdt1*	*HDT-N1*	*60*	*11*	*Lê Thị C*	*201A3*	*thứ 2*	*1 (t1-15) + 3(t1-7)*

(Lặp lại các bước 2-8 cho đến khi đạt số tín chỉ tối thiểu, nút lưu sẽ active)
9. Sinh viên click vào nút lưu.
10. Hệ thống báo thành công và quay về giao diện chính của sinh viên.

Ngoại lệ	*4. Không có môn nào để đăng kí* *6. Các lớp học phần đều hết slot hoặc trùng giờ học với các môn đã chọn trước nên không chọn được.*

Với kịch bản như trên, một số vấn đề và câu hỏi chi tiết liên quan tới chức năng của hệ thống cần trao đổi như sau:

- chưa thực hiện chọn ngành học (nhất là trường hợp học 2 ngành) ,
- cần bổ sung các ngoại lệ liên quan tới học nhiều ngành, đăng kí ít/nhiều hơn số tín chỉ cho phép, học lại, học nâng điểm
- trường hợp 1 lớp học phần học vào nhiều buổi, mỗi buổi liên quan đến 1 tuần khác nhau, 1 ngày khác nhau, 1 kíp khác nhau, 1 phòng khác nhau, 1 giảng viên khác nhau thì hiển thị giao diện Chọn lớp học phần hiển thị thế nào?

Ví dụ 2: với kịch bản cho modul nhập điểm được viết như sau:

Use case	*Nhập điểm*
Actor	*Giảng viên*
Tiền điều kiện	*Giảng viên đã đăng nhập thành công, đã có sinh viên đăng kí vào các lớp do giảng viên dạy*
Hậu điều kiện	*Giảng viên nhập xong điểm cho lớp học phần*

Kịch bản chính

1. Giảng viên chọn chức năng nhập điểm từu giao diện chính sau khi đăng nhập
2. Giao diện chọn môn học hiện lên: có ô chọn kì học đang active nhưng chưa chọn kì nào, danh sách các môn học do giảng viện dạy trong kì tương ứng đang trống.
3. Giàng viên chọn kì 2, năm học 2020-2021
4. Vẫn giao diện chọn môn học: danh sách các môn học do giảng viên dạy kì tương ứng được cập nhật:

TT	Mã	Tên môn học	số tín chỉ	mô tả
1	INT68	Lập trình hướng đối tượng	3	

5. Giảng viên click chọn môn Lập trình hướng đối tượng.
6. Giao diện các lớp học phần do giảng viên dạy hiện lên:

TT	Mã	Tên	sĩ số hiện tại	Phòng
1	hdt1	HDT-N1	5	201A3
2	hdt2	HDT-N2	19	201A3
3	hdt3	HDT-N3	35	201A3

7. Giảng viên click vào nhóm số 1
8. Giao diện nhập điểm chi tiết cho nhóm số 1 hiện lên: có thông tin nhóm số 1, môn học lập trình hướng đối tượng, và danh sách điểm sinh viên, dưới cùng là nút lưu:

TT	Mã sv	Họ và tên	CC	TBKT	BTL	Thi	TB chung	TB chữ
1	cn11	Cao văn A	5	5	5			
2	cn22	Lê Thị C	6	6	6			
3	cn33	Trần Minh D	7	7	7			
4	cn44	Sầm Ngọc S	8	8	8			
5	cn55	Vũ Tiến V	9	9	9			

9. Giảng viên nhập đầu điểm thi:
   A: 5, C:6, D:7, S:8, V:9 và click lưu
10. Hệ thống báo thành công và quay về giao diện chính của giảng viên.

Ngoại lệ	*4. Chọn học kì xong, không có môn học nào hiện lên* *6. Chọn môn học xong, không có lớp học phần nào hiện lên* *8. Chọn lớp xong, không có sinh viên nào hiện lên.*

1

Với kịch bản như trên, một số vấn đề và câu hỏi chi tiết liên quan tới chức năng của hệ thống cần trao đổi như sau: 2

- trường hợp có 2 giáo viên cùng dạy 1 môn, có cần kiểm soát ai là người nhập điểm không? 3
- Bổ sung ngoại lệ liên quan tới việc nhập điểm chưa hết danh sách lớp, nhập điểm sai định dạng (<0, >10, nhập chữ…), 2 giáo viên cùng dạy, trường hợp xin hoãn học.
- trường hợp 1 lớp học phần học vào nhiều buổi, mỗi buổi liên quan đến 1 tuần khác nhau, 1 ngày khác nhau, 1 kíp khác nhau, 1 phòng khác nhau, 1 giảng viên khác nhau thì hiển thị giao diện Chọn Lớp học phần hiển thị như thế nào?
- Nên bổ sung chức năng confirm cho nhập điểm hoặc giới hạn thời gian nhập điểm để đảm bảo giáo viên không chỉnh sửa điểm của các môn ở kì trước.

6.1.3 Rà soát thiết kế 4

Để rà soát tài liệu thiết kế, chúng ta sử dụng checklist sau 5

Bảng 6.4: Checklist rà soát thiết kế phần mềm 6

Mã	Mục	Y/N/NA	Ghi chú
1	Các functions chung và chia sẻ đã được xác định?		
2	Từng mô-đun của phân tích phần mềm được mô tả?		
3	Tất cả các điều kiện ngoại lệ đã được xử lý?		
4	Liệu thiết kế có phù hợp với phương pháp hướng đối tượng?		
5	Có các biện pháp bảo vệ chống data overflow?		
6	Mô tả chương trình có thể dễ dàng được lập trình?		
7	Các điều kiện kết thúc vòng lặp đã được quan tâm?		
8	Các điều kiện lồng nhau có phù hợp với tiêu chuẩn?		
9	Các mô-đun độc lập lẫn nhau?		
10	Bố cục màn hình, báo cáo, và việc sử dụng bảng, file trong tài tài liệu thiết kế có giống với mô tả trong tài liệu phân tích?		
12	Các thông báo lỗi đã được xác định?		
13	Các mô tả chương trình có phù hợp với logic trong tài liệu phân tích?		
Rà soát thiết kế cơ sở dữ liệu: Phần này cung cấp rà soát cho thiết kế cơ sở dữ liệu để đảm bảo rằng thiết kế cơ sở dữ liệu phù hợp với tiêu chuẩn			

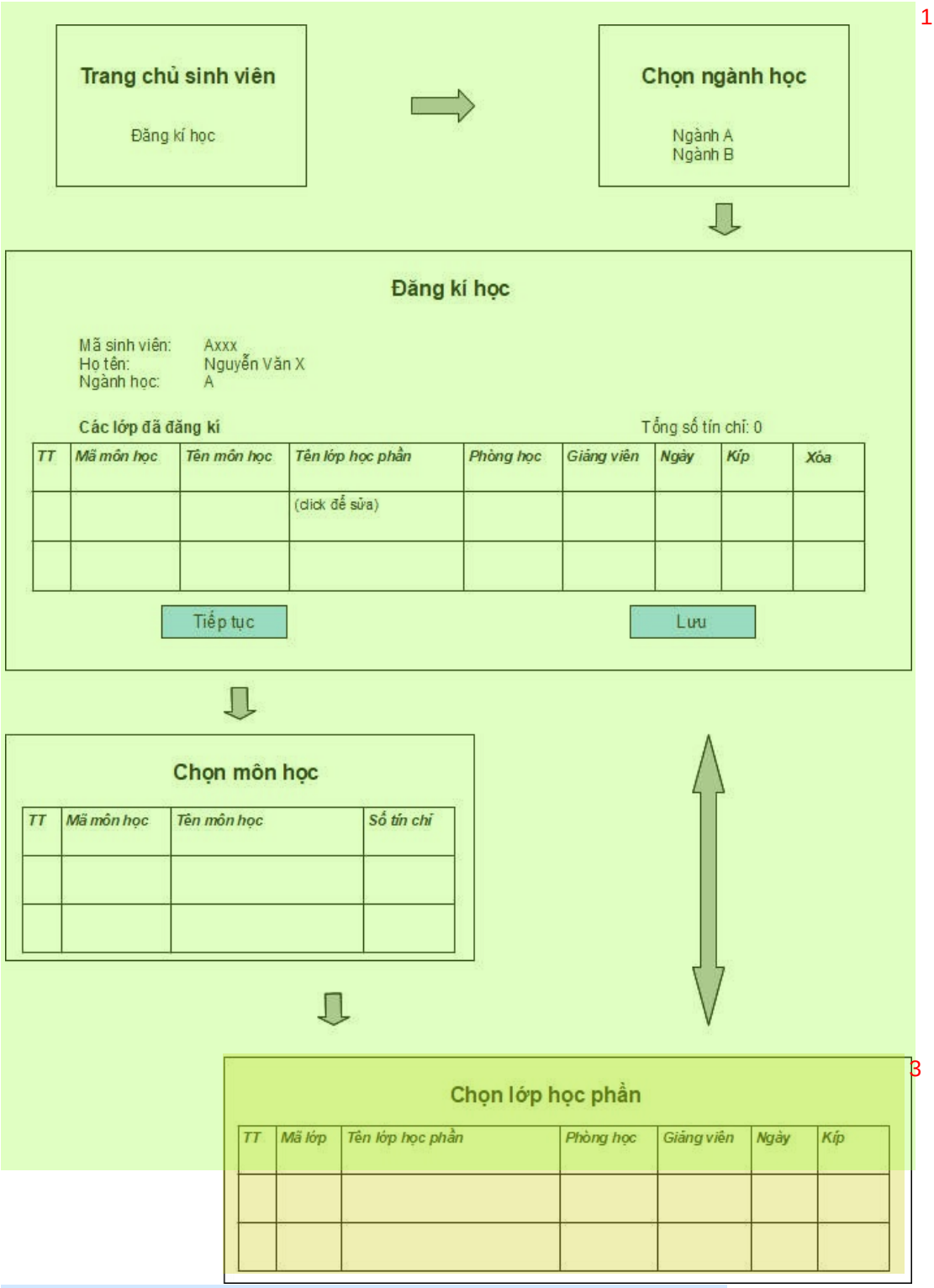
7

14	Có phải mỗi thực thể chuyển thành một bảng?		
15	Có phải mỗi thuộc tính đã được ánh xạ tới một trường trong bảng cơ sở dữ liệu của thực thể?		
16	Có phải mỗi trường khóa đã được ánh xạ tới một chỉ mục?		
17	Có tất cả các khóa ngoại được chỉ định NOT NULL (nếu có)?		
18	Các quy tắc toàn vẹn dữ liệu không vi phạm quy tắc toàn vẹn tham chiếu của hệ thống quản lý cơ sở dữ liệu quan hệ?		
19	Các triggers, events, objects như các thủ tục lưu trữ và các hàm và hành động đã được định nghĩa trong DBMS?		
20	Có các tham số free space được đặt tùy thuộc vào kích thước của bảng và mục đích sử dụng bảng? Có xác định đủ free space?		
21	Tất cả các truy vấn đã được phân tích để xác định các cột được sử dụng thường xuyên nhất trong các mệnh đề DISTINCT, GROUP BY, ORDER BY, và WHERE?		
22	Tất cả các chỉ mục được tạo đã được sử dụng?		
23	Không có vướng mắc trong việc đáp ứng các yêu cầu về hiệu năng?		
24	Cơ sở dữ liệu cung cấp ngầm dịch vụ khóa?		
25	Timeout cho khóa và phiên được tối ưu hóa?		
26	Các phân quyền và schemas đã được xác định đầy đủ để giải quyết vấn đề an ninh?		
27	Đặc quyền và quyền hạn, kiểm soát truy cập dữ liệu, v.v. đã được thực hiện?		

1

Việc rà soát cần thực hiện với toàn tài liệu thiết kế, 2 ví dụ dưới đây minh hoạ cách rà soát và phát hiện các vấn đề trong pha thiết kế. 2

Ví dụ 1: trường hợp giao diện cho module đăng ký học được thiết kế như dưới đây 3

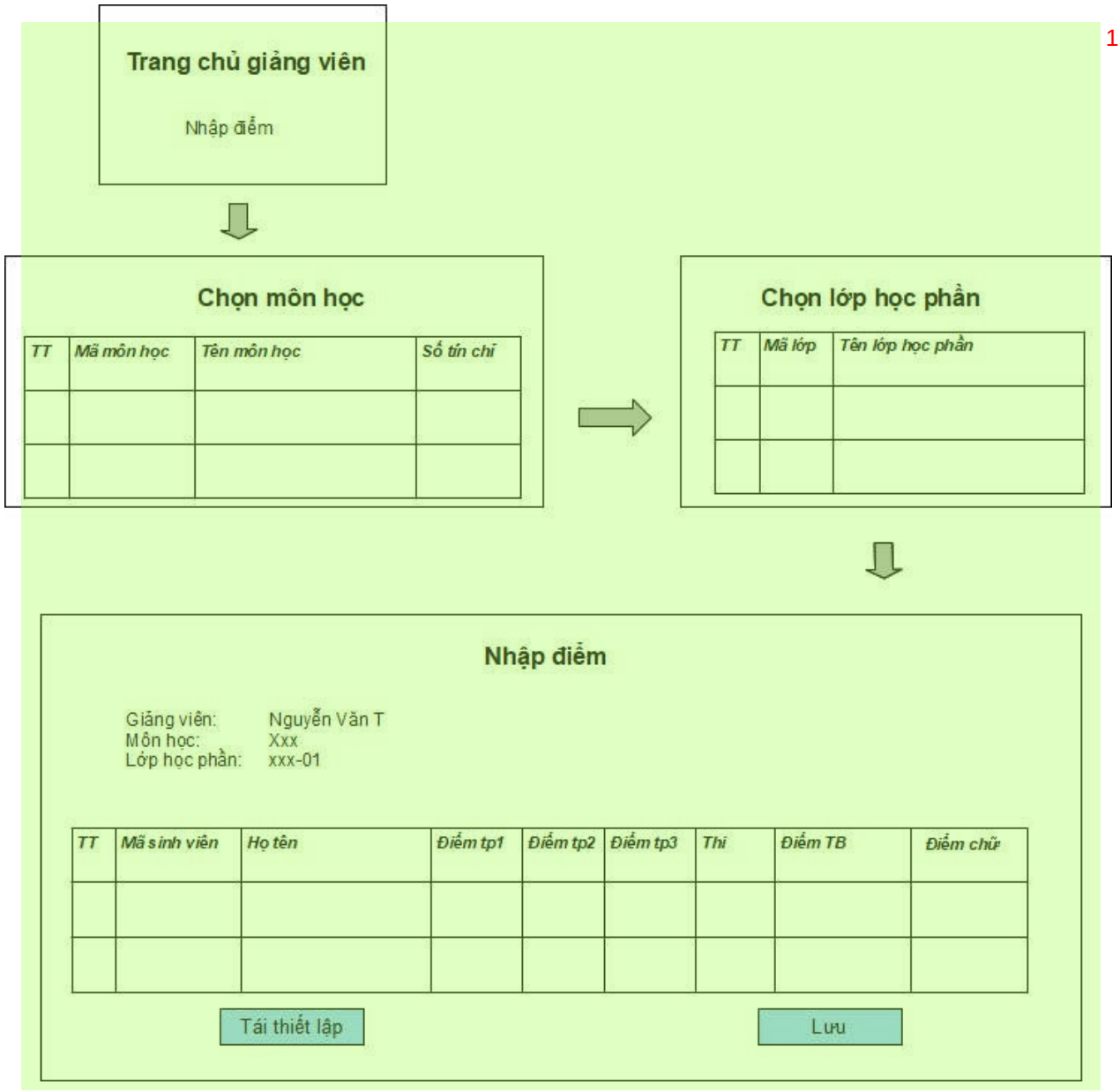


Hình 6.1: Ví dụ về thiết kế giao diện module đăng ký học có lỗi 2

Một số vấn đề và câu hỏi chi tiết cần trao đổi như sau: 4

1. Thiết kế giao diện Đăng kí học: So với đặc tả, thiếu: chọn thông tin kì học muốn đăng kí, cột sĩ số tối đa, cột sĩ số hiện tại. Nên bổ sung button đóng màn hình hoặc quay về màn hình chính. 1
2. Thiết kế giao diện Chọn môn học: So với đặc tả, thiếu cột mô tả. Nên bổ sung button đóng màn hình hoặc quay về màn hình chính.
3. Thiết kế Giao diện Chọn lớp học phần: So với đặc tả, thiếu cột sĩ số tôi đa, sĩ số hiện tại. Nên bổ sung button đóng màn hình hoặc quay về màn hình chính.
4. Thiết kế giao diện Chọn môn học: so với đặc tả, thiếu thông tin chọn học kì, thiếu cột mô tả. Nên bổ sung button đóng màn hình hoặc quay về màn hình chính.
5. Thiết kế giao diện Chọn lớp học phần: so với đặc tả, thiếu cột sĩ số thực, phòng học, ngày học, kíp học. Nên bổ sung button đóng màn hình hoặc quay về màn hình chính, bổ sung thông tin về môn học.

Ví dụ 2: trường hợp giao diện modul nhập điểm được thiết kế như dưới đây 2



1

Hình 6.2: Ví dụ về thiết kế giao diện modul nhập điểm lỗi 2

Với thiết kế như trên, một số vấn đề cần trao đổi như sau: 3

1. Thiết kế giao diện Nhập điểm: 4
   - Nên bổ sung button đóng màn hình hoặc quay về màn hình chính, bổ sung thông tin học kì.
   - Nút "Tái thiết lập" chưa được mô tả về chức năng, tái thiết lập lại toàn bộ về chưa nhập hay chỉ những sinh viên vừa nhập ở lần này?

2. Thiết kế giao diện Điểm môn học của sinh viên: so với phân tích, thiếu dòng TB Môn, Điểm chữ, Điểm hệ 4. 1

6.1.4 Rà soát code 2

Bảng dưới là một Checklist code mẫu sử dụng khi rà soát code. 3

Bảng 6.5: Checklist cho rà soát code 4

5

Mã	Câu hỏi	Y/N/NA	Ghi chú
1	Đã có coding guide chưa?		
2	Code có thể tự tài liệu không? (inline document)		
3	Các quy ước đặt tên có phù hợp với kế hoạch quản lý cấu hình không?		
4	Code đã được định dạng đúng chưa?		
5	Có một tập hợp các hàm chung được viết mà để tránh viết trùng hàm?		
6	Có bất kỳ code dư thừa hoặc rác?		
7	Có bất kỳ lable nào không được tham chiếu?		
8	Con trỏ đã được đặt thành NULL nếu cần thiết?		
9	Con trỏ không trỏ tới vùng ngoài phạm vi bộ nhớ?		
10	Có phải tất cả các chỉ số mảng nằm trong giới hạn?		
11	Có phải tất cả các chỉ số mảng được khởi tạo chính xác?		
12	Có phải tất cả các điều kiện rẽ nhánh chính xác?		
13	Tất cả các vòng lặp có thể kết thúc?		
14	Điều kiện để chấm dứt một vòng lặp có thực tế?		
15	Không có mẫu số bằng 0 trong phép chia?		
16	Có bất kỳ câu lệnh nào được đặt bên trong một vòng lặp có thể được đặt bên ngoài?		
17	Có phần nào trong code mà luồng thực hiện không bao giờ chạy tới?		
18	Có nhiều hơn 3 câu lệnh if lồng nhau?		
19	Các tham số khai báo và khi cài đặt của interface có giống nhau?		
20	Có biến nào khai báo mà không được sử dụng?		
21	Bộ nhớ được khởi tạo chính xác?		
22	Bộ nhớ động đã được cấp phát được giải phóng khi kết thúc?		
23	Các truy vấn trên table có thực thi bằng việc sử dụng các index ?		
24	Trạng thái lỗi được kiểm tra sau mỗi câu truy vấn SQL?		
25	Khóa được thực hiện trước khi update khi cần thiết?		

26	Các điều kiện sau đây đã được kiểm tra trong biểu thức:		
	a. Làm tròn số?		
	b. Có phép chia cho 0?		
27	Các yêu cầu về thời gian phản hồi sẽ được đáp ứng?		
28	Có một sự thay thế tốt hơn để cải thiện thời gian phản hồi?		
29	Có các kiểm tra sau đây đã được thực hiện:		
	a. Kiểm tra bảng và file trống?		
	b. Kiểm tra lỗi IO?		
30	Các thông báo lỗi có rõ ràng không? Các thông báo lỗi có đầy đủ không?		
31	Tất cả các điều kiện lỗi được bắt và xử lý?		
32	Trong các biểu thức số học, có vấn đề sau đã được xác định:		
	a. Thứ tự xử lý không rõ ràng?		
	b. Có cần cuộn ngang để đọc toàn bộ biểu thức không?		
	c. Tả các dấu ngoặc đơn đã đóng đúng? Có đảm bảo đúng trình tự xử lý không?		
	d. Làm tròn số được thực hiện cùng với các biểu thức?		
	e. Phép chia thực hiện cùng với biểu thức khác?		
	f. Có các biểu thức sử dụng trực tiếp các trường trong bảng ở trong biểu thức?		
33	Trong biểu thức quan hệ, các vấn đề sau cần được xác định:		
	a. So sánh cùng kiểu dữ liệu?		
	b. Có nhiều hơn một kết quả cho một biểu thức?		
	c. Biểu thức có phục vụ cho mục đích mà nó được sử dụng?		
	d. Có cần thanh cuộn ngang để đọc toàn bộ biểu thức?		
34	Trong biểu thức logic, các vấn đề sau cần được xác định:		
	a. Biểu thức logic có phục vụ cho mục đích mà nó được sử dụng?		
	b. Mỗi biểu thức quan hệ sử dụng kết quả là true hoặc false?		
	c. Mỗi biểu thức quan hệ được đặt bên trong ngoặc đơn?		
	d. Tại bất kỳ thời điểm nào chỉ có hai biểu thức quan hệ được so sánh?		
	e. Có cần thanh cuộn ngang để xem được toàn bộ biểu thức?		
35	Trong các hàm sử dụng file và table, các vấn đề sau cần được xác định:		
	a. Có table hay file nào được mở sớm hơn so với yêu cầu?		
	b. Có bảng hay file nào vẫn mở khi kết thúc hàm?		
36	Trong khai báo biến, các vấn đề sau cần được xác định:		
	a. Tất các các biến được khai báo là toàn cục hoặc tĩnh có thực sự cần phải là toàn cục hoặc tĩnh không?		
	b. Có biến nào không cần thiết được khai báo không?		
	c. Có tên biến nào bị xung đột với từ khóa của ngôn ngữ lập		

	trình được sử dụng không?		
	d. Có bất kì hard coding nào trong code không?		

1

6.2. THỰC HIỆN TEST CHỨC NĂNG 2

Việc thực hiện test chức năng nhằm đảm bảo các chức năng của hệ thống hoạt động theo đúng yêu cầu của đặc tả. 3

Quy trình tiến hành các bước như sau: 4

- Bước 1: Xây dựng checklist các nội dung cần test cho từng chức năng (tuỳ chọn). 5
- Bước 2: Thực hiện viết test cases
  - GUI (giao diện)
  - chức năng
  - luồng, nghiệp vụ
  - khác
- Bước 3: Thực hiện chuẩn bị data test (nếu cần)
- Bước 4: thực hiện test và ghi nhận kết quả pass/false. Trường hợp false cần mô tả rõ lỗi và trao đổi với Lập trình viên để sửa.

Ta áp dụng với hệ thống Quản lí đăng kí tín chỉ với các bước 1,2,3. Riêng Bước 4, ta cần có hệ thống chạy được thì mới thực hiện được. 6

Bước 1: xây dựng checklist chung 7

Bảng 6.6: Checklist test case chức năng 8

	Question
Chung	Giao diện người dùng có đúng với prototype không?
	Danh sách menu và màn hình có thống nhất không?
	Các link html có hoạt động không?
	Tất cả các cửa sổ đều có thể truy cập từ toolbar?
	Tất cả các trang web/cửa sổ đều có thể truy cập từ menu?
	Các màn hình được gọi từ button có hiển thị đúng không?
	Các label, textbox, combobox, etc có đúng font chữ, cỡ chữ, màu chữ theo yêu cầu không?
	Căn lề, độ rộng, khoảng cách có đồng bộ và đúng theo yêu cầu không?
	Dữ liệu kiểu chữ căn lề trái?

9

	Dữ liệu kiểu số căn lề phải?
	Các form có được bố trí hợp lý và dễ sử dụng không?
	Nếu các phím tắt được sử dụng, các phím tắt được gán có hoạt động đúng không?
	Màn hình KHÔNG có lỗi chính tả, cấu trúc câu, ngữ pháp?
	Nếu các chữ viết tắt được sử dụng, nó có thống nhất trong tất cả các giao diện không? Người dùng có hiểu được không?
	Định dạng số, ngày tháng, thời gian có nhất quán không?
	Có hiển thị mô tả khi di chuyển chuột vào tooltip không?
	Nếu có tooltip, nó có ý nghĩa hoặc có hữu ích không?
	Có một button hoặc check box được xét mặc định chưa?
	Con trỏ di chuyển lần lượt theo thứ tự: từ trái qua phải, từ trên xuống dưới khi nhấn Tab liên tục?
	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái khi nhấn Shift-Tab ?
	Nếu chuột không focus vào button nào thì đã thực hiện chức năng của button chính khi nhấn Enter chưa?
	Nếu đang focus vào 1 button thì đã thực hiện chức năng của button khi nhấn Enter chưa?
	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện khi nhấn Ctrl - và Ctrl + không ?
	Các trường bắt buộc có dấu * chưa?
	Thông tin bản ghi hiển thị trên giao diện có đúng với thông tin bản ghi được lưu trong Database không?
	Kiểm tra thay đổi ngôn ngữ, ngôn ngữ thay đổi có đúng nội dung không?
	Mã captcha có hoạt động hay không?
	Các trường KHÔNG cho phép nhập dữ liệu đã cùng màu theo đúng yêu cầu chưa?
Validate các trường dữ liệu	**Đối với các kiểu dữ liệu kiểu text, string, etc:**
	Kiểm tra max length có đúng yêu cầu không?
	Nếu trường dữ liệu có phân biệt chữ hoa và chữ thường, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG phân biệt chữ hoa và chữ thường, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?

	Nếu trường dữ liệu KHÔNG cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?
	Đối với các kiểu dữ liệu integer, float, double, etc:
	Kiểm tra max length có đúng yêu cầu không?
	Kiểm tra các giá trị biên có đúng yêu cầu không?
	Nếu trường dữ liệu cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép nhập ký tự chữ, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép nhập dữ liệu chữ, dữ liệu nhập vào đã thoả mãn chưa?
	Kiểm tra lỗi chia cho 0?
	Đối với kiểu dữ liệu time, date:
	Kiểm tra max length có đúng yêu cầu không?
	Kiểm tra ngày, tháng, năm, giờ, phút, giây có hợp lệ không?
	Nếu trường dữ liệu cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép các ký tự đặc biệt, dữ liệu nhập vào đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu KHÔNG cho phép null, đã thoả mãn chưa?
	Nếu trường dữ liệu cho phép nhập ký tự chữ, dữ liệu nhập vào đã thoả mãn chưa?
	Có cho phép người dùng kích chọn ngày không?
	Có cho phép người dùng nhập ngày không?
	Nếu trường dữ liệu KHÔNG cho phép nhập dữ liệu chữ, dữ liệu nhập vào đã thoả mãn chưa?
	Kiểm tra format có đúng yêu cầu không?
	Đối với kiểu dữ liệu file:
	Đảm bảo giới hạn tối thiểu file không?
	Đảm bả giới hạn tối đa file không?
	Cho phép upload nhiều file một lúc không?
	Có cho phép null không?
	Có yêu cầu loại file nào được phép upload không?
Thông báo	Thông báo lỗi chung của cả dự án có thống nhất không?
	Thông báo lỗi đã báo lỗi đúng chưa?

1

	Thông báo lỗi viết bằng ngôn ngữ tự nhiên, dễ hiểu cho người dùng chưa?
	Thông báo lỗi không đổ lỗi cho người dùng và gợi ý cách sửa lỗi?
	Khi có một hành động ảnh hưởng đến hệ thống (sửa, xoá) được thực hiện, người dùng có được hỏi lại để xác nhận không?
	Thông báo cho ngoại lệ của các điều kiện
	Thông báo cho đầu vào không hợp lệ
	Thông báo cho điều kiện biên
Các trạng thái	Các trạng thái có hiển thị đúng không?
	Màu trạng thái có khác nhau để người dùng dễ phân biệt không?
Database	
Thêm mới	Khi thêm mới thành công một bản ghi thì nó hiển thị dưới danh sách bản ghi không ?
	Khi thêm mới một bản ghi bị thiếu những trường bắt buộc thì có cảnh báo đến người dùng không ?
	Khi thêm mới không thành công, bản ghi sẽ không được lưu vào DB ?
	Khi thêm mới thành công, bản ghi sẽ được lưu vào DB ?
Xóa	Trước khi thực hiện một thao tác ảnh hưởng tới hệ thống (ví dụ xóa một bản ghi), người dùng có được yêu cầu xác nhận không?
	Khi xóa một bản ghi có liên quan đến một bản ghi của một bảng khác thì dữ liệu tại bản ghi có liên quan có được cập nhật không ?
	Khi xóa thành công, bản ghi không bị xóa khỏi DB mà chỉ thay đổi trạng thái?
Sửa	Khi bản ghi được sửa bị duplicate với bản ghi khác đã có trong hệ thống thì có cảnh báo cho người dùng không ?
	Khi bản ghi được sửa thành công có hiển thị thông báo đến người dùng không ?
	Khi sửa một bản ghi có liên quan đến một bản ghi tại một bảng khác thì dữ liệu tại bản ghi có liên quan có được cập nhật không ?
	Khi bản ghi sửa thành công, dữ liệu được update trong DB ?
Chung	Đã có cơ chế xử lý truy cập đồng thời tới 1 bản ghi để update?
	Đã Rollback dữ liệu khi 1 giao dịch không thành công?
	Đã thực hiện trim space ở đầu và cuối trường dữ liệu khi lưu vào DB khi nhập giá trị text có giá trị space ở đầu và cuối ?
	Các giá trị combobox có được lưu dưới dạng id: 0, 1, 2...
Nghiệp vụ	
	Đã phủ hết các kịch bản valids theo đặc tả?
	Đã phủ hết các kịch bản invalids theo đặc tả?
	Quy trình thực hiện đã tối ưu số thao tác?
	Quy trình thực hiện có dễ hiểu, dễ dùng?

	Chức năng/giao diện có dễ chỉnh sửa trong trường hợp người dùng có nhu cầu thay đổi/thêm mới?

2

Bước 2: Thực hiện viết test cases 1

Các nội dung chính của Test case bao gồm mô tả input (mục đích kiểm thử, các bước thực hiện, test data) và kết quả mong muốn. 3

Một lưu ý là một số test cases bên dưới không yêu data test (ví dụ: test giao diện, test hành động…), nhưng một số test case cần có data test (ví dụ: test case về chức năng…). Bản test case bên dưới đã mô tả các ràng buộc về data test ở cột Các bước thực hiện (Bảng 6.7…). Trong phạm vi case study ở tài liệu này, các ràng buộc khá đơn giản, việc xây dựng test data cụ thể có thể dễ dàng thực hiện theo mô tả ở Cột các bước thực hiện. Vì vậy, bảng Test case bên dưới sẽ không trình bày chi tiết test data, dành cho người học tự thực hiện. 4

Module Đăng kí tín chỉ: 5

Bảng 6. 7: Hướng dẫn xây dựng Test case chức năng cho module Đăng kí tín chỉ 6

Mã trường hợp kiểm thử	**Mục đích kiểm thử**	**Các bước thực hiện**	**Kết quả mong muốn**
DKH_1			
	Chức năng Đăng ký học 1.1		
	Precond: Sinh viên đã đăng nhập thành công, đang trong thời gian sinh viên được phép đăng kí môn học đầu học kì Chọn Đăng ký --> Hiển thị màn hình Đăng ký		
	Giao diện (Phần này viết các trường hợp kiểm thử cho giao diện chung và các giao diện cho các control)		
	Giao diện chung		
	Giao diện Chọn ngành học		
	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình

7

			- Form được bố trí hợp lý và dễ sử dụng
DKH_2	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọn ngành học 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Ngành học: Dropdownlist + Kỳ học: Dropdownlist 4. Button: + Vào đăng ký
DKH_3	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
DKH_4	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
DKH_5	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
DKH_6	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Đăng kí học		

DKH_7	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
DKH_8	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Đăng kí học 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Mã SV: textview + Họ tên: textview + Ngành học: Textview + Tổng số tín chỉ: textview + Các lớp đã đăng ký: table ++ TT, Mã môn học, Tên môn học, Tên lớp học phần, Xem lịch, Sửa, Xoá 4. Button: + Table: Sửa + Table: Xoá + Về trang chủ + Tiếp tục + Lưu đăng kí 5. Hyperlink + Tên lớp học phần + Xem lịch
DKH_9	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
DKH_10	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.

DKH_11	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
DKH_12	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện: Chọn môn học		
DKH_13	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
DKH_14	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọn môn học 2. Focus được set vào trường đầu tiên có thể edit: Chọn môn học 3. Hiển thị đầy đủ các trường : +Table: TT, Mã Môn học, Tên môn học, Số tín chỉ, Chọn 4. Button + Chọn: Tick TT + Quay lại 5. Hyperlink + Mã môn học

DKH_15	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
DKH_16	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
DKH_17	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
DKH_18	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện: Chọn lớp học phần		
DKH_19	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
DKH_20	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọn lớp học phần 2. Hiển thị đầy đủ các trường : +Table: TT, Mã lớp, Tên lớp học phần, Xem lịch, chọn 4. Button

			+ Chọn môn học: Tick TT + Quay lại
DKH_21	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
DKH_22	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
DKH_23	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
DKH_24	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện: Lịch học		
DKH_25	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng

DKH_26	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Lịch học 2. Hiển thị đầy đủ các trường : +Lớp học phần + Table: TT, Tuần, Thứ, Kíp, Giảng viên, Phòng học 4. Button + Quay lại
DKH_27	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
DKH_28	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
DKH_29	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
DKH_30	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Validate các trường thông tin **Chú ý:** *Khi validate thông tin một trường, tất cả các trường khác hợp lệ*		
	Trường Ngành học, Dropdownlist		
DKH_31	Kiểm tra giá trị mặc định	1. Trên màn hình Đăng ký 2. Kiểm tra giá trị mặc định của trường: Ngành học	Giá trị mặc định của Ngành: *để trống*
DKH_32	Kiểm tra danh sách ngành học	1. Focus chuột vào dropdown list ngành học	Hệ thống hiển thị danh sách các ngành học

	Trường Học kì, Dropdown list		
DKH_33	Kiểm tra giá trị mặc định	1. Trên màn hình Đăng ký 2. Kiểm tra giá trị mặc định của trường: Học kì	Giá trị mặc định của học kì: Học kì hiện tại
DKH_34	Kiểm tra danh sách học kì	1. Focus chuột vào dropdown list ngành học	Hệ thống hiển thị danh sách các học kì
	Kiểm tra button, link		
DKH_35	Kiểm tra button Vào đăng ký	Kiểm tra cho phép click button Vào đăng ký	Luôn cho phép click button Vào đăng ký
DKH_36	Kiểm tra button Lưu đăng kí	Trường hợp chưa có môn học được đăng kí Kiểm tra cho phép click button Lưu môn học	Không cho phép click nút Lưu môn học
DKH_37	Kiểm tra button Lưu đăng kí	Trường hợp đã chọn được môn học Kiểm tra cho phép click button Lưu môn học	Cho phép click nút Lưu môn học
DKH_38	Kiểm tra button Sửa/Xóa	Trường hợp chưa có môn học được đăng kí Kiểm tra cho phép click button Sửa/Xóa	Không cho phép click nút Sửa/Xóa
DKH_39	Kiểm tra button Sửa/Xóa	Trường hợp đã chọn được môn học Kiểm tra cho phép click button Sửa/Xóa	Cho phép click nút Sửa/Xóa
DKH_40	Kiểm tra click Link Đăng kí học	Click vào link Đăng kí học tại tab menu	Hệ thống chuyển hướng đến màn hình đăng kí môn học
DKH_41	Kiểm tra button Chọn môn học	Trường hợp - Trong thời gian đăng kí - Môn học mong muốn ko trùng thời gian với môn học khác - CL > 0	Cho phép click nút Chọn môn học
DKH_42	Kiểm tra button Chọn	Trường hợp ngoài thời gian đăng kí	Hệ thống hiển thị ngoài thời gian đăng ký

	môn học		Không cho phép click chọn môn học
DKH_43	Kiểm tra button Chọn môn học	Trường hợp môn học mong muốn chọn có tiết bắt đầu và số lượng tiết trùng với khoảng thời gian của môn học khác	Không cho phép click chọn môn học
DKH_44	Kiểm tra button Chọn môn học	Trường hợp môn học còn lại: CL = Hết	Không cho phép click chọn môn học
DKH_45	Kiểm tra bỏ chọn môn học	Trường hợp môn học đã được chọn	Cho phép bỏ chọn môn học
	Chức năng		
	Màn hình Đăng kí học		
DKH_46	Kiểm tra hiển thị đúng thông tin - trường hợp chưa đăng kí	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách	Hiển thị đúng thông tin sinh viên Hiển thị đúng thông tin ngành học Tổng số tín chỉ = 0 Danh sách lớp đăng kí là trống
DKH_47	Kiểm tra hiển thị đúng thông tin - trường hợp đã đăng kí	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách	Hiển thị đúng thông tin sinh viên Hiển thị đúng thông tin ngành học Tổng số tín chỉ = tổng số tín chỉ danh sách lớp bên dưới Danh sách lớp đăng kí tương ứng với các bản ghi trong tblDangKiHoc
	Màn hình Chọn môn học		

DKH_48	Kiểm tra hiển thị đúng thông tin - trường hợp không có môn học	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn button [Tiếp tục] để chuyển sang màn hình Chọn môn học	Không hiển thị môn học nào
DKH_49	Kiểm tra hiển thị đúng thông tin - trường hợp có danh sách môn học	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn button [Tiếp tục] để chuyển sang màn hình Chọn môn học	Hiển thị đúng thông danh sách môn học tương ứng với tblMonHocKiHoc
	Màn hình Chọn lớp học phần		
DKH_50	Kiểm tra hiển thị đúng thông tin - trường hợp không có môn học	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn button [Tiếp tục] để chuyển sang màn hình Chọn môn học 5. Tại màn hình Chọn môn học, chọn 1 môn học (mà	Màn hình Chọn lớp học phần trống

		không có lớp học phần)	
DKH_51	Kiểm tra hiển thị đúng thông tin - trường hợp có danh sách môn học	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách	Màn hình Chọn lớp học phần hiển thị đúng danh sách lớp tương ứng với tblLopHocPhan
	Luồng Đăng kí học		
DKH_52	Kiểm tra thực hiện thao tác [Lưu] khi danh sách lớp học đã đăng kí trống	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Lưu đăng kí]	Đăng ký học không thành công

DKH_53	Kiểm tra đăng kí học khi số tín chỉ đủ điều kiện, các môn và lớp học phần đúng yêu cầu	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần thuộc trường Tên lớp học phần tại màn 'Chọn lớp học phần' 7. Lặp lại các bước từ 4 đến 6 để chọn các lớp học phần còn lại sao cho đủ số X tín chỉ 8. Nhấn chuột vào button [Lưu đăng kí] tại màn 'Đăng kí học'	Thông báo đăng ký học thành công. Màn hình Đăng kí học hiển thị đúng danh sách các môn đã đăng kí. Sĩ số hiện tại của các lớp học phần tương ứng tăng 1 Tổng số tín chỉ đúng tblDangKiHoc bổ sung các bản ghi tương ứng
DKH_54			Thông báo đăng ký học thành công. Màn hình Đăng kí học hiển thị đúng danh sách các môn đã đăng kí. Sĩ số hiện tại của các lớp học phần tương ứng tăng 1 Tổng số tín chỉ đúng tblDangKiHoc bổ sung các bản ghi tương ứng
DKH_55	Kiểm tra đăng ký học với số tín chỉ vượt ngưỡng cho phép		Thông báo đăng ký không thành công, số tín chỉ quá số lượng qui định tblDangKiHoc không thay đổi

DKH_56	Kiểm tra đăng ký học với số tín chỉ chưa đạt yêu cầu		Thông báo đăng ký không thành công, số tín chỉ chưa đạt mức qui định tblDangKiHoc không thay đổi
DKH_57	Kiểm tra chọn môn học đã có trong danh sách các môn học đã đăng ký	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn 1 môn học trùng với môn học có trong danh sách các môn đã đăng ký trước đó tại màn 'Chọn môn học'	Đăng ký không thành công, thao tác chọn bị hủy
DKH_58	Kiểm tra việc thao tác chọn môn học khi danh sách môn học rỗng	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học'	Đăng ký không thành công, giao diện trở lại màn 'Đăng kí học'

DKH_59	Kiểm tra việc chọn lớp học phần khi sĩ số tối đa bằng sĩ số hiện tại	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần (có sĩ số tối đa bằng sĩ số hiện tại) tại màn 'Chọn lớp học phần'	Đăng ký không thành công, thao tác chọn bị hủy
DKH_60	Kiểm tra khi chọn học phần đã trùng lịch học, kíp học với học phần đã đăng ký trước đó	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần (trùng lịch học với lịch học của các lớp học phần đã đăng kí) tại màn 'Chọn lớp học phần'	Đăng ký không thành công, thao tác chọn bị hủy

DKH_61	Khi chưa nhấn lưu thao tác đăng ký, tài khoản bi đăng xuất hoặc thực hiện load lại trang, dữ liệu có được lưu lại?	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần thuộc trường Tên lớp học phần tại màn 'Chọn lớp học phần' 7. Lặp lại các bước từ 4 đến 6 để chọn các lớp học phần còn lại sao cho đủ số tín chỉ 8. Nhấn phím f5 trên bàn phím	tblDangKiHoc không thay đổi , màn hình trở về trạng thái mặc định
DKH_62	Kiểm tra đăng kí học khi sĩ số tối đa bằng sĩ số hiện tại	1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần thuộc	Đăng ký không thành công, thông báo lớp học phần đã đủ số lượng

		trường Tên lớp học phần tại màn 'Chọn lớp học phần' 7. Lặp lại các bước từ 4 đến 6 để chọn các lớp học phần còn lại sao cho đủ số tín chỉ 8. Nhấn chuột vào button [Lưu] tại màn 'Đăng kí học'	
DKH_63	Kiểm tra đăng kí học khi có 2 người cùng đăng kí 1 lớp học phần, lớp học phần chỉ còn đăng kí thêm đúng 1 sv	Sử dụng 2 trình duyệt và 2 tài khoản sinh viên thực hiện đăng kí học cùng 1 lớp học phần. Các thao tác sau tuần tự thực hiện cùng lúc: Cả 2 cùng thực hiện các thao tác như sau theo trình tự thời gian 1. Nhấn vào button [Đăng kí học] tại màn 'Trang chủ sinh viên' 2. Chọn ngành học M tại màn 'Chọn ngành học' 3. Chọn học kỳ N từ danh sách 4. Nhấn chuột vào button [Tiếp tục] tại màn 'Đăng kí học' 5. Chọn môn học thuộc trường Tên môn học tại màn 'Chọn môn học' 6. Chọn lớp học phần thuộc trường Tên lớp học phần tại	Chỉ có 1 sinh viên đăng kí thành công Sinh viên còn lại nhận thông báo lớp đã đủ sinh viên, không đăng kí được nữa.

		màn 'Chọn lớp học phần' 7. Nhấn chuột vào button [Lưu] tại màn 'Đăng kí học'	
	Sửa môn học đã đăng kí		
DKH_64	Thực hiện sửa nhưng chưa lưu	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Sửa] sau lớp học phần muốn sửa 4. Không click button [Lưu], thoát ra rồi đăng nhập lại, vào lại màn hình Đăng kí học	không thực hiện sửa bản ghi trong tương ứng trong tblDangKiHoc màn hình đăng kí học hiển thị đủ danh sách môn học, kể cả môn đã thực hiện sửa từ trước
DKH_64	Sửa thành công	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Sửa] sau lớp học phần muốn sửa, thực	Thông báo cập nhật thành công. Màn hình Đăng kí học hiển thị đúng danh sách các môn đã đăng kí, với lớp học phần vừa sửa. tblDangKiHoc sửa bản ghi tương ứng

		hiện sửa sang 1 lớp học phần khác 4: Nhấn vào button [Lưu]	
	Xóa môn học đã đăng kí		
DKH_66	Kiểm tra thông báo người dùng khi thực hiện thao tác xóa	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa	Thông báo cho người dùng có chắc chắn muốn xóa hay không.
DKH_67	Kiểm tra thông tin của "lớp học học muốn xóa" trên giao diện người dùng sau khi thực hiện thao tác xóa	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa 4. Kiểm tra thông tin của "lớp học phần vưà xóa" trên giao diện người dùng	Thông tin của "lớp học muốn xóa" không hiện trên listbox Các lớp đã đăng kí, thao tác xóa thành công
DKH_68	Thực hiện xoá nhưng chưa lưu	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa 4. Không click button [Lưu], thoát ra rồi đăng nhập lại, vào lại màn hình Đăng kí học	không thực hiện xoá bản ghi trong tương ứng trong tblDangKiHoc màn hình đăng kí học hiển thị đủ danh sách môn học, kể cả môn đã thực hiện xoá từ trước

DKH_69	Xoá thành công	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa 4: Nhấn vào button [Lưu]	Thông báo cập nhật thành công. Màn hình Đăng kí học hiển thị đúng danh sách các môn đã đăng kí, trừ môn vừa xoá. Tổng số tín chỉ giảm đúng tblDangKiHoc xoá bản ghi tương ứng
DKH_70	Khi xóa học phần đã đăng ký có đăng ký được các học phần trùng lịch học , trùng môn học với học phần đó hay không?	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa 4. Thực hiện các thao tác đăng ký lại học phần vừa xóa (lớp học phần đủ điều kiện để đăng ký)	Thông báo cập nhật thành công. Màn hình Đăng kí học hiển thị đúng danh sách các môn đã cập nhật. Tổng số tín chỉ cập nhật đúng tblDangKiHoc xoá bản ghi tương ứng với lớp đã xoá và thêm bản ghi tương ứng với lớp đã đăng kí bổ sung
DKH_71	Xoá không thành công	1: Nhấn chuột vào button [Đăng kí học] từ màn 'Trang chủ sinh viên' 2: Chọn ngành học M từ màn 'Chọn ngành học' 3: Nhấn vào button [Xóa] sau lớp học phần muốn xóa 4: Nhấn vào button [Lưu đăng kí]	Thông báo tổng số tín chỉ < số tín chỉ tối thiểu, không thực hiện cập nhật. tblDangKiHoc không thay đổi.

Module Nhập điểm

Bảng 6.8: Hướng dẫn xây dựng Test case chức năng cho module Nhập điểm 1

2

Mã trường hợp kiểm thử	Mục đích kiểm thử	Các bước thực hiện	Kết quả mong muốn
	Chức năng 1.1	**Giảng viên nhập điểm**	
	Precond: Giảng đã đăng nhập thành công Chọn Nhập Điểm -> Hiển thị màn hình Nhập điểm		
	Giao diện (Phần này viết các trường hợp kiểm thử cho giao diện chung và các giao diện cho các control)		
	Giao diện chung		
	Giao diện Chọn kì học		
ND_1	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
ND_2	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọc kì học 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Kì học: Dropdownlist 4. Button: + Tiếp tục
ND_3	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện

ND_4	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
ND_5	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
ND_6	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuột không focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Chọn môn học		
ND_7	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
ND_8	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọc môn học 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Table: TT, Mã môn học, Tên môn học, Số tín chỉ 4. Button: + Quay lại 5. Hyperlink:

			+ Mã môn học
ND_9	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
ND_10	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
ND_11	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
ND_12	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuột không focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện: Chọn lớp học phần		
ND_13	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng

ND_14	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Chọc lớp học phần 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Table: TT, Mã lớp, Tên lớp học phần 4. Button: + Quay lại 5. Hyperlink: + Mã lớp
ND_15	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
ND_16	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
ND_17	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
ND_18	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuột không focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện: Nhập điểm		
ND_19	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu

			trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
ND_20	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Nhập điểm 2. Hiển thị đầy đủ các trường : + Giảng viên: Tên giảng viên + Môn học: Tên môn học + Lớp học phần: Mã lớp học phần + Table: TT, Mã sinh viên, Họ tên, Điểm tp1, Điểm tp2, Điểm tp3, Điểm thi, Điểm TB, Điểm chữ 3. Button + Quay lại + Tái thiết lập + Lưu
ND_21	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
ND_22	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
ND_23	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
ND_24	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn	Nhấn phím Enter	1. Nếu chuột không focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button

	Enter		
	Validate các trường thông tin **Chú ý:** *Khi validate thông tin một trường, tất cả các trường khác hợp lệ*		
	Trường Học kì, Dropdown list - màn hình Chọn học kì		
ND_25	Kiểm tra giá trị mặc định	1. Trên màn hình Chọn môn học 2. Kiểm tra giá trị mặc định của trường: Chọn học kì	Giá trị mặc định của học kì: Học kì 1, năm học 2019-2020
ND_26	Kiểm tra danh sách học kì	1. Focus chuột vào dropdown list Học kỳ	Hệ thống hiển thị danh sách các học kì trong năm
	Kiểm tra button, link		
ND_27	Kiểm tra button Lưu/màn hình Nhập điểm	Trường hợp chọn học kỳ Active Kiểm tra cho phép click button Lưu điểm	Cho phép click nút Lưu
ND_28	Kiểm tra button Lưu/Màn hình Nhập điểm	Trường hợp chọn học kỳ InActive Kiểm tra cho phép click button Lưu điểm	Không cho phép click nút Lưu
ND_29	Kiểm tra button Tái thiết lập/Màn hình nhập điểm	Trường hợp chọn học kỳ Active Kiểm tra cho phép click button Tái thiết lập	Cho phép click nút Tái thiết lập
ND_30	Kiểm tra button Tái thiết lập/Màn hình nhập điểm	Trường hợp chọn học kỳ InActive Kiểm tra cho phép click button Tái thiết lập	Không cho phép click nút Tái thiết lập
ND_31	Check hyperlink Mã môn học/Màn hình Chọn môn học	Kiểm tra cho phép click hyperlink Mã Môn Học	Hệ thống cho phép click hyperlink Mã Môn Học, hệ thống hiện danh sách các lớp học phần

ND_32	Check hyperlink Mã lớp/Màn hình Chọn lớp học phần	Kiểm tra cho phép click hyperlink Mã Lớp	Hệ thống cho phép click hyper Mã Lớp, hệ thống hiện giao diện nhập điểm cho sinh viên
	Trường Điểm thành phần, textbox - màn hình Nhập điểm		
ND_33	Kiểm tra giá trị mặc định	1. Trên màn hình Nhập điểm 2. Kiểm tra giá trị mặc định của các trường Điểm thành phần	Giá trị mặc định là trống
ND_34	Kiểm tra bỏ trống trường nhập	1. Trên màn hình Nhập điểm 2. Không nhập dữ liệu vào trường điểm thành phần 3. Click Button [Lưu]	1. Hiển thị thông báo bắt buộc nhập điểm thành phần 2. Set focus vào trường lỗi
ND_35	Kiểm tra khi thực hiện CTRL+V để paste nội dung ở nơi khác vào textbox	1.Thực hiện CTRL+V để paste nội dung ở nơi khác vào trường Điểm thành phần 2. Click button [Lưu]	Cho phép Copy/paste và thỏa mãn điều kiện như khi nhập dữ liệu
ND_36	Kiểm tra trim dấu space	1. Nhập điểm có space ở đầu và cuối 2. Các thông tin khác được nhập hợp lệ 3. Ấn button [Lưu]	1. Thực hiện thành công 2. Thực hiện trim space ở đầu và cuối trường dữ liệu
ND_37	Kiểm tra nhập giá trị không hợp lệ	1. Nhập điểm không hợp lệ (là chữ, số <0, >10) 2. Các thông tin khác được nhập hợp lệ 3. Ấn button [Lưu]	1. Hiển thị thông báo nhập sai, yêu cầu nhập lại 2. Set focus vào trường lỗi
ND_38	Kiểm tra nhập giá trị hợp lệ	1. Nhập điểm hợp lệ tại biên (0, 10) và trong biên (từ 0 đến 10) 2. Các thông tin khác được nhập hợp lệ	1. Thực hiện thành công

		3. Ấn button [Lưu]	
	Chức năng (Phần này viết các trường hợp kiểm thử cho các chức năng trong module Nhập điểm)		
	Màn hình: Chọn môn học		
ND_39	Kiểm tra hiển thị đúng thông tin - trường hợp chọn học kỳ Active	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active	Danh sách môn học tương ứng với CSDL: các môn học (tblMonHoc) trong học kì (tblMonHocKiHoc) có lớp học phần (tblLopHocPhan) mà giảng viên (tblGiangVien) có lịch dạy (tblLichHoc)
ND_40	Kiểm tra hiển thị không có thông tin - trường hợp chọn học kỳ Active, nhưng giáo viên không dạy môn nào	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active nhưng giáo viên đó không dạy môn nào	Hiển thị danh sách rỗng
ND_41	Kiểm tra hiển thị không có thông tin - trường hợp chọn học kỳ InActive	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ in active	Hiển thị thông báo "Học kì không hiệu lực, không thể nhập điểm!" và không cho chọn môn học.
	Màn hình: Chọn lớp học phần		

ND_42	Kiểm tra hiển thị đúng thông tin	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học	Danh sách lớp học phần tương ứng với CSDL: lớp học phần (tblLopHocPhan) mà giảng viên (tblGiangVien) có lịch dạy (tblLichHoc) của môn học (tblMonHoc)
	Màn hình: Nhập điểm		
ND_43	Kiểm tra hiển thị đúng thông tin - sinh viên chưa có điểm.	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, kiểm tra thông tin sinh viên hiển thị ra	Hệ thống hiển thị đúng tên giảng viên, tên môn học, mã lớp học phần vừa chọn. Hệ thống hiển thị đúng mã sinh viên, tên sinh viên (tblSinhVien, tblThanhVien, tblHoTen) của lớp học phần (tblLopHocPhan) mà sinh viên đã đăng ký học (tblSinhVienKhoa, tblDangkihoc); các trường điểm rỗng
ND_44	Kiểm tra hiển thị đúng thông tin - sinh viên đã có điểm.	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm,	Hệ thống hiển thị đúng điểm của từng sinh viên trong danh sách tương ứng với tblKetQua Cột điểm trung bình và điểm chữ được tính đúng.

		kiểm tra thông tin sinh viên hiển thị ra	
ND_45	Nhập điểm thành công - Chưa có điểm sinh viên - Nhập đầu điểm cho tất cả sinh viên	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, Giảng viên nhập điểm hợp lệ cho đầu điểm muốn nhập cho tất cả sinh viên trong danh sách 5. Click nút Lưu	Cột điểm trung bình và điểm chữ được tính đúng. Hệ thống lưu điểm và quay về giao diện chính của giảng viên. Điểm được lưu vào CSDL tblKetQua tương ứng với từng sinh viên

ND_46	Nhập điểm thành công - Đã có điểm sinh viên - Nhập đầu điểm cho tất cả sinh viên	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, Giảng viên nhập điểm hợp lệ cho đầu điểm muốn nhập cho tất cả sinh viên trong danh sách 5. Click nút Lưu	Cột điểm trung bình và điểm chữ được cập nhật đúng. Hệ thống lưu điểm và quay về giao diện chính của giảng viên Điểm được lưu vào CSDL tblKetQua tương ứng với từng sinh viên
ND_47	Nhập điểm thành công - Chưa có điểm sinh viên - Nhập đầu điểm cho tất cả sinh viên sao cho có đủ loại điểm chữ A, B, C, D, E, F	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, Giảng viên nhập điểm hợp lệ cho đầu điểm muốn nhập cho tất cả sinh viên trong danh sách sao cho có đủ loại điểm chữ A, B, C, D, E, F tại biên và trong biên	Cột điểm trung bình và điểm chữ được cập nhật đúng. Hệ thống lưu điểm và quay về giao diện chính của giảng viên Điểm được lưu vào CSDL tblKetQua tương ứng với từng sinh viên

		5. Click nút Lưu	
ND_48	Nhập điểm thành công - Đã có điểm sinh viên - Nhập đầu điểm cho một số sinh viên	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, Giảng viên nhập điểm hợp lệ cho đầu điểm muốn nhập cho một số sinh viên trong danh sách 5. Click nút Lưu	Cột điểm trung bình và điểm chữ được cập nhật đúng. Hệ thống lưu điểm mới và quay về giao diện chính của giảng viên Điểm được lưu vào CSDL tblKetQua tương ứng với các sinh viên đã cập nhật

ND_49	Nhập điểm không thành công - Chưa có điểm sinh viên - Nhập đầu điểm cho một số sinh viên	1. Từ màn hình Trang chủ giảng viên, click button [Nhập điểm] 2. Tại màn hình Chọn môn học, nhấn vào Dropdownlist [Chọn học kỳ], chọn học kỳ active 3. Chọn 1 môn học 3. Tại màn hình Lớp học phần hiển thị ra, chọn lớp học phần muốn nhập điểm 4. Tại màn hình nhập điểm, Giảng viên nhập điểm KHÔNG HỢP LỆ cho đầu điểm muốn nhập cho một số sinh viên trong danh sách 5. Click nút Lưu	Afer step 1: hệ thống hiển thị danh sách các môn học do giảng viên dạy của kì đã chọn (Mã, Tên, Số tín chỉ, Mô tả) After step 2: Hệ thống hiển thị danh sách các lớp học phần do giảng viên dạy (Mã, tên, sĩ số thực, phòng học , ngày học, kíp học) After step 3: Hệ thống hiển thị danh sách các sinh viên đăng ký lớp học phần: Thứ tự, Mã sinh viên, Họ tên, Các đầu điểm thàn phần, điểm thi, cột trung bình môn, điểm chữ được tự tính sau khi nhập After step 5: Hệ thống không lưu điểm của sinh viên
ND_50	Nhập điểm thành công - Đã có điểm sinh viên - Nhập đầu điểm cho một số sinh viên	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhậpđiểm hợp lệ cho đầu điểm muốn nhập cho tất cả sinh viên trong danh sách 5. Không click nút Lưu điểm	Afer step 1: hệ thống hiển thị danh sách các môn học do giảng viên dạy của kì đã chọn (Mã, Tên, Số tín chỉ, Mô tả) After step 2: Hệ thống hiển thị danh sách các lớp học phần do giảng viên dạy (Mã, tên, sĩ số thực, phòng học , ngày học, kíp học) After step 3: Hệ thống hiển thị danh sách các sinh viên đăng ký lớp học phần, và điểm nếu code (ở chế độ Xem) : Thứ tự, Mã sinh viên, Họ tên, Các đầu điểm thàn phần, điểm thi, cột trung bình môn, điểm chữ được tự tính sau khi nhập After step 5: Hệ thống không lưu điểm cho sinh viên

ND_51	Nhập điểm thành công - Chưa có điểm sinh viên - Nhập đầu điểm cho một số sinh viên	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhậpđiểm hợp lệ cho đầu điểm muốn nhập cho một số sinh viên trong danh sách 5. Click nút Lưu 6 Click Nút Hủy	Afer step 1: hệ thống hiển thị danh sách các môn học do giảng viên dạy của kì đã chọn (Mã, Tên, Số tín chỉ, Mô tả) After step 2: Hệ thống hiển thị danh sách các lớp học phần do giảng viên dạy (Mã, tên, sĩ số thực, phòng học , ngày học, kíp học) After step 3: Hệ thống hiển thị danh sách các sinh viên đăng ký lớp học phần: Thứ tự, Mã sinh viên, Họ tên, Các đầu điểm thàn phần, điểm thi, cột trung bình môn, điểm chữ được tự tính sau khi nhập After step 5: Hệ thống hiển thị thông báo: Bạn chưa nhập điểm cho tất cả sinh viên After step 6: Hệ thống đóng popup message, và ở lại trang nhập điểm
ND_52	Nhập điểm thành công - Đã có điểm sinh viên - Nhập đầu điểm cho một số sinh viên	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhậpđiểm hợp lệ cho đầu điểm muốn nhập cho một số sinh viên trong danh sách 5. Click nút Lưu 6 Click nút Hủy	Afer step 1: hệ thống hiển thị danh sách các môn học do giảng viên dạy của kì đã chọn (Mã, Tên, Số tín chỉ, Mô tả) After step 2: Hệ thống hiển thị danh sách các lớp học phần do giảng viên dạy (Mã, tên, sĩ số thực, phòng học , ngày học, kíp học) After step 3: Hệ thống hiển thị danh sách các sinh viên đăng ký lớp học phần, và điểm nếu code (ở chế độ Xem) : Thứ tự, Mã sinh viên, Họ tên, Các đầu điểm thàn phần, điểm thi, cột trung bình

			môn, điểm chữ được tự tính sau khi nhập After step 5: Hệ thống hiển thị thông báo: Bạn chưa nhập điểm cho tất cả sinh viên After step 6: Hệ thống không chưa lưu điểm của sinh viên và vẫn ở trang nhập điểm
ND_53	Nhập điểm không hợp lệ	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhập điểm có kí tự đặc biệt (khác kí tự .)	Hệ thống tự động chuyển kí tự đặc biệt thành trống
ND_54	Nhập điểm không hợp lệ	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhập điểm >10	Hệ thống tự động chuyển thành trống
ND_52	Nhập điểm không hợp lệ	1. Tại màn hình nhập điểm, giảng viên chọn học kì Active 2. Giảng viên tìm kiếm môn học và chọn môn học 3. Giảng viên chọn lớp học phần muốn nhập điểm 4. Giảng viên nhập điểm có	Hệ thống tự động làm tròn theo quy tắc lên xuống theo format A.C (C có 1 chữ số)

		dạng A.B(B> 1 chữ số)	

1

Module Thống kê theo loại học lực 2

Bảng 6.9: Hướng dẫn xây dựng Test case chức năng cho module thống kê theo học lực 3

4

Mã trường hợp kiểm thử	**Mục đích kiểm thử**	**Các bước thực hiện**	**Kết quả mong muốn**
	Chức năng 1.3	**Quản lý xem thống kê theo học lực**	
	Precond: Quản lý đăng nhập thành công Chọn Xem thống kê -> Hiển thị màn hình Thống kê-> Chọn thống kê Loại học lực		
	Giao diện (Phần này viết các trường hợp kiểm thử cho giao diện chung và các giao diện cho các control)		
	Giao diện chung		
	Giao diện Trang chủ Quản lý		
TK_1	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng

TK_2	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Mã NV: textview + Tên NV: Textview 4. Button + Xem thống kê
TK_3	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
TK_4	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
TK_5	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
TK_6	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Chọn thống kê		
TK_7	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và

			dễ sử dụng
TK_8	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng 2. Focus được set vào trường đầu tiên có thể edit 3. Hiển thị đầy đủ các trường : + Chọn Thống kê: Dropdown list: TK Loại học lực, TK sinh viên, TK môn học + Chọn học kì: Dropdownlist 4. Button + Xem
TK_9	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
TK_10	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
TK_11	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
TK_12	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Thống kê loại học lực		

TK_13	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
TK_14	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng 2. Focus được set vào trường đầu tiên có thể edit: 3. Hiển thị đầy đủ các trường : + Học kì: Textview + Table: TT, Loại HL, Số lượng SV đạt, Điểm TB sinh viên đạt 4. Button + Quay lại 5. Hyperlink + Loại HL
TK_15	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
TK_16	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
TK_17	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.

TK_18	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Thống kê sinh viên		
TK_19	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
TK_20	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng 2. Hiển thị đầy đủ các trường : + Loại học lực: Textview + Học kì: Textview + Table: TT, Mã sv, Tên sv, Ngành, Tổng tc, Điểm TB 3. Hyperlink + Tên SV 4. Button + Quay lại
TK_21	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
TK_22	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.

TK_23	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
TK_24	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Điểm sinh viên		
TK_25	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng
TK_26	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Điểm sinh viên 2. Hiển thị đầy đủ các trường : + Tên Sinh viên: Textview + Học kì: Textview + Table: TT, Mã MH, Tên MH, Số tc, Điểm TB ++dòng cuối của table là : Tổng số tc, điểm tb Tổng 3. Hyperlink + Mã MH 4.Button + Quay lại
TK_27	Kiểm tra giao diện khi thu	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao

	nhỏ, phóng to		diện
TK_28	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
TK_29	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
TK_30	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Giao diện Điểm môn học của sinh viên		
TK_31	Kiểm tra tổng thể giao diện màn hình	Kiểm tra về bố cục, font chữ, chính tả, màu chữ	Các label, textbox cùng font chữ cỡ chữ, căn lề trái, có độ dài, rộng và khoảng cách bằng nhau, không xô lệch. - Không có lỗi về chính tả, cấu trúc câu, ngữ pháp trên màn hình - Form được bố trí hợp lý và dễ sử dụng

TK_32	Kiểm tra bố cục giao diện	1. Kiểm tra title của màn hình 2. Kiểm tra focus của chuột 3. Kiểm tra hiển thị thông tin các trường và button trên màn hình	1. Hiển thị title của chức năng: Điểm môn học của sinh viên 2. Hiển thị đầy đủ các trường : + Học kì: Textview + Tên Sinh viên: Textview + Môn học: Textview + Table: TT, Đầu điểm ,Trọng số, Kết quả, 3 dòng cuối là TB Môn, Điểm chữ, Điểm hệ 4 3. Button + Print + Xuất file Excel
TK_33	Kiểm tra giao diện khi thu nhỏ, phóng to	1. Nhấn phím Ctrl - 2. Nhấn phim Ctrl +	Màn hình thu nhỏ, phóng to tương ứng và không bị vỡ giao diện
TK_34	Kiểm tra thứ tự di chuyển trỏ trên màn hình khi nhấn phím Tab	Forcus vào màn hình. Nhấn Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: Từ phải qua trái, từ trên xuống dưới.
TK_35	Kiểm tra thứ tự con trỏ di chuyển ngược lại trên màn hình khi nhấn Shift-Tab	Forcus vào màn hình. Nhấn phím Shift-Tab liên tục	Con trỏ di chuyển lần lượt theo thứ tự: từ dưới lên trên, từ phải qua trái.
TK_36	Kiểm tra thực hiện chức năng chính của màn hình khi nhấn Enter	Nhấn phím Enter	1. Nếu chuộc ko focus vào button nào thì Thực hiện chức năng của button chính 2. Nếu đang focus vào 1 button thì sẽ thực hiện chức năng của button
	Validate các trường thông tin **Chú ý:** *Khi validate thông tin một trường, tất cả các trường khác hợp lệ*		
	Trường Chọn thống kê, Dropdown list		
TK_37	Kiểm tra giá trị mặc định	1. Trên màn hình Xem thống kê 2. Kiểm tra giá trị mặc định	Giá trị mặc định của Chọn thống kê: TK loại học lực

		của trường: Chọn thống kê	
TK_38	Kiểm tra danh sách các loại thống kê	1. Focus chuột vào dropdown list Chọn thống kê	Hệ thống hiển thị danh sách các loại Thống kê
	Trường Chọn Học kì, Dropdownlist		
TK_39	Kiểm tra giá trị mặc định	1. Trên màn hình Xem thống kê 2. Kiểm tra giá trị mặc định của trường: Chọn thống kê	Giá trị mặc định của Chọn thống kê: TK loại học lực
TK_40	Kiểm tra danh sách các loại thống kê	1. Focus chuột vào dropdown list Chọn thống kê	Hệ thống hiển thị danh sách các loại Thống kê
	Kiểm tra button, link		
TK_41	Kiểm tra button Tìm kiếm	Kiểm tra cho phép click button Tìm kiếm	Luôn cho phép click button tìm kiếm
TK_42	Check hyperlink Loại Học lực	Kiểm tra cho phép click hyperlink Loại học lực	Hệ thống cho phép click hyper Loại học lực trên dòng, hệ thống hiện danh sách các lsinh viên theo học lực đã chọn
TK_43	Check hyperlink Mã sinh viên	Kiểm tra cho phép click hyperlink Mã sinh viên	Hệ thống cho phép click hyper Mã sinh viên trên dòng, hệ thống hiện danh sách điểm sinh viên
TK_44	Check hyperlink Mã môn học	Kiểm tra cho phép click hyperlink Mã môn học	Hệ thống cho phép click hyper Mã môn học, hệ thống hiện Điểm môn học của sinh viên
	Chức năng (Phần này viết các trường hợp kiểm thử cho các chức năng trong module Thống kê)		
	Chứng năng: Thống kê theo loại học lực		
	Chứng năng: Thống kê loại học lực		
TK_45	Nhập chính xác học kì - Đã có điểm trung bình của học kì	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì vào trường: Học kì 2. Click nút Tìm kiếm	Hệ thống hiển thị danh sách Thống kê loại học lực theo học kì đã chọn, mỗi loại học lực được xếp trên một dòng, xếp theo thứ tự cao nhất đến thấp nhất của loại học lực trong

			bảng đánh giá, Ưu tú, Xuất sắc, Giỏi, Khá, Trung bình, Yếu kém: TT, Loại học lực, tổng số sinh viên đạt loại đó, điểm trung bình sinh viên đạt
TK_46	Nhập chính xác học kì - Học kì đang học	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì vào trường: Học kì 2. Click nút Tìm kiếm	Hệ thống hiển thị danh sách trống
TK_47	Nhập không chính xác tên học kì	1. Tại màn hình thống kê loại học lực, nhập không chính xác tên học kì vào trường: Học kì 2. Click nút Tìm kiếm	Hệ thống hiển thị danh sách trống
TK_48	Click bất kì một loại học lực	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì đã có dữ liệu học lực vào trường: Học kì 2. Click bất kì hyperlink học lực mong muốn	Hệ thống chuyển hướng hiển thị giao diện: Thống kê sinh viên
	Chứng năng: Thống kê sinh viên		
TK_49	Thống kê sinh viên - Có sinh viên theo học lực đã chọn	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì đã có dữ liệu học lực vào trường: Học kì 2. Click bất kì hyperlink học lực mong muốn có số lượng sinh viên > 0	Sau bước 2: Hệ thống hiển thị danh sách các sinh viên đạt loại học lực đã chọn, xếp theo thứ tự các ngành học, đến thứ tự abc của sinh viên: Thứ tự, mã sinh viên, họ và tên, ngành học, khóa học, tổng số tín chỉ của học kì, điểm trung bình của học kì
TK_50	Thống kê sinh viên - Không có sinh viên theo học lực đã chọn	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì đã có dữ liệu học lực vào trường: Học kì 2. Click bất kì hyperlink học lực mong muốn có số lượng sinh viên = 0	Sau bước 2: Hệ thống hiển thị danh sách trống các sinh viên đạt loại học lực đã chọn dưới danh sách lưới: Thứ tự, mã sinh viên, họ và tên, ngành học, khóa học, tổng số tín chỉ của học kì, điểm trung bình

1

			của học kì
TK_51	Click bất kì một tên sinh viên	Tại màn hình thống kê sinh viên, click bất kì một tên sinh viên	Hệ thống chuyển hướng đến giao diên điểm sinh viên
	Chứng năng: Xem điểm sinh viên		
TK_52	Thống kê điểm sinh viên	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì đã có dữ liệu học lực vào trường: Học kì 2. Click bất kì hyperlink học lực mong muốn có số lượng sinh viên > 0 3. Tại màn hình Thống kê sinh viên, click bất kì hyperlink Tên sinh viên trên dòng.	Sau bước 3: Hệ thống hiển thị lên danh sách các môn và kết quả của sinh viên đã học trong học kì đó, xếp theo thứ tự abc của tên môn học: Thứ tự, Tên môn học, Số tín chỉ, điểm trung bình môn đó của sinh viên. Dòng cuối cùng là Tổng số tín chỉ, điểm trung bình của cả học kì
TK_53	Click bất kì một Mã môn học	Tại màn hình Điểm sinh viên, Click bất kì một mã môn học	Hệ thống chuyển hướng đến giao diêện điểm sinh viên
	Chứng năng: Xem điểm môn học của sinh viên		
TK_54	Thống kê điểm sinh viên	1. Tại màn hình thống kê loại học lực, nhập chính xác tên học kì đã có dữ liệu học lực vào trường: Học kì 2. Click bất kì hyperlink học lực mong muốn có số lượng sinh viên > 0 3. Tại màn hình Thống kê sinh viên, click bất kì hyperlink Tên sinh viên trên dòng. 4. Tại màn hình điểm sinh	Sau bước 4: Hệ thống hiện lên chi tiết điểm môn học của sinh viên: - Mã môn - Tên môn học - Tổng tín chỉ - Các đầu điểm thành phần dạng bảng: +TT: Thứ tự +Đầu điểm: Tên đầu điểm thành phần + Trọng số: Tỉ lệ % của đầu điểm thành phần

		viên, click bất kì một tên môn học	+ Kết quả: Điểm của sinh viên

1

Yêu cầu xây dựng test data cho chức năng thông kê: 2

Do chức năng thống kê cần có dữ liệu đầy đủ trong các bảng thì mới thống kê được, do vậy việc xây dựng test data cho chức năng thống kê là quan trọng và cần thiết để phủ hết các tình huống xảy ra. Ví dụ về yêu cầu xây dựng test data cho chức năng thống kê: 3

STT	Mục	Yêu cầu dữ liệu
1	Loại sinh viên	Đủ các loại: - Ưu tú, - Xuất sắc, - Giỏi, - Khá, - Trung bình, - Yếu kém
2	Số lượng học kỳ	>=2 học kỳ có điểm
3	Số lượng sinh viên thuộc mỗi loại của mỗi học kỳ là 0	= 0
4	Số lượng sinh viên thuộc mỗi loại của mỗi học kỳ là ít	Trong khoảng 1-10
5	Số lượng sinh viên thuộc mỗi loại của mỗi học kỳ là trung bình	>100
6	Số lượng sinh viên thuộc mỗi loại của mỗi học kỳ là nhiều	>1000

4

6.3 CÂU HỎI ÔN TẬP 5

1. Thực hiện các hoạt động rà soát cho module 6
2. Thực hiện test chức năng cho module

BÀI TẬP DỰ ÁN [1]

HỆ THỐNG QUẢN LÍ [2]