

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA CÔNG NGHỆ PHẦN MỀM



**CHUYÊN ĐỀ MOBILE AND
PERVASIVE COMPUTING**

Lớp:SE405.H11

ĐỀ TÀI:ỨNG DỤNG ORDERFOOD

GVHD: Nguyễn Trác Thức

Sinh Viên Thực Hiện:

Trịnh Đình Loan - 12520231

Trịnh Chấn Phát - 12520311

Tp. Hồ Chí Minh, ngày 18 tháng 12 năm 2016

LỜI CẢM ƠN

Trên thực tế, không có sự thành công nào mà không gắn liền với những sự hỗ trợ, giúp đỡ dù ít hay nhiều, dù trực tiếp hay gián tiếp của người khác. Trong suốt thời gian từ khi bắt đầu học tập tại trường đại học đến nay, chúng em đã nhận được rất nhiều sự quan tâm, giúp đỡ của quý Thầy Cô, gia đình và bạn bè.

Với lòng biết ơn sâu sắc nhất, chúng em xin gửi đến quý Thầy Cô ở Khoa Công nghệ Phần mềm – Trường Đại Học Công Nghệ Thông Tin đã cùng với tri thức và tâm huyết của mình để truyền đạt vốn kiến thức quý báu cho chúng em trong suốt thời gian học tập tại trường. Và đặc biệt, trong học kỳ này, Khoa đã tổ chức cho chúng em được tiếp cận với môn học mà theo chúng em là rất hữu ích đối với sinh viên ngành Công nghệ Phần mềm. Đó là môn học “Chuyên đề Mobile and Pervasive Computing”.

Chúng em xin chân thành cảm ơn Thầy Nguyễn Trác Thức đã tận tâm hướng dẫn chúng em thông qua những buổi giảng dạy, nói chuyện và thảo luận. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì chúng em nghĩ bài thu hoạch này sẽ rất khó có thể hoàn thiện được. Một lần nữa, chúng em xin chân thành cảm ơn thầy.

Sau cùng, chúng em xin kính chúc quý Thầy Cô trong Khoa Công nghệ Phần mềm và Thầy Nguyễn Trác Thức thật dồi dào sức khỏe, niềm tin để tiếp tục thực hiện sứ mệnh cao đẹp của mình là truyền đạt kiến thức cho thế hệ mai sau.

Trân trọng!

TP.HCM, ngày 29 tháng 12 năm 2016

Sinh viên thực hiện:

Trịnh Đình Loan

Trịnh Chấn Phát

NHẬN XÉT CỦA GIẢNG VIÊN







[illegible]

MỤC LỤC

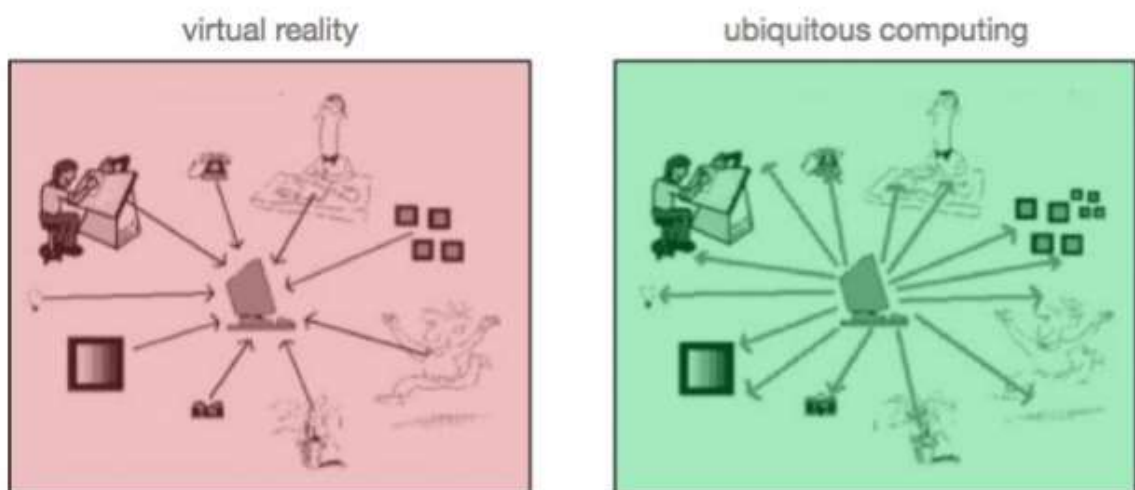
CHƯƠNG 1: GIỚI THIỆU	6
1.1 Tổng quan về Mobile & Pervasive Computing.....	6
1.2 Giới thiệu về Project sẽ phát triển.....	16
CHƯƠNG 2: CÔNG NGHỆ	17
2.1 Mô hình 3 lớp trong android	17
2.2 Cơ sở dữ liệu SQLite trong android	19
2.2.1 Giới thiệu:.....	19
CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT:.....	21
3.1 Thiết kế:	21
3.1.1 Thiết kế chức năng:	21
3.1.3.Thiết kế cơ sở dữ liệu:	53
3.1.4.Thiết kế kiến trúc:.....	53
3.1.5.Thiết kế giao diện:	56
CHƯƠNG 4: PHỤ LỤC- HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG:	63
4.1 Hướng dẫn cài đặt:	63
4.1.1 . Giới thiệu về Android Studio	63
4.1.2 Hướng dẫn download Android Studio cho Window	63
Hướng dẫn cài đặt Android Studio.....	63
4.2.Hướng dẫn sử dụng	68

CHƯƠNG 1: GIỚI THIỆU

1.1 Tổng quan về Mobile & Pervasive Computing

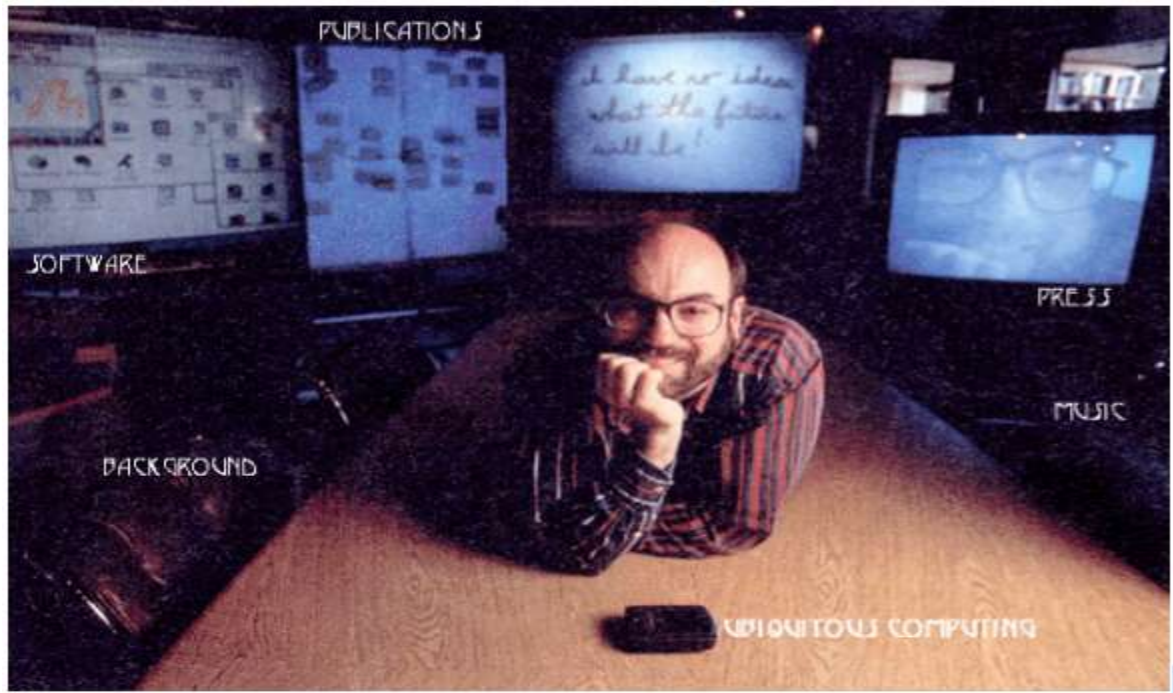
- Ubiquitous, Mobile, và Nomadic Computing.
 - Mobile Computing: “on-the-go”, ví dụ, khi bạn đang ngồi trên một chuyến tàu, bạn vẫn có thể kết nối mạng như bình thường.
 - Máy tính có ở hầu hết mọi nơi.
- Mobile Computing.
 - Mobile Computing đang ngày càng phát triển để đạt tới truyền dẫn không dây.
 - Nhiều loại Mobile Computing đã được giới thiệu từ những năm 1990, bao gồm:
 -  Personal Digital Assistant.
 -  Enterprise Digital Assistant.
 -  Smart phones.
 -  UMPC.
- Mobile Computing Vision.
 - Những kết nối phổ biến – bất cứ nơi đâu, bất cứ lúc nào.
 - Điều chỉnh sự đồng nhất của liên kết mạng lưới và truyền thông.
 - Môi trường thông minh Ubiquitous – máy tính chạy hệ thống của chúng ở khắp mọi nơi.
 - Tương tác người dùng dễ dàng.
 - Truy cập độc lập với các dịch vụ và các thông tin phụ thuộc.
- Ubiquitous Computing (ubicomp).
 - Ubicomp là một mẫu máy tính để bàn của con người nhằm tương tác với máy tính, trong đó quá trình xử lý thông tin đã được tích hợp một cách cẩn thận vào đối tượng và các hoạt động.
 - Tích hợp máy tính để tương tác với toàn thể giới.
 -  Invisible, máy tính ở khắp mọi nơi.
 -  Thường được gọi là pervasive/invisible computing.

- Máy tính chủ yếu không phải là invisible, chúng chiếm ưu thế trong việc tương tác với con người.
 - Ubicomp sẽ làm cho các máy tính trở nên invisible.
 - Ubiquitous computing = mobile computing + môi trường thông minh.
- Technology View.
- Các hệ thống nhúng có ở khắp mọi nơi – tủ lạnh, máy giặt, ổ khóa, xe hơi, đồ nội thất.
 - Môi trường thông minh.
 - Các thiết bị di động và máy tính xách tay.
 - Thông tin liên lạc không dây – điện thoại di động/cố định.
- User View.
- Invisible – tương tác ngầm với môi trường thiết bị của bạn.
 - Tăng cường khả năng của con người trong tình huống xử lý các task.
- Ubicomp vs Virtual Reality.
- Chúng ta có nên sống trong thế giới ảo hay không? Hay là nên loại bỏ máy tính và sống trong thế giới thực của chúng ta?
 - VR là mô phỏng thế giới vật chất và đặt con người vào trong thế giới máy tính ảo (bị giới hạn các ứng dụng và hoạt động).
 - Ubicomp là việc đưa máy tính đến với thế giới thực của con người, kết hợp các đối tượng và các hoạt động hằng ngày lại với nhau.
 - Ubiquitous computing là sự kết hợp của yếu tố con người, khoa học máy tính, kỹ thuật và khoa học xã hội.








➤ History of Ubicomp.

- Mark Weiser đặt ra cụm từ “Ubicomp computing” vào năm 1988, trong nhiệm kỳ nắm giữ chức vụ trưởng khoa công nghệ của trung tâm nghiên cứu Xerox Palo Alto.
- Cùng với giám đốc PARC và giám đốc khoa học John Seely Brown, Weiser đã viết một số bài báo đầu tiên về đề tài này, chủ yếu là xác định và phác thảo những mối quan tâm lớn nhất về nó.
- Andy Hopper từ Đại học Cambridge vương quốc Anh đã đề xuất và chứng minh khái niệm “Teleport” - ứng dụng theo dõi người dùng bất cứ khi nào họ di chuyển.
- Bill Schilit (nay làm việc tại Google) cũng đã thực hiện một số công việc trước đó trong vấn đề này, và tham gia vào các hội thảo về Mobile Computing được tổ chức tại Santa Cruz năm 1996.
- Tiến sĩ Ken Sakamura của Đại học Tokyo, Nhật Bản dẫn đầu phòng thí nghiệm Ubiquitous Networking (UNL), Tokyo cũng như diễn đàn T-Engine.
- Mục tiêu chung của Sakamura's Ubiquitous Networking và diễn đàn T-Engine là cho phép bất kì thiết bị nào cũng có thể phát và nhận thông tin mỗi ngày.
- Roy Want, trong khi đang là một nhà nghiên cứu và sinh viên đang làm việc dưới quyền Andy Hopper tại Đại học Cambridge, đã làm việc trên “Active Badge System”, một hệ thống xác định vị trí tiên tiến, nơi mà người dùng di động cá nhân đã được sát nhập với máy tính.
- MIT cũng đã đóng góp những nghiên cứu quan trọng về lĩnh vực này, đáng chú ý là tập đoàn Things That Think tại Media Lab và những nỗ lực của CSAIL vào Project Oxygen.
- Những đóng góp quan trọng khác bao gồm University of Washington's Ubicomp Lab, Georgia Tech's College of Computing, Cornell University's People Aware Computing Lab, NYU's Interactive Telecommunications Program, UC Irvine's Department of Informatics, Microsoft Research, Intel Research.



➤ Ubicomp core concepts.

- Ubiquitous computing (ubicomp) là một khái niệm điện toán tiên tiến, nơi mà máy tính có thể xuất hiện ở bất cứ khi nào và bất cứ nơi đâu.
- Ngược lại với máy tính để bàn, ubiquitous computing có thể xuất hiện bằng việc sử dụng bất kì thiết bị, bất kì vị trí và bất kì định dạng nào.
- Một người dùng tương tác với máy tính, có thể tồn tại ở nhiều dạng khác nhau, bao gồm máy tính xách tay, máy tính bảng, thiết bị đầu cuối và điện thoại.
- Công nghệ cơ bản hỗ trợ ubiquitous computing bao gồm Internet, các phần mềm trung gian tiên tiến, hệ điều hành, mã điện thoại, cảm biến, bộ vi xử lý, I/O mới, giao diện người dùng, mạng, các giao thức điện thoại di động, vị trí, định vị, và ngay cả những vật liệu mới.
- Ví dụ, một môi trường ubiquitous computing trong nước có thể kết nối ánh sáng và các môi trường điều khiển với màn hình sinh trắc học cá nhân để có thể tạo nên những bộ quần áo, chiếu sáng và sưởi ấm trong điều kiện một căn phòng có thể được điều chế, liên tục và không đáng kể.

- Một viễn cảnh khác là tủ lạnh có thể “nhận thức” các nội dung đã được lập trình trên chúng, chúng ta có thể cầm một loạt các loại thực phẩm trên tay và tủ lạnh sẽ cảnh báo người dùng những thực phẩm đã cũ hoặc bị hư hỏng.
- Ubiquitous computing là những thách thức trên khoa học máy tính: trong thiết kế hệ thống và kỹ thuật, xây dựng mô hình hệ thống, và trong thiết kế giao diện người dùng.
- Mô hình hiện đại của tương tác giữa con người và máy tính, cho dù dòng lệnh, menu điều khiển, hoặc dựa trên GUI, không phù hợp và không đủ để giải quyết các trường hợp phổ biến.
- Ubiquitous computing có thể được nhìn thấy bao gồm nhiều lớp, mỗi lớp với vai trò của mình, kết hợp lại với nhau tạo thành một hệ thống duy nhất.
- Lớp 1: Lớp quản lý công việc:
 -  Nhiệm vụ của người sử dụng cần cho các dịch vụ trong môi trường.
 -  Quản lý những thuộc tính phức tạp.
- Lớp 2: Lớp quản lý môi trường:
 -  Để theo dõi một nguồn lực và khả năng.
- Lớp 3: Lớp môi trường:
 -  Để theo dõi một nguồn tài nguyên có liên quan.
 -  Để quản lý tin cậy của các nguồn tài nguyên.

➤ Intelligence.

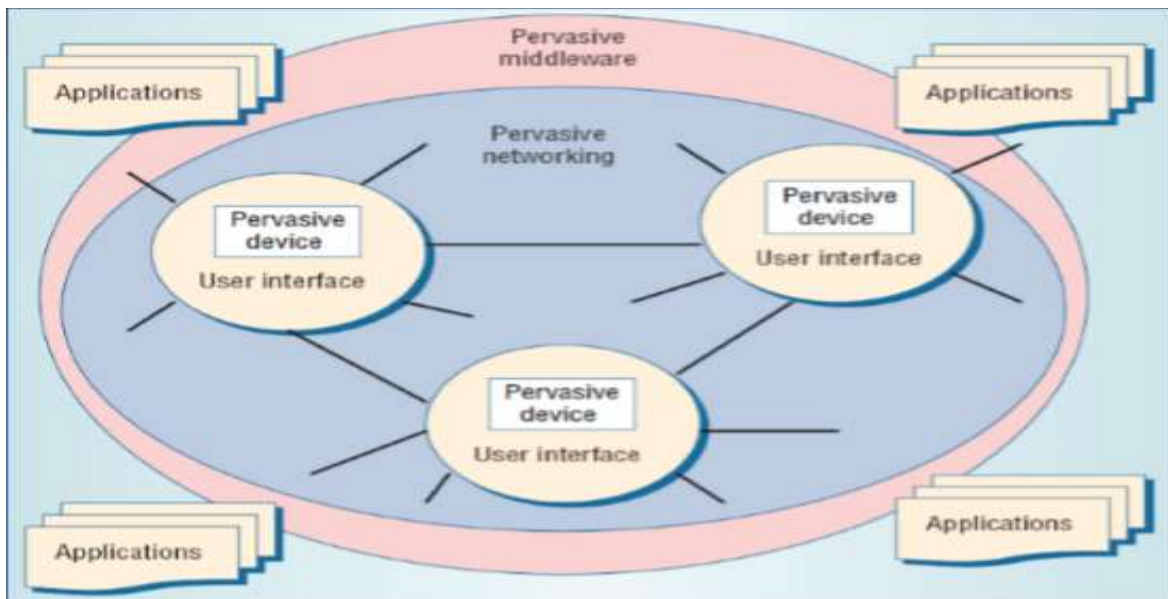
- Máy tính nhúng để tăng cường đối tượng vật lý.
- Đạt được trí thông minh thông qua kết nối của các đối tượng vật lý.
- Đạt được trí thông minh thông qua nhận thức vị trí (không có AI). Ví dụ: Chuyển tiếp cuộc gọi tự động (context awareness) điều khiển ánh sáng → tường cảm biến thông minh sưởi ấm → kiểm soát ánh sáng.

➤ Early work Tabs:

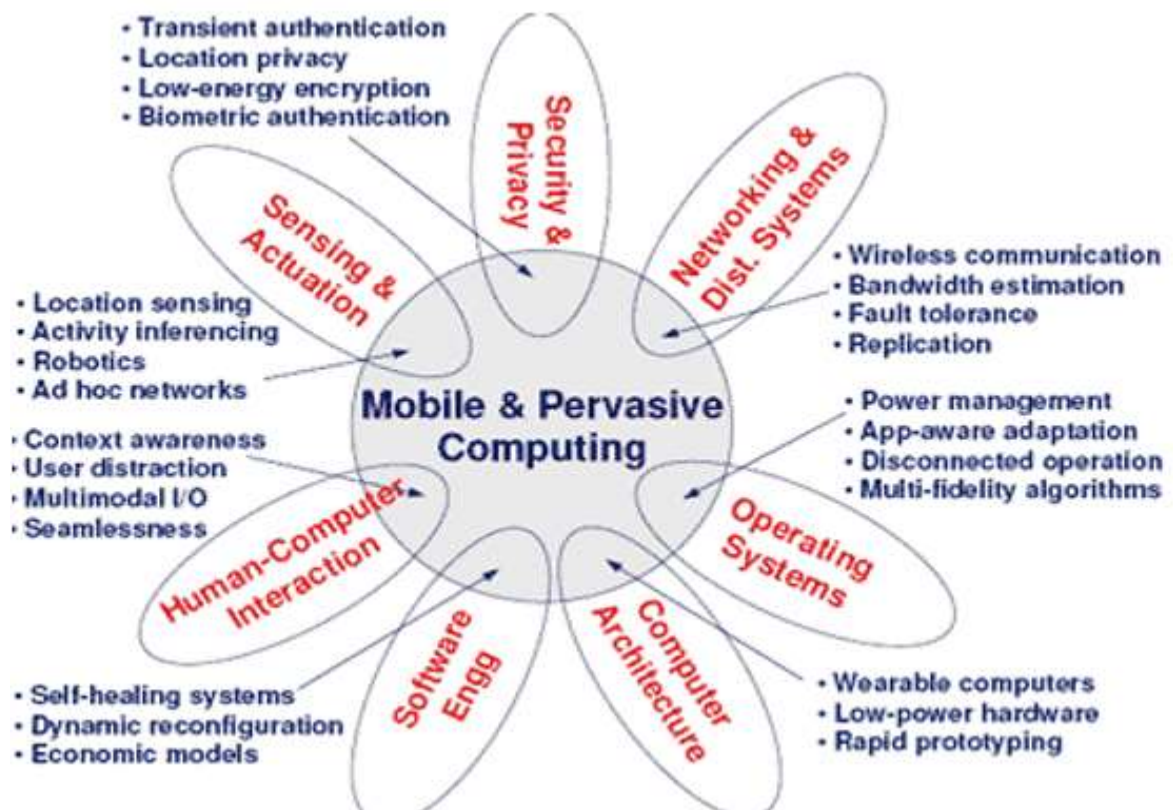
- Rất nhỏ - huy hiệu thông minh với thông tin người dùng, lịch, nhật ký,...

- Cho phép thiết lập cá nhân để theo dõi người dùng.
- Tiến hành xung quanh bởi một người.
- Current Trends Touch Pads:
 - Foot-scale Ubicomp devices:
 - ✚ Tờ giấy/ máy tính bảng.
 - ✚ Máy tính xách tay nhưng không phải laptop.
 - Tens in a room:
 - ✚ Giống như giấy phế liệu có thể được nắm lấy và sử dụng bất cứ nơi nào, không có ID đặc trưng.
- Current Technology.
 - Thiết bị thông tin di động:
 - ✚ Laptops, notebooks, and sub-notebooks.
 - ✚ Máy tính xách tay.
 - ✚ PDAs và điện thoại thông minh.
 - Mạng lưới thông tin liên lạc không dây:
 - ✚ Nhiều mạng bao phủ toàn cầu.
 - Internet:
 - ✚ TCP / IP và các giao thức ứng dụng de-facto.
- Usability.
 - Giao diện người dùng chung cho các máy trạm và các ứng dụng thiết bị di động.
 - Hiển thị thông tin thay đổi.
 - Nhận dạng giọng nói + văn bản để chuyển thành hội thoại.
 - Nhận diện cử chỉ.
 - Intelligent agents.

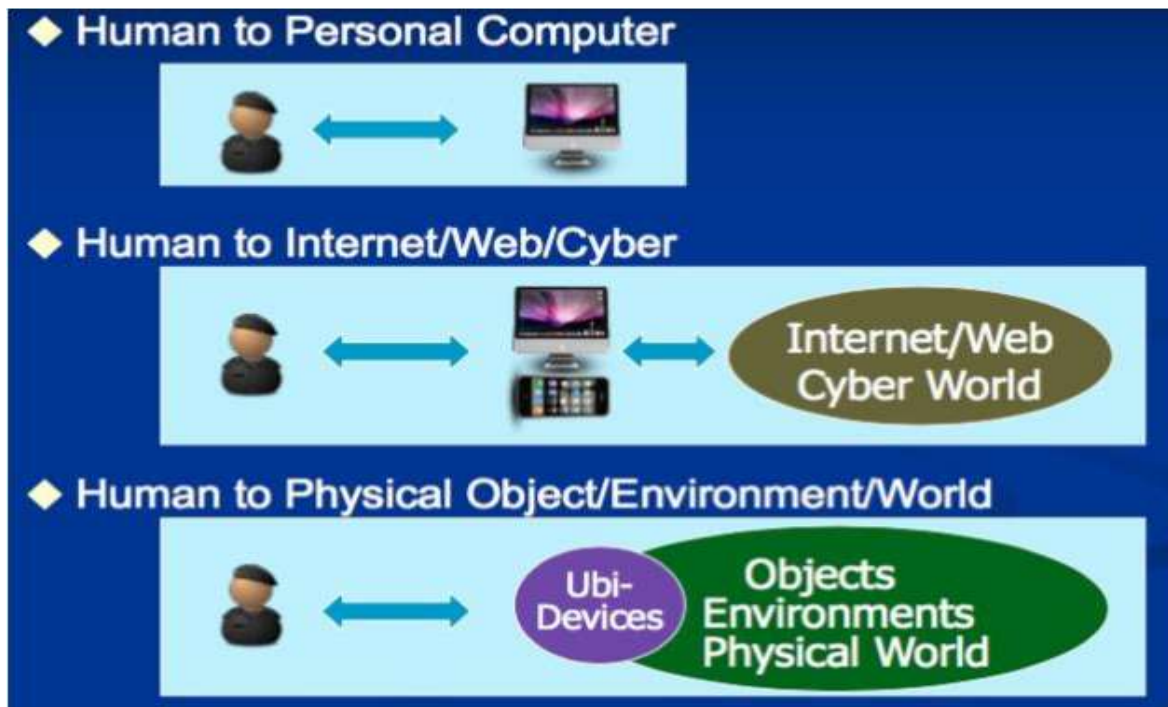
➤ Ubiquitous/Pervasive Computing Framework.



➤ Related Fields.



➤ Human-Computer Interaction.



➤ Integration of Mobile Systems.

- Không để thiết bị mất kết nối: Cần tương tác với các hệ thống thông tin phức tạp. Ví dụ: Cơ sở dữ liệu lớn – sáp nhập thông tin cập nhật, bảng hiển thị,...
- Phát triển hệ thống:
 - ✚ Yêu cầu đặc điểm kỹ thuật cho các hệ thống thích nghi.
 - ✚ Thành phần để đáp ứng QoS toàn cầu, an ninh, yêu cầu độ tin cậy và hiệu suất.
- Mô hình di động:
 - ✚ Đặc điểm kỹ thuật và phân tích hành vi.
 - ✚ Mô hình bối cảnh hệ thống nhận biết.

➤ Context Aware Computing.

- Nó là một khái niệm phổ biến trong tương tác giữa người và máy tính.
- Mục tiêu của context-aware computing là tiếp thu và sử dụng thông tin về bối cảnh của một thiết bị để cung cấp các dịch vụ thích hợp cho riêng người, địa điểm, thời gian, sự kiện,...

- Ví dụ, một điện thoại di động sẽ luôn luôn rung và không bao giờ kêu bíp trong một buổi hòa nhạc, nếu hệ thống có thể biết được vị trí của điện thoại di động và lịch trình buổi hòa nhạc.
- Context Adaptation.
 - Một hệ thống context adaptive cho phép người sử dụng dùng để duy trì một số ứng dụng (trong các hình thức khác nhau) trong khi chuyển vùng giữa các công nghệ khác nhau như truy cập không dây, địa điểm, thiết bị và thậm chí đồng thời thực hiện các công việc hàng ngày như các cuộc họp, lái xe,...
- Issues : Context Awareness.
 - Vị trí hiện tại: Dùng để xác định vị trí, ví dụ: GPS – radio beacon, IR,...
 - Hoạt động người dùng: Đi bộ, lái xe, đi xe buýt - làm thế nào để phát hiện điều này?
 - Môi trường xung quanh: Trong rạp chiếu phim, nhà hát, ở một mình.
 - Tài nguyên hoặc các dịch vụ có sẵn: Khả năng của thiết bị.
 - Màn hình, thiết bị đầu vào, sức mạnh xử lý, tuổi thọ pin,...
 - QoS hiện tại sẵn có - đặc biệt là cho các liên kết vô tuyến.
- Intelligent Environment.
 - Một môi trường thông minh là một vị trí (ví dụ: nhà, văn phòng, bệnh viện, ...) được trang bị cảm biến, thiết bị truyền động và các máy tính được nối mạng với nhau và với internet.
 - Các thành phần được điều khiển "Cảnh báo thông minh" bởi phần mềm.
 - Nó điều chỉnh môi trường cho phù hợp với con người.
 - Con người có thể nói chuyện với các môi trường bằng việc sử dụng lời nói và ngôn ngữ tự nhiên và các cảm biến có thể giám sát môi trường.
- Smart Dust.
 - “Smart Dust” là thiết bị rất nhỏ không dây Micro Electro Cơ Sensors (MEMS), có thể phát hiện tất cả mọi thứ từ ánh sáng đến rung động.

- Cảm biến nhiệt độ, độ ẩm, ánh sáng, chuyển động với đài phát thanh hai chiều hoặc laser + pin.
- Các ứng dụng tiêu biểu:
 - ✚ Liên quan đến quốc phòng cảm biến ở chiến trường, phát hiện chuyển động,...
 - ✚ Giám sát chất lượng của sản phẩm - độ rung, độ ẩm, độ nóng.
 - ✚ Giám sát các thành phần của xe hơi,...
- Issues.
 - Pin sẽ là nguồn năng lượng không thực tế cho 100K vì xử lý cho mỗi người.
 - Năng lượng mặt trời là không thích hợp cho tất cả các môi trường.
 - Sức mạnh không phải là tốc độ là vấn đề quan trọng cho các thiết kế bộ vi xử lý trong tương lai.
- Major Challenges Hardware Prototype Issue.
 - Công suất tiêu thụ: Không thể thay pin cho nhiều thiết bị Ubicomp thường xuyên.
 - Cân bằng tính năng HW/ SW: Màn hình, mạng, xử lý, bộ nhớ, khả năng lưu trữ, đa nhiệm, QoS,...
 - Dễ mở rộng và sửa đổi (tích hợp so với mô-đun).
- Network Issue.
 - Wireless Media Access (802.11, Bluetooth, Cellular Networks).
 - Quality of Services (RSVP, etc.).
 - Ubicomp devices thay đổi điểm gắn mạng. (Mobile IP).
- Application Issue.
 - Locating people (active badges).
 - Shared drawing in virtual meeting.
- Security.
 - Tương tác sẽ được qua nhiều ranh giới tổ chức đặc điểm kỹ thuật, phân tích và tích hợp cho hệ điều hành không đồng nhất, cơ sở dữ liệu, tường lửa, router.

- Mọi thứ có khả năng bị hack.
 - Là máy truyền động nhỏ, với các dữ liệu bí mật, có thể dễ dàng bị mất hoặc bị đánh cắp - xác thực sinh trắc học.
 - Tồn tại công nghệ bảo mật cần thiết.
- Privacy.
- Dịch vụ định vị theo dõi chuyển động với đơn vị metres.
- Management.
- Không lồ, hệ thống phức tạp:
 - ✚ Hàng tỉ vi xử lý.
 - ✚ Nhiều tổ chức.
 - ✚ Quản lý thế giới vật lý, kiểm soát các cảm biến, thiết bị truyền động.
 - Thiên đường của hacker và virus.
 - Hệ thống truyền thông tin sai lệch về các cá nhân hoặc tổ chức.
 - Sự phức tạp của s / w cài đặt trên máy trạm hay máy chủ - làm thế nào để bạn đối phó với hàng tỷ vấn đề?

1.2 Giới thiệu về Project sẽ phát triển

Ngày nay khoa học kỹ thuật phát triển như vũ bão, lao động trí óc dần thay thế cho lao động chân tay bằng những ứng dụng của khoa học kỹ thuật. Và góp phần đắc lực trong cuộc cách mạng khoa học này phải kể đến lĩnh vực công nghệ thông tin. Công nghệ thông tin được ứng dụng trong nhiều lĩnh vực. Với sự xuất hiện của công nghệ thông tin, công việc của chúng ta được giải quyết một cách nhanh gọn, tiết kiệm được thời gian và của cải từ đó làm cho cuộc sống của chúng ta ngày càng được cải thiện một cách đáng kể. Ứng dụng của công nghệ thông tin là rất rộng rãi đặc biệt là ứng dụng trong các thiết bị mobile ngày càng phát triển và bùng nổ. Như chúng ta đã biết, Hiện nay tại hầu hết các quán ăn, quán cafe, nhà hàng... khi thực khách chọn món ăn trên thực đơn, nhân viên phục vụ phải đứng chờ để ghi chép. Hoặc sau khi chọn xong, khách hàng phải chờ gọi phục vụ đến bàn để đọc món ăn mà mình đã chọn. Việc ghi chép thực đơn có thể sai sót dẫn đến tranh cãi sau này và gây mất uy tín của nhà hàng. Sau khi ghi món xong, nhân viên phục vụ phải đích

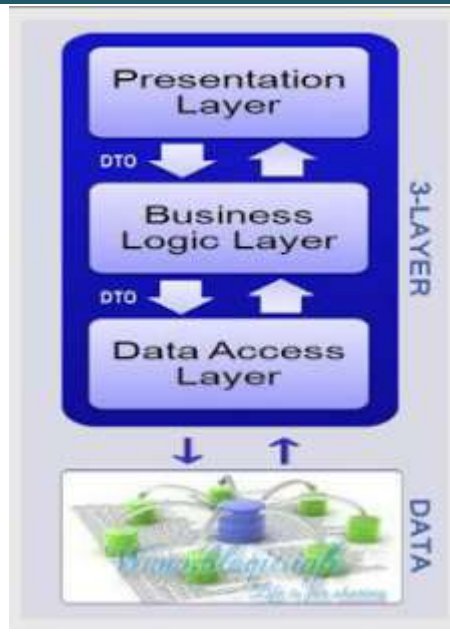
thân mang phiếu xuống nhà bếp hoặc quầy thu ngân. Tại nhà bếp sẽ có người quản lý để phân phối các món ăn cho đầu bếp. Khi món ăn được thực hiện xong, phải gọi cho nhân viên phục vụ để mang thức ăn cho khách. Quãng đường đi lại cũng khá xa, gây mất thời gian và tốn thêm nhân lực phục vụ khách ngoài sảnh. Tất cả công việc này nếu không được tổ chức tốt sẽ gây ách tắc, dễ nhầm lẫn, khó kiểm soát và mất thời gian. Hơn nữa, với phương cách này thì khách hàng cũng không thể biết được hiện các món ăn của mình đã được thực hiện tới đâu. Mặt khác, mỗi lần khách hàng cần nhân viên phục vụ phải ngóng chờ để được phục vụ. từ thực tế như vậy mà nhóm chúng em đã tiến hành xây dựng ứng dụng trên mobile: “**Ứng dụng OrderFood**”.

Ứng dụng là giải pháp tiên tiến, hiệu quả trong việc phục vụ thực khách. Hệ thống sẽ đáp ứng tức thời các nhu cầu của khách hàng, giúp họ hài lòng, thỏa mãn với cung cách phục vụ của nhà hàng. Tất cả các khâu từ lúc khách hàng lựa chọn món ăn, chuyển thực đơn xuống nhà bếp, đầu bếp thực hiện món, chuyển cho nhân viên phục vụ mang đến cho khách, gọi tính tiền, in hóa đơn....đều do hệ thống thực hiện một cách hoàn toàn tự động và khắc phục được hoàn toàn những nhược điểm trên

CHƯƠNG 2: CÔNG NGHỆ

2.1 Mô hình 3 lớp trong android

- **Tại sao lại sử dụng mô hình 3-layer:** Để dễ quản lý các thành phần của hệ thống, cũng như không bị ảnh hưởng bởi các thay đổi, người ta hay nhóm các thành phần có cùng chức năng lại với nhau và phân chia trách nhiệm cho từng nhóm để công việc không bị chồng chéo và ảnh hưởng lẫn nhau. Một trong những mô hình lập trình như vậy đó là Mô hình 3 lớp (Three Layers).
- **Các thành phần trong 3-layer:** Mô hình 3 lớp được cấu thành từ: Presentation Layers, Business Logic Layers, và Data Access Layers.



- **Business Logic Layer:** Đây là layer xử lý chính các dữ liệu trước khi được đưa lên hiển thị trên màn hình hoặc xử lý các dữ liệu trước khi chuyển xuống Data Access Layer để lưu dữ liệu xuống cơ sở dữ liệu.
- **Presentation Layers:** Đây là nơi để kiểm tra ràng buộc, các yêu cầu nghiệp vụ, tính toán, xử lý các yêu cầu và lựa chọn kết quả trả về cho Presentation Layers.
- **Data Access Layer:** Lớp này thực hiện các nghiệp vụ liên quan đến lưu trữ và truy xuất dữ liệu của ứng dụng như đọc, lưu, cập nhật cơ sở dữ liệu.

➤ **Cách vận hành của mô hình:** Đối với 3-Layer, yêu cầu được xử lý tuần tự qua các layer như hình.

- Đầu tiên User giao tiếp với Presentation Layers (GUI) để gửi đi thông tin và yêu cầu. Tại layer này, các thông tin sẽ được kiểm tra, nếu OK chúng sẽ được chuyển xuống Business Logic Layer (BLL).
- Tại BLL, các thông tin sẽ được nhào nặn, tính toán theo đúng yêu cầu đã gửi, nếu không cần đến Database thì BLL sẽ gửi trả kết quả về GUI, ngược lại nó sẽ đẩy dữ liệu (thông tin đã xử lý) xuống Data Access Layer (DAL).
- DAL sẽ thao tác với Database và trả kết quả về cho BLL, BLL kiểm tra và gửi nó lên GUI để hiển thị cho người dùng.

- Một khi gặp lỗi (các trường hợp không đúng dữ liệu) thì đang ở layer nào thì quăng lên trên layer cao hơn nó 1 bậc cho tới GUI thì sẽ quăng ra cho người dùng biết.
- Các dữ liệu được trung chuyển giữa các Layer thông qua một đối tượng gọi là Data Transfer Object (DTO), đơn giản đây chỉ là các Class đại diện cho các đối tượng được lưu trữ trong Database.

➤ **Tổ chức mô hình 3-Layer:** Có rất nhiều cách đặt tên cho các thành phần của 3 lớp như:

- Cách 1: GUI, BUS, DAL.
- Cách 2: GUI, BLL, DAO, DTO.
- Cách 3: Presentation, BLL, DAL

2.2 Cơ sở dữ liệu SQLite trong android

2.2.1 Giới thiệu:

SQLite là cơ sở dữ liệu mã nguồn mở, SQLite hỗ trợ chuẩn quan hệ cơ sở dữ liệu giống như hệ cơ sở dữ liệu SQL. SQLite yêu cầu giới hạn bộ nhớ trong thời gian thực thi là 250 kb. SQLite hỗ trợ chuẩn dữ liệu kiểu Text giống như kiểu dữ liệu String trong Java, Integer tương tự kiểu Long trong Java, và kiểu REAL tương tự kiểu Double trong Java. Tất cả các kiểu dữ liệu khác phải được chuyển đổi thành các kiểu dữ liệu này trước khi được lưu trong cơ sở dữ liệu SQLite không xác định các kiểu dữ liệu được ghi vào các cột thực tế của các kiểu được định nghĩa, có thể viết kiểu Integer trong cột String và ngược lại.

- ❖ **Sqlite in Android:** SQLite được nhúng trong các thiết bị Android, sử dụng một cơ sở dữ liệu SQLite trong Android không yêu cầu thủ tục thiết lập hoặc quản lý cơ sở dữ liệu. Bạn chỉ phải xác định các câu lệnh SQL để tạo và cập nhật cơ sở dữ liệu. Sau đó các cơ sở dữ liệu được quản lý tự động cho bạn bởi các nền tảng Android. Truy cập vào một cơ sở dữ liệu SQLite liên quan đến việc truy cập vào hệ thống tập tin. Điều này có thể được làm chậm. Vì vậy nó được khuyến khích để thực hiện các hoạt động cơ sở dữ liệu không đồng bộ. Nếu ứng dụng của bạn tạo ra một cơ sở dữ liệu, cơ sở dữ liệu này là bởi mặc

DATA: Là đường dẫn mà bạn có thể get bởi câu lệnh `Environment.getDataDirectory()`.

- APP_NAME: Là tên ứng dụng của bạn.
- FILENAME: Là tên cơ sở dữ liệu mà bạn tạo trong ứng dụng.

➤ **Kiến trúc SQLite:**

- Để sử dụng SQLite chúng ta cần import gói `android.database` trong ứng dụng, gói này chứa tất cả các phương thức cần thiết để làm việc với cơ sở dữ liệu. Để làm việc cơ sở dữ liệu SQLite android chúng ta sử dụng 2 phương thức chính là `SQLiteOpenHelper` và `SQLiteDatabase`.
- `SQLiteOpenHelper` là lớp cung cấp các phương thức `getReadableDatabase()` và `getWritableDatabase()` để truy cập đến các đối tượng của lớp `SQLiteDatabase` để đọc và ghi dữ liệu.
- `SQLiteDatabase` là lớp cung cấp các phương thức thao tác với cơ sở dữ liệu.

➤ **Tạo và cập nhật cơ sở dữ liệu với `SQLiteOpenHelper` :**

- Tạo và nâng cấp cơ sở dữ liệu trong ứng dụng Android của bạn, bạn tạo ra một lớp con của lớp `SQLiteOpenHelper`. Trong hàm khởi tạo của lớp con, bạn gọi `super()` phương thức của `SQLiteOpenHelper`, ghi tên cơ sở dữ liệu và phiên bản cơ sở dữ liệu hiện hành. Trong lớp này chúng ta cần ghi đè phương thức tạo và cập nhật cơ sở dữ liệu. Chúng ta sẽ có hai phương thức được ghi đè là:
 - `onCreate()`: Phương thức này được gọi bởi framework, cơ sở dữ liệu đã được truy cập nhưng chưa được tạo.
 - `onUpgrade()`: Phương thức này cho phép bạn cập nhật phiên bản mới cơ sở dữ liệu của bạn hoặc xóa cơ sở dữ liệu đã tồn tại và cập nhật cơ sở dữ liệu mới thông qua phương thức `onCreate()`. Cả hai phương thức này đều nhận đối tượng `SQLiteDatabase` làm tham số.

➤ **Thao tác cơ sở dữ liệu với SQLiteDatabase:**

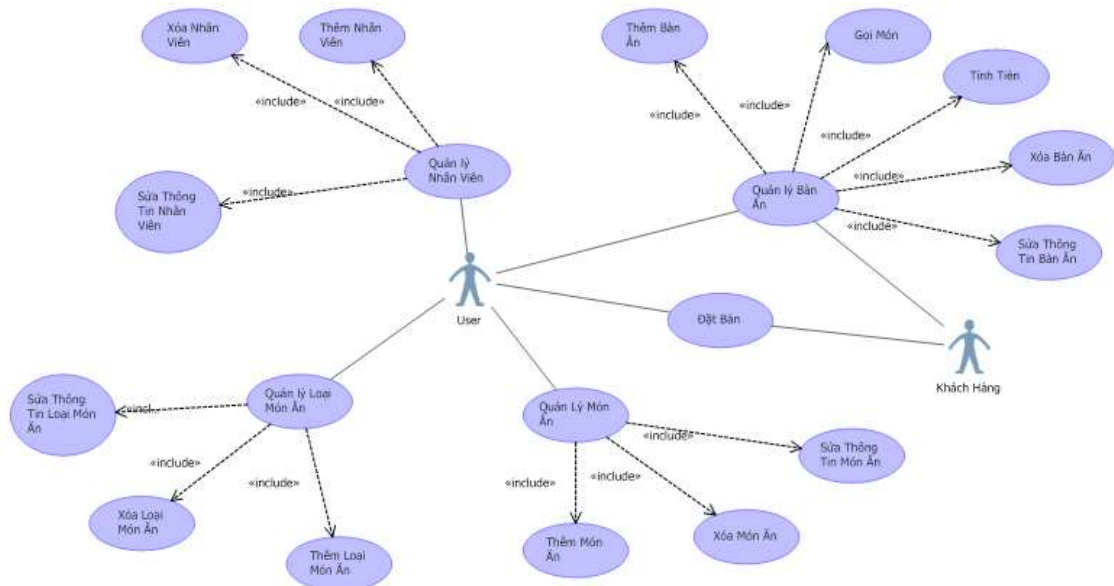
- SQLiteDatabase là lớp cơ sở để làm việc với cơ sở dữ liệu SQLite trong Android và nó cung cấp các phương thức mở, truy vấn, cập nhật và đóng cơ sở dữ liệu, ngoài ra còn cung cấp phương thức execSQL() để truy vấn trực tiếp đến cơ sở dữ liệu.
- Sử dụng đối tượng ContentValues để định nghĩa key/values. Key định danh cho cột của bảng và value là giá trị của bản ghi của cột này. **ContentValues** sử dụng để insert và cập nhật dữ liệu trong bản ghi. Ngoài ra để truy vấn cơ sở dữ liệu chúng ta sử dụng phương thức.rawQuery() và query() thông qua lớp SQLiteQueryBuilder.
- SQLiteQueryBuilder là lớp rất thuận tiện giúp truy vấn cơ sở dữ liệu.
- .rawQuery() chấp nhận một câu lệnh lựa chọn như một đầu vào.

CHƯƠNG 3: THIẾT KẾ VÀ CÀI ĐẶT:

3.1 Thiết kế:

3.1.1 Thiết kế chức năng:

- **Sơ đồ UseCase:** tổng quát chức năng của ứng dụng “Order Food”:



➤ **Danh sách các Actor:**

STT	Tên Actor	Ý nghĩa/Ghi chú
1	User	Người sử dụng ứng dụng, bao gồm nhân viên và chủ nhà hàng.
2	Khách hàng	Người có nhu cầu ăn uống

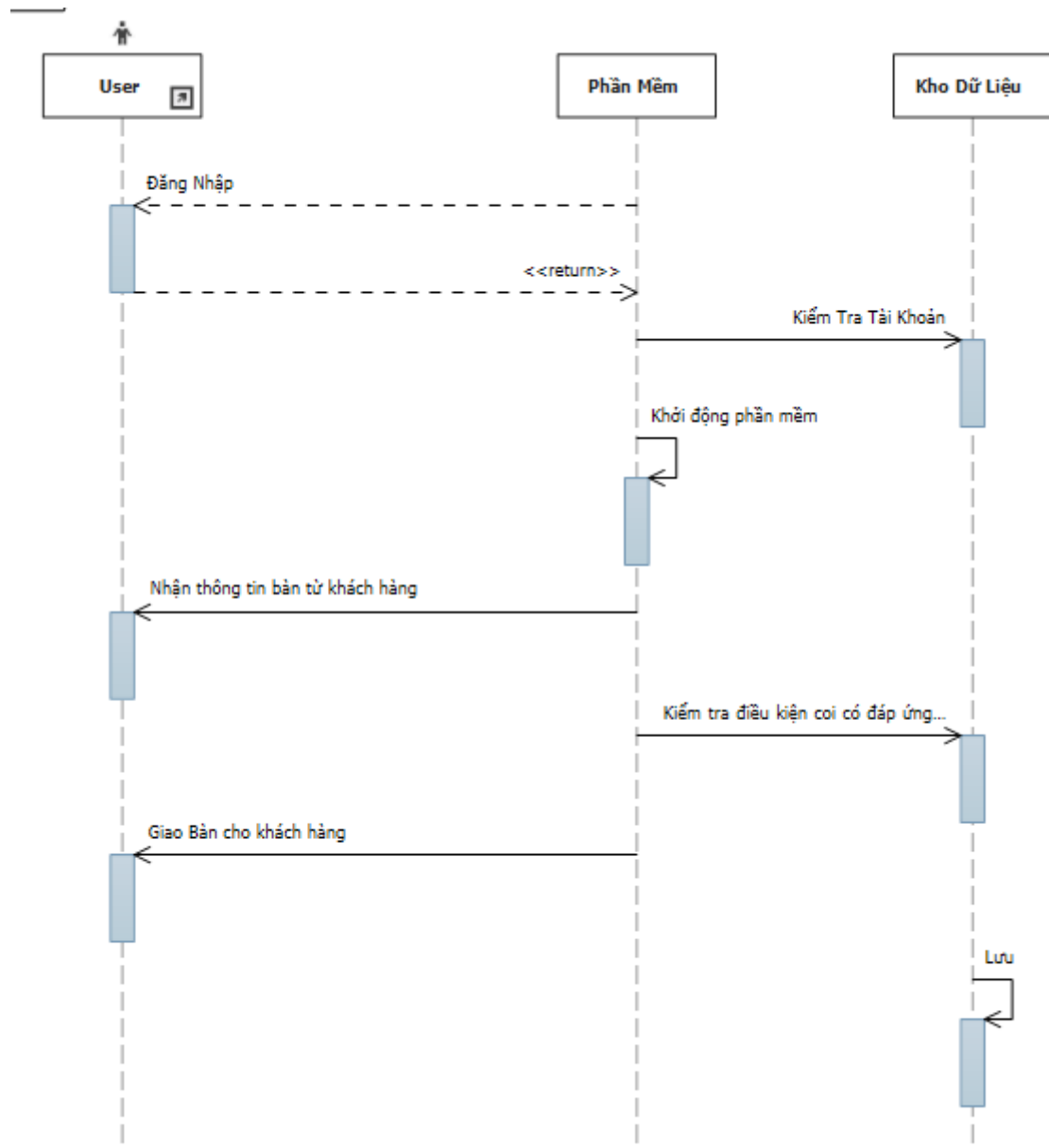
➤ **Danh sách các UseCase:**

STT	Tên Use-case	Ý nghĩa/Ghi chú
1	Đặt Bàn	Khách hàng vào nhà hàng và gọi bàn.
2	Quản lý Nhân Viên	Quản lý người dùng
3	Quản lý Bàn Ăn	Quản lý bàn ăn có trong nhà hàng
4	Quản Lý Loại Món Ăn	Quản lý các loại món ăn mà nhà hàng có
5	Quản lý Món Ăn	Quản lý món ăn của nhà hàng

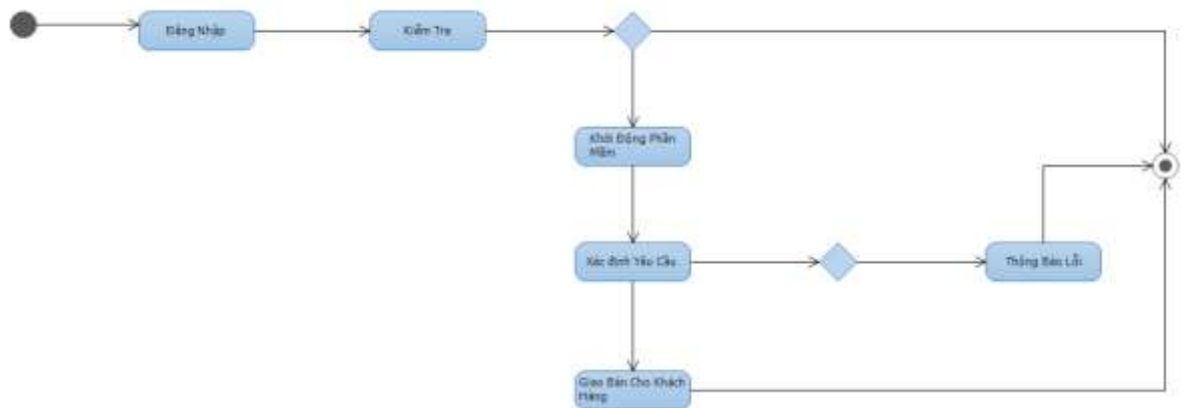
3.1.2 Đặc tả UseCase:**a. Đặc tả Use-case “Đặt Bàn”**

- Tóm tắt
 - User và Khách hàng sử dụng Use-case.
 - Use-case thực hiện chức năng chính là đặt bàn cho Khách hàng.
- Dòng sự kiện chính
 - Nhận thông tin bàn khách hàng đặt.
 - Kiểm tra trong kho dữ liệu coi có đáp ứng được yêu cầu của khách hàng.
 - Giao bàn cho khách hàng.
 - Thay đổi tình trạng bàn mà khách hàng đã đặt.
 - Lưu thông tin trên CSDL.
- Dòng sự kiện phụ
 - Nếu không đáp ứng được yêu cầu của khách hàng thì thông báo cho khách hàng.

- Tiền điều kiện
 - Có thông tin bàn trong kho dữ liệu.
- Hậu điều kiện
 - Thực hiện Use-case thành công.
- Sequence Diagram:



▪ Activity Diagram:



b. Đặc tả Use-case “Quản lý Nhân Viên”

▪ Tóm tắt

- Chủ nhà hàng sử dụng Use-case.
- Use-case thực hiện chức năng chính là Quản lý nhân viên.

▪ Dòng sự kiện chính

- Thông tin tài khoản: tên user, tên đăng nhập, password...
- Kiểm tra đã tồn tại tài khoản trong cơ sở dữ liệu hay chưa.
- Thêm tài khoản vào dữ liệu phần mềm.
- Cấp tài khoản cho người sử dụng.

▪ Dòng sự kiện phụ

▪ Tiền điều kiện

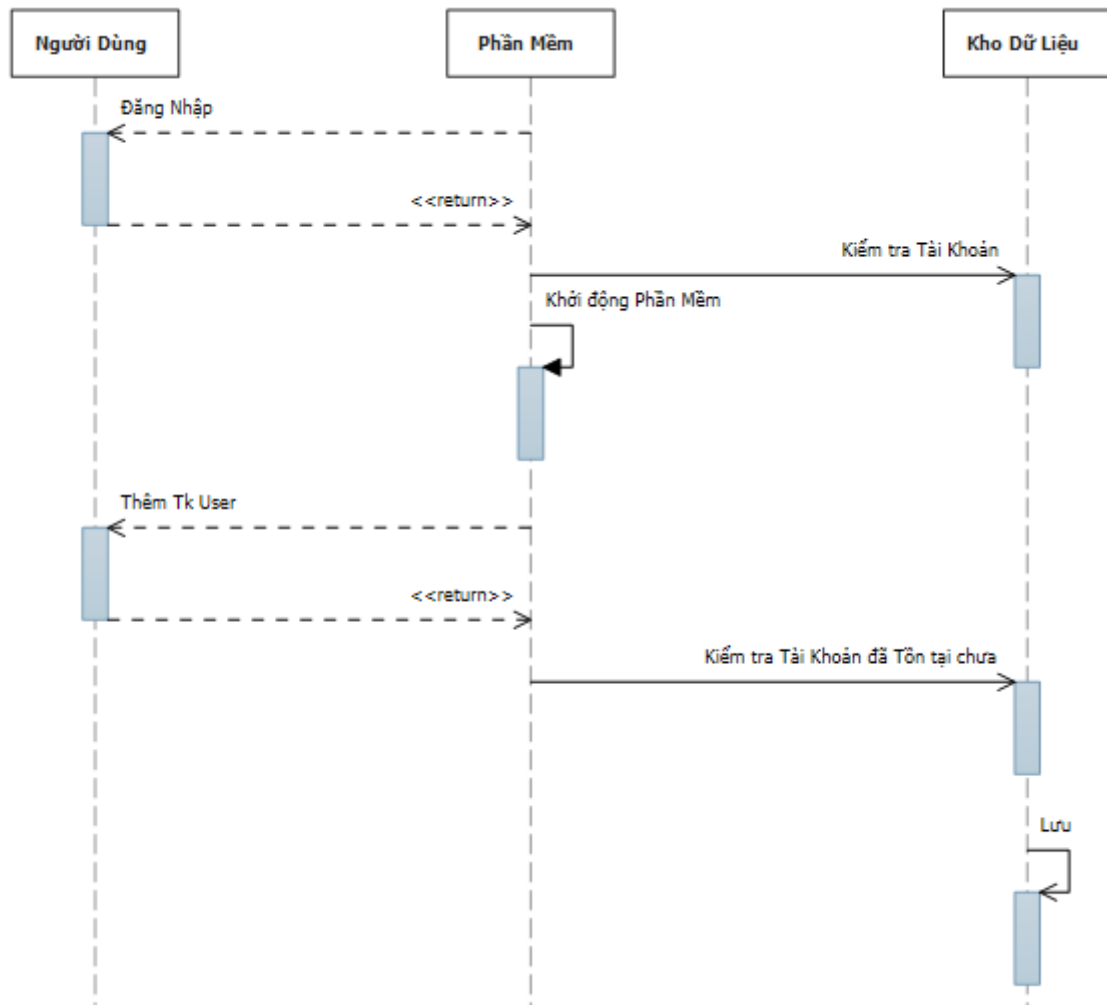
- Tải được dữ liệu từ cơ sở dữ liệu.

▪ Hậu điều kiện

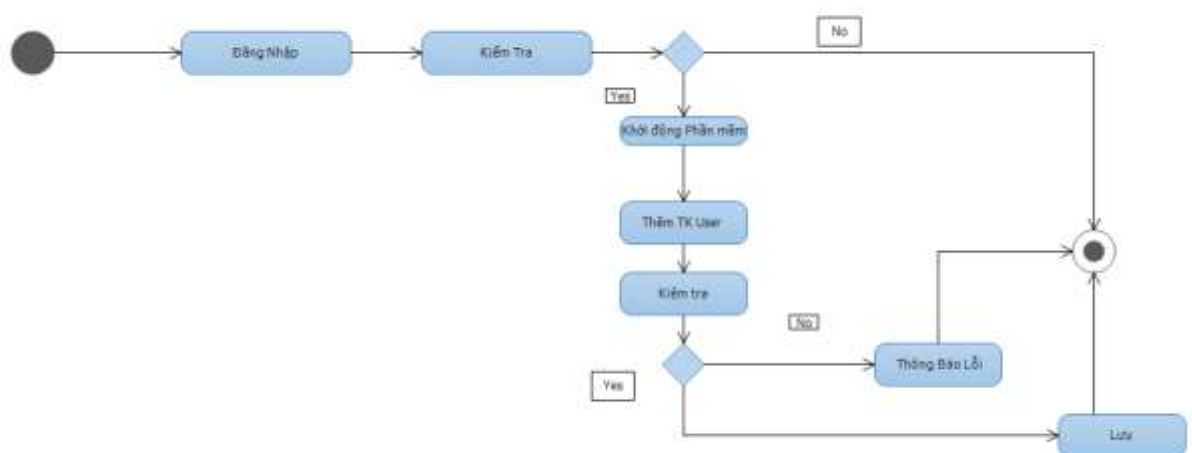
- Lưu được vào cơ sở dữ liệu.
- Thực hiện Use-case thành công.

Thêm User

➤ Sequence Diagram

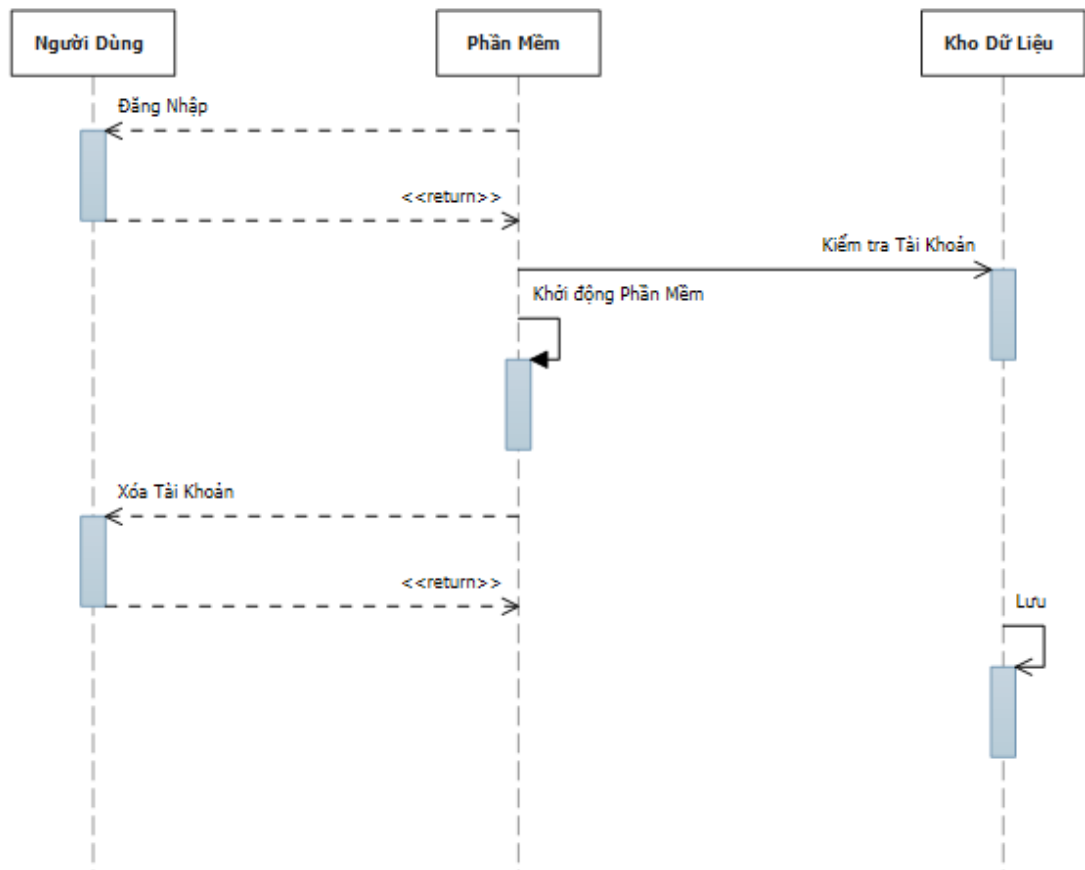


➤ Activity Diagram

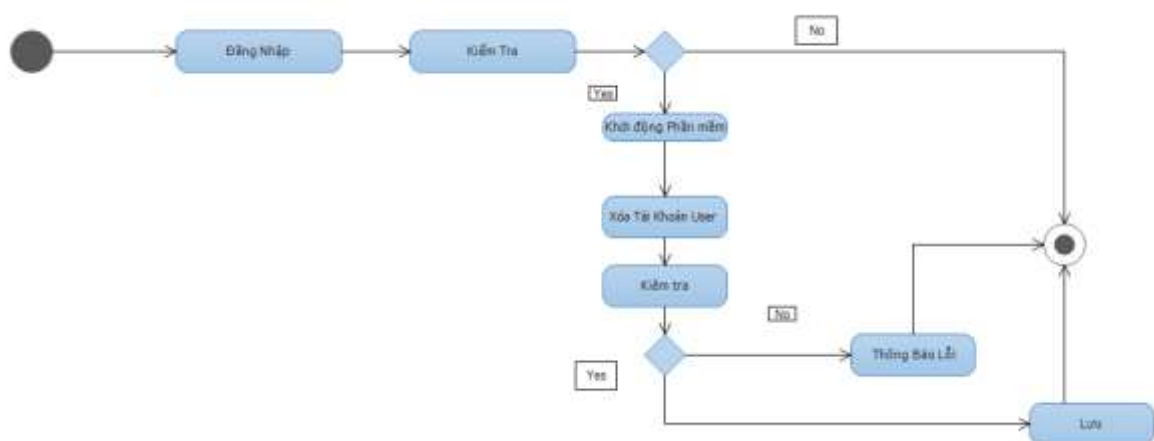


Xóa User

➤ Sequence Diagram

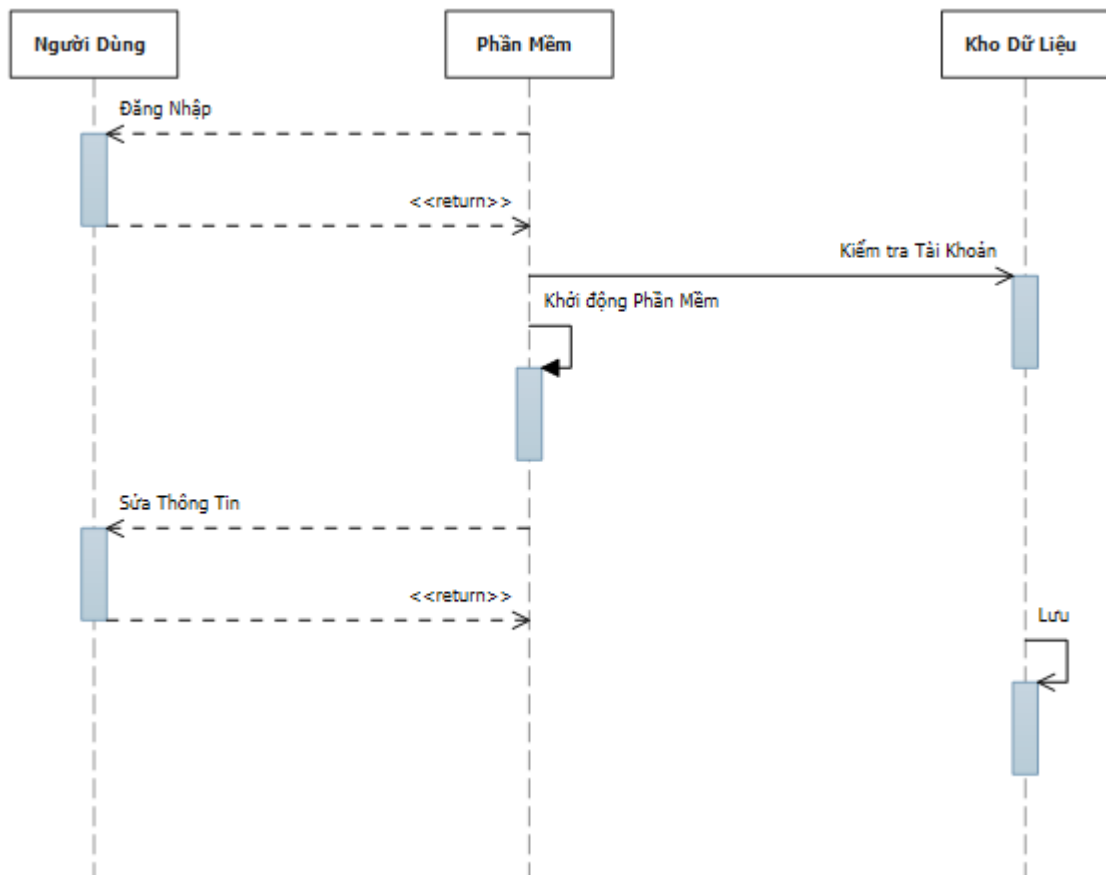


➤ Activity Diagram

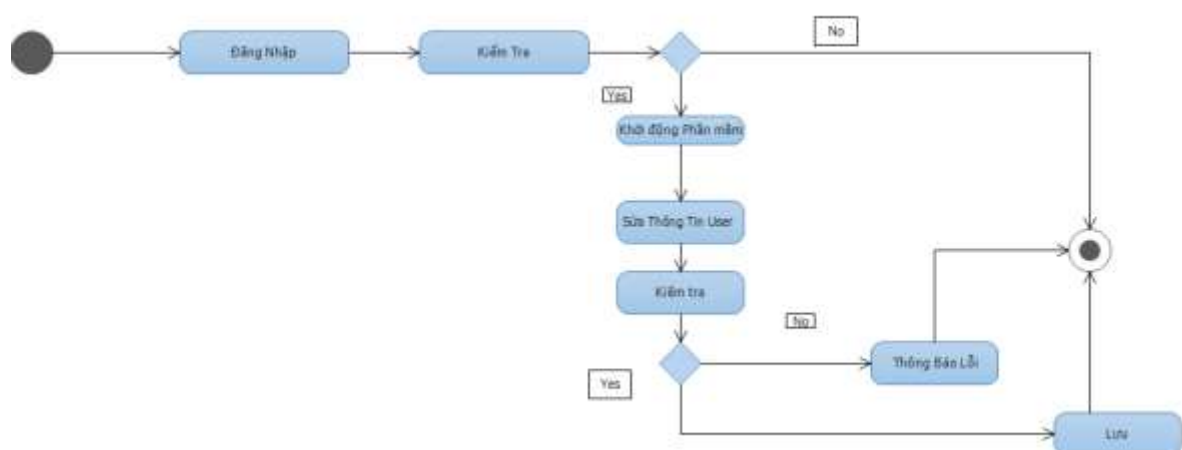


Cập nhật User

➤ Sequence Diagram



➤ Activity Diagram



c. Đặc tả Use-case “Quản lý Loại Món Ăn”**❖ Thêm Loại Món Ăn****➤ Tóm tắt**

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là Thêm Loại Món Ăn.

➤ Dòng sự kiện chính

- Nhận thông tin Loại Món Ăn.
- Nhập thông tin Loại Món Ăn.
- Kiểm tra thông tin.
- Lưu thông tin lại.
- Xem thông tin loại món ăn.

➤ Dòng sự kiện phụ

- Nếu đã tồn tại loại món ăn trong cơ sở dữ liệu thì không cho nhập.

➤ Tiên điều kiện

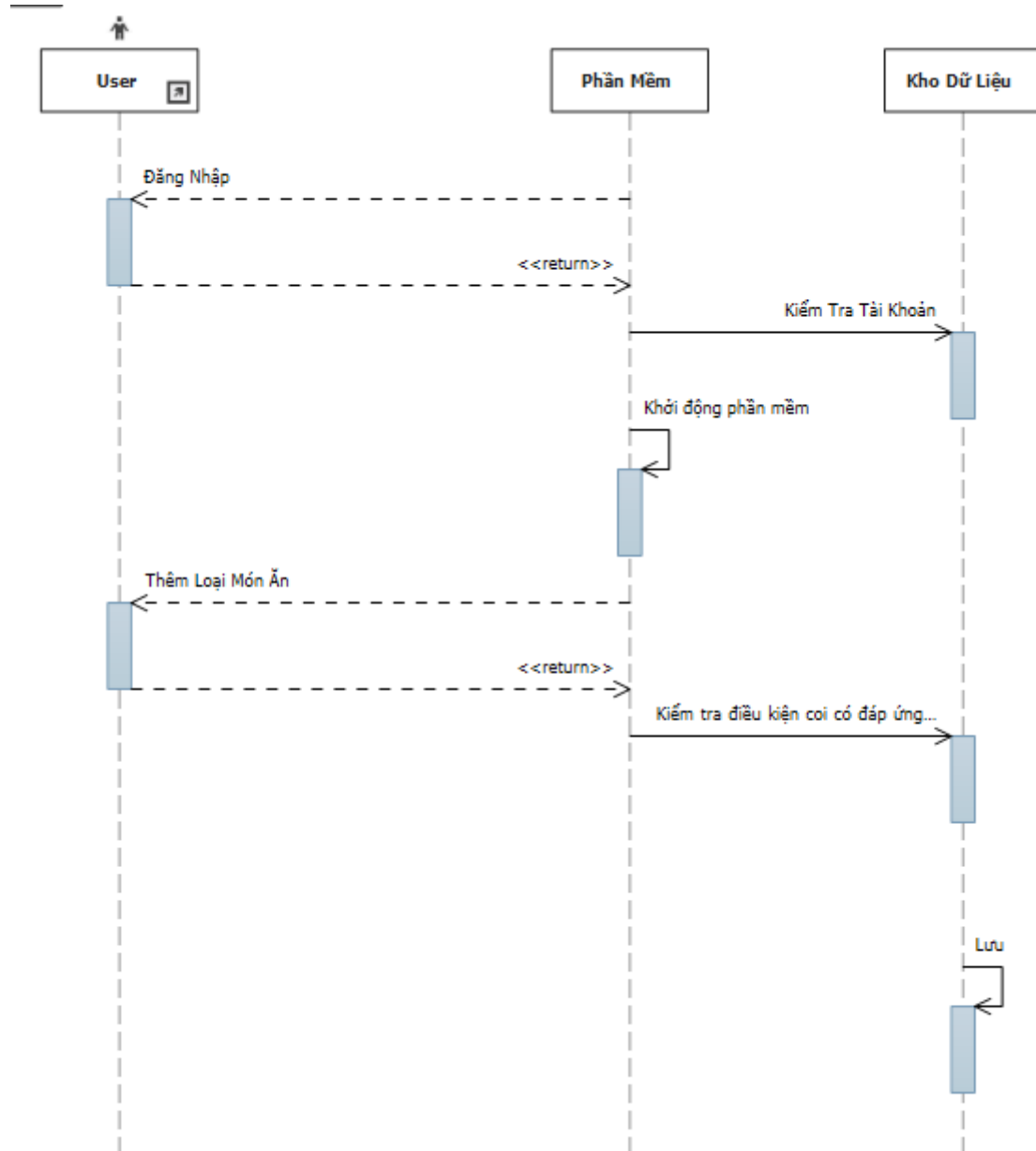
- Thông tin loại món ăn chưa có trong cơ sở dữ liệu.

➤ Hậu điều kiện

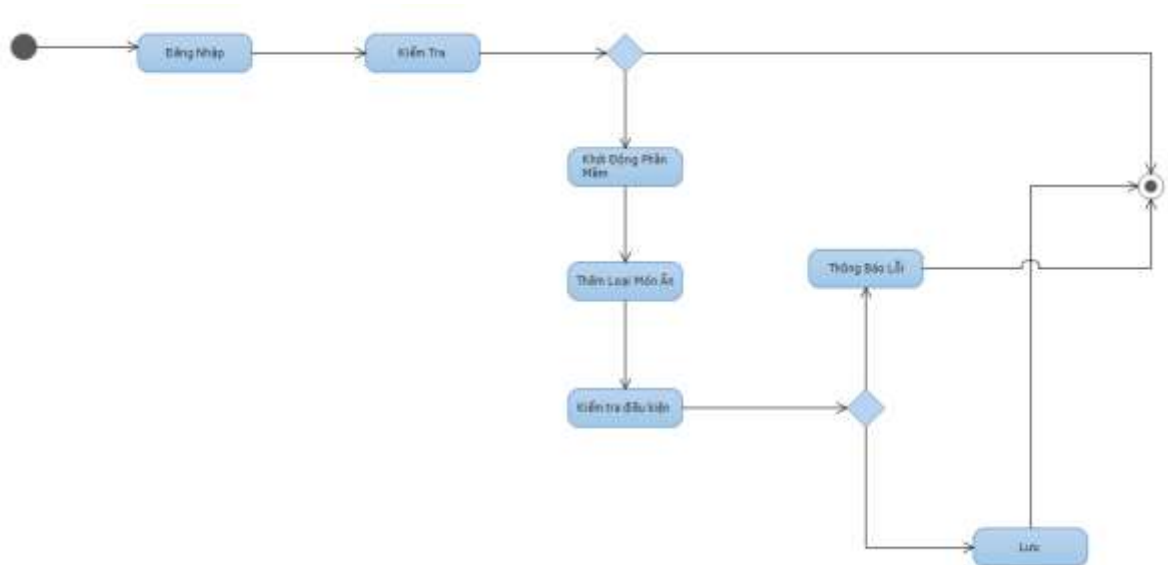
- Hệ thống lưu được thông tin loại món ăn xuống cơ sở dữ liệu.
- Thực hiện Use-case thành công.

Thêm món ăn

➤ Sequence Diagram



➤ Activity Diagram



❖ Xóa Loại Món Ăn

➤ Tóm tắt

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là Xóa Loại Món Ăn.

➤ Dòng sự kiện chính

- Xác định thông tin loại Món Ăn cần xóa.
- Xóa Loại Món Ăn.
- Cập nhật lại cơ sở dữ liệu.

➤ Dòng sự kiện phụ

- Nếu không thể xóa được loại món ăn cần kiểm tra lại hệ thống.

➤ Tiền điều kiện

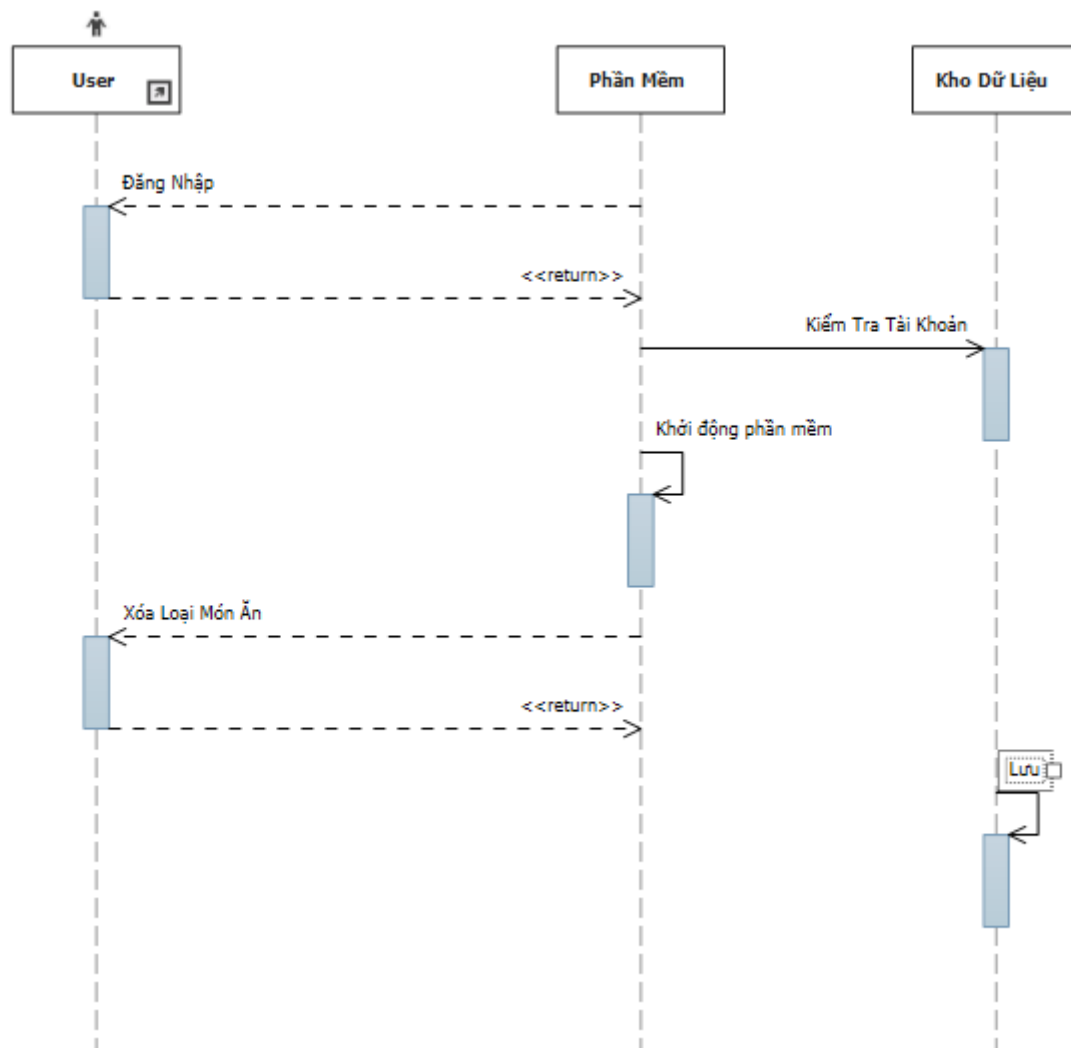
- Tồn tại thông tin loại món ăn trong cơ sở dữ liệu.

➤ Hậu điều kiện

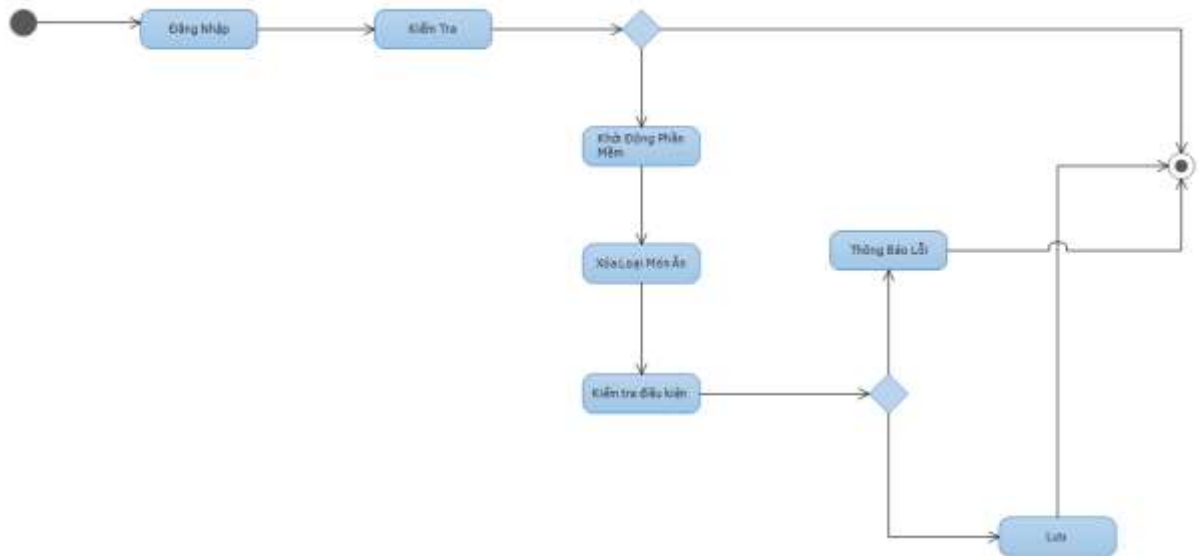
- Hệ thống cập nhật lại thông tin loại món ăn.
- Thực hiện Use-case thành công.

Xóa Loại Món Ăn

➤ Sequence Diagram



➤ Activity Diagram

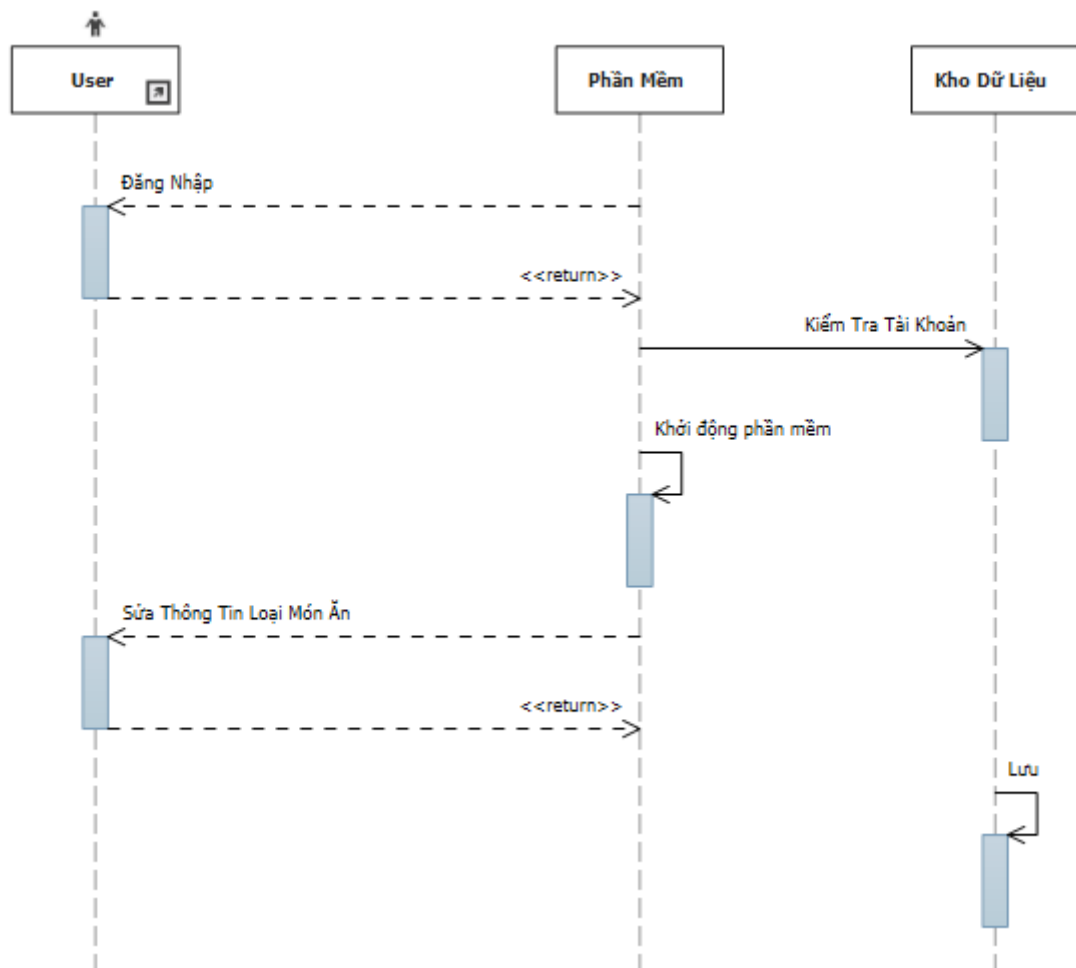


❖ Cập nhật Loại Món Ăn

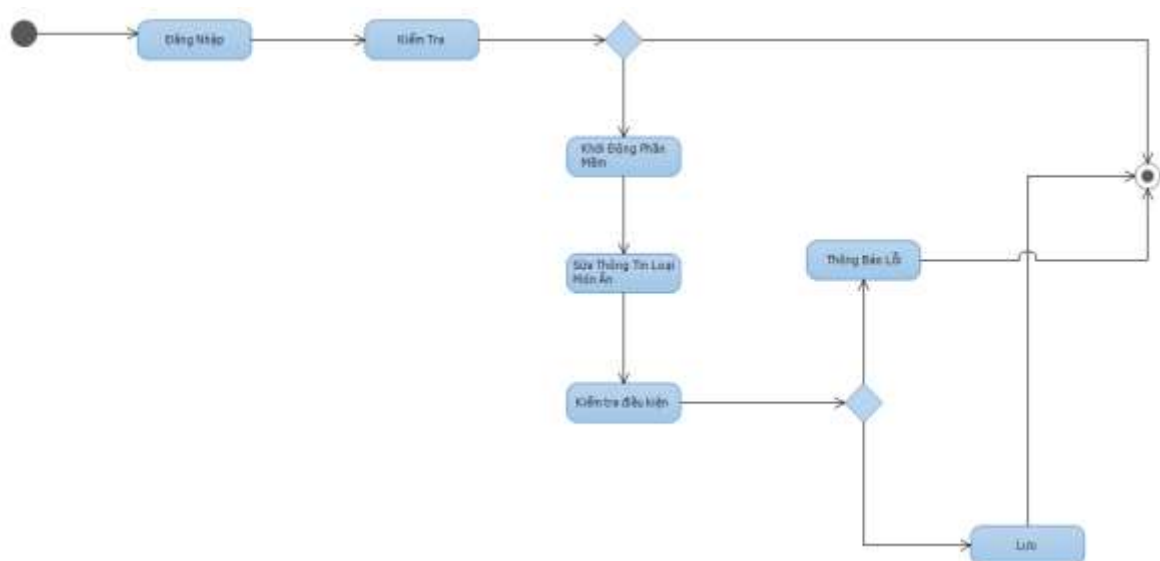
- Tóm tắt
 - User sử dụng Use-case.
 - Use-case thực hiện chức năng chính là Cập nhật thông tin loại món ăn.
- Dòng sự kiện chính
 - Xác định thông tin loại món ăn cần cập nhật.
 - Cập nhật lại thông tin Loại Món ăn.
 - Lưu xuống cơ sở dữ liệu.
- Dòng sự kiện phụ
 - Nếu không cập nhật được thông tin Loại Món Ăn cần kiểm tra lại hệ thống.
- Tiên điều kiện
 - Tồn tại thông tin loại món ăn trong cơ sở dữ liệu.
- Hậu điều kiện
 - Hệ thống cập nhật lại thông tin loại món ăn đã chỉnh sửa.
 - Thực hiện Use-case thành công.

Cập nhật Loại Món Ăn

➤ Sequence Diagram



➤ Activity Diagram



d. Đặc tả Use-case “Quản lý Món ăn ”**❖ Thêm Món Ăn****➤ Tóm tắt**

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là Quản lý sản phẩm.

➤ Dòng sự kiện chính

- Nhập thông tin món ăn: tên , đơn giá, ...
- Kiểm tra món ăn có trong CSDL chưa.
- Lưu thông tin món ăn.
- Xem thông tin món ăn.

➤ Dòng sự kiện phụ

- Nếu đã tồn tại món ăn trong cơ sở dữ liệu thì không cho nhập.

➤ Tiên điều kiện

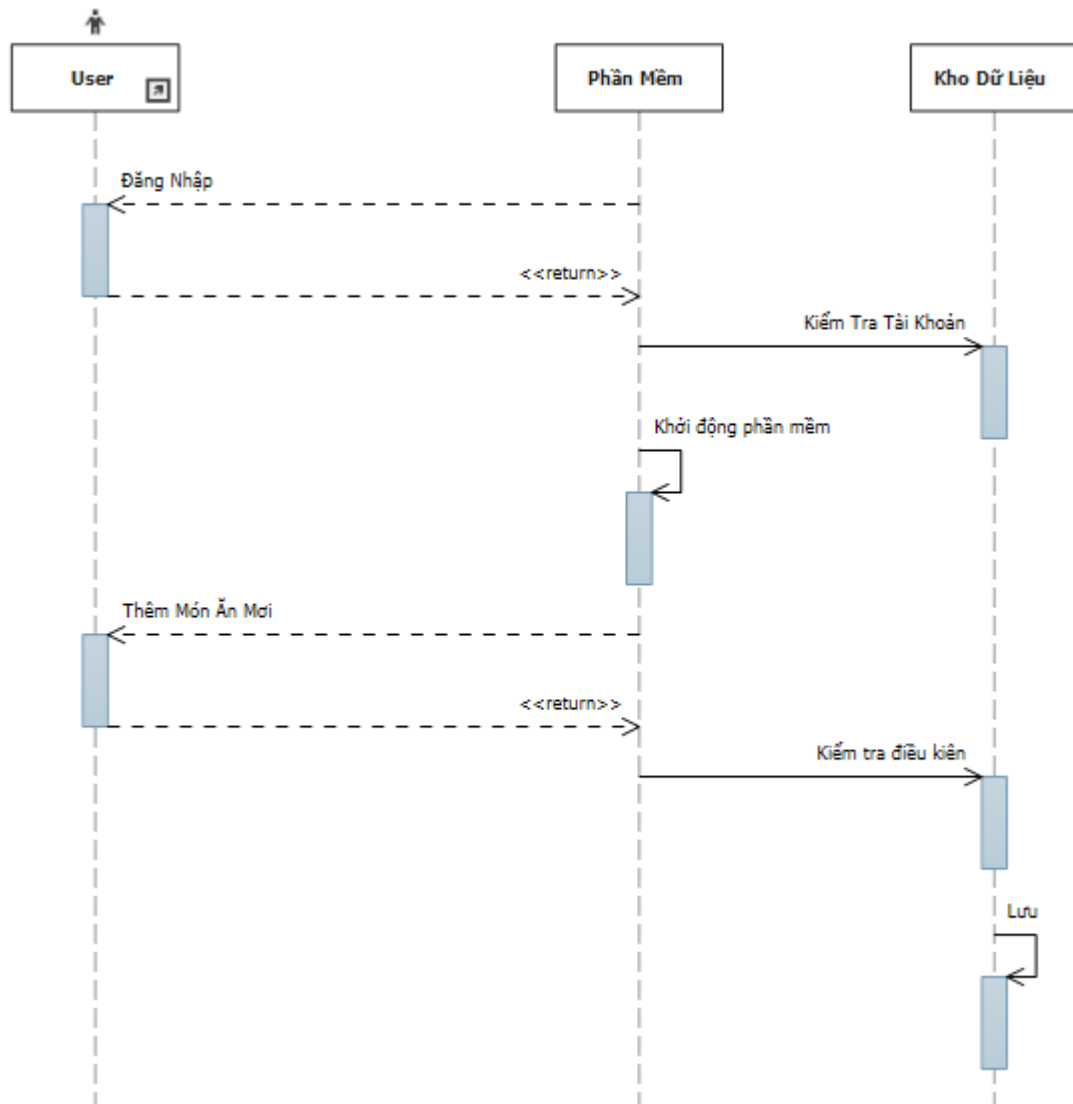
- Thông tin món ăn chưa có trong cơ sở dữ liệu.

➤ Hậu điều kiện

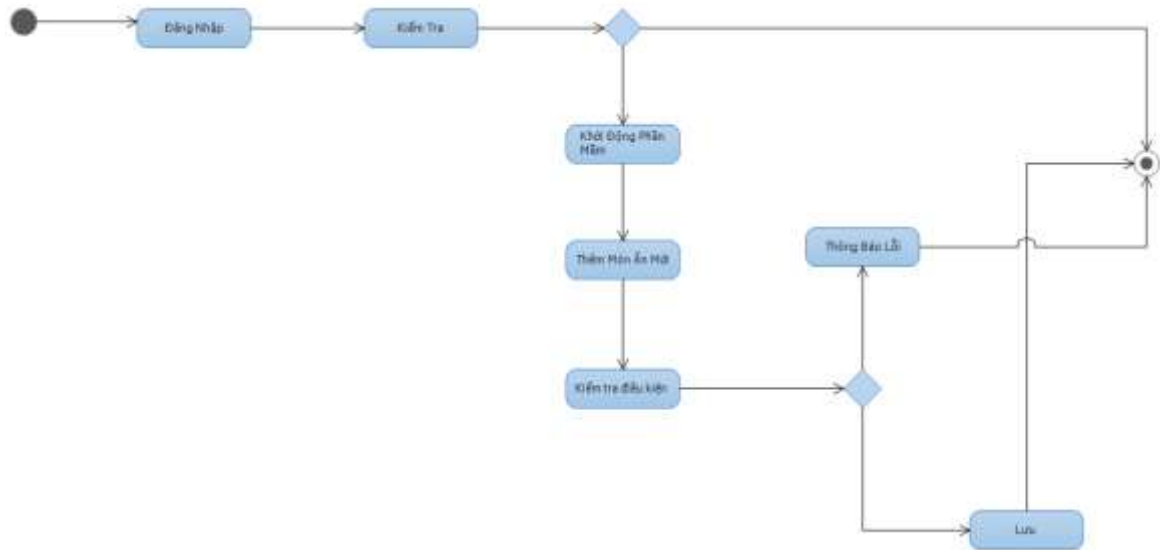
- Hệ thống lưu được thông tin món ăn xuống cơ sở dữ liệu.
- Thực hiện Use-case thành công.

Thêm Món Ăn

➤ Sequence Diagram



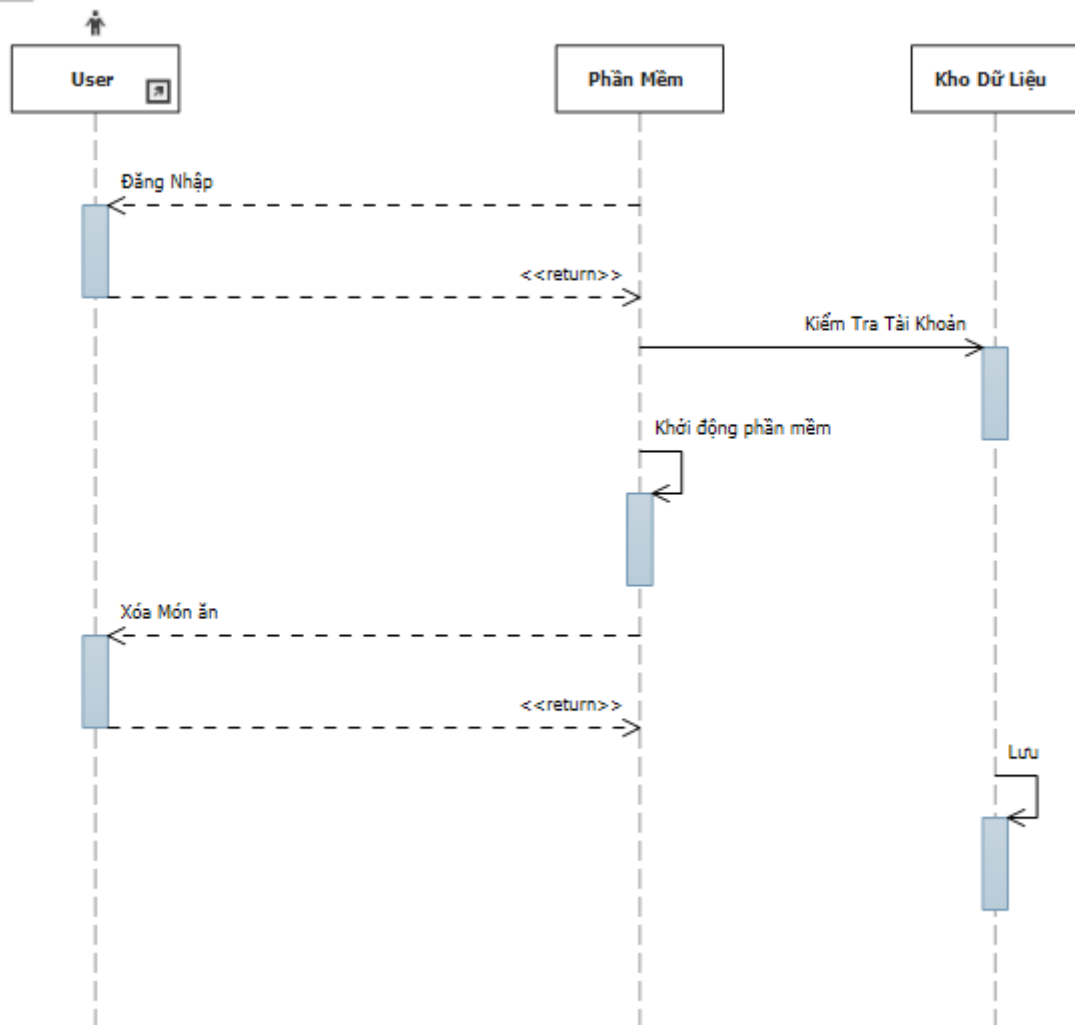
➤ Activity Diagram



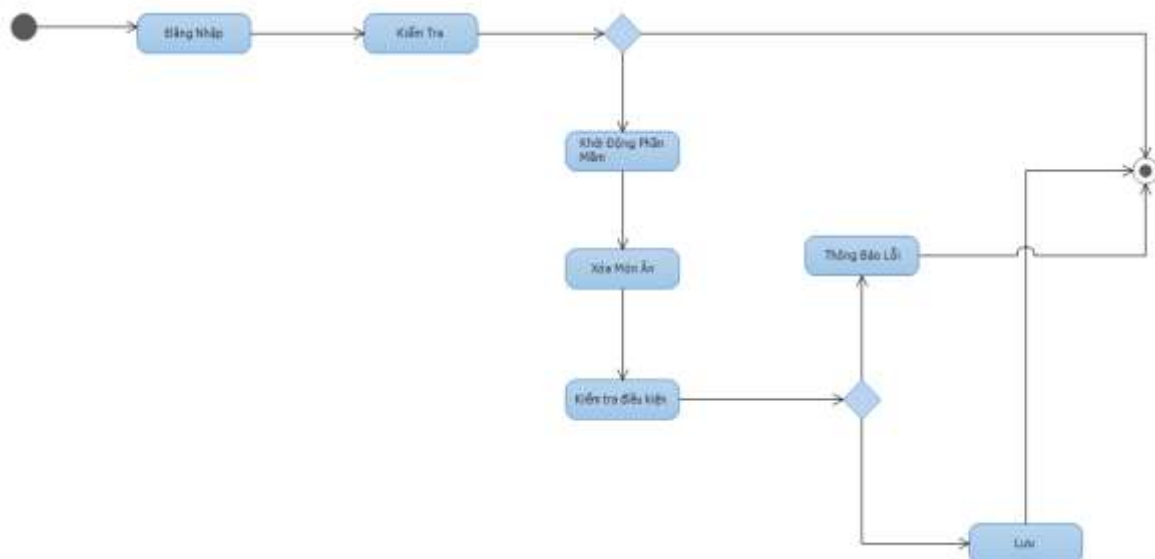
- ❖ Xóa Món ăn
 - Tóm tắt
 - User sử dụng Use-case.
 - Use-case thực hiện chức năng chính là Xóa món ăn.
 - Dòng sự kiện chính
 - Xác định thông tin món ăn cần xóa.
 - Xóa món ăn.
 - Cập nhật lại cơ sở dữ liệu.
 - Dòng dự kiện phụ
 - Nếu không thể xóa được món ăn cần kiểm tra lại hệ thống.
 - Tiên điều kiện
 - Tồn tại thông tin món ăn trong cơ sở dữ liệu.
 - Hậu điều kiện
 - Hệ thống cập nhật lại thông tin món ăn.
 - Thực hiện Use-case thành công.

Xóa Món ăn

➤ Sequence Diagram



➤ Activity Diagram



❖ Sửa thông tin món ăn.

➤ Tóm tắt

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là sửa thông tin món ăn.

➤ Dòng sự kiện chính

- Xác định thông tin món ăn cần sửa.
- Cập nhật lại thông tin món ăn.
- Lưu xuống cơ sở dữ liệu.

➤ Dòng sự kiện phụ

- Nếu không cập nhật được thông tin món ăn cần kiểm tra lại hệ thống.

➤ Tiên điều kiện

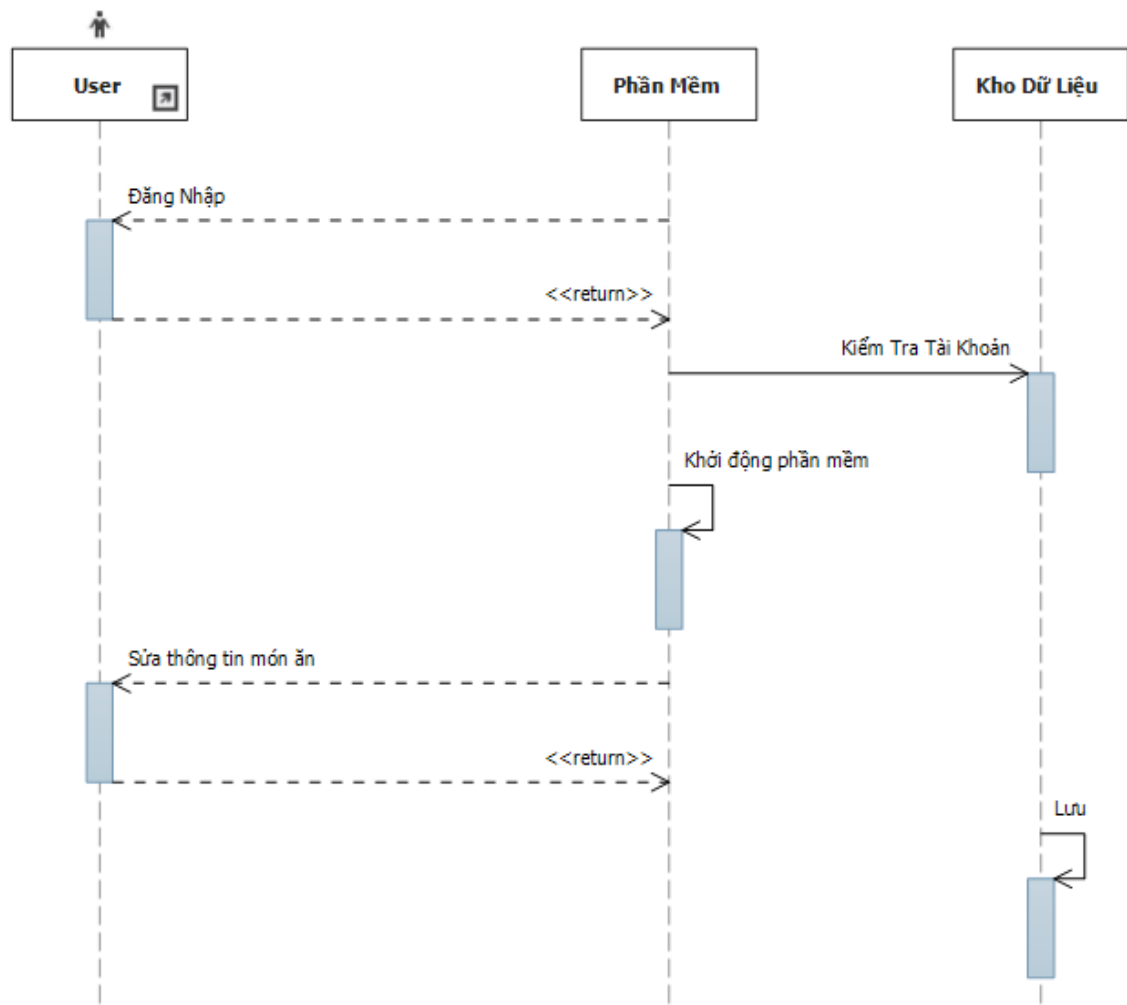
- Tồn tại thông tin món ăn trong cơ sở dữ liệu.

➤ Hậu điều kiện

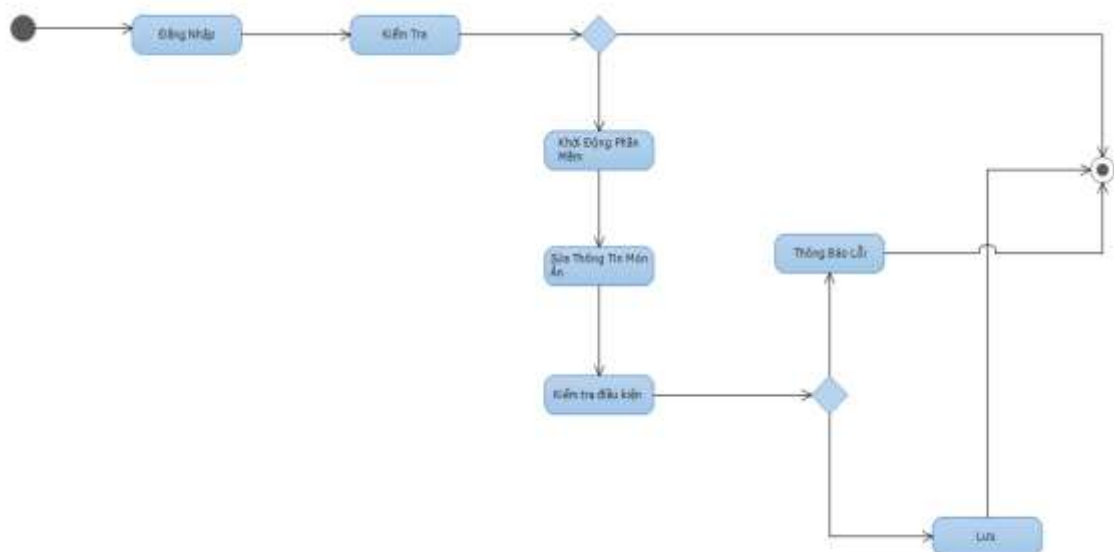
- Hệ thống cập nhật lại thông tin món ăn đã chỉnh sửa.
- Thực hiện Use-case thành công.

Sửa thông tin món ăn.

➤ Sequence Diagram



➤ Activity Diagram

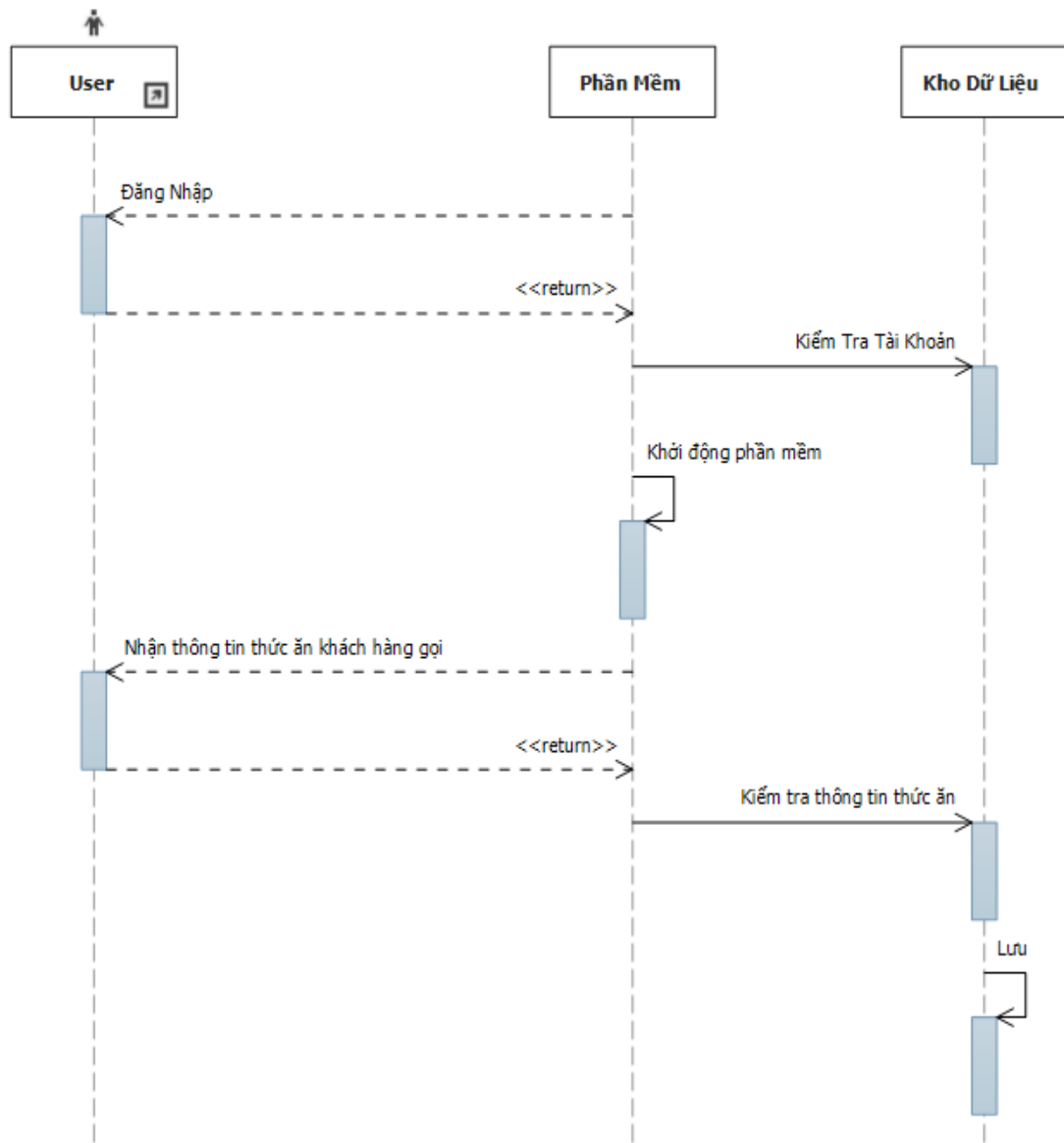


e. Đặc tả Use-case “Quản lý Bàn”**❖ Gọi Món**

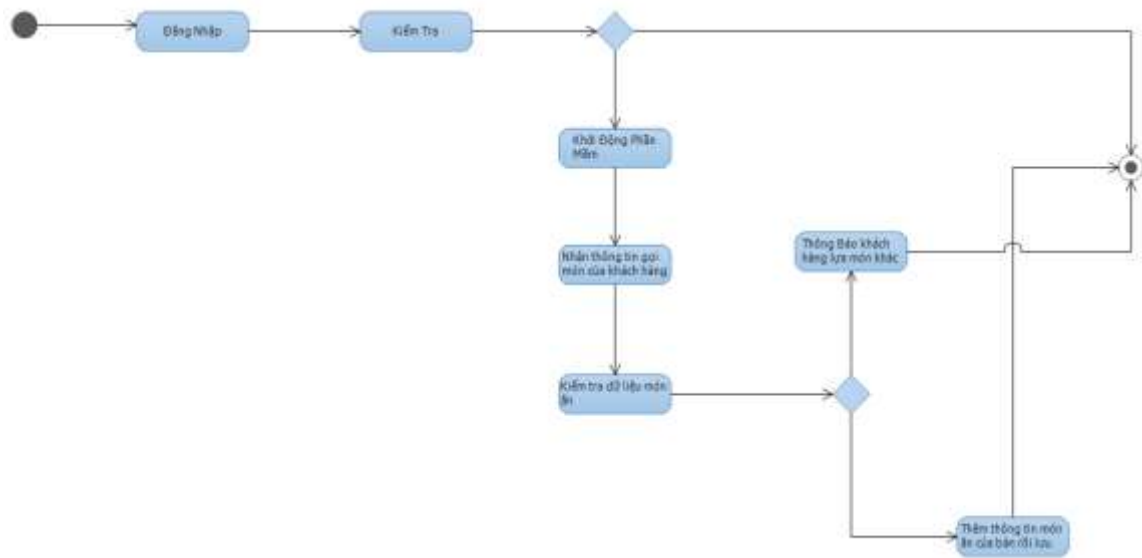
- Tóm tắt
 - User và khách Hàng sử dụng Use-case.
 - Use-case thực hiện chức năng chính là gọi món.
- Dòng sự kiện chính
 - Nhận thông tin món ăn cần gọi từ khách hàng.
 - Nhập thông tin món ăn cần gọi.
 - Kiểm tra thông tin thức ăn cần gọi.
 - Lưu thông tin thức ăn được gọi vào bàn tương ứng.
 - Xem thông tin thức ăn của bàn.
- Dòng sự kiện phụ
- Tiền điều kiện
 - Bàn phải có khách hàng sử dụng.
- Hậu điều kiện
 - Hệ thống lưu được thông tin món ăn của bàn xuống cơ sở dữ liệu.
 - Thực hiện Use-case thành công.

Gọi Món

➤ Sequence Diagram



➤ Activity Diagram



❖ Tính Tiền

➤ Tóm tắt

- User và khách Hàng sử dụng Use-case.
- Use-case thực hiện chức năng chính là tính tiền cho bàn ăn.

➤ Dòng sự kiện chính

- Nhận thông tin từ khách hàng.
- Kiểm tra thông tin món ăn của bàn ăn .
- Thực hiện tính tiền.
- Lưu thông tin.

➤ Dòng sự kiện phụ

➤ Tiền điều kiện

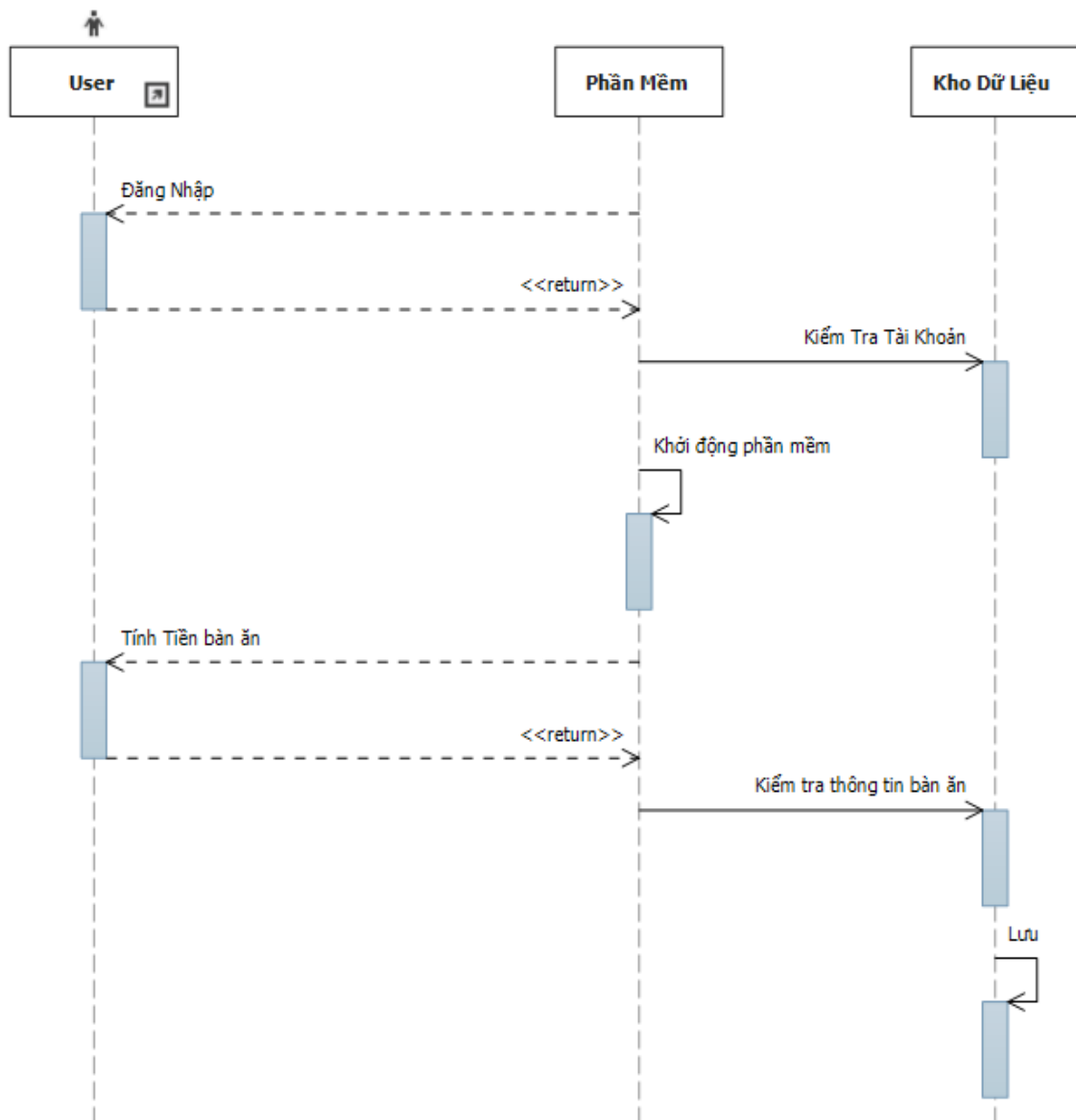
- Bàn đã có khách hàng sử dụng.

➤ Hậu điều kiện

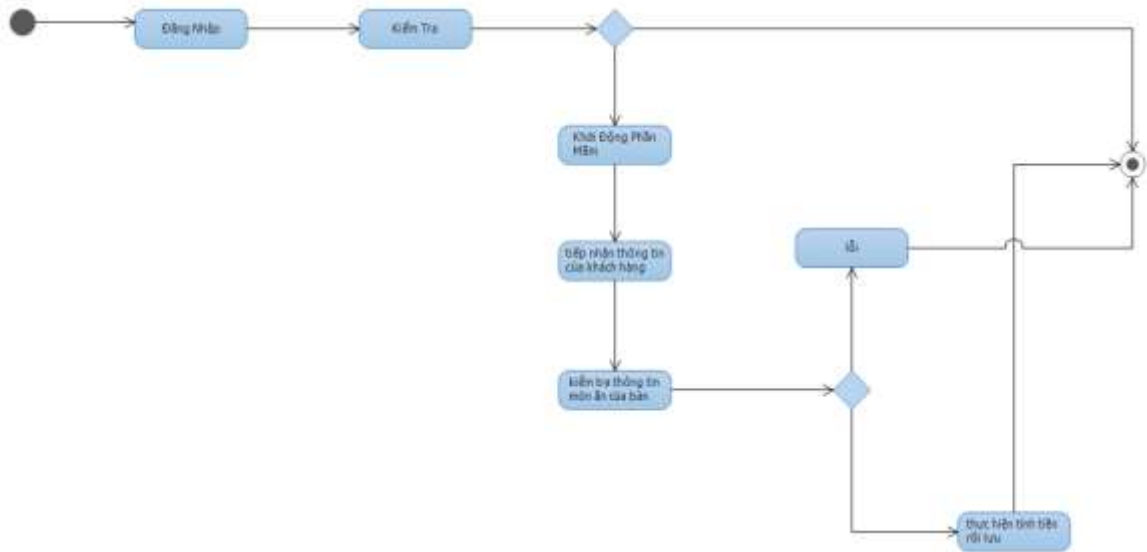
- Hệ thống lưu được thông tin.
- Thực hiện Use-case thành công.

Tính Tiền

➤ Sequence Diagram



➤ Activity Diagram



❖ Thêm Bàn

➤ Tóm tắt

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là thêm bàn.

➤ Dòng sự kiện chính

- Nhập thông tin bàn cần thêm.
- Kiểm tra thông tin bàn.
- Lưu thông tin bàn.
- Xem thông tin bàn.

➤ Dòng sự kiện phụ

- Nếu đã tồn tại bàn trong cơ sở dữ liệu thì không cho nhập.

➤ Tiên điều kiện

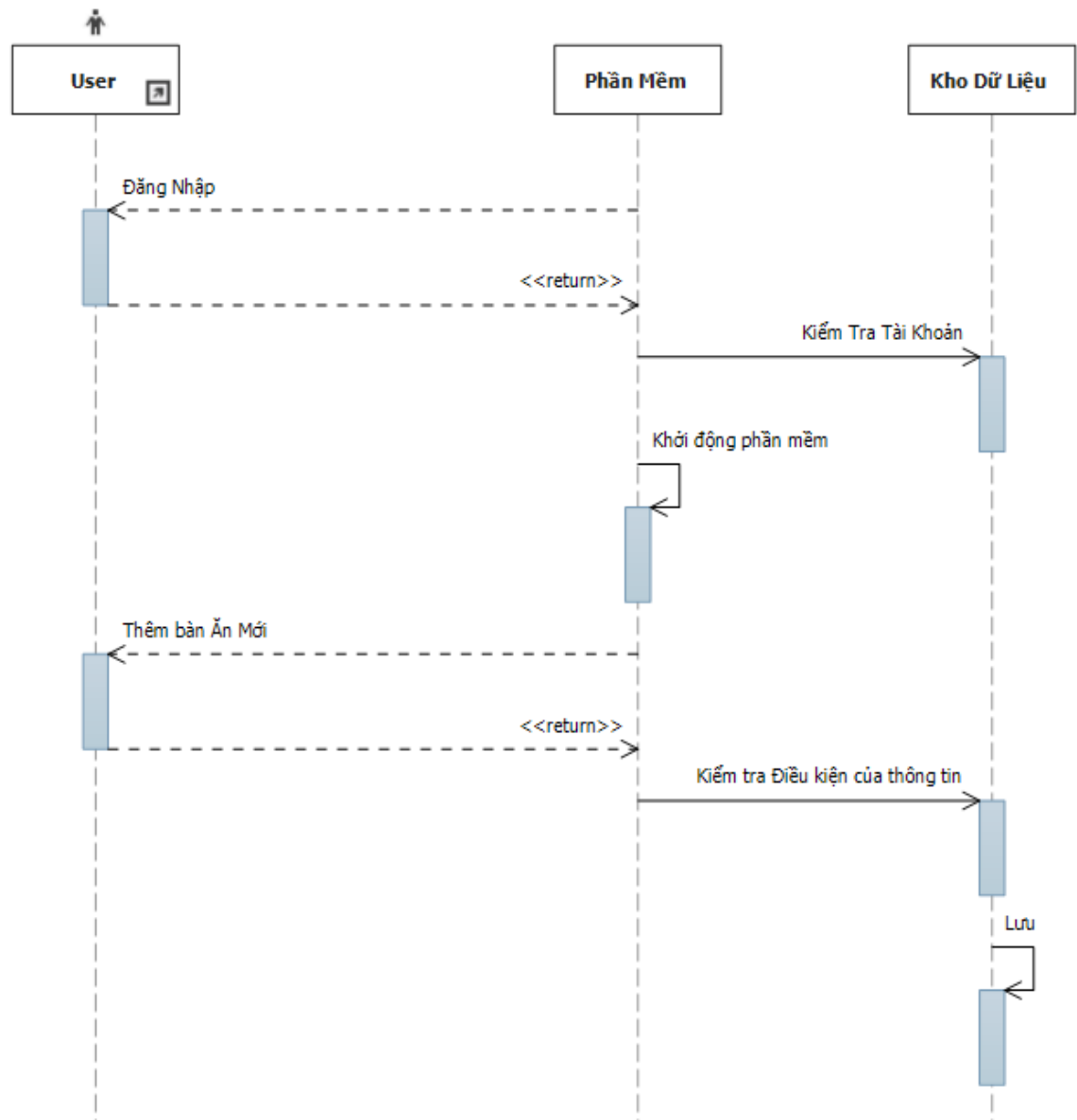
- Thông tin bàn chưa có trong cơ sở dữ liệu.

➤ Hậu điều kiện

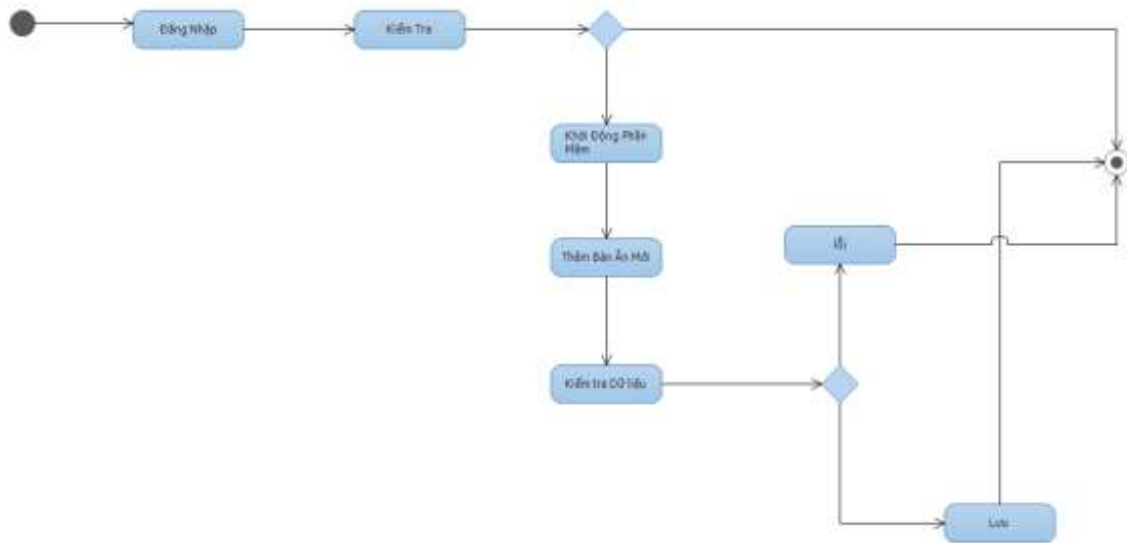
- Hệ thống lưu được thông tin bàn xuống cơ sở dữ liệu.
- Thực hiện Use-case thành công.

Thêm Bàn

➤ Sequence Diagram



➤ Activity Diagram



❖ Xóa Thông tin bàn

➤ Tóm tắt

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là Xóa bàn.

➤ Dòng sự kiện chính

- Xác định thông tin nhà bàn xóa
- Xóa thông tin bàn.
- Cập nhật lại cơ sở dữ liệu.

➤ Dòng sự kiện phụ

- Nếu không thể xóa được bàn cần kiểm tra lại hệ thống.

➤ Tiền điều kiện

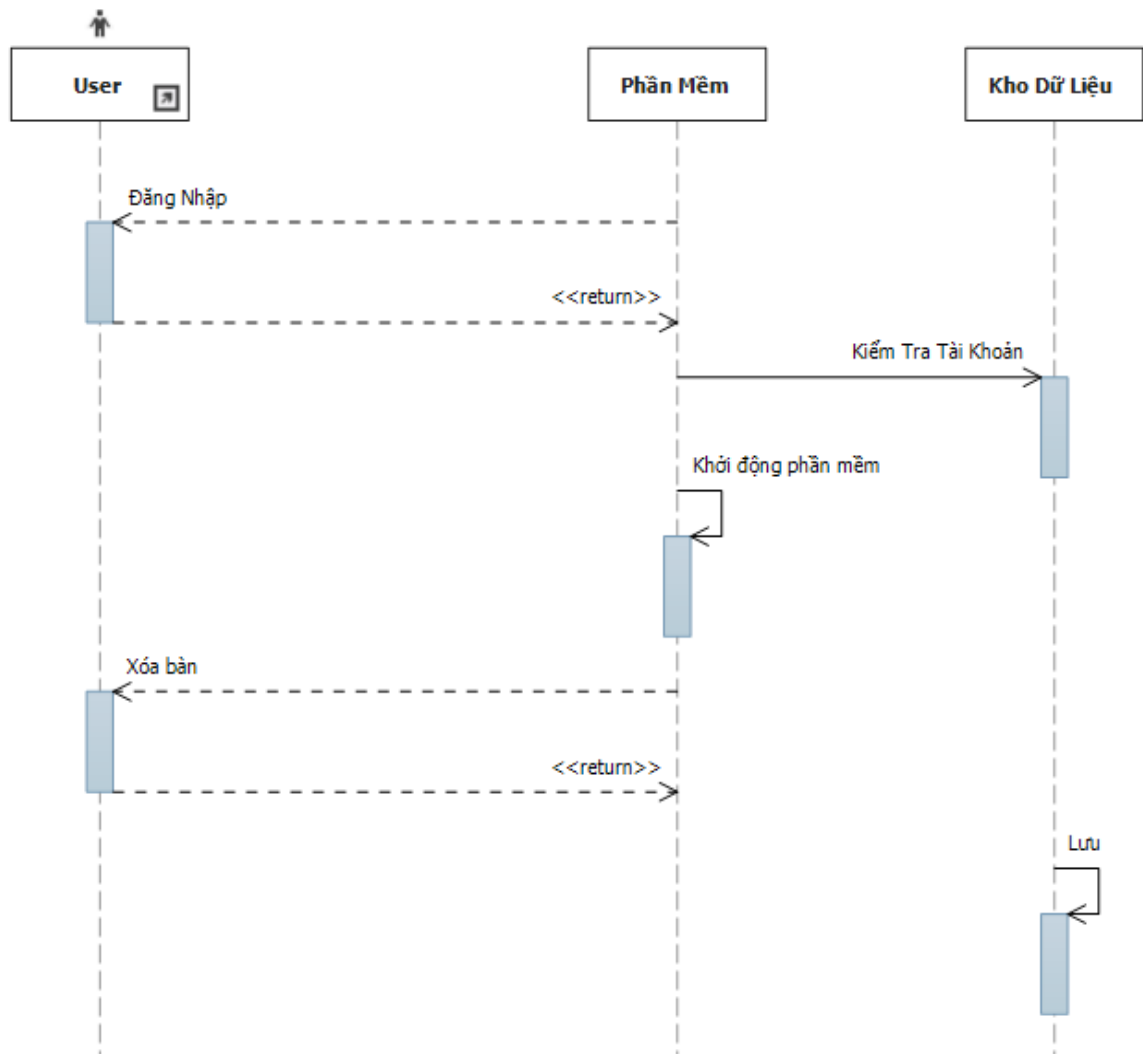
- Tồn tại thông tin bàn trong cơ sở dữ liệu.

➤ Hậu điều kiện

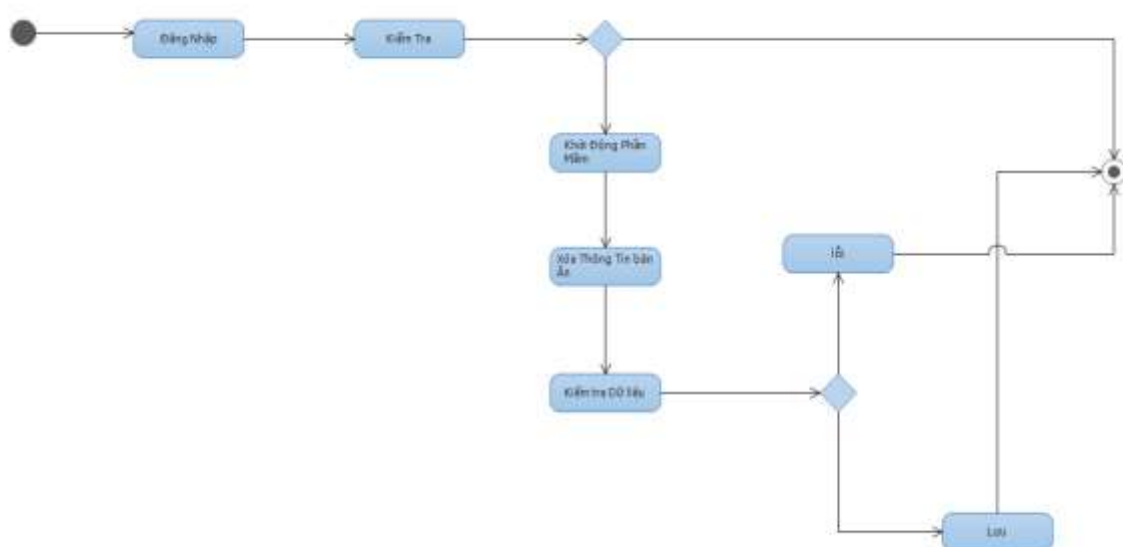
- Hệ thống cập nhật lại thông tin bàn.
- Thực hiện Use-case thành công.

Xóa Thông tin bàn

➤ Sequence Diagram



➤ Activity Diagram



❖ Sửa thông tin bàn**➤ Tóm tắt**

- User sử dụng Use-case.
- Use-case thực hiện chức năng chính là sửa bàn.

➤ Dòng sự kiện chính

- Xác định thông tin bàn cần cập nhật.
- Cập nhật lại thông tin bàn.
- Lưu xuống cơ sở dữ liệu.

➤ Dòng sự kiện phụ

- Nếu không cập nhật được thông tin bàn cần kiểm tra lại hệ thống.

➤ Tiên điều kiện

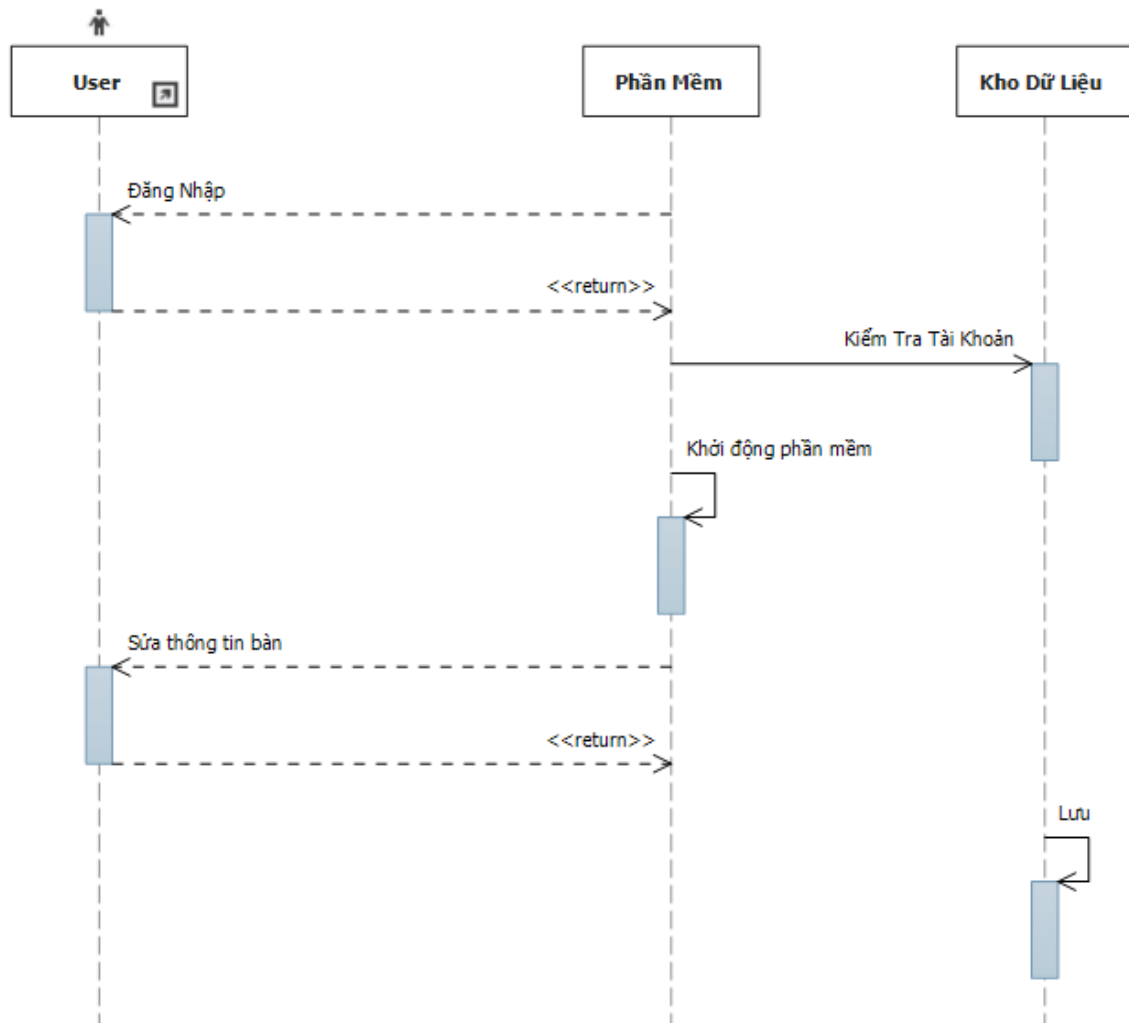
- Tồn tại thông tin bàn trong cơ sở dữ liệu.

➤ Hậu điều kiện

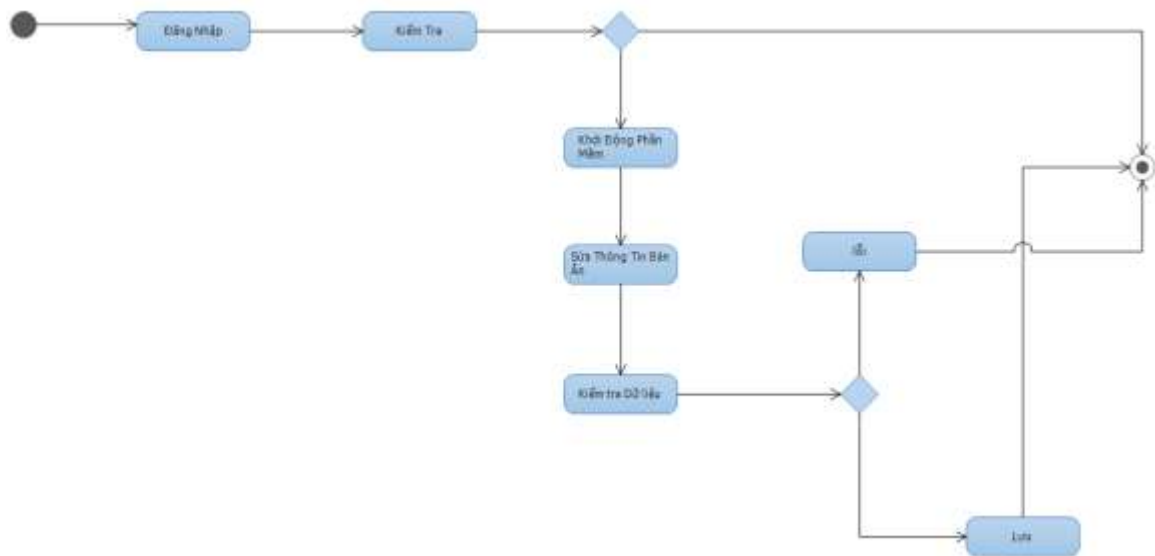
- Hệ thống cập nhật lại thông tin bàn đã chỉnh sửa.
- Thực hiện Use-case thành công.

Sửa thông tin bàn

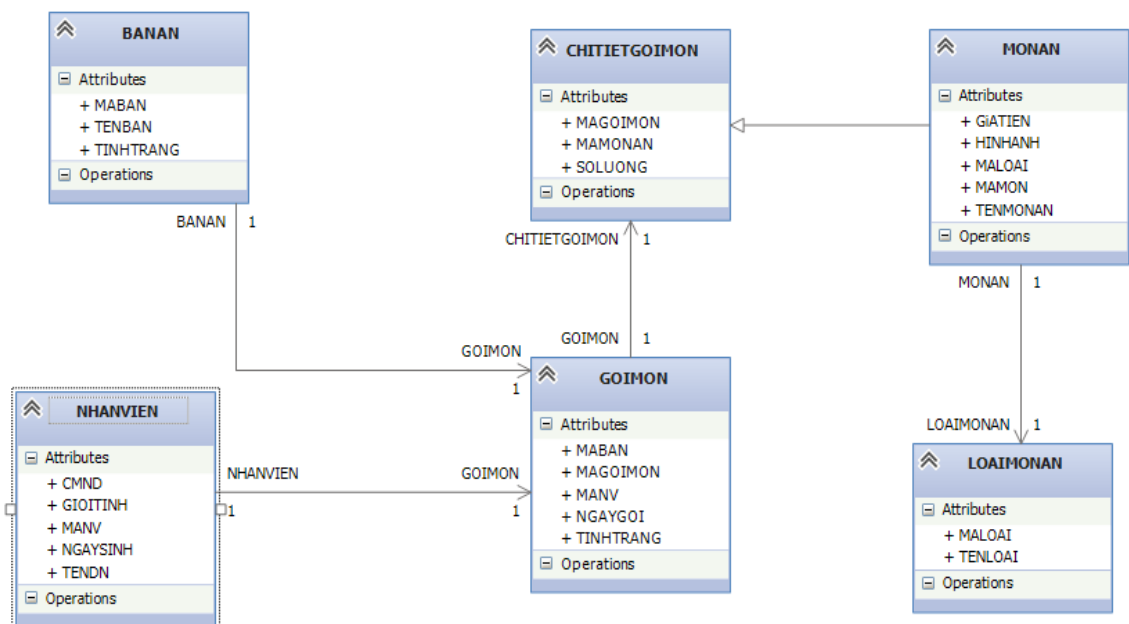
➤ Sequence Diagram



➤ Activity Diagram



3.1.3.Thiết kế cơ sở dữ liệu:



3.1.4.Thiết kế kiến trúc:

3.1.4.1.Kiến trúc hệ thống:

Sử dụng mô hình 3 lớp :Lớp DTO,Lớp DAO,Lớp Custom Adapter, FragmentApp

3.1.4.2.Mô tả chi tiết từng thành phần của hệ thống:

Lớp DTO:lớp DTO đối tượng để trao đổi dữ liệu

- BanAnDTO
- ChiTietGoiMonDTO
- GoiMonDTO
- LoaiMonAnDTO
- NhanVienDTO
- ThanhToanDTO

Lớp DAO:Chức năng của lớp này là kết nối với cơ sở dữ liệu

➤ BanAnDAO

- **public** BanAnDAO(Context context) {}
- **public boolean** ThemBanAn(String tenbanan){}
- **public** List<BanAnDTO> LayTatCaBanAn(){}
- **public** String LayTinhTrangBanTheoMa(**int** maban){}
- **public boolean** XoaBanAnTheoMa(**int** maban){}
- **public boolean** CapNhatTenBan(**int** maban, String tenban){}
- **public boolean** CapNhatTinhTrangBan(**int** maban, String tinhtrang) {}

➤ GoiMonDAO

- **public** GoiMonDAO(Context context){ }
- **public long** ThemGoiMon(GoiMonDTO goiMonDTO) { }
- **public long** LayMaGoiMonTheoBan(**int** maban,String tinhtrang) { }
- **public boolean** KiemTraMonTonTai(**int** magoimon,**int** mamonan) { }
- **public int** LaySLMonAnCu(**int** magoimon,**int** mamonan) { }
- **public boolean** CapNhatSL(ChiTietGoiMonDTO chiTietGoiMonDTO){ }
- **public** **boolean** ThemChiTietGoiMon(ChiTietGoiMonDTOchiTietGoiMonDTO) { }
- **public** List<ThanhToanDTO> LayDanhSachMonAnTheoMaGoiMon(**int** magoimon){ }
- **public boolean** CapNhatTrangThaiGoiMonTheoMaBan(**int** maban,String tinhTrang)

➤ LoaiMonAnDAO

- **public** LoaiMonAnDAO(Context context){ }
- **public boolean** ThemLoaiMonAn(String tenLoai){ }
- **public** List<LoaiMonAnDTO> LayDanhSachLoaiMonAn(){ }
- **public boolean** XoaMaLoai(**int** maloai){ }
- **public** String LayHinhMaLoai(**int** maLoai){ }

➤ MonAnDAO

- **public** MonAnDAO(Context context){ }
- **public boolean** ThemMonAn(MonAnDTO monAnDTO){ }
- **public boolean** XoaMonAn(**int** mamonan){ }
- **public boolean** SuaMonAn(MonAnDTO monAnDTO){ }
- **public** MonAnDTO LayDanhSachMonAnTheoMa(**int** mamonan){ }
- **public** List<MonAnDTO> LayDanhSachMonAnTheoLoai(**int** maLoai){ }

➤ NhanVienDAO

- **public** NhanVienDAO(Context context){ }
- **public long** ThemNhanVien(NhanVienDTO nhanVienDTO){ }
- **public boolean** SuaNhanVien(NhanVienDTO nhanVienDTO){ }
- **public boolean** KiemTraNhanVien(){ }
- **public int** KiemTraDangNhap(String tendangnhap,String matkhau){ }
- **public boolean** XoaNhanVien(**int** manv){ }
- **public** List<NhanVienDTO> LayDanhSachNhanVien(){ }
- **public** NhanVienDTO LayDanhSachNhanVienTheoMa(**int** manv){ }

-Lớp CustomAdapter: Là một thuật ngữ nói chung nhằm để tùy chỉnh một [View](#) (hiển thị)

theo ý của bạn. Và bằng thủ thuật này bạn có thể làm chủ được các đối tượng do chính các

bạn tạo ra.

Trong lớp này có các class:

- ❖ AdapterHienThiBanAn
- ❖ AdapterHienThiDanhSachMonAn
- ❖ AdapterHienThiLoaiMonAn
- ❖ AdapterHienThiLoaiMonAnThucDon
- ❖ AdapterHienThiNhanVien
- ❖ AdapterHienThiThanhToan

-Lớp FragmentApp: **Fragment** là một phần giao diện người dùng hoặc hành vi của một ứng dụng. **Fragment** có thể được đặt trong **Activity**, nó có thể cho phép thiết kế **activity**

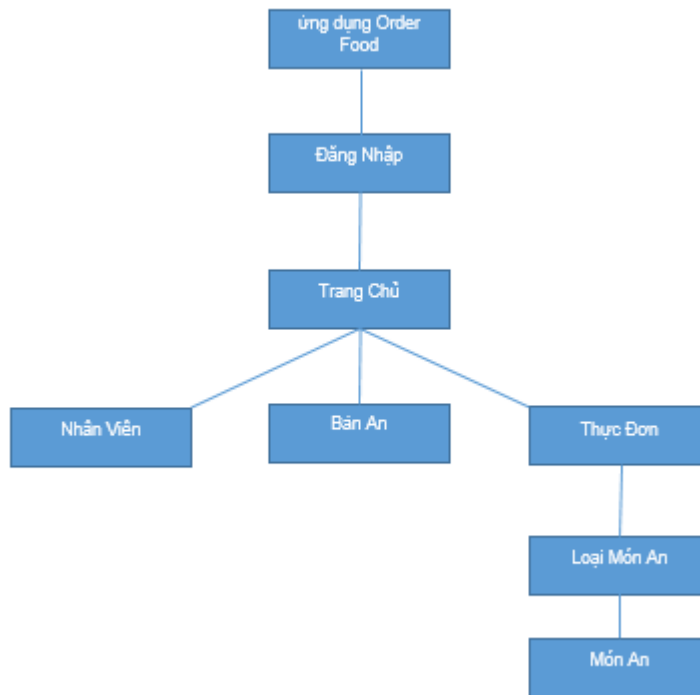
với nhiều mô-đun. Có thể nói **Fragment** là một loại **sub-Activity**.

Trong lớp này có Class:

- ❖ DatePickerFragment
- ❖ HienThiBanAnFagment
- ❖ HienThiDanhSachMonAnFagment
- ❖ HienThiNhanVienFagment
- ❖ HienThiThucDonFagment

+Lớp Activity: Activity là vòng đời trong 1 ứng dụng, là giao diện của một “màn hình” ứng dụng, mỗi màn hình chỉ chứa 01 Activity

- ❖ DangKyActivity
- ❖ DangNhapActivity
- ❖ SoLuongActivity
- ❖ SuaBanAnActivity
- ❖ ThanhToanActivity
- ❖ ThemBanAnActivity
- ❖ ThemLoaiThucDonActivity
- ❖ ThemThucDonActivity
- ❖ TrangChuActivity

3.1.5.Thiết kế giao diện:**3.1.5.1.Mô hình phân rã giao diện:****3.1.5.2.Danh sách các giao diện:**

STT	Tên màn hình	Ý nghĩa/Ghi chú
1	Đăng nhập	Đăng nhập vào chương trình
2	Đăng ký	Đăng ký tài khoản
3	Trang chủ	Hiển thị các menu
4	Bàn ăn	Hiển thị các bàn ăn
5	Loại món ăn	Hiển thị các loại món ăn
6	Món ăn	Hiển thị các chi tiết món ăn
7	Nhân viên	Hiển thị nhân viên

3.1.5.3Mô tả chi tiết mỗi giao diện○ **Giao diện 1:Đăng nhập**

**Giao diện:****Giao diện đăng ký:**

Đăng ký

Tên đăng nhập

Mật khẩu

☒ Nam ☐ Nữ

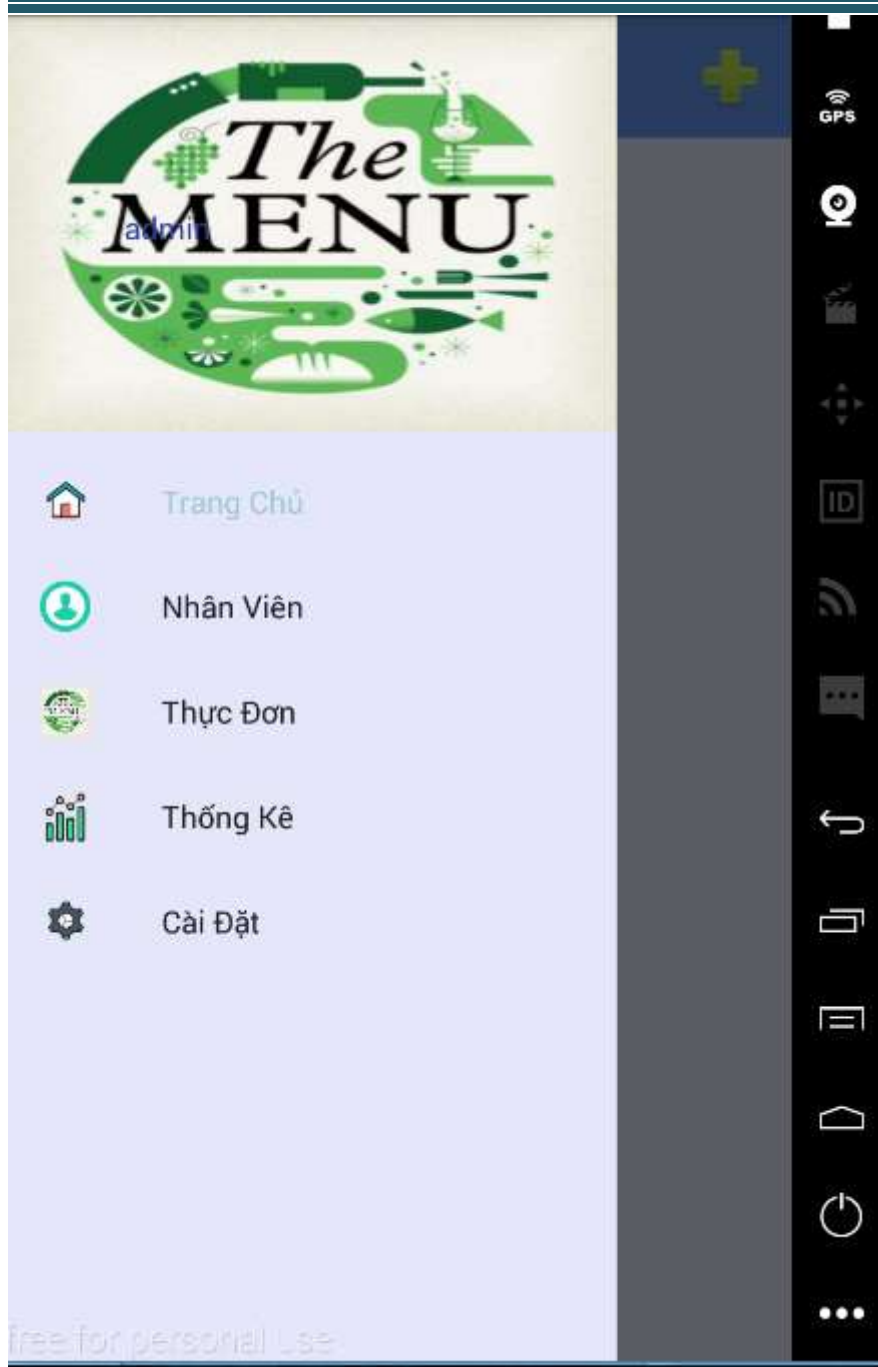
Ngày sinh

Chứng minh nhân dân

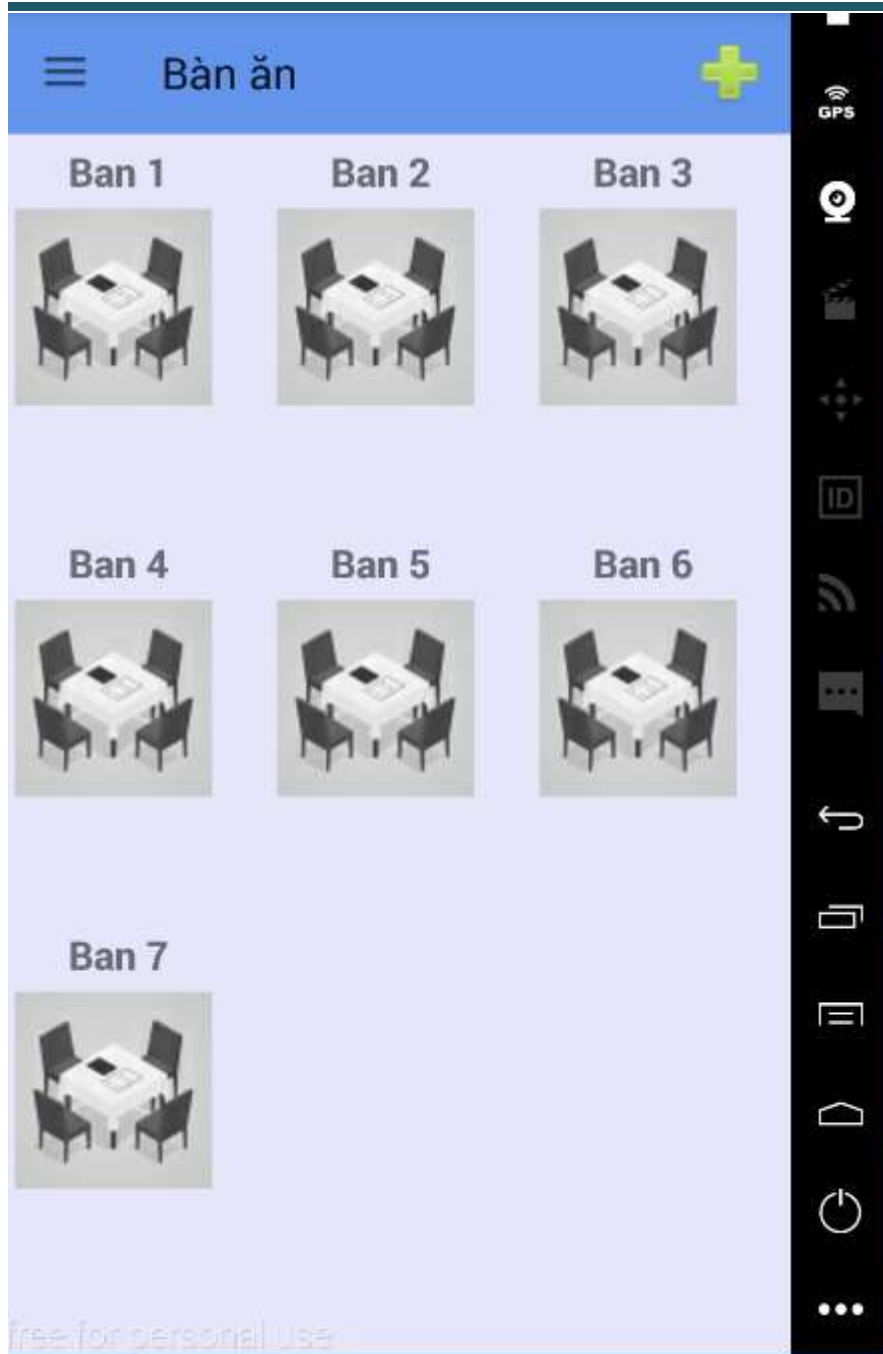
ĐỒNG Ý THOÁT

free for personal use

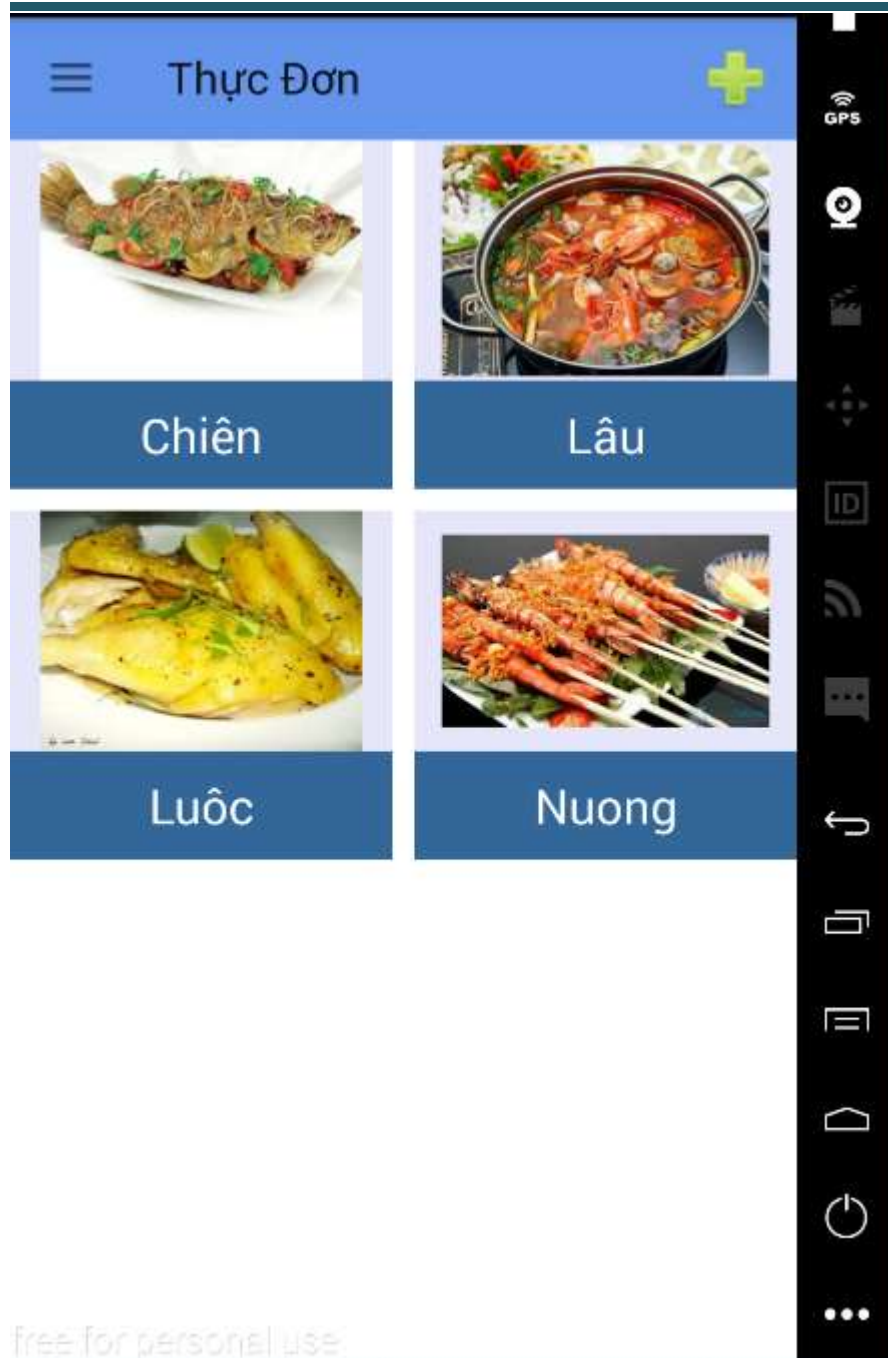
Giao diện trang chủ



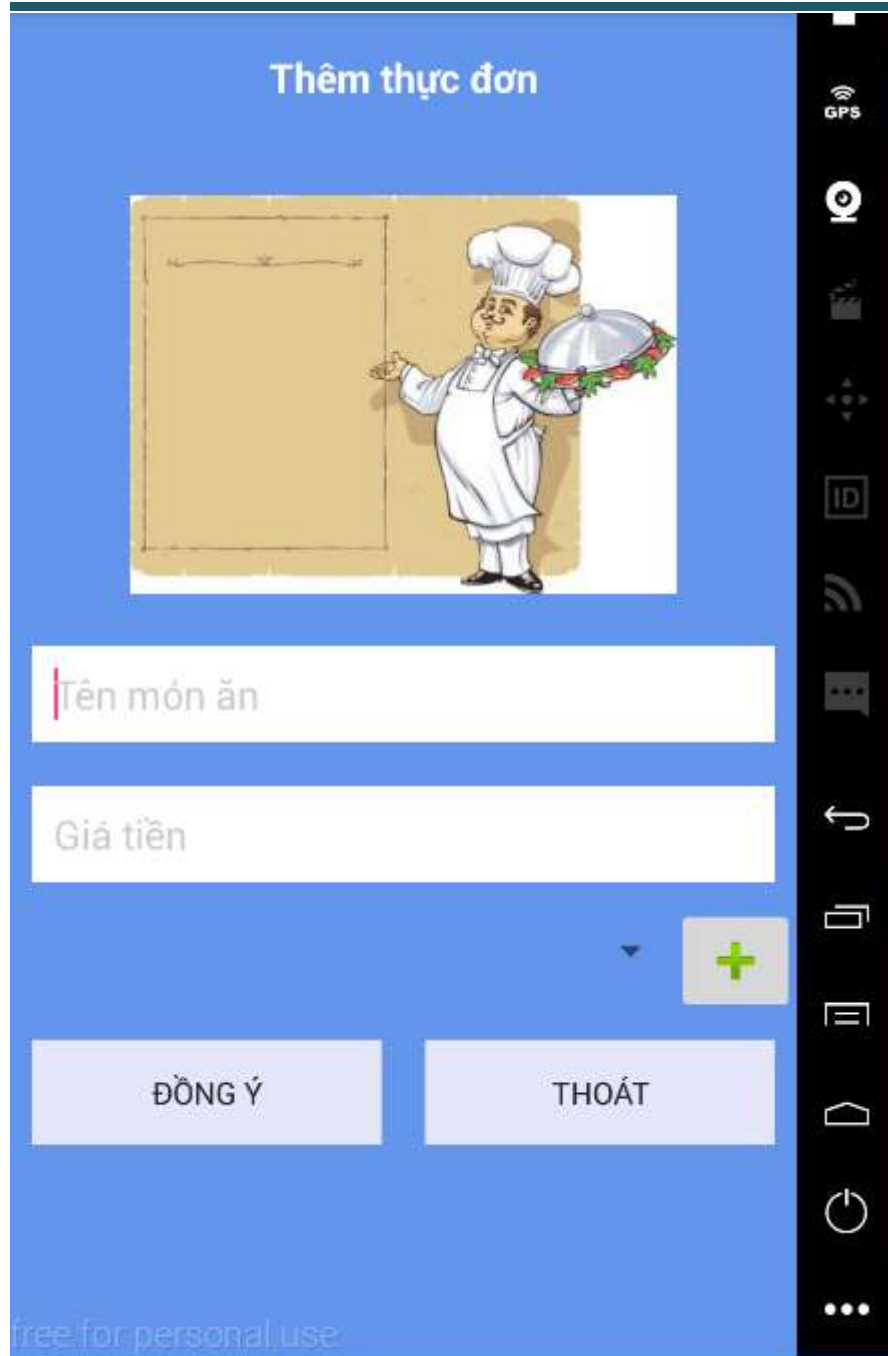
Giao diện bàn ăn:



Giao diện thực đơn



Giao diện thêm món ăn



CHƯƠNG 4: PHỤ LỤC- HƯỚNG DẪN CÀI ĐẶT VÀ SỬ DỤNG:

4.1 Hướng dẫn cài đặt:

4.1.1 . Giới thiệu về Android Studio

Android Studio là một nền tảng IDE (integrated development environment) dùng để phát triển các ứng dụng android, được Google release vào khoảng đầu năm 2015 thay thế cho bản Eclipse cũ. Android Studio được phát triển dựa trên IntelliJ IDEA Community Edition – công cụ lập trình tốt nhất cho java, giúp cho các lập trình viên tạo ứng dụng, thực hiện các thay đổi một cách dễ dàng, bên cạnh đó có thể xem trước trong thời gian thực và thiết kế giao diện đẹp hơn trước. Tiếng Việt cũng đã được tích hợp trong Android Studio. Đặc biệt, Android Studio cho phép người dùng Import Project từ Eclipse sang và logic lập trình cũng tương tự.

4.1.2 Hướng dẫn download Android Studio cho Window

Hướng dẫn cài đặt Android Studio

Một số yêu cầu về cấu hình khi cài đặt Android Studio

Microsoft®Windows®8/7/Vista/2003(32or64-bit)

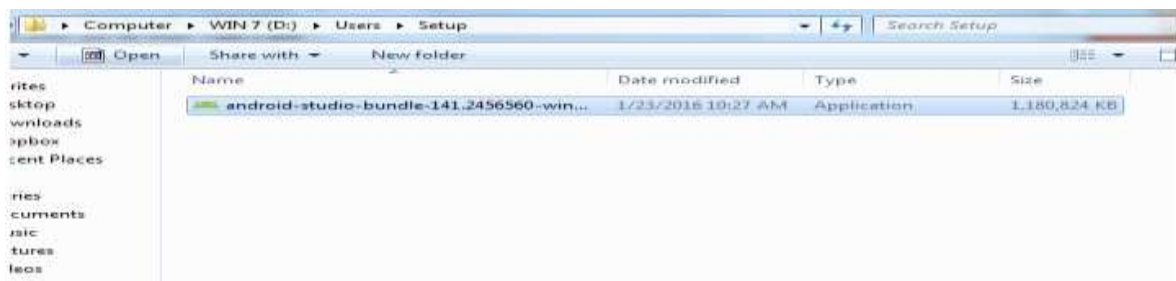
Tối thiểu 2 GB RAM, cấu hình đề nghị: 4 GB RAMĐộ phân giải tối thiểu 1280 x 800

Lựa chọn thêm cho accelerated emulator: Intel® processor with support for Intel® VT-x, Intel® EM64T (Intel® 64), and Execute Disable (XD) Bit functionality

* Cài đặt

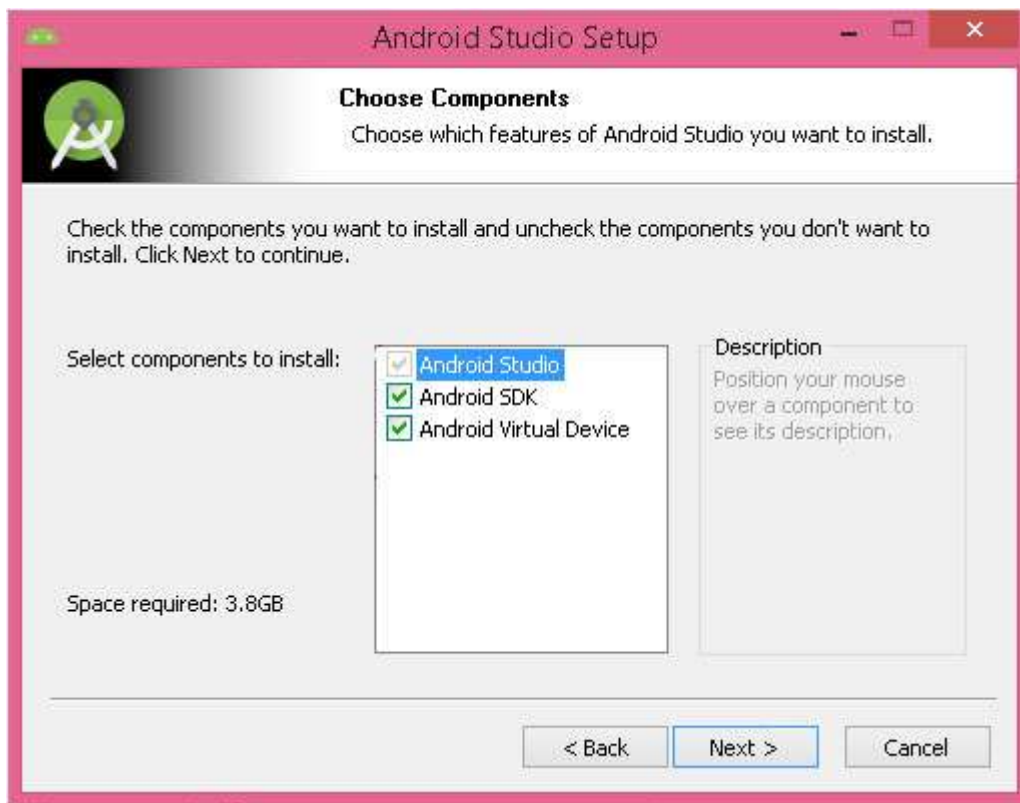
– Bước 1:

Vào thư mục chứa tập tin android studio bạn vừa download về, click vào **android-studio-bundle-141.2456560-windows**

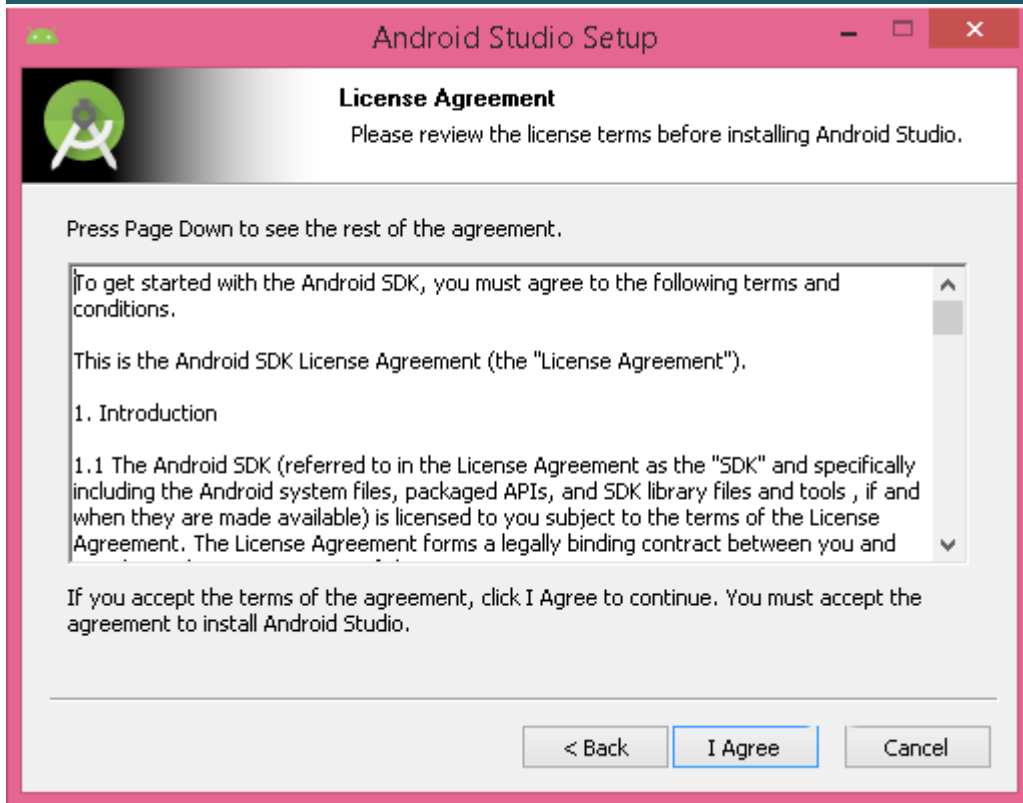


– Bước 2:

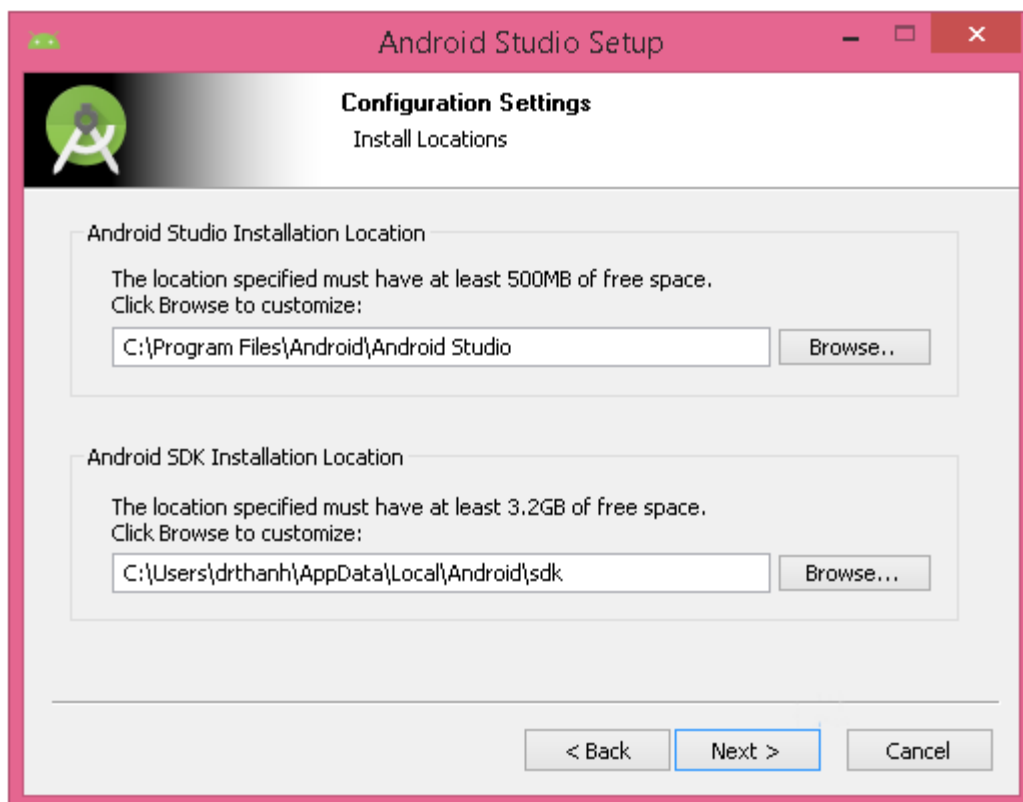
Màn hình chọn thành phần cài đặt. Ta chọn cấu hình cài đặt tất cả như trên rồi bấm Next.



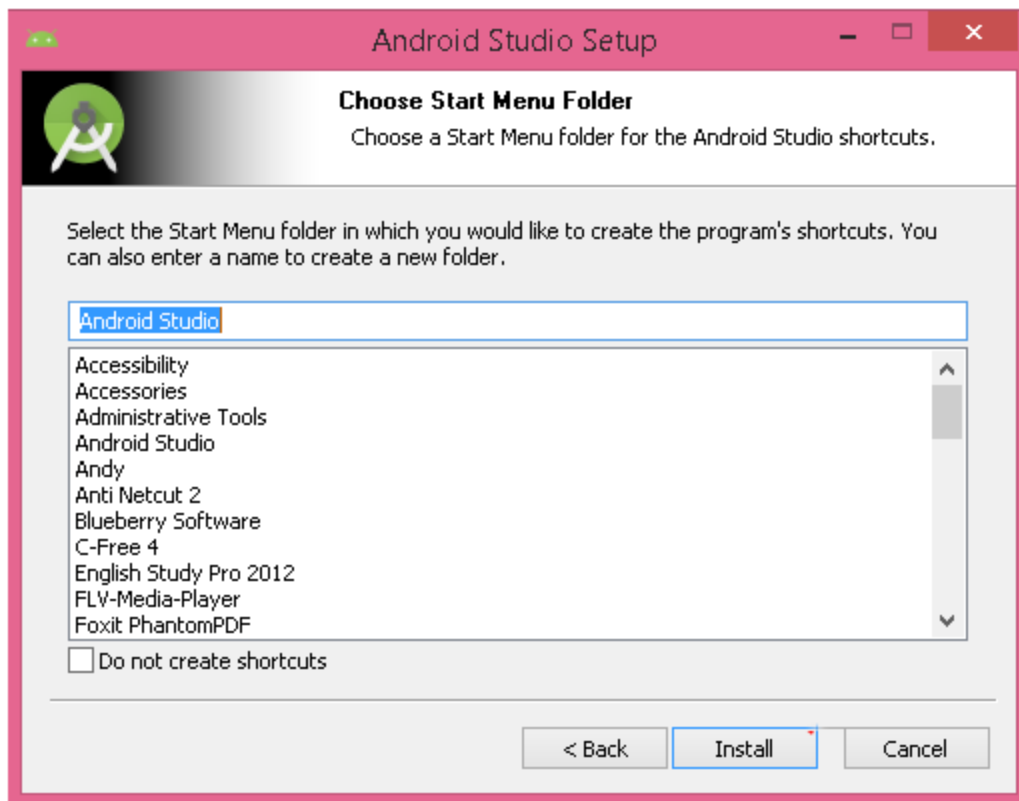
– Bước 3: Chọn “I agree” để xác nhận, đồng ý với các điều khoản và tiếp tục.



– Bước 4: Chọn nơi cài đặt Android Studio và Android SDK

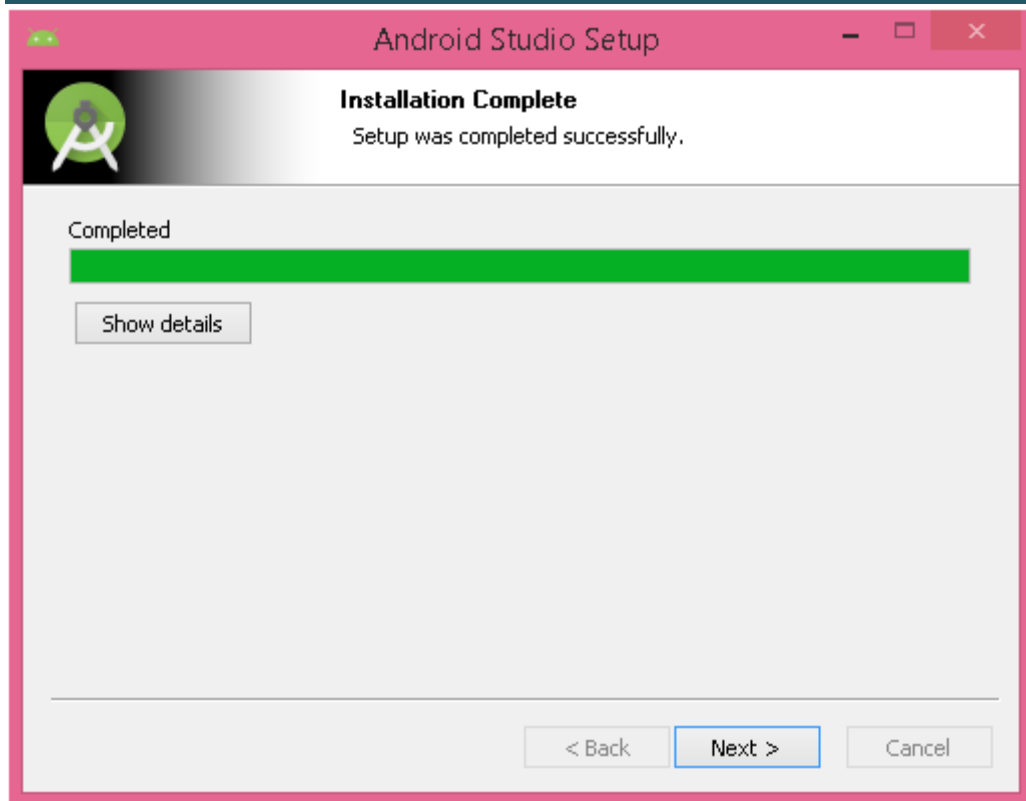


– Bước 5: Chọn Start Menu Folder. Ở bước này, bạn có thể tích tạo short cut cho ứng dụng hoặc không

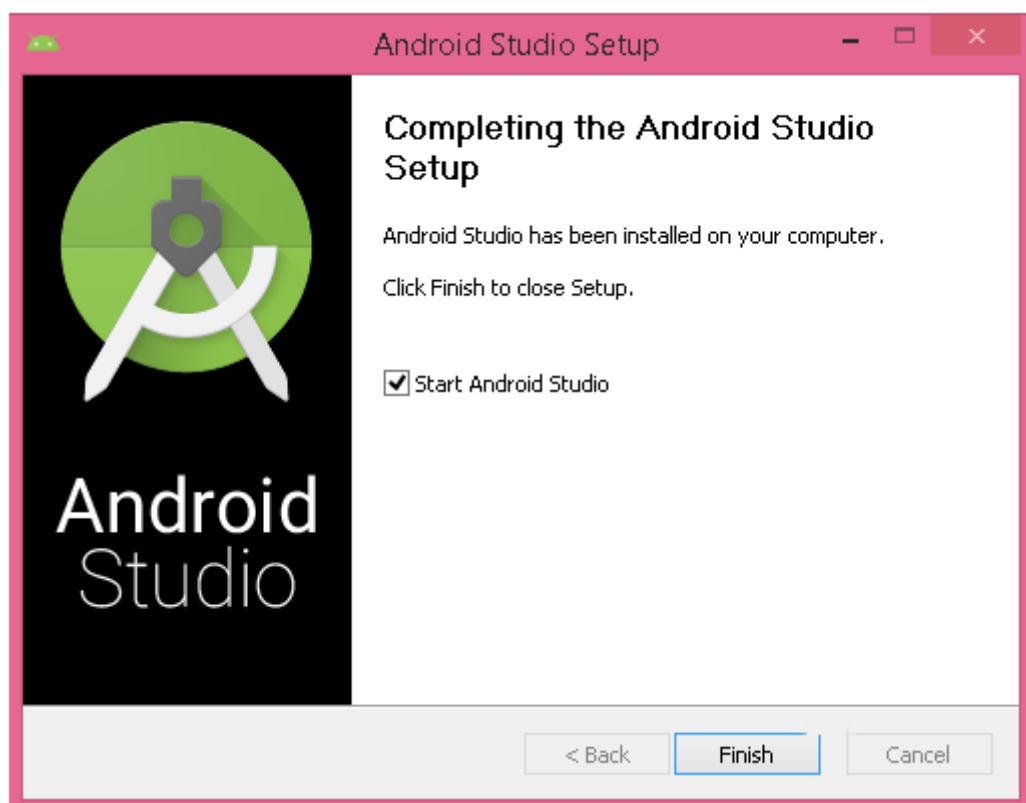


Bấm Install để bắt đầu cài đặt.

– Bước 6: Chờ hệ thống cài đặt các package cần thiết cho tới khi hoàn thành và bấm Next để tiếp tục



– Bước 7: Bấm Finish và click Start Android Studio để khởi động phần mềm



4.2.Hướng dẫn sử dụng

- Để chạy được project ta import project vào Android studio
- Cài máy ảo Genymotion
- Lấy file Apk cài trên điện thoại