ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

BÁO CÁO ĐỒ ÁN CHUYÊN NGÀNH

Đề tài: Phát Triển Ứng Dụng Don’t Eat Alone

Giảng viên hướng dẫn
Ths. Thái Huy Tân

Sinh viên thực hiện
Lê Hoàng Hân – 13520237
Trần Thị Hằng Nga - 13520534

Hồ Chí Minh, tháng 6 năm 2017

LỜI MỞ ĐẦU

Trong những năm gần đây, việc sử dụng các ứng dụng trên các thiết bị di động dần trở nên phổ biến hơn và gần như trở thành một phần trong cuộc sống hằng ngày. Các ứng dụng di động có mặt trong nhiều lĩnh vực như giáo dục, tài chính – kinh doanh, báo chí, giao thông, y tế, công nghiệp, nông nghiệp, du lịch, dịch vụ, giải trí và thể thao. Xây dựng ứng dụng trên thiết bị di động đã và đang tiếp tục là xu thế trong nền công nghiệp phần mềm.

Sự phát triển nhanh chóng của các ứng dụng di động cũng như internet đã dần hiện thực hóa việc liên kết với thế giới của người dùng. Việc kết bạn, làm quen với những người lạ, chia sẻ kế hoạch du lịch, vui chơi, giải trí, học tập, làm việc của bạn thân qua các ứng dụng di động để tìm người có chung ý tưởng cùng thực hiện đang dần trở nên phổ biến hơn và trở thành xu hướng hiện nay.

Nắm bắt được xu hướng của thị trường ứng dụng hiện nay. Đồng thời nhìn nhận tình hình thực tế, mặc dù có nhiều ứng dụng hỗ trợ tìm kiếm các địa điểm ăn uống đã thành công nhưng vẫn chưa đáp ứng được nhu cầu chia sẻ kế hoạch ăn uống của bản thân để tìm và mời những người lạ cùng thực hiện từ đó tạo cơ hội được giao lưu, kết bạn với những người lạ có cùng chung lý tưởng, tính cách và sở thích. Vì vậy, nhóm chúng tôi đã mạnh dạng lựa chọn xây dựng ứng dụng “Don’t Eat Alone” làm đề tài nghiên cứu, phát triển trong đồ án chuyên ngành. Chúng tôi mong muốn ứng dụng sẽ tạo ra cơ hội để mọi người có thể trải nghiệm việc làm quen, kết bạn với một người lạ có sở thích, tính cách phù hợp với mình thông qua những bữa ăn cùng nhau.

Trong bài báo cáo này, chúng tôi xin được trình bày về những kiến thức, cơ sở lý thuyết mà chúng tôi đã nghiên cứu và ứng dụng, cũng những phân tích, đánh giá của nhóm trong quá trình làm việc, xây dựng và phát triển ứng dụng “Don’t Eat Alone”. Bài báo cáo của chúng tôi bao gồm 5 chương sau:

- Chương 1: Tổng quan về đề tài
- Chương 2: Cơ sở lý thuyết
- Chương 3: Phân tích và thiết kế hệ thống
- Chương 4: Ứng dụng “Don’t Eat Alone”

- Chương 5: Tổng kết và hướng phát triển

Để đạt được những kết quả như ngày hôm nay, chúng tôi xin chân thành gửi lời cảm ơn đến thầy Thái Huy Tân – giảng viên hướng dẫn. Cảm ơn vì thầy đã tận tình quan tâm, hướng dẫn và giúp đỡ cho nhóm chúng tôi trong suốt quá trình thực hiện đồ án chuyên ngành. Những định hướng, bổ sung, góp ý của thầy là nền tảng cơ sở để chúng tôi có thể thực hiện các bước điều chỉnh, sửa chữa, từ đó góp phần giúp chúng tôi có được những nghiên cứu đúng đắn, thu được kết quả tốt nhất cho ứng dụng và hoàn thiện báo cáo đồ án chuyên ngành.

Đồng thời, chúng tôi cũng xin được cảm ơn Khoa Mạng máy tính và Truyền thông đã tạo điều kiện tốt nhất để chúng tôi được nghiên cứu và hoàn thành đồ án chuyên ngành. Những kiến thức, kinh nghiệm mà quý thầy, cô và Khoa đã cung cấp là tiền đề cơ sở để nhóm chúng tôi có thể thực hiện, áp dụng trong việc xây dựng và phát triển ứng dụng di động cho đồ án chuyên ngành. Cũng như các công việc thực tế trong thời gian sắp tới.

Mặc dù chúng tôi đã cố gắng và nỗ lực để hoàn thành báo cáo đồ án chuyên ngành. Nhưng vẫn khó tránh khỏi có những sai sót, hạn chế. Chúng tôi kính mong nhận được sự thông cảm, chia sẻ và góp ý của quý thầy, cô và Khoa Mạng máy tính và Truyền thông để chúng tôi có thể sửa chữa kịp thời và ngày một hoàn thiện hơn nữa.

Một lần nữa, chúng tôi xin chân thành cảm ơn và xin gửi lời chúc sức khỏe đến quý thầy, cô. Kính chúc Khoa Mạng máy tính và Truyền thông ngày càng phát triển và thành công.

Hồ Chí Minh, 25 tháng 06 năm 2017

Nhóm sinh viên thực hiện

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

NHẬN XÉT CỦA KHOA MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC SƠ ĐỒ

DANH MỤC BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT

Từ	Viết tắt của	Ý nghĩa
HTTP	Hypertext Transfer Protocol	Giao thức truyền tải siêu văn bản
JSON	JavaScript Object Notation	Định dạng trao đổi dữ liệu tuân thủ theo quy luật định sẵn
DEA	Don’t Eat Alone	Tên của ứng dụng
API	Application Programming Interface	Giao diện lập trình ứng dụng

Chương 1 **TỔNG QUAN VỀ ĐỀ TÀI**

1.1. Đặt vấn đề

Nhu cầu ăn uống là một nhu cầu cấp thiết hàng ngày của con người. Bên cạnh đó khi xã hội ngày càng tiến bộ, đời sống vật chất được cải thiện. Đồng nghĩa với việc nhu cầu ấy sẽ được nâng lên một mức cao hơn. Và khi đã giải quyết được vấn đề "Hôm nay nên ăn gì?" "Ăn ở đâu cho cảm giác ngon nhất?", con người dần quan tâm nhiều hơn vào việc "Cùng đi ăn với ai?". Đa phần các cuộc hẹn, bữa ăn thường được thực hiện cùng với gia đình, bạn bè. Tuy nhiên, hiện nay vẫn có rất đông đảo những người phải đi ăn một mình hay đơn giản nhiều người mong muốn trải nghiệm cảm giác tham gia một bữa ăn tối cùng một người lạ nhưng có cùng sở thích, tính cách từ đó tạo dựng được các mối quan hệ xã hội mới, học thêm được nhiều bài học, kinh nghiệm.

Trong bối cảnh sự phát triển mạnh mẽ và nhanh chóng của các ứng dụng trên thiết bị di động những năm gần đây. Hàng loạt những ứng dụng hỗ trợ tìm địa điểm ăn uống, review món ăn, đăng hàng, hướng dẫn cách chế biến món ăn, được xây dựng nhằm đáp ứng nhu cầu sử dụng của con người. Nhiều ứng dụng thu hút được nhiều người dùng và trở nên thành công. Tuy nhiên những ứng dụng hiện nay trên thị trường di động theo khát sát thông qua CHPlay và AppStrore vẫn chỉ dừng lại ở mức đáp ứng nhu cầu về ăn uống, tập trung khai thác trên phương diện món ăn, địa điểm ăn uống mà chưa giải quyết được nhu cầu chia sẻ kế hoạch và tìm một người phù hợp, sắp xếp một cuộc hẹn và mời người đó cùng đi ăn. Đây là một vấn đề mới và đang dần trở thành xu hướng sắp tới cần được nghiên cứu và khai thác.

1.2. Mục tiêu

Xây dựng một ứng dụng di động trên nền tảng Android đảm bảo tính khoa học và tính mới vừa tận dụng được sức mạnh của internet, sự phát triển của cộng động lập trình di động và mạng xã hội ảo, đồng thời có thể tạo ra giá trị thực, mạng lại những lợi ích trực tiếp, những hoạt động thực tế cho người dùng cho phép giải quyết vấn đề đặt ra ban đầu về nhu cầu chia sẻ kế hoạch, dự định ăn uống và tìm kiếm, thiết lập

một bữa ăn cùng với một người dùng lạ mặt được chọn lọc dựa theo những thông tin về độ tuổi, giới tính, sở thích, tính cách và địa điểm mà chính người dùng đã cung cấp.

Ngoài chức năng chính là chia sẻ kế hoạch ăn uống và tìm kiếm người đi ăn chung phù hợp, ứng dụng còn có các chức năng cơ bản như quản lý tài khoản, quản lý và sửa chữa thông tin, quản lý blog và gửi notification cho người dùng khi có thông báo mới.

Ứng dụng được thiết kế theo phong cách Material Design đảm bảo tính thẩm mĩ, thân thiện và dễ sử dụng. Các đối tượng, layout phải được custom để bắt kịp với xu hướng hiện này trên thị trường ứng dụng di động. Được áp dụng các công nghệ, kỹ thuật mới nhất nhằm đẩy mạnh hiệu suất, tốc độ của ứng dụng.

Chủ động xây dựng server riêng để xử lý các yêu cầu từ phía client nhằm đạt được sự chủ động nhất khi xử lý các yêu cầu dữ liệu, nâng cao khả năng mở rộng, phát triển ứng dụng trong tương lai.

Chương 2. **CƠ SỞ LÝ THUYẾT**

2.1. Node.js

2.1.1. Giới thiệu về Node.js

Dựa theo định nghĩa từ website chính thức của Node.js https://nodejs.org thì Node.js là một JavaScript thời gian thực được xây dựng trên Chrome's V8 JavaScript engine. Node.js sử dụng các phần phát sinh các sự kiện (event-driven), mô hình non-blocking I/O để tạo ra các ứng dụng nhẹ và hiệu quả cao cho các ứng dụng về dữ liệu thời gian thực chạy trên các thiết bị phân tán.

Node.js là một mã nguồn mở được phát triển bởi Ryan Dahl. Phiên bản đầu tiên của Node.js được cho ra mắt vào năm 2009. Node.js có thể chạy được trên nhiều nền tảng khác nhau như Windows, Linux hay Mac OS.

Điều đặc biệt ở Node.js mang lại là mặc dù các JavaScript thường được sử dụng trên client thì Node.js dù được xây dựng trên nền tảng JavaScript một ngôn ngữ thuần client nhưng có thể sử dụng để thực hiện các công việc trên cả client và server.

2.1.2. Tại sao nên sử dụng Node.js

Một số đặc điểm quan trọng giúp Node.js trở nên rất đáng để nghiên cứu sử dụng đó là:

- Không đồng bộ và phát sinh sự kiện (Event Driven): Tất cả các API của thư viện Node.js đều không đồng bộ, nghĩa là blocking (khóa). Nó rất cần thiết vì Node.js không bao giờ đợi một API trả về dữ liệu. Server chuyển sang một API sau khi gọi nó và có cơ chế thông báo về sự kiện của Node.js giúp Server nhận được phản hồi từ các API gọi trước đó.
- Tốc độ xử lý nhanh: Được thực thi dựa trên V8 JavaScript Engine của Google Chorme. Khả năng tự build server bằng ngôn ngữ client giúp cho việc trung chuyển dữ liệu diễn ra nhanh hơn. Đồng thời giảm khả năng rủi ro bị ngắt (interupt). Ngoài ra nhờ cơ chế Non I/O Blocking, tận dụng tối đa tài nguyên của server, không tạo ra độ trễ như các ngôn ngữ server-side khác.

- Các tiến trình đơn giản nhưng hiệu năng cao: Node.js sử dụng một mô hình luồng đơn (single thread) với các sự kiện lặp. Các cơ chế sự kiện giúp Server trả lại các phản hồi với một cách không khóa và tạo cho Server hiệu quả cao ngược lại với các cách truyền thống tạo ra một số lượng luồng hữu hạn để quản lý request. Nodejs sử dụng các chương trình đơn luồng và các chương trình này cung cấp các dịch vụ cho số lượng request nhiều hơn so với các Server truyền thống như Apache HTTP Server.
- JavaScript là một ngôn ngữ phổ biến, điều này đã giúp node.js trở nên dễ dàng được sử dụng và áp dụng rộng rãi hơn.
- Node.js hỗ trợ rất tốt cho việc xây dựng ứng dụng thời gian thực.

2.1.3. Áp dụng Node.js vào đề tài xây dựng ứng dụng Don’t Eat Alone

Nhóm chúng tôi sử dụng Node.js vào việc xây dựng server cho ứng dụng, tạo các phương thức Get/Post nhận và gửi kết quả cho client bởi những lợi ích to lớn mà Node.js mang lại. Khả năng xử lý dữ liệu thời gian thực, tốc độ xử lý nhanh chóng, thực hiện được nhiều yêu cầu cùng lúc và dễ dàng sử dụng rất phù hợp để xây dựng server cho ứng dụng di động. Bên cạnh đó, việc sử dụng Node.js để xây dựng server cho phép dễ dàng can thiệp sâu vào việc xử lý, lưu trữ dữ liệu và khả năng mở rộng lớn.

2.2. Socket.IO

2.2.1. Giới thiệu về Socket.IO

Socket.IO là một thư viện JavaScript cho các ứng dụng web, mobile thời gian thực cho phép thực hiện giao tiếp hai chiều giữa các client và server theo thời gian thực. Socket.IO hoạt động trên mọi nền tảng, trình duyệt hoặc thiết bị và tập trung chủ yếu vào độ tin cậy và tốc độ. Giống như Node.js, nó được phát sinh theo sự kiện.

Thư viện này gồm 2 phần và đều có một API giống hệt nhau:

- Client-side: gồm bộ thư viện viết cho web(JavaScript), iOS, Android.
- Server-side: viết bằng JavaScript và dùng cho các máy chủ Node.js

Socket.IO cung cấp khả năng thực hiện phân tích thời gian thực, luồng nhị phân, nhắn tin nhanh và kết hợp về tài liệu. Socket.IO được sử dụng bởi nhiều người dùng như Microsoft Office, Yammer, Zendesk, Trello. Chính điều này đã giúp Socket.IO trở thành một trong những JavaScript framework mạnh mẽ nhất trên Github.

2.2.2. Tại sao nên sử dụng Socket.IO

Socket.IO mang lại rất nhiều lợi ích trong xây dựng và phát triển các ứng dụng thời gian thực. Những ưu điểm mà Socket.IO có thể kể đến như:

- WebSockets cung cấp khả năng giao tiếp hai chiều mạnh mẽ, có độ trễ thấp và dễ xử lý lỗi.
- Không cần phải có nhiều kết nối như phương pháp Comet long-polling và cũng không có những nhược điểm như Comet streaming.
- API rất dễ sử dụng trực tiếp mà không cần bất kỳ các tầng bổ sung nào, so với Comet, thường đòi hỏi một thư viện tốt để xử lý kết nối lại, thời gian chờ timeout, các Ajax request (yêu cầu Ajax), các tin báo nhận và các dạng truyền tải tùy chọn khác nhau (Ajax long-polling và jsonp polling).

Đồng thời Socket.IO là một module của Node.js được xây dựng nhằm mục đích tạo ra realtime Node.js application. Nên việc sử dụng Socket.IO sẽ hỗ trợ tốt cho Node.js trong việc giao tiếp giữa client và server.

2.2.3. Áp dụng Socket.IO vào đề tài xây dựng ứng dụng Don’t Eat Alone

Từ những ưu điểm có được của Socket.IO, nhóm đã áp dụng Socket nhằm mục đích đảm bảo được tính realtime cho ứng dụng khi tương tác với đối tượng online khác.

Trong ứng dụng, tính realtime được thể hiện thông qua các tính năng “Send Invitaion” và “Response Invitation”. Khi người dùng bật chế độ tìm kiếm người đi ăn chung, ngay đồng thời server và ứng dụng sẽ bắt đầu lắng nghe sự kiện. Khi có sự kiện “Send Invitation” từ ứng dụng người gửi, server bắt được, xử lý và gửi ngay về phía ứng dụng người nhận. Khi ứng dụng người nhận cũng đang chế độ lắng nghe sẽ nhanh chóng đón được sự kiện “mời đi ăn chung”. Nhờ có những ưu điểm

thuận lợi của Socket.IO mà các tính năng này đảm bảo được người dùng trong hệ thống tương tác với nhau gần như ở thời gian thực.

2.3. Google API

2.3.1. Giới thiệu về API và Google API

API được viết đầy đủ trong tiếng Anh là "application programming interface" và được hiểu trong tiếng Việt là "giao diện lập trình ứng dụng". Tuy nhiên "giao diện ở đây không được hiểu theo nghĩa thông thường mà API được hiểu là một "giao diện" giữa phần mềm với phần mềm. API là cách để các phần mềm (hệ điều hành, ứng dụng, các module trong hệ thống doanh nghiệp) giao tiếp với nhau và tận dụng năng lực của nhau. Có thể hiểu đơn giản, API là khớp nối giữa các thành phần phần mềm. Các phần mềm muốn tương tác với nhau thì phải "gọi" API của nhau.

Google API là các API được công ty Google phát triển và cung cấp cho các lập trình viên. Khi chúng ta muốn tương tác với các phần mềm, thư viện của Google, bắt buộc chúng ta phải cài đặt Google API và Google Play Service.

2.3.2. Tại sao nên sử dụng Google API

Các phần mềm, ứng dụng liên kết, làm việc với nhau thông qua API vì vậy việc sử dụng API là điều nhất thiết khi xây dựng và phát triển ứng dụng. Trong khi đó, Google được xem như là "gã khổng lồ" trong công nghệ, nên việc "đứng trên vai" của Google sẽ mang lại nhiều lợi ích khi thực hiện đề tài. Cụ thể như:

- Chức năng được cung cấp rất đa dạng và phong phú
- Dễ dàng cài đặt và sử dụng
- Các API được Google cung cấp miễn phí với số lượng người dùng lớn.
- Các API được cập nhật thường xuyên.
- Nhận được sự hỗ trợ rất tốt từ Google cũng như cộng đồng lập trình viên.

2.3.3. Áp dụng Google API vào đề tài xây dựng ứng dụng Don't Eat Alone

Google API được áp dụng trong ứng dụng nhằm giúp cho người dùng dễ dàng lấy được địa chỉ mình cần tìm kiếm.

Trong ứng dụng, Google API được sử dụng là Google Place API_Place AutoCompleTextView qua các tính năng "xác định địa chỉ người dùng" và "xác định địa chỉ yêu cầu".

Nhờ kho dữ liệu lớn của Google và tính gợi nhắc tự động chính là chức năng autocompletexview có trong api, mà khi người dùng tìm kiếm địa điểm cần tìm ở hai tính năng, đều nhanh chóng được Google API xuất về những địa điểm phù hợp nhất. Điều này sẽ tạo được sự dễ chịu cho người dùng, đồng thời giúp ứng dụng dễ dàng hơn trong việc xác định địa chỉ người dùng và lấy được Latlng cụ thể, phục vụ cho các thao tác tính toán khoảng cách trong ứng dụng.

2.4. Retrofit 2

2.4.1. Giới thiệu về Retrofit 2

Retrofit là một thư viện HTTP Client cho Android và Java. Khi sử dụng Retrofit lập trình viên sẽ dễ dàng kết nối tới một REST web service bằng cách dịch API thành các java interface.

Thư viện mạnh mẽ này giúp chúng ta làm việc dễ dàng với dữ liệu Json hay XML sau đó phân tích thành các đối tượng object cơ bản. Nó cung cấp đầy đủ các phương thức GET, POST, PUT, PATCH và DELETE.

Giống với các thư viện mã nguồn mở khác, Retrofit được xây dựng dựa trên một số thư viện, việc phân tích JSON thì có thể sử dụng thư viện:

- Gson: com.squareup.retrofit:converter-gson
- Jackson: com.squareup.retrofit:converter-jackson
- Moshi: com.squareup.retrofit:converter-moshi
- Protobuf: com.squareup.retrofit2:converter-protobuf
- Wire: com.squareup.retrofit2:converter-wire

Phân tích XML, sử dụng thêm thư viện:

- Simple Framework: com.squareup.retrofit2:converter-simpleframework

2.4.2. Tại sao nên sử dụng Retrofit 2

Retrofit 2 sở hữu nhiều đặc điểm nổi bật giúp nó trở thành một thư viện networking đang được sử dụng rộng rãi trong cộng đồng lập trình viên. Trong đó có thể kể đến như:

- Tốc độ xử lý vượt trội.
- Xử lý hiệu quả các request, đơn giản và dễ bảo trì, sửa chữa.
- Thường được sử dụng cùng với thư viện Picasso để xử lý hình ảnh.
- Retrofit sử dụng thư viện OkHttp và sử dụng thread hỗ trợ nhiều request một lúc. Có thể sử dụng kết hợp với RxAndroid.
- Hỗ trợ đầy đủ multipart uploads trong việc tải file lên server.

2.4.3. Áp dụng Retrofit 2 vào đề tài xây dựng ứng dụng Don't Eat Alone

Nhờ có các tính năng trong thư viện Retrofit2, đã hỗ trợ nhóm trong việc xây dựng các API giao tiếp dễ dàng với server trong quá trình tương tác, xử lý dữ liệu.

Nhóm đã tạo ra một file java interface sử dụng các phương thức được hỗ trợ trong thư viện Retrofit2 như: get, post, put, delete liên kết với server thông qua các đường dẫn được định nghĩa sẵn để dễ dàng trong việc tương tác lấy, thêm, sửa, xóa dữ liệu từ server, đồng thời dễ dàng hơn trong việc gọi và sử dụng lại từ phía ứng dụng.

2.5. MongoDB

2.5.1. Giới thiệu về MongoDB

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở NoSQL phổ biến được viết bằng C++. Được phát triển bởi MongoDB Inc. Tính đến tháng 2/2015, MongoDB được xếp thứ 4 trong số các hệ thống cơ sở dữ liệu phổ biến nhất.

MongoDB là cơ sở dữ liệu hướng tài liệu, nó lưu trữ dữ liệu trong các document dạng JSON với schema động rất linh hoạt. Nghĩa là bạn có thể lưu các bản ghi mà không cần lo lắng về cấu trúc dữ liệu như là số trường, kiểu của trường lưu trữ. Tài liệu MongoDB tương tự như các đối tượng JSON.

MongoDB là một cơ sở dữ liệu NoSQL hỗ trợ đa nền tảng, nó có thể chạy trên Windows, Linux và Mac và hỗ trợ hầu hết các ngôn ngữ lập trình phổ biến như C#, Java, PHP, Javascript trong các môi trường phát triển khác nhau.

2.5.2. Tại sao nên sử dụng MongoDB

Dựa trên những lợi ích của MongoDB mang lại và nhu cầu xây dựng ứng dụng thời gian thực đặt ra ban đầu. MongoDB thật sự là một trong những lựa chọn thích hợp bởi vì:

- Tốc độ xử lý nhanh chóng: do được viết bởi C++ nên nó có khả năng tính toán với tốc độ cao chứ không giống như các hệ quản trị cở sở dữ liệu hiện nay.
- Mô hình dữ liệu linh hoạt: MongoDB lưu trữ dữ liệu trong các tài liệu thay vì các bảng cho phép thay đổi cấu trúc các document đơn giản chỉ bằng cách thêm mới các trường hoặc xóa các trường có sẵn. Khả năng này của MongoDB giúp bạn trình bày các quan hệ dạng thứ bậc, để lưu trữ mảng, và các cấu trúc phức tạp khác một cách dễ dàng, cung cấp khả năng mở rộng tốt.
- Giảm thiểu rủi ro: đặt trong trường hợp đề tài phát triển ứng dụng di động yêu cầu thời gian thực và nhiều người dùng tương tác, nếu quá trình load bị lỗi tại một thời điểm nào đó thì nó sẽ bỏ qua phần đó nên sẽ an toàn hơn.
- Phù hợp với việc phát triển ứng dụng di động và server Node.js.

2.5.3. Áp dụng Mongo DB vào đề tài xây dựng ứng dụng Don’t Eat Alone

Trong ứng dụng, Mongo DB đơn giản là hệ quản trị cơ sở Nosql. Ứng dụng sẽ tương tác lấy dữ liệu Mongo DB thông qua server được viết trên nền tảng Node js.

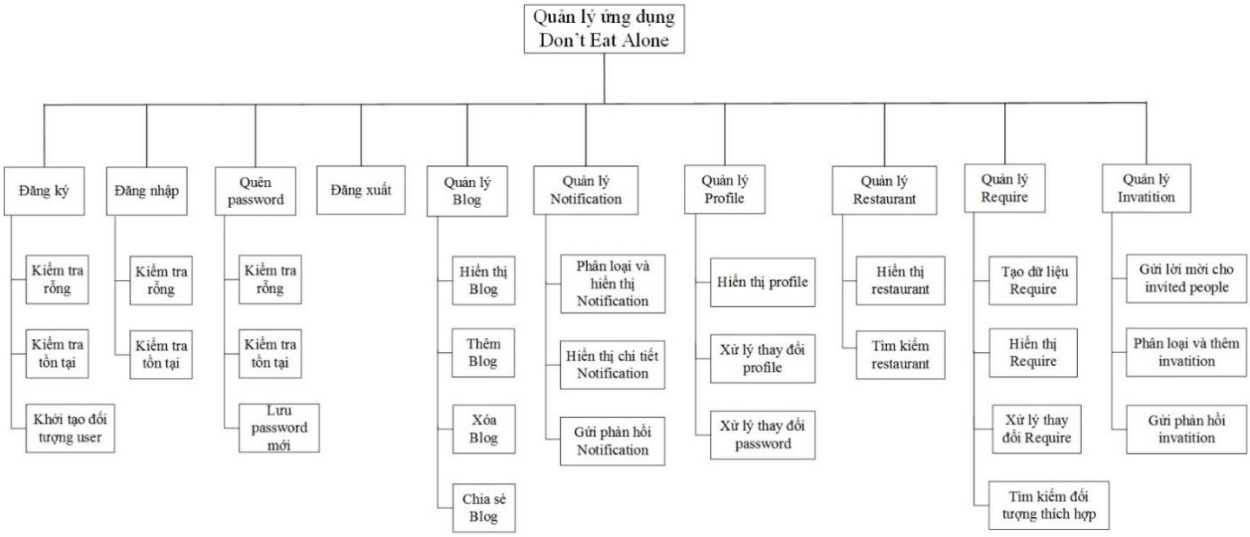
Với việc lưu dữ liệu mà không cần định nghĩa sẵn và không cần thiết lập mối quan hệ giữa các cơ sở dữ liệu giúp cho việc lưu trữ cũng như lấy dữ liệu được thực hiện nhanh chóng.

Nhờ việc lưu trữ dưới dạng JSON, đối tượng lồng đối tượng, việc quản lý và truy xuất cơ sở dữ liệu trong Mongo DB được thực hiện và kiểm soát hiệu quả hơn. Và việc lưu trữ nhiều cấu trúc cơ sở dữ liệu trong ứng dụng thì việc sử dụng Mong DB là một lời thế.

Chương 3. **PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG**

3.1. Mô hình phân rã chức năng

Mô hình phân rã chức năng cho phép xác định phạm vi các chức năng được xây dựng và phát triển trong ứng dụng, cho phép người sử dụng có một cái nhìn tổng thể về các chức năng có trong ứng dụng thông qua việc biểu diễn phân rã có thứ bậc đơn giản các chức năng của ứng dụng.

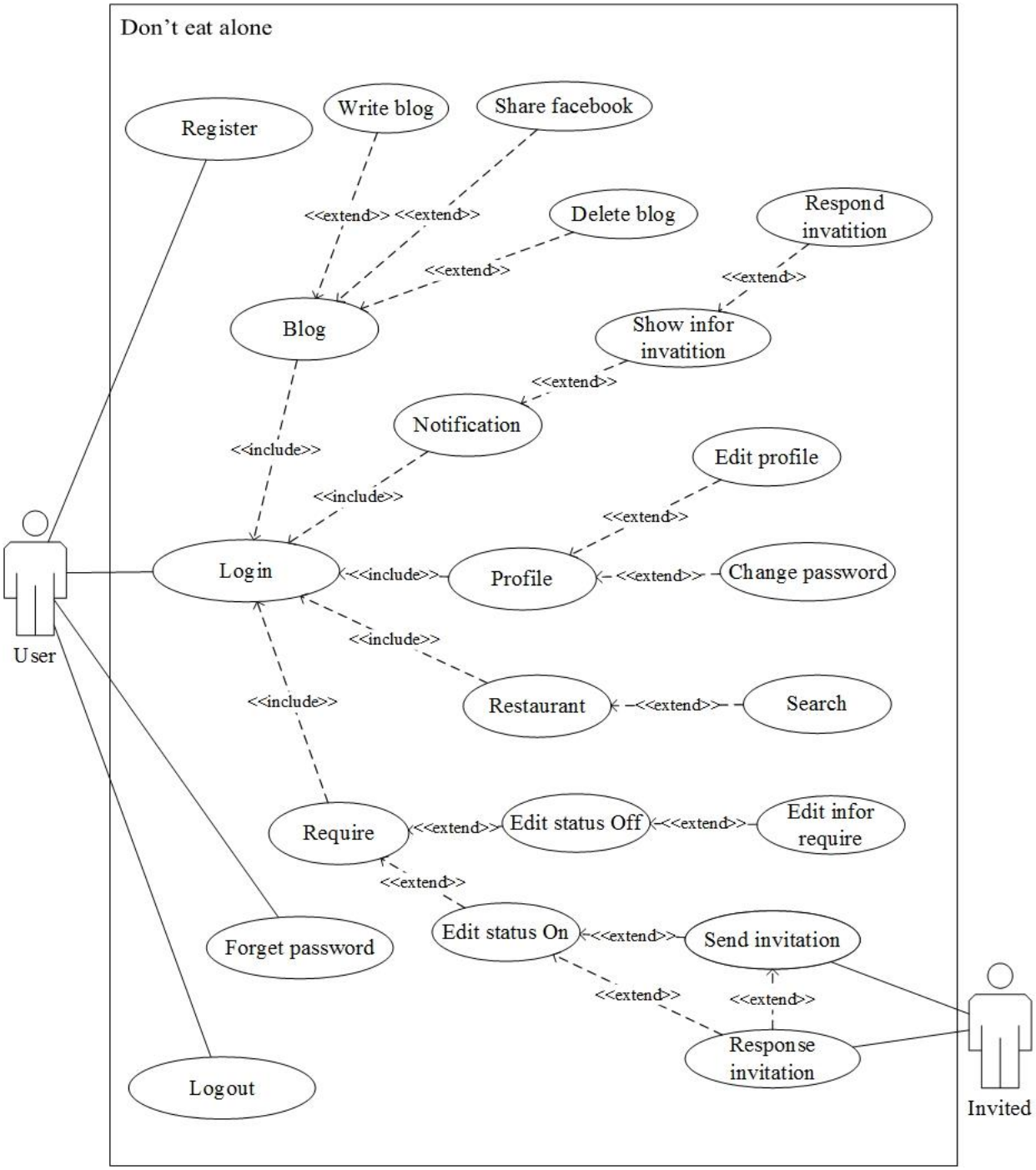


Sơ đồ 3.1. Mô hình phân rã chức năng của ứng dụng Don't Eat Alone

3.2. Mô hình Usecase

3.2.1. Mô hình

Use case mô tả sự tương tác đặc trưng giữa người dùng và hệ thống. Nó thể hiện ứng xử của hệ thống đối với bên ngoài, trong một hoàn cảnh nhất định, xét từ quan điểm của người sử dụng.



Sơ đồ 3.2. Mô hình User case của ứng dụng Don't Eat Alone

Các đối tượng tương tác trong ứng dụng:

- User: Chính là người dùng đã có tài khoản được đăng ký tại app, và đăng nhập vào hệ thống.

- Invited: Cũng chính là user sử dụng hệ thống, nhưng ở trong sơ đồ này, đảm nhiệm vai trò là người có thể tương tác với user trong chức năng require.

Hiện thực hóa sơ đồ Usercase:

Bảng 3.1. Hiện thực hóa sơ đồ Usecase

STT	Tên Usecase	Ý nghĩa/Ghi chú
1	Register	Đăng ký tài khoản nếu chưa có, để trở thành user của ứng dụng, có thể đăng nhập vào ứng dụng.
2	Login	Nhập tài khoản đã có, đăng nhập vào ứng dụng.
3	Forget password	Lấy lại mật khẩu nếu quên mật khẩu tài khoản.
4	Logout	Thoát tài khoản đnag dùng ra khỏi ứng dụng
5	Write Blog	Viết blog, là kiểu viết nhật ký, tâm sự hằng ngày dưới chế độ riêng tư.
6	Delete Blog	Xóa blog.
7	Share Blog	Chia sẻ Blog lên facebook để bạn bè có thể nhìn thấy nội dung.
8	Show info Notification	Nhấn vào thông báo bất kỳ trong tab Notification để thấy thông tin chi tiết hơn của thông báo đó.
9	Response Notification	Phản hồi lại thông báo đã nhận được.
10	Edit Profile	Chỉnh sử Profile cá nhân để có Profile ưng ý nhất sau khi đã có thông tin Profile ở phần Register.

11	Change Password	Thay đổi password
12	Search Restaurant	Tìm kiếm nhà hàng thích hợp, phụ thuộc vào địa điểm được nhập vào.
13	Edit Status Off/On	Thay đổi trạng thái on/ off để xác định tìm kiếm người đi ăn chung.
14	Edit info Require	Chỉnh sửa chi tiết yêu cầu ở trạng thái off, để hệ thống tìm kiếm cho người dùng, người đi ăn phù hợp nhất.
15	Send Invitation	Gửi yêu cầu mời đi ăn cho đối tượng.
16	Reponse Invitation	Phản hồi lại Invitation khi có thông báo lời mời.

3.2.2. Mô tả từng chức năng

3.2.2.1. Chức năng Register, Login, Forget password, Logout

Bảng 3.2. Mô tả chức năng Register, Login, Forget password, Logout

Tên Usecase	Register, Login, Forget password, Logout	
Mô tả/mục đích	Đây là các chức năng cơ bản cho ứng dụng. Register cho phép người dùng đăng ký tài khoản. Login cho phép đăng nhập vào ứng dụng. Forget password cho phép người dùng lấy lại mật khẩu và logout để đăng xuất tài khoản khỏi ứng dụng.	
Yêu cầu tiên quyết	Cài đặt ứng dụng trên thiết bị di động.	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng mở ứng dụng 3. Nếu người dùng chưa có tài khoản, người dùng	2. Hệ thống yêu cầu người dùng đăng nhập.

	phải nhấp vào chức năng Register để đăng ký tài khoản.	
		4. Hệ thống sẽ chuyển qua màn hình đăng ký tài khoản, yêu cầu người dùng nhập những thông tin nhất định.
	5. Người dùng nhập thông tin được yêu cầu	
		6. Hệ thống kiểm tra các thông tin được nhập vào. Nếu thông tin không hợp lệ, yêu cầu người dùng nhập lại, nếu thông tin hợp lệ, cho phép người dùng nhập các thông tin tiếp theo.
	7. Nhấn nút Register để hoàn thành quá trình đăng ký.	
		8. Lưu thông tin người dùng vào database, xác thực đăng ký tài khoản thành công, chuyển về màn hình đăng nhập.
	9. Chức năng Login, để đăng nhập, xác thực tài	

	khoản đúng cần đăng nhập.	
		10. Kiểm tra tài khoản, nếu không phù hợp, yêu cầu người dùng nhập lại. Nếu đúng, chuyển sang màn hình chính của ứng dụng.
	11. Chức năng Forget password, nếu người dùng quên mật khẩu, nhấn forget password, nhập các thông tin yêu cầu để được thay đổi mật khẩu mới.	
		12. Kiểm tra thông tin người dùng nhập vào, nếu đúng, cho phép thay đổi password, lưu password mới vào database.
	13. Chức năng Logout, nhấn logout để thoát ứng dụng.	
		14. Thoát ứng dụng, trở về màn hình Login.
Kết quả đạt được	Tạo được tài khoản cho ứng dụng, được phép đăng nhập vào hệ thống, thay đổi password khi quên mật khẩu, thoát tài khoản ra khỏi ứng dụng.	

3.2.2.2. Các chức năng trong tab Blog

Bảng 3.3. Mô tả các chức năng trong tab Blog

Tên Usecase	Write Blog, Delete Blog, Share Blog	
Mô tả/mục đích	Các chức năng này cho phép người dùng viết Blog.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Blog	
		2. Hệ thống gửi về các Blog đã được viết trước đó của người dùng nếu có.
	3. Chức năng Write Blog: Người dùng có thể, nhấn vào button "What do you think" để viết Blog.	
		4. Hệ thống sẽ chuyển qua màn hình Post Blog để người dùng thêm Blog.
	5. Sau khi viết xong Blog, người dùng nhấn Post để hoàn thành hoặc Cancel để hủy.	
		6. Hệ thống sẽ lưu lại nội dung Blog lên database nếu người dùng nhấn Post. Và trở lại màn hình Blog. Load lại toàn bộ Blog.

	7. Chức năng Delete Blog: Người dùng xóa Blog bằng cách nhấp vào button "Delete" tại bài viết cần xóa.	8. Hệ thống sẽ xóa bài viết cần xóa khỏi database, load lại toàn bộ Blog.
	9. Chức năng Share Blog: Người dùng muốn share Blog của mình lên facebook, nhấn button "Share" để thực hiện.	10. Hệ thống share bài biết được yêu cầu lên facebook nếu người dùng đã đăng nhập tài khoản. Hoặc yêu cầu người dùng xác nhận tài khoản facebook nếu chưa đăng nhập trước đó.
Kết quả đạt được	Viết Blog, xóa Blog và share Blog yêu cầu.	

3.2.2.3. Các chức năng của Notification

Bảng 3.4. Mô tả chức năng Notification

Tên Usecase	Show infor Notification, Response Notification	
Mô tả/mục đích	Hiển thi các thông tin của các Invatition và các phản hồi thông tin từ Invitation. Đây là các thông tin từ người các đối tượng được mời đi ăn chung.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Notification	
		2. Hệ thống hiển thị các Notification nếu có
	3. Chức năng Show infor Notification: Người dùng nhấn vào Notification muốn xem chi tiết.	
		4. Hệ thống sẽ lấy các dữ liệu chi tiết của Notification yêu cầu ở database và hiển thị chi tiết lên dialog cùng với các button cho phép phản hồi: “accept”, “refuse”, “profile”.
	5. Chức năng Reponse Notification: Người dùng nhấn vào một trong các	

	button trên dialog để phản hồi lại thông tin cho người mời. 7. Khi có thông tin phản hồi, Nếu là thông tin mới, ở tab Notification sẽ hiển thị số lượng tin Notification mới.	6. Hệ thống sẽ phân loại và lưu lại nội dung phản hồi lên database, đồng thời gửi lại thông tin phản hồi cho người mời.
Kết quả đạt được	Xem các thông tin chi tiết của Notification, và phản hồi lại các kết quả cho người mời.	

3.2.2.4. Các chức năng của Profile

Bảng 3.5. Mô tả chức năng Profile

Tên Usecase	Edit Profile, Change password	
Mô tả/mục đích	Hiển thi các thông tin của Profile, cho phép người dùng chỉnh sửa các thông tin trong tài khoản người dùng như: avatar, name, age, gender, address, hobby, character. Chức năng change password cho phép thay đổi password.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Profile.	

		2. Hệ thống hiển thị các thông tin về tài khoản người dùng.
	3. Chức năng Edit Profile: Người dùng nhấn vào thông tin muốn thay đổi để tiến hành thay đổi.	
		4. Hệ thống sẽ hiển thị các dialog tương ứng với các nội dung muốn thay đổi.
	5. Nhập nội dung muốn thay đổi vào dialog, nhấn "OK" để hoàn thành thay đổi.	
		6. Hệ thống sẽ xác nhận lại thông tin phù hợp, sau đó lưu thông tin đã thay đổi lên database.
		7. Hiển thị lại thông tin đã thay đổi cho người dùng.
	8. Chức năng Change password, nếu người dùng muốn thay đổi mật khẩu, nhấn chage password, nhập các thông tin yêu cầu để được thay đổi mật khẩu mới.	

		9. Kiểm tra thông tin người dùng nhập vào, nếu đúng, cho phép thay đổi password, lưu password mới vào database.
Kết quả đạt được	Thay đổi thông tin trong profile của người dùng, và thay đổi mật khẩu mới.	

3.2.2.5. Chức năng trong Restaurant

Bảng 3.6. Mô tả chức năng Restaurant

Tên Usecase	Search Restaurant	
Mô tả/mục đích	Hiển thi các Restaurant phù hợp với địa điểm của người dùng trong hệ thống, và thông tin nhập vào để tìm kiếm của người dùng.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Restaurant	
		2. Hệ thống hiển thị danh sách các Restaurant phù hợp với địa điểm của người dùng.
	3. Chức năng Search Restaurant: Người dùng nhập địa chỉ cần tìm kiếm Restaurant vào ô search, để tiến hành tìm kiếm.	

		4. Hệ thống sẽ kiểm tra database, lấy những Restaurant phù hợp với yêu cầu tìm kiếm của người dùng, hiển thị danh sách Restaurant thích hợp.
Kết quả đạt được	Hiển thị danh sách Restaurant phù hợp với yêu cầu của người dùng.	

3.2.2.6. Các chức năng của Require

Bảng 3.7. Mô tả chức năng Require khi ở chế độ on

Tên Usecase	Edit status Off, Edit Require	
Mô tả/mục đích	Điều chỉnh trạng thái ở tab Require, trạng thái off sẽ cho phép người dùng chỉnh sửa thông tin require.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng.	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Require, mặc định ở trạng thái ở đây là off	2. Hệ thống kiểm tra xem trong cơ sở dữ liệu của máy đã có thông tin về Require chưa. 3. Nếu chưa, hệ thống sẽ phân tích dữ liệu từ thông tin Profile của người dùng để tạo dữ liệu

		Require thích hợp và lưu vào database người dùng. 4. Nếu đã có dữ liệu Require, hiển thị thông tin Require cho người dùng.
	5. Chức năng Edit Require: Người dùng nhấp vào thông tin muốn thay đổi.	
		6. Hệ thống sẽ hiển thị các dialog tương ứng với các nội dung muốn thay đổi.
	7. Nhập nội dung muốn thay đổi vào dialog, nhấn "OK" để hoàn thành thay đổi.	
		8. Hệ thống sẽ xác nhận lại thông tin phù hợp, sau đó lưu thông tin đã thay đổi lên database. 9. Hiển thị lại thông tin đã thay đổi cho người dùng.
Kết quả đạt được	Phân tích, tạo, lưu trữ và hiển thị thông tin Require ở trạng thái offline.	

Bảng 3.8. Mô tả chức năng Require khi ở chế độ on

Tên Usecase	Edit status On, Send Invitation, Response Invitation	
Mô tả/mục đích	Điều chỉnh trạng thái ở tab Require, trạng thái on sẽ cho phép người dùng tìm kiếm người đi ăn phù hợp với Require. Send Invitation: cho phép người dùng gửi lời mời cho đối tượng thích hợp. Chức năng Response Invitation: gửi phản hồi của người nhận đối với lời mời đi ăn.	
Yêu cầu tiên quyết	Đăng nhập thành công vào ứng dụng, xác định được Require mong muốn.	
Dòng sự kiện	Người dùng thực hiện	Hệ thống phản hồi
	1. Người dùng nhấn vào tab Require, điều chỉnh sang chế độ On.	2. Từ thông tin Require của người dùng, truy cập database Require đang tham gia tìm kiếm người ăn chung và database thông tin người dùng. Xác định những đối tượng phù hợp nhất với yêu cầu người dùng. 3. Hiển thị danh sách những người phù hợp về cho người dùng theo thứ tự độ phù hợp giảm dần.

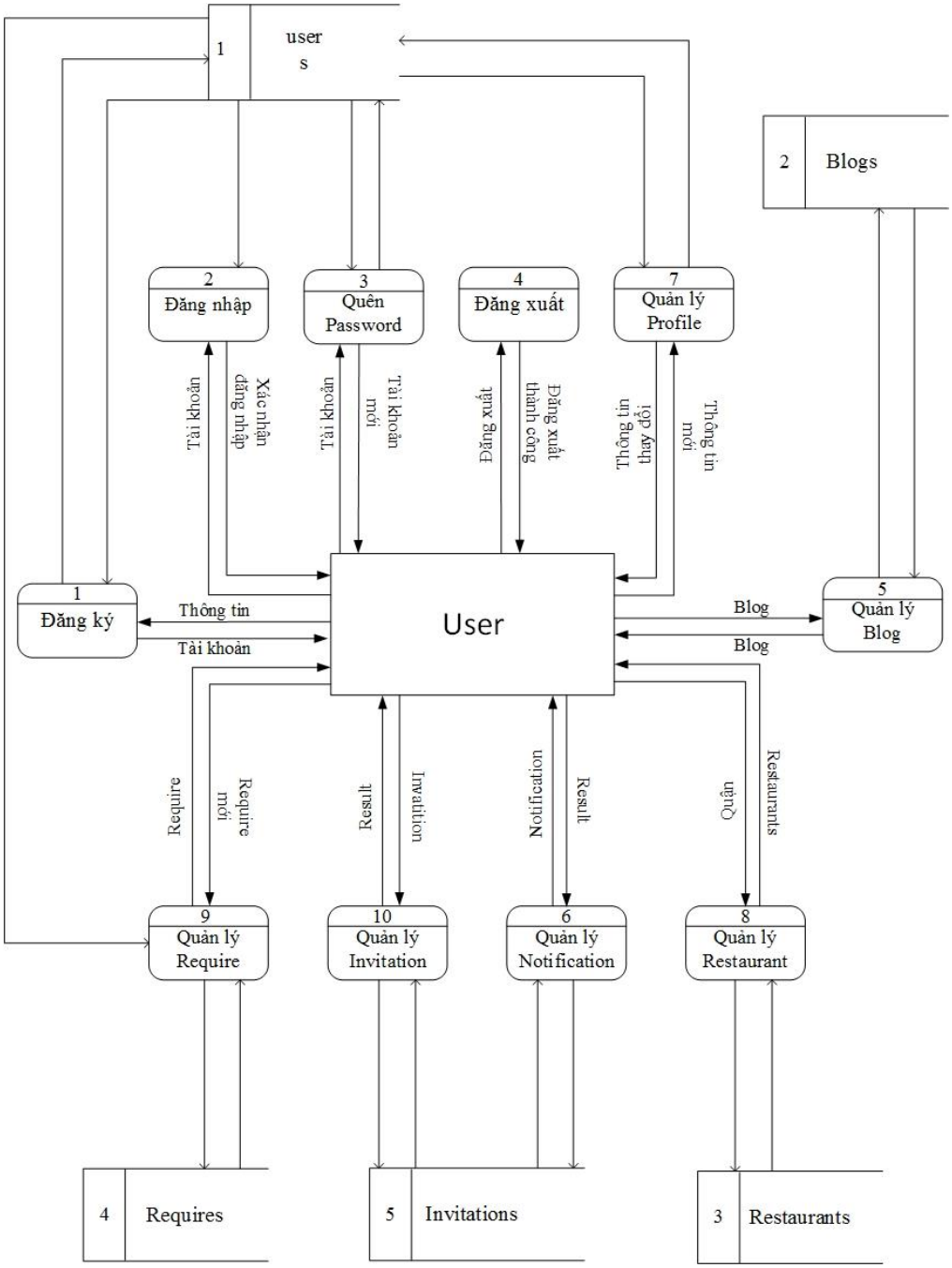
	4. Chức năng Send Invitation: Người dùng nhấn vào ImageButton "Invite" để xác thực chọn đối tượng mời đi ăn.	5. Hệ thống sẽ hiển thị dialog, trong đó có thời gian và địa điểm. 6. Thời gian được xác định là thời gian người dùng nhấn chọn đối tượng. 7. Địa điểm được xác định cách vị trí trung bình của người dùng và đối tượng khoảng 15km.
	8. Người dùng sau khi nhận được dialog thời gian, địa điểm có thể nhấn vào từng giá trị để thay đổi phù hợp.	9. Hệ thống nhận thông tin thời gian, địa điểm và lời mời lưu vào database. 10. Hệ thống gửi dialog gồm các thông tin chi tiết về lời mời cho đối tượng được mời. 11. Đồng thời tắt chức năng Invite của người dùng

		trong 5 phút để chờ kết quả lời mời từ đối tượng. Nếu sau 5 phút, không có thông tin phản hồi, người dùng được phép mời người khác.
	12. Chức năng Response Invatition: Người dùng sau khi nhận được dialog mời đi ăn chung bao gồm: thông tin người mời, thời gian, địa điểm và các button phản hồi. Người dùng gửi kết quả lời mời trong vòng 5 phút cho người mời. 13. Nếu sau 5 phút, người dùng không phản hồi kết quả cho người mời, lời mời hết giá trị, người dùng mất chức năng phản hồi kết quả với lời mời đó.	14. Sau khi nhận được kết quả của lời mời, hệ thống sẽ phân tích kết quả, lưu thông tin kết quả vào database.

		15. Đồng thời, gửi kết quả phản hồi về cho đối tượng mời.
Kết quả đạt được	Tìm kiếm được đối tượng mời đi ăn chung, gửi lời mời đi ăn chung, phản hồi kết quả lời mời cho người mời.	

3.3. Mô hình luồng dữ liệu

Mô hình luồng dữ liệu cho phép xác định được chi tiết mối liên hệ giữa dữ liệu với các chức năng được thực hiện trong ứng dụng. Biểu diễn biểu đồ mức khung cảnh các tác nhân của hệ thống và các luồng dữ liệu tổng quan.



Sơ đồ 3.3. Mô hình luồng dữ liệu ở mức đỉnh

3.4. Thiết kế cơ sở dữ liệu

3.4.1. Hệ thống cơ sở dữ liệu

Bảng 3.9. Cơ sở dữ liệu trong hệ thống

Tên cơ sở dữ liệu	Thuộc tính		Kiểu dữ liệu	Ý nghĩa
Users	Phone		String	Số điện thoại
	Full Name		String	Tên đầy đủ
	Password		String	Mật khẩu tài khoản
	Gender		String	Giới tính
	Avatar		String	Hình đại diện
	Birthday		String	Ngày, tháng, năm sinh
	Address		String	Địa chỉ hiện tại
	Latlng		String	Tọa độ địa chỉ
	Hobby		String	Sở thích
	Character		String	Tính cách
Blogs	Phone		String	Số điện thoại
	myBlog	Content	String	Nội dung Blog
		Feeling	String	Cảm xúc viết Blog
		Image	ArrayString	Hình ảnh đính kèm
		Date	String	Ngày viết Blog
	Name		String	Tên quán ăn

Restaurants	Address		String	Địa chỉ quán ăn
	Latlng		String	Tọa độ địa chỉ
	OpenDay		String	Thời gian mở cửa
Requires	Phone		String	Số điện thoại
	Full Name		String	Tên đầy đủ
	Gender		String	Số điện thoại
	Avatar		String	Tên đầy đủ
	Address		String	Địa chỉ hiện tại
	Latlng		String	Tọa độ địa chỉ
	Hobby Food		String	Sở thích về món ăn
	Hobby Character		String	Sở thích về tính cách
	Hobby Style		String	Sở thích về phong cách
Invatitions	Phone		String	Số điện thoại
	myInvatition	Phone Send	String	Số điện thoại gửi
		Name Send	String	Tên người gửi
		Time Send	String	Thời gian gửi
		Place	String	Địa điểm đi ăn
		Time	String	Thời gian đi ăn
		Result	String	
		Read	String	
		Send	String	

			Kết quả phản hồi Đã đọc lời mời Xem lời mời

3.4.2. Mô hình thực thể kết hợp (ERD)

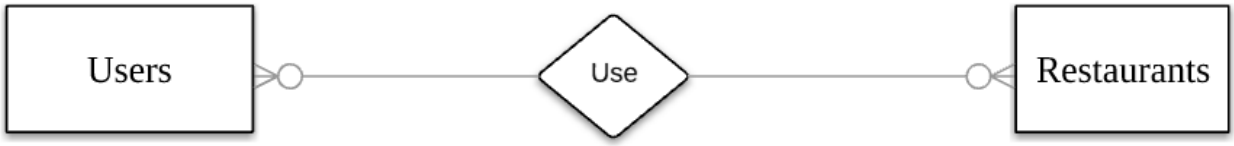
Mô hình thực thể kết hợp được sử dụng để biểu diễn cơ sở dữ liệu ở mức khái niệm, bao gồm có các thực thể, danh sách thuộc tính và những mối kết hợp.

Các thực thể trong ứng dụng: Users, Blogs, Restaurants, Requires, Invatitions.

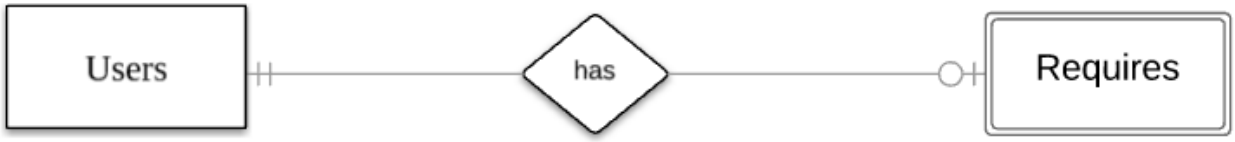
Mối kết hợp của các thực thể trong hệ thống:

Mỗi User viết một hoặc nhiều hoặc không viết Blog, nhưng mỗi Blog chỉ được viết bởi một User.

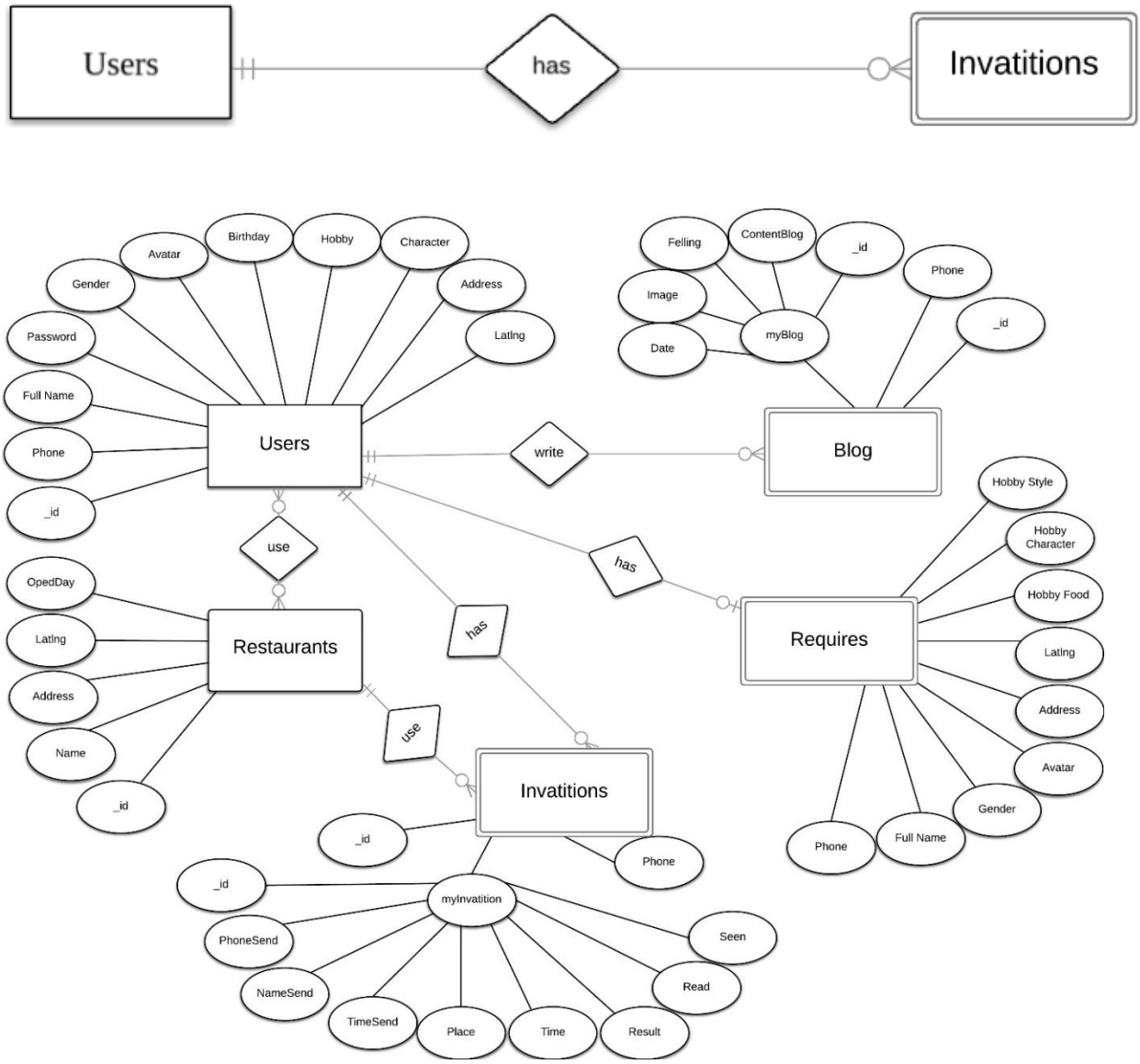
Mỗi User không sử dụng hoặc sử dụng một hoặc nhiều Restaurant và mỗi Restaurant không được sử dụng hoặc được sử dụng bởi một hoặc nhiều User.



Mỗi User không có hoặc có một Require và mỗi Require chỉ được tạo bởi một User.



Mỗi User không có, có một hoặc nhiều Invatition, nhưng mỗi Invatition chỉ được tạo bởi một User.



Sơ đồ 3.4. Mô hình thực thể kết hợp (ERD) cho ứng dụng

Chương 4. ỨNG DỤNG DON'T EAT ALONE

4.1. Tổng quan về ứng dụng Don't Eat Alone

Ứng dụng DEA được xây dựng trên nền tảng hệ điều hành Android. DEA là ứng dụng "tìm kiếm" và "sắp xếp một cuộc hẹn đi đi ăn chung" với một người lạ cùng sử dụng ứng dụng DEA.

Dựa trên những thông tin về độ tuổi, giới tính, sở thích, tính cách hay địa điểm mà người dùng cung cấp, DEA sẽ chọn lọc và đưa ra danh sách gợi ý những người dùng phù hợp với họ. Từ đó người dùng DEA có thể lựa chọn và quyết định gửi lời mời cho một buổi hẹn cùng đi ăn.

DEA tập trung quan tâm đến nhu cầu "Tìm một người cùng đi ăn" mà hầu như các ứng dụng hiện nay trên thị trường chưa khai thác và hỗ trợ. Mong muốn sử dụng sức mạnh công nghệ thông tin và internet để tăng cường sự kết nối giữa con người với nhau, tạo ra những trãi nghiệm thú vị, mới mẽ trong những mối quan hệ xã hội mới.

Một số chức năng chính mà DEA mang lại như:

- Chức năng quản lý tài khoản như đăng ký, đăng nhập, quên mật khẩu, đăng xuất.
- Chức năng quản lý blog.
- Chức năng quản lý profile.
- Chức năng quản lý require.
- Chức năng quản lý invitation.
- Chức năng quản lý notification.

4.2. Đặc tả giao diện của ứng dụng

4.2.1. Giao diện đăng ký

Người dùng trong ứng dụng DEA được quản lý thông qua tài khoản. Vì vậy để tham gia trải nghiệm các tính năng của DEA đòi hỏi người dùng cần phải đăng ký một tài khoản cá nhân.

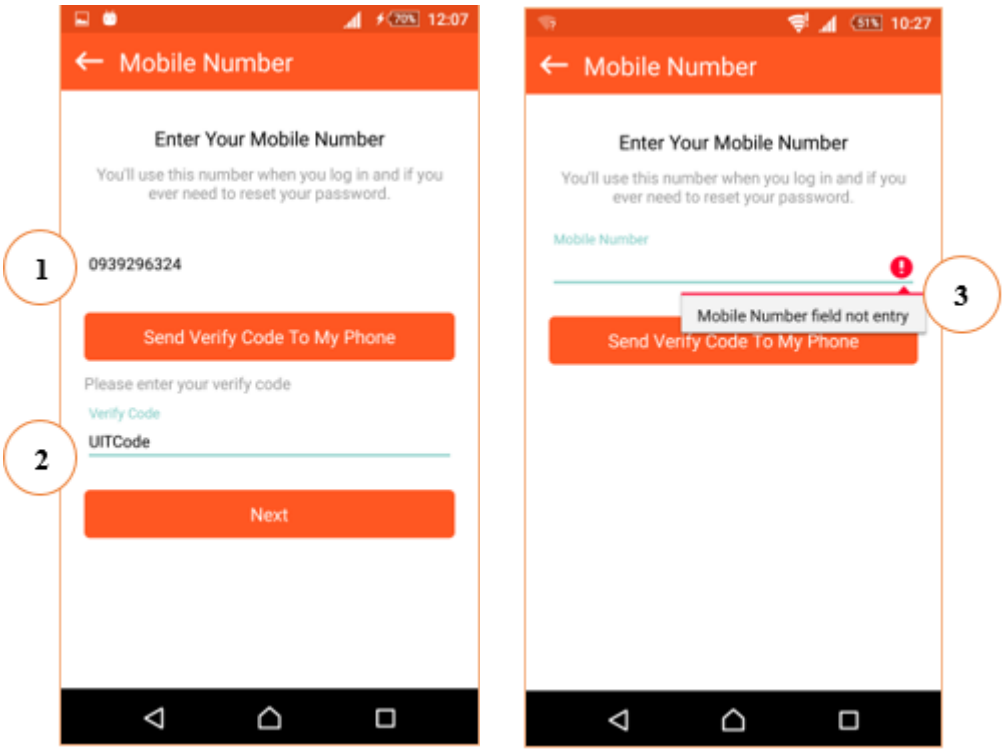
Giao diện đăng ký được thiết kế lấy thông tin theo từng bước. Ứng dụng yêu cầu thực hiện 7 bước để thực hiện đăng ký. Với kiểu thiết kế này, tại mỗi bước đều có

giải thích và nếu rõ mục đích sử dụng của thông tin mà người dùng cung cấp. Khác với các kiểu đăng ký nhanh với nhiều trường thông tin tập trung trong một giao diện, thiết kế này đảm bảo rằng người dùng có thể tập trung vào từng trường thông tin và cung cấp một cách chính xác nhất.

Giao diện đăng ký được thực hiện chủ yếu dựa trên các fragment kết hợp sử dụng CustomViewPager. Các Layout được thiết kế đơn giản nhưng cũng rất chuyên nghiệp thông qua việc sử dụng các đối tượng như LinearLayout, RelativeLayout, TextView, ImageView kết hợp với các thư viện custom EditText, WheelPicker.

4.2.1.1. Số điện thoại

Ứng dụng DEA sử dụng số điện thoại di động như tên tài khoản của người dùng. Nên để thực hiện đăng ký tài khoản, DEA sẽ yêu cầu bắt buộc đối với thông tin số điện thoại của người dùng.



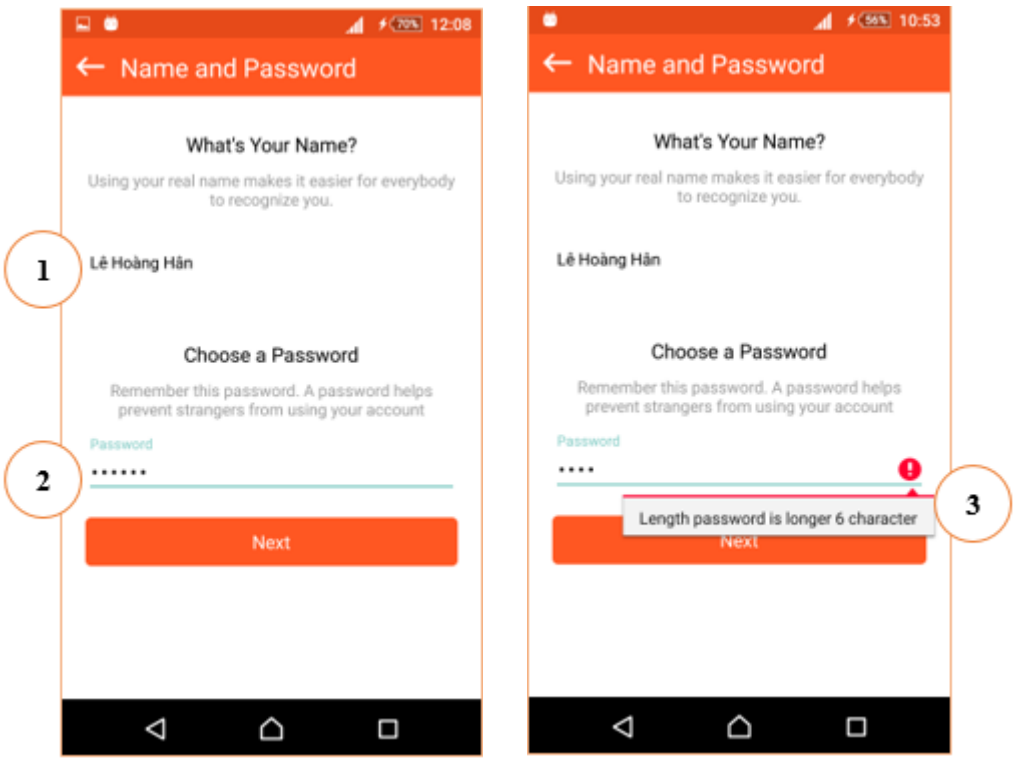
Hình 4.1. Giao diện đăng ký số điện thoại

Đặc tả giao diện:

- (1): Người dùng nhập số điện thoại. Trong trường hợp số điện thoại được nhập đúng định dạng và chưa được đăng ký trước đó, hệ thống sẽ gửi một mã xác nhận đến số điện thoại người dùng đã cung cấp. Ngược lại, khi kết quả kiểm tra là xảy ra lỗi, hệ thống sẽ xuất thông báo lỗi cho người dùng.
- (2): Mặc định được thiết lập cho phép ẩn đi và hiện lên khi người dùng cung cấp số điện thoại chính xác. Người dùng nhập mã xác nhận.
- (3): Thông báo lỗi cho người dùng khi số điện thoại nhập vào có lỗi.

4.2.1.2. Tên người dùng và mật khẩu

Thực hiện đăng ký tên người dùng và mật khẩu cho tài khoản.



Hình 4.2. Giao diện đăng ký tên người dùng và mật khẩu

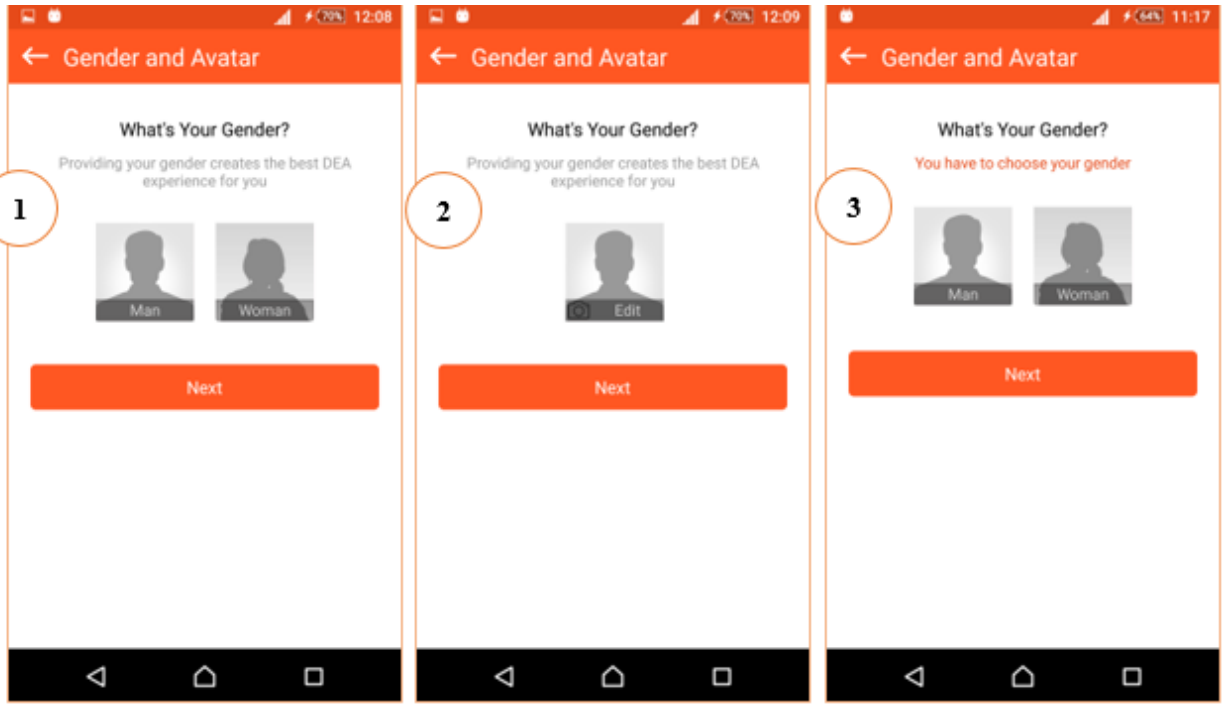
Đặc tả giao diện:

- (1): Người sử dụng nhập họ tên.

- (2): Chọn mật khẩu. DEA yêu cầu mật khẩu tối thiểu cần 6 ký tự và có phân biệt chữ in hoa.
- (3): Thông báo lỗi khi người dùng đăng ký mật khẩu không hợp lệ.

4.2.1.3. Giới tính và ảnh đại diện

Layout này cho phép người dùng được lựa chọn giới tính và cập nhật ảnh đại diện.



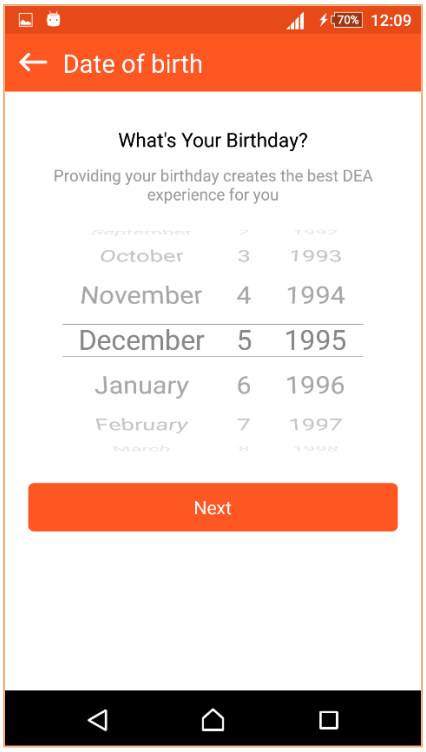
Hình 4.3. Giao diện đăng ký giới tính và cập nhật ảnh đại diện

Đặc tả giao diện:

- (1): Lựa chọn giới tính. Người dùng thực hiện click vào hình ảnh đại diện cho giới tính để thực hiện chọn giới tính phù hợp cho mình.
- (2): Cập nhật ảnh đại diện. Tiếp tục click vào ảnh để có thể cập nhật ảnh. Ứng dụng DEA cho phép người dùng được chọn ảnh đã có sẵn từ điện thoại hoặc thực hiện chụp một tấm ảnh ngay lúc đăng ký.
- (3) Thông báo lỗi nếu người dùng không thực hiện đúng yêu cầu trước khi muốn tiếp tục.

4.2.1.4. Ngày sinh

Cho phép người dùng được thực hiện đăng ký thông tin ngày sinh.



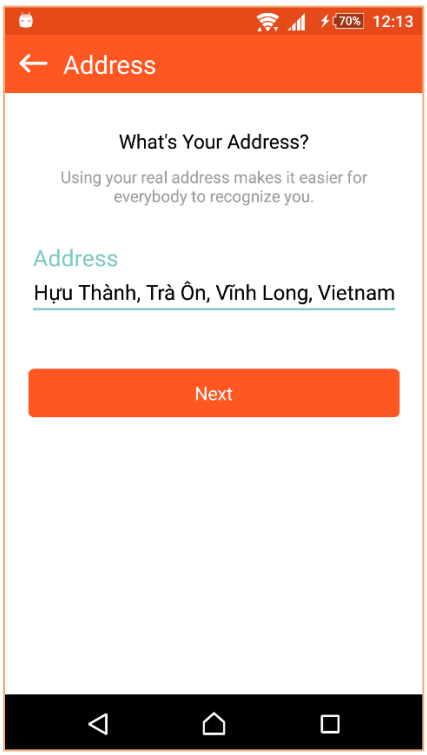
Hình 3.4. Giao diện đăng ký ngày sinh

Đặc tả giao diện:

- Thư viện WheelPicker cho phép thể hiện dữ liệu ngày, tháng, năm một cách nhỏ gọn, tiện lợi và thẩm mĩ giống như trong hệ điều hành iOS.
- Thay đổi giá trị lựa chọn bằng cách vuốt lên hoặc xuống.
- Dữ liệu được xây dựng đảm bảo xử lý logic về số ngày của tháng, tính toán năm nhuận, số ngày của tháng 2 trong năm bình thường và năm nhuận, giới hạn độ tuổi cho phép đăng ký.

4.2.1.5. Địa chỉ

Layout cho phép ứng dụng thực hiện lấy thông tin về địa chỉ của người dùng. Thông tin về địa chỉ chi tiết nhất sẽ giúp người dùng dễ dàng tìm kiếm người cùng đi ăn phù hợp nhất.



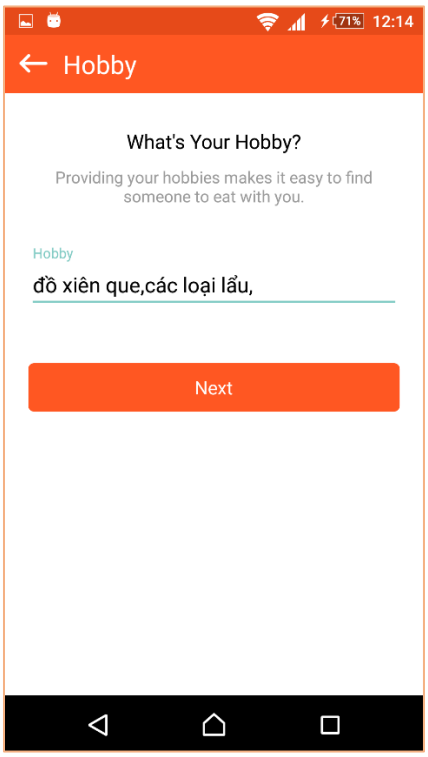
Hình 4.5. Giao diện đăng ký địa chỉ

Đặc tả giao diện:

- Sử dụng Google Place API và cụ thể là Place Autocomplete cho phép người dùng có thể cập nhật địa chỉ một cách tự động khi người dùng gỏ một vài ký tự
- Thông tin địa chỉ là không bắt buộc và có thể được cập nhật sau nên layout được thiết kế cho phép người dùng được lướt nhanh qua bước này.

4.2.1.6. Sở thích

Layout cho phép ứng dụng thực hiện lấy thông tin về sở thích của người dùng. Những thông tin về sở thích sẽ giúp người dùng trải nghiệm các chức năng của ứng dụng một cách tốt nhất.



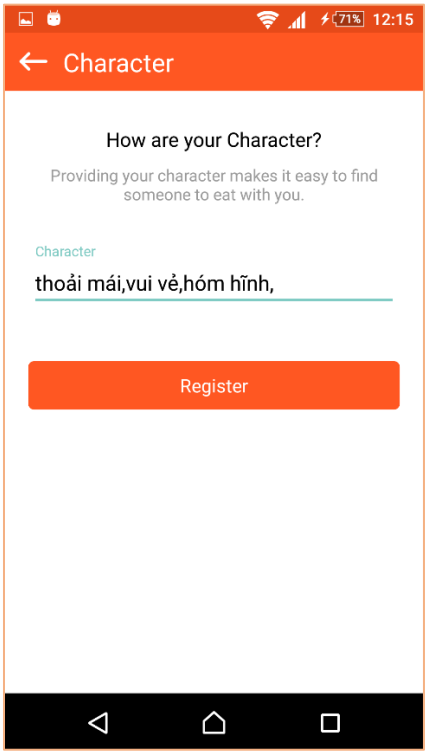
Hình 4.6. Giao diện đăng ký sở thích của người dùng

Đặc tả giao diện:

- Sử dụng Autocomplete TextView cho phép người dùng có thể cập nhật sở thích một cách tự động, nhanh chóng với các dữ liệu mà ứng dụng gợi ý.
- Thông tin sở thích là không bắt buộc và có thể được cập nhật sau nên layout được thiết kế cho phép người dùng được lướt nhanh qua bước này.

4.2.1.7. Tính cách

Layout cho phép ứng dụng thực hiện lấy thông tin về sở thích của người dùng. Những thông tin về sở thích sẽ giúp người dùng trải nghiệm các chức năng của ứng dụng một cách tốt nhất.



Hình 4.7. Giao diện đăng ký tính cách của người dùng

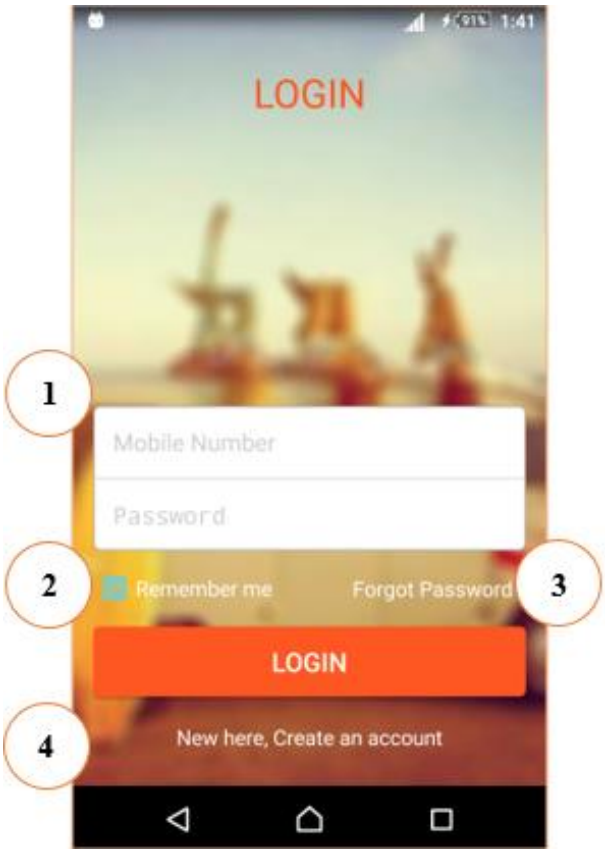
Đặc tả giao diện:

- Sử dụng Autocomplete TextView cho phép người dùng có thể cập nhật tính cách một cách tự động, nhanh chóng với các dữ liệu mà ứng dụng gợi ý.
- Thông tin về tính cách là không bắt buộc và có thể được cập nhật sau nên layout được thiết kế cho phép người dùng được lướt nhanh qua bước này.
- Đây là fragment cuối cùng, nên button được thay đổi thành “Register”.

4.2.2. Giao diện đăng nhập

Người dùng cần sử dụng tài khoản với thông tin số điện thoại và mật khẩu để thực hiện đăng nhập vào ứng dụng.

Giao diện đăng nhập được thiết kế đơn giản, thân thiện và dễ nhìn. Được thiết kế trên activity với background là ảnh đã được làm mờ. Layout được xây dựng thông qua việc sử dụng LinearLayout, RelativeLayout, TextView, ImageView và custom EditText.



Hình 4.8. Giao diện đăng nhập

Đặc tả giao diện:

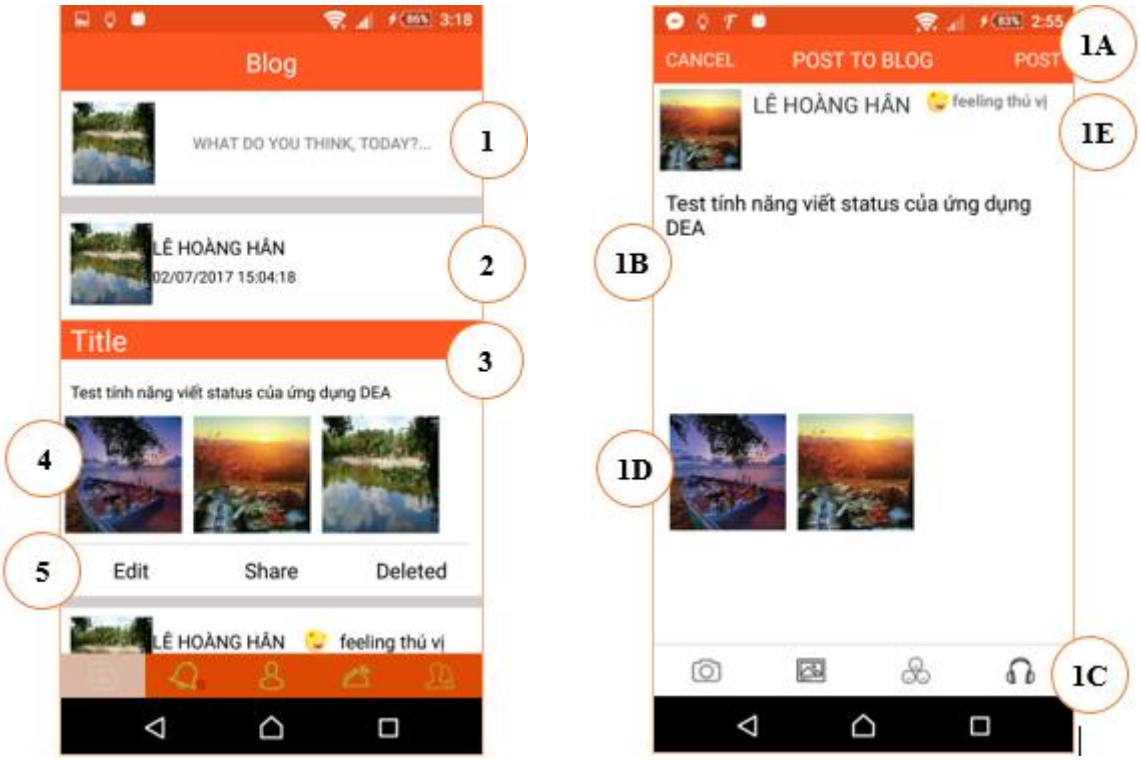
- (1): Thực hiện đăng nhập với số điện thoại và mật khẩu của người dùng. Khi thực hiện đăng nhập, hệ thống sẽ kiểm tra dữ liệu. Trong trường hợp có lỗi về định dạng hoặc sai mật khẩu, tài khoản không có thật ứng dụng sẽ thông báo lỗi ra cho người dùng
- (2): Chức năng nhớ tài khoản cho phép ghi nhớ tài khoản người dùng.
- (3): Chức năng quên mật khẩu cho phép người dùng reset lại mật khẩu của mình.

- (4): Cho phép đăng ký một tài khoản mới nếu người dùng chưa có tài khoản thông qua việc thực hiện link tới chức năng đăng ký tài khoản.

4.2.3. Giao diện quản lý blog

Giao diện Blog cho phép người dùng có thể xem lại những bài blog mà họ đã viết trong quá khứ hoặc có thể viết một blog mới để ghilại những suy nghĩ, câu chuyện, kỉ niệm của họ ở hiện tại.

Giao diện được thiết kế theo phong cách Material Design. Blog là một tab được xây dựng trong Viewpager. Ngoài các đối tượng thường được sử dụng trong xây dựng giao diện cho ứng dụng, giao diện Blog còn sử dụng các đối tượng như ImageButton, ScrollView, RecyclerView.



Hình 4.9. Giao diện chính của chức năng xem và viết blog

Đặc tả giao diện:

- (1): Cho phép người dùng viết blog. Khi click vào đây ứng dụng sẽ mở một activity để người dùng sử dụng chức năng viết blog, status.

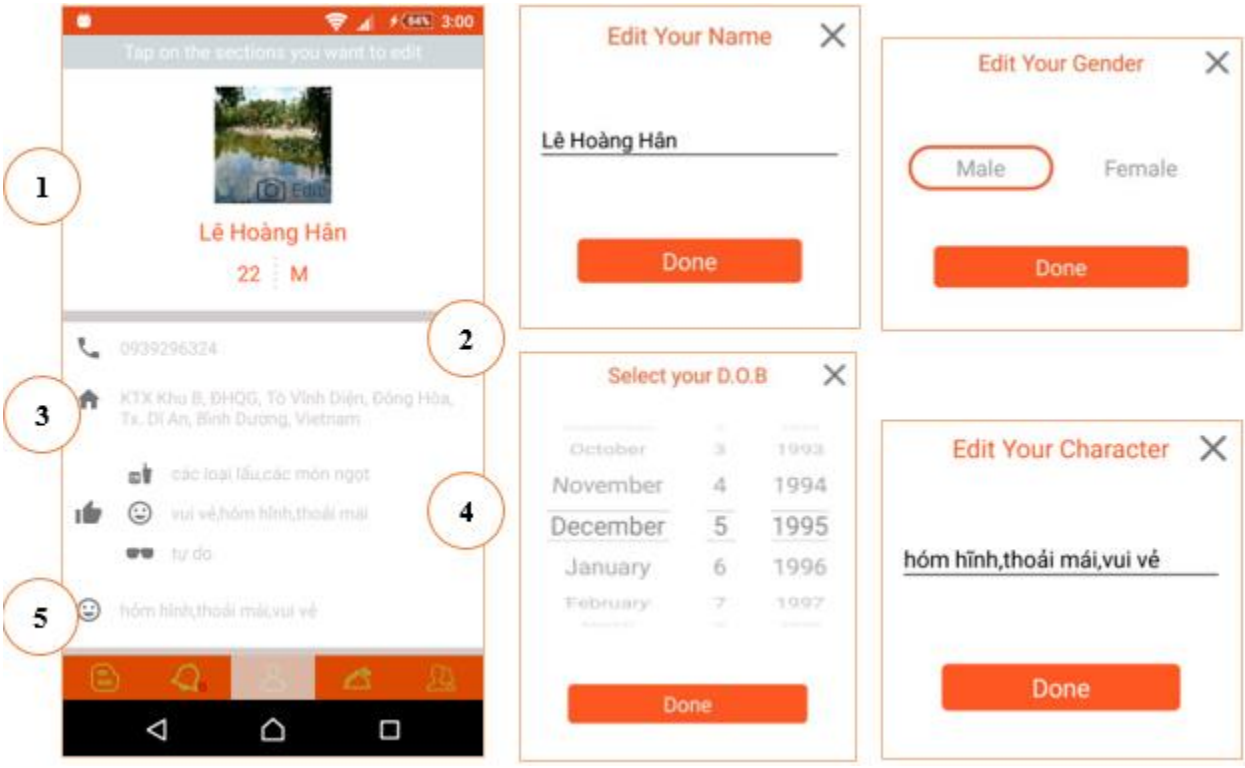
- (1A): button “Post” cho phép đăng nội dung blog lên giao diện xem Blog. Trong trường hợp khi người dùng chưa có nội dung, button “Post” sẽ được thiết lập ở dạng “disable”.
- (1B): Nội dung blog. Cho phép người dùng được trình bày nội dung bài blog một cách trực tiếp khi click vào vị trí này.
- (1C): các button tính năng giúp người dùng có thể thực hiện việc thêm ảnh từ album của điện thoại hoặc chụp mới một tấm ảnh; bày tỏ cảm xúc, hành động cho bài blog.
- (1D): Hiển thị danh sách ảnh vừa thêm trong HorizontalScrollView.
- (1E): Hiển thị cảm xúc cho bài blog.

- (2): Hiển thị thời gian đăng bài viết.
- (3): Hiển thị nội dung bài viết.
- (4): Hiển thị các hình ảnh trong bài viết.
- (5): Các button với các chức năng riêng biệt cho phép người dùng có thể thực hiện chỉnh sửa bài viết, chia sẻ bài viết sang một ứng dụng khác như facebok hay xóa bài viết.

4.2.4. Giao diện quản lý profile

Hiển thị các thông tin cá nhân của người dùng. Đồng thời cho phép sửa chữa các thông tin cá nhân.

Bên cạnh các layout và các đối tượng thường dùng như LinearLayout, RelativeLayout, TextView, ImageView giao diện còn sử dụng ScrollView và thực hiện custom dialog để thiết kế và xây dựng.



Hình 4.10. Giao diện profile và một số custom dialog để chỉnh sửa profile

Đặc tả giao diện:

- (1): Hiển thị những thông tin cơ bản của người dùng như tên, tuổi, giới tính và ảnh đại diện. Khi có nhu cầu sửa chữa, thay đổi những thông tin cá nhân nào người dùng có thể chạm trực tiếp vào thông tin đó. Ứng dụng sẽ lắng nghe các sự kiện khi người dùng thực hiện hành động click, sau đó gọi các custom dialog tương ứng đối với từng thông tin để hỗ trợ người dùng thực hiện sửa chữa và thay đổi thông tin.
- (2): Hiển thị thông tin về số điện thoại đã đăng ký của người dùng. Trường thông tin này không được phép thay đổi.
- (3): Hiển thị địa chỉ của người dùng. Sử dụng Place Autocomplete từ Google Place API hỗ trợ người dùng lấy địa điểm một cách nhanh chóng và tự động từ các gợi ý khi người dùng muốn thay đổi dữ liệu ở trường thông tin này.

- (4): Hiển thị các thông tin về sở thích của người dùng. Tương tự như ở (1), ứng dụng sẽ gọi các dialog để hỗ trợ người dùng thay đổi thông tin khi click vào từng trường thông tin tương ứng.
- (5): Hiển thị các thông tin về tính cách của người dùng. Tương tự như ở các thông tin khác, ứng dụng sẽ gọi các dialog để hỗ trợ người dùng thay đổi thông tin.

4.2.5. Giao diện quản lý require

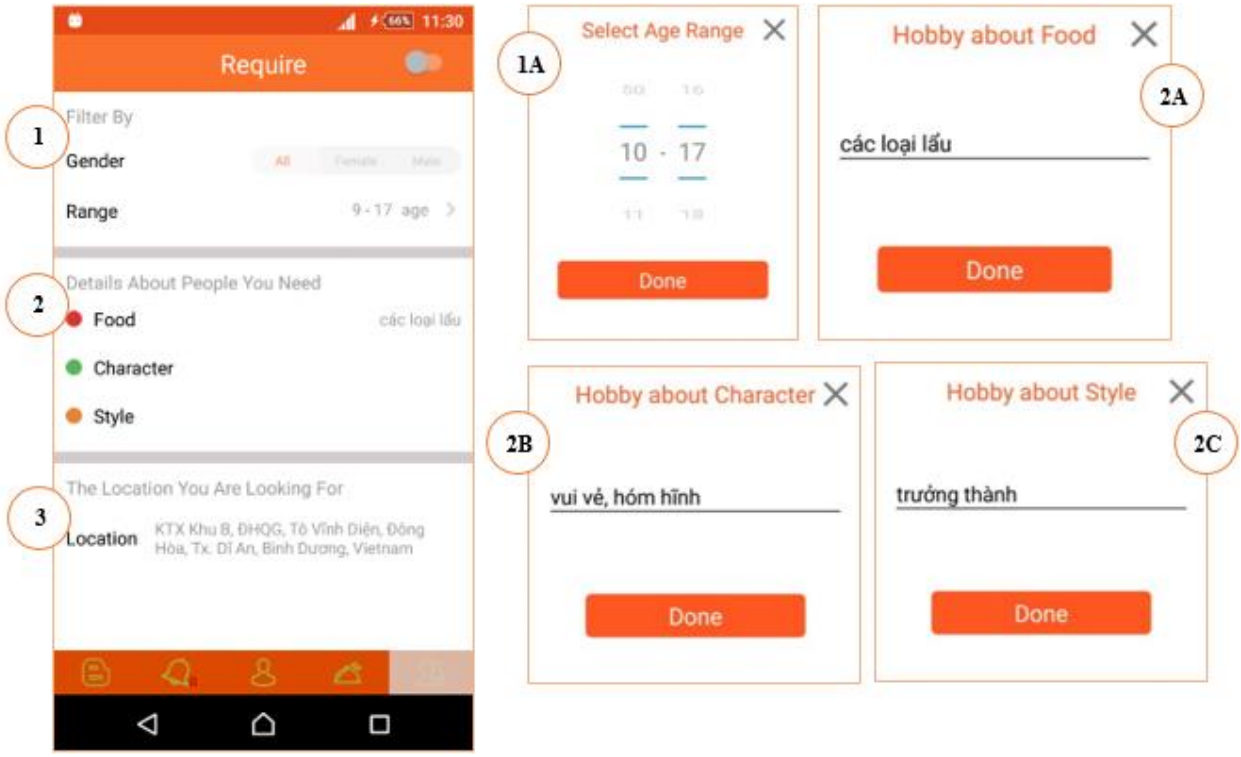
Thực hiện chức năng hỗ trợ người dùng tìm kiếm một người phù hợp cùng đi ăn chung dựa trên các yêu cầu tiêu chuẩn mà người dùng cung cấp.

Giao diện require được thiết kế với 2 chế độ off và on tương ứng với 2 giao diện khác nhau. Sự thay đổi qua lại giữa 2 chế độ được thực hiện thông qua một switch button.

4.2.5.1. Chế độ off

Giao diện require ở chế độ off được thiết kế nhằm hỗ trợ người dùng tùy chỉnh, thay đổi các thông tin yêu cầu tiêu chuẩn của họ đối với đối tượng muốn tìm kiếm.

Giao diện được thiết kế từ các LinearLayout, RelativeLayout, TextView, ImageView, các custom dialog và Place Autocomplete.



Hình 4.11. Giao diện Require ở chế độ off và các custom dialog tương ứng

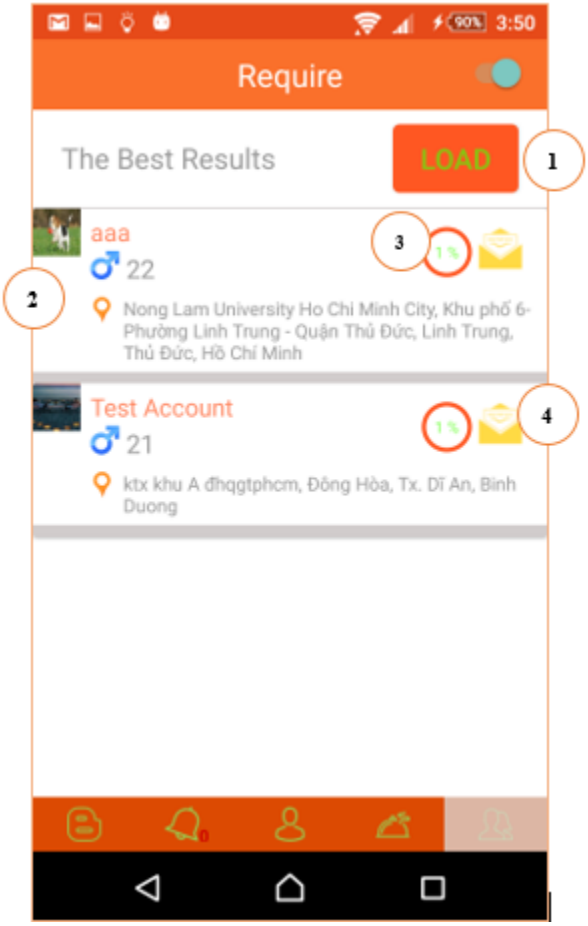
Đặc tả giao diện:

- (1): Tùy chỉnh thông tin cơ bản của đối tượng cần tìm kiếm. Người dùng có thể giới hạn phạm vi tìm kiếm thông qua việc tùy chỉnh thông tin đối tượng như giới tính và độ tuổi.
    - Ứng dụng cho phép người dùng được tìm kiếm các đối tượng có giới tính là nam hoặc nữ hoặc cả nam và nữ. Trong lần đăng nhập đầu tiên, ứng dụng thiết lập mặc định ở đây sẽ tìm kiếm cả nam lẫn nữ.
    - (1A): Dialog hỗ trợ người dùng thiết lập độ tuổi của đối tượng mong muốn tìm kiếm. Độ tuổi mặc định được thiết lập là trong khoảng hơn kém tuổi của người dùng 3 tuổi.
- (2): Tìm kiếm nâng cao với những điều kiện tìm kiếm như sở thích, tính cách, phong cách của đối tượng.
- (3): Đặt giới hạn tìm kiếm các đối tượng gần khu vực mà người dùng cung cấp.

4.2.5.2. Chế độ on

Giao diện require ở chế độ off được thiết kế để hiển thị kết quả tìm kiếm, đưa ra danh sách gợi ý những người dùng phù hợp nhất được sắp xếp theo mức độ phù hợp từ cao đến thấp.

Giao diện được xây dựng với một button và RecyclerView với mỗi item là thông tin của một người dùng phù hợp.



Hình 4.12. Giao diện require khi ở chế độ on

Đặc tả giao diện:

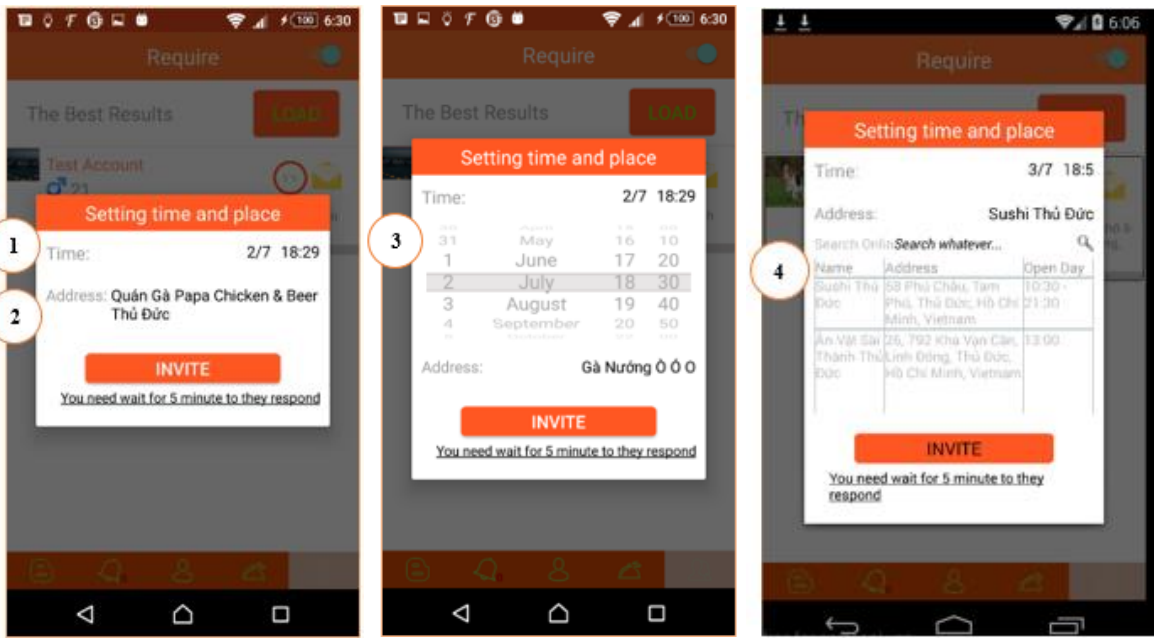
- (1): Load button hỗ trợ làm mới dữ liệu. Ngoài ra người dùng cũng có thể làm mới dữ liệu bằng cách thực hiện hành động vuốt xuống dưới.

- (2): Item thể hiện thông tin người dùng phù hợp. Hiển thị các thông tin của đối tượng như tên, giới tính, độ tuổi và vị trí hiện tại.
- (3): Mực độ phù hợp của đối tượng so với các yêu cầu tìm kiếm của người dùng. Mức độ này được đánh giá dưa trên thang điểm của các tiêu chí. Càng trùng khớp các thông tin thì mức độ phù hợp càng lớn.
- (4): Gửi lời mời cùng đi ăn đến đối tượng. Chuyển đến chức năng invitation.

4.2.6. Giao diện quản lý invitation

Chức năng invitation xây dựng giao diện trên các custom dialog, cho phép người dùng thực hiện thiết lập các thông tin cho cuộc hẹn và gửi lời mời cùng đi ăn chung đến người dùng khác.

Giao diện chủ yếu sử dụng LinearLayout, RelativeLayout, TextView, RecyclerView.

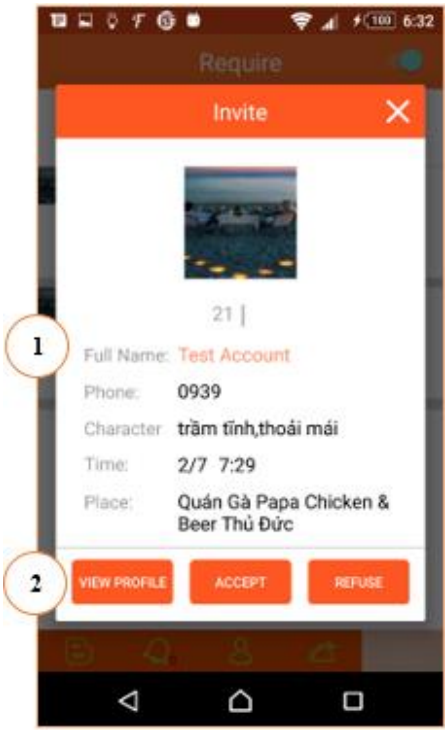


Hình 4.13. Thiết lập các thông tin cuộc hẹn trước khi gửi lời mời đến đối tượng

Đặc tả giao diện thiết lập cuộc thời gian và địa điểm trước khi gửi lời mời:

- (1): Hiển thị thời gian cuộc hẹn. Mặc định sẽ là thời giạn hiện tại. Người dùng cần chạm vào thông tin thời gian để mở/đóng thiết lập thời gian.

- (2): Hiển thị địa điểm hẹn. Ngườ dùng cần chạm vào thông tin địa chỉ để mở/đóng thiết lập địa chỉ.
- (3): Thiết lập thời gian cuộc hẹn.
- (4): Thiết lập địa điểm hẹn. Người dùng có thể tìm kiếm các nhà hàng từ danh sách gợi ý hoặc tìm kiếm các nhà hàng trên mạng từ cụ tìm kiếm trực tuyến.



Hình 4.14. Giao diện khi nhận được lời mời từ những người dùng khác

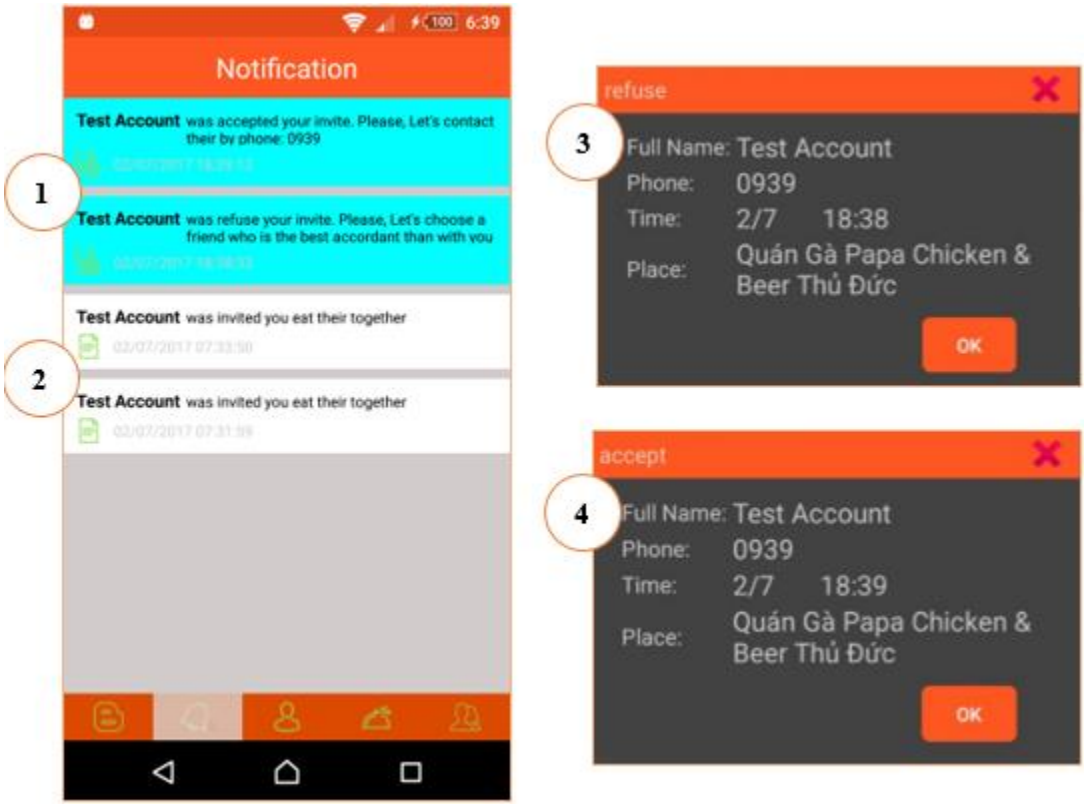
Đặc tả giao diện khi nhận được lời mời từ người dùng khác:

- (1): Hiển thị thông tin của người đã gửi lời mời đến người dùng. Các thông tin gồm: ảnh đại diện, tuổi, giới tính, họ tên, số điện thoại, tính cách, thời gian và địa điểm mà người đó muốn hẹn.
- (2): Các button chức năng
  - View Profile: Thực hiện xem thông tin chi tiết của đối tượng trước khi quyết định.
  - Accept: Đồng ý cuộc hẹn.
  - Refuse: Từ chối cuộc hẹn.

4.2.7. Giao diện quản lý notification

Giao diện Notification có chức năng hiển thị thông báo về các hoạt động, nội dung mà người dùng có liên quan như khi được người khác mời hoặc khi họ mời người khác đi ăn chung.

Giao diện notification sử dụng RecyclerView và các item được thiết kế với TextView, ImageView.



Hình 4.15. Giao diện notification

Đặc tả giao diện:

- Nội dung thông báo thể hiện tên đối tượng, nội dung thông báo và thời gian nhận thông báo
- (1): Những thông báo mà người dùng chưa đọc. Đối với những thông báo người dùng chưa xem qua sẽ được hiển thị với nền màu xanh.
- (2): Những thông báo mà người dùng đã đọc. Đối với những thông báo người dùng đã xem qua sẽ được hiển thị với nền màu trắng.

- (3) và (4): Nội dung thông báo. Khi người dùng click vào các item thông báo, hệ thống sẽ trả về dialog thể hiện chi tiết thông báo.

4.3. Kết quả đạt được

Sau quá trình xây dựng và phát triển ứng dụng DEA trên nền tảng android, chúng tôi đã đạt được kết quả nhiều kết quả tích cực.

Chúng tôi đã tự xây dựng server cho ứng dụng, hỗ trợ hiệu quả trong việc xử lý các yêu cầu cho ứng dụng.

Đối với các chức năng của ứng dụng. Chúng tôi đã hoàn thành cơ bản những mục tiêu, yêu cầu về chức năng đặt ra ban đầu. Đảm bảo ứng dụng hoạt động tốt và hiệu quả. Cụ thể ứng dụng DEA sở hữu những chức năng:

- Thực hiện việc quản lý tài khoản người dùng thông qua chức năng đăng ký, đăng nhập.
- Cho phép người dùng ứng dụng có thể xem lại hoặc viết mới blog, ghi lại những kỷ niệm đẹp, những cảm nghĩ của bản thân, cập nhật hình ảnh về cuộc hẹn đi ăn cùng người người bạn mới.
- Thực hiện quản lý profile cho phép người dùng có thể xem, cập nhật, thay đổi những thông tin cá nhân của họ.
- Tìm kiếm và gợi ý danh sách những người phù hợp theo yêu cầu của người dùng từ đó cho phép họ chia sẻ kế hoạch đi ăn, lựa chọn đối tượng phù hợp nhất và gửi lời mời cùng đi ăn chung đến những người cùng sử dụng ứng dụng.
- Gửi thông báo đến người dùng khi có những hoạt động liên quan đến họ.

Về mặt thiết kế giao diện người dùng. Ứng dụng DEA được thiết kế đơn giản, dễ nhìn, hợp lí và chuyên nghiệp bắt kịp với xu hướng thiết kế hiện tại của các ứng dụng di động trên thị trường. Sử dụng hiệu quả các layout, thư viện trong quá trình thiết kế và xây dựng giao diện cho ứng dụng.

Chương 5. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

5.1. Kết luận

Đề tài này được chúng tôi thực hiện sau thời gian nghiên cứu, tìm hiểu về các công nghệ, cơ sở lý thuyết đang được sử dụng phổ biến và hiệu quả hiện nay như Node.js, Socket.IO, Google API, Retrofit2 và MongoDB. Kết hợp với những kiến thức phát triển ứng dụng di động trên nền Android trong quá trình học tập tại trường, các kinh nghiệm thực tế qua đồ án lập trình ứng dụng mạng, đồ án trong quá trình làm việc và thực tập doanh nghiệp.

Kết quả đề tài này chính là sự ứng dụng các nghiên cứu, các kiến thức, kinh nghiệm vào việc thực hiện xây dựng ứng dụng di động hoàn chỉnh đảm bảo tính khoa học và tính mới. Ứng dụng với chức năng chính nhằm thực hiện tìm kiếm một người cùng đi ăn chung dựa trên những yêu cầu mà người dùng ứng dụng đưa ra.

Qua quá trình phân tích, thiết kế, xây dựng mô hình và thực hiện ứng dụng, nhóm chúng tôi đã xây dựng cơ bản hoàn thành ứng dụng DEA. Ứng dụng DEA đáp ứng được mục tiêu ban đầu mà nhóm đã đặt ra trong chương 1:

- DEA có đầy đủ các chức năng quản lý tài khoản; quản lý thông tin; quản lý blog; gửi notification; chia sẻ kế hoạch và tìm kiếm người cùng đi ăn chung.
- DEA có giao diện được thiết kế phù hợp, thẩm mĩ, thân thiện và dễ sử dụng. Được sử dụng nhiều layout, thư viện phù hợp để thiết kế.
- DEA sở hữu server riêng được xây dựng bằng Node.js, dữ liệu được quản lý, lưu trữ bằng MongoDB.

Tuy nhiên, do thời gian nghiên cứu, công nghệ, kỹ thuật vẫn còn hạn chế, ứng dụng DEA vẫn còn xuất hiện một số vấn đề tồn tại cần nghiên cứu khắc phục, cải thiện và mở rộng trong tương lai.

Tóm lại, trong đề tài này, chúng tôi đã xây dựng và phát triển hoàn thành ứng dụng DEA trên nền tảng android theo những mục tiêu ban đầu đặt ra. Điều này giúp mang đến giải pháp giải quyết vấn đề về nhu cầu chia sẻ kế hoạch và tìm một người phù hợp, sắp xếp một cuộc hẹn và mời người đó cùng đi ăn trong xã hội hiện nay. Chúng

tôi đã cố gắng phát triển ứng dụng với tinh thần, nỗ lực cao nhất và tin tưởng rằng ứng dụng DEA có nhanh chóng được mở rộng, cải thiện về chức năng cũng như giao diện để sớm được phổ biến và đưa vào sử dụng trong thực tiễn.

5.2. Hướng phát triển

Ứng dụng DEA đã đạt được những kết quả khả quan khi hoàn thành các mục tiêu ban đầu mà chúng tôi đặt ra. Tuy nhiên, để có thể trở thành một sản phẩm thương mại, có sức cạnh tranh với các ứng dụng trên thị trường. Đồng thời để đảm bảo về quy mô và chất lượng cho việc phát triển lên khóa luận tốt nghiệp. Chúng tôi đã định hướng cải thiện, khắc phục những hạn chế còn tồn tại cũng như phát triển, mở rộng ứng dụng.

- Thực hiện nâng cấp, cải thiện, khắc phục những hạn chế còn tồn tại như:
    - Gửi mã code về cho người dùng sử dụng module sim 808. Điều này sẽ giúp người dùng xác thực số điện thoại, chính là số mà người dùng đang sử dụng.
    - Xử lý chụp, cắt, chỉnh sửa ảnh trước khi chọn hình đại diện, chọn hình cho blog tùy theo ý của người dùng.
    - Ở chức năng profile, bổ sung thêm phần đánh giá yêu thích của các đối tượng người dùng khác dành cho profile của người bạn bất kỳ.
    - Hoàn thiện hơn về phần giao diện và luồng xử lý cho ứng dụng.
- Nghiên cứu và tiếp tục phát triển ứng dụng theo hướng:
    - Phát triển phần cho đánh giá về từng lần đi ăn chung và hiển thị tất cả lịch sử cuộc hẹn kèm nhận xét của người dùng trong chức năng History
    - Chia nội dung ở chức năng Restaurant ra hai tab: tab quán ăn và tab nhà hàng. Ở tab nhà hàng, người dùng có thể lựa chọn chỗ, thời gian và đặt bàn trên ứng dụng.
    - Xây dựng ứng dụng quản lý và liên kết với các nhà hàng để tương tác với nhà hàng thực hiện chức năng đặt bàn online ngay trên ứng dụng.
    - Phát triển ngôn ngữ tự nhiên để hoàn thành phần so sánh các sở thích trong yêu cầu tìm kiếm đối tượng đi ăn chung của người dùng.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] [Online]. Available: https://stackoverflow.com/. [Accessed 2017].

[2] [Online]. Available: https://freetuts.net/tong-quan-ve-mongodb-203.html. [Accessed 26 June 2017].

[3] [Online]. Available: http://smartjob.vn/gioi-thieu-he-quan-tri-co-so-du-lieu-nosql-mongodb/. [Accessed 26 June 2017].

[4] [Online]. Available: https://tedu.com.vn/mongo-db/mongodb-la-gi-tai-sao-lai-dung-mongodb-26.html. [Accessed 26 June 2017].

[5] [Online]. Available: https://www.mongodb.com/what-is-mongodb. [Accessed 25 June 2017].

[6] [Online]. Available: https://viblo.asia/p/su-dung-retrofit-va-rest-trong-java-bWrZnLdY5xw. [Accessed 23 June 2017].

[7] [Online]. Available: https://viblo.asia/p/su-dung-retrofit-va-rest-trong-java-bWrZnLdY5xw. [Accessed 22 June 2017].

[8] [Online]. Available: http://blog.rikkeisoft.com/seminar-gioi-thieu-ve-websocket-va-node-js/. [Accessed 21 June 2017].

[9] [Online]. Available: https://github.com/socketio/socket.io. [Accessed 21 June 2017].

[10] [Online]. Available: https://socket.io/. [Accessed 21 June 2017].

[11] [Online]. Available: https://nodejs.org/en/. [Accessed 20 June 2017].

[12] [Online]. Available: http://vietjack.com/nodejs/nodejs_la_gi.jsp. [Accessed 20 June 2017].