TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

──────── \* ───────

ĐỒ ÁN

**TỐT NGHIỆP ĐẠI HỌC**

NGÀNH CÔNG NGHỆ THÔNG TIN

**TÊN ĐỀ TÀI**

**XÂY DỰNG ỨNG DỤNG GỌI XE ONLINE**

Sinh viên thực hiện : **Hoàng Khắc Hiếu**

Lớp CN - CNTT K59

Giáo viên hướng dẫn : **TS**. **Nguyễn Nhất Hải**

HÀ NỘI 2018

PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

1. Thông tin về sinh viên

* Họ và tên sinh viên: Hoàng Khắc Hiếu
* Điện thoại liên lạc: 01636458600 Email: [20146271@student.hust.edu.vn](mailto:20146271@student.hust.edu.vn)
* Lớp: CN - CNTT K59 Hệ đào tạo: Đại học chính quy
* Đồ án tốt nghiệp được thực hiện tại: Bộ môn Công nghệ phần mềm, Viện Công nghệ thông tin và Truyền thông, Đại học Bách Khoa Hà Nội.
* Thời gian làm ĐATN: Từ 01/2018 đến xx/2018.

1. Mục đích nội dung của ĐATN

* Xây dựng hệ thống gọi xe online trên thiết bị di động

1. Các nhiệm vụ cụ thể của ĐATN

* Khảo sát nhu cầu người dùng về hệ thống tra cứu gọi xe online
* Khảo sát các ứng dụng trên thị trường có chức năng tương tự.
* Phân tích các chức năng cần có cho hệ thống.
* Tìm hiểu các công nghệ, giải pháp cho hệ thống.
* Phân tích, thiết kế và xây dựng hệ thống.

1. Lời cam đoan của sinh viên:

* Tôi *Hoàng Khắc Hiếu* cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của *TS.* *Nguyễn Nhất Hải*.
* Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

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

1. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

|  |  |
| --- | --- |
|  | *Hà Nội, ngày tháng năm 2018*  Giáo viên hướng dẫn  *TS. Nguyễn Nhất Hải* |

TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

**Ngày nay, với sự phổ cập rộng rãi của minh, điện thoại di động đã trở thành một trong những công cụ liên lạc thiết yếu của con người. Ngoài chức năng liên lạc, cùng với sự phát triển của công nghệ, kết nối toàn cầu, điện thoại di động còn được trang bị nhiều ứng dụng giải trí, định vị, mua sắm, thanh toán trực tuyến,… Các ứng dụng di động cũng được chú trọng phát triển, ứng dụng nhiều công nghệ mới, 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 đi lại, 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 đi lại, vận chuyển riêng cho mình. Không có khả năng quảng cáo, tiếp cận thị trường khiến cho những tài xế cung cấp dịch vụ đi lại, vận chuyển gặp khó khăn trong việc 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 đi lại, vận chuyển, gây ra nhiều bất cập.

Chính vì nhu cầu cấp thiết đó, với mục tiêu xây dựng một hệ thống thông tin kết nối giữa tài xế với người dân có nhu cầu đi lại, tận dụng thế mạnh của thiết bị di động thông minh đồng thời kết hợp nghiên cứu phân tích các ứng dụng đã có trên thị trường. Đồ án tập trung xây dựng thiết kế tổng thể cho ứng dụng, tích hợp các công nghệ , dịch vụ mới, hữu ích cho người sử dụng.

Nội dung báo cáo gồm các phần sau:

**Chương 1:** *Đặt vấn đề.*

Nội dung chương 1 tập trung vào giới thiệu bài toán, tìm hiểu yêu cầu và đưa ra định hướng giải pháp và công nghệ cho bài toán.

**Chương 2:** *Công nghệ sử dụng.*

Nội dung chương 2 trình bày tóm tắt về các công nghệ được sử dụng trong ứng dụng.

**Chương 3:** *Khảo sát và mô hình hóa yêu cầu.*

Nội dung chương 3 dựa vào các khảo sát nhu cầu người dùng và tiến hành phân tích chức năng của hệ thống.

**Chương 4:** *Phân tích, thiết kế, xây dựng hệ thống và đánh giá kết quả.*

Nội dung chương 4 trình bày các phân tích thiết kế của hệ thống, thiết kế cấu trúc dữ liệu và xây dựng phát triển hệ thống.

**Chương 5:** *Kết luận và hướng phát triển.*

Chương 5 sẽ trình bày tóm tắt các nội dung mà đồ án đã đạt được, đồng thời đánh giá ưu và nhược điểm của hệ thống cùng với định hướng giải pháp trong tương lai.

Nội dung

[PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP 2](#_Toc510456825)

[TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP 3](#_Toc510456826)

[CHƯƠNG I: ĐẶT VẤN ĐỀ 7](#_Toc510456827)

[**1.1** **Giới thiệu bài toán** 7](#_Toc510456828)

[**1.2** **Một số định hướng, giải pháp về công nghệ** 8](#_Toc510456829)

[**1.2.1** **Tổ chức hệ thống** 8](#_Toc510456830)

[**1.2.2** **Hệ thống cơ sở dữ liệu** 8](#_Toc510456831)

[**1.2.3** **Công nghệ bản đồ** 8](#_Toc510456832)

[**1.2.4** **Giải pháp về xử lý dữ liệu thời gian thực** 9](#_Toc510456833)

[**1.2.5** **Giải pháp xác thực người dùng** 9](#_Toc510456834)

[**1.2.6** **Giải pháp lưu trữ tập tin, hình ảnh** 10](#_Toc510456835)

[CHƯƠNG II: NGHIÊN CỨU CÁC GIẢI PHÁP CÔNG NGHỆ 11](#_Toc510456836)

[**2.1** **Các dịch vụ liên quan đến bản đồ- Google Maps API** 11](#_Toc510456837)

[**2.1.1 Tích hợp Google Map bằng Android API** 12](#_Toc510456838)

[**2.1.2** **Xác định vị trí với Google Play services location API** 13](#_Toc510456839)

[**2.1.3 Xác định đường đi với Google Directions API** 15](#_Toc510456840)

[**2.2 Xử lý dữ liệu theo thời gian thực với Firebase Realtime Database** 18](#_Toc510456841)

[**2.3 Xác thực người dùng với Firebase Authencations** 20](#_Toc510456842)

[**2.4 Lưu trữ dữ liệu với Firebase Storage** 21](#_Toc510456843)

[**2.5 Giải pháp thiết kế giao diện - Material Design** 23](#_Toc510456844)

[CHƯƠNG III: KHẢO SÁT VÀ XÁC ĐỊNH YÊU CẦU BÀI TOÁN 25](#_Toc510456845)

[3.1 Khảo sát một số hệ thống tương tự. 25](#_Toc510456846)

[3.2 Phân tích yêu cầu bài toán 28](#_Toc510456847)

[**3.2.1 Phân tích yêu cầu người dùng** 28](#_Toc510456848)

[**3.2.2** **Biểu đồ use case tổng quát** 29](#_Toc510456849)

[**3.2.3** **Biểu đồ usecase phân rã** 30](#_Toc510456850)

[**3.2.4** **Quy trình nghiệp vụ** 33](#_Toc510456851)

CHƯƠNG I: ĐẶT VẤN ĐỀ

* 1. **Giới thiệu bài toán**

Kinh tế ngày càng phát triến, chất lượng cuộc sống được cải thiện, nhu cầu đi vận chuyển hàng hóa lại ngày càng gia tăng. Chính vì thế các mô hình dịch vụ đi lại , vận chuyển như xe ôm, taxi, xe tải chở hàng theo yêu cầu,... ra đời, phục vụ nhu cầu của xã hội.

Tuy nhiên sau một thời gian dài phát triển, những mô hình kể trên không còn bắt kịp xu hướng phát triển, trở nên lỗi thời và phát sinh nhiều điểm tiêu cực. Trên đường phố Việt Nam hiện nay, không khó để bắt gặp hình ảnh những chiếc xe máy chở hàng cồng kềnh, gánh trên mình khối lượng hàng hóa quá quy định. Hay thực trạng xe taxi chặt chém khách hàng, đi đường vòng để tăng thêm cước. Người dân mua sắm hàng hóa nhưng phải loay hoay mất thời gian tìm đơn vị vận chuyển trong khi lực lượng xe tải chở hàng lại không tiếp cận được khách hàng vì phương thức liên lạc còn hạn chế.

Thêm nữa, trong vài năm gần đây với sự gia nhập của các mô hình mới như Uber, Grab, thị trường dịch vụ đi lại, vẩn chuyển biến động mạnh mẽ. Kèm theo đó là các chủ trương chính sách giảm thiểu các phương tiện cá nhân, giảm tải cho hệ thống giao thông của Nhà nước được đăng hằng ngày trên các phương tiện thông tin cho thấy nhu cầu cấp thiết về một hệ thống kết nối nhu cầu đi lại, vận chuyển giữa người dân và tài xế.

Cùng với đó, qua quá triình khảo sát trên thị trường, các ứng dụng vẫn chưa giải quyết được yêu cầu bài toan đặt ra, nếu có cũng còn thiếu một số chức năng, cung cấp các dịch vụ một cách toàn diện.

Chính vì vậy, yêu cầu đặt ra là xây dựng một hệ thống thông tin giúp kết nối người có nhu cầu đi lại, vận chuyển hàng hóa với các tài xế cung cấp dịch vụ. Dựa trên lợi thế về sự phổ biến của thiết bị điện thoại di dộng thông minh và các công nghệ tiên tiến có sẵn để tạo nên một giải pháp nhanh chóng và tiện lợi nhất.

* 1. **Một số định hướng, giải pháp về công nghệ**
     1. **Tổ chức hệ thống**

Hệ thống được xây dựng theo mô hình Client – Sever, trong đó:

* Client: là ứng dụng trên thiết bị chạy hệ điều hành Android, với đối tượng hướng đến là: tài xế, khách hàng và quản trị viên. Dựa trên vai trò và tính chất của từng đối tượng ứng dụng sẽ cung cấp chức năng tương ứng. Ứng dụng đáp ứng tiêu chí đơn giản, thao tác nhanh gọn, cung cấp thông tin đầy đủ, chính xác.
* Server: xây dựng dựa trên **ASP.NET Web API** và **Firebase.** Trong đó **ASP.NET Web API** là framework cung cấp các api giúp client có thể truy xuất thông tin còn **Firebase** với Realtime Database giúp trao đổi thông tin về vị trí liên tục giữa tài xế với khách hàng.
  + 1. **Hệ thống cơ sở dữ liệu**

Hệ thống cơ sở dữ liệu chủ yếu tập trung lưu trữ trên sever, sử dụng kiểu lưu trữ dữ liệu dạng SQL. Phía Client truy xuất CSDL thông qua các API được viết sẵn.

* + 1. **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 …

Hiện nay có rất nhiều nền tảng bản đồ được cung cấp, phát hành có thể kể đến như: Bing Maps (Microsoft), Here (Nokia), Foursquare (Foursquare) … Trong đó, 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, sinh viên đã quyết định sử dụng nền tảng này trong ứng dụng của mình. Cụ thể là nhóm giải pháp Google Maps API sau:

* ***Google Play services location API***: Công nghệ này giúp ta khai thác xây dựng các tính năng thông minh dựa trên vị trí.
* ***Google Maps API***: Cho phép nhúng bản đồ Google Maps vào ứng dụng. Các thông tin về vị trí, tuyến đường, các địa điểm,… sẽ được hiển thị trực tiếp lên bản đồ.
* ***Google Maps Directions API***: Có chức năng tính toán đường đi giữa các vị trí. Qua đó bạn có thể xem hướng dẫn đường đi cụ thể cho các phương thức di chuyển, bao gồm: lái xe, đi bộ hoặc đi xe đạp.
  + 1. **Giải pháp về xử lý dữ liệu thời gian thực**

Hệ thống không chỉ cho phép người dùng theo dõi vị trí trực tuyến mà còn hỗ trợ hệ thống nhắn tin trao đổi trực tiếp giữa người dùng và tài xế. 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.

***Firebase Realtime Database*** *là một hệ thống cơ sở dữ liệu được lưu trữ trên đám mây. Dữ liệu được lưu trữ dưới dạng JSON và được đồng bộ hóa theo thời gian thực tới mọi thiết bị kết nối với nó. Khi bạn xây dựng các ứng dụng nền tảng cho iOS, Android và* JavaScript SDKs*, tất cả khách hàng của bạn sẽ chia sẻ một phiên bản CSDL theo thời gian thực và tự động cập nhật những dữ liệu mới nhấ khi có sự thay đổi.*

* + 1. **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. Hiện nay việc xác thực người dùng thường thông qua 3 phương pháp phổ biến là: xác thực qua email, qua số điện thoại và Facebook. Để đáp ứng tính năng này, ứng dụng tích hợp giải pháp **Firebase Authentication** của Google.

*Hầu hết các ứng dụng cần biết danh tính của người dùng. Biết được danh tính của người dùng cho phép ứng dụng lưu trữ an toàn dữ liệu người dùng trên đám mây và cung cấp trải nghiệm cá nhân giống nhau trên tất cả các thiết bị của người dùng.*

*Firebase Authentication cung cấp các dịch vụ phụ trợ, SDK dễ sử dụng và các thư viện UI để xác thực người dùng đến ứng dụng của bạn. Nó hỗ trợ xác thực bằng cách sử dụng mật khẩu, số điện thoại, các nhà cung cấp nhận dạng liên kết phổ biến như Google, Facebook và Twitter và nhiều hơn nữa.*

* + 1. **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

***Cloud Storage*** *là dịch vụ lưu trữ đối tượng mạnh mẽ, đơn giản và tiết kiệm chi phí. Các Firebase SDK cho Cloud Storage thêm tính năng bảo mật của Google để tải lên và tải tệp cho các ứng dụng Firebase của bạn, bất kể chất lượng mạng. Bạn có thể sử dụng SDK của chúng tôi để lưu trữ hình ảnh, âm thanh, video hoặc các nội dung do người dùng tạo khác. Trên máy chủ, bạn có thể sử dụng*[*Google Cloud Storage*](https://cloud.google.com/storage)*, để truy cập các tệp giống nhau.*

Phần giới thiệu, phân tích chi tiết hơn về các giải pháp công nghệ sẽ được trình bày ở chương tiếp theo.

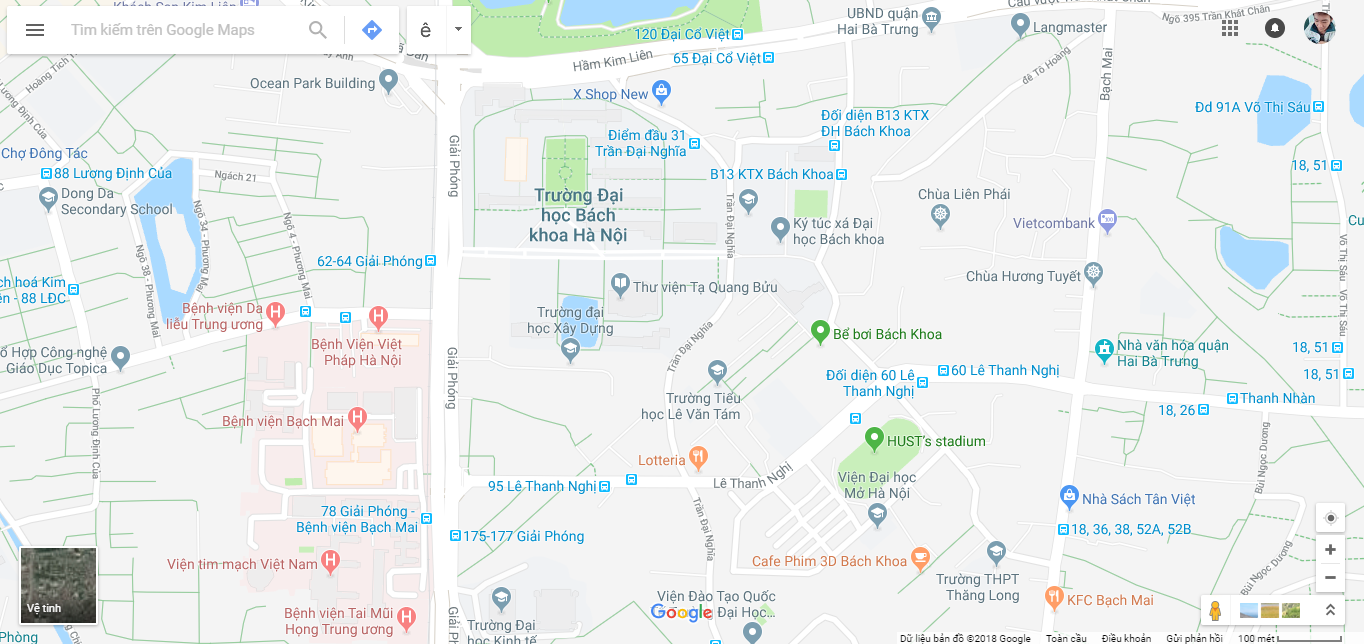
CHƯƠNG II: NGHIÊN CỨU CÁC GIẢI PHÁP CÔNG NGHỆ

Như đã trình bày tóm tắt trong phần phân tích một số định hướng giải pháp công nghệ ở trên, Chương này sẽ tập trung vào phân tích và giải thích rõ các giải pháp đó. 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***

## **2.1 Các dịch vụ liên quan đến bản đồ- Google Maps API**

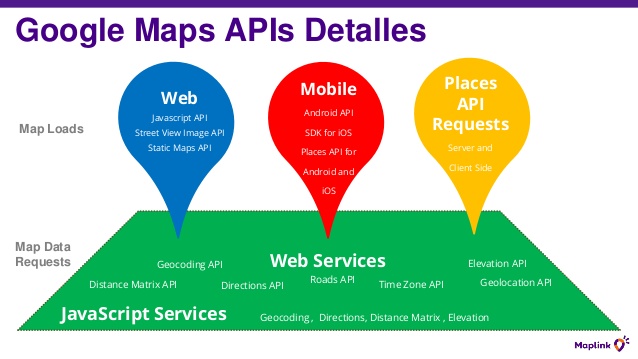
Google Map là một dịch vụ ứng dụng và công nghệ bản đồ trực tuyến miễn phí được cung cấp bởi Google. Nó cho phép thấy bản đồ đường sá, đường đi cho xe đạp, cho người đi bộ và xe hơi, và những địa điểm kinh doanh trong khu vực cũng như khắp nơi trên thế giới. Với nhiều tính năng vượt trội, Google Map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



**Hình 1: Bản đồ khu vực trường đại học Bách Khoa Hà Nội chụp từ Google Map**

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 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...).

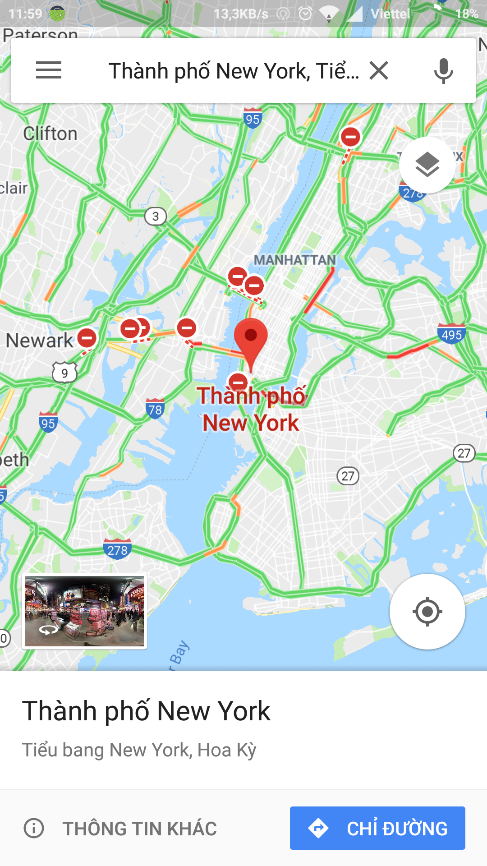


**Hình 2: Các thành phần của Google Map API [ Nguồn: Maplink ]**

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 .

### **2.1.1 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 3: Giao diện Google Map trên một thiết bị Android**

*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.*

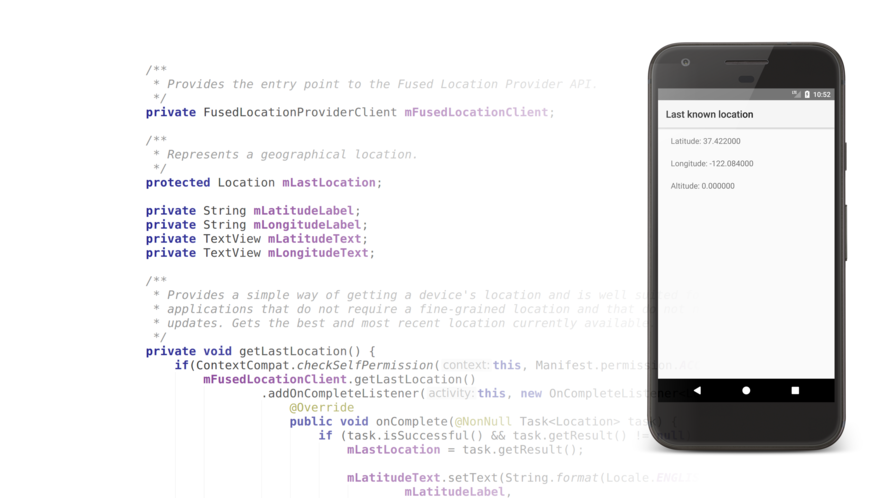
### **2.1.2 Xác định vị trí với Google Play services location API**

**Google Play services location API** 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:

* **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.
* **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.
* **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.



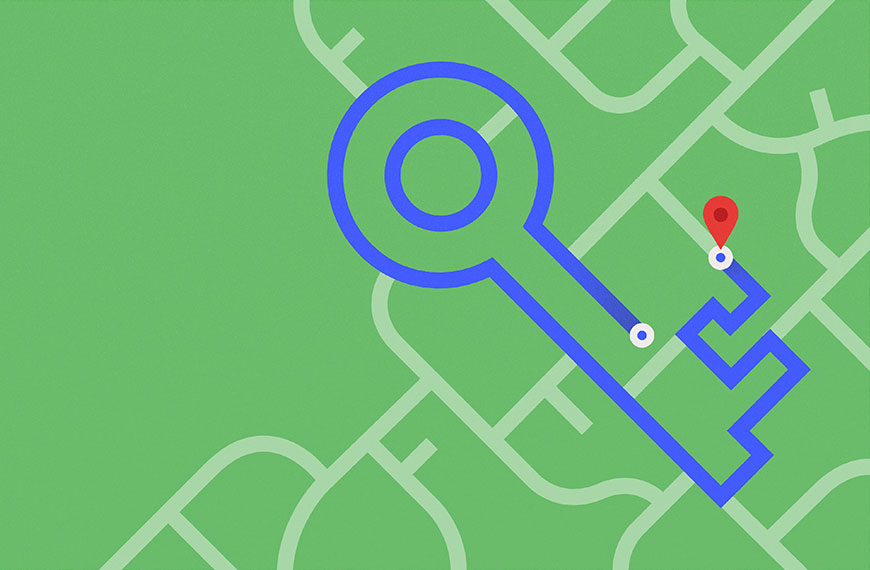
**Hình 4: Dễ dàng tích hợp Google play service location API vào ứng dụng**

**[nguồn Google Developers]**

### **2.1.3 Xác định đường đi với Google Directions API**

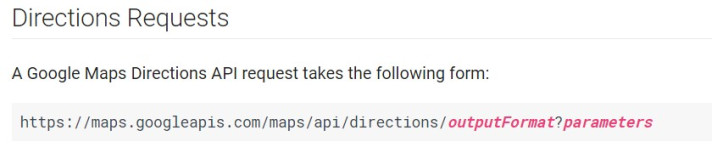
Việc xác định đường đi từ đó xác định được khoảng cách, thời gian di chuyển trung bình là một trong những tính năng quan trọng của ứng dụng. Chi phí phát sinh cơ bản của một hợp đồng đi lại, vận chuyển trên ứng dụng là kết quả tính toán từ chức năng này.

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.



**Hình 5: Directions API giúp xác định tuyến đường giữa 2 điểm trên bản đồ**

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.



**Hình 6: Cấu trúc 1 Request**

Ở đây chúng ta chú ý đến **outputFormat** sẽ là kiểu dữ liệu mà ta muốn nhận về từ respone. 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:

* **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.
* **Destination**: mục này là địa điểm mà ta chọn làm địa điểm đích đến, và cấu trúc cũng tương tự như **Origin.**
* **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:

* **Mode**: chọn loại phương tiện mà ta muốn di chuyển
* **Alternatives**: có thể trả về nhiều hơn 1 tuyến đường khả dụng.
* **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.
* … (tham khảo thêm trên trang document của Directions API).

Sau khi gửi 1 **Request** lên, một chuỗi **Response** 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.

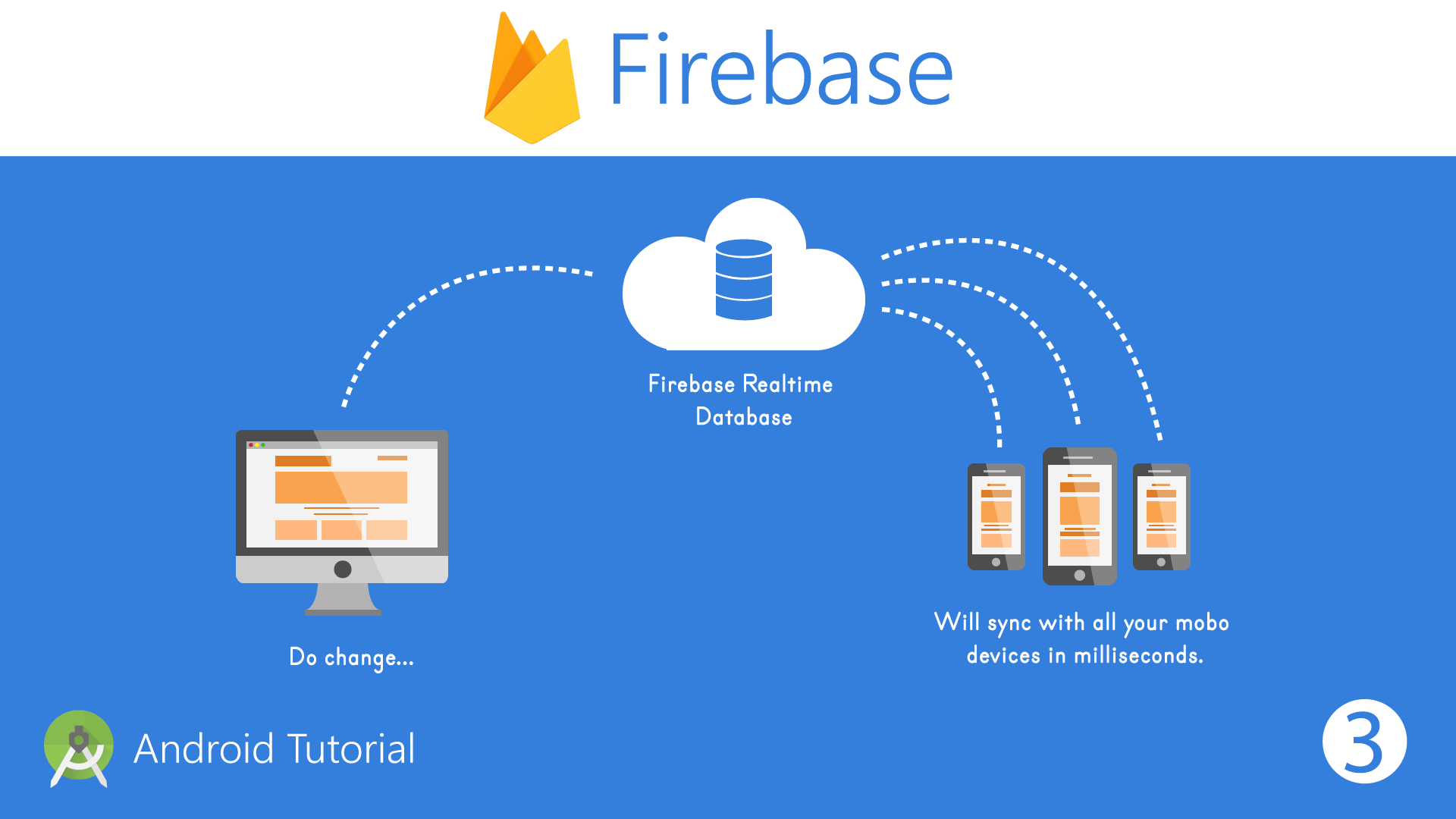
|  |
| --- |
| "status": "OK",   "geocoded\_waypoints" : [      { "geocoder\_status" : "OK",         "place\_id" : "ChIJ7cv00DwsDogRAMDACa2m4K8",         "types" : [ "locality", "political" ]},      { "geocoder\_status" : "OK",         "place\_id" : "ChIJ69Pk6jdlyIcRDqM1KDY3Fpg",         "types" : [ "locality", "political" ]},      { "geocoder\_status" : "OK",         "place\_id" : "ChIJgdL4flSKrYcRnTpP0XQSojM",         "types" : [ "locality", "political" ]},      { "geocoder\_status" : "OK",         "place\_id" : "ChIJE9on3F3HwoAR9AhGJW\_fL-I",         "types" : [ "locality", "political" ]}],   "routes": [ {     "summary": "I-40 W",     "legs": [ {       "steps": [ {         "travel\_mode": "DRIVING",         "start\_location": {           "lat": 41.8507300,           "lng": -87.6512600},         "end\_location": {           "lat": 41.8525800,           "lng": -87.6514100},         "polyline": {           "points": "a~l~Fjk~uOwHJy@P"},         "duration": {           "value": 19,           "text": "1 min"},         "html\_instructions": "Head \u003cb\u003enorth\u003c/b\u003e on \u003cb\u003eS Morgan St\u003c/b\u003e toward \u003cb\u003eW Cermak Rd\u003c/b\u003e",         "distance": {           "value": 207,           "text": "0.1 mi"         }       },       ... |

**Bảng 1: Cấu trúc một chuỗi Responses dưới dạng JSON**

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

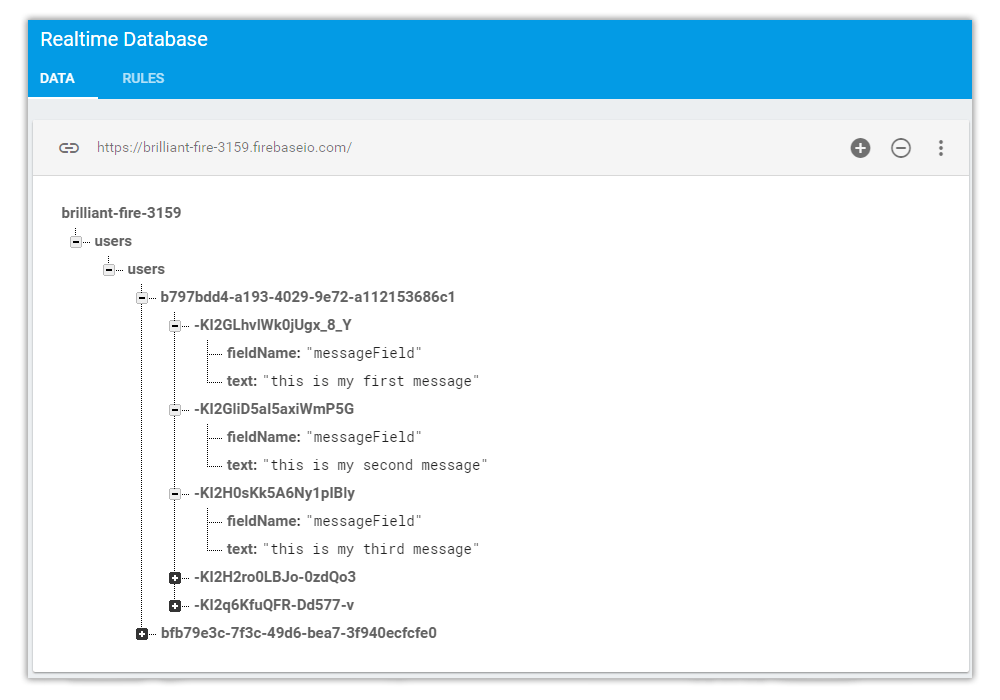
## **2.2 Xử lý dữ liệu theo thời gian thực với Firebase Realtime Database**

**Firebase Realtime Database** 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 7: Mô hình hoạt động của Firebase Realtime Database [Nguồn Android Tutorial]**

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 (trees), với mỗi nhánh giống như một container, chỉ chứa hoặc là dữ liệu ứng với nhánh đó (tức là value tương ứng với key), hoặc một tập hợp các nhánh con cũng được tổ chức theo một cách tương tự.



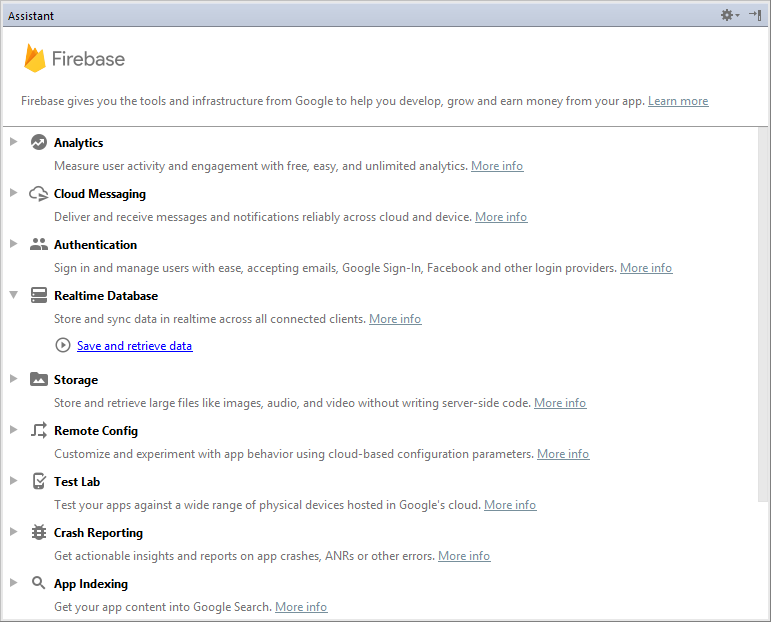
**Hình 8: Ví dụ về tổ chức cây dữ liệu trên Firebase Realtime Database**

Mỗi nhánh được đại diện bởi một **key** và **value** tương ứng. Các **key** luôn có dạng là String. Còn các **value** 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:

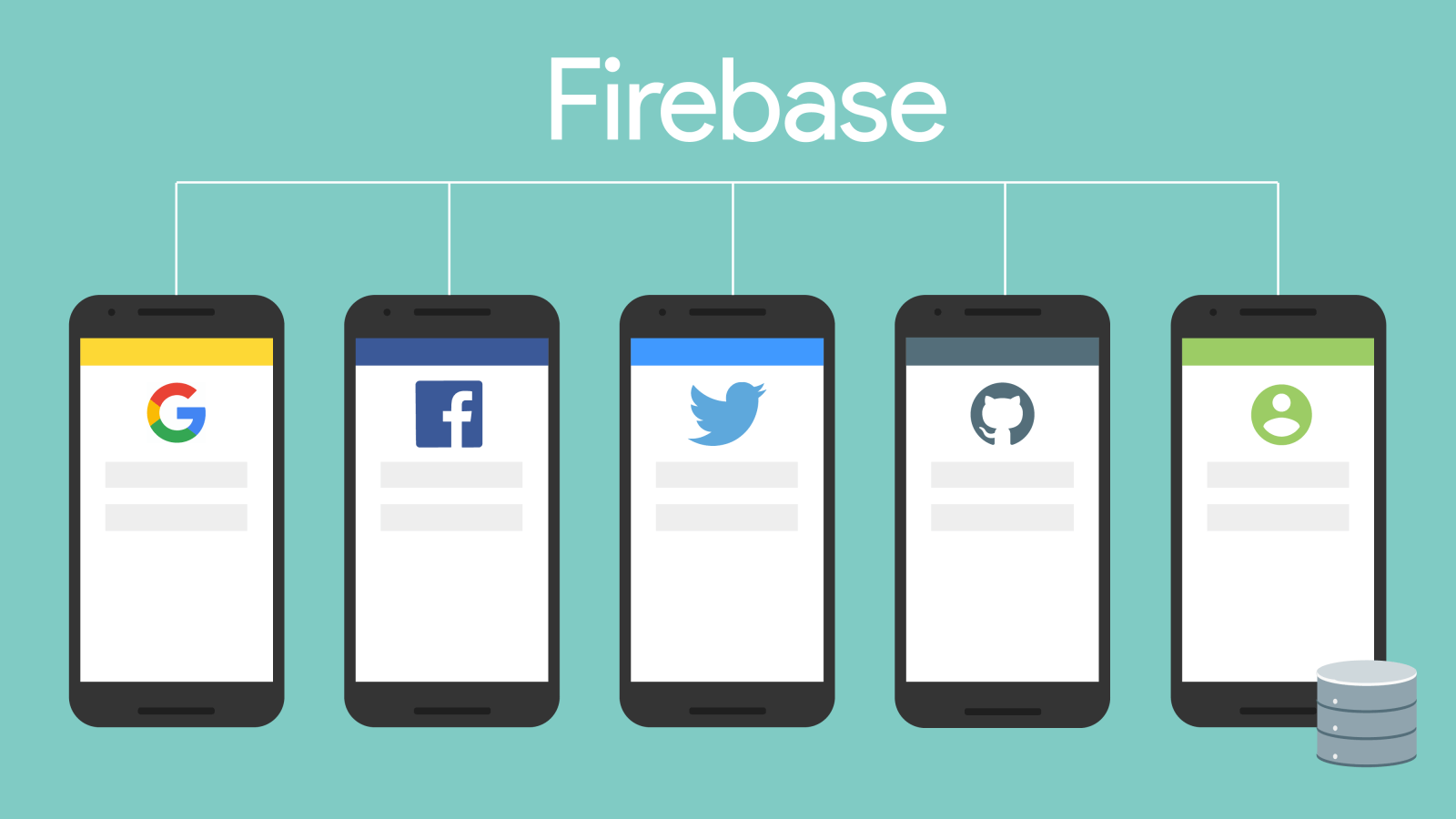
* Key chưa tồn tại: Firebase Database sẽ tạo một Object mới tại vị trí mà bạn muốn lưu, với **key – value** là các giá trị bạn đã định nghĩa. Event này được gọi là “add”, tức là thêm.
* Key đã tồn tại: Firebase Database sẽ sửa value tại vị trí của key đó theo value mới. Event này được gọi là “chànge”.

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.

**Hình 9: Tích hợp Realtime Database vào ứng dụng thông qua Assistant trên Android Studio**

## **2.3 Xác thực người dùng với Firebase Authencations**

Firebase Authencations 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 10: Firebase Authencations cung cấp đa dạng về giải pháp xác thực người dùng [nguồn Firebase - Google]**

*Firebase Authentication hoạt động như thế nào?*

* 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 liên đới (Facebook, Twitter,…)
* Sau đó, bạn chuyển những chứng chỉ này tới SDK Authentication Firebase. Các dịch vụ phụ trợ của Firebase Authentication sẽ xác minh các thông tin và trả lời cho khách hàng.
* 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.

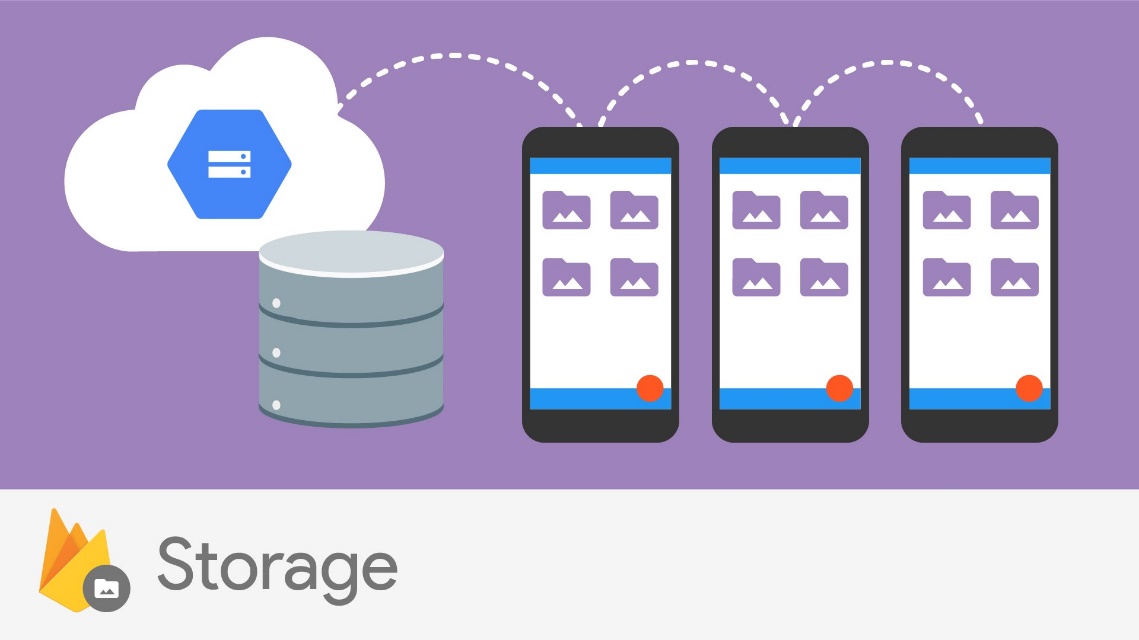
## **2.4 Lưu trữ dữ liệu với Firebase Storage**

Firebase Storage 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:

* **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.
* **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.
* **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.

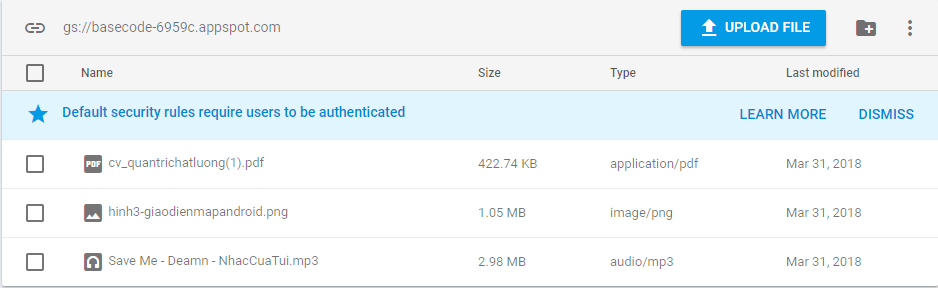


**Hình 11: Firebase Storage với nền tảng Cloud storage của Google**

*Firebase Storage hoạt động như thế nào?*

* 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.
* 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 sever như lọc hình ảnh hoặc chuyển mã video thông qua Google Cloud Platform.
* 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 12: Giao diện quản lý file của Firebase storage trên website quản lý Firebase Console**

## **2.5 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" 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.

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



**Hình 13: Mô típ thẻ là đặc trưng nổi bật của thiết kế Material Design [nguồn thenextweb]**

Đặc điểm của Material Design

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

*Lợi ích của Material Design*

Material Design là một ngôn ngữ được Google tạo ra nhằm “phát triển một hệ thống nền duy nhất cho phép xây dựng các trải nghiệm đồng bộ giữa nhiều loại và kích thước thiết bị khác nhau. Việc thiết kế cho thiết bị di động được đặt lên hàng đầu, nó còn phải chú trọng đến việc nhập liệu bằng cảm ứng, giọng nói, chuột, bàn phím”. [6]

Google muốn mang lại trải nghiệm thống nhất cho người dùng trong hệ sinh thái của hãng. Người dùng cuối như chúng ta sẽ có cảm giác quen thuộc hơn, dành ít thời gian để học hỏi hơn khi sử dụng cùng một app nhưng trên nhiều máy khác nhau. Các lập trình viên thì có thể đảm bảo rằng cách người dùng trải nghiệm app của họ là như nhau không quan trọng thiết bị đang chạy là gì.

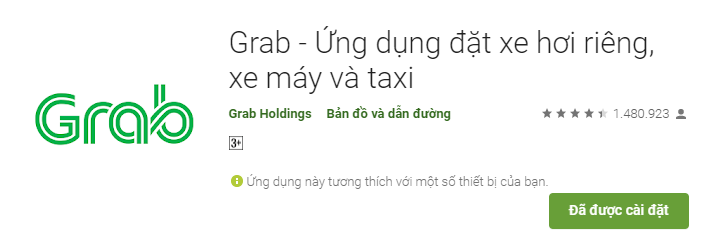
CHƯƠNG III: KHẢO SÁT VÀ XÁC ĐỊNH YÊU CẦU BÀI TOÁN

3.1 Khảo sát một số hệ thống tương tự.

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 đi lại, vận chuyển với tài xế cung cấp các dịch vụ này.

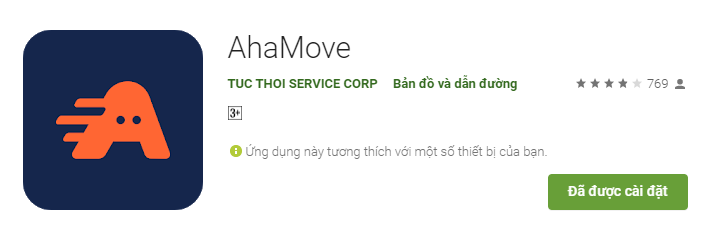
Hiện nay trên thị trường có một số hệ thống, ứng dụng có tính năng tương tự với yêu cầu bài toán, có thể kể đến như sau:

**Grab:** là ứng dụng gọi xe theo yêu cầu của công ty Grab. Ứng dụng cung cấp dịch vụ đa dạng dựa theo nhu cầu của khách hàng.



**Hình 14: Ứng dụng Grab trên Play store**

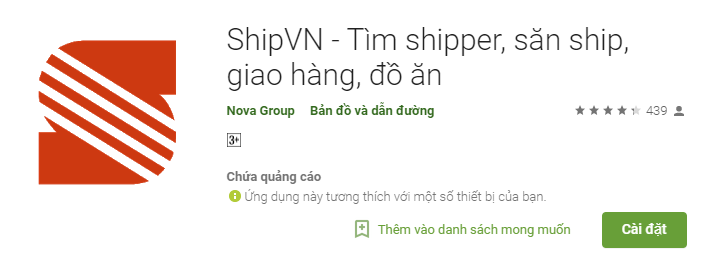
**AhaMove**: AhaMove là ứng dụng đặt xe cho nhu cầu giao hàng. Kết nối khách hàng với đội ngũ tài xế xe giàu kinh nghiêm. Khi bạn cần chuyển các đơn hàng gấp hoặc cần lên lịch chuyển hàng chủ động, AhaMove sẽ giúp bạn đặt xe một cách hiệu quả.



**Hình 15: Ứng dụng AhaMove trên Play Store**

**ShipVN:** ShipVN là hệ thống dịch vụ phần mềm thông minh 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.

Khi shop sử dụng ShipVN đăng tin tìm shipper thông tin sẽ được thông báo và hiển thị cho hàng nghìn shipper gần đó, ngoài ra tin tìm shipper của shop cũng sẽ được tự động đăng trên rất nhiều groups facebook nơi cho hàng triệu shipper đang theo dõi, shop tìm được một shipper ưng ý chỉ trong vài phút**.**

****

**Hình 16 Ứng dụng ShipVN trên Play Store**

Các ứng dụng kể trên cung cấp các chức năng khá đa dạng:

|  |  |  |
| --- | --- | --- |
| STT | Ứng dụng | Tính năng |
| 1 | Grab | Đăng ký, đăng nhập tài khoản  Gọi xe quanh một khu vực theo yêu cầu.  Gọi dịch vụ chuyển hàng trong một khu vực.  Đặt lịch gọi xe tự động.  Chỉ đường theo bản đồ.  Tính giá tự động theo quãng đường.  Theo dõi trực tuyến quá trình di chuyển từ lúc gọi xe đến lúc xuống xe.  Quản lý lịch sử di chuyển  Thanh toán trực tiếp, qua thẻ hoặc ví điện tử |
| 2 | AhaMove | Đăng ký, đăng nhập tài khoản  Gọi shipper trong một khu vực theo yêu cầu  Chuyển hàng đến nhiều địa chỉ trong cùng một hành trình.  Chỉ đường theo bản đồ.  Theo dõi trạng thái chuyển hàng  Nhận thu hộ  Tính giá tự động theo quãng đường. |
| 3 | ShipVN | Đăng ký, đăng nhập tài khoản  Gọi shipper trong một khu vực theo yêu cầu  Chuyển hàng đến nhiều địa chỉ trong cùng một hành trình.  Xem bản đồ mật độ đơn hàng  Chỉ đường bằng bản đồ |

**Bảng 2: Bảng tính năng của các ứng dụng được khảo sát**

Với các ứng dụng được khảo sát ở trên, hầu hết đều đủ các tính năng cơ bản. Tuy nhiên chúng vẫn còn có một vài nhược điểm, tính năng cần cải tiến như phân loại hàng hóa vận chuyển theo loại xe, hệ thống đánh giá tài xế, khách hàng, …

Qua khảo sát nói trên, hệ thống mà sinh viên mong muốn xây dựng một hệ thống với chức năng đầy đủ hơn. Các tính năng được kế thừa, hoàn thiện từ các ưu điểm của các hệ thống trên, đồng thời bổ sung thêm các chức năng mới mà các ứng dụng trên thị trường chưa có.

3.2 Phân tích yêu cầu bài toán

### **3.2.1 Phân tích yêu cầu người dùng**

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 sau:

* **Khách hàng**: là đối tượng có nhu cầu về dịch vụ đi lại, 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.
* **Tài xế**: là đối tượng cung cấp dịch vụ đi lại, 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
* **Quản trị viên:** Quản trị viên sẽ có thể quản trị 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.

### **3.2.2 Biểu đồ use case tổng quát**

Hệ thống gồm có 3 tác nhân chính: **khách hàng, tài xế và quản trị viên.**

|  |  |  |
| --- | --- | --- |
| **STT** | **Tác nhân** | **Các use case** |
| 1 | Khách hàng | * Đăng ký tài khoản * Đăng nhập hệ thống * Quản lý thông tin cá nhân * Gọi xe theo yêu cầu * Quản lý lịch sử gọi xe * Quản lý danh sách tài xế ưa thích |
| 2 | Tài xế | * Đăng ký tài khoản * Đăng nhập hệ thống * Quản lý thông tin cá nhân * Quản lý thông tin phương tiện * Nhận yêu cầu gọi xe * Quản lý lịch sử di chuyển |
| 3 | Quản trị viên | * Quản trị người dùng * Quản trị dịch vụ |

**Bảng 3: Bảng các usecase tổng quát cho tưng loại người dùng**



**Hình 17: Biểu đồ usecase tổng quát**

### **3.2.3 Biểu đồ usecase phân rã**

* Biểu đồ usecase phân rã cho 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.



* 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.

****

* 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.

****

* 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.

****

* 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.

****

### **3.2.4 Quy trình nghiệp vụ**

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

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

Cụ thể chi tiết các quy trình sẽ được phân tích như sau:

* 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 đi lại, vận chuyển, quản lý lịch sử, quản lý các thông tin cá nhân,…

* 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**



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 cũng như xem thống kê về người dùng, dịch vụ.