­Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

\_\_\_\_\_\_\_\_\_\*\_\_\_\_\_\_\_\_\_

**ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**XÂY DỰNG**

**ỨNG DỤNG GỌI XE GIAO HÀNG ONLINE**

Hoàng Khắc Hiếu

Hà Nội, 06/2018

Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ Thông Tin và Truyền Thông

Đồ án Tốt nghiệp Đại học

Xây dựng

ứng dụng Gọi xe Giao hàng Online

|  |  |
| --- | --- |
| SV thực hiện | Hoàng Khắc Hiếu |
| Người hướng dẫn | TS. Nguyễn Nhất Hải |

­­

Hà Nội, 05/2018

# Lời cam kết

Họ và tên SV: Hoàng Khắc Hiếu

Điện thoại liên lạc: 01636458600 Email: hoangkhachieu96@gmail.com

Lớp: CN – CNTT 01 – K59 Hệ đào tạo: Đại học chính quy

Em – *Hoàng Khắc Hiếu* – cam kết Đồ án Tốt nghiệp (ĐATN) là công trình nghiên cứu của bản thân em dưới sự hướng dẫn của thầy giao *TS. Nguyễn Nhất Hải*. Các kết quả nêu trong ĐATN là trung thực, là thành quả của riêng em, không sao chép theo bất kỳ công trình nào khác. Tất cả những tham khảo trong ĐATN – bao gồm hình ảnh, bảng biểu, số liệu, và các câu từ trích dẫn – đều được ghi rõ ràng và đầy đủ nguồn gốc trong danh mục tài liệu tham khảo. Em xin hoàn toàn chịu trách nhiệm với dù chỉ một sao chép vi phạm quy chế của nhà trường.

|  |  |
| --- | --- |
|  | *Hà Nội, ngày 28 tháng 05 năm 2018*  Tác giả ĐATN  *Hoàng Khắc Hiếu* |

# Lời cảm ơn

Để luận văn này đạt kết quả tốt đẹp, em đã nhận được sự hỗ trợ, giúp đỡ của nhiều cơ quan, tổ chức, cá nhân. Với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến tất cả các cá nhân và cơ quan đã tạo điều kiện giúp đỡ trong quá trình học tập và nghiên cứu đề tài.

Trước hết em xin gửi tới các thầy cô viện Công nghệ Thông tin và Truyền Thông, Trường đại học Bách Khoa Hà Nội lời chào trân trọng, lời chúc sức khỏe và lời cảm ơn sâu sắc. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của thầy cô, đến nay em đã có thể hoàn thành luận văn, đề tài với điều kiện tốt nhất.

Đặc biệt em xin gửi lời cảm ơn chân thành nhất tới thầy giáo – TS. Nguyễn Nhất Hải đã quan tâm giúp đỡ, hướng dẫn em hoàn thành tốt luận văn này trong thời gian qua.

Em xin bày tỏ lòng biết ơn đến lãnh đạo Trường Đại học Bách Khoa Hà Nội, các Khoa Phòng ban chức năng đã trực tiếp và gián tiếp giúp đỡ em trong suốt quá trình học tập và nghiên cứu đề tài.

Với điều kiện thời gian cũng như kinh nghiệm còn hạn chế của một sinh viên, Đồ án tốt nghiệp này không thể tránh được những thiếu sót. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy cô để em có điều kiện bổ sung, nâng cao kiến thức của mình, phục vụ tốt hơn công tác thực tế sau này.

Em xin chân thành cảm ơn !

# Tóm tắt

Hiện nay tại Việt Nam, với sự phát triển không ngừng của nền kinh tế kèm theo đó là sự chuyển đổi từ mô hình thương mại truyền thống sang thương mại điện tử ngày càng mạnh mẽ, nhu cầu vận chuyển hàng hóa của người dân ngày càng lớn. Tuy nhiên, với thách thức về giao thông do vấn đề mật độ dân cư cao, hạ tầng giao thông hạn chế nên không phải ai cũng có thể đầu tư phương tiện hàng vận chuyển riêng cho mình. Nhiều dịch vụ giao hàng, chuyển hàng ra đời nhưng do mô hình thiếu khả năng tiếp cận thị trường nên gặp khó khăn trong việc phân phối, cung cấp dịch vụ của mình trong khi đó phía người dân cũng phải loay hoay tìm kiếm thông tin mỗi khi có nhu cầu hàng vận chuyển, gây ra nhiều bất cập.

Chính vì nhu cầu cấp thiết đó, đồng thời nhận thấy tiềm năng về sự phổ biến của Interner và điện thoại thông mình tại Việt Nam, ĐATN được ra đời với mục tiêu xây dựng một hệ thống thông tin kết nối giữa tài xế cung cấp dịch vụ với người dân có nhu cầu vận chuyển hàng hóa. Hệ thống tận dụng các thế mạnh về công nghệ của thiết bị di động thông mình, dựa trên kết quả nghiên cứu nhu cầu thị trường và phân tích các ứng dụng tương tự đã có. Từ đó tập trung xây dựng một thiết kế tổng thể cho hệ thống, tích hợp các công nghệ mới, dịch vụ mới, hữu ích cho người sử dụng.

# Mục lục

[Lời cam kết iii](#_Toc514897208)

[Lời cảm ơn iv](#_Toc514897209)

[Tóm tắt v](#_Toc514897210)

[Mục lục vi](#_Toc514897211)

[Danh mục hình vẽ x](#_Toc514897212)

[Danh mục bảng xiii](#_Toc514897213)

[Danh mục các từ viết tắt xiv](#_Toc514897214)

[Chương 1 Giới thiệu đề tài 1](#_Toc514897215)

[1.1 Đặt vấn đề 1](#_Toc514897216)

[1.2 Mục tiêu và phạm vi đề tài 1](#_Toc514897217)

[1.3 Định hướng giải pháp 2](#_Toc514897218)

[1.4 Bố cục đồ án 4](#_Toc514897219)

[Chương 2 Khảo sát và phân tích yêu cầu 5](#_Toc514897220)

[2.1 Khảo sát hiện trạng 5](#_Toc514897221)

[2.2 Tổng quan chức năng 9](#_Toc514897222)

[**2.2.1 Biểu đồ use case tổng quan 9**](#_Toc514897223)

[**2.2.2 Biểu đồ usecase phân rã 11**](#_Toc514897224)

[**2.2.3 Quy trình nghiệp vụ 14**](#_Toc514897225)

[2.3 Đặc tả chức năng 16](#_Toc514897226)

[**2.3.1 Đặc tả use case UC001 – Đăng ký tài khoản 16**](#_Toc514897227)

[**2.3.2 Đặc tả use case UC002 – Đăng nhập tài khoản 17**](#_Toc514897228)

[**2.3.3 Đặc tả use case UC003 – Khách hàng thực hiện gọi xe 18**](#_Toc514897229)

[**2.3.4 Đặc tả use case UC004 – Nhận yêu cầu gọi xe 19**](#_Toc514897230)

[**2.3.5 Đặc tả use case UC005 – Kết thúc đơn hàng 19**](#_Toc514897231)

[**2.3.6 Đặc tả use case UC006 – Đánh giá quá trình sử dụng dịch vụ 20**](#_Toc514897232)

[**2.3.7 Đặc tả use case UC007 – Xem thông tin lịch sử sử dụng 21**](#_Toc514897233)

[**2.3.8 Đặc tả use case UC008 – Khóa người dùng 22**](#_Toc514897234)

[2.4 Các yêu cầu phi chức năng 23](#_Toc514897235)

[**2.4.1 Thân thiện, dễ dùng 23**](#_Toc514897236)

[**2.4.2 Hiệu suất 23**](#_Toc514897237)

[**2.4.3 Độ tin cậy 23**](#_Toc514897238)

[Chương 3 Công nghệ sử dụng 24](#_Toc514897239)

[3.1 Giải pháp bản đồ với Google Maps API 24](#_Toc514897240)

[**3.1.1 Tích hợp Google Map bằng Android API 26**](#_Toc514897241)

[**3.1.2 Xác định vị trí với Google Play services location API 27**](#_Toc514897242)

[**3.1.3 Xác định đường đi với Google Directions API 27**](#_Toc514897243)

[3.2 Xử lý dữ liệu theo thời gian thực với Firebase Realtime Database 30](#_Toc514897244)

[3.3 Xác thực người dùng với Firebase Authencations 32](#_Toc514897245)

[3.4 Lưu trữ dữ liệu với Firebase Storage 33](#_Toc514897246)

[3.5 Giải pháp thiết kế giao diện - Material Design 35](#_Toc514897247)

[Chương 4 Phát triển và triển khai ứng dụng 37](#_Toc514897248)

[4.1 Thiết kế kiến trúc 37](#_Toc514897249)

[**4.1.1 Lựa chọn kiến trúc phần mềm 37**](#_Toc514897250)

[**4.1.2 Thiết kế tổng quan 39**](#_Toc514897251)

[**4.1.3 Thiết kế chi tiết gói 40**](#_Toc514897252)

[4.2 Thiết kế chi tiết 43](#_Toc514897253)

[**4.2.1 Thiết kế giao diện 43**](#_Toc514897254)

[**4.2.2 Thiết kế cơ sở dữ liệu 46**](#_Toc514897255)

[4.3 Xây dựng ứng dụng 51](#_Toc514897256)

[**4.3.1 Thư viện và công cụ sử dụng 51**](#_Toc514897257)

[**4.3.2 Kết quả đạt được 52**](#_Toc514897258)

[**4.3.3 Mình hoạ các chức năng chính 53**](#_Toc514897259)

[**4.3.4 Giao diện màn hình gọi xe của khách hàng 54**](#_Toc514897260)

[Chương 5 Các giải pháp và đóng góp nổi bật 59](#_Toc514897261)

[5.1 Giải pháp luồng xử lý yêu cầu gọi xe của khách hàng 59](#_Toc514897262)

[**5.1.1 Giới thiệu bài toán 59**](#_Toc514897263)

[**5.1.2 Giải pháp xử lý 60**](#_Toc514897264)

[5.2 Bài toán tối ưu hóa quãng đường 63](#_Toc514897265)

[**5.2.1 Giới thiệu bài toán 63**](#_Toc514897266)

[**5.2.2 Giải pháp xử lý 64**](#_Toc514897267)

[Chương 6 Kết luận và hướng phát triển 66](#_Toc514897268)

[6.1 Kết luận 66](#_Toc514897269)

[6.2 Hướng phát triển 67](#_Toc514897270)

[Tài liệu tham khảo 68](#_Toc514897271)

# Danh mục hình vẽ

[Hình 1 Ứng dụng Grab trên GooglePlay (nguồn Google Play) 5](#_Toc514897272)

[Hình 2 Ứng dụng AhaMove trên GooglePlay (nguồn Google Play) 6](#_Toc514897273)

[Hình 3 Ứng dụng ShipVN trên Google Play (nguồn Google Play) 7](#_Toc514897274)

[Hình 4 Usecase tổng quát chức năng của hệ thống 10](#_Toc514897275)

[Hình 5 UC phân rã chức năng quản lý danh sách tài xế ưa thích 11](#_Toc514897276)

[Hình 6 UC phân rã cho chức năng quản lý Lịch sử 12](#_Toc514897277)

[Hình 7 UC phân rã cho chức năng quản lý Phương tiện 12](#_Toc514897278)

[Hình 8 UC phân rã cho chức năng quản lý Người dùng 13](#_Toc514897279)

[Hình 9 UC phân rã cho chức năng quản lý Dịch vụ 13](#_Toc514897280)

[Hình 10 Quy trình nghiệp vụ của Tài xế 14](#_Toc514897281)

[Hình 11 Quy trình nghiệp vụ của Khách Hàng 15](#_Toc514897282)

[Hình 12 Quy trình nghiệp vụ của Quản Trị Viên 15](#_Toc514897283)

[Hình 13 Giao diện Google Map trên nền Web 25](#_Toc514897284)

[Hình 14 Google Map API với nhiều dịch vụ hỗ trợ ( nguồn MapLink) 25](#_Toc514897285)

[Hình 15 Tích hợp Google Map vào thiết bị di động bằng API Google Maps 26](#_Toc514897286)

[Hình 16 Thiết lập đường đi bằng Direction API (nguồn Google MAP API) 28](#_Toc514897287)

[Hình 17 Cấu trúc 1 request (nguồn Google Map API) 28](#_Toc514897288)

[Hình 18 Đồng bộ hóa trực tiếp trên nhiều thiết bị với Realtime Database ( nguồn Google Firebase) 30](#_Toc514897289)

[Hình 19 Tổ chức dữ liệu dạng cây của Realtime Database 31](#_Toc514897290)

[Hình 20 Firebase Aunthencation hỗ trợ nhiều loại xác mình đăng nhập khách nhau (nguồn Firebase Aunthencation) 32](#_Toc514897291)

[Hình 21 Giao diện quản lý Storage trên Firebase Console 34](#_Toc514897292)

[Hình 22 Thiết kế Material Design nổi bật với các thẻ (nguồn Material Design) 35](#_Toc514897293)

[Hình 23 Mô hình MVC [Nguồn: Wikipedia] 38](#_Toc514897294)

[Hình 24 Mô hình MVP (nguồn Wikipedia) 39](#_Toc514897295)

[Hình 25 Thiết kế tổng quan hệ thống 40](#_Toc514897296)

[Hình 26 Thiết kế chi tiết gói chức năng Đăng ký 41](#_Toc514897297)

[Hình 27 Thiết kế chi tiết gói chức năng Đăng Nhập 41](#_Toc514897298)

[Hình 28 Thiết kế chi tiết gói chức năng Gọi xe 41](#_Toc514897299)

[Hình 29 Thiết kế chi tiết gói chức năng Quản lý dịch vụ 42](#_Toc514897300)

[Hình 30 Các gói hình ảnh cho các loại màn hình của Android 43](#_Toc514897301)

[Hình 31 Sắp xếp các đối tượng thẳng hàng xuyên suốt 44](#_Toc514897302)

[Hình 32 Nhóm các thông tin có đặc điểm chung lại với nhau 44](#_Toc514897303)

[Hình 33 Mẫu thiết kế các nút, điều khiển 45](#_Toc514897304)

[Hình 34 Mẫu quy chuẩn thông điệp 45](#_Toc514897305)

[Hình 35 Hai màu cơ bản của ứng dụng 46](#_Toc514897306)

[Hình 36 Sơ đồ thực thể liên kết 47](#_Toc514897307)

[Hình 37 Biểu đồ lớp đối tượng của hệ thống 48](#_Toc514897308)

[Hình 38 Giao diện màn hình đăng ký của ứng dụng 54](#_Toc514897309)

[Hình 39 Giao diện màn hình gọi xe của ứng dụng 55](#_Toc514897310)

[Hình 40 Giao diện màn hình nhận yêu cầu gọi xe của ứng dụng 56](#_Toc514897311)

[Hình 41 Giao diện màn hình lịch sử đặt xe của ứng dụng 56](#_Toc514897312)

[Hình 42 Giao diện màn hình tài xế ưa thích của ứng dụng 57](#_Toc514897313)

[Hình 43 Giao diện màn hình thông tin cá nhân của ứng dụng 57](#_Toc514897314)

[Hình 44 Giao diện màn hình thông báo của ứng dụng 58](#_Toc514897315)

[Hình 45 nhánh trạng thái - SHARED\_USER 60](#_Toc514897316)

[Hình 46 nhánh lắng nghe - DRIVER\_AVAIABLE 61](#_Toc514897317)

[Hình 47 Biểu đồ hoạt động gọi xe 62](file:///F:\DA2018\BaseCodeWeDelivery\Report\BaoCao-HoangKhacHieu-20146271-v2.0.docx#_Toc514897318)

[Hình 48 So sánh quãng đường khi chưa tối ưu và đã tối ưu 63](#_Toc514897319)

# Danh mục bảng

[Bảng 1 Các nhóm chức năng cần có của hệ thống 8](#_Toc514897320)

[Bảng 2 Đặc tả Usecase Đăng ký tài khoản 16](#_Toc514897321)

[Bảng 3 Đặc tả Usecase Đăng nhập tài khoản 17](#_Toc514897322)

[Bảng 4 Đặc tả Usecase Khách hàng thực hiện gọi xe 18](#_Toc514897323)

[Bảng 5 Đặc tả Usecase Nhận yêu cầu gọi xe 19](#_Toc514897324)

[Bảng 6 Đặc tả Usecase Kết thúc đơn hàng 19](#_Toc514897325)

[Bảng 7 Đặc tả Usecase Đánh giá quá trình sử dụng 20](#_Toc514897326)

[Bảng 8 Đặc tả Usecase Xem thông tin lịch sử sử dụng 21](#_Toc514897327)

[Bảng 9 Đặc tả Usecase Khóa người dùng 22](#_Toc514897328)

[Bảng 10 Ví dụ cấu trúc dữ liệu của chuỗi Response 29](#_Toc514897329)

[Bảng 11 Mô tả chi tiết lớp Account 49](#_Toc514897330)

[Bảng 12 Mô tả chi tiết lớp Vehicle 49](#_Toc514897331)

[Bảng 13 Mô tả chi tiết lớp Request 50](#_Toc514897332)

[Bảng 14 Mô tả chi tiết lớp Trip 50](#_Toc514897333)

[Bảng 15 Mô tả chi tiết lớp Wallert 51](#_Toc514897334)

[Bảng 16 Mô tả chi tiết lớp Service 51](#_Toc514897335)

[Bảng 17 Các thư viện và công cụ sử dụng 51](#_Toc514897336)

[Bảng 18 Thông tin thống kế về ứng dụng 53](#_Toc514897337)

[Bảng 19 Code tối ưu hóa tuyến đường 64](#_Toc514897338)

# Danh mục các từ viết tắt

|  |  |
| --- | --- |
| API | Application Programming Interface  Giao diện lập trình ứng dụng |
| **IDE** | Integrated Development Environment  Môi trường phát triển tích hợp |
| **CNTT** | Công nghệ thông tin |
| **ĐATN** | Đồ án tốt nghiệp |
| **SV** | Sinh viên |
| **UC** | Use Case |
| **MVP** | Model - View - Presenter. |
| **MVC** | Model – View - Controller |
| **GPS** | Global Positioning System.  Hệ thống Định vị Toàn cầu |

# Giới thiệu đề tài

## Đặt vấn đề

Ngày nay, với giá thành phù hợp và mức độ phổ biến rộng rãi, điện thoại di động thông mình đã trở thành một trong những công cụ liên lạc thiết yếu của con người. Ngoài chức năng liên lạc, với sự phát triển của công nghệ, kết nối toàn cầu, điện thoại di động thông mình được trang bị nhiều tính năng hữu ích, góp phần nâng cao hiệu của công việc, tiết kiệm thời gian, tiền bạc cho con người.

Hiện nay tại Việt Nam, với hơn một nửa dân số thường xuyên được tiếp cận với Internet, kèm theo đó là sự chuyển đổi từ mô hình thương mại truyền thống sang thương mại điện tử ngày càng mạnh mẽ, nhu cầu vận chuyển hàng hóa của người dân ngày càng lớn. Tuy nhiên, với thách thức về giao thông do vấn đề mật độ dân cư cao, hạ tầng giao thông còn hạn chế nên không phải ai cũng có thể đầu tư phương tiện vận chuyển hàng hóa riêng cho mình. Trong khi đó, do không có khả năng quảng cáo, tiếp cận thị trường nên những tài xế cung cấp dịch vụ vận chuyển hàng hóa đang gặp khó khăn trong việc cung cấp dịch vụ đến khách hàng có nhu cầu.

Chính vì nhu cầu cấp thiết đó, đồ án tập trung vào xây dựng, thiết kế tổng thể một ứng dụng cho phép tích hợp các công nghệ, dịch vụ mới, nhằm kết nối tài xế có khả năng vận chuyển hàng hóa với khách hàng có nhu cầu giao hàng.

## Mục tiêu và phạm vi đề tài

Để nắm bắt được nhu cầu thực tế của khách hàng và tài xế, SV đã tiến hành khảo sát thực tế và tổng hợp lại được một số thông tin như sau:

1. Phần lớn các ứng dụng, hệ thống được khảo sát chỉ mới giải quyết được một phần nhỏ nhu cầu của người dùng. Nhiều chức năng còn thiếu, chưa được hỗ trợ, một số ứng dụng còn khó sử dụng, …
2. Các khách hàng được khảo sát mong muốn có một hệ thống giúp họ kết nối với tài xế cung cấp dịch vụ giao hàng một cách nhanh chóng thuận tiện nhất. Hệ thống phải có giao diện đơn giản dễ dùng, cung cấp đầy đủ các chức năng. Các tài xế được kết nối phải có thông tin cá nhân rõ ràng, đảm bảo uy tín, chất lượng trong khâu vận chuyển. Các chi phí phát sinh trong hệ thống phải được tinh toán dựa trên một quy chuẩn chung với giá cả hợp lý, tiết kiệm.
3. Các tài xế được khảo sát mong muốn có một hệ thống hỗ trợ họ kết nối, mở rộng tập người dùng nhiều hơn. Giúp họ linh hoạt trong công việc, hỗ trợ đầy đủ các chức năng cơ bản, dựa trên loại hình dịch vụ, loại hình phương tiện mà họ cung cấp, sở hữu.

Với những thông tin trên và thực tế yêu cầu bài toán, SV đặt ra mục tiêu, nhiệm vụ cần giải quyết cho ĐATN như sau:

1. Áp dụng công nghệ có khả năng xử lý trực tuyến, cơ sở dữ liệu tập trung.
2. Chương trình chạy trên thiết bị di động.
3. Giao diện thân thiện, dễ sử dụng.
4. Tương tác tốt với người dùng, giúp người dùng dễ dàng thực hiện các thao tác.
5. Đa dạng loại hình dịch vụ, phương tiện hỗ trợ cho khách hàng và tài xế.
6. Tối ưu hóa chi phí cho khách hàng và tài xế
7. Đáp ứng yêu cầu về hiệu năng
8. Đáp ứng khả năng mở rộng của hệ thống.
9. Đáp ứng tốt các yêu cầu nghiệp vụ về quản lý hệ thống.
10. Hệ thống linh hoạt, có khả năng tùy biến cao.

## Định hướng giải pháp

Để giải quyết được những mục tiêu đặt ra ở mục 1.2, ứng dụng cần được tích hợp các giải pháp công nghệ sau:

1. Giải pháp phía Server – công nghệ dữ liệu thời gian thực Firebase Realtime Database: Ứng dụng không chỉ cho phép kết nối trực tiếp giữa khác hàng và tài xế mà còn hỗ trợ chia sẻ các thông tin về địa điểm một cách trực tuyến. Chính vì vậy yêu cầu của hệ thống đòi hỏi phải thực hiện tốt việc xử lý dữ liệu theo thời gian thực. Để thực hiện tính năng này, ứng dụng sử dụng Firebase Realtime Database để xử lý, đồng bộ hóa các dữ liệu giữa các thiết bị theo thời gian thực.
2. Giải pháp xác thực người dùng: Hệ thống không chỉ cho phép tra cứu các dịch vụ mà còn nhắm tới mục tiêu cung cấp các dịch vụ (có phí hay miễn phí) tới người dùng. Chính vì vậy, yêu cầu của hệ thống đòi hỏi phải thực hiện tốt việc xác thực người dùng. Để đáp ứng tính năng này, ứng dụng tích hợp giải pháp Firebase Authentication của Google.
3. Giải pháp lưu trữ tập tin, hình ảnh: Vì hệ thống cung cấp các dịch vụ liên quan đến các phương tiện giao thông (ô tô, xe máy,…) nên yêu cầu cung cấp các hình ảnh, mô tả về phương tiện, các loại giấy phép, giấy xác nhận, hình ảnh đại diện của người dùng,… Do đó, cần phải có một giải pháp lưu trữ tốt các tập tin, hình ảnh kể trên. Để thực hiện điều này, hệ thống sử dụng giải pháp Firebase Cloud Storage của Google.
4. Công nghệ bản đồ: Bản đồ là một thanh phần quan trọng của ứng dụng. Hầu hết các chức năng được xây dựng dựa trên việc xử lý , hiển thị thông tin trên bản đồ. Bản đồ cho phép người dùng tìm kiếm địa điểm, theo dõi việc di chuyển, kèm theo đó là các tính năng tự động lập tuyến đường, tính toán thời gian trung bình, ước lượng khoảng cách … Google Maps với lợi thế là nền tảng đáng tin cậy và được sử dụng bởi rất nhiều nhà phát triển khi muốn xây dựng các tính năng liên quan đến bản đồ. Chính vì vậy, SV đã quyết định sử dụng nền tảng này trong ứng dụng của mình.

Chi tiết các giải pháp công nghệ sẽ được phân tích trong Chương 3

## Bố cục đồ án

Phần còn lại của báo cáo ĐATN này được tổ chức như sau.

* Chương 2: Khảo sát và phân tích yêu cầu.

Nội dung chương 2 trình bày kết quả khảo sát các ứng dụng tương tự trên thị trường và phân tích yêu cầu của hệ thống để đưa ra các giải pháp cho bài toán.

* Chương 3: Công nghệ sử dụng.

Nội dung chương 3 trình bày về các công nghệ mà SV áp dụng vào để xây dựng hệ thống.

* Chương 4: Phát triển và triển khai ứng dụng

Nội dung chương 4 trình bày tổng quan về thiết kế ứng dụng dựa trên yêu cầu bài toán và các chức năng đưa ra

* Chương 5: Các giải pháp nổi bật

Nội dung chương 5 trình bày các đóng góp nổi bật của SV đối với đề tài

* Chương 6: Kết luận.

Nội dung chương 6 tóm tắt các yêu cầu đã đạt được đối với ĐATN và các dự định cho tương lai.

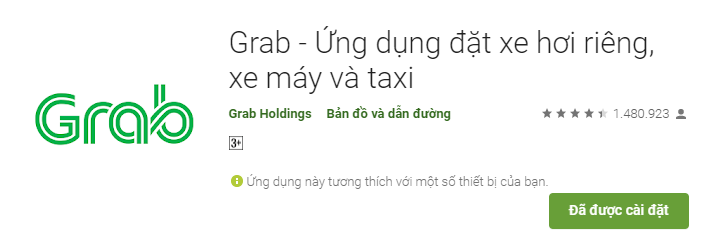
# ­­Khảo sát và phân tích yêu cầu

Tiếp nối Chương 1, ở chương này sẽ tập trung phân tích yêu cầu của bài toán dựa trên việc khảo sát các ứng dụng tương tự có trên thị trường, từ đó đưa ra được danh sách các chức năng cần có, các công nghệ cần thiết để xây dựng nên hệ thống.

## Khảo sát hiện trạng

Theo như bài toán đặt ra, chúng ta cần xây dựng một hệ thống có các chức năng cho phép tìm kiếm, kết nối và trao đổi thông tin giữa khách hàng có nhu cầu giao hàng với tài xế cung cấp các dịch vụ này. Hiện nay trên thị trường có một số hệ thống, ứng dụng có tính năng tương tự với yêu cầu bài toán, có thể kể đến như sau:

* **Grab giao hàng (Hình 1)**: Grab là ứng dụng gọi xe theo yêu cầu của công ty Grab. Ứng dụng cung cấp dịch vụ đa dạng dựa trên nhu cầu của khách hàng. Trong đó có dịch vụ giao hàng theo yêu cầu.



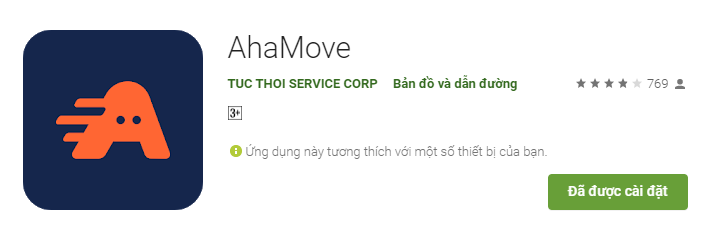
**Hình 1** Ứng dụng Grab trên GooglePlay (nguồn Google Play)

Các chức năng chính của ứng dụng:

1. Đăng nhập.
2. Quản lý thông tin cá nhân.
3. Gọi xe giao hàng theo yêu cầu.
4. Nhận yêu cầu giao hàng.
5. Quản lý lịch sử sử dụng.
6. Ví điện tử.
7. Theo dõi chuyến đi qua bản đồ.

Với lợi thế sử dụng chung nền tảng mạnh mẽ của Grab. Grab giao hàng có số lượng tài xế lớn, phủ rộng khắp với các nhiều chức năng đầy đủ, tiện ích. Tuy nhiên hiện nay Grab giao hàng vẫn còn một số điểm hạn chế như chỉ hỗ trợ giao hàng bằng xe máy, loại hình dịch vụ giao hàng còn hạn chế, giới hạn chỉ một điểm đến trên đơn hàng.

* **AhaMove (Hình 2):** AhaMove là ứng dụng đặt xe cho nhu cầu giao hàng. Kết nối khách hàng với đội ngũ tài xế xe giàu kinh nghiêm. Khi bạn cần chuyển các đơn hàng gấp hoặc cần lên lịch chuyển hàng chủ động, AhaMove sẽ giúp bạn đặt xe một cách hiệu quả.



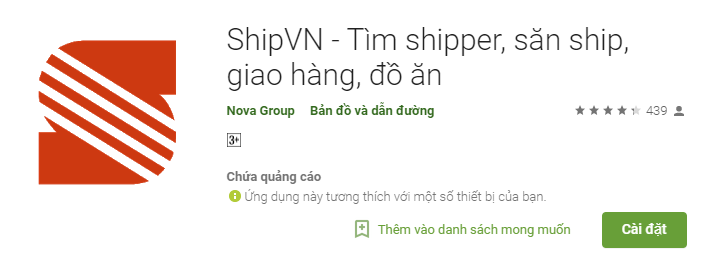
**Hình 2** Ứng dụng AhaMove trên GooglePlay (nguồn Google Play)

Các chức năng chính của ứng dụng:

1. Đăng nhập.
2. Quản lý thông tin cá nhân.
3. Gọi xe giao hàng theo yêu cầu.
4. Nhận yêu cầu giao hàng.
5. Quản lý lịch sử sử dụng.
6. Ví điện tử.
7. Theo dõi chuyến đi qua bản đồ.

Có thể nói AhaMove là một ứng dụng cung cấp đầy đủ các tính năng của một ứng dụng kết nối giao hàng, cung cấp đa dạng các loại dịch vụ giao hàng cho khách hàng lựa chọn với nhiều tính năng hay. Tuy nhiên ứng dụng vẫn chỉ dừng lại ở mức độ giao hàng bằng xe máy, chưa hỗ trợ liên kết với các tài xế xe tải.

* **ShipVN** (**Hình 3**): ShipVN là hệ thống dịch vụ phần mềm thông mình hỗ trợ tương tác giữa người bán hàng (shop) và người giao hàng (shipper) thông qua thiết bị di động (điện thoại) hoặc máy tính bảng.

****

**Hình 3** Ứng dụng ShipVN trên Google Play (nguồn Google Play)

Các chức năng chính của ứng dụng:

1. Đăng nhập.
2. Quản lý thông tin cá nhân.
3. Gọi xe giao hàng theo yêu cầu.
4. Nhận yêu cầu giao hàng.
5. Ví điện tử.

ShipVn là một ứng dụng cung cấp dịch vụ kết nối giữa các shipper với các shop có nhu cầu. Ứng dụng cung cấp đa dạng các loại hình giao hàng, chuyển hàng, liên kết đăng nhận tin yêu cầu giao hàng từ facebook. Tuy nhiên tương tự như hai ứng dụng khác ở trên, ShipVn vẫn còn một vài hạn chế như giao diện khó sử dụng, đối tượng hướng đến còn hạn chế, chỉ giới hạn ở quy mô nhỏ, chưa có hệ thống tính giá đồng bộ.

Từ việc khảo sát các ứng dụng kể trên, với việc kết hợp các ưu điểm của các ứng dụng với nhau và hạn chế các điểm yếu, SV đề xuất ra hệ thống các chức năng cần có cho một ứng dụng gọi xe giao hàng, đồng thời cũng đưa ra một số chức năng mới bổ sung, được chia thanh hai nhóm chức năng là nhóm chức năng cơ bản cần có và nâng cao, cụ thể như **Bảng 1**:

**Bảng 1** Các nhóm chức năng cần có của hệ thống

|  |  |  |
| --- | --- | --- |
|  | Tên chức năng | Mô tả |
| Nhóm chức năng cơ bản | Đăng nhập | Cho phép tài xế, khách hàng đăng nhập vào hệ thống bằng số điện thoại, gmail, facebook,… |
| Quản lý thông tin cá nhân | Cho phép khách hàng và tài xế quản lý các thông tin cá nhân của mình. Đối với tài xế cho phép quản lý loại hình dịch vụ cung cấp và quản lý thông tin về phương tiện. |
| Gọi xe giao hàng theo yêu cầu | Cho phép khách hàng thực hiện yêu cầu gọi xe giao hàng theo nhu cầu. Hệ thống sẽ kết nối khách hàng với tài xế gần nhất. |
| Nhận yêu cầu giao hàng | Khi tài xế sử dụng chức năng này, hệ thống sẽ kết nối tài xế với khách hàng ở xung quanh khu vực hiện tại đang có nhu cầu giao hàng. |
| Theo dõi chuyến đi qua bản đồ | Cho phép khách hàng cũng như tài xế theo dõi trực tiếp quá trình vận chuyển hàng thông qua giao diện bản đồ. Các thông tin về tuyến đường, khoảng cách, thời gian, vị trí hiện tại đều được thể hiện lên bản đồ. |
| Quản lý lịch sử sử dụng | Cho phép tài xế, khách hàng xem lại lịch sử sử dụng dịch vụ của mình. |
| Nhóm chức năng nâng cao | Quản lý ví điện tử | Cho phép tài xế, khách hàng tạo ví điện tử GrapPay để thanh toán cho các dịch vụ mà mình sử dụng |
| Giao hàng tiết kiệm | Cho phép khách hàng đặt nhiều điểm dừng trên mỗi yêu cầu giao hàng nhằm |
| Phân loại xe theo nhu cầu | Cho phép khách hàng chọn loại xe phù hợp với loại hàng hóa cần chuyển. Dựa trên kích thước và khối lượng của hàng hóa, khách hàng có thể chọn xe máy hoặc ô tô để tối ưu hóa chi phí dịch vụ |
| Gọi ý hành trình | thiết lập tuyến đường di chuyển ngắn nhất giữa các đơn hàng hiện có của tài xế. |

Việc xây dựng nên hệ thống các chức năng ở trên giúp ĐATN tập trung xây dựng nên một ứng dụng đầy đủ chức năng đáp ứng được nhu cầu của người dùng.

## Tổng quan chức năng

Hệ thống xây dựng với mục đích kết nối khách hàng có nhu cầu đi lại vận chuyển với tài xế cung cấp dịch vụ , là kênh trung gian giúp người dùng và lái xe có thể tìm kiếm thông tin. Dựa vào đó, ta thấy hệ thống cần có 3 tác nhân chính gồm: khách hàng, tài xế và quản trị viên. Dựa trên mỗi tác nhân mà có từng loại chức năng khác nhau.

### Biểu đồ use case tổng quan

Vai trò của mỗi tác nhân được thể hiện cụ thể:

1. **Khách hàng**: là đối tượng có nhu cầu về dịch vụ hàng vận chuyển. Do đó khách hàng có thể thực hiện gọi xe theo yêu cầu, xem lại lịch sử gọi xe của mình. Sau mỗi chuyến đi khách hàng có thể đánh giá mức độ hài lòng. Nếu đánh giá cao một số tài xế, khách hàng có thể thêm tài xế vào danh sách ưa thích để dễ dàng liên hệ.Bên cạnh đó, khách hàng còn có thể quản lý các thông tin cá nhân, phương thức thanh toán của mình.
2. **Tài xế**: là đối tượng cung cấp dịch vụ hàng vận chuyển. Do đó tài xế có thể nhận các yêu cầu gọi xe. Thông tin về vị trí điểm đi, điểm đến, tuyến đường sẽ được hiện thị lên bản đồ cùng với thông tin khách hàng cũng như chi phí của chuyến đi. Ngoài ra, tài xế có thể quản lý dịch vụ mình cung cấp, quản lý thông tin cá nhân, thông tin phương tiện. Hệ thống cũng cho phép tài xe xem lại lịch sử hoạt động, thực hiện đánh giá chất lượng sau mỗi chuyến đi
3. **Quản trị viên**: Quản trị viên sẽ có thể quản lý thông tin của khách hàng và tài xế. Khi có một số vấn đề cần được thông báo cho người dùng, quản trị viên có thể gửi thông báo đến họ. Ngoài ra quản trị viên có thể quản lý việc sử dụng các loại mã khuyến mãi, giảm giá.

Từ vai trò của các tác nhân ở trên, ta có thể xây dựng nên biểu đồ usecase tổng quát của hệ thống như **Hình 4**:



**Hình 4** Usecase tổng quát chức năng của hệ thống

### Biểu đồ usecase phân rã

#### Biểu đồ usecase phân rã chức năng Quản lý danh sách tài xế ưa thích

Sau mỗi chuyến đi nếu khách hàng cảm thấy hài lòng với dịch vụ mà tài xế mang lại và có nhu cầu làm việc tiếp trong thời gian tới thì khách hàng có thể đánh dấu tài xế vào mục ưa thích. Với chức năng này, khách hàng có thể xem danh sách tài xế ưa thích, thêm, sửa thông tin, xóa, liên hệ với tài xế ưa thích. (**Hình 5**)



**Hình 5** UC phân rã chức năng quản lý danh sách tài xế ưa thích

#### Biểu đồ usecase phân rã cho chức năng quản lý lịch sử gọi xe

Sau mỗi chuyến đi hệ thống tự động ghi lại lịch sử. Người dùng có thể sử dụng chức năng này để xem lại thông tin về chuyến đi của mình, thêm lịch sử mới. Ngoài ra chức năng này cũng cung cấp khả năng sửa, xóa lịch sử các chuyến đi (**Hình 6**).

****

**Hình 6** UC phân rã cho chức năng quản lý Lịch sử

#### Biểu đồ usecase phân rã cho chức năng quản lý phương tiện

Với chức năng quản lý phương tiện, tài xế có thể thực hiện các thao tác xem, sửa, xóa phương tiện thông qua hệ thống (**Hình 7**).

****

**Hình 7** UC phân rã cho chức năng quản lý Phương tiện

#### Biểu đồ usecase cho chức năng quản trị người dùng

Hệ thống cung cấp cho quản trị viên bộ tính năng quản lý người dùng, cho phép quản trị viên có thể tìm kiếm người dùng, xem thông tin, chặn những người dùng xấu hoặc bỏ chặn (**Hình 8**)

****

**Hình 8** UC phân rã cho chức năng quản lý Người dùng

#### Biểu đồ usecase cho chức năng quản lý dịch vụ

Chức năng quản lý dịch vụ cho phép quản trị viên quả lý các dịch vụ vận chuyển, đi lại và các chương trình khuyến mãi mà hệ thống cung cấp. Quản trị viên có thể thêm, sửa thông tin, đóng mở các loại dịch vụ.(**Hình 9**)

****

**Hình 9** UC phân rã cho chức năng quản lý Dịch vụ

### Quy trình nghiệp vụ

Với 3 đối tượng sử dụng hệ thông có 3 luồng quy trình nghiệp vụ chính là:

1. Quy trình sử dụng ứng dụng của tài xế.
2. Quy trình sử dụng dịch vụ của khách hàng.
3. Quy trình sử dụng dịch vụ của quản trị viên.

Cụ thể các quy trình nghiệp vụ như sau:

* Quy trình nghiệp vụ của tài xế (**Hình 10**)



**Hình 10** Quy trình nghiệp vụ của Tài xế

Người dùng có thể đăng ký và lựa chọn loại người dùng là tài xế. Sau khi đăng ký tài khoản, đăng ký thông tin phương tiện và đăng nhập với vai trò tài xế vào hệ thống. Tài xế có thể thực hiện nhận các yêu cầu dịch vụ hàng vận chuyển, quản lý lịch sử, quản lý các thông tin cá nhân,…

* Quy trình nghiệp vụ của khách hàng (**Hình 11**)



**Hình 11** Quy trình nghiệp vụ của Khách Hàng

Sau khi đăng ký và đăng nhập với vai trò khách hàng vào hệ thống. Khách hàng có thể bắt đầu sử dụng các dịch vụ của hệ thống như gọi xe theo yêu cầu, quản lý các thông tin cá nhân, quản lý lịch sử, …

* Quy trình nghiệp vụ của quản trị viên (**Hình 12**)



**Hình 12** Quy trình nghiệp vụ của Quản Trị Viên

Sau khi đăng nhập vào hệ thống thành công, quản trị viên có thể thực hiện các chức năng quản trị người dùng, quản trị dịch vụ.

## Đặc tả chức năng

### Đặc tả use case UC001 – Đăng ký tài khoản

**Bảng 2** Đặc tả Usecase Đăng ký tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC001 | **Tên usecase** | Đăng ký tài khoản |
| **Tác nhân** | Người dùng | | |
| **Mô tả** | Usecase mô tả quá trình người dùng đăng ký tài khoản trong hệ thống | | |
| **Tiền điều kiện** | Người dùng yêu cầu đăng ký tài khoản | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Hệ thống | Hiển thị form nhập số điện thoại | | 2 | Người dùng | Nhập số điện thoại và gửi yêu cầu đăng ký | | 3 | Hệ thống | Kiểm tra số điện thoại | | 4 | Hệ thống | Gửi mã xác nhận đến đến số điện thoại và hiển thị form điền mã xác nhận | | 5 | Người dùng | Điền mã xác nhận và gửi yêu cầu | | 6 | Hệ thống | Kiểm tra mã xác nhận | | 7 | Hệ thống | Chuyển vào giao diện chính của ứng dụng | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 4a | Hệ thống | Thông báo lỗi nếu số điện thoại nhập vào là sai | | 7a | Hệ thống | Thông báo lỗi nếu nhập sau mã xác nhận | | | |
| **Hậu điều kiện** | Đăng ký tài khoản thanh công hoặc thất bại | | |

### Đặc tả use case UC002 – Đăng nhập tài khoản

**Bảng 3** Đặc tả Usecase Đăng nhập tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC002 | **Tên usecase** | Đăng nhập |
| **Tác nhân** | Người dùng | | |
| **Mô tả** | Usecase mô tả quá trình người dùng đăng nhập tài khoản trong hệ thống | | |
| **Tiền điều kiện** | Người dùng yêu cầu đăng nhập hệ thống | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Hệ thống | Hiển thị form nhập số điện thoại | | 2 | Người dùng | Nhập số điện thoại và gửi yêu cầu đăng ký | | 3 | Hệ thống | Kiểm tra số điện thoại | | 4 | Hệ thống | Gửi mã xác nhận đến đến số điện thoại và hiển thị form điền mã xác nhận | | 5 | Người dùng | Điền mã xác nhận và gửi yêu cầu | | 6 | Hệ thống | Kiểm tra mã xác nhận | | 7 | Hệ thống | Chuyển vào giao diện chính của ứng dụng | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1a.1 | Hệ thống | Hiển thị form chọn tài khoản gmail | | 1a.2 | Người dùng | Chọn tài khoản facebook/ email. | | 1a.3 | Hệ thống | Kiểm tra tài khoản và đăng nhập vào hệ thống | | 3a | Hệ thống | Thông báo lỗi nếu số điện thoại nhập vào là sai | | 6a | Hệ thống | Thông báo lỗi nếu nhập sau mã xác nhận | | | |
| **Hậu điều kiện** | Đăng nhập thành công hoặc thất bại | | |

### Đặc tả use case UC003 – Khách hàng thực hiện gọi xe

**Bảng 4** Đặc tả Usecase Khách hàng thực hiện gọi xe

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC003 | **Tên usecase** | Khách hàng thực hiện gọi xe |
| **Tác nhân** | Khách hàng | | |
| **Mô tả** | Usecase mô tả quá khách hàng sử dụng dịch vụ gọi xe theo yêu cầu | | |
| **Tiền điều kiện** | Khách hàng đã đăng nhập vào hệ thống | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Người dùng | Chọn loại hình vận chuyển và loại phương tiện vận chuyển | | 2 | Người dùng | Nhập vào điểm đi | | 3 | Người dùng | Nhập vào điểm đến và thông tin người nhận tại địa điểm này | | 4 | Người dùng | Gửi yêu cầu gọi xe | | 5 | Hệ thống | Hiển thị form xác nhận yêu cầu gọi xe | | 6 | Người dùng | Xác nhận yêu cầu | | 7 | Hệ thống | Tìm kiếm và kết nối khách hàng với tài xế gần nhất | | 8 | Hệ thống | Khi tài xế nhận yêu cầu gọi xe, hệ thống hiển thị thông tin về vị trí, thông tin cá nhân của tài xế, phương tiện, thời gian ước tính từ vị trí của tài xế đến vị trí của khách hàng. | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 7a | Người dùng | Hủy yêu cầu | | 8a | Hệ thống | Trở về giao diện chính nếu người dùng hủy yêu cầu | | 9a | Hệ thống | Chuyển yêu cầu sang tài xế khác nếu tài xế trước đó không chấp nhận hoặc thông báo cho khách hàng nếu không tìm được tài xế | | | |
| **Hậu điều kiện** | Không | | |

### Đặc tả use case UC004 – Nhận yêu cầu gọi xe

**Bảng 5** Đặc tả Usecase Nhận yêu cầu gọi xe

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC004 | **Tên usecase** | Nhận yêu cầu gọi xe |
| **Tác nhân** | Tài xế | | |
| **Mô tả** | Usecase mô tả quá trình tài xế nhận yêu cầu gọi xe | | |
| **Tiền điều kiện** | Tài xế đã đăng nhập vào hệ thống và kích hoạt chế độ nhận yêu cầu gọi xe, có yêu cầu gọi xe từ khách hàng | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Hệ thống | Hiển thị thông tin về yêu cầu gọi xe và thời gian đếm ngược nhận yêu cầu gọi xe | | 2 | Tài xế | Xác nhận đồng ý nhận yêu cầu gọi xe | | 3 | Hệ thống | Hiển thị bản đồ thể hiện vị trí hiện tại của tài xế, khách hàng, tuyến đường từ vị trí cảu tài xế đến vị trí khách hàng | | 4 | Tài xế | Xác nhận đã nhận hàng từ khách hàng | | 5 | Hệ thống | Thêm đơn hàng vừa nhận vào danh sách chuyển hàng của tài xế | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 2a | Tài xế | Không nhận yêu cầu gọi xe hoặc để thời gian nhận trôi qua | | 3a | Hệ thống | Tắt chế độ nhận yêu cầu gọi xe của tài xế trong 5 phút | | | |
| **Hậu điều kiện** | Không | | |

### Đặc tả use case UC005 – Kết thúc đơn hàng

**Bảng 6** Đặc tả Usecase Kết thúc đơn hàng

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC005 | **Tên usecase** | Kết thúc đơn hàng |
| **Tác nhân** | Tài xế | | |
| **Mô tả** | Usecase mô tả quá trình tài xế xác nhận thực hiện thành công yêu cầu chuyển hàng | | |
| **Tiền điều kiện** | Tài xế đã đăng nhập vào hệ thống, nhận yêu cầu chuyển hàng từ khách hàng và đã chuyển hàng đến địa điểm cần giao hoặc hủy yêu cầu vì một lý do. | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Tài xế | Gửi yêu cầu kết thúc đơn hàng cần chuyển | | 2 | Hệ thống | Hiển thị form xác nhận loại kết thúc | | 3 | Tài xế | Xác nhận đã chuyển hàng đến nơi yêu cầu | | 4 | Hệ thống | Hiển thị chuyển thành công đơn hàng và thông báo cho khách hàng | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 4a | Tài xế | Hiển thị hủy đơn hàng thành công và thông báo cho khách hàng | | | |
| **Hậu điều kiện** | Không | | |

### Đặc tả use case UC006 – Đánh giá quá trình sử dụng dịch vụ

**Bảng 7** Đặc tả Usecase Đánh giá quá trình sử dụng

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC006 | **Tên usecase** | Đánh giá quá trình sử dụng dịch vụ |
| **Tác nhân** | Khách hàng | | |
| **Mô tả** | Usecase mô tả quá trình khách hàng đánh giá về chất lượng của dịch vụ vửa sử dụng | | |
| **Tiền điều kiện** | Khách hàng đã đăng nhập vào hệ thống, thực hiện yêu cầu gọi xe chuyển hàng, tài xế đã chấp nhập và chuyển hàng đến địa chỉ hoặc đơn hàng bị hủy. | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Hệ thống | Thông báo tình trạng của đơn hàng (thành công hoặc đã bị hủy) | | 2 | Khách hàng | Xác nhận tình trạng của đơn hàng | | 3 | Hệ thống | Hiển thị form đánh giá về đơn hàng | | 4 | Khách hàng | Đánh giá chất lượng của đơn hàng và gửi xác nhận | | 5 | Hệ thống | Hiển thị thông báo xác nhận thành công | | | |
| **Luồng sự kiện thay thế** | Không | | |
| **Hậu điều kiện** | Không | | |

### Đặc tả use case UC007 – Xem thông tin lịch sử sử dụng

**Bảng 8** Đặc tả Usecase Xem thông tin lịch sử sử dụng

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã usecase** | UC007 | **Tên usecase** | Xem thông tin lịch sử sử dụng |
| **Tác nhân** | Khách hàng/tài xế | | |
| **Mô tả** | Usecase mô tả quá trình khách hàng / tài xế xem lại lịch sử sử dụng dịch vụ của mình | | |
| **Tiền điều kiện** | Khách hàng / tài xế đã đăng nhập vào hệ thống | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Khách hàng / tài xế | Chọn chức năng xem lại lịch sử | | 2 | Hệ thống | Hiển thị giao diện lịch sử các dịch vụ đã sử dụng | | 3 | Khách hàng / tài xế | Tìm kiếm lịch sử sử dụng | | 4 | Hệ thống | Hiển thị danh sách lịch sử sử dụng | | 5 | Khách hàng / tài xế | Chọn 1 lịch sử | | 6 | Hệ thống | Hiển thị thông tin của lịch sử của dịch vụ đó. | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 4a | Hệ thống | Thông báo nếu lịch sử sử dụng trống | | | |
| **Hậu điều kiện** | Không | | |

### Đặc tả use case UC008 – Khóa người dùng

**Bảng 9** Đặc tả Usecase Khóa người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã use case** | UC008 | **Tên use case** | Khóa người dùng |
| **Tác nhân** | Quản trị viên | | |
| **Mô tả** | Use case mô tả quá trình quản trị viên khóa một người dùng. | | |
| **Tiền điều kiện** | Quản trị viên đã đăng nhập vào hệ thống. | | |
| **Luồng sự kiện chính** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 1 | Quản trị viên | Chọn chức năng quản trị người dùng. | | 2 | Hệ thống | Hiển thị giao diện quản trị người dùng. | | 3 | Quản trị viên | Nhập từ khóa tìm kiếm một người dùng. | | 4 | Quản trị viên | Gửi yêu cầu tìm kiếm | | 5 | Hệ thống | Hiển thị danh sách người dùng. | | 6 | Quản trị viên | Chọn một người dùng, yêu cầu xem thông tin | | 7 | Hệ thống | Hiển thị thông tin người dùng được chọn. | | 8 | Quản trị viên | Gửi yêu cầu khóa người dùng. | | 9 | Hệ thống | Cập nhật trạng thái của người dùng. | | | |
| **Luồng sự kiện thay thế** | |  |  |  | | --- | --- | --- | | STT | Thực hiện bởi | Hành động | | 5a | Hệ thống | Thông báo nếu danh sách người dùng trống. | | | |
| **Hậu điều kiện** | Không | | |

## Các yêu cầu phi chức năng

### Thân thiện, dễ dùng

* Các chức năng, giao diện dễ sử dụng và thân thiện với người dùng.
* Có phản hồi với người dùng về các vấn đề lỗi đường truyền mạng.

### Hiệu suất

* Hệ thống sẽ hoạt động 24 giờ một ngày, 7 ngày một tuần.
* Thời gian ngưng hoạt động để khắc phục lỗi của hệ thống không lớn hơn 1 giờ.
* Thời gian đáp ứng yêu cầu của người dùng không quá 2s.
* Thông tin hiển thị lên giao diện cho người dùng không quá 3s

### Độ tin cậy

* Hệ thống có tính an toàn, bảo mật thông tin cá nhân của người dùng.
* Dữ liệu chính xác, được cập nhật liên tục theo thời gian thực

Như vậy ­­Khảo sát và phân tích yêu cầuChương 2 đã đưa ra được kết quả khảo sát và phân tích yêu cầu. Xây dựng được hệ thống các chức năng cần có và phân tích các chức năng này. Từ đó tạo nền tảng để xây phân tích dựng ứng dụng ở các chương tiếp theo. Cụ thể là lựa chọn các công nghệ sử dụng ở Chương 3.

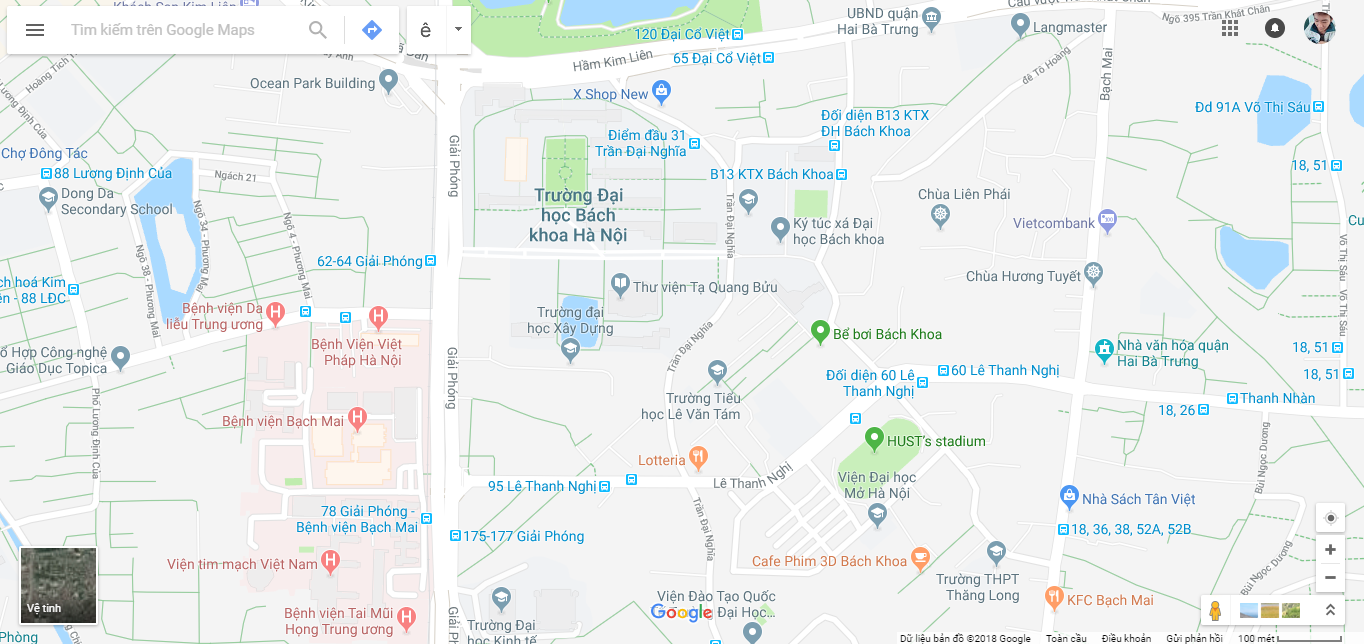
# Công nghệ sử dụng

Như đã trình bày tóm tắt trong phần phân tích một số định hướng giải pháp ở chương 1, và dựa trên việc phân tích yêu cầu ở Chương 2, chương này sẽ tập trung vào phân tích và giải thích rõ các giải pháp công nghệ được sử dụng để xây dựng ứng dụng. Bao gồm:

1. Nhóm các giải pháp liên quan đến dịch vụ bản đồ, sử dụng các dịch vụ của Google Maps API.
2. Giải pháp xử lý dữ liệu theo thời gian thực với Google Firebase Realtime Database
3. Giải pháp xác thực danh tính với Google Firebase Authencation
4. Giải pháp lưu trữ và quản lý tập tin với Google Firebase Cloud Storage).
5. Giải pháp thiết kế giao diện

## Giải pháp bản đồ với Google Maps API

Google Map (Hình 13) là một dịch vụ ứng dụng và công nghệ bản đồ trực tuyến miễn phí được cung cấp bởi Google. Nó cho phép thấy bản đồ đường sá, đường đi cho xe đạp, cho người đi bộ và xe hơi, và những địa điểm kinh doanh trong khu vực cũng như khắp nơi trên thế giới. Với nhiều tính năng vượt trội, Google Map là dịch vụ bản đồ số được ưa chuộng và sử dụng nhiều nhất trên thế giới.



**Hình 13** Giao diện Google Map trên nền Web

Tháng 6/2005, Google đã cho ra mắt Google Maps API, một dịch vụ miễn phí cho phép các nhà phát triển tích hợp Google Maps vào các trang web của họ.

Google Maps API [1] cung cấp rất nhiều các thư viện hỗ trợ mạnh mẽ cho nhà phát triển bao gồm các bộ SDK trên ứng dụng (như Place API, Android API, iOS SDK, Javascript API...) cùng với các Web Service API (như Directions API, Roads API...).



**Hình 14** Google Map API với nhiều dịch vụ hỗ trợ ( nguồn MapLink)

Đến thời điểm hiện tại, Google Map API đã được nâng cấp lên phiên bản v3, không chỉ hỗ trợ cho các máy để bàn truyền thống mà cho cả các thiết bị di động; các ứng dụng nhanh hơn và nhiều hơn .

### Tích hợp Google Map bằng Android API

Với mục tiêu ban đầu là xây dựng hệ thống phía client trên nền tảng Android nên ứng dụng được tích hợp Android API để xử lý hiển thị bản đồ. Việc tích hợp và sử dụng Android API khá dễ dàng. Google đã cung cấp các tài liệu hướng dẫn chi tiết để lập trình viên có thể tiếp cận và phát triển ứng dụng bản đồ của mình.



Hình 15 Tích hợp Google Map vào thiết bị di động bằng API Google Maps

*Với Google Maps Android API, bạn có thể thêm bản đồ dựa trên dữ liệu Google Maps vào ứng dụng của bạn. API tự động xử lý quyền truy cập vào các máy chủ của Google Maps, tải dữ liệu, hiển thị bản đồ và phản ứng lại các thao tác trên bản đồ. Bạn cũng có thể sử dụng các hàm gọi API để thêm các điểm đánh dấu, đa giác và lớp phủ vào một bản đồ cơ bản và để thay đổi chế độ xem của người dùng về một khu vực bản đồ cụ thể. [1]* (Hình 15)

### Xác định vị trí với Google Play services location API

Google Play services location API [1] là dịch vụ cho phép chúng ta lấy được vị trí hiện tại của thiết bị. Hiện nay có 3 nguồn cung cấp để lấy vị trí bao gồm:

1. GPS: nguồn này xác định vị trí dựa trên vệ tinh. Tùy thuộc vào điều kiện trong nhà hay ngoài trời mà nguồn này cần một khoảng thời gian để có thể trả về vị trí hiện tại với độ chính xác phụ thuộc.
2. NETWORK PROVIDER : nguồn này xác định vị trí dựa trên cột thu phát sóng của mạng di động và các điểm truy cập WIFI. Kết quả vị trí thu được nhờ tìm kiếm trong mạng.
3. PASSIVE PROVIDER: nguồn này trả về vị trí sinh ra bởi các nguồn khác. Ví dụ bạn nhận được vị trí cập nhật một cách thụ động khi mà các ứng dụng khác hoặc dịch vụ khác yêu cầu mà ko phải bạn thực sự yêu cầu điều này.

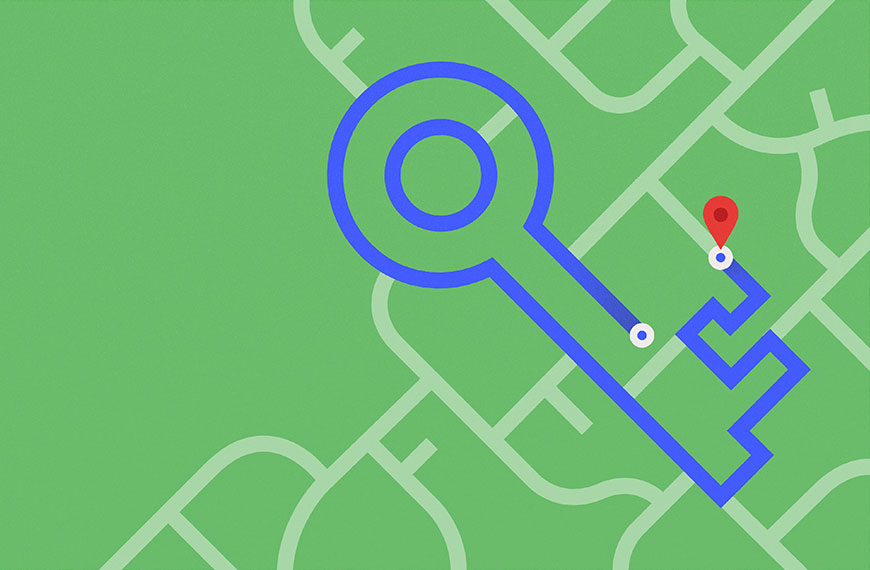
Google Play services location API là cách mới mà Google giới thiệu để có thể lấy được vị trí hiện tại của thiết bị. API này dùng "Fused Location Provider" - tự động chọn nguồn cung cấp vị trí để sử dụng dựa trên độ chính xác và mức tiêu thụ pin. API này cũng được Google khuyến khích sử dụng thay thế cho Android Location API cũ.

Việc tích hợp Google Play services location API vào ứng dụng khá đơn giản. Lập trình viên chỉ cần tích hợp gói Google Play services vào ứng dụng và thực hiện gọi các hàm API để lấy dữ liệu về vị trí.

### Xác định đường đi với Google Directions API

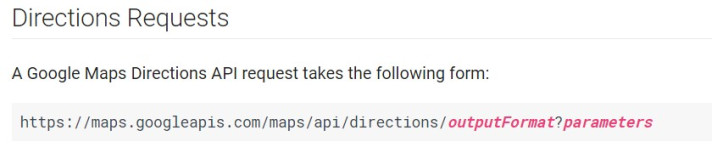
Việc xác định đường đi từ đó xác định được khoảng cách, thời gian di chuyển trung bình là một trong những tính năng quan trọng của ứng dụng. Chi phí phát sinh cơ bản của một hợp đồng hàng vận chuyển trên ứng dụng là kết quả tính toán từ chức năng này.

Để hiển thị đường đi từ một điểm đến một điểm khác, hệ thống lựa chọn sử dụng dịch vụ Directions API. Dịch vụ này giúp tính toán đường đi giữa các vị trí và trả về kết quả một tập các địa điểm GPS có thứ tự mà con đường đi qua khi nhận một yêu cầu từ người dùng, ví dụ như (**Hình 16** Thiết lập đường đi bằng Direction API)



**Hình 16** Thiết lập đường đi bằng Direction API (nguồn Google MAP API)

Việc tích hợp vụ **Directions API** được thông qua bằng giao diện HTTP với cấu trúc **Request – Response**. Trong đó **Request** với các yêu cầu được xây dựng dưới dạng chuỗi URL, sử dụng chuỗi văn bản hoặc vĩ độ / kinh độ để xác định vị trí, cùng với khóa API của bạn.[1] (**Hình 17)**



**Hình 17** Cấu trúc 1 request (nguồn Google Map API)

Ở đây chúng ta chú ý đến outputFormat sẽ là kiểu dữ liệu mà ta muốn nhận về từ Response. Hiện tại Google hỗ trợ 2 định dạng là JSON và XML.

Ở phần parameters ta cần phải xác định 3 yếu tố bắt buộc sau đây:

1. Origin: Là địa điểm mà ta chọn là địa điểm xuất phát, bao gồm các định dạng được chấp nhận như sau: Địa chỉ, giá trị vĩ độ / kinh độ văn bản, hoặc ID nơi bạn muốn tính chỉ đường.
2. Destination: mục này là địa điểm mà ta chọn làm địa điểm đích đến, và cấu trúc cũng tương tự như Origin.
3. Key: Khóa API cho ứng dụng . Việc sử dụng khóa này giúp Google xác định được mức hạn ngạch Request mỗi ngày của ứng dụng. Chúng ta có thể tăng mức hạn ngạch này bằng cách sử dụng các gói cước của Google.

Bên cạnh đó, ta có thể bổ sung thêm 1 số parameters tùy chọn khác như sau:

1. Mode: chọn loại phương tiện mà ta muốn di chuyểnAlternatives: có thể trả về nhiều hơn 1 tuyến đường khả dụng.
2. Avoid: tuyến đường trả về cần phải tránh 1 số vấn đề như tránh đường cao tốc, tránh đi qua các trạm thu phí cầu đường.
3. … (tham khảo thêm trên trang document của Directions API).

Sau khi gửi 1 Request lên, một chuỗi Response trả về chỉ đường được trả lại dưới định dạng được yêu cầu trong đường dẫn của URL Request (dạng JSON hoặc XML). Cụ thể chuỗi Response này được chia thành các step nhỏ, mỗi step là 1 đoạn trong tuyến đường di chuyển giữa 2 điểm. (**Bảng 10**)

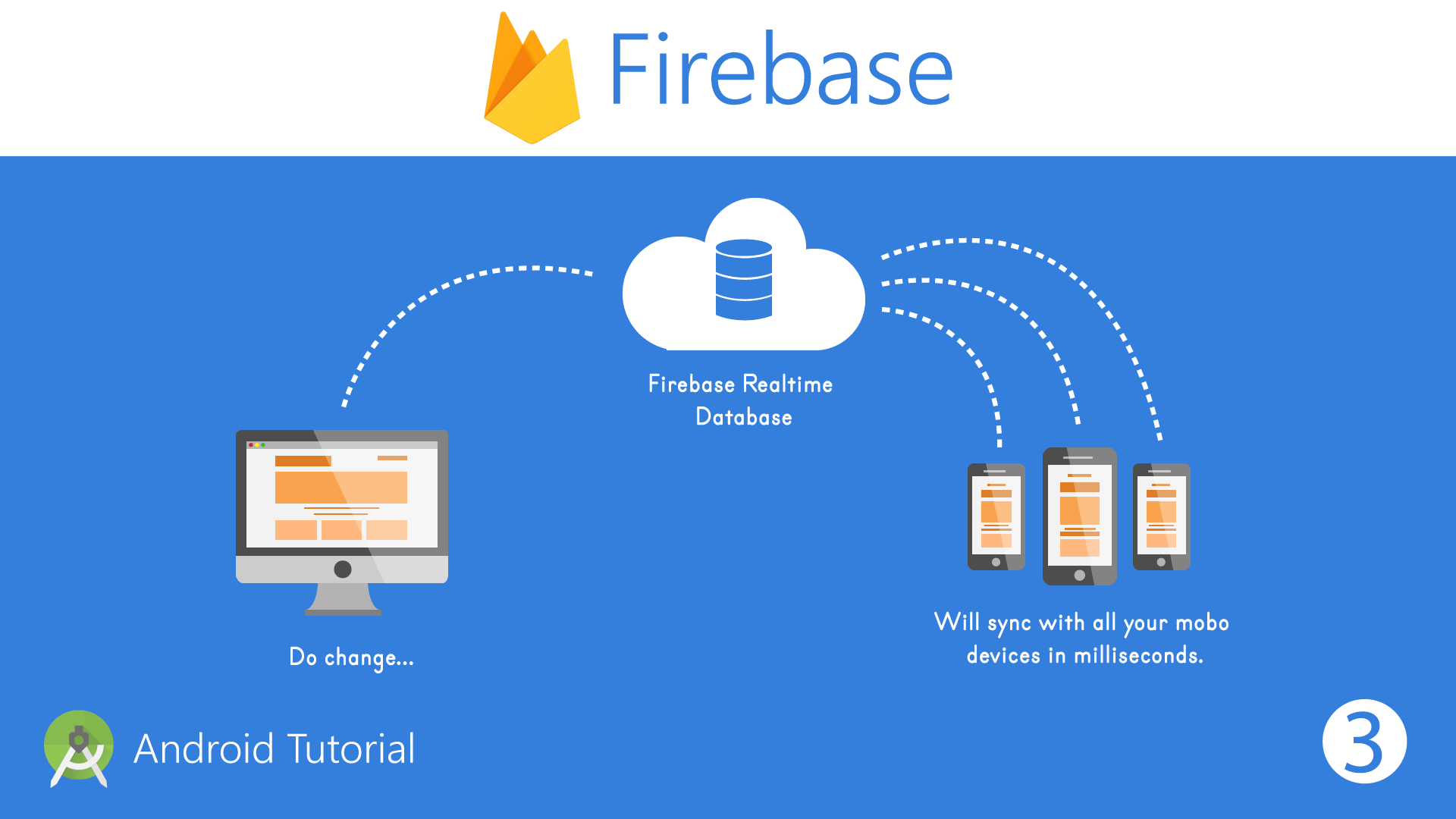
**Bảng 10** Ví dụ cấu trúc dữ liệu của chuỗi Response

|  |
| --- |
| "status": "OK",  "geocoded\_waypoints" : [  { "geocoder\_status" : "OK",  "place\_id" : "ChIJ7cv00DwsDogRAMDACa2m4K8",  "types" : [ "locality", "political" ]},  { "geocoder\_status" : "OK",  "place\_id" : "ChIJ69Pk6jdlyIcRDqM1KDY3Fpg",  "types" : [ "locality", "political" ]},  { "geocoder\_status" : "OK",  "place\_id" : "ChIJgdL4flSKrYcRnTpP0XQSojM",  "types" : [ "locality", "political" ]},  { "geocoder\_status" : "OK",  "place\_id" : "ChIJE9on3F3HwoAR9AhGJW\_fL-I",  "types" : [ "locality", "political" ]}],  ... |

Sau khi nhận được chuỗi Response, lập trình viên chỉ cần xử lý chuỗi này dựa trên yêu cầu của bài toán.

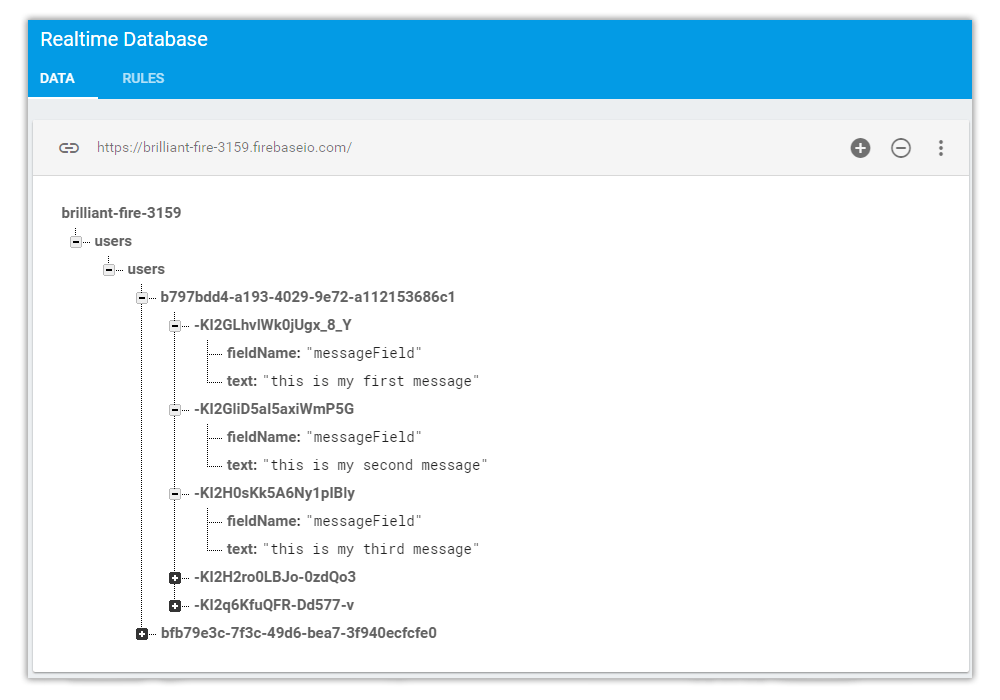
## Xử lý dữ liệu theo thời gian thực với Firebase Realtime Database

Firebase Realtime Database [2] là một Cloud hosted database hỗ trợ đa nền tảng: Android, IOS, Web, C++, Unity. Tất cả dữ liệu được lưu trữ ở định dạng JSON. Khi có một sự thay đổi dữ liệu nào thì có sự phản hồi ngay lập tức, hiển thị đồng bồ trên các tất cả các nền tảng và các thiết bị liên kết với nó. (**Hình 18**)



**Hình 18** Đồng bộ hóa trực tiếp trên nhiều thiết bị với Realtime Database ( nguồn Google Firebase)

Khác với các kiểu dữ liệu thường thấy như SQL database được tổ chức theo dạng bảng, Firebase Realtime được tổ chức theo dạng cây (**Hình 19** Tổ chức dữ liệu dạng cây của Realtime Database), với mỗi nhánh giống như một container, chỉ chứa hoặc là dữ liệu ứng với nhánh đó (tức là giá trị tương ứng với key); hoặc một tập hợp các nhánh con cũng được tổ chức theo một cách tương tự.



**Hình 19** Tổ chức dữ liệu dạng cây của Realtime Database

Mỗi nhánh được đại diện bởi một key và value tương ứng. Các key luôn có dạng là String. Còn các giá trị thì có thể là một đối tượng kiểu String, một đối tượng kiểu Int, hay là một Object, được thể hiện dưới dạng một nhánh con. Và nhánh con đó được coi như là một value ứng với key đó. Do được tổ chức theo hình thức này, dữ liệu được gửi về các thiết bị có định dạng là JSON.

Các key cùng tầng phải đảm bảo được tính duy nhất. Các key khác tầng có thể được phép giống nhau. Để đảm bảo các key cùng tầng là duy nhất, Firebase cung cấp cho chúng ta các hàm để yêu cầu Firebase server tạo ngẫu nhiên các giá trị key duy nhất, ví dụ như: push() trong Android và Web, hay childByAutoId() trên iOS.

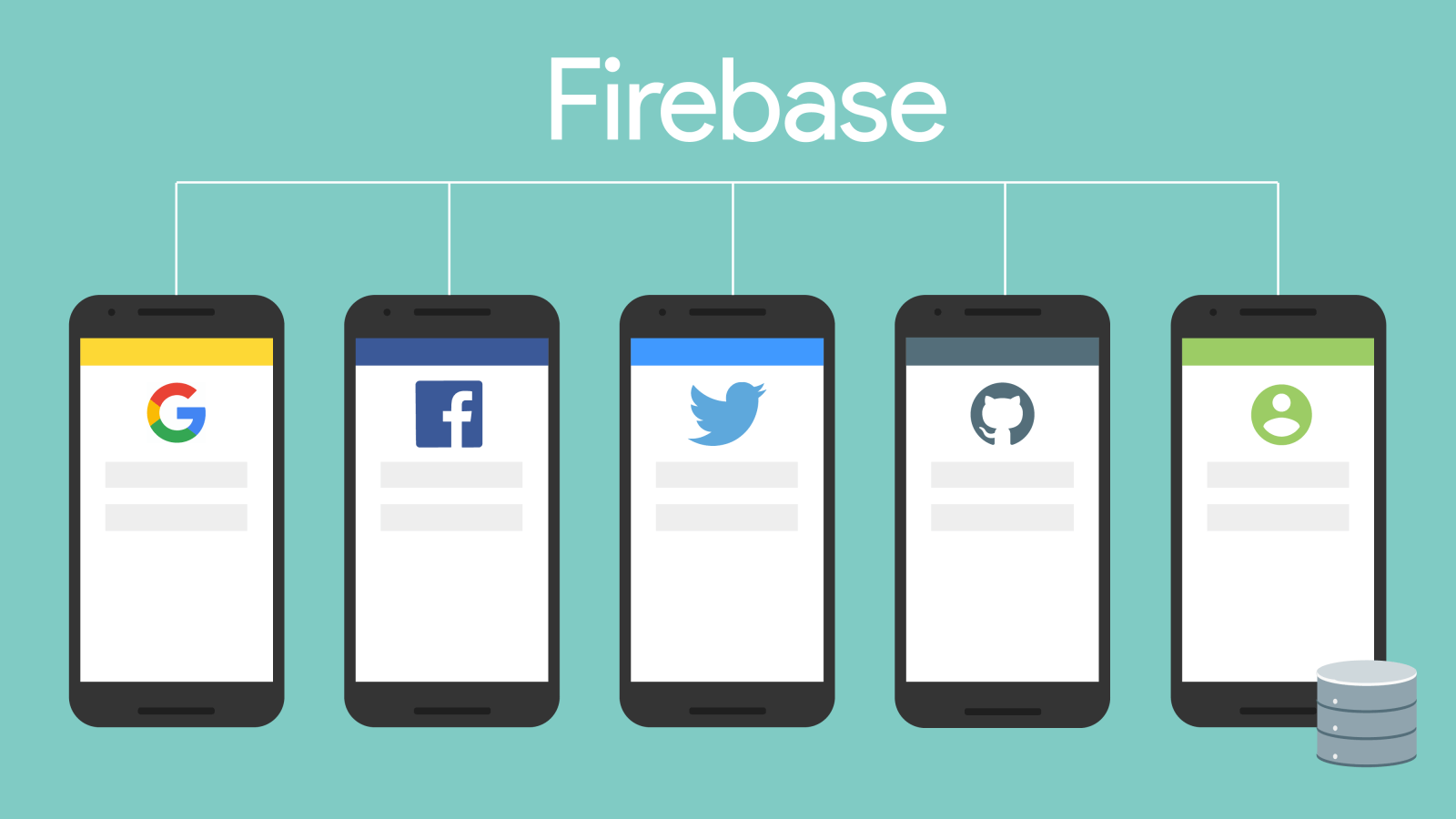
Khi bạn thêm một bộ dữ liệu mới, tức bao gồm 1 bộ key – value mới, thì sẽ xảy ra hai trường hợp sau:

1. Key chưa tồn tại: Firebase Database sẽ tạo ra một Object mới tại vị trí mà bạn muốn lưu, với cặp key – value là các giá trị bạn đã định nghĩa. Event này được gọi là “add”, tức là thêm.
2. Key đã tồn tại: Firebase Database sẽ sửa value tại vị trí của key đó theo value mới. Event này được gọi là “change”.

Việc tích hợp Firebase Realtime Database vào ứng dụng khá đơn giản. Lập trình viên chỉ cần tạo Project trên trang chủ Firebase và lựa chọn liên kết với Firebase Realtime Database trên IDE Android studio bằng tài khoản và Project đã tạo.

## Xác thực người dùng với Firebase Authencations

Firebase Authencations [2] là giải pháp giúp xây dựng các hệ thống xác thực an toàn một cách dễ dàng, đồng thời nâng cao trải nghiệm đăng nhập và cập nhật cho người dùng. Nó cung cấp khá nhiều giải pháp đầu cuối như: hỗ trợ tài khoản email, mật khẩu, xác thực số điện thoại, đăng nhập qua tài khoản Google, Twitter, Facebook và GitHub.(**Hình 20**)



**Hình 20** Firebase Aunthencation hỗ trợ nhiều loại xác mình đăng nhập khách nhau (nguồn Firebase Aunthencation)

Firebase Authentication hoạt động như thế nào?

1. Để xác thực người dùng đăng nhập vào ứng dụng của bạn, trước tiên bạn nhận được thông tin xác thực từ người dùng. Những thông tin đăng nhập này có thể là địa chỉ email và mật khẩu của người dùng, hoặc mã OAuth từ một nhà cung cấp nhận dạng của bên thứ 3 (Facebook, Twitter,…)
2. Sau đó, bạn chuyển những chứng chỉ này tới SDK Authentication Firebase. Các dịch vụ phụ trợ của Firebase Authentication sẽ xác mình các thông tin và trả lời cho khách hàng.
3. Sau khi đăng nhập thành công, ứng dụng của bạn có thể truy cập vào thông tin tiểu sử cơ bản của người dùng và bạn có thể kiểm soát quyền truy cập của người dùng vào dữ liệu được lưu trữ trong ứng dụng của mình.

Lập trình viên có thể tích hợp Firebase Aunthencation vào ứng dụng bằng cách sử dụng FirebaseUI – phương thức xác thực qua bộ thư viện kéo thả bằng trợ lý Assistant trên Android Studio hoặc sử dụng Firebase Authentication SDK theo cách thủ công.

## Lưu trữ dữ liệu với Firebase Storage

Firebase Storage [2] là dịch vụ được xây dựng cho mục đích lưu trữ và quản lý các nội dung mà người dùng ứng dụng tạo ra như ảnh, videos hay dữ liệu dạng file.

Firebase Storage cung cấp các API cho việc uploads và download các file từ ứng dụng của bạn một cách bảo mật và bạn không cần quan tâm đến chất lượng đường truyền mạng.

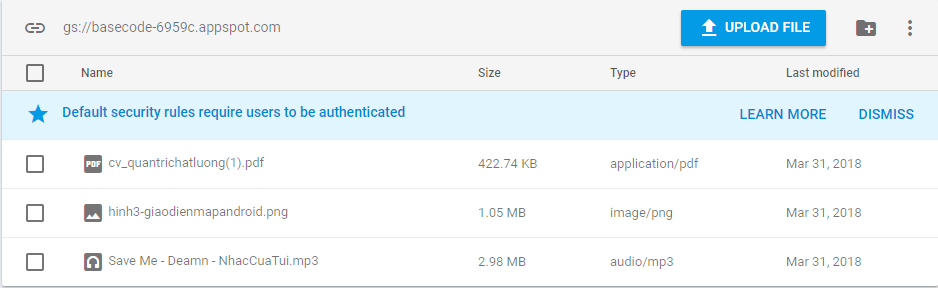
Với việc được xây dựng trên nền tảng Google Cloud Platform nên Firebase Storage có nhiều lợi thế, cụ thể:

1. Robust operations (Hoạt động mạnh mẽ): Với Firebase Storage bạn có thể thực hiện tải dữ liệu lên và xuống bất kể chất lượng của đường truyền mạng. Hỗ trợ tiếp tục tải lên / tải xuống tại nơi người dùng dừng lại, giúp tiết kiệm thời gian và băng thông.
2. Strong security (Tính bảo mật mạnh): Được tích hợp Firebase Authentication cho việc bảo mật nên bạn có thể dễ dàng quản lý quyền truy cập vào các files cho từng đối tượng người dùng.
3. High scalability (Khả năng mở rộng): Firebase Storage được xây dựng trên nền tảng Google Cloud Platform nên khả năng mở rộng có thể lên đến hàng Petabyte dữ liệu.

*Firebase Storage hoạt động như thế nào?*

1. Lập trình viên sử dụng Firebase SDK cho Cloud để tải lên và tải các tệp trực tiếp từ thiết bị của người dùng. Nếu chất lượng đường truyền không ổn định, người dùng có thể thử lại tại nơi có đường truyền ổn định hơn.
2. Cloud Storage lưu trữ tệp mà người dùng tải lên vào Google Cloud Storage, giúp các tệp này có thể truy cập qua cả Firebase và Google Cloud. Điều này cho phép bạn linh hoạt trong việc tải lên hoặc xuống tệp tin từ người dùng đồng thời thực hiện xử lý các công việc phía server như lọc hình ảnh hoặc chuyển mã video thông qua Google Cloud Platform.
3. Firebase Storage sử dụng các Reference để chỉ đến một bucket hoặc một file, bạn có thể làm bất cứ thứ gì với bucket, files thông qua Reference đó như upload, download hay thay đổi metadata hoặc xóa file. Reference mang tính chất lightweight vì vậy bạn có thể sử dụng nó bao nhiêu tùy thích hoặc có thể sử dụng lại một Reference cho nhiều hành động khác nhau.

Việc tích hợp Firebase Storage vào ứng dụng khá đơn giản thông qua Firebase Assistant trên Android Studio IDE.



**Hình 21** Giao diện quản lý Storage trên Firebase Console

## Giải pháp thiết kế giao diện - Material Design

Material Design là một ngôn ngữ thiết kế được phát triển vào năm 2014 bởi Google. Mở rộng dựa trên mô típ "thẻ" (**Hình 22**) có mặt trên Google Now, Material Design đem đến phong cách tự do hơn với các cách bố trí dạng lưới, các phản hồi hoạt họa chuyển động, kéo giãn, và các hiệu ứng chiều sâu như ánh sáng và đổ bóng.[3]

Google giới thiệu Material Design vào ngày 25 tháng 6 năm 2014, tại hội nghị Google I/O 2014.



**Hình 22** Thiết kế Material Design nổi bật với các thẻ (nguồn Material Design)

Đặc điểm của Material Design:

1. Sử dụng các màu nổi bật, thường có một mảng màu chủ đạo nằm ở cạnh trên ứng dụng.
2. Các biểu tượng phẳng, đơn giản nhưng dễ hiểu.
3. Một số ứng dụng sẽ có một nút tròn to nằm ở góc dưới bên phải, thường có chức năng tạo mới.
4. Giao diện phẳng, ít hoặc không có hiệu ứng chuyển màu, có hoặc không có hiệu ứng đổ bóng đen.
5. Menu, nút nhấn, chữ viết… có nhiều khoảng cách trắng nên trông thoáng đãng.
6. Có các hiệu ứng chuyển động tự nhiên, dễ hiểu, có thể gợi ý cho một tính năng nào đó.

Tóm lại, chương 3 đã phân tích làm rõ ưu thế các công nghệ được lựa chọn để xây dựng ứng dụng. Nắm rõ được các ưu thế này giúp cho quá trình thiết kế hệ thống trở nên đơn giản và nhanh chóng hơn.

# Phát triển và triển khai ứng dụng

Chương 4 sẽ tập trung xây dựng kiến trúc của ứng dụng đồng thời đưa ra các quy chuẩn về thiết kế kiến trúc, thiết kế giao diện.

## Thiết kế kiến trúc

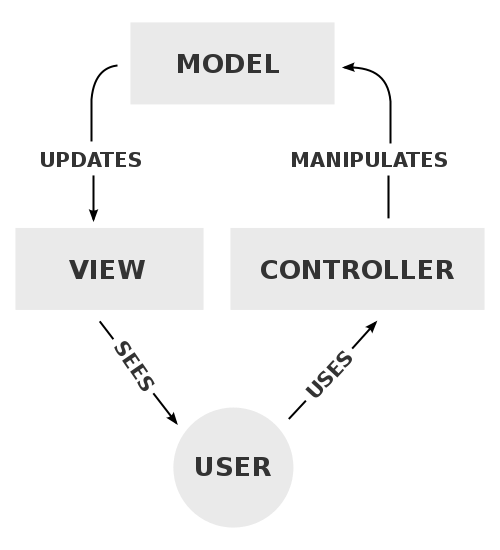
### Lựa chọn kiến trúc phần mềm

Ứng dụng được thiết kế dựa trên mô hình MVP với đặc tinh kế thừa những ưu điểm từ mô hình MVC (**Hình 23**).

*“MVC (Model-View-Controller) là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tính. MVC chia một ứng dụng thành ba phần tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dùng.”* [5]

MVC bao gồm 3 thành phần: Model, View, Controller.

1. Model: Nơi lưu trữ giữ liệu và được lấy ra bởi câu lệnh từ bên lớp điều khiển (Controller) để hiển thị lên giao diện (View)
2. View: Nơi xuất đầu ra mới cho người dùng dựa trên những thay đổi của dữ liệu (model).
3. Controller: Nơi gửi những câu lệnh sang Model để cập nhật trạng thái dữ liệu. Sau đấy, nó có thể gửi trả dữ liệu đấy về View được liên kết để thay đổi giao diện tới người dùng.

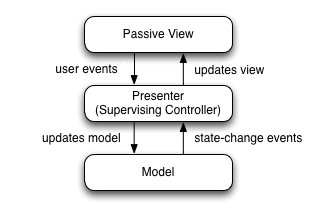


**Hình 23** Mô hình MVC [Nguồn: Wikipedia]

Với MVC, Model và View đã được tách ra và như vậy các Model có thể dễ dàng test bởi vì nó không gắn với bất kỳ thành phần nào. Tuy nhiên Controller có một vài vấn đề.

1. Testability: Controller được gắn rất chặt chẽ đến các API của Andoird mà rất khó để unit test.
2. Modularity & Flexibility: Các Controller được kết chặt chẽ với các View. Nó cũng có thể là một phần mở rộng của View. Nếu chúng ta thay đổi View, chúng ta phải quay trở lại và thay đổi Controller.
3. Maintenance: Theo thời gian, càng ngày code của Controller càng nhiều, làm cho chúng cồng kềnh và dễ gãy.

MVP (Model-View-Presenter) là một cải tiến của mô hình MVC. Trong MVP, View và Model có sự tách biệt rõ ràng hơn trong khi Controller được thay thế bởi Presenter. Đấy cũng là điểm khác biệt nhất giữa hai mô hình này bởi Presenter mang gánh vác nhiện vụ “người đứng giữa”. Tất cả xử lý logic đều được đẩy hết cho Presenter. (**Hình 24**)

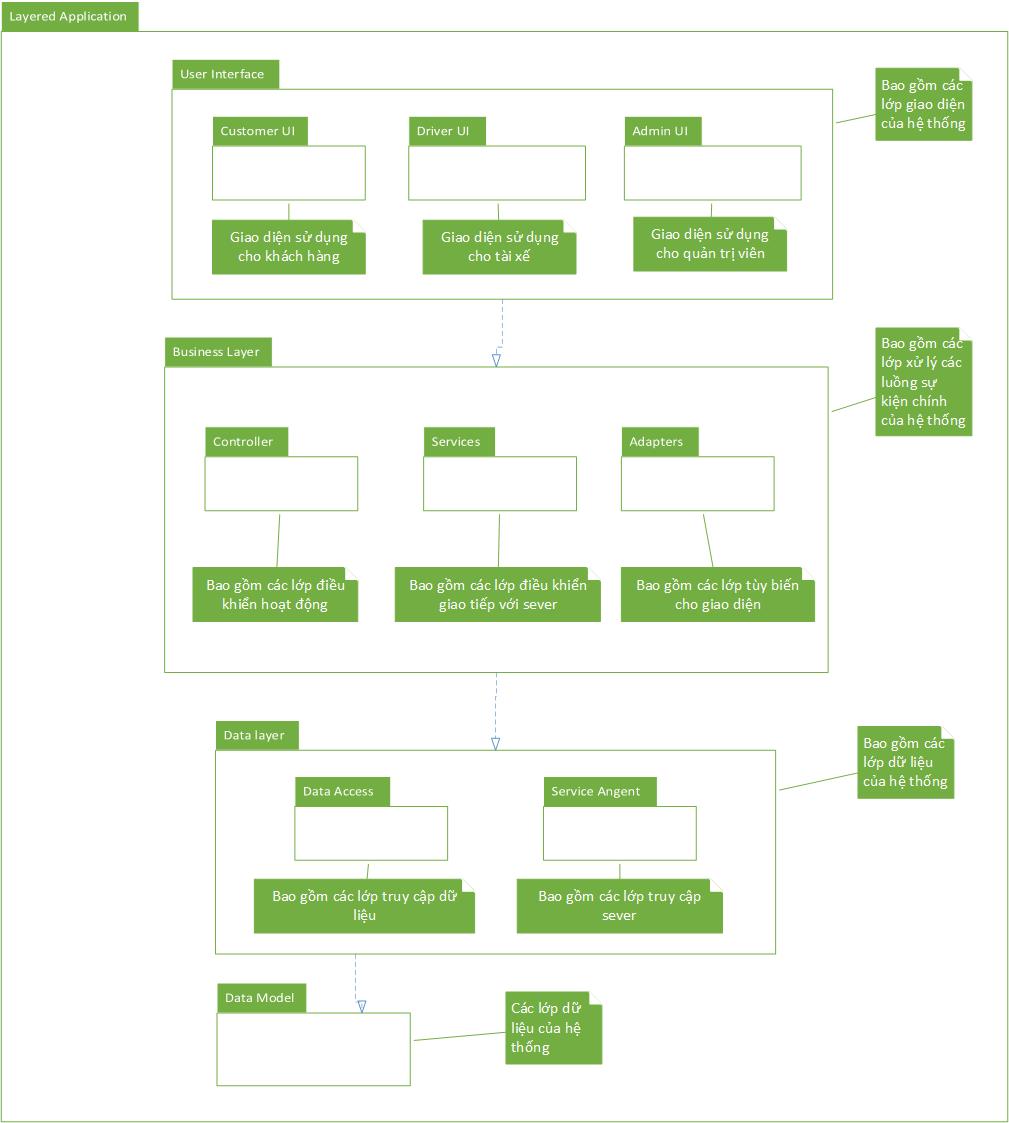


**Hình 24** Mô hình MVP (nguồn Wikipedia)

Với thiết kế như vậy, so với MVC thì MVP đã rõ ràng hơn rất nhiều. Chúng ta có thể dễ dàng thực hiện unit test hay logic test cho Presenter bởi vì nó không gắn với bất kỳ View cụ thể nào cả.

### Thiết kế tổng quan

Dựa trên kiến trúc phần mềm, hệ thống được xây dựng thành 3 tầng xử lý chính, mỗi tâng có chức năng cụ thể như sau (**Hình 25**):

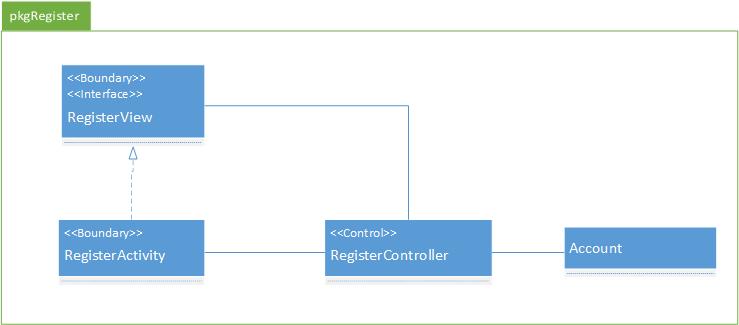


**Hình 25** Thiết kế tổng quan hệ thống

### Thiết kế chi tiết gói

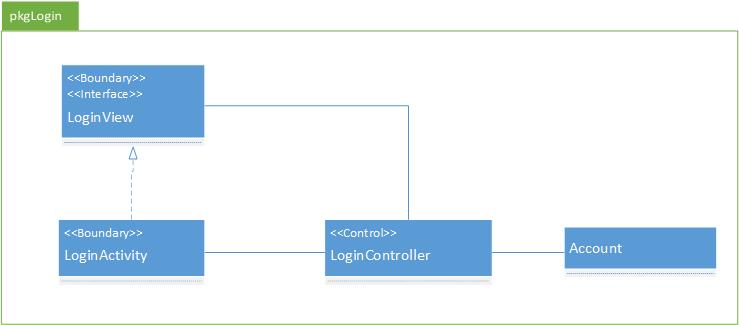
Sau đây em xin trình bày biểu đồ lớp phân tích gộp 1 số use case liên quan tới nhau của một vài chức năng của ứng dụng:

* Thiết kế chi tiết gói chức năng Đăng ký (**Hình 26**)



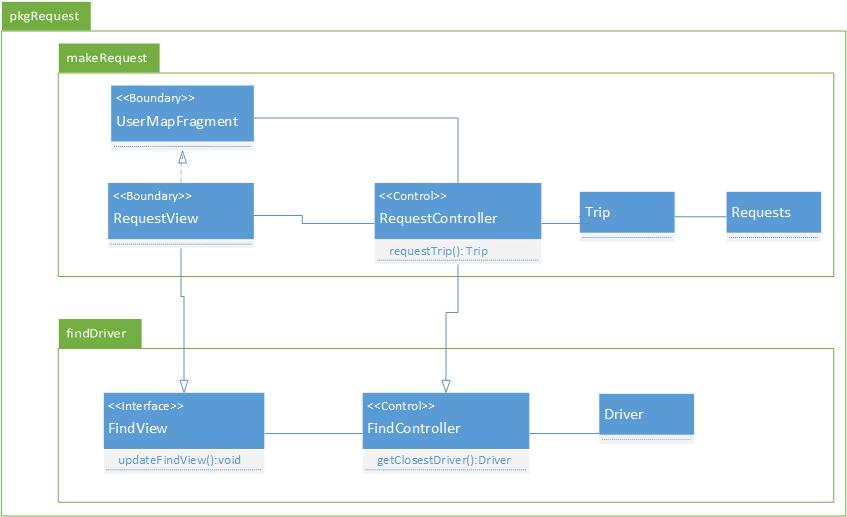
**Hình 26** Thiết kế chi tiết gói chức năng Đăng ký

* Thiết kế chi tiết gói chức năng Đăng nhập (**Hình 27**)



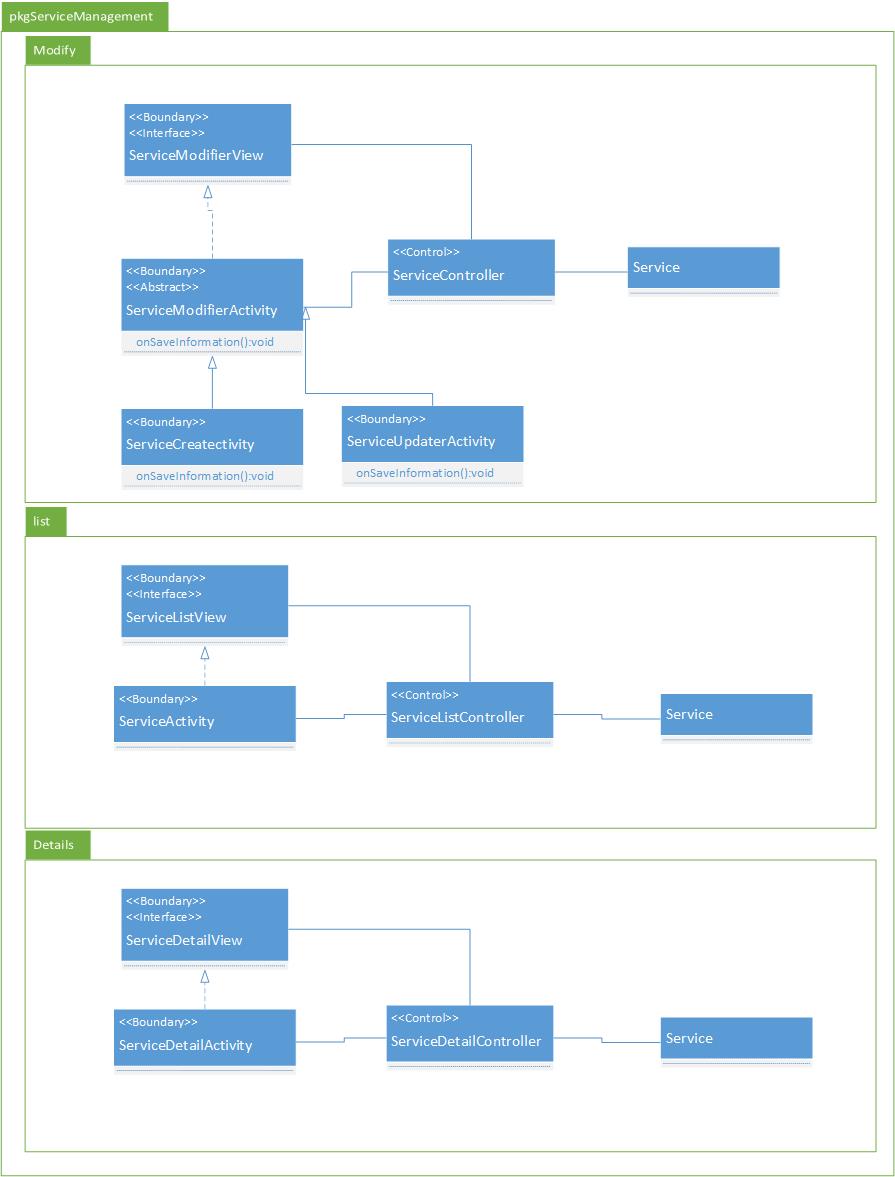
**Hình 27** Thiết kế chi tiết gói chức năng Đăng Nhập

* Thiết kế chi tiết gói chức năng Gọi xe (**Hình 28**)

****

**Hình 28** Thiết kế chi tiết gói chức năng Gọi xe

* Thiết kế chi tiết gói chức năng Quản lý dịch vụ (**Hình 29**)

****

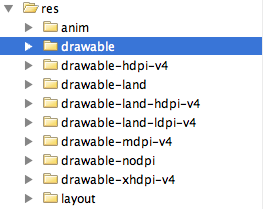
**Hình 29** Thiết kế chi tiết gói chức năng Quản lý dịch vụ

## Thiết kế chi tiết

### Thiết kế giao diện

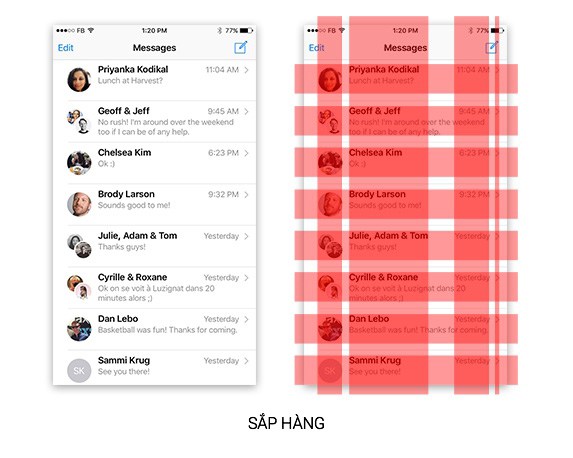
Việc thiết kế giao diện cho ứng dụng cần đảm bảo tính đơn giản, dễ sử dụng cho người dùng. Do đó cần thống nhất các quy chuẩn thiết kế cho ứng dụng. Để đảm bảo yêu cầu này SV đưa ra các quy tắc thiết kế giao diện sau:

1. Quy định về màn hình: Hệ thống xây dựng trên hệ điều hành Android. Hiện nay trên thị trường các hàng sản xuất điện thoại Android rất nhiều, với nhiều loại điện thoại khác nhau, kích cỡ màn hình khác nhau. Do đó ứng dụng cần thiết kế phù hợp với các loại thiết bị. Không để bị mất thông tin, gây khó khăn cho người dùng. Các file hình ảnh, thiết kế cần được tùy chỉnh theo từng loại màn hình theo quy chuẩn thiết kế của Android (**Hình 30**)



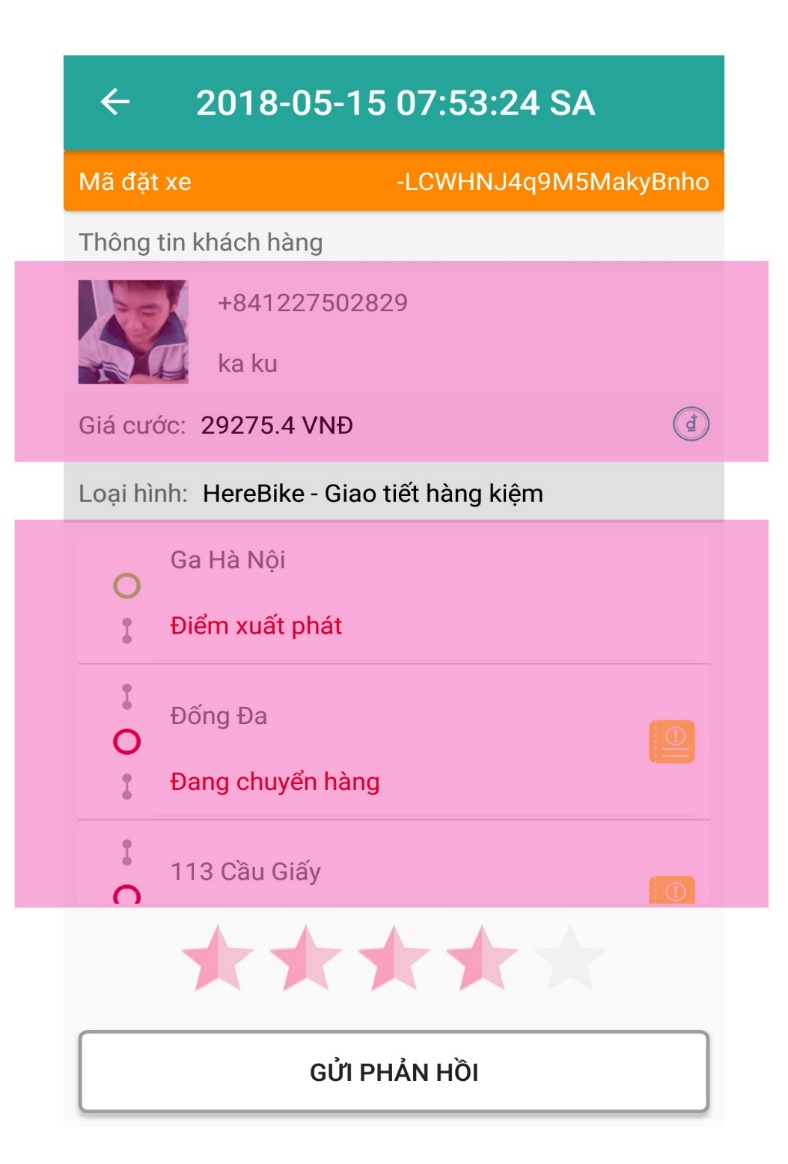
**Hình 30** Các gói hình ảnh cho các loại màn hình của Android

1. Quy định về sắp xếp thông tin: Bất kỳ thông tin được sắp xếp phải thẳng hàng với đối tượng đầu xuyên suốt trong các nguyên tắc thiết kế.Dựa trên nguyên tắc trái sang phải dưới giống như thói quen đọc sách. (**Hình 31**)

****

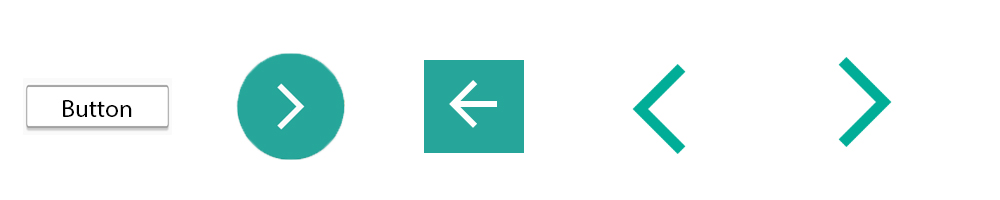
**Hình 31** Sắp xếp các đối tượng thẳng hàng xuyên suốt

1. Tổ chức thông tin theo nguyên tắc quan hệ: Tổ chức thông tin theo các nguyên tắc quan hệ, gần gũi với các thông tin khác có liên quan, nhóm chúng lại với nhau. Nếu nhiều mục tồn tại gần nhau có quan hệ chặt chẽ nó sẽ trở thành một nhóm trực quan, chứ không phải là các yếu tố bị cô lập, đơn lẻ. Điều này sẽ giúp làm giảm sự nhầm lẫn và cung cấp cho người đọc một cấu trúc rõ ràng, dễ tiếp cận. (**Hình 32**)



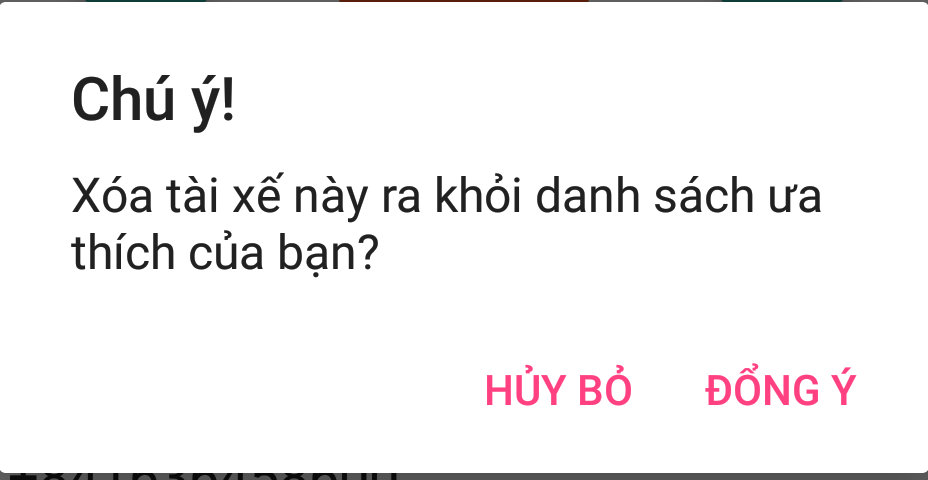
**Hình 32** Nhóm các thông tin có đặc điểm chung lại với nhau

1. Thiết kế nút, điều khiển: Các nút xác nhận thiết kế đơn giản hình chữ nhật, viền xám, nền trắng, các phím điều khiển chức năng hình tròn, nền xanh với icon trắng, các phim điều hướng tiếp tục, quay lại được thiết kế như **Hình 33**:



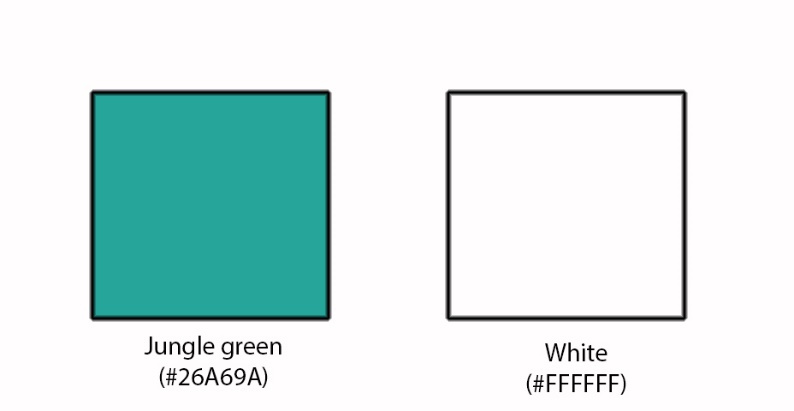
**Hình 33** Mẫu thiết kế các nút, điều khiển

1. Hiển thị các thông điệp: Các thông điệp của ứng dụng hiển thị chính giữa màn hình. Với tiêu đề, nội dung và các phim xác nhận (nếu có). Nội dung của thông điệp ngắn gọn, dễ hiểu, như **Hình 34**:



**Hình 34** Mẫu quy chuẩn thông điệp

1. Nguyên tắc phối màu: ứng dụng sử dụng 2 màu chính là Jungle Green (#26A69A) và màu trắng (**Hình 35**). Thiết kế cần phối hợp hai màu này một cách hợp lý để làm nổi bật các thông tin nhưng vẫn đảm bảo tính đơn giản cho giao diện hệ thống.



**Hình 35** Hai màu cơ bản của ứng dụng

### Thiết kế cơ sở dữ liệu

#### Yêu cầu của bài toán

Một người dùng khi sử dụng hệ thống có thể đăng ký làm một trong hai loại người dùng : Lái xe, khách hàng. Các thông tin của người dùng bao gồm họ tên, số điện thoại, địa chỉ email, hình ảnh đại diện.

Lái xe sở hữu phương tiện giao thông với các thông tin: biển số, và loại phương tiện (xe máy, ô tô), hãng sản xuất. Tài xế cũng cần cung cấp cho hệ thống các thông tin về giấy phép lái xe, giấy tờ xe.

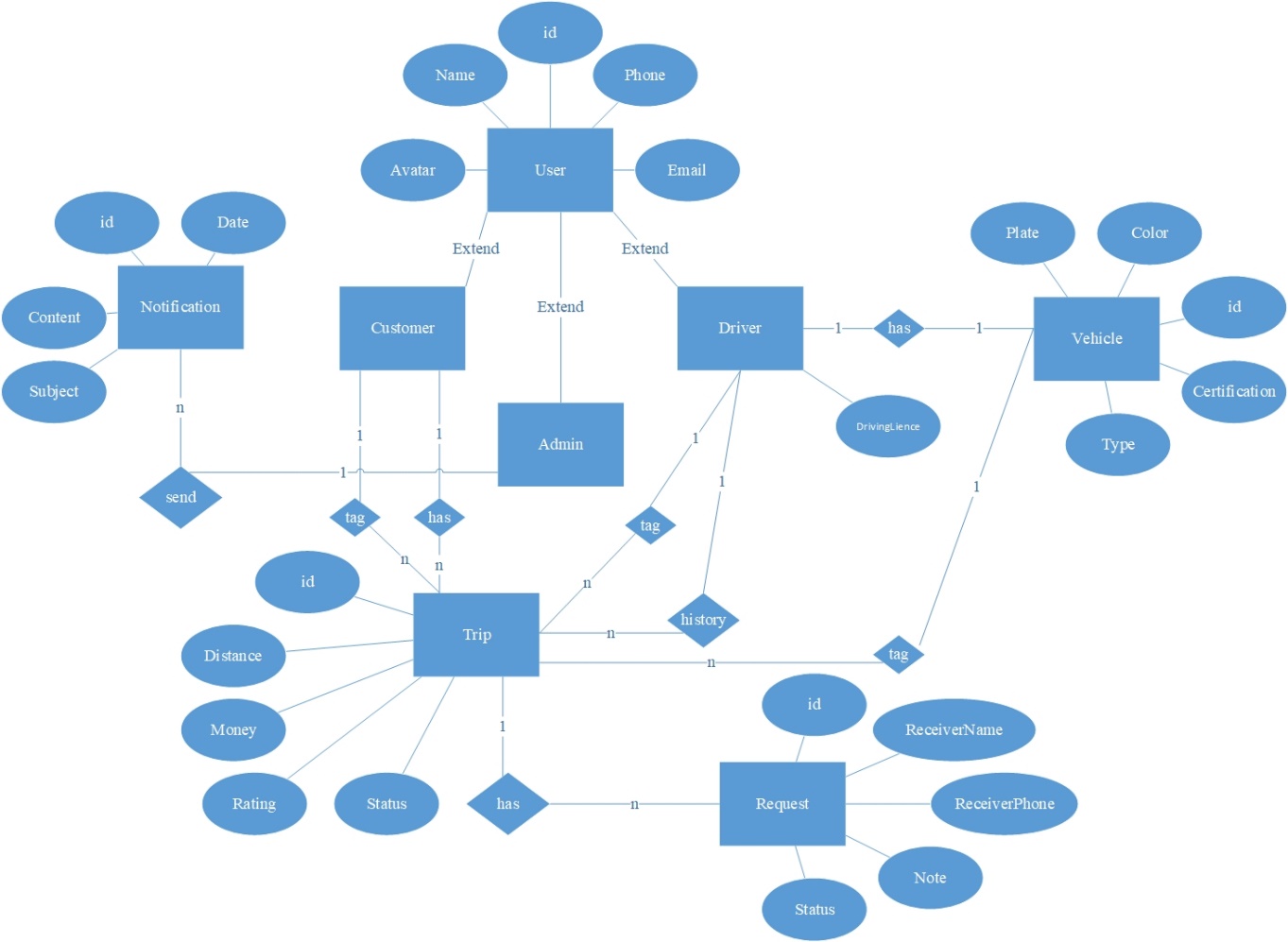
Khi khách hàng thực hiện yêu cầu gọi xe hệ thống sẽ tự động lưu dưới dạng hành trình với thông tin của khách hàng, thông tin tài xế, phương tiện sử dụng, các thông tin về dịch vụ sử dụng, quãng đường phát sinh, chi phí sử dụng, trạng thái của hành trình… Cả khách hàng và tài xế đều có thể xem lại thông tin hành trình mà mình tham gia

Mỗi 1 hành trình có thể có 1 hoặc nhiều điểm đến. Mỗi điểm đến chứa thông tin của người nhận, loại hàng hóa, ghi chú cho tài xế, và trạng thái xử lý.

Hệ thống có quản trị viên có thể gửi thông báo đến người dùng, nội dung của thông báo bao gồm: chủ đề và nội dung, ngày gửi thông báo.

#### Biểu đồ thực thể liên kết

Từ những yêu cầu bài toán ở trên có thể xây dựng thành biểu đồ thực thể liên kết sau: (**Hình 36**)



**Hình 36** Sơ đồ thực thể liên kết

Trong đó User là thực thể đại diện cho người dùng với các thông tin về id, tên người dùng, số điện thoại, email, và đường dẫn đến hình đại diện.

Customer,Admin, Driver là 3 thực thể con của User ứng với 3 loại người dùng của hệ thống.

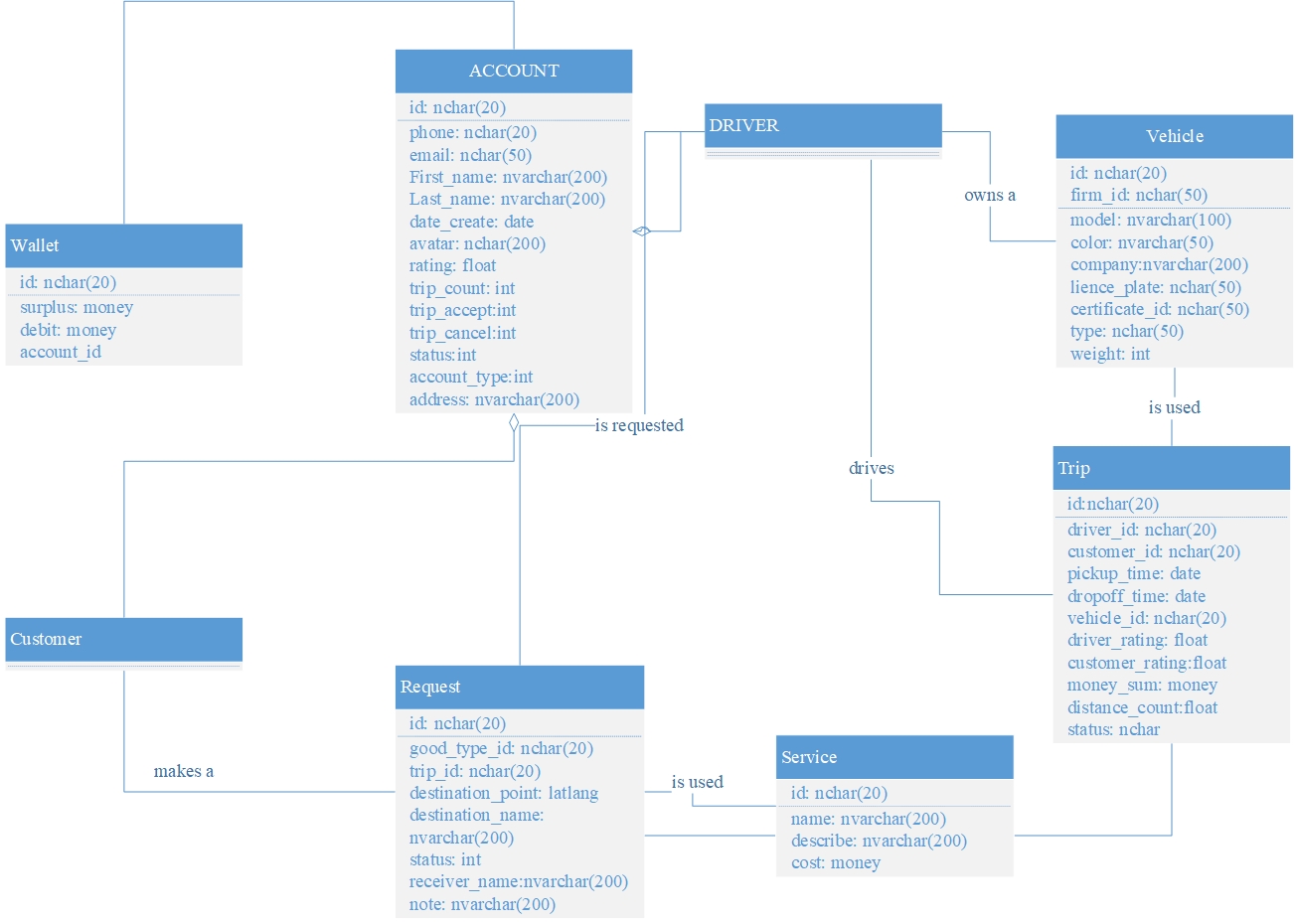
Trip là thông tin hành trình được hệ thống tạo ra khi người dùng yêu cầu giao hàng và tài xế chấp nhận yêu cầu đó. Bao gồm các thông tin về id, quãng đường, số tiền thanh toán, đánh giá, trạng thái hành trình. Mỗi khách hàng và tài xế có thể có nhiều hành trình và có thể lưu hành trình đó vào lịch sử sử dụng.

Một Trip có thể có nhiều điểm dừng gọi là Request, bao gồm các thông tin về id, tên người nhận, số điện thoại người nhận, loại hàng hóa vận chuyển, ghi chú,..

Mỗi tài xế sở hữu một loại phương tiện biểu diễn dưới dạng thực thể Device với các thông tin về id, biển số, màu sắc, hãng, loại (ô tô, xe máy). Tài xế dùng phương tiện trong mỗi hành trình xử lý yêu cầu của khách hàng.

Admin có thể gửi thông báo đến người dùng dưới dạng thực thể liên kết Notification với các thông tin về id, tiêu đề, ngày đăng, nội dung thông báo.

#### Biểu đồ lớp



**Hình 37** Biểu đồ lớp đối tượng của hệ thống

#### Mô tả chi tiết lớp

* Lớp Account: lớp này mô tả các thông tin của tài khoản người dùng hệ thống.

**Bảng 11** Mô tả chi tiết lớp Account

|  |  |  |
| --- | --- | --- |
| Tên | Kiểu | Mô tả |
| id | int | Mã người dùng |
| phone | nchar | Số điện thoại |
| email | nchar | Địa chỉ thư điện tử |
| first\_name | nchar | Tên người dùng |
| last\_name | nchar | Họ người dùng |
| date\_created | date | Ngày kích hoạt |
| avartar | nchart | Đường dẫn ảnh đại diện |
| rating | int | Độ đánh giá |
| trip\_count | int | Số lượng dịch vụ đã sử dụng |
| trip\_accept | int | Số lượt dịch vụ thành công |
| trip\_cancel | int | Số lượt dịch vụ thất bại |
| status | boolean | Trạng thái (chặn/ không bị chặn) |
| account\_type | int | loại tài khoản |

* Lớp Vehicle: lớp này mô tả các thông tin phương tiện của tài xế

**Bảng 12** Mô tả chi tiết lớp Vehicle

|  |  |  |
| --- | --- | --- |
| Tên | Kiểu | Mô tả |
| id | nchar | Mã phương tiện |
| firm\_id | nchar | Biển số xe |
| ;model | nchar | Loại xe |
| color | nchar | Màu xe |
| company | nvarchar | Hãng xe |
| lience\_palate | nchar | Số đăng ký xe |
| type | int | Loại xe (ô tô hoặc xe máy) |
| weight | int | Tải trọng của phương tiện |

* Lớp Request: lớp này mô tả một yêu cầu điểm dừng trên hành trình

**Bảng 13** Mô tả chi tiết lớp Request

|  |  |  |  |
| --- | --- | --- | --- |
| Tên | Kiểu | Mô tả | |
| id | int | Mã yêu cầu |
| good\_type | nchar | Mã loại hàng hóa |
| destination\_point | latlang | Điểm đến |
| Destination\_name | Nvarchar | Tên điểm đến |
| trip\_id | nchar | Mã đơn hàng |
| Receiver\_name | Nchar | Tên người nhận |
| note | nchar | Ghi chú |
| status | int | Trạng thái |
| Receiver\_phone | Nchar | Số điện thoại người nhận |

* Lớp Trip: lớp này mô tả một yêu cầu điểm dừng trên hành trình

**Bảng 14** Mô tả chi tiết lớp Trip

|  |  |  |  |
| --- | --- | --- | --- |
| Tên | Kiểu | Mô tả | |
| id | int | Mã đơn hàng |
| driver\_id | nchar | Mã tài xế |
| customer\_id | nchar | Mã khách hàng |
| pickup\_time | date | Thời gian nhận |
| vehicle\_id | ncahr | Mã phương tiện |
| driver\_rating | int | Đánh giá tài xế |
| customer\_rating | int | Đánh giá khách hàng |
| money\_sum | money | Tổng chi phí phát sinh |
| distance\_count | float | Tổng quãng đường |
| status | nchar | Trạng thái của hành trình |

* Lớp Wallet: lớp này mô tả thông tin ví của khách hàng

**Bảng 15** Mô tả chi tiết lớp Wallert

|  |  |  |  |
| --- | --- | --- | --- |
| Tên | Kiểu | Mô tả | |
| id | int | Mã ví |
| surplus | money | số dư |
| debit | money | số nợ |
| account\_id | nchar | Mã người dùng |

* Lớp Service: Bảng này mô tả các loại dịch vụ riêng mà tài xế cung cấp. Vd: bốc xếp, dỡ hàng lên phương tiện, chi phí đỗ xe, chi phí cầu đường

**Bảng 16** Mô tả chi tiết lớp Service

|  |  |  |  |
| --- | --- | --- | --- |
| Tên | Kiểu | Mô tả | |
| id | int | Mã dịch vụ |
| name | nvarchar | Tên dịch vụ |
| describe | nvarchar | Mô tả về dịch vụ |
| cost | money | Giá dịch vụ |

## Xây dựng ứng dụng

### Thư viện và công cụ sử dụng

**Bảng 17** Các thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| **Kiểu loại** | **Công cụ** | **Địa chỉ URL** |
| IDE lập trình | Android Studio 3.1 | http://www.eclipse.org/ |
| Console server | Firebase Console | https://console.firebase.google.com |
| Tùy chỉnh giao diện | Nice Spinner | https://github.com/arcadefire/nice-spinner |
| CircleImageView | https://github.com/hdodenhof/CircleImageView |
| Country Code Picker | https://github.com/joielechong/CountryCodePicker |
| Cat Loading View | https://github.com/Rogero0o/CatLoadingView |
| Xử lý bản đồ | Google MAP API | https://cloud.google.com/maps-platform/ |
| Tính toán đường đi | Google Direction Library | https://github.com/akexorcist/Android-GoogleDirectionLibrary |

### Kết quả đạt được

Dựa trên các kiến trúc thiết kế, công nghệ sử dụng đã đề cập ở các chương trước, SV đã hoàn thành và cho ra đời ứng dụng gọi xe giao hàng online với các tính năng sau:

* 1. Hệ thống đã được phát triển hoàn chỉnh bao gồm cả phía client và server, có thể chạy thử nghiệm (demo).
  2. Qua cài đặt trong thực tế thì phần mêm hoạt động ổn định, cung cấp được nhiều dịch vụ cho người dùng.
  3. Phản hồi giữa người dùng và hệ thống thực hiện một cách nhanh chóng, chính xác.
  4. Cho phép khách hàng và tài xế tham gia vào hệ thống.
  5. Nâng cao khả năng an toàn, an ninh thông tin của hệ thống thông qua giải pháp xác thực người dùng.

Thống kê ứng dụng:

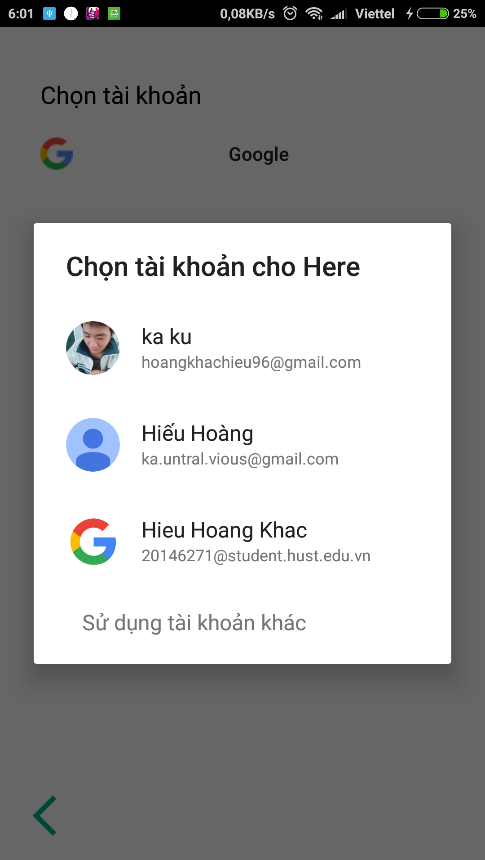
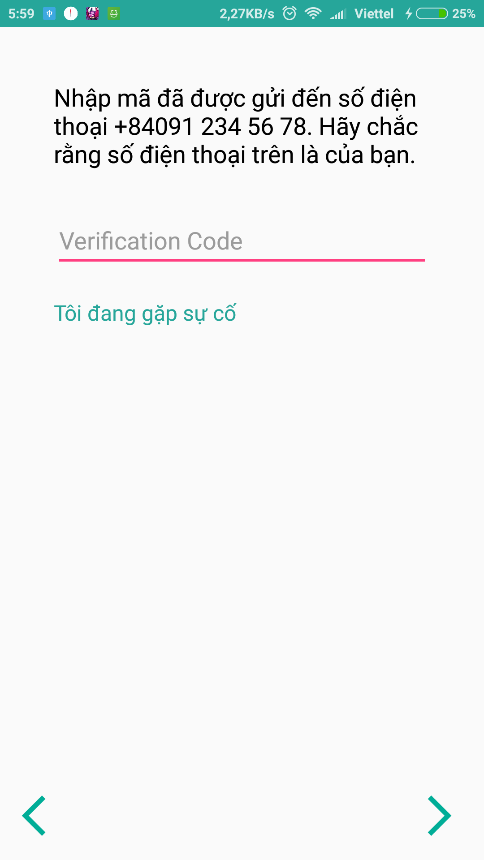
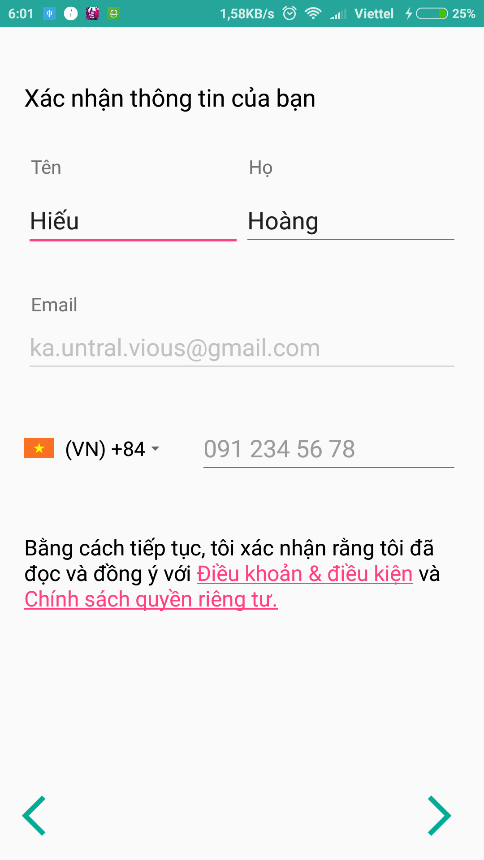
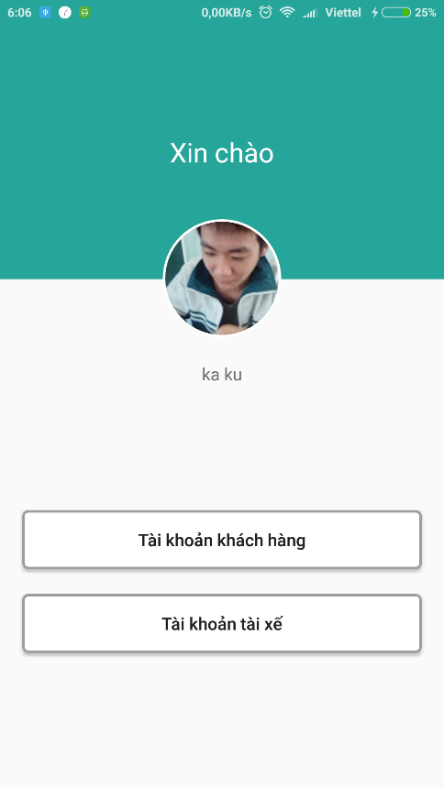
**Bảng 18** Thông tin thống kế về ứng dụng

|  |  |
| --- | --- |
| Nội dung | Thống kê |
| Số dòng code | 20000 dòng code |
| Số lớp | 10 lớp |
| Số gói | 10 gói |
| Dung lượng mã nguồn | 258MB |
| Dung lượng sản phẩm đóng gói | 37MB |

### Mình hoạ các chức năng chính

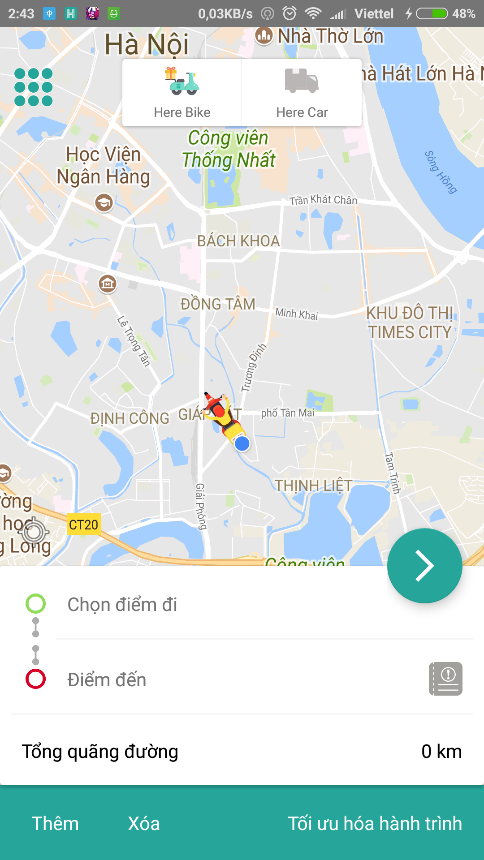
#### Giao diện màn hình đăng ký

Với giao diện màn hình đăng ký, người dùng lựa chọn đăng ký bằng tài khoản gmail, hệ thống sẽ lấy thông tin được cung cấp từ Gmail của người dùng và hiển thị lên form đăng ký. Người dùng thay đổi thông tin của mình và điền số điện thoại xác nhận tại form này. Hệ thống sẽ gửi mã xác mình đến số điện thoại người dùng đăng ký. Tại màn hình xác nhận mã khách hàng điền mã xác nhận vào đề hoan tất đăng ký tài khoản. (**Hình 38**)

**Hình 38** Giao diện màn hình đăng ký của ứng dụng

### Giao diện màn hình gọi xe của khách hàng



Mở Menu chính

Lựa chọn loại phương tiện

Tài xế xung quanh

Danh sách điểm dừng

Yêu cầu gọi xe

Tối ưu hóa lại đường đi

Thêm/ xóa điểm dừng

**Bản đồ**

**Hình 39** Giao diện màn hình gọi xe của ứng dụng

Với giao diện màn hình gọi xe, khác hàng có thể theo dõi số lượng các tài xế xung quanh khu vự hiện tại của mình, chọn loại phương tiện vận chuyển theo yêu cầu. Tạo hành trình với điểm xuất phát và các điểm dừng. Tối ưu hóa lại đường đi với chức năng tối ưu hóa hành trình và thực hiện yêu cầu gọi xe. (**Hình 39**)

#### Giao diện màn hình nhận yêu cầu gọi xe của tài xế



Thông tin ví tài khoản

Vị trí hiện tại của tài xế

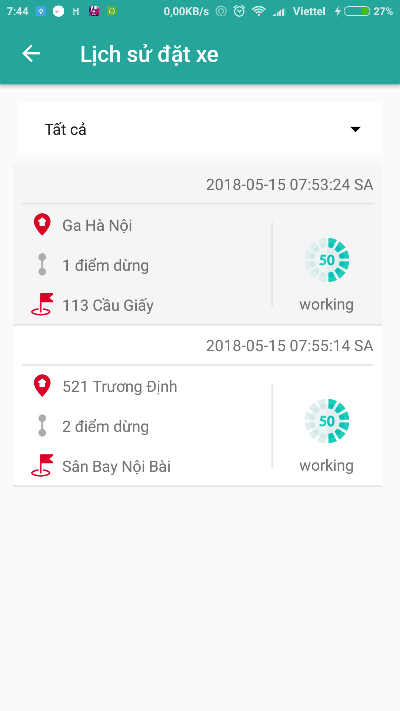
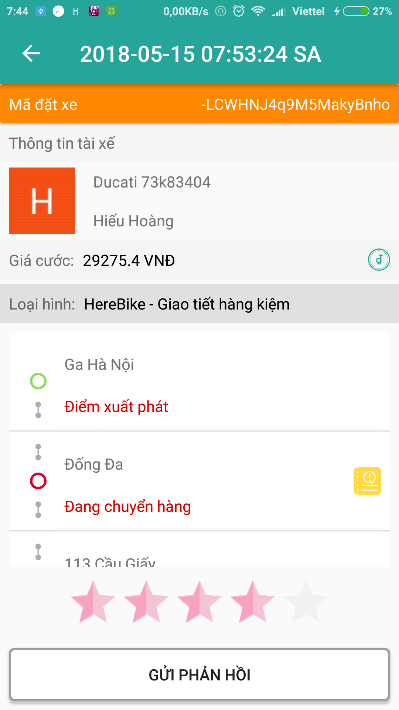
Bản đồ hoạt động của tài xế

Trạng thái hoạt động hiện tại

Kích hoạt trạng thái hoạt động

**Hình 40** Giao diện màn hình nhận yêu cầu gọi xe của ứng dụng

#### Giao diện màn hình lịch sử đặt xe

Thông tin chi tiết lịch sử

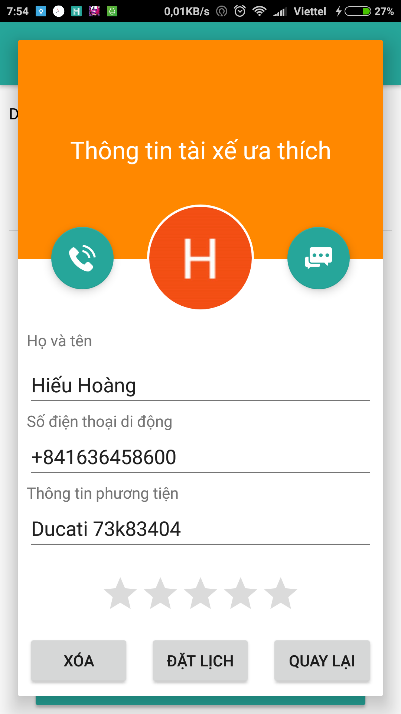
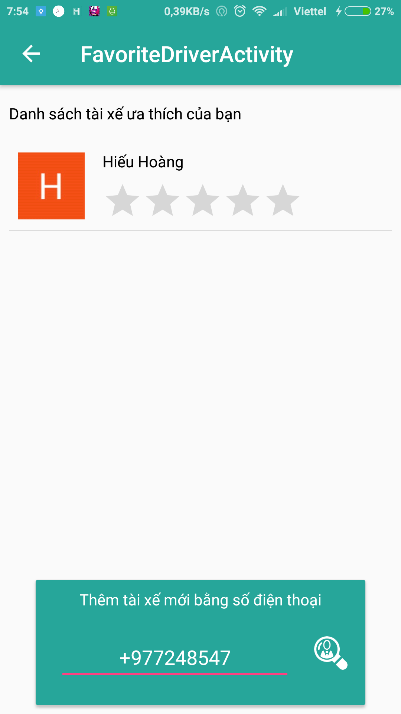
Danh sách lịch sửcccccccccccccccccccccc

Bộ lọc lịch sử

**Hình 41** Giao diện màn hình lịch sử đặt xe của ứng dụng

Với màn hình lịch sử đặt xe (**Hình 41**), tài xế và khách hàng có thể xem lại danh sách các đơn hàng mình đã nhận/đã đặt, tình trạng của đơn hàng

#### Giao diện tài xế ưa thích



Chi tiết tài xế ưa thích

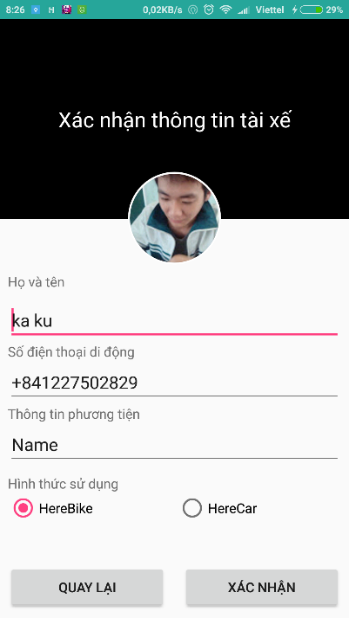
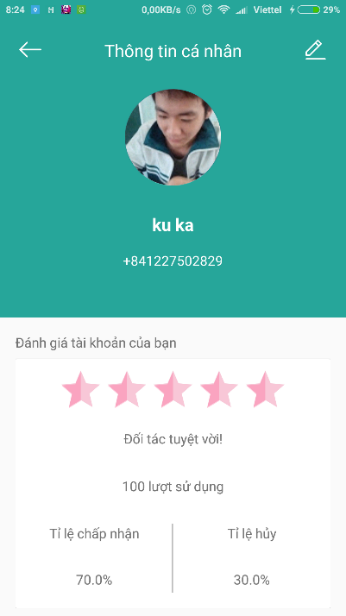
Thêm tài xế ưa thích

Danh sách tài xế ưa thích

**Hình 42** Giao diện màn hình tài xế ưa thích của ứng dụng

Với giao diện tài xế ưa thích (**Hình 42**), khách hàng có thể quản lý danh sách các tài xế mà mình ưa thích để dễ dàng liên hệ, đặt yêu cầu gọi xe trực tiếp với tài xế ưa thích

#### Giao diện thông tin cá nhân



Giao diện thông tin cá nhân tài xế

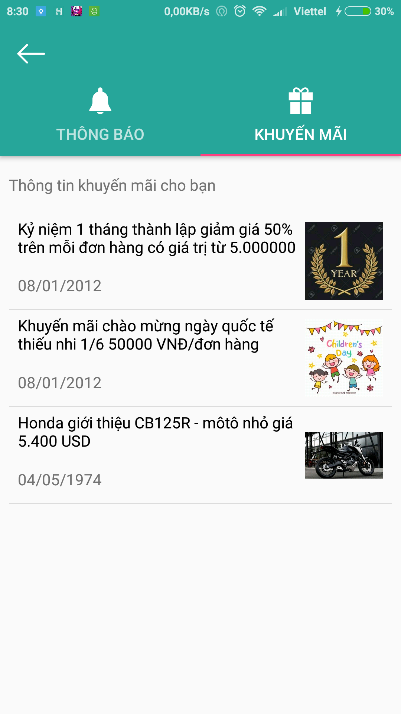
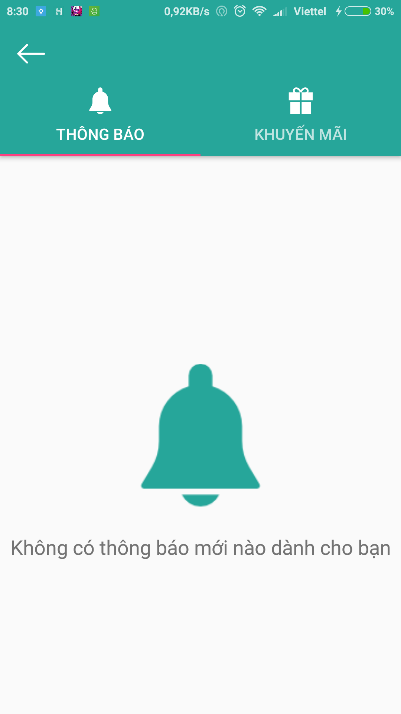
Giao diện thông tin cá nhân khách hàng

**Hình 43** Giao diện màn hình thông tin cá nhân của ứng dụng

Với giao diện thông tin cá nhân (**Hình 43**) tài xế và khách hàng có thể quản lý các thông tin cá nhân của mình.

#### Giao diện thông báo

Giao diện thông báo (**Hình 44**) hiển thị các thông tin thông báo mới dành cho người dùng. Các chương trình khuyến mãi mà quản trị viên đưa ra.



Khu vực hiển thị thông báo mới

Khu vực hiển thị thông tin khuyến mãi

**Hình 44** Giao diện màn hình thông báo của ứng dụng

# Các giải pháp và đóng góp nổi bật

Nội dung chương 5 tập trung vào phân tích các giải pháp, giải thuật mà SV đã đưa ra để giải quyết các vấn đề mà bài toán đặt ra.

## Giải pháp luồng xử lý yêu cầu gọi xe của khách hàng

### Giới thiệu bài toán

Yêu cầu gọi xe là một trong những chức năng cơ bản của ứng dụng. Khi sử dụng chức năng này người dùng sẽ được kết nối với các tài xế trong vòng bán kính 5km theo mô hình 4 bước:

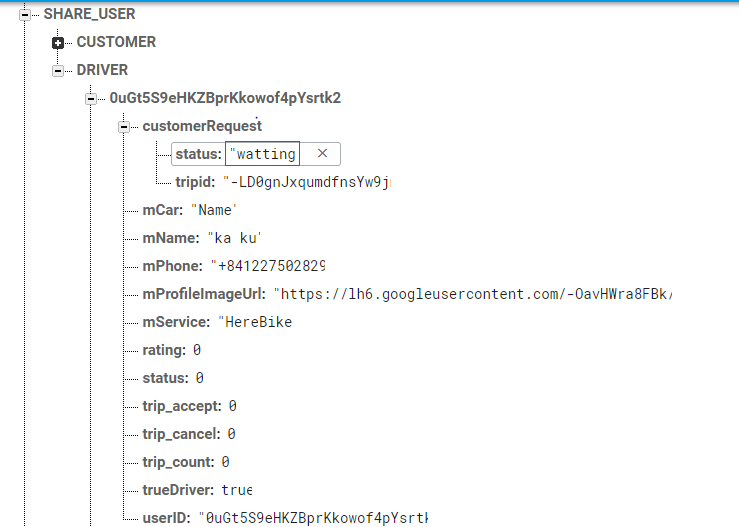
1. Bước 1 (Yêu cầu gọi xe): Khách hàng sử dụng ứng dụng trên điện thoại thông mình để thiết lập 1 đơn giao hàng và yêu cầu một tài xế xe máy/ ô tô để thực hiện đơn hàng ấy.
2. Bước 2 (Kết hợp): Ngay sau khi yêu cầu được gửi đi, thông báo về chi tiết đơn hàng sẽ được gửi đến tài xế gần nhất. Tài xế có tùy chọn chấp nhận hoặc từ chối yêu cầu. Trong trường hợp tài xế từ chối, yêu cầu sẽ được gửi đến một tài xế khác trong khu vực đó.
3. Bước 3 (Chuyển hàng): Khách hàng có thể theo dõi quá trình di chuyển của tài xế từ lúc tài xế di chuyển đến điểm nhận hàng và lúc chuyển hàng đi.
4. Bước 4 (Thanh toán & Đánh giá): Khi chuyến đi kết thúc, khách hàng sẽ có tùy chọn để xếp hạng tài xế. Hệ thống xếp hạng là một phần quan trọng vì nó cho phép khách hàng tin tưởng tài xế khi trao hàng hóa để chuyển đi.

Phía server của hệ thống sử dụng Realtime Database với đặc điểm là cloud hosted database. Tất cả dữ liệu được lưu trữ ở định dạng JSON và với bất kể một sự thay đổi dữ liệu nào thì có sự phản hồi ngay lập tức. Do đó để xử lý quá trình gọi xe một cách logic cần có sự tổ chức CSDL một cách hợp lý.

### Giải pháp xử lý

Để giải quyết bài toán gọi xe đặt ra ở trên, SV xây dựng CSDL Realtime Database cho yêu cầu gọi xe theo mô hình nhánh hoạt động:

1. **Nhánh trạng thái (SHARED\_USER):** Mỗi tài xế khi đăng ký tài khoản ngoài việc lưu các thông tin của tài xế ở nhánh USER thì thông tin tài xế cũng được lưu ở nhánh trạng thái - SHARED\_USER. Nhánh này chỉ lưu các thông tin cơ bản của tài xế để cung cấp cho khách hàng, tránh lộ các thông tin khác của tài xế.

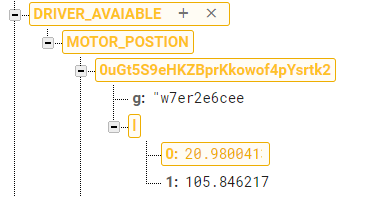


Thuộc tính này chỉ tồn tại khi có yêu cầu gọi xe được kết nối với tài xế

**Hình 45** nhánh trạng thái - SHARED\_USER

Tại nhánh trạng thái này có 1 thuộc tính là customerRequest, thuộc tính này chỉ tồn tại khi có yêu cầu gọi xe được kết nối đến tài xế

1. **Nhánh lắng nghe** - **DRIVER\_AVAIABLE:** Khi tài xế kích hoạt chức năng nhận yêu cầu gọi xe, 1 bộ thông tin tài xế với key là mã id tài xề và giá trị là vị trí hiện tại sẽ được đẩy vào nhánh lắng nghe - DRIVER\_AVAIABLE. Nhánh này bao gồm các tài xế đang kích hoạt chế độ nhận yêu cầu gọi xe và chưa nhận yêu cầu nào. Tại nhánh lắng nghe thông tin về vị trí của tài xế sẽ tồn tại và được cập nhật liên tục theo thời gian cho đến khi tài xế tắt chế độ nhận yêu cầu đi hoặc tài xế đã chấp nhận 1 yêu cầu từ khách hàng.



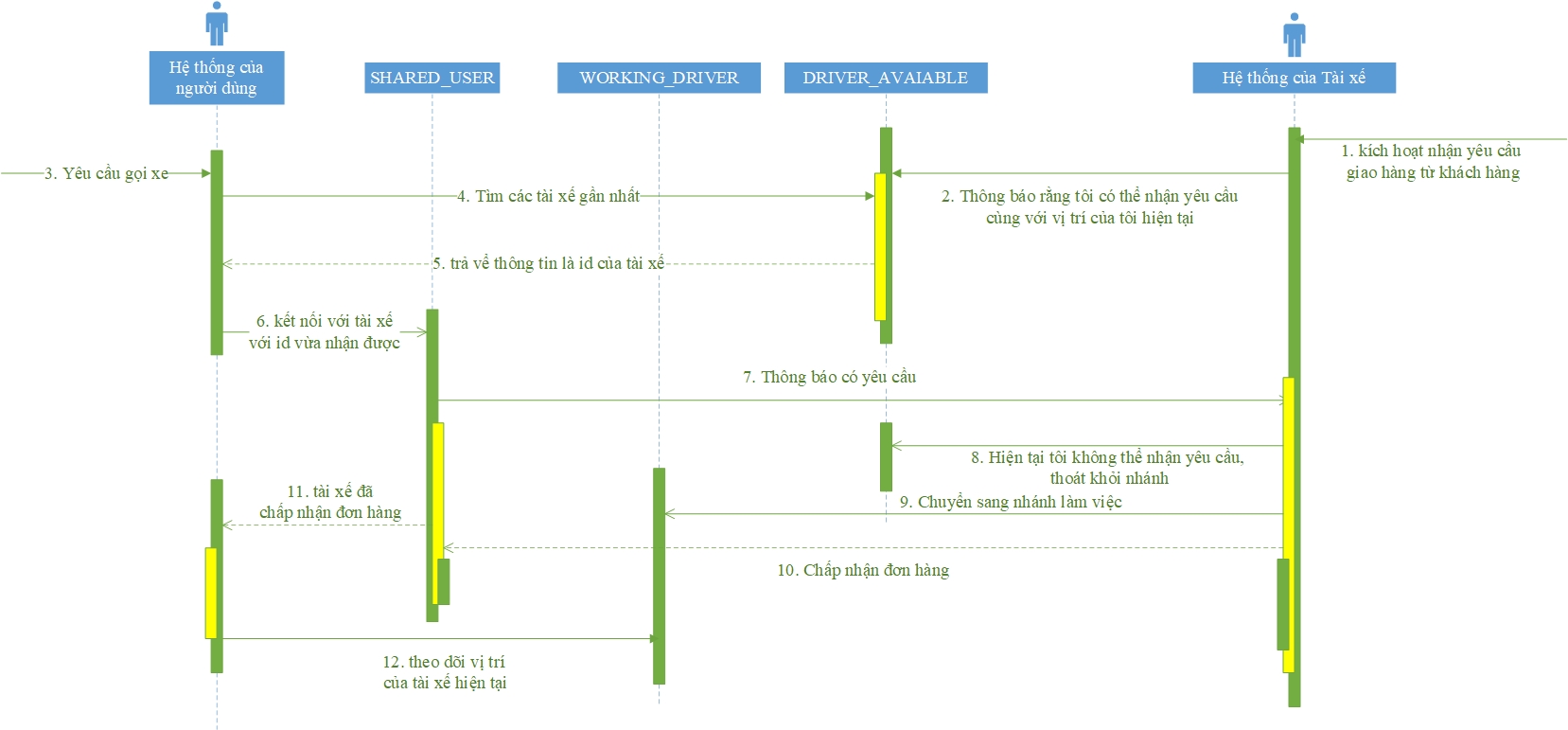
Thông tin về vị trí

Id tài xế

**Hình 46** nhánh lắng nghe - DRIVER\_AVAIABLE

1. **Nhánh tiếp nhận – WORKING\_DRIVER:** Khi tài xế tiếp nhận một đơn hàng thì thông tin của tài xế trong nhánh lắng nghe sẽ bị xóa đi và chuyển vào nhánh tiếp nhận.

Với ba nhánh trên ta có thể xây dựng nên biểu đồ hoạt động xử lý yêu cầu bài toán như **Hình 47**:

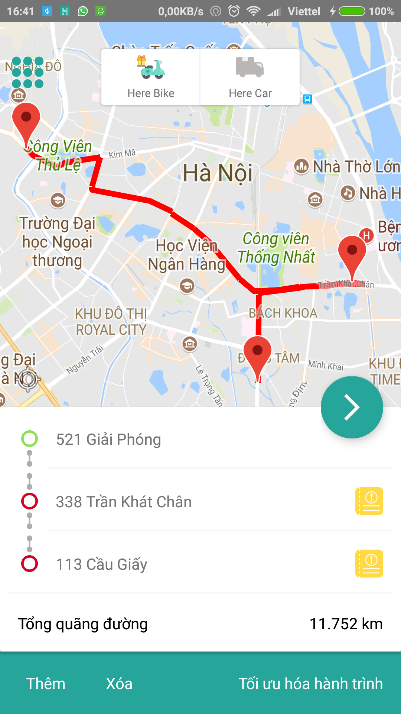
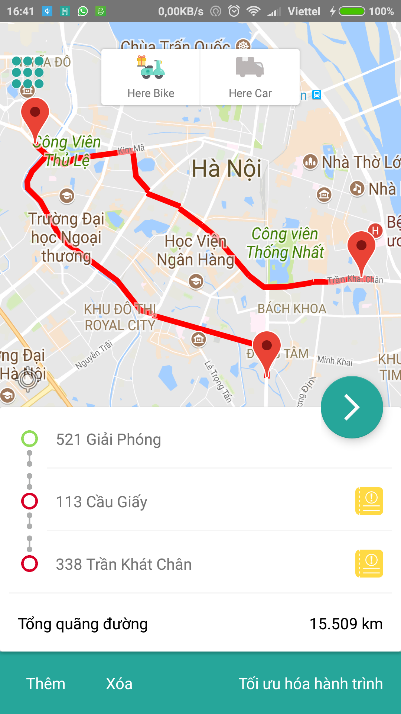


**Hình 47** Biểu đồ hoạt động gọi xe

## Bài toán tối ưu hóa quãng đường

### Giới thiệu bài toán

Với việc xây dựng chức năng mới cho phép một khách hàng đặt nhiều điểm dừng trên một đơn hàng (tối đa 5 điểm dừng). Ứng dụng đã giúp khách hàng giảm thiểu thời gian, đảm bảo tính rõ ràng khi yêu cầu giao hàng. Tuy nhiên khi xây dựng chức năng này, nảy sinh một vấn đề là khi người dùng nhập các điểm dừng vào thì thường nhập theo thứ tự trí nhớ chứ không theo thứ tự khoảng cách giữa các điểm dừng sao cho ngắn nhất. Do đó gây nên hiện tượng quãng đường di chuyển hệ thống tính ra không phải là quãng đường ngắn nhất, phát sinh chi phí thêm rất nhiều. Ví dụ như **Hình 48** cùng là 3 điểm dừng nhưng đi theo hướng “521 Giải Phóng – 338 Trần Khát Chân – 113 Cầu Giấy” ngắn hơn hẳn gần 4km so với hướng “521 Giải Phóng - 113 Cầu Giấy - 338 Trần Khát Chân”.



**Hình 48** So sánh quãng đường khi chưa tối ưu và đã tối ưu

Chính vì lý do trên, hệ thống cần tích hợp một chức năng giúp tối ưu hóa lại các điểm dừng sao cho quãng đường di chuyển ngắn nhất.

### Giải pháp xử lý

Tới thời điểm hiện tại, Google Map vẫn chưa hỗ trợ chức năng tính đường đi gần nhất giữa các điểm dừng cho nên cần phải có một giải pháp thay thế để giải quyết vấn đề. Chính vì vậy SV đề xuất nên phương pháp tính đường đi ngắn nhất giữa các điểm dừng trên bản đồ với giới hạn tối đa 10 điểm dừng.

Phương pháp này dựa trên tính năng tính quãng đường di chuyển giữa các điểm dừng của Google Direction API kết hợp với việc xây dựng các bộ hoán vị không lặp các điểm dừng. Từ đó chọn ra bộ hoán vị có quãng đường nhỏ nhất.

Cụ thể việc tìm tối ưu hóa tuyến đường được thể hiện qua đoạn code sau:

**Bảng 19** Code tối ưu hóa tuyến đường

|  |
| --- |
| *// Hàm tối ưu hóa tuyến đường* private void optimizeTrip() {  *// khởi tạo cho khoản cách luôn là lớn nhất* maxDistance = 10000000;   *// sao chép lại bộ các điểm dừng lúc chưa tối ưu* List<Request> list = new ArrayList<>();  for (int i = 1; i < tripRequests.size(); i++) {  list.add(tripRequests.get(i));  }  *// khởi tạo danh sách lưu các bộ hoán vị điểm dừng* listChoice = new ArrayList<>();  *// Tính các bộ hóa vị* printPermutation(list, 0, true);  *// tìm bộ hoán vị có tuyến đường đi nhỏ nhất*  *// dựa trên Direction API của Google cung cấp* caculatorTripDistance(0); }  private void printPermutation(List<Request> array, int start, boolean display) {  if (display) {  List<Request> requests = new ArrayList<>();  for (Request item : array) {  requests.add(item);  }  listChoice.add(requests);  }   for (int j = start; j < array.size(); j++) {  Request temp = array.get(start);  array.set(start, array.get(j));  array.set(j, temp);  if (j == start) {  printPermutation(array, start + 1, false);  } else {  printPermutation(array, start + 1, true);  }  temp = array.get(start);  array.set(start, array.get(j));  array.set(j, temp);  } } |

Chú ý việc tính toán khoảng cách tuyến đường giữa các điểm trong bộ hoán vị chỉ được thực hiện sau khi đã tạo xong danh sách các bộ hoán vị do độ trễ phản hồi của chức năng tính khoảng cách đường đi giữa các điểm của Google Direction API.

Những giải pháp kể trên là những đóng góp lớn vào việc xây dựng một ứng dụng với các chức năng hoàn chỉnh, tiện ích cho người dùng. Mỗi giải pháp là một điều tâm đắc trong suốt quá trình làm ĐATN của SV.

# Kết luận và hướng phát triển

## Kết luận

Qua quá trình thực hiện ĐATN em đã thu được các kết quả sau:

* Xây dựng thành công ứng dụng Gọi xe chuyển hàng Online với đầy đủ các chức năng cơ bản đã đặt ra, tích hợp vào một số tính năng mới giúp giải quyết các vấn đề còn tồn đọng so với các ứng dụng khác trên thị trường như chức năng phân loại xe, chuyển hàng tiết kiệm, tối ưu hóa đường đi, …
* Tìm hiểu và làm chủ được một số công nghệ quan trọng trong việc lập trình ứng dụng cho thiết bị di động hiện nay:

1. Tìm hiểu và nắm được các kiến thức cơ bản về các dịch vụ mà Google cung cấp: Google Maps API, Firebase.
2. Nắm được những kiến thức cơ bản về lập trình trên nền tảng Android
3. Cũng cố kiến thức về phân tích, thiết kế và phát triển phần mềm, hệ thống..

* Nắm vững được việc xây dựng một quy trinh phát triển phần mềm trọn vẹn, từ đặt vấn đề, phân tích thiết kế, phát triển sản phẩm, test chạy thử, đến việc nghiên cứu các giải pháp công nghệ để giải quyết các vấn đề bài toán đặt ra.
* Rèn luyện khả năng phân tích và làm việc độ lâpk

## Hướng phát triển

Qua đánh giá hệ thống em nhận thấy sản phẩm của mình vẫn còn nhiều hạn chế vì thế trong tương lai em sẽ cố gắng hoàn thiện ứng dụng, khắc phục những nhược điểm còn tồn tại của ứng dụng, những lỗi phát sinh trong quá trình sử dụng ứng dụng

Để ứng dụng ngày một hoàn thiện và có thể triển khai thực tế một cách rộng rãi, đem đến một sự trải nghiệm tốt nhất cho người dùng thì phải bổ sung và cải thiện một số chức năng sau:

1. Cải thiện lại giao diện cho ứng dụng một cách đẹp hơn, có tính thẩm mỹ cao, dễ sử dụng.
2. Tối ưu lại về cách quản lý cơ sở dữ liệu.
3. Nghiên cứu và phát triển thêm một số tính năng như: cho phép thanh toán qua ví điện tử, tính năng gợi ý khu vực có nhu cầu chuyển hàng cao cho tài xế, tính năng hẹn giờ đặt lịch gọi xe, hỗ trợ thêm nhiều loại phương tiện khác, dịch vụ khác, …

# Tài liệu tham khảo

1. “Google Maps API”. Google Developers. Website:https://developers.google.com/maps
2. “Firebase Documentation”. Google Developers. Last Release 2018-05
3. “Material Degisn”. Google Developers. Website: https://material.io
4. “Google Announces Launch of Google Maps for mobile with my Location Technology”. News from Google. Google. November 28, 2007. Retrieved April 25, 2017
5. “Phân tích thiết kế hệ thống thông tin”. Nguyễn Văn Ba. NXB Giáo Dục. Retrieved 2004.
6. “Phát triển hệ thống hướng đối tượng với UML 2.0 và C++”. Nguyễn Văn Ba
7. “Các biểu mẫu Graduation Research”. HaNoi University of Science and Technology