

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO THÍ NGHIỆM/THỰC NGHIỆM
LẬP TRÌNH JAVA

XÂY DỰNG CHƯƠNG TRÌNH QUẢN LÝ
ĐỘI THANH NIÊN TÌNH NGUYỆN KHOA CNTT

GVHD: *ThS. Vũ Thị Dương*

Sinh viên:

Mai Thị Hường

Diệp Thị Linh

Phùng Tuấn Minh

Lê Ngọc Trường

Trần Đình Tuấn

Nhóm: **7**

Lớp: **20223IT6019001. Khóa: 15**

Hà Nội – Năm 2023

MỤC LỤC

PHẦN 1 MỞ ĐẦU	3
1.1. Mô tả tổng quát chủ đề nghiên cứu	3
1.1.1. Tóm lược ý tưởng	3
1.1.2. Chủ đề nghiên cứu	3
1.1.3. Xác định nội dung học tập	3
1.1.4. Các kỹ năng	3
1.2. Mục tiêu và tiêu chuẩn đầu ra của học phần.....	4
PHẦN 2 KẾT QUẢ NGHIÊN CỨU	4
2.1. Giới thiệu	4
2.2. Khảo sát hệ thống	6
2.2.1. Khảo sát sơ bộ.....	6
2.2.2. Tài liệu đặc tả yêu cầu	11
2.3. Phân tích hệ thống.....	13
2.3.1. Mô hình hóa chức năng hệ thống	13
2.3.2. Mô hình hóa dữ liệu và giao diện hệ thống	20
2.4. Thực hiện bài toán	26
2.4.1. Mai Thị Hương – Thống kê.....	26
2.4.2. Diệp Thị Linh – Quản lý sự kiện	28
2.4.3. Phùng Tuấn Minh – Đăng nhập, đăng ký, lên lịch phỏng vấn	33
2.4.4. Lê Ngọc Trường – Quản lý thành viên, quản lý thông tin cá nhân, đăng xuất, thêm thành viên tham gia sự kiện.....	36
2.4.5. Trần Đình Tuấn – Quản lý chi phí hoạt động.....	43
PHẦN 3 KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM	47

3.1. Nội dung đã thực hiện.....	47
3.2. Hướng phát triển	47
TÀI LIỆU THAM KHẢO	49

PHẦN 1 MỞ ĐẦU

1.1. Mô tả tổng quát chủ đề nghiên cứu

1.1.1. Tóm lược ý tưởng

- Đây là chương trình hỗ trợ khoa, nhà trường trong việc quản lý đội thanh niên tình nguyện.
- Chương trình giúp cho khoa, đoàn trường dễ dàng lên lịch phỏng vấn, quản lý sinh viên tình nguyện trong các sự kiện và báo cáo thống kê một cách rõ ràng, rành mạch.

1.1.2. Chủ đề nghiên cứu

- Chủ đề nghiên cứu: Xây dựng chương trình quản lý đội thanh niên tình nguyện khoa Công Nghệ Thông Tin.

1.1.3. Xác định nội dung học tập

- Để hoàn thiện được chủ đề nghiên cứu, chúng em cần nắm chắc các nội dung sau:
 - Cấu trúc của một chương trình Java Core.
 - Các toán tử, câu lệnh trong Java
 - Các tính chất hướng đối tượng trong Java.
 - Xử lý ngoại lệ, I/O theo luồng và thao tác với tệp.
 - Giao diện GUI Java Swing.

1.1.4. Các kỹ năng

1.1.4.1. Các kiến thức, kỹ năng đã có để thực hiện nghiên cứu

- Lập trình Java cơ bản.
- Lập trình Java hướng đối tượng.
- Xử lý ngoại lệ và thao tác với tệp.
- Lập trình giao diện Java.

1.1.4.2. Kỹ năng then chốt trong chủ đề nghiên cứu.

- Kỹ năng làm việc nhóm.
- Kỹ năng xây dựng ý tưởng.

- Kỹ năng thu thập thông tin và chọn lọc thông tin.

1.2. Mục tiêu và tiêu chuẩn đầu ra của học phần

- Hoàn thiện Demo sản phẩm chương trình quản lý đội thanh niên tình nguyện khoa Công Nghệ Thông Tin.
- Trang bị thêm kiến thức còn thiếu.
- Nắm chắc được các kiến thức đã học được và sử dụng nhuần nhuyễn và xử lý bài toán theo hướng tối ưu nhất.

PHẦN 2 KẾT QUẢ NGHIÊN CỨU

2.1. Giới thiệu

Kết quả nghiên cứu của nhóm em là một chương trình “Quản lý đội thanh niên tình nguyện khoa Công Nghệ Thông Tin” được sử dụng trên một số hệ điều hành như Mac, Windows ... Chương trình này được phát triển nhằm giúp đỡ đoàn trường, khoa quản lý đội và các sự kiện của đội một cách dễ dàng nhất.

Trước khi bắt tay vào phát triển sản phẩm thì nhóm em đã cùng nhau nghiên cứu và chọn lọc được mô hình phát triển sản phẩm phù hợp với bài toán được đưa ra. Nhóm chúng em quyết định sử dụng mô hình thác nước (Waterfall) để xác định được các quy trình trong xây dựng sản phẩm. Vì mô hình này dễ quản lý, dễ tiếp cận, sử dụng và nó rất phù hợp cho các sản phẩm vừa và nhỏ.

Các bước tổng quan:

- Thu thập và phân tích yêu cầu (Requirement Analysis): Tất cả các yêu cầu có thể có của hệ thống được phát triển đều được ghi lại trong giai đoạn này và được ghi lại trong tài liệu đặc tả yêu cầu để phục vụ cho các giai đoạn sau.

- Thiết kế hệ thống (System Design): Thiết kế hệ thống giúp xác định các yêu cầu phần cứng và hệ thống cũng như giúp xác định kiến trúc hệ thống tổng thể.
 - Thực hiện (Implementation): Với đầu vào từ thiết kế hệ thống, tiến hành cài đặt chương trình.
 - Tích hợp và Kiểm thử (Integration and Testing).
 - Triển khai hệ thống (Deployment of system).
 - Bảo trì (Maintenance).
- Kết quả đạt được: Sản phẩm sau khi hoàn thành là một chương trình quản lý đội thanh niên tình nguyện. Chương trình có giao diện và 1 số chức năng cơ bản để quản lý đoàn đội của trường, cụ thể ở đây là trường Đại học Công nghiệp Hà Nội. Chương trình được cài đặt và triển khai dựa trên Netbeans do Netbeans là 1 công cụ phát triển tích hợp cung cấp các tính năng như syntax highlighting, auto-completion, debugging, và phân tích mã nguồn giúp lập trình viên phát triển ứng dụng một cách dễ dàng và hiệu quả hơn.
 - Mô tả sản phẩm:
 - Tên sản phẩm: Chương trình quản lý đội thanh niên tình nguyện khoa Công nghệ thông tin.
 - Hình thức sản phẩm: Phần mềm ứng dụng chạy trên desktop
 - Cấu trúc: Ngôn ngữ Java, đọc ghi File
 - Nội dung sản phẩm gồm các chức năng:
 - Đăng kí để phỏng vấn.
 - Đăng nhập.
 - Sinh viên (Thành viên trong đội thanh niên tình nguyện) có thể xem, sửa, xóa thông tin của mình.
 - Quản lý sinh viên: xem, thêm, sửa, xóa sinh viên.
 - Lên lịch sự kiện (Thêm, Sửa, Xóa sự kiện).
 - Chọn sinh viên đi tình nguyện và tham gia các sự kiện.
 - Thêm, sửa các khoản phí hoạt động.

- Lên lịch phỏng vấn cho sinh viên có nguyện vọng đăng kí và muốn trở thành thành viên của đội thanh niên tình nguyện.
- Chỉnh sửa ngân sách.
- Thống kê các khoản chi trong các sự kiện của đội.

2.2. Khảo sát hệ thống

2.2.1. Khảo sát sơ bộ

2.2.1.1. Đối tượng khảo sát

- Đối tượng sử dụng phần mềm: Đoàn trường, Khoa Công nghệ Thông tin, Thành viên của đoàn, hoặc sinh viên có mong muốn vào đoàn.

2.2.1.2. Phương pháp phỏng vấn

- Kế hoạch phỏng vấn

Kế hoạch phỏng vấn	
Người được hỏi: Nguyễn Văn Dũng	Người phỏng vấn: Diệp Thị Linh, Lê Ngọc Trường
Địa chỉ: Đại học Công nghiệp Hà Nội (298 Cầu Diễn, Bắc Từ Liêm, Hà Nội)	Thời gian hẹn: 9h ngày 27/2/2023 Thời điểm bắt đầu: 9h15 phút ngày 27/2/2023 Thời điểm kết thúc: 10h ngày 27/2/2023
Đối tượng: Là thành viên của đội thanh niên tình nguyện khoa Công nghệ Thông tin trường đại học Công nghiệp Hà Nội	Các yêu cầu đòi hỏi: Vai trò: Là thành viên của đội thanh niên tình nguyện Trình độ: đang tham gia học đại học tại trường Công nghiệp Hà Nội

	Vị trí: thành viên đội
<p>Chương trình:</p> <ul style="list-style-type: none"> - Giới thiệu: Buổi phỏng vấn nhằm thu thập thông tin chi tiết để xây dựng hệ thống quản lý đội thanh niên tình nguyện khoa CNTT. - Tổng quan về dự án: Thu thập và phân tích các thông tin cần để thiết kế xây dựng hệ thống quản lý đội thanh niên. - Tổng quan về buổi phỏng vấn: Nội dung câu hỏi phỏng vấn sẽ liên quan đến các chi tiết và đặc điểm của đội TNTT cũng như các yêu cầu đặt ra. <p>Xin phép được ghi âm cuộc phỏng vấn:</p> <ul style="list-style-type: none"> - Chủ đề 1: Câu hỏi và trả lời - Chủ đề 2: Câu hỏi và trả lời <p>Tập hợp các nội dung chính</p> <p>Ý kiến, quan điểm của người được hỏi</p> <p>Kết thúc</p>	<p>Ước lượng thời gian phỏng vấn:</p> <p>2 phút</p> <p>5 phút</p> <p>10 phút</p> <p>10 phút</p> <p>10 phút</p> <p>10 phút</p>
	Dự kiến: tổng 47 phút

➤ Phiếu phỏng vấn

Phiếu phỏng vấn	
Tên dự án: Xây dựng hệ thống quản lý đội CNTT khoa Công nghệ Thông tin trường Đại học Công nghiệp Hà Nội	
Người được hỏi: Nguyễn Văn Dũng	Ngày 28/2/2023 Người hỏi: Diệp Thị Linh, Lê Ngọc Trường
Câu hỏi	Ghi chú
Câu 1: Đội CNTT thuộc bộ phận nào trong khoa, trường?	Liên chi hội sinh viên khoa Công nghệ thông tin
Câu 2: Những người sẽ thường sử dụng chương trình quản lý là những ai?	Là tất cả thành viên trong đội CNTT
Câu 3: Anh muốn biết rõ ràng hơn về vấn đề tài chính của đội không?	Có, tôi muốn biết các khoản phí thu vào và chi ra sau mỗi sự kiện của đội. Dễ hiểu và dễ sử dụng
Câu 4: Anh muốn giao diện như nào?	
Câu 5: Anh có muốn thống kê các sự kiện của đội không?	Có, tôi muốn biết hết các sự kiện của đội trong năm
Câu 6: Anh có muốn những sinh viên chưa là thành viên trong đội được sử dụng chương trình không?	Có, tôi muốn những sinh viên chưa là thành viên đội đăng kí để chúng tôi có thể nắm bắt được thông tin các bạn đó và hẹn đi phỏng vấn.

Câu 7: Anh mong muốn điều gì để giúp cho các bạn quản lý dễ dàng hơn trong việc tuyển thành viên không?	Tôi muốn người quản lý có thể dễ dàng hẹn các bạn đi phỏng vấn mà không cần nhắn tin liên hệ riêng từng bạn.
Câu 8: Anh có muốn xem được các thông tin mà mình đã đăng kí hay không	Có, tôi muốn xem các thông tin của mình
<p>Đánh giá chung:</p> <ul style="list-style-type: none"> - Người được hỏi có thẩm quyền trong đội TNTT. - Đã kết luận được một vài vấn đề, tuy nhiên có những vấn đề cần về thảo luận, bàn bạc lại. 	

2.2.1.3. Phương pháp lập phiếu điều tra

➤ Phiếu điều tra

**Phiếu điều tra về việc quản lý đội thanh niên tình nguyện khoa
Công nghệ Thông tin trường đại học Công Nghiệp Hà Nội**

Bạn hãy khoanh tròn vào mục lựa chọn, hoặc bỏ phiếu vào hòm thư.

Câu 1: Bạn giữ vai trò gì trong đội thanh niên tình nguyện?

- A. Đội trưởng, đội phó
- B. Thành viên

Câu 2: Bạn có gặp khó khăn gì trong quá trình tham gia đội TNTT không?

- A. Có
- B. Không

Câu 3: Nếu bạn đã gặp khó khăn thì bạn gặp những khó khăn nào dưới đây?

- A. Không biết lịch phỏng vấn khi đăng kí tuyển thành viên.
- B. Không rõ ràng các khoản thu chi của đội.
- C. Không biết rõ đội có những sự kiện, hoạt động gì.

Câu 4: Bạn đã cảm thấy quản lý đội như thế đã hợp lý chưa?

- A. Hợp lý
- B. Chưa hợp lý

Câu 5: Bạn thấy việc sử dụng chương trình, ứng dụng để đăng kí phỏng vấn có thuận tiện hơn không?

- A. Có, thuận tiện hơn.
- B. Không, rắc rối hơn.

Câu 6: Bạn có muốn sử dụng chương trình, ứng dụng để quản lý đội thanh niên tình nguyện không?

Câu 7: Bạn có góp ý gì cho chúng tôi về ứng dụng này không ?

Chúc bạn có một ngày tốt lành.

2.2.2. Tài liệu đặc tả yêu cầu

➤ Mô tả hệ thống:

- Hệ thống quản lý đội thanh niên tình nguyện khoa Công nghệ thông tin trường Đại học Công nghiệp Hà Nội giúp cho đoàn trường, khoa có thể theo dõi, quản lý quá trình hoạt động của đội.

- Hệ thống sẽ thông báo lịch đi phỏng vấn cho những sinh viên đăng kí có nguyện vọng tham gia đội.
- Hệ thống sẽ lên lịch sự kiện và admin sẽ chọn những sinh viên tham gia và hoạt động trong sự kiện đó.
- Hệ thống sẽ xuất ra các thống kê chi phí sau các sự kiện của đội.
- Hệ thống gồm 3 quyền chính là: admin (đoàn trường, khoa) và sinh viên chưa là thành viên của đội thanh niên tình nguyện, sinh viên đã là thành viên của đội thanh niên tình nguyện. Trong đó: admin có thể quản lý các chức năng của chương trình, sinh viên chưa là thành viên của đội có thể đăng kí và được nhận lịch phỏng vấn, sinh viên đã là thành viên của đội có thể được chọn đi tình nguyện ...
- Chức năng hệ thống phân ra làm 3 quyền: admin, thành viên đội, sinh viên chưa là thành viên đội.
 - Admin:
 - Đăng nhập.
 - Quản lý sinh viên (thành viên đội và chưa là thành viên đội nói chung) (gồm có thêm thành viên, sửa thông tin thành viên, xóa thành viên).
 - Quản lý sự kiện (gồm có thêm sự kiện, sửa thông tin sự kiện).
 - Chọn sinh viên đi sự kiện, quản lý phí hoạt động (gồm có thêm phí hoạt động, sửa phí hoạt động).
 - Lên lịch phỏng vấn những thành viên chưa là chính thức.
 - Thống kê toàn đội (như số lượng thành viên, số thành viên chính thức, chưa chính thức, số sự kiện đội hoạt động, tổng chi phí...).
 - Đăng xuất.
 - Thành viên đội:

- Đăng nhập.
- Đăng kí.
- Được chọn trạng thái bản thân (rảnh hoặc bận, là cơ sở để admin có thể chọn đi sự kiện).
- Chỉnh sửa thông tin của bản thân (ngoại trừ mã sinh viên).
- Đăng xuất.
- Sinh viên chưa là thành viên đội:
 - Đăng nhập.
 - Đăng kí.
 - Sửa thông tin của bản thân.
 - Được chọn trạng thái bản thân (rảnh hoặc bận, là cơ sở để admin có thể chọn đi phỏng vấn).
 - Đăng xuất.

2.3. Phân tích hệ thống

2.3.1. Mô hình hóa chức năng hệ thống

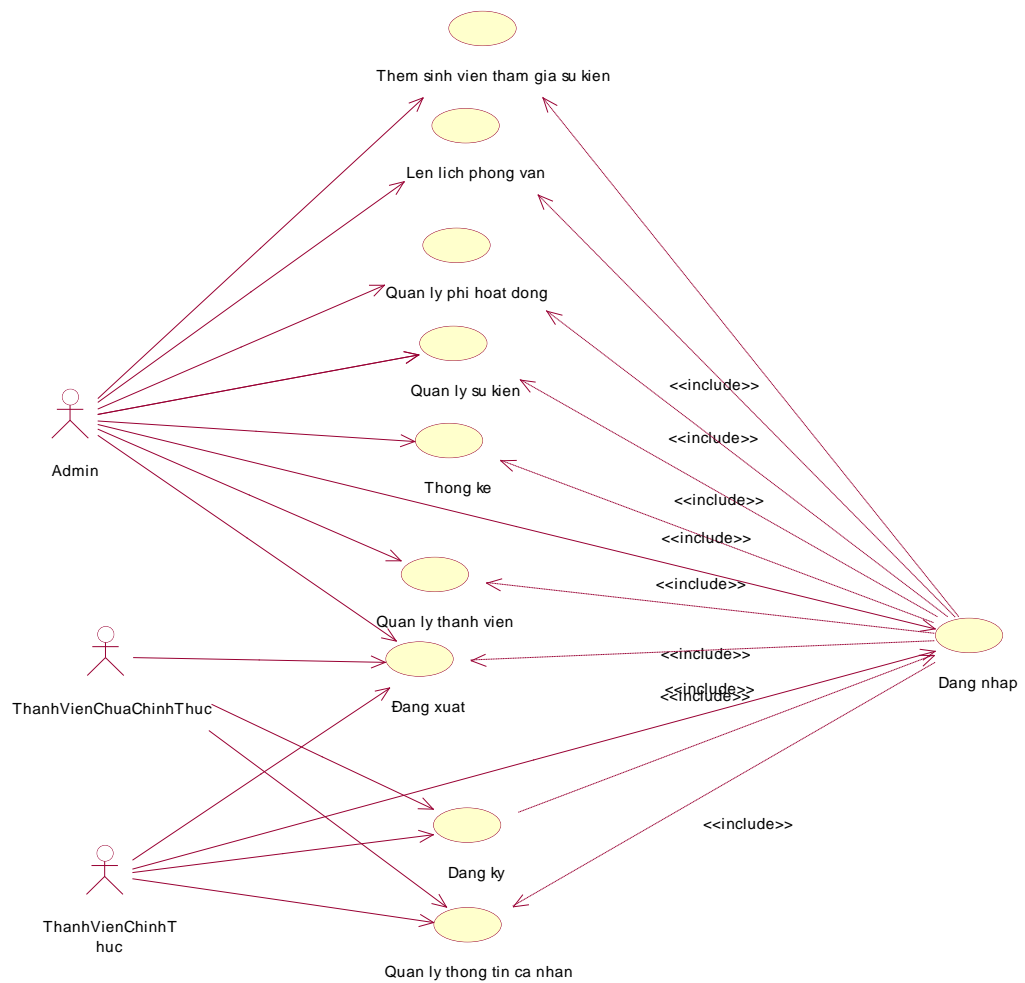
➤ Các actor:

- Quản trị viên (Admin): quản trị viên thực hiện nhiệm vụ quản lý thành viên, quản lý sự kiện, quản lý chi phí hoạt động, lên lịch phỏng vấn, thống kê toàn đội.
- Thành viên (chính thức và chưa chính thức): thực hiện nhiệm vụ đăng ký, đăng nhập, quản lý thông tin cá nhân và trạng thái hoạt động của bản thân.

➤ Các usecase:

- Đăng nhập: cho phép người dùng đăng nhập vào hệ thống với quyền tương ứng.
- Đăng ký: cho phép thành viên đăng ký tài khoản để truy cập hệ thống.
- Đăng xuất: cho phép người dùng thoát khỏi hệ thống.
- Quản lý thành viên: cho phép quản trị viên thêm, sửa thông tin, xóa thành viên.

- Quản lý sự kiện: cho phép quản trị viên thêm, sửa sự kiện của đội.
- Quản lý phí hoạt động: cho phép quản trị viên thêm, sửa phí hoạt động cho 1 sự kiện bất kì.
- Thêm sinh viên tham gia sự kiện: cho phép quản trị viên tham gia chọn thành viên chính thức trong đội tham gia 1 sự kiện bất kì.
- Lên lịch phỏng vấn: cho phép quản trị viên lên lịch phỏng vấn cho thành viên không chính thức.
- Thống kê: khi quản trị viên click vào nút thống kê. Hệ thống sẽ thống kê ra tổng số thành viên trong đội, tổng số thành viên chính thức, tổng số thành viên không chính thức, tổng số sự kiện đội hoạt động, tổng chi phí của tất cả các sự kiện ấy hoạt động, ngân sách của đội còn lại, danh sách các sự kiện đội đoàn hoạt động. Khi quản trị viên ấn lưu, hệ thống sẽ lưu giữ những thông tin kia vào file.
- Quản lý thông tin cá nhân: cho phép người dùng xem, thay đổi thông tin và trạng thái của người dùng.



Hình 1. Biểu đồ UseCase tổng quát

➤ Đặc tả tóm tắt các UseCase

ID	Tên use case	Mô tả ngắn gọn use case	Chức năng	Ghi chú
UC1	Đăng nhập	Use case cho phép người dùng đăng nhập vào hệ thống.	Hệ thống kiểm tra mã sinh viên và mật khẩu trong file “User.DAT”. Đăng nhập đúng, cho phép truy cập vào hệ thống.	Người dùng

			Đăng nhập sai, báo lỗi và yêu cầu nhập lại.	
UC2	Đăng ký	Use case cho phép sinh viên đăng ký để trở thành thành viên của đội thanh niên tình nguyện.	Hệ thống cho phép sinh viên đăng ký và đợi admin duyệt làm thành viên chính thức. Hệ thống ghi thông tin vào file “User.DAT”	Thành viên chưa chính thức
UC3	Quản lý thông tin cá nhân	Use case cho phép người dùng xem thông tin cá nhân	Hệ thống đọc dữ liệu từ file “User.DAT” và hiển thị thông tin người dùng đó lên hình.	Thành viên chưa chính thức và chính thức
		Use case cho phép sửa thông tin cá nhân và trạng thái của bản thân.	Cho phép sinh viên sửa thông tin cá nhân và trạng thái của bản thân và ghi dữ liệu thay đổi vào file “User.DAT”. Hệ thống hiển thị các thông báo từ admin.	
UC4	Quản lý	Use case cho phép admin	Hệ thống đọc dữ	Quản trị

	thành viên	thêm thành viên.	liệu từ file “User.DAT” và hiển thị danh sách thành viên lên màn hình. Cho phép admin thêm thành viên và cập nhật lại danh sách thành viên vào file “User.DAT”.	viên
		Use case cho phép admin sửa thông tin thành viên.	Hệ thống hiển thị thông tin thành viên mà quản trị viên lựa chọn. Khi quản trị viên ấn lưu, hệ thống sẽ cập nhật lại thông tin vào file “User.DAT”.	
		Use case cho phép admin xóa thành viên.	Hệ thống xóa thành viên mà quản trị viên lựa chọn, sau đó ghi lại danh sách thành viên vào file “User.DAT”.	
UC5	Quản lý sự kiện	Use case cho phép quản trị viên thêm mới sự kiện.	Hệ thống cho phép quản trị viên thêm mới một sự kiện và	Quản trị viên

			ghi thông tin sự kiện ấy vào file “Event.DAT”.	
		Use case cho phép quản trị viên sửa thông tin sự kiện.	Hệ thống cho phép quản trị viên sửa thông tin sự kiện mà quản trị viên lựa chọn, ghi lại thông tin sự kiện vào file “Event.DAT”.	
UC6	Quản lý phí hoạt động	Use case cho phép quản trị viên thêm mới chi phí hoạt động của 1 sự kiện.	Hệ thống cho phép quản trị viên thêm mới 1 chi phí hoạt động của 1 sự mà quản trị viên lựa chọn. Sau đó ghi thông tin chi phí vào file “OperatingFee.DAT”.	Quản trị viên
		Use case cho phép quản trị viên sửa 1 chi phí hoạt động của 1 sự kiện.	Hệ thống cho phép quản trị viên sửa 1 chi phí của sự kiện mà quản trị viên lựa chọn. Sau đó ghi lại thông tin thay đổi vào file	

			"OperatingFee.DAT".	
UC7	Lên lịch phòng vấn	Use case cho phép quản trị viên lên lịch phòng vấn những sinh viên đăng ký.	Hệ thống đọc file "User.DAT" và lấy ra những thành viên có trạng thái rảnh (0) và chưa là thành viên chính thức (0) sau đó hiển thị ra màn hình. Quản trị viên lên lịch cho các thành viên và hệ thống sẽ lưu dữ liệu vào file "DataInterview.DAT"	Quản trị viên
UC8	Thống kê	Use case cho phép quản trị viên thống kê lại thông tin của tất cả các danh mục trong hệ thống.	Hệ thống thống kê lại dữ liệu tổng quát của tất cả các danh mục và lưu các thông tin vào file Statistical.txt.	Quản trị viên
UC9	Thêm sinh viên tham gia sự kiện.	Use case cho phép quản trị viên chọn các thành viên rảnh và là chính thức tham gia các sự kiện.	Hệ thống đọc file "User.DAT" và đưa ra những thành viên chính thức và có trạng thái rảnh lên	Quản trị viên

	kiện		màn hình. Cho phép quản trị viên chọn những thành viên và sự kiện. Hệ thống sẽ sửa lại trạng thái của những thành viên được chọn thành bận và idEvent của thành viên những thành viên đây là idEvent mà quản trị viên chọn. Hệ thống sẽ ghi lại sự thay đổi vào file “User.DAT”	
UC1 0	Đăng xuất	Use case cho phép người dùng thoát khỏi hệ thống.	Hệ thống sẽ hiển thị ra màn hình giao diện đăng nhập.	Người dùng

2.3.2. Mô hình hóa dữ liệu và giao diện hệ thống

2.3.2.1. Use case đăng nhập

- Thông tin đầu vào: mã sinh viên, mật khẩu.
- Phác thảo giao diện:

Hình 2. Giao diện đăng nhập – Phùng Tuấn Minh

2.3.2.2. Use case đăng ký

- Thông tin đầu vào: mã sinh viên, họ tên, khoa, lớp, email, mật khẩu, xác nhận mật khẩu.
- Phác thảo giao diện:

The image shows a wireframe of a registration form. It consists of seven input fields stacked vertically, each with a label to its left: 'Mã sinh viên', 'Họ và tên', 'Khoa', 'Lớp', 'Email', 'Mật khẩu', and 'Nhập lại mật khẩu'. Below the input fields are two buttons: 'Đăng ký' (green) and 'Hủy' (green). The entire form is enclosed in a rectangular border.

Hình 3. Giao diện đăng ký – Phùng Tuấn Minh

2.3.2.3. Use case quản lý thông tin cá nhân

- Thông tin đầu vào: mã sinh viên, họ tên, khoa, lớp, mật khẩu, email, tình trạng.
- Phác thảo giao diện:

Đăng xuất

Mã sinh viên

Họ và tên

Khoa

Lớp

Mật khẩu

Email

Trạng thái ☐ Bạn ☐ Rảnh ☐ Có nhiệm vụ

Sửa

Hình 4. Giao diện quản lý thông tin cá nhân – Lê Ngọc Trường

2.3.2.4. Use case quản lý thành viên

- Thông tin đầu vào: danh sách các thành viên, mỗi thành viên gồm các thông tin về id, mã sinh viên, họ tên, khoa, lớp, mật khẩu, email, trạng thái, thông tin.
- Phác thảo giao diện:

Danh sách thành viên ➡

ID	Mã sinh viên	Họ tên	Khoa	Lớp	Mật khẩu	Email	Trạng thái	Thông tin

Chức năng

Thêm thành viên Sửa thông tin thành viên Xóa thành viên Thoát

Hình 5. Giao diện quản lý thành viên – Lê Ngọc Trường

2.3.2.5. Use case quản lý sự kiện

- Thông tin đầu vào: danh sách các sự kiện, mỗi sự kiện gồm các thông tin về id, tên sự kiện, ngày bắt đầu, ngày kết thúc, số lượng thành viên cần, số lượng thành viên thiếu, địa chỉ, tổng chi phí.
- Phác thảo giao diện:

Danh sách sự kiện

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số lượng thành viên cần	Số lượng thành viên thiếu	Địa chỉ	Tổng chi phí

Chức năng

Hình 6. Giao diện quản lý sự kiện – Diệp Thị Linh

2.3.2.6. Use case quản lý phí hoạt động

- Thông tin đầu vào: danh sách các phí hoạt động, mỗi phí hoạt động gồm các thông tin về id, tên khoản phí, kinh phí, tên sự kiện.
- Phác thảo giao diện:

Danh sách chi phí

ID	Tên khoản phí	Kinh phí	Sự kiện

Chức năng

Hình 7. Giao diện quản lý phí hoạt động – Trần Đình Tuấn

2.3.2.7. Use case lên lịch phỏng vấn

- Thông tin đầu vào: mã sinh viên, họ tên, khoa, lớp, ngày phỏng vấn, email, trạng thái, thông tin.

- Phác thảo giao diện:

Danh sách sinh viên chưa chính thức

Mã sinh viên	Họ tên	Khóa	Lớp	Ngày phỏng vấn	Email	Trạng thái	Thông tin

Ngày phỏng vấn

Hình 8. Giao diện lên lịch phỏng vấn – Phùng Tuấn Minh

2.3.2.8. Use case thống kê

- Thông tin đầu vào: tổng số lượng thành viên đội, số lượng thành viên chưa chính thức, số lượng thành viên chính thức, số lượng sự kiện, tổng chi phí hoạt động tất cả các sự kiện, ngân sách, danh sách các sự kiện gồm có id, tên sự kiện, ngày bắt đầu, ngày kết thúc, số lượng thành viên cần, địa chỉ, tổng chi phí.
- Phác thảo giao diện:

Tổng thành viên: Sự kiện đội:

Số thành viên chính thức: Tổng chi phí hoạt động:

Số thành viên chưa chính thức: Ngân sách:

Danh sách sự kiện

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số lượng thành viên cần	Số lượng thành viên thiếu	Địa chỉ	Tổng chi phí

Hình 9. Giao diện thống kê – Mai Thị Hương

2.3.2.9. Use case thêm sinh viên tham gia sự kiện

- Thông tin đầu vào: danh sách thành viên, mỗi thành viên gồm các thông tin về mã sinh viên, họ tên, trạng thái, thông tin. Danh sách các sự kiện, mỗi sự kiện bao gồm các thông tin về id, tên sự kiện, ngày bắt đầu, ngày kết thúc, số thành viên cần, số lượng thành viên thiếu, địa chỉ.
- Phác thảo giao diện:

Mã sinh viên	Họ tên	Trạng thái	Thông tin

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số thành viên cần	Số thành viên thiếu	Địa chỉ

Hình 10. Giao diện thêm sinh viên tham gia sự kiện – Lê Ngọc Trường

2.3.2.10. Use case đăng xuất

- Thông tin đầu vào: giao diện đăng nhập.
- Phác thảo giao diện:

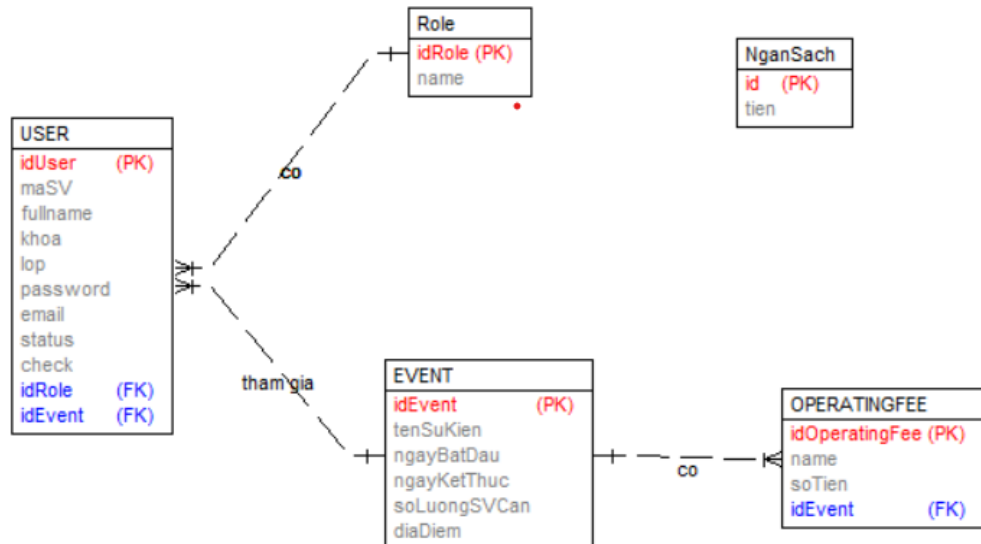
Đăng nhập

Mã sinh viên

Mật khẩu

Hình 11. Giao diện đăng xuất – Lê Ngọc Trường

❖ Sơ đồ lớp dữ liệu sẽ cài đặt trong hệ thống



Hình 12. Sơ đồ lớp dữ liệu

2.4. Thực hiện bài toán

2.4.1. Mai Thị Hường – Thống kê

- Giao diện “Thống kê”

Tổng thành viên: 8 Số sự kiện đội: 4

Số thành viên chính thức: 5 Tổng chi phí hoạt động: 87345

Số thành viên chưa chính thức: 3 Ngân sách đội: 2112655

Danh sách sự kiện đội

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số lượng thành viên	Địa chỉ	Tổng chi phí
1	Tuyển thành viên	15/09/2023	15/09/2023	15	Ha Nam	30000
2	Du lịch	14/05/2023	15/05/2023	100	Soc Son Ha Noi	12000
3	Tính nguyên	25/09/2023	27/09/2023	46	Chùa Ha Ha Noi	33000
4	Thi	02/03/2023	02/03/2023	61	Dai hoc CNHN	12345

In danh sách... Thoát

Hình 13. Giao diện thống kê

➤ Mã lệnh:

- Hướng đối tượng: sử dụng tính chất kế thừa

```
public class Statistical extends javax.swing.JFrame {
```

- Bắt lỗi và gom rác: sử dụng try-catch bắt lỗi đọc file IOException, tác dụng của bắt lỗi này là tránh không tìm thấy file, hoặc file không tồn tại để đọc

```
// Lấy ra tổng số user, số user chưa chính thức, số user chính thức
try {
    numberUser = userController.getNumberUser();
} catch (IOException e) {
    noti.showNotiError("Có lỗi: " + e.toString());
}

// Lấy ra danh sách sự kiện của đội
try {
    listEvent = eventController.getListEvents();
} catch (IOException e) {
    noti.showNotiError("Có lỗi: " + e.toString());
} catch (Exception e2) {
    noti.showNotiError("Có lỗi: " + e2.toString());
}
```

- Tập hợp : ArrayList để lưu dữ liệu

```
// Lưu thông tin tổng số tv, số tv chưa chính thức, số thành viên chính thức
ArrayList<Integer> numberUser = new ArrayList<>();
// Lưu thông tin danh sách Event
ArrayList<Event> listEvent;
```

- Thao tác file: lựa chọn thao tác đọc file với luồng kiểu char: dùng Scanner để đọc tập tin.

```
public List<User> readUsersFromFile(String filename) throws IOException {
    fileController.OpenFileToRead( fileName: filename);
    List<User> users = new ArrayList<>();

    while (fileController.scanner.hasNext()) {
        String data = fileController.scanner.nextLine();
        String[] a = data.split( regex: "\\|");
        users.add(new User( id:Long.parseLong(a[0]), a[1], a[2], a[3], a[4], a[5], a[6],
    })

    fileController.CloseFileAfterRead( fileName: filename);

    return users;
}
```

Ghi dữ liệu vào file:

```
// Ghi thông tin thống kê được vào file
public boolean ghiDuLieu(String tongTV, String tvChua, String tvChinh, String soSK, String tongCP, String nganSach, ArrayList<Event> listEvent) {
    fileController.OpenFileToWrite( fileName: Constant.STATISTICAL_FILE);

    fileController.getPrintWriter().println("Tong thanh vien: " + tongTV + "\tSo su kien doi: " + soSK);
    fileController.getPrintWriter().println("So thanh vien chinh thuc: " + tvChinh + "\tTong chi phi hoat dong: " + tongCP);
    fileController.getPrintWriter().println("So thanh vien chua chinh thuc: " + tvChua + "\tNgan sach doi: " + nganSach);
    fileController.getPrintWriter().println(x: "");
    fileController.getPrintWriter().println(x: "\t=====Danh sach su kien doi=====");
    fileController.getPrintWriter().println(x: String.format( format: "%-5s %-20s %-15s %-15s %-15s %-15s", args: "ID", args:
    for (Event e : listEvent) {
        fileController.getPrintWriter().println(x: String.format( format: "%-5s %-20s %-15s %-15s %-15s %-15s", e.getId() +
    })

    fileController.CloseFileAfterWrite();
    return true;
}
```

2.4.2. Diệp Thị Linh – Quản lý sự kiện

- Giao diện “Quản lý sự kiện”

Danh sách sự kiện

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số lượng thành viên cần	Số lượng thành viên thiếu	Địa chỉ	Tổng chi phí
1	Tuyen thanh vien	15/09/2023	15/09/2023	15	14	Ha Nam	30000
2	Du lich	14/05/2023	15/05/2023	100	98	Soc Son Ha Noi	12000
3	Tinh nguyen	25/09/2023	27/09/2023	46	45	Chua Ha Ha Noi	33000
4	Thi	02/03/2023	02/03/2023	61	60	Dai hoc CNHN	12345

Chức năng

Thêm sự kiện

Sửa thông tin sự kiện

Thêm thành viên tham gia sự kiện

Quản lý chi phí

Thoát

Quản lý sự kiện

Danh sách sự kiện

ID	Tên sự kiện	Ngày bắt đầu	Ngày kết thúc	Số lượng thành viên cần	Số lượng thành viên thiếu	Địa chỉ	Tổng chi phí
1	Tuyen thanh vien	15/09/2023	15/09/2023	15	14	Ha Nam	30000
2	Du lich	14/05/2023	15/05/2023	100	98	Soc Son Ha Noi	12000
3	Tinh nguyen	25/09/2023	27/09/2023	46	45	Chua Ha Ha Noi	33000
4	Thi	02/03/2023	02/03/2023	61	60	Dai hoc CNHN	12345

Chức năng

Thêm sự kiện

Sửa thông tin sự kiện

Thêm thành viên tham gia sự kiện

Quản lý chi phí

Thoát

—

□

×

Tên sự kiện

Ngày bắt đầu

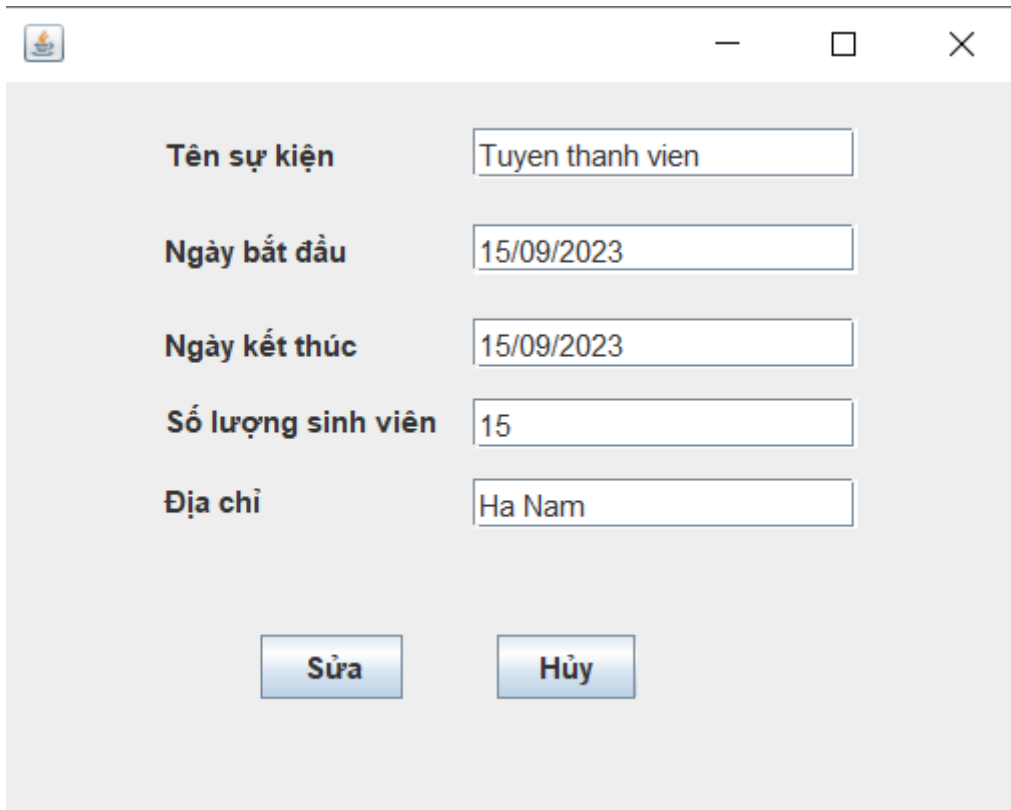
Ngày kết thúc

Số lượng sinh vi...

Địa chỉ

Thêm

Hủy



The image shows a Java Swing window with a title bar containing a small icon and standard window controls (minimize, maximize, close). The window has a light gray background and contains the following elements:

- Tên sự kiện** (Event Name): A text field containing "Tuyen thanh vien".
- Ngày bắt đầu** (Start Date): A text field containing "15/09/2023".
- Ngày kết thúc** (End Date): A text field containing "15/09/2023".
- Số lượng sinh viên** (Number of Students): A text field containing "15".
- Địa chỉ** (Address): A text field containing "Ha Nam".
- Buttons**: Two buttons at the bottom, "Sửa" (Edit) and "Hủy" (Cancel), both with a blue gradient and a 3D effect.

➤ Mã lệnh:

- Hướng đối tượng:
 - sử dụng tính chất kế thừa

```
public class QLEvent extends javax.swing.JFrame {
```

- sử dụng tính chất bao đóng

```

private Long id;
private String nameEvent;
private String startDay;
private String endDay;
private Integer numberOfStudent;
private String address;
private List<OperatingFee> operatingFees = new ArrayList<>();
private List<User> users = new ArrayList<>();

public Event() {
}

public Event(Long id, String nameEvent, String startDay, String endDay, :
    this.id = id;
    this.nameEvent = nameEvent;
    this.startDay = startDay;
    this.endDay = endDay;
    this.numberOfStudent = numberOfStudent;
    this.address = address;
}

public Long getId() {
    return id;
}

public void setId(Long id) {
    this.id = id;
}

public String getNameEvent() {
    return nameEvent;
}

```

- Bắt lỗi và gom rác: sử dụng throws đẩy lỗi cho phương thức gọi nó và dùng try-catch để bắt lỗi.

```

// set list OperatingFee cho event có idEvent trùng với idEvent bên OperatingFee
public void setListOperatingFee(Event e) throws IOException {
    // 1 đối tượng OperatingFeeController để ta có thể thao tác với dữ liệu phí hoạt động
    OperatingFeeController operatingFeeController = new OperatingFeeController();
    ArrayList<OperatingFee> listOperatingFee = operatingFeeController.getListCost();
    ArrayList<OperatingFee> listCostOfEvent = new ArrayList<>();
    for (OperatingFee fee : listOperatingFee) {
        if (fee.getIdEvent() == e.getId()) {
            listCostOfEvent.add(e: fee);
        }
    }
    e.setOperatingFees(operatingFees: listCostOfEvent);
}

```

- Bắt lỗi thông tin không được để trống.


```

try {
    if (name.isEmpty() || end.isEmpty() || start.isEmpty() || address.isEmpty()) {
        noti.showNotiError( noti: "Vui lòng nhập đầy đủ thông tin");
    }
    Integer numberOfStudent = Integer.parseInt(s:number.getText().trim());
    try {
        eventController.createEvent( nameEvent:name, startDay:start, endDay:end, numberOfSt
        noti.showNotiInformation( noti: "Thêm sự kiện thành công");
        new QLEvent( masv:masV, password).setVisible( b:true);
        this.setVisible( b:false);
    } catch (Exception ex) {
        noti.showNotiError( noti: "Thêm sự kiện không thành công");
    }
} catch (Exception e) {
    noti.showNotiError( noti: "Số lượng thành viên là số.");
}
}

```

- Tập hợp: sử dụng interface List và triển khai ArrayList

```

// Lưu thông tin danh sách Event
ArrayList<Event> listEvent;

```

- Thao tác file: Sử dụng thao tác đọc, ghi file bằng luồng kiểu char. Dùng lớp PrintWriter để ghi file và dùng Scanner để đọc dữ liệu luồng kiểu char. Hàm đọc, lưu file tổng quát được lưu ở lớp FileController.java. Dùng PrintWriter để ghi dữ liệu vào file vì nó cung cấp một cách dễ dàng để ghi các giá trị vào file văn bản. Nó cũng tự động thực hiện một số thao tác như chuyển đổi số thành chuỗi và ngược lại. Ngoài ra, có thể sử dụng phương thức println() để ghi các giá trị xuống dòng mới. Dùng Scanner để đọc dữ liệu vào file vì cung cấp một cách dễ dàng để đọc các giá trị từ file văn bản. Nó cũng tự động thực hiện một số thao tác như chuyển đổi chuỗi thành số và ngược lại. Ngoài ra, có thể sử dụng phương thức nextLine() để đọc từng dòng dữ liệu từ file.

```
// Method get Events từ File
public List<Event> readEventsFromFile(String fileName) throws IOException, Exception {
    fileController.OpenFileToRead(fileName);
    List<Event> listEvents = new ArrayList<>();

    while (fileController.scanner.hasNext()) {
        String data = fileController.scanner.nextLine();
        String[] arrData = data.split( regex: "\\|");
        listEvents.add(new Event( id:Long.parseLong(arrData[0]), arrData[1], arrData[2],
    )

    fileController.CloseFileAfterRead(fileName);

    return listEvents;
}

// Method ghi Event vào file
public List<Event> writeEventToFile(List<Event> listEvent, String fileName) throws IOException, Exception {
    fileController.OpenFileToWrite(fileName);
    for (Event e : listEvent) {
        fileController.getPrintWriter().println(e.getId() + "|" + e.getNameEvent() + "|" + e.getStartDay() + "|" +
    }

    fileController.CloseFileAfterWrite();
    return listEvent;
}
```

Thêm / sửa sự kiện sau đó ghi vào file

```
public Boolean createEvent(String nameEvent, String startDay, String endDay, Integer numberOfStudent, String address) throws Exception {
    List<Event> events = readEventsFromFile( fileName: Constant.EVENT_FILE);
    Integer id = events.size();
    long idnew = id + 1;
    Event event = new Event(id:idnew, nameEvent, startDay, endDay, numberOfStudent, address);

    events.add(e:event);
    writeEventToFile( listEvent:events, fileName:Constant.EVENT_FILE);
    return true;
}

public Boolean repairEvent(long idEvent, String nameEvent, String startDay, String endDay, Integer numberOfStudent, String address) throws Exception {
    List<Event> events = readEventsFromFile( fileName:Constant.EVENT_FILE);

    for (int i = 0; i < events.size(); i++) {
        if (events.get( index:i).getId().equals( obj:idEvent)) {
            Event event = events.get( index:i);
            event.setAddress(address);
            event.setEndDay(endDay);
            event.setNameEvent(nameEvent);
            event.setNumberOfStudent(numberOfStudent);
            event.setStartDay(startDay);

            writeEventToFile( listEvent:events, fileName:Constant.EVENT_FILE);
            return true;
        }
    }

    return false;
}
```

2.4.3. Phùng Tuấn Minh – Đăng nhập, đăng ký, lên lịch phỏng vấn

➤ Giao diện “Đăng nhập”

ĐĂNG NHẬP

Mã sinh viên

Mật khẩu

➤ Giao diện “đăng ký”:

Mã sinh viên

Họ tên sinh viên

Khoa

Lớp

Email

Mật khẩu

Xác nhận mật khẩu

➤ Giao diện “lên lịch phỏng vấn”:

Lên lịch phỏng vấn

Danh sách viên chưa chính thức							
Mã sinh viên	Họ tên	Khoa	Lớp	Ngày phỏng vấn	Email	Trạng thái	Thông tin
2020600871	Nguyen Thi M	K14	KHMT	03/03/2023	nguyenthim@gmail.com	Rảnh	TV chưa chính thức
123456789	Le Van Toan	K14	KTPM	02/03/2023	abc@gmail.com	Rảnh	TV chưa chính thức

Ngày phỏng vấn

➤ Mã lệnh:

- Hướng đối tượng: sử dụng tính chất kế thừa

```
public class SetDateInterview extends javax.swing.JFrame {
```

- Bẫy lỗi và gom rác:

- Bẫy lỗi nhập thông tin đăng nhập không chính xác:

```
public Long logIn(String maSV, String password) throws IOException {
    List<User> users = readUsersFromFile( filename: "User.DAT");
    User cur_user = new User();
    for (int i = 0; i < users.size(); i++) {
        if (users.get( index:i ).getMaSV().equalsIgnoreCase( anotherString:maSV ) &&
            cur_user = users.get( index:i );
        List<User> user_cur = new ArrayList<>();
        user_cur.add( e:users.get( index:i ));
        writeUsersToFile( users:user_cur, filename: Constant.USER_CUR_FILE);
        return cur_user.getIdRole();
    }
}

return (long)-1;
}
```

- Bẫy lỗi nhập thông tin đăng ký không hợp lệ:

```
// Kiểm tra nhập đủ dữ liệu
if (maSV.trim().isEmpty() || fullname.trim().isEmpty() || khoa.trim().isEmpty() || lop.trim().isEmpty() || password.trim().isEmpty() || email.trim().isEmpty()) {
    noti.showNotiError(noti: "Vui lòng nhập đầy đủ thông tin!");
} else {
    // Kiểm tra RePassword trùng với Password
    if (password.equalsIgnoreCase(anotherString: repassword)) {
        // Kiểm tra rằng buộc dữ liệu của password và email (Check Validate) ở đây sẽ check xem
        // mật khẩu hợp lệ phải chứa ít nhất một chữ cái thường, một số, có ít nhất 8 ký tự và chỉ bao gồm các ký tự được liệt kê
        // Email hợp lệ có dạng abc@xyz.com
        if (Constant.regexPassword.matcher(input: password).matches()) {
            if (Constant.regexEmail.matcher(input: email).matches()) {
                try {
                    boolean kt = control.addUser(maSV, hoTen: fullname, khoa, lop, password, email);
                    if (kt) {
                        // Hiện thị dialogBox thông tin
                        noti.showNotiInformation(noti: "Thêm thành viên thành công!!!");
                        new Login().setVisible(b: true);
                        this.setVisible(b: false);
                    } else {
                        // Hiện thị DialogBox báo lỗi
                    }
                }
            }
        }
    }
}
```

- Tập hợp: sử dụng tập hợp ArrayList

- Sử dụng phương thức add()

```
listUsers.add(e: newUser);
```

- Thao tác với file: Sử dụng thao tác đọc file luồng kiểu char. Dùng Scanner để đọc file và PrintWriter để ghi file.

- Đọc file

```
while (fileController.scanner.hasNext()) {
    String data = fileController.scanner.nextLine();
    String[] a = data.split(regex: "\\|");
    users.add(new User(id: Long.parseLong(a[0]), a[1], a[2], a[3], a[4], a[5], a[6],
    fileController.CloseFileAfterRead(fileName: filename);
```


- Ghi file

```
fileController.getPrintWriter().println(user.getId() + "|" + user.getMaSV() + "|" + user.getFullName()
```

2.4.4. Lê Ngọc Trường – Quản lý thành viên, quản lý thông tin cá nhân, đăng xuất, thêm thành viên tham gia sự kiện

- Giao diện:

- Giao diện “quản lý thành viên”


 Quản lý thành viên

Danh sách thành viên

ID	Mã sinh viên	Họ tên	Khoa	Lớp	Password	Email	Trạng thái	Thông tin
0	2020600871	Nguyen Thi M	K14	KHMT	nguyenthm1010*	nguyenthm@gmail...	Rảnh	TV chưa chính thức
1	2020600873	Nguyen Van B	K15	CNTT02	nguyenvanb1010*	nguyenvanb@gmail...	Bận	TV chưa chính thức
2	2020600874	Nguyen Van C	K15	CNTT03	nguyenvanc1010*	nguyenvanc@gmail...	Rảnh	TV chính thức
3	2020600877	Nguyen Van F	K15	CNTT05	nguyenvanf1010*	nguyenvanf@gmail...	Có nhiệm vụ	TV chính thức
4	2020600872	Lam Linh	K15	CNTT01	lamLinh1005*	lamlinh@gmail.com	Có nhiệm vụ	TV chính thức
5	2020601391	Le Ngoc Truong	K15	HTTT01	Truongwf0701@	ngoctruongcv@gma...	Có nhiệm vụ	TV chính thức
6	202012345	Le Van K	K14	KTMP01	Truongwf701@	levank@gmail.com	Có nhiệm vụ	TV chính thức
7	123456789	Le Van Toan	K14	KTPM	Truongwf0701	abc@gmail.com	Rảnh	TV chưa chính thức

Chức năng

[Thêm thành viên](#)
[Sửa thông tin thành viên](#)
[Xóa thành viên](#)
[Thoát](#)

 Thông tin sinh viên

Mã sinh viên

Họ và tên

Khoa

Lớp


Password

Email

Tình trạng ☐ Bận ☐ Rảnh ☐ Có nhiệm vụ

Thông tin

[Thêm](#)
[Hủy](#)

 Thông tin sinh viên

Mã sinh viên

Họ và tên

Khoa

Lớp

Password

Email

Tình trạng ☐ Bận ☐ Rảnh ☒ Có nhiệm vụ

Thông tin

[Sửa](#)
[Hủy](#)

- Giao diện “quản lý thông tin cá nhân”

Thông tin sinh viên

Đăng xuất

Mã sinh viên: 2020600871

Password:

Họ và tên: Nguyen Thi M

Email: nguyenthim@gmail.com

Khoa: K14

Tình trạng:
 ☐ Bận
 ☒ Rảnh
 ☐ Có nhiệm vụ

Lớp: KHMT

Sửa

Thông báo

Thời gian phỏng vấn của bạn: 03/03/2023

- Giao diện “thêm thành viên tham gia sự kiện”

Chọn sinh viên đi sự kiện

Danh sách thành viên rảnh

Mã sinh viên	Họ tên	Trạng thái	Thông tin
2020600874	Nguyen Van C	Rảnh	TV chính thức

Thông tin sự kiện

ID	Tên sự ki...	Ngày bắt ...	Ngày kết ...	Số thành ..	Số thành ..	Địa chỉ
1	Tuyen th...	15/09/20...	15/09/20...	15	14	Ha Nam
2	Du lịch	14/05/20...	15/05/20...	100	98	Soc Son ...
3	Tinh ngu...	25/09/20...	27/09/20...	46	45	Chua Ha...
4	Thi	02/03/20...	02/03/20...	61	60	Dai hoc ...

Thêm Thoát

➤ Mã lệnh:

- Hướng đối tượng:
 - Thực hiện các tính đa hình thông qua nạp chồng (overloading).

```

public User getUser(String maSV, String password) throws IOException {
    List<User> users = readUsersFromFile(filename: "User.DAT");
    User cur_user = new User();
    for (int i = 0; i < users.size(); i++) {
        if (users.get(i).getMaSV().equalsIgnoreCase(anotherString: maSV) && users.get(i).getPassword().equalsIgnoreCase(anotherString: password)) {
            cur_user = users.get(i);
        }
    }
    return cur_user;
}

public User getUser(String maSV) throws IOException {
    List<User> users = readUsersFromFile(filename: Constant.USER_FILE);
    User cur_user = new User();
    for (int i = 0; i < users.size(); i++) {
        if (users.get(i).getMaSV().equals(anObject: maSV)) {
            cur_user = users.get(i);
        }
    }
    return cur_user;
}

```

ở đây có 2 phương thức cùng tên là `getUse` nhưng khác nhau về số lượng tham số.

- Thực hiện tính kế thừa: kế thừa lớp `JFrame`

```
public class QLSinhVien extends javax.swing.JFrame {
```

- Thực hiện tính chất đóng gói:

```
// Tạo 1 userController để có thể xử lý logic liên quan đến dữ liệu của User
private UserController control = new UserController();
// Tạo 1 đối tượng Notification để có thể dùng hiển thị các thông báo khi thực hiện 1 điều gì đó. Tối ưu sử dụng lại code
private Noti noti = new Noti(frame:this);
// 2 biến lưu mã sinh viên của admin và password admin sau khi thoát hiển thị lại
private String maSVAdmin;
private String passwordAdmin;

public AddStudent(String masvadmin, String passwordadmin) {
    initComponents();
    this.maSVAdmin = masvadmin;
    this.passwordAdmin = passwordadmin;
}

public AddStudent() {
    initComponents();
}
```

```
public class User {
    private Long id;
    private String maSV;
    private String fullName;
    private String khoa;
    private String lop;
    private String password;
    private String email;
    private Integer status; //
    private Integer check; //
    private Long idRole; // 1
    private Long idEvent;

    public String getMaSV() {
        return maSV;
    }

    public void setMaSV(String maSV) {
        this.maSV = maSV;
    }

    public String getFullName() {
        return fullName;
    }

    public void setFullName(String fullName) {
        this.fullName = fullName;
    }

    public String getKhoa() {
        return khoa;
    }

    public void setKhoa(String khoa) {
        this.khoa = khoa;
    }

    public String getLop() {
        return lop;
    }
}
```

➤ Bắt lỗi và gom rác:

- Bắt lỗi nhập thông tin: không được để trống, email phải dạng abc@xyz.kmn.


```

// Kiểm tra nhập đủ dữ liệu
if (maSV.trim().isEmpty() || hoTenSV.trim().isEmpty() || khoa.trim().isEmpty() || lop.trim().isEmpty() || password.trim().isEmpty())
    noti.showNotiError(noti: "Vui lòng nhập đầy đủ thông tin!");
} else {
    // Kiểm tra rằng buộc dữ liệu của password và email (Check Validate) ở đây sẽ check xem
    // mật khẩu hợp lệ phải chứa ít nhất một chữ cái thường, một số, có ít nhất 8 ký tự và chỉ bao gồm các ký tự được liệt kê
    // Email hợp lệ có dạng abc@xyz.com
    if (Constant.regexPassword.matcher(input: password).matches()) {
        if (Constant.regexEmail.matcher(input: email).matches()) {

```

- Bắt lỗi đọc ghi file dùng throws đầy lỗi và try-catch để bắt:

```

public User getUse(String maSV) throws IOException {
    List<User> users = readUsersFromFile( filename: Constant.USER_FILE);
    User cur_user = new User();
    for (int i = 0; i < users.size(); i++) {
        if (users.get( index: i ).getMaSV().equals( anObject: maSV)) {
            cur_user = users.get( index: i );
        }
    }
    return cur_user;
}

// Lấy user thông qua controller
try {
    model = control.getUse(maSV);
} catch (IOException e) {
    noti.showNotiError("Loi " + e.toString());
}

```

- Tập hợp: sử dụng interface List để dễ dàng chuyển qua các triển khai của interface như ArrayList, Vector, LinkedList và sử dụng triển khai ArrayList để lưu trữ dữ liệu

```

public List<User> readUsersFromFile(String filename) throws IOEx
    fileController.OpenFileToRead( fileName: filename);
    List<User> users = new ArrayList<>();

    while (fileController.scanner.hasNext()) {
        String data = fileController.scanner.nextLine();
        String[] a = data.split( regex: "\\|");
        users.add(new User( id: Long.parseLong(a[0]), a[1], a[2],

    }

// Get ra list User trong File User
public ArrayList<User> getListUsers() throws IOException {
    ArrayList<User> users = (ArrayList< User>) readUsersFromFile( filename: Constant.USER_FILE);
    return users;
}

```

- Sử dụng phương thức add() và remove(), sử dụng iterator

```
User newUser = new User(id, maSV, fullName: hoTen, khoa, lop,
System.out.println(x:newUser.toString());
users.add(e:newUser);
writeUsersToFile(users, filename: Constant.USER_FILE);

List<User> users = readUsersFromFile(filename: Constant.USER_FILE);
Iterator<User> iterator = users.iterator();
while(iterator.hasNext()){
    User user = iterator.next();
    if(user.getMaSV().equalsIgnoreCase(anotherString: maSV)){
        iterator.remove();
    }
}
```

- Thao tác với file

- Sử dụng thao tác đọc, ghi file bằng luồng kiểu char. Dùng lớp PrintWriter để ghi file và dùng Scanner để đọc dữ liệu luồng kiểu char. Hàm đọc, lưu file tổng quát được lưu ở lớp FileController.java. Dùng PrintWriter để ghi dữ liệu vào file vì nó cung cấp một cách dễ dàng để ghi các giá trị vào file văn bản. Nó cũng tự động thực hiện một số thao tác như chuyển đổi số thành chuỗi và ngược lại. Ngoài ra, có thể sử dụng phương thức println() để ghi các giá trị xuống dòng mới. Dùng Scanner để đọc dữ liệu vào file vì cung cấp một cách dễ dàng để đọc các giá trị từ file văn bản. Nó cũng tự động thực hiện một số thao tác như chuyển đổi chuỗi thành số và ngược lại. Ngoài ra, có thể sử dụng phương thức nextLine() để đọc từng dòng dữ liệu từ file.
- Đọc/ghi file

```

public List<User> readUsersFromFile(String filename) throws IOException {
    fileController.OpenFileToRead( fileName: filename);
    List<User> users = new ArrayList<>();

    while (fileController.scanner.hasNext()) {
        String data = fileController.scanner.nextLine();
        String[] a = data.split( regex: "\\|");
        users.add(new User( id:Long.parseLong(a[0]), a[1], a[2], a[3], a[4], a[5], a[6],
    }

    fileController.CloseFileAfterRead( fileName: filename);

    return users;
}

```

```

public List<User> writeUsersToFile(List<User> users, String filename) throws IOException {
    fileController.OpenFileToWrite( fileName: filename);
    for (User user : users) {
        fileController.getPrintWriter().println(user.getId() + "|" + user.getMaSV() + "|" + user.getFullName() +
    }

    fileController.CloseFileAfterWrite();
    return users;
}

```

Lưu file sau khi thêm/ sửa/ xóa thành viên:

```

public boolean addUser(String maSV, String hoTen, String khoa, String lop, String password, String email, int status, int check, long idRole) throws IOException {
    List<User> users = readUsersFromFile( filename: Constant.USER_FILE);
    for (User u : users) {
        if (u.getMaSV().equalsIgnoreCase( anotherString: maSV)) {
            return false;
        }
    }
    long id = users.size();
    long idEvent = 0;
    User newUser = new User(id, maSV, fullName:hoTen, khoa, lop, password, email, status, check, idRole, idEvent);
    System.out.println( newUser.toString());
    users.add( newUser);
    writeUsersToFile(users, filename:Constant.USER_FILE);
    return true;
}

```

```

public boolean repairInformatin(String maSV, String tenSV, String khoa, String lop, String password, String email, int status, int checks, long idRole) {
    List<User> users = readUsersFromFile( filename: "User.DAT");
    User cur_user = new User();
    int check = -1;
    for (int i = 0; i < users.size(); i++) {
        if (users.get( index:i).getMaSV().equalsIgnoreCase( anotherString:maSV)) {
            cur_user = users.get( index:i);
            check = 1;
        }
    }
    if (check == 1) {
        cur_user.setFullName( fullName: tenSV);
        cur_user.setKhoa( khoa);
        cur_user.setLop( lop);
        cur_user.setPassword( password);
        cur_user.setEmail( email);
        cur_user.setStatus( status);
        cur_user.setCheck( check: checks);
        cur_user.setIdRole( idRole);
        writeUsersToFile(users, filename: "User.DAT");
        return true;
    }
    return false;
}

```

```

public boolean removeUser(String maSV) throws IOException{
    List<User> users = readUsersFromFile( filename: Constant.USER_FILE);
    Iterator<User>iterator = users.iterator();
    while(iterator.hasNext()){
        User user = iterator.next();
        if(user.getMaSV().equalsIgnoreCase( anotherString: maSV)) {
            iterator.remove();
        }
    }
    for(int i = 0 ;i < users.size();i++){
        users.get( index:i).setId((long)i);
    }
    writeUsersToFile(users, filename: Constant.USER_FILE);
    return true;
}

```

2.4.5. Trần Đình Tuấn – Quản lý chi phí hoạt động

➤ Giao diện “quản lý chi phí hoạt động”

Quản lý chi phí

Danh sách chi phí

ID	Tên khoản phí	Kinh phí	Sự kiện
1	Tien in banner	20000	Tuyen thanh vien
2	Tien mua nuoc	10000	Tuyen thanh vien
3	Tien in poster	12000	Du lich
4	Tien quyen gop	30000	Tinh nguyen
5	Phi di chuyen	2000	Tinh nguyen
6	Test	1000	Tinh nguyen
7	Mua rau	0	Du lich
8	In Giay	12345	Thi

Chức năng

Tên khoản phí

Kinh phí

Sự kiện: ▼

Tên khoản phí	<input type="text" value="Tien in banner"/>
Kinh phí	<input type="text" value="20000"/>
Sự kiện:	<input type="text" value="Tuyen thanh vien"/> ▼
<div><input type="button" value="Sửa"/> <input type="button" value="Thoát"/></div>	

➤ Mã lệnh:

- Hướng đối tượng:
 - Sử dụng tính kế thừa kế thừa lớp JFrame

```
public class QLOperationFee extends javax.swing.JFrame {
```

- Sử dụng tính chất bao đóng

```

private Long id;
private String nameFee;
private Long money;
private Long idEvent;

public OperatingFee() {
}

public OperatingFee(Long id, String nameFee, Long money, Long idEvent) {
    this.id = id;
    this.nameFee = nameFee;
    this.money = money;
    this.idEvent = idEvent;
}

public Long getId() {
    return id;
}

public void setId(Long id) {
    this.id = id;
}

public String getNameFee() {
    return nameFee;
}

```

- Bẫy lỗi và gom rác:
 - sử dụng throws đẩy lỗi cho phương thức gọi nó và dùng try-catch để bắt lỗi.

```

public OperatingFee getOperatingFee(long idOF) throws Exception{
    ArrayList<OperatingFee> listOFs = getListCost();
    for(OperatingFee of : listOFs){
        if(of.getId() == idOF){
            return of;
        }
    }
    return null;
}

```

```

// lấy operating fee thông qua controller
try{
    model = operatingFeeController.getOperatingFee(idOF);
} catch (Exception e) {
    noti.showNotiError("Có lỗi: " + e.toString());
}

```

- Tập hợp : Sử dụng tập hợp ArrayList để lưu trữ dữ liệu

```

// Lưu thông tin danh sách Operation Fee
ArrayList<OperatingFee> listOperatingFee;

```

- Sử dụng phương thức add()

```

OperatingFee of = new OperatingFee( id:idOF, nameFee:nameCost, money:cost, idEvent);
listOperatingFee.add( e:of);

```

- Thao tác với file
 - Sử dụng thao tác đọc, ghi file bằng luồng kiểu char. Dùng lớp PrintWriter để ghi file và dùng Scanner để đọc dữ liệu luồng kiểu char. Hàm đọc, lưu file tổng quát được lưu ở lớp FileController.java.
 - Đọc/ghi file

```

// Phương thức get Fees từ File
public List<OperatingFee> readOperatingFeeFromFile(String fileName) throws IOException{
    fileController.OpenFileToRead(fileName);
    List<OperatingFee> listFee = new ArrayList<OperatingFee>();

    while(fileController.scanner.hasNext()){
        String data = fileController.scanner.nextLine();
        String[] a = data.split( regex:"\\|");
        listFee.add(new OperatingFee( id:Long.parseLong(a[0]), a[1], money:Long.parseLong(a[2]), idEvent:Long.parseLong(a[3])));
    }

    fileController.CloseFileAfterRead(fileName);

    return listFee;
}

public List<OperatingFee> writeOperatingFeeToFile(List<OperatingFee> listFee, String filename) throws IOException{
    fileController.OpenFileToWrite( fileName: filename);
    for(OperatingFee fee : listFee){
        fileController.getPrintWriter().println(fee.getId() + "|" + fee.getNameFee() + "|" + fee.getMoney() + "|" + fee.getIdEvent());
    }

    fileController.CloseFileAfterWrite();
    return listFee;
}

```

PHẦN 3 KẾT LUẬN VÀ BÀI HỌC KINH NGHIỆM

3.1. Nội dung đã thực hiện

Sau khi hoàn thành đề tài “Xây dựng chương trình quản lý đội thanh niên tình nguyện khoa Công Nghệ Thông Tin”, nhóm chúng em đã có thể nắm được những kiến thức chuyên môn của môn học như:

- Lập trình Java cơ sở.
- Lập trình Java hướng đối tượng.
- Xử lý ngoại lệ, bắt lỗi.
- Thao tác với file.
- Tập hợp.
- Biết cách sử dụng công cụ NetBeans.

Những chuẩn đầu ra của học phần đã đạt được:

- Phát biểu được bài toán cần xây dựng, phân tích và đưa ra sơ đồ lớp mô tả bài toán theo hướng đối tượng.
- Vận dụng các kỹ thuật đã học trong lập trình Java cài đặt được bài toán theo chủ đề được giao.
- Sản phẩm nghiên cứu: Quyển thuyết minh chủ đề “Xây dựng phần mềm quản lý trông xe Đại học Công nghiệp Hà Nội” và sản phẩm mã nguồn.

Ngoài những kiến thức liên quan đến môn học, chúng em còn tích lũy, trau dồi thêm những kỹ năng mềm khác như:

- Kỹ năng làm việc nhóm.
- Kỹ năng thu thập và chọn lọc thông tin.
- Kỹ năng quản lý thời gian, phân chia công việc.
- Biết lắng nghe và tiếp thu ý kiến của các thành viên.

3.2. Hướng phát triển

Đề tài "Xây dựng chương trình quản lý đội thanh niên tình nguyện khoa Công Nghệ Thông Tin" là một đề tài rất cần thiết trong việc quản lý và tăng

cường hoạt động của đội thanh niên tình nguyện. Sau khi hoàn thành đề tài chúng em nhận thấy rằng ưu điểm của đề tài sẽ giúp đơn giản hóa các hoạt động đăng ký làm thành viên đội, quản lý thành viên, phân chia công việc, thống kê kết quả và cải thiện hiệu quả hoạt động của đội tình nguyện viên. Tuy nhiên, do lượng kiến thức còn hạn chế và thời gian thực hiện đề tài ngắn nên việc tìm hiểu triển khai tính năng vẫn còn nhiều khó khăn, thiếu sót, giao diện còn khá đơn giản, và một vài thành phần có thời gian xử lý lâu chưa được tối ưu và tính bảo mật cũng chưa đảm bảo nhiều. Tuy nhiên, đây là một chủ đề nghiên cứu khả thi và tiềm năng. Chương trình có thể nâng cấp lên thành hệ thống quản lý các câu lạc bộ, các đội đoàn của các khoa khác hoặc cho toàn trường, đó là hướng phát triển tiếp theo của đề tài.

TÀI LIỆU THAM KHẢO

1. Tài liệu học tập:

- Giáo trình Lập trình HĐT với Java, Nguyễn Bá Nghiễn, Ngô Văn Bình, Vương Quốc Dũng, Đỗ Sinh Trường; NXB Thống kê, 2020.
- Bộ slide bài giảng lập trình java- Bộ môn CMPM- trường ĐHCN HN.
- The Java Programming Language; Author: K. Arnold, J. Gosling; Published: Addison-Wesley, 1996, ISBN 0-201-63455-4.
- Lập trình hướng đối tượng với Java; Đoàn Văn Ban; NXB Khoa học và Kỹ thuật, Hà Nội 2006 (Tái bản).
- Lập trình Java nâng cao, Đoàn Văn Ban, NXB Khoa học và Kỹ thuật, Hà Nội 2006.

2. Phương tiện, nguyên liệu thực hiện bài tập lớn:

- Máy tính, bộ công cụ phát triển ứng dụng JDK phiên bản 19.
- Phần mềm hỗ trợ viết mã nguồn NetBeans 16.
- Phần mềm soạn thảo văn bản Microsoft word.
- Phần mềm hỗ trợ phân tích thiết kế hệ thống: Rational Rose, CaseStudio 2, Balsamiq Mockups 3.