# **MỤC LỤC**

[**MỤC LỤC** a](#_Toc504675174)

[**TÓM TẮT** d](#_Toc504675175)

[**DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT** e](#_Toc504675176)

[**DANH MỤC HÌNH ẢNH, BẢNG BIỂU** f](#_Toc504675177)

[**I-** **Giới thiệu đề tài.** 1](#_Toc504675178)

[**1.** **Đặt vấn đề.** 1](#_Toc504675179)

[**2.** **Mục tiêu đề tài.** 2](#_Toc504675180)

[**3.** **Phạm vi đề tài.** 2](#_Toc504675181)

[**4.** **Nội dung nghiên cứu.** 3](#_Toc504675182)

[**5.** **Cấu trúc báo cáo.** 4](#_Toc504675183)

[**II-** **Công nghệ ứng dụng, giải pháp, thuật toán, hiện thực hệ thống.** 5](#_Toc504675184)

[**1.** **Công nghệ ứng dụng** 5](#_Toc504675185)

[***1.1*** ***Tổng quan về GPS.*** 5](#_Toc504675186)

[**1.1.1** **Giới thiệu về GPS.** 5](#_Toc504675187)

[**1.1.2** **Đánh giá về hệ thống đinh vị GPS.** 5](#_Toc504675188)

[***1.2*** ***Tổng quan về hệ điều hành Android.*** 5](#_Toc504675189)

[**1.2.1** **Giới thiệu về hệ điều hành Android.** 5](#_Toc504675190)

[**1.2.2** **Thị phần sử dụng các phiên bản của hệ điều hành Android** 6](#_Toc504675191)

[**1.2.3** **Đánh giá về hệ điều hành Android.** 8](#_Toc504675192)

[***1.3*** ***Tổng quan về Google Maps.*** 8](#_Toc504675193)

[**1.3.1** **Giới thiệu về Google Maps.** 8](#_Toc504675194)

[**1.3.2** **Một số tính năng của Google Maps.** 9](#_Toc504675195)

[**1.3.3** **Chính sách miễn phí của Google Maps đối với người sử dụng API.** 11](#_Toc504675196)

[**1.3.4** **Đánh giá về Google Maps.** 12](#_Toc504675197)

[***1.4*** ***Tổng quan về Google Maps Roads API.*** 12](#_Toc504675198)

[**1.4.1** **Giới thiệu về Google Maps Roads API.** 12](#_Toc504675199)

[**1.4.2** **Đánh giá về Google Maps Roads API.** 13](#_Toc504675200)

[***1.5*** ***Tổng quan về FireBase.*** 13](#_Toc504675201)

[**1.5.1** **Giới thiệu về FireBase.** 13](#_Toc504675202)

[**1.5.2** **Các tính năng chính của Firebase** 13](#_Toc504675203)

[**1.5.3** **Đánh giá về FireBase.** 14](#_Toc504675204)

[***1.6*** ***Tổng quan về RSS và Web Crawler.*** 15](#_Toc504675205)

[**1.6.1** **Giới thiệu về RSS.** 15](#_Toc504675206)

[**1.6.2** **Giới thiệu về Web Crawler.** 15](#_Toc504675207)

[**2.** **Giải Pháp.** 16](#_Toc504675208)

[***2.1*** ***Mô tả kiến trúc phần mềm.*** 16](#_Toc504675209)

[***2.2*** ***Website.*** 18](#_Toc504675210)

[***2.2.1*** ***Sơ đồ chức năng (Use Case Diagram)*** 18](#_Toc504675211)

[***2.2.2*** ***Use Case Scenario*** 19](#_Toc504675212)

[***2.3*** ***Ứng dụng di động.*** 21](#_Toc504675213)

[***2.3.1*** ***Sơ đồ chức năng*** 21](#_Toc504675214)

[***2.3.2*** ***Use Case Scenario*** 22](#_Toc504675215)

[***2.4*** ***Cấu trúc cơ sở dữ liệu.*** 25](#_Toc504675216)

[***2.4.1*** ***Tổng quan về cấu trúc cơ sở dữ liệu của hệ thống.*** 25](#_Toc504675217)

[***2.4.2*** ***Mô hình cấu trúc cơ sở dữ liệu.*** 25](#_Toc504675218)

[***2.4.3*** ***Chức năng các Node.*** 26](#_Toc504675219)

[**3.** **Hiện thực hệ thống.** 32](#_Toc504675220)

[***3.1*** ***Website.*** 32](#_Toc504675221)

[***3.1.1*** ***WorkFlow.*** 32](#_Toc504675222)

[***3.1.2*** ***Sơ đồ kiến trúc.*** 33](#_Toc504675223)

[***3.1.3*** ***Giao diện.*** 35](#_Toc504675224)

[***3.2*** ***Ứng dụng di động.*** 41](#_Toc504675225)

[***3.2.1*** ***WorkFlow.*** 41](#_Toc504675226)

[***3.2.2*** ***Sơ đồ kiến trúc*** 42](#_Toc504675227)

[***3.2.3*** ***Giao diện*** 44](#_Toc504675228)

[**III-** **Kết quả.** 48](#_Toc504675229)

[**1.** **Kiểm thử hệ thống.** 48](#_Toc504675230)

[***1.1*** ***Tiêu chí kiểm thử.*** 48](#_Toc504675231)

[***1.2*** ***Kế hoạch kiểm thử.*** 48](#_Toc504675232)

[***1.2.1*** ***Kiểm thử các khối chức năng chính của hệ thống.*** 48](#_Toc504675233)

[***1.2.2*** ***Kiểm thử về hiệu năng của hệ thống (độ hao pin của thiết bị).*** 48](#_Toc504675234)

[***1.2.3*** ***Kiểm thử lộ trình.*** 49](#_Toc504675235)

[***1.2.4*** ***Kiểm thử về dịch bệnh và thiên tai.*** 49](#_Toc504675236)

[***1.3*** ***Quá trình kiểm thử.*** 49](#_Toc504675237)

[***1.3.1*** ***Kiểm thử các khối chức năng của hệ thống.*** 49](#_Toc504675238)

[***1.3.1.1*** ***Admin website*** 49](#_Toc504675239)

[***1.3.1.2*** ***Android App*** 50](#_Toc504675240)

[***1.3.2*** ***Kiểm thử về hiệu năng của hệ thống (độ hao pin của thiết bị).*** 52](#_Toc504675241)

[***1.3.3*** ***Kiểm thử về lộ trình.*** 52](#_Toc504675242)

[***1.3.4*** ***Kiểm thử về thông báo dịch bệnh, thiên tai.*** 54](#_Toc504675243)

[***1.3.4.2*** ***Thông báo thiên tai.*** 54](#_Toc504675244)

[**2.** **Đánh giá hệ thống.** 54](#_Toc504675245)

[***2.1*** ***Ưu điểm.*** 54](#_Toc504675246)

[***2.2*** ***Nhược điểm.*** 55](#_Toc504675247)

[**IV-** **Kết luận.** 56](#_Toc504675248)

[**1.** **Kết luận.** 56](#_Toc504675249)

[**2.** **Hướng phát triển.** 56](#_Toc504675250)

[**TÀI LIỆU THAM KHẢO** 58](#_Toc504675251)

# **TÓM TẮT**

Đề tài thực tập của nhóm chúng tôi là xây dựng ứng dụng mạng là trò chơi “Kéo búa bao”. Ứng dụng này dựa trên mô hình “client – server” truyền tải thông tin dựa trên giao thức TPC/IP cùng với các công nghệ Java Socket, Thread, JSON và một protocol được tạo riêng để quản lý các thông điệp qua lại giữa hai phía. Trong quá trình thực hiện đề tài thực tập này, nhóm chúng tôi đã thực hiện thông qua hai giai đoạn cụ thể sau:

Giai đoạn đầu: Tiến hành nghiên cứu cơ sở lý thuyết và các công nghệ để phân tích, đánh giá và đề ra những chức năng cần được phát triển cũng như phương pháp thực hiện.

Giai đoạn sau: Tiến hành viết ứng dụng hoàn chỉnh dựa trên những chức năng đã đề ra. Theo đó, Những chức năng được phát triển bao gồm:

+ Chức năng đăng ký, đăng nhập, đăng xuất.

+ Chức năng thách đấu và tham gia trận đấu.

Từ khi ra đời cho đến nay, mạng máy tính đã và đang là một phần không thể thiếu cho xã hội hiện đại. Những ứng dụng của mạng máy tính vô cùng đa dạng và được áp dụng trên rất nhiều lĩnh vực của đời sống, là một trong những nhân tố bản lề, làm bàn đạp cho sự phát triển của xã hội.

Với mục đích có được hiểu biết nền tảng về mạng máy tính, trong bài thực tập tốt nghiệp này, chúng tôi sẽ tiến hành xây dựng một ứng dụng mạng đơn giản là trò chơi “Kéo Búa Bao dựa trên mô hình client - server”, thông qua đó có được một cái nhìn rõ hơn về mạng máy tính và cách một ứng dụng mạng hoạt động.

Nội dung bài báo cáo này gồm 5 phần:

* 1. Giới thiệu đề tài: Nội dung phần này sẽ tập trung giới thiệu tổng quan về đề tài, làm rõ mục tiêu, phạm vi đề tài và giới thiệu cấu trúc báo cáo.
* 2. Công nghệ ứng dụng, giải pháp, thuật toán, hiện thực hệ thống: Phần này là thân bài chính của bài báo cáo. Nội dung sẽ tập trung trình bày những kiến thức cần thiết cho quá trình thực hiện đề tài gồm những kiến thức tổng quan về Socket, port, Java Socket library, Thread, JSON. Bên cạnh đó đưa ra những phân tích cũng như cách thức hiện thực ứng dụng.
* 3 Kết quả: Nội dung của phần này là đưa ra kết quả cuối cùng của quá trình hiện thực. Hiển thị cách thức hoạt động, luồng chạy cơ bản của ứng dụng.
* 4. Kết luận: Nội dung của phần này là đánh giá quá trình hiện thực đề tài, nhìn nhận những mặt hạn chế và đưa ra những phương hướng phát triển, mở rộng đề tài và đưa vào thực tế.

# **DANH MỤC CÁC KÝ HIỆU, CÁC CHỮ VIẾT TẮT**

Admin: Người quản trị hệ thống

API: Application Programming Interface

CDN: Content Delivery Network

GPS: Global Positioning System

HAL: Hardware Abstraction Layer

JSON: JavaScript Object Notation

km: Kilometer

RSS: Really Simple Syndication

SSL: Secure Sockets Layer

TV: Television

UID: Unique Identification

User: Nhân viên, người dùng

OS: Operating system: Hệ điều hành

XML: eXtensible Markup Language, tức "Ngôn ngữ đánh dấu mở rộng”

HTML: HyperText Markup Language

XHTML: Extensible HyperText Markup Language

# **DANH MỤC HÌNH ẢNH, BẢNG BIỂU**

[Hình 1. 1 Nội dung nghiên cứu của đề tài 3](#_Toc502184824)

----------------------------------------------------------------------------------------------------------------

[Hình 2. 1: Giới thiệu về NHK News 5](#_Toc502080823)

[Hình 2. 2: Giao diện ứng dụng NHK News 7](#_Toc502080824)

[Hình 2. 3: Giới thiệu ứng dụng MAPS.ME 9](#_Toc502080825)

-----------------------------------------------------------------------------------------------------------------

[Hình 3.1 2: Biểu đồ thị phần sử dụng các phiên bản Android (tính đến ngày 5 tháng 6 năm 2017 theo Google) [3] 13](#_Toc502184014)

[Hình 3.1 6: Một số tính năng của Google Maps 16](#_Toc502184015)

----------------------------------------------------------------------------------------------------------------

[Hình 3.2 1: Cấu trúc hệ thống phần mềm 23](#_Toc502184002)

[Hình 3.2 2: Sơ đồ chức năng dành cho website của người quản lý. 24](#_Toc502184003)

[Hình 3.2 3: Sơ đồ chức năng của ứng dụng di động dành cho nhân viên. 27](#_Toc502184004)

[Hình 3.2 4: Tổng quan về cấu trúc cơ sở dữ liệu. 30](#_Toc502184005)

[Hình 3.2 5: Mô hình cấu trúc cơ sở dữ liệu 31](#_Toc502184006)

[Hình 3.2 6: Cấu trúc cơ sở dữ liệu của “disasterAndDisease”. 31](#_Toc502184007)

[Hình 3.2 7: Cấu trúc cơ sở dũ liệu của “disaster”. 32](#_Toc502184008)

[Hình 3.2 8: Cấu trúc cơ sỡ dữ liệu của các node con. 32](#_Toc502184009)

[Hình 3.2 9: Cấu trúc cơ sở dữ liệu của “users” 33](#_Toc502184010)

[Hình 3.2 10: Cấu trúc cơ sở dữ liệu của “users\_location”. 34](#_Toc502184011)

[Hình 3.2 11: Toàn bộ cấu trúc cơ sở dữ liệu của “users\_location”. 35](#_Toc502184012)

[Hình 3.2 12: Cấu trúc cơ sở dữ liệu của “vehicles” 36](#_Toc502184013)

----------------------------------------------------------------------------------------------------------------

[Hình 3.3 1: WorkFlow website người quản trị hệ thống 37](#_Toc502184197)

[Hình 3.3 2: Kiến trúc hệ thống website quản lý. 38](#_Toc502184198)

[Hình 3.3 3: Giao diện đăng nhập của người quản trị hệ thống. 40](#_Toc502184199)

[Hình 3.3 4: Trang chủ website hệ thống 41](#_Toc502184200)

[Hình 3.3 5: Trang chủ của website hệ thống – phần thông báo thiên tai 41](#_Toc502184201)

[Hình 3.3 6: Trang chủ website của hệ thống - phần xem danh sách nhân viên. 42](#_Toc502184202)

[Hình 3.3 7: Bản đồ di chuyển trong ngày của một nhân viên cụ thể. 42](#_Toc502184203)

[Hình 3.3 8: Trang thêm thành viên. 43](#_Toc502184204)

[Hình 3.3 9: Trang cập nhật công thức tính toán chi phí. 43](#_Toc502184205)

[Hình 3.3 10: Trang lựa chọn đối tượng xem thống kê. 44](#_Toc502184206)

[Hình 3.3 11: Biểu đồ cột của tất cả nhân viên từ 01/12/2017 – 25/12/2017. 44](#_Toc502184207)

[Hình 3.3 12: Biểu đồ tròn của tất cả nhân viên từ 01/12/2017 – 25/12/2017. 45](#_Toc502184208)

[Hình 3.3 13: WorkFlow của ứng dụng di động. 46](#_Toc502184209)

[Hình 3.3 14: Sơ đồ kiến trúc của ứng dụng di động. 47](#_Toc502184210)

[Hình 3.3 15: Giao diện ứng dụng di động từ “log in” đến màn hình chính. 50](#_Toc502184211)

[Hình 3.3 16: Xem cụ thể một thiên tai (Bão TEMBIN). 51](#_Toc502184212)

[Hình 3.3 17: Giao diện chuyển tiếp giữa các chức năng. 52](#_Toc502184213)

----------------------------------------------------------------------------------------------------------------

[Bảng 3.1 1: Thị phần sử dụng các phiên bản Android. 12](#_Toc502080846)

[Bảng 3.1 2: Chính sách của Google Maps 17](#_Toc502080847)

1. **Giới thiệu đề tài.**
2. **Đặt vấn đề.**

Sự phát triển mạnh mẽ của công nghệ cho ra đời hàng loạt những thiết bị thông minh (Smartphone) và đang được sử dụng một cách phổ biến và rộng rãi trên toàn thế giới để phục vụ các lĩnh vực đời sống với những chức năng linh hoạt với độ chính xác cao. Chiếc điện thoại đã không còn giới hạn trong chức năng đàm thoại thông thường mà nó đã trở thành một công cụ hỗ trợ thực sự cần thiết cho người sử dụng. Với khả năng định vị với độ chính xác cao thông qua hệ thống định vị GPS nên các dịch vụ sử dụng vị trí ngày càng phát triển mạnh và rộng rãi hơn để phục vụ nhu cầu của con người như đi lại, kiểm soát thiết bị cũng như tìm kiếm địa điểm, đường đi một cách thuận tiện nhất.

Ngày nay, sự phát triển về kinh tế hình thành nhiều công ty đa quốc gia tạo nên những đội ngũ nhân viên làm việc từ xa. Để đảm bảo những mục tiêu kinh doanh của công ty thì cần kiểm soát đội ngũ nhân viên một cách hợp lý và hiệu quả về chi phí ăn uống, đi lại. Hiện tại, việc quản lý nhân viên là rất quan trọng đặc biệt là những nhân viên bán hàng phải di chuyển liên tục để gặp khách hàng thường xuyên. Các giải pháp công nghệ ngày nay sẽ giúp giải quyết vấn đề này trở nên dễ dàng hơn.

Việc quản lý một đội ngũ nhân viên làm việc từ xa đặc biệt là nhân viên bán hàng một cách hiệu quả nhất là rất quan trọng đối với vấn đề kinh doanh của công ty. Các chính sách hỗ trợ nhân viên về chi phí đi lại là một phần hầu như không thể thiếu. Hiện tại, việc quản lý này thường thông qua các số liệu do nhân viên cung cấp lại (vé xe, hóa đơn, nhân viên tự khai báo,…) mà chưa được quản lý một cách tự động.

Đề tài này được phát triển để hỗ trợ công ty quản lý vị trí và chi phí đi lại của nhân viên trong các hoạt động hàng ngày. Bên cạnh đó, do công việc nên những nhân viên phải làm việc trong nhiều địa điểm, nhiều điều kiện thời tiết khác nhau nên cần xây dựng một ứng dụng tích hợp việc quản lý chi phí để thông báo cho nhân viên được làm việc hiệu quả và tránh được những khó khăn do thiên tai, dịch bệnh xảy ra trong quá trình làm việc.

1. **Mục tiêu đề tài.**

Các mục tiêu của đề tài:

* Định vị và quản lý lịch trình của nhân viên.
* Quản lý nhân viên thông qua định vị tự động GPS dùng Smartphone
* Tính toán chi phí đi lại của nhân viên.
* Thông báo nhắc nhở nhân viên.
* Báo cáo đến thiết bị của nhân viên về các vùng có thiên tai (động đất, sóng thần,…) và vùng có các đại dịch bệnh.
* Quản lý truy xuất đến dữ liệu lịch sử di chuyển.
* Bảo vệ tính riêng tư về đường đi.
* Quản lý truy xuất đến dữ liệu lịch sử di chuyển.

Kiến trúc hệ thống gồm:

* Một ứng dụng di động (nền tảng Android) dành cho nhân viên.
* Một Website cho người quản trị hệ thống, người quản lý dữ liệu của công ty.

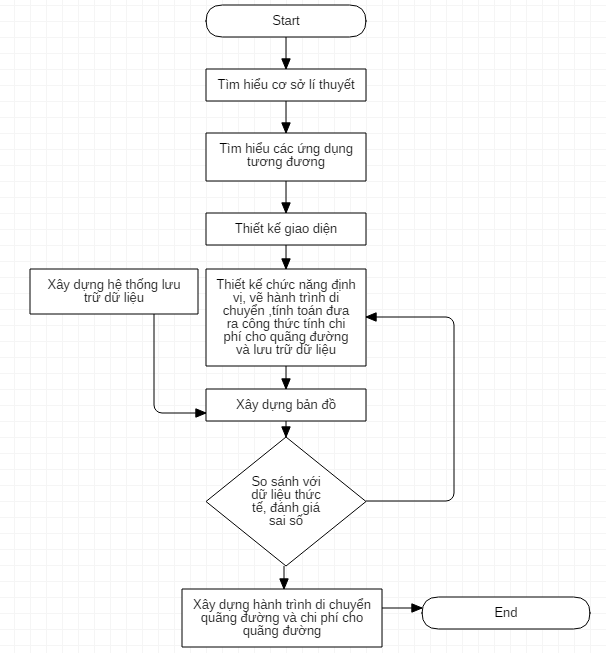
1. **Phạm vi đề tài.**

Phạm vi nghiên cứu của đề tài:

* Đề tài phát triển với giả định là các nhân viên trung thực, không sử dụng những gian lận về vị trí (Fake GPS), không lựa chọn sai phương tiện di chuyển (Car, Motorbike, Bus), cung cấp những thông tin chính xác.
* Lúc mở ứng dụng đòi hỏi nhân viên phải kết nối mạng để đăng nhập vào tài khoản cá nhân của mình.
* Cảnh báo về thiên tai và dịch bệnh trả về dạng thông báo trên ứng dụng của toàn bộ nhân viên chứ không phải là riêng một nhân viên ở một địa điểm xác định đang xảy ra thiên tai, dịch bệnh.
* Công thức tính toán chi phí là do người quản lý website nhập liệu phụ thuộc vào chính sách riêng của mỗi công ty. Vì vậy đề tài này không sử dụng một công thức tính toán cụ thể nào cả.
* Sử dụng dịch vụ của bên thứ ba (Google Firebase, Google Maps API, RSS WHO,...).

1. **Nội dung nghiên cứu.**

Quá trình nghiên cứu được thực hiện theo hình dưới:



Hình 1. 1 Nội dung nghiên cứu của đề tài

1. **Cấu trúc báo cáo.**

Báo cáo gồm các phần với các nội dung chính như sau:

* Giới thiệu đề tài: Phần này sẽ tập trung giới thiệu tổng quan về đề tài, làm rõ mục tiêu, phạm vi đề tài và giới thiệu cấu trúc của bài báo cáo.
* Ứng dụng tương đương: Nội dung phần này sẽ giới thiệu các ứng dụng tương đương với hệ thống để thông qua đó tham khảo những công nghệ ứng dụng và giao diện của ứng dụng.
* Công nghệ ứng dụng, giải pháp, thuật toán: Phần này là phần chính của bài báo cáo. Nội dung phần này sẽ giới thiệu những kiến thức cần thiết cho quá trình hiện thực đề tài (kiến thức tổng quan về GPS, Android OS, Google Maps, Google Maps Roads API, Firebase và RSS, Web), mô tả cấu trúc và các chức năng của hệ thống và các sơ đồ tuần tự của các chức năng chính, giới thiệu về kiến trúc hệ thống, các workflow, các giao diện và cách hiện thực các chức năng của hệ thống.
* Kết quả: Phần này sẽ trình bày về các cách thức vận hành hệ thống, kiểm thử chất lượng hệ thống và đánh giá các ưu nhược điểm cũng như những hạn chế của hệ thống thông qua các kết quả từ quá trình kiểm thử.
* Kết luận: Phần này tổng kết lại quá trình hiện thực đề tài và đưa ra những kết luận về hướng phát triển của đề tài đối với đời sống thực tiễn.

1. **Công nghệ ứng dụng, giải pháp, thuật toán, hiện thực hệ thống.**
2. **Công nghệ ứng dụng**

Qua quá trình tham khảo các ứng dụng tương đương. Có rất nhiều nhà cung cấp về dịch vụ bản đồ số khác nhau như: Google Maps, Microsoft Bing Maps, OpenStreetMap, OpenLayers,… Nhóm đã bàn bạc kỹ lưỡng quyết định chọn Google là nhà cung cấp chính về công cụ và các chính sách hỗ trợ cho việc hoàn thành dự án. Và dưới đây là sơ lược về các công nghệ mà nhóm đã áp dụng vào quá trình hoàn thành đề tài.

* 1. ***Sơ lược về ứng dụng client – server***

Mô hình client-server là một mô hình nổi tiếng trong mạng máy tính, được áp dụng rất rộng rã. Ý tưởng của mô hình này là máy con (client - đóng vài trò là máy khách) gửi một yêu cầu (request) để máy chủ (server - đóng vai trò người cung ứng dịch vụ), máy chủ sẽ xử lý và trả kết quả về cho máy khách.

Việc giao tiếp giữa client và server được thực hiện dưới hình thức trao đổi các thông điệp (Message). Để được phục vụ, client sẽ gởi một thông điệp yêu cầu (Request Message) mô tả về công việc muốn server thực hiện. Khi nhận được thông điệp yêu cầu, server tiến hành phân tích để xác định công việc cần phải thực thi. Nếu việc thực hiện yêu cầu này có sinh ra kết quả trả về, server sẽ gởi nó cho client trong một thông điệp trả lời (Reply Message). Dạng thức (format) và ý nghĩa của các thông điệp trao đổi giữa client và server được qui định rõ bởi giao thức (protocol) của ứng dụng.

* 1. ***Kiến trúc mạng máy tính cơ bản***

Để có thể thiết kế thành công ứng dụng mạng, trước hết chúng ta cần phải hiểu cách thức giao tiếp thông tin của chúng trong mạng máy tính.

Thông thường, khi một thiết bị muốn gửi thông tin cho thiết bị khác, thông tin đó sẽ được bao đóng qua các tầng khác nhau để đảm bảo mục đích bảo mật và tính toàn vẹn. Ở đây chúng ta sẽ tìm hiểu kiến trúc của mô hình mạng thông dụng nhất – mô hình bảy tầng OSI.

* + 1. **Mô hình OSI**

Mô hình OSI (Open Systems Interconnection Reference Model, viết ngắn là OSI Model hoặc OSI Reference Model) - tạm dịch là Mô hình tham chiếu kết nối các hệ thống mở - là một thiết kế dựa vào nguyên lý tầng cấp, lý giải một cách trừu tượng kỹ thuật kết nối truyền thông giữa các máy vi tính và thiết kế giao thức mạng giữa chúng. Mô hình này được phát triển thành một phần trong kế hoạch Kết nối các hệ thống mở (Open Systems Interconnection) do ISO và IUT-T khởi xướng. Nó còn được gọi là Mô hình bảy tầng của OSI.

* + 1. **Các tầng trong mô hình OSI**

Hình 1.1 mô tả các tầng của một mạng máy tính, chức năng chi tiết của các tầng như sau:

**Tầng vật lý (physical)**

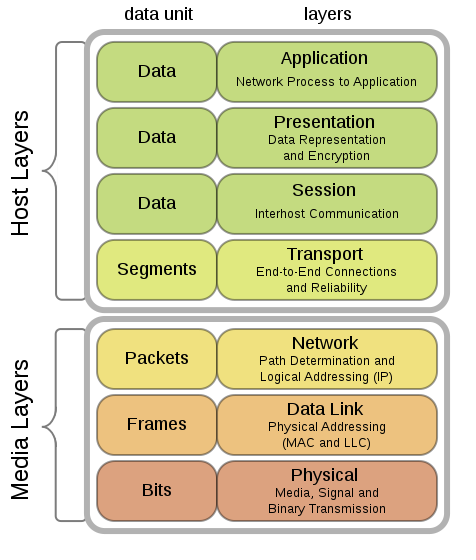
Tầng vật lý liên quan đến truyền dòng các bit giữa các máy với nhau bằng đường truyền vật lý. Tầng này liên kết các giao diện hàm cơ, quang và điện với cáp. Ngoài ra nó cũng chuyển tải những tín hiệu truyền dữ liệu do các tầng ở trên tạo ra.

Việc thiết kế phải bảo đảm nếu bên phát gửi bít 1 thì bên thu cũng phải nhận bít 1 chứ không phải bít 0. Tầng này phải quy định rõ mức điện áp biểu diễn dữ liệu 1 và 0 là bao nhiêu von trong vòng bao nhiêu giây

Chiều truyền tin là 1 hay 2 chiều, cách thức kết nối và huỷ bỏ kết nối

Định nghĩa cách kết nối cáp với card mạng: bộ nối có bao nhiêu chân, chức năng của mỗi chân

Tóm lại: Thiết kế tầng vật lý phải giải quyết các vấn đề ghép nối cơ, điện, tạo ra các hàm, thủ tục để truy nhập đường truyền, đường truyền các bít.



Bảng 1.1: Các tầng của mô hình OSI

**Tầng liên kết dữ liệu (data link)**

Cung cấp phương tiện để truyền thông tin qua liên kết vật lý đảm bảo tin cậy: gửi các khối dữ liệu với cơ chế đồng bộ hoá, kiểm soát lỗi và kiểm soát luồng dữ liệu cần thiết.

Các bước tầng liên kết dữ liệu thực hiện:

+ Chia nhỏ thành các khối dữ liệu frame (vài trăm bytes), ghi thêm vào đầu và cuối của các frame những nhóm bít đặc biệt để làm ranh giới giữa các frame.

+ Trên các đường truyền vật lý luôn có lỗi nên tầng này phải giải quyết vấn đề sửa lỗi (do bản tin bị hỏng, mất và truyền lại) giữ cho sự đồng bộ tốc độ giữa bên phát và bên thu.

Tóm lại: tầng liên kết dữ liệu chịu trách nhiệm chuyển khung dữ liệu không lỗi từ máy tính này sang máy tính khác thông qua tầng vật lý. Tầng này cho phép tầng mạng truyền dữ liệu gần như không phạm lỗi qua liên kết mạng.

**Tầng mạng (Network)**

Lập địa chỉ các thông điệp, diễn dịch địa chỉ và tên logic thành địa chỉ vật lý.

Kiểm soát và điều khiển đường truyền: Định rõ các bó tin được truyền đi theo con đường nào từ nguồn tới đích. Các con đường đó có thể là cố định đối với những mạng ít thay đổi, cũng có thể là động, nghĩa là các con đường chỉ được xác định trước khi bắt đầu cuộc nói chuyện. Các con đường đó có thể thay đổi tuỳ theo trạng thái tải tức thời.

Quản lý lưu lượng trên mạng: chuyển đổi gói, định tuyến, kiểm soát sự tắc nghẽn dữ liệu (nếu có nhiều gói tin cùng được gửi đi trên đường truyền thì có thể xảy ra tắc nghẽn).

Kiểm soát luồng dữ liệu và cắt hợp dữ liệu (nếu cần).

**Tầng giao vận (Transport)**

Thực hiện việc truyền dữ liệu giữa hai đầu nút (end - to - end)

Thực hiện kiểm soát lỗi, kiểm soát luồng dữ liệu từ máy → máy. Đảm bảo gói tin truyền không phạm lỗi, theo đúng trình từ, không bị mất mát hay sao chép.

Thực hiện việc ghép kênh, phân kênh cắt hợp dữ liệu (nếu cần). Đóng gói thông điệp, chia thông điệp dài thành nhiều gói tin và gộp các gói nhỏ thành một bộ.

Tầng này tạo ra một kết nối cho mỗi yêu cầu của tầng trên nó. Khi có nhiều yêu cầu từ tầng trên với thông lượng cao thì nó có thể tạo ra nhiều kết nối và cùng một lúc có thể gửi đi nhiều bó tin trên đường truyền.

**Tầng phiên (Session)**

Cung cấp phương tiện truyền thông giữa các ứng dụng: cho phép người sử dụng trên các máy khác nhau có thể thiết lập, duy trì, huỷ bỏ và đồng bộ hoá các phiên truyền thông giữa họ với nhau.

Nhiệm vụ chính:

Quản lý thẻ bài đối với những nghi thức: hai bên kết nối để truyền thông tin không đồng thời thực hiện một số thao tác. Để giải quyết vấn đề này tầng phiên cung cấp 1 thẻ bài, thẻ bài có thể được trao đổi và chỉ bên nào giữ thẻ bài mới có thể thực hiện một số thao tác quan trọng.

Vấn đề đồng bộ: khi cần truyền đi những tập tin dài tầng này chèn thêm các điểm kiểm tra (check point) vào luồng dữ liệu. Nếu phát hiện thấy lỗi thì chỉ có dữ liệu sau điểm kiểm tra cuối cùng mới phải truyền lại.

**Tầng trình diễn (Presentation)**

Quyết định dạng thức trao đổi dữ liệu giữa các máy tính mạng. Người ta có thể gọi đây là bộ dịch mạng. ở bên gửi, tầng này chuyển đổi cú pháp dữ liệu từ dạng thức do tầng ứng dụng gửi xuống sang dạng thức trung gian mà ứng dụng nào cũng có thể nhận biết. ở bên nhận, tầng này chuyển các dạng thức trung gian thành dạng thức thích hợp cho tầng ứng dụng của máy nhận.

Tầng trình diễn chịu trách nhiệm chuyển đổi giao thức, biên dịch dữ liệu, mã hoá dữ liệu, thay đổi hay chuyển đổi ký tự và mở rộng lệnh đồ hoạ.

Nén dữ liệu nhằm làm giảm bớt số bít cần truyền. Ở tầng này có bộ đổi hướng hoạt đông để đổi hướng các hoạt động nhập/xuất để gửi đến các tài nguyên trên mấy phục vụ

**Tầng ứng dụng (Application)**

Cung cấp các phương tiện để người sử dụng có thể truy nhập được vào môi trường OSI, đồng thời cung cấp các dịch vụ thông tin phân tán.

Tầng này đóng vai trò như cửa sổ dành cho hoạt động xử lý các trình ứng dụng nhằm truy nhập các dịch vụ mạng. Nó biểu diễn những dịch vụ hỗ trợ trực tiếp các ứng dụng người dùng, chẳng hạn như phần mềm chuyển tin, truy nhập cơ sở dữ liệu và email.

Xử lý truy nhập mạng chung, kiểm soát lỗi và phục hồi lỗi.

* 1. ***Socket, port và Java socket***

Kiến trúc OSI có thể phân thành hai tầng lớn là là Hosts layer gần với người dùng cuối hơn và Media layer gần với các thiết bị mạng hơn. Có thể xem Tầng giao vận (Transport) là tầng nằm giữa 2 phía, làm bản lề cho sự giao tiếp thông tin của ứng dụng mạng. Chúng ta sẽ tìm hiểu kỹ hơn về tầng này.

* + 1. **Socket**

**Socket** là một cổng logic mà một chương trình sử dụng để kết nối với một chương trình khác chạy trên một máy tính khác trên Internet. Chương trình mạng có thể sử dụng nhiều Socket cùng một lúc, nhờ đó nhiều chương trình có thể sử dụng Internet cùng một lúc.

Có 2 loại Socket:

+ **Stream Socket**: Dựa trên giao thức TCP (Tranmission Control Protocol) việc truyền dữ liệu chỉ thực hiện giữa 2 quá trình đã thiết lập kết nối. Giao thức này đảm bảo dữ liệu được truyền đến nơi nhận một cách đáng tin cậy, đúng thứ tự nhờ vào cơ chế quản lý luồng lưu thông trên mạng và cơ chế chống tắc nghẽn.

**+ Datagram Socket**: Dựa trên giao thức UDP (User Datagram Protocol) việc truyền dữ liệu không yêu cầu có sự thiết lập kết nối giữa 2 quá trình. Ngược lại với giao thức TCP thì dữ liệu được truyền theo giao thức UDP không được tin cậy, có thế không đúng trình tự và lặp lại. Tuy nhiên vì nó không yêu cầu thiết lập kết nối không phải có những cơ chế phức tạp nên tốc độ nhanh.

*Có thể nhận thấy Stream Socket sử dụng một cơ chế truyền thông tin đáng tin cậy, vì vậy, chúng ta sẽ tập trung nghiên cứu cách áp dụng loại hình socket này trong ứng dụng “Kéo búa bao”.*

* + 1. **Port**

**Port** xác định duy nhất một quá trình (process) trên một máy trong mạng. Hay nói cách khác là cách mà phân biệt giữa các ứng dụng.

VD: Khi máy bạn chạy nhiều ứng dụng mạng như Yahoo, Firefox, game online…

Ví dụ chương Yahoo sử dụng (port 5150 hay 5050) thì khi ai đó gửi tin nhắn đến cho bạn, lúc tin nhắn đến máy bạn nó sẽ dựa vào port để nhận biết đó là

chương trình Yahoo (port 5150) chứ ko pải là chương trình khác. Sau đó thông tin sẽ đc xử lý và hiễn thị tin nhắn lên.

– Một TCP/IP Socket gồm một địa chỉ IP kết hợp với một port xác định duy nhất một tiến trình (process) trên mạng. Hay nói cách khác Luồng thông tin trên mạng dựa vảo IP là để xác định máy một máy trên mạng còn port xác định 1 tiến trình trên 1 máy.

* + 1. **Java socket**

Nhằm tạo sự thuận tiện trong việc lập trình, Java cung cấp cho chúng ta các lớp các lớp Socket được sử dụng để biểu diễn kết nối giữa client và server. Gói java.net cung cấp hai lớp - Socket và ServerSocket - thể hiện kết nối giữa client và server.

Các Socket cung cấp kỹ thuật giao tiếp giữa hai máy tính sử dụng TCP. Một chương trình Client tạo một socket trên đầu cuối của giao tiếp và cố gắng để kết nối socket đó tới một Server.

Khi kết nối được tạo, Server tạo một đối tượng Socket trên đầu cuối của giao tiếp. Client và Server bây giờ có thể giao tiếp bằng việc đọc và ghi từ Socket.

Lớp java.net.Socket biểu diễn một Socket, và lớp java.net.ServerSocket cung cấp một kỹ thuật cho chương trình Server để nghe thông tin từ các Client và thành lập các kết nối với chúng.

**Các bước sau xảy ra khi thành lập một kết nối TCP giữa hai máy tính sử dụng Socket:**

Server khởi tạo một đối tượng ServerSocket, biểu thị số hiệu cổng (port) nào để xuất hiện giao tiếp.

Server gọi phương thức *accept()* của lớp ServerSocket. Phương thức này đợi tới khi một Client kết nối với Server trên cổng đã cho.

Sau khi Server đang đợi, một Client khởi tạo một đối tượng Socket, xác định tên Server và số hiệu cổng để kết nối tới.

Constructor của lớp Socket cố gắng để kết nối Client tới Server và số hiệu cổng đã xác định. Nếu giao tiếp được thành lập, bây giờ Client có một đối tượng Socket có khả năng giao tiếp với Server.

Trên Server-side, phương thức *accept()* trả về một tham chiếu tới một socket mới trên Server mà được kết nối với socket của Client.

Sau khi các kết nối được thành lập, giao tiếp có thể xảy ra bởi sử dụng I/O stream. Mỗi Socket có cả một *OutputStream* và *InputStream*. *OutputStream* của Client được kết nối với *InputStream* của Server, và *InputStream* của Client được kết nối với *OutputStream* của Server.

TCP là một giao thức giao tiếp hai chiều, vì thế dữ liệu có thể được gửi qua cả hai luồng tại cùng một thời điểm. Các lớp hữu ích sau đây cung cấp đầy đủ các phương thức để triển khai các Socket.

* 1. ***Đa luồng (multi-threading)***
     1. **Đa luồng là gì**

Đa luồng hay còn được gọi là Multithreading. Một chương trình đa luồng luôn có 2 tiến trình trở lên chạy song song nhau, mỗi tiến trình đó người ta gọi là một luồng (thread). Luồng là đơn vị nhỏ nhất trong java có thể thực hiện được 1 công việc riêng biệt và các luồng được quản lý bởi máy ảo java (Java Virtual Machine - JVM). Một ứng dụng java ngoài luồng chính có thể có các luồng khác thực thi đồng thời. Đa luồng giúp cho các tác vụ được xử lý độc lập giúp công việc được hoàn thành nhanh chóng. Vậy đa luồng có thể hiểu đơn giản là quá trình xử lý nhiều thread song song nhau và thực hiện các nhiệm vụ khác nhau cùng một lúc.

Server của ứng dụng cần phải quản lý rất nhiều người dùng truy cập đến một lúc, multithread là lựa chọn thích hợp để giải quyết vấn đề này.

* + 1. **Java Thread**

Ngôn ngữ java cung cấp cho ta đối tượng Thread nhằm thể hiện cơ chế đa luồng. Có hai cách chính để tạo luồng đó là tạo 1 đối tượng của lớp được thừa kế từ lớp Thread hoặc implements từ giao diện Runnable.

So sánh hai cách tạo ra thread hay dùng ở trên:

+ Giống nhau: cùng xử lý để có thể tạo ra một luồng

+ Khác nhau: nếu bạn tạo thread bằng cách implements từ interface Runnable thì bạn có thể kể thừa một class khác ngoài Class Thread, còn nếu bạn tạo thread bằng cách kế thừa Class Thread thì bạn sẽ không thể extend class nào khác vì Java không hỗ trợ đa kế thừa.

* 1. ***JSON***
     1. **Giới thiệu về JSON.**

JSON là chữ viết tắt của Javascript Object Notation, đây là một dạng dữ liệu tuân theo một quy luật nhất định mà hầu hết các ngôn ngữ lập trình hiện nay đều có thể đọc được, bạn có thể sử dụng lưu nó vào một file, một record trong CSDL rất dễ dàng. JSON có định dạng đơn giản, dễ dàng sử dụng và truy vấn hơn XML rất nhiều nên tính ứng dụng của nó hiện nay rất là phổ biến.

**Cú pháp của một chuỗi JSON:**

Cú pháp của JSON rất đơn giản là mỗi thông tin dữ liệu sẽ có 2 phần đó là key và value, điều này tương ứng trong CSDL là tên field và giá trị của nó ở một record nào đó. Tuy nhiên nhìn qua thì đơn giản nhưng nếu ta mổ xẻ nó ra thì có một vài điều như sau:

Chuỗi JSON được bao lại bởi dấu ngoặc nhọn {}.

Các key, value của JSON bắt buộc phải đặt trong dấu nháy kép {"}, Nếu có nhiều dữ liệu (nhiều cặp key => value) thì ta dùng dấu phẩy (,) để ngăn cách.

Các key của JSON bạn nên đặt chữ cái không dấu hoặc số, dấu \_ và không có khoảng trắng. Ký tự đầu tiên không nên đặt là số.

* + 1. **Kiểu dữ liệu của JSON**

JSON có 5 kiểu dữ liệu chính

- Number: kiểu số bao gồm số nguyên và số thực.

- String: kiểu chuỗi, nội dung bao bởi cặp dấu nháy kép “, những ký tự đặt biệt được escape bởi dấu \.Theo chuẩnJSON thì không sử dụng dấu nháy đơn như Javascript để bọc chuỗi.

- Boolean: kiểu luận lý bao gồm 2 giá trị là true và false

- Array: kiểu mảng, gồm các phần tử phân cách nhau bởi dấu phẩy ‘,’ và mảng được bao bởi cặp dấu [].

- Object: kiểu đối tượng, gồm những cặp giá trị đi cùng nhau, mỗi cặp phân cách bởi dấu phẩy’,’, đối tượng được bao bởi cặp dấu { và }, cặp giá trị bao gồm tên và giá trị được phân cách bởi dấu hai chấm’:’.

- Null: giá trị null.

* + 1. **Ứng dụng của chuỗi JSON**

+ Lưu trữ dữ liệu đơn thuần:

- Cấu trúc của JSON gồm các key, value bên cạnh đó còn lưu được các kiểu dữ liệu cơ bản nên thường được sử dụng để lưu trữ các dữ liệu đơn giản dưới dạng chuỗi. Dữ liệu được lưu trữ bằng JSON thường là những dữ liệu đơn giản, gọn, nhẹ.

+ Chuẩn hóa, mô tả dữ liệu:

- Các dữ liệu có cấu trúc dễ dàng được chuyển đổi sang chuỗi JSON để truyền tải qua lại trong cùng một máy tính hay giữa các máy tình trong mạng, sau khi truyền tải thì chuỗi JSON cũng dễ dàng chuyển đổi ngược lại thành dữ liệu như cũ.

1. **Phân tích chức năng hệ thống**
   1. ***Mô tả kiến trúc phần mềm.***

Ứng dụng trò chơi “Kéo Búa Bao” sẽ là ứng dụng mạng theo mô hình client – server với cấu trúc như sau:

- Một Server với SocketServer được mở sẵn và thực hiện 2 chức năng chính sau:

+ Chờ đợi kết nối từ phía client.

+ Nhận – xử lý – phản hồi các yêu cầu từ phía client dưới dạng thông điệp (Message).

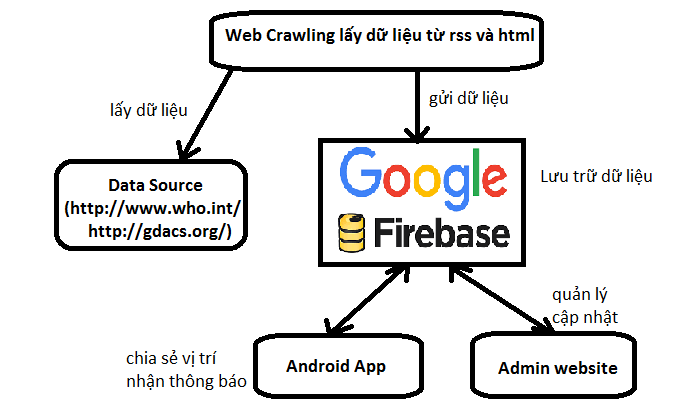
*Server phải đảm bảo có thể quản lý, xử lý, phản hồi được nhiều client cùng lúc.*

- Các Client với

+ Socket được tạo ra nhằm gửi các yêu cầu và nhận kết quả từ Server.

+ Màn hình trận đấu để quản lý việc thi đấu giữa hai Client với nhau.

- Một Database Server quản lý người dùng và trạng thái của họ (Đang offline/online/trong trận).



Hình 3.2 1: Cấu trúc hệ thống phần mềm

* 1. ***Server***
     1. ***Chức năng của server***

Server của ứng dụng gồm có các chức năng chính sau:

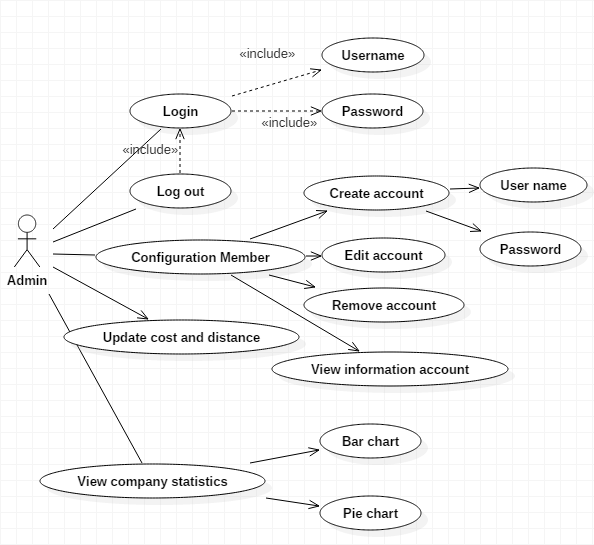
* Nhận kết nối và thông điệp của các client.
* Xử lý các yêu cầu của client và trả về kết quả, cụ thể:
  + Nếu user đăng ký thành viên thì kiểm tra thông tin trong danh sách user đã tồn tại:

Nếu thông tin user không tồn tại trước đó trong danh sách, trả về thông báo đăng ký thành công và thêm user mới và cơ sở dữ liệu.

Nếu thông tin user đã tồn tại trong hệ thống, trả về thông báo đăng ký không thành công và yêu cầu user sử dụng thông tin khác để đăng ký.

* + Nếu user đã đăng ký thành viên và đăng nhập thì kiểm tra trong database đã có user chưa, nếu có thay đổi trạng thái của user trong database thành online, thêm user vào danh sách người chơi và gửi danh sách cập nhật về cho các user hiện đang online.
  + Nếu có yêu cầu thi đấu giữa một người chơi với một người chơi khác, kiểm tra xem người được thách đấu có đang trong trận không, nếu không gửi lời mời đến người được thách đấu.
  + Nếu có người được thách đấu chấp nhận tham gia trận đấu, gửi thông tin chấp nhận về cho người thách đấu và bắt đầu trận đấu.

* Trận đấu sẽ đếm ngược trong 10 giây theo thời gian của Server. Sau 10 giây Server nhận kết quả lựa chọn gửi về từ client và tính toán người thắng, cuối cùng gửi thông tin của người chiến thắng về cho cả hai người chơi.



Hình 3.2 2: Sơ đồ chức năng dành cho website của người quản lý.

* + 1. ***Use Case Scenario***
* ***Use Case Login , Log out (người quản trị đăng nhập, đăng xuất khỏi hệ thống quản lý)***

|  |  |
| --- | --- |
| Use case | Login, Log out |
| Actor | Admin |
| Main | Đăng nhập |
| Detail | Người quản trị đăng nhập vào website bằng username và password.  Người quản trị đăng xuất khỏi website sau khi hoàn thành công việc. |

* ***Use Case Configuration Member (Chức năng thiết lập thành viên)***

|  |  |
| --- | --- |
| Usecase | Configuration Member |
| Actor | Admin |
| Main | Config |
| Detail | Xem thông tin cá nhân của nhân viên ,chỉnh sửa( thêm, sửa, xóa) và update thông tin |
| Drerequisite | Admin phải đăng nhập thành công |

* ***Use Case Update Cost And Distance (Chức năng thiết lập công thức tính toán chi phí)***

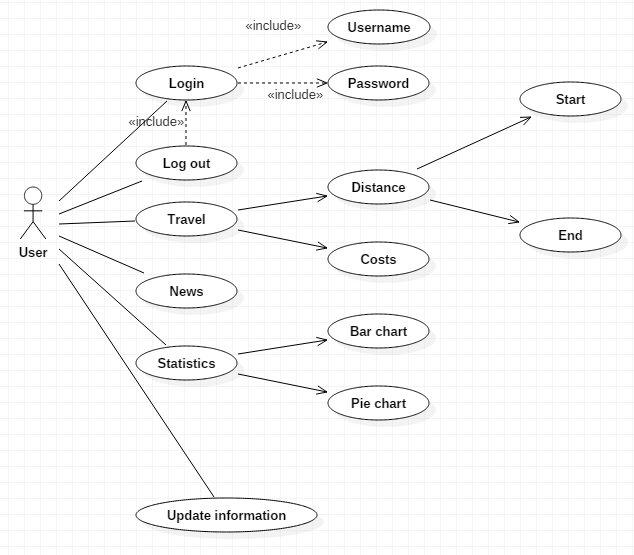
|  |  |
| --- | --- |
| Usecase | Update cost and distance |
| Actor | Admin |
| Main | Tính toán và cập nhật |
| Detail | Thiết lập / Cập nhật công thức tính toán chi phí đi lại cho từng phương tiện dựa vào điểm đến và điểm đi của nhân viên |
| Drerequisite | Nhân viên phải đăng nhập thành công |

|  |  |
| --- | --- |
| Usecase | View company statistics |
| Actor | Admin |
| Main | Thống kê toàn công ty |
| Detail | Thống kế quãng đường đi được và chi phí của toàn nhân viên công ty |
| Drerequisite | Nhân viên phải đăng nhập thành công |

* ***Use Case View Company Statistics (Chức năng xem thống kê công ty)***
  1. ***Ứng dụng di động.***
     1. ***Sơ đồ chức năng***

Ứng dụng di động dành cho nhân viên công ty có các chức năng sau đây:

* Đăng nhập vào ứng dụng
* Đăng xuất khỏi ứng dụng
* Quản lí lộ trình, chi phí đi lại: Nhân viên chọn phương tiện và bắt đầu di chuyển. Hệ thống sẽ cập nhật vị trí nhân viên theo GPS từ đó tính toán chi phí theo công thức được nhập từ Admin
* Thông báo về thiên tai và dịch bệnh mới: có một nút thông báo. Ở đây người dùng có thể xem thông tin tình trạng thiên tai và dịch bệnh mới.
* Xem thống kê về chi phí và đường đi của nhân viên thông qua biểu đồ cột và biểu đồ tròn.
* Cập nhật thông tin cá nhân của nhân viên



Hình 3.2 3: Sơ đồ chức năng của ứng dụng di động dành cho nhân viên.

* + 1. ***Use Case Scenario***
* ***Use Case Login (Chức năng đăng nhập)***

|  |  |
| --- | --- |
| Use case | Login |
| Actor | User |
| Main | Đăng nhập |
| Detail | Nhân viên đăng nhập vào ứng dụng bằng username và password |

* ***Use Case Log out (Chức năng đăng xuất)***

|  |  |
| --- | --- |
| Use case | Log out |
| Actor | User |
| Main | Đăng xuất |
| Detail | Nhân viên đăng xuất ra khỏi ứng dụng |

* ***Use Case Travel (Chức năng quản lí lộ trình và chi phí)***

|  |  |
| --- | --- |
| Use case | Travel |
| Actor | User |
| Main | Lộ trình và chí phí |
| Detail | Nhân viên chọn điểm đến và điểm đi và phương tiện di chuyển cho mình, hệ thống sẽ hiện ra đường đi và giá phù hợp nhất cho nhân viên , khi nhân viên di chuyển sẽ thống sẽ cập nhật vị trí của nhân viên liên tục. |
| Drerequisite | Nhân viên phải đăng nhập thành công |

* ***Use Case News (Chức năng xem thông báo về thiên tai, dịch bệnh)***

|  |  |
| --- | --- |
| Usecase | New |
| Actor | User |
| Main | Thông báo |
| Detail | Xem thông tin tình trạng thiên tai dịch bệnh mới |

* ***Use Case Statistics (Chức năng xem thống kê)***

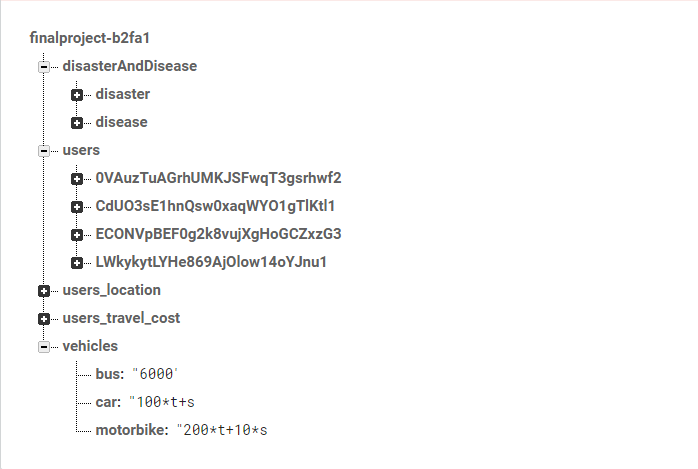
|  |  |
| --- | --- |
| Usecase | Statistics |
| Actor | User |
| Main | Status |
| Detail | Xem thống kế về chi phí và đường đi của nhân viên thông qua biểu đồ cột và biểu đô tròn |
| Drerequisite | Nhân viên phải đăng nhập thành công |

* ***Use Case Update Information (Chức năng cập nhật thông tin)***

|  |  |
| --- | --- |
| Usecase | Update information |
| Actor | User |
| Main | Cập nhật thông tin |
| Detail | Cập nhật thông tin cá nhân của nhân viên |
| Drerequisite | Nhân viên phải đăng nhập thành công |

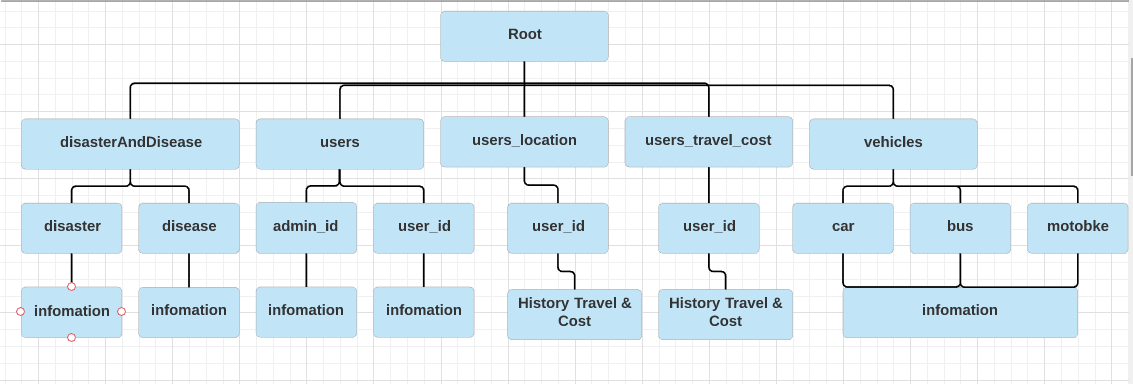
* 1. ***Cấu trúc cơ sở dữ liệu.***
     1. ***Tổng quan về cấu trúc cơ sở dữ liệu của hệ thống.***

Hệ thống quản lý chi phí đi lại của nhân viên bán hàng trong công ty sử dụng dịch vụ Firebase Realtime Database của Google. Cơ sở dữ liệu của hệ thống bao gồm 1 node duy nhất (node “finalproject-b2fa1”), node này có tên được đặt dựa theo đề tài là đề tài luận văn. Tất cả các node con được tạo ra sau đó đều bắt nguồn từ node gốc này. Có 5 node con chính: disasterAndDisease, users, users\_location, users\_travel\_cost và vehicles.



Hình 3.2 4: Tổng quan về cấu trúc cơ sở dữ liệu.

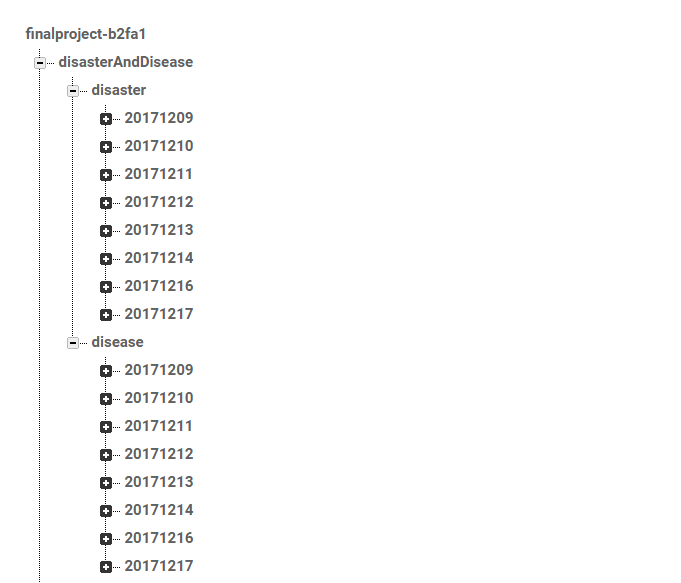
* + 1. ***Mô hình cấu trúc cơ sở dữ liệu.***

******

Hình 3.2 5: Mô hình cấu trúc cơ sở dữ liệu

* + 1. ***Chức năng các Node.***
       - 1. ***Node “disasterAndDisease”.***

Node “disasterAndDisease” lưu trữ thông tin của các thiên tai, dịch bệnh được xử lý thông qua việc chiếc xuất dữ liệu từ các tập tin RSS nhờ thủ thuật “Web crawler” bao gồm 2 node con nhỏ là “disaster” và “disease”. Bộ phận này là cơ sở dữ liệu cho việc cung cấp thông báo về thiên tai và dịch bệnh đến ứng dụng di động được sắp xếp cụ thể theo từng ngày.



Hình 3.2 6: Cấu trúc cơ sở dữ liệu của “disasterAndDisease”.

* + Node “disaster”.

Node này lấy thông tin từ trang http://gdacs.org/ và có nhiệm vụ lưu trữ thông tin theo các node con theo định dạng tên là năm/tháng/ngày mà dữ liệu trả về từ trang này.



Hình 3.2 7: Cấu trúc cơ sở dũ liệu của “disaster”.

Trong các node con được định dạng tên là năm/tháng /ngày còn có các node con nhỏ hơn là những node lưu trữ thông tin của thiên tai xảy ra bao gồm: thời gian xảy ra thiên tai, mức độ báo động, quốc gia, kinh độ, vĩ độ, thiệt hại... Các node này được đặt tên tự động theo thứ tự sự kiện diễn ra và bắt đầu từ 0.

Hình 3.2 8: Cấu trúc cơ sỡ dữ liệu của các node con.

* + Node “disease”.

Cơ sở dữ liệu của node này hoàn toàn tương tự với cấu trúc cơ sở dữ liệu của node “disaster” và lấy thông tin từ trang http://www.who.int/en/.

* + - * 1. ***Node “users”.***

Node “users” lưu trữ thông tin của toàn bộ tài khoản của người quản trị hệ thống, các tài khoản này thường được các công ty giao cho bộ phận quản lý nhân sự. Bộ phận này sẽ làm nhiệm vụ thêm tài khoản khi có nhân viên mới tới, xóa tài khoản khi có nhân viên rời đi. Theo sau node “users” là các node con được đặt tên bằng UID, là mã định danh người dùng, được tạo tự động từ hệ thống (ví dụ: CdUO3sE1hnQsw0xaqWYO1gTlKtl1). Mỗi node con này sẽ chứa thông tin của một tài khoản của người quản trị hệ thống khác nhau, bao gồm các node:

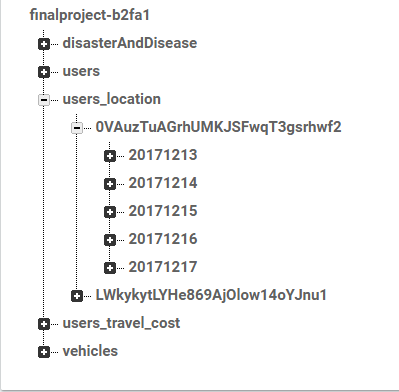
* “email”: địa chỉ e-mail dùng để đăng nhập.
* “employeeCode”: ID của nhân viên trong công ty
* “imageProfiles”: ảnh đại diện của nhân viên là một đường dẫn đến một url hình ảnh.
* “name”: Tên của nhân viên (chủ tài khoản).
* “password”: là mật khẩu xác thực đăng nhập tài khoản.
* “type”: dùng để phân quyền tài khoản như là “admin” hay là “user”.



Hình 3.2 9: Cấu trúc cơ sở dữ liệu của “users”

* + - * 1. ***Node “users\_location”.***

Node “users\_location” chưa các node con nhỏ hơn là UID của nhân viên trong công ty. Trong các node có tên là UID của nhân viên có các node lưu trữ toàn bộ thông tin về quãng đường và danh sách các tọa độ đã đi qua của một nhân viên và được lưu trữ theo từng ngày, được đặt tên bằng ngày mà dữ liệu đó được gửi lên theo định dạng năm/tháng/ngày.

******

Hình 3.2 10: Cấu trúc cơ sở dữ liệu của “users\_location”.

Các chuyến đi trong cùng một ngày sẽ được lưu trữ vào cùng một node được đặt tên tự động đã nêu trên. Từng chuyến đi sẽ được lưu trữ vào từng node riêng và được đặt tên một cách tự động. Trong node này bao gồm:

* “currentVehicle” : phương tiện di chuyển.
* “lat”: vĩ độ.
* “lng”: kinh độ.
* “time” : thời gian di chuyển được tính bằng giây.
* “totalCost”: tổng chi phí của chuyến đi được tính bằng đơn vị tiền tệ.
* “totalDistance”: tổng khoảng cách của chuyến đi được tính bằng ki lô mét.
* “totalTime”: tổng thời gian di chuyển của cả chuyến đi



Hình 3.2 11: Toàn bộ cấu trúc cơ sở dữ liệu của “users\_location”.

* + - * 1. ***Node “users\_travel\_cost”.***

Node này có cấu trúc cơ sở dữ liệu giống như node “users\_location” nhưng node này làm nhiệm vụ là cung cấp cơ sở dữ liệu cho việc vẽ các loại biểu đồ hằng ngày cũng như hàng tháng của các nhân viên.

* + - * 1. ***Node “vehicles”.***

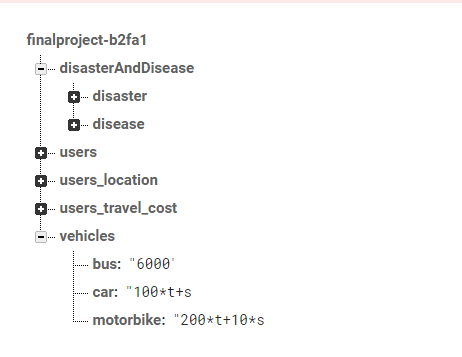
Node “vehicles” chứa công thức tính toán chi phí đi lại cho tất cả các loại phương tiện hiện có trong hệ thống, theo sau là các node con được đặt tên theo phương tiện, bao gồm: “Bus” (xe buýt), “Car” (xe ô tô), “Motorbike” (xe máy) có giá trị là các công thức tính toán chi phí dành cho phương tiện đó. Mỗi công thức có thể bao gồm:

* Các hằng số: hỗ trợ kiểu số thực.
* Các biến số: hỗ trợ 2 biến số chính:

➢ s (tính theo m): quãng đường mà phương tiện đó đã đi.

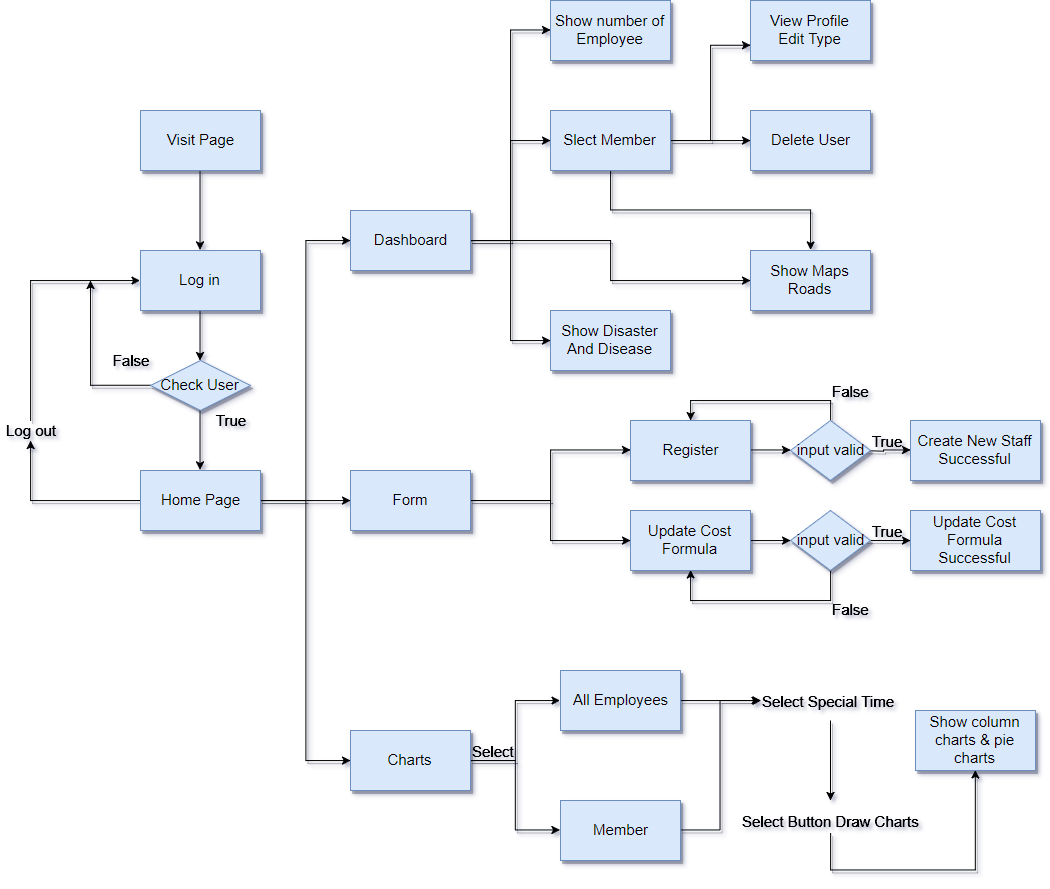
➢ t (tính theo giây): thời gian mà nhân viên sử dụng phương tiện đó.

* Các toán tử: cộng (+), trừ (–), nhân (\*), chia (/), lũy thừa (^), modulo (%).
* Các hàm tính toán: abs (giá trị tuyện đối), cbrt (lập phương), ceil (giới hạn trên), cos (cos của một góc), sin (sin của một góc), log (logarit tự nhiên – logarit cơ số e), sqrt (căn bậc 2 của một số), …



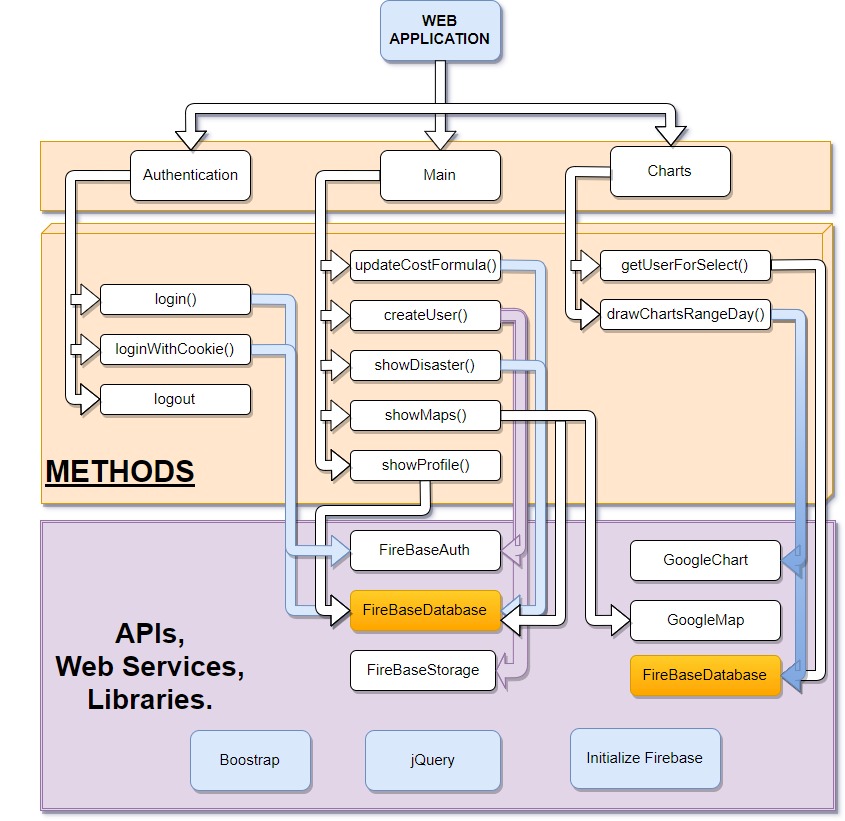
Hình 3.2 12: Cấu trúc cơ sở dữ liệu của “vehicles”

1. **Hiện thực hệ thống.**
   1. ***Website.***
      1. ***WorkFlow.***



Hình 3.3 1: WorkFlow website người quản trị hệ thống

* + 1. ***Sơ đồ kiến trúc.***

******

Hình 3.3 2: Kiến trúc hệ thống website quản lý.

***Các khối chức năng chính***

* ***Authentication***

Khối này có nhiệm vụ xác thực thông tin đăng nhập của người quản trị vào hệ thống cũng như nhận lệnh thoát khi người quản trị không còn trong phiên làm việc.

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| Login() | Kiểm tra thông tin đăng nhập của người quản trị hệ thống thông qua FirebaseAuth. |
| loginWithCookie() | Nếu người quản trị hệ thống đăng nhập vào và lựa chọn “Ghi nhớ đăng nhập” thì hệ thống sẽ lưu lại email và password cho phiên làm việc tiếp theo mà không cần nhập email và password |
| Logout() | Đăng xuất khỏi hệ thống khi hết phiên làm việc |

* ***Main***

Khối này có nhiệm vụ lấy thông tin từ FirebaseDatabase sau đó hiển thị lên website theo thao tác của người quản trị. Ngoài ra, khối này còn có chức năng thêm, xóa người dùng, cập nhật công thức tính toán chi phí cho từng loại phương tiện

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| updateCostFormula() | Cập nhật công thức tính toán chi phí do người quản trị hệ thống định nghĩa, thông tin công thức được lưu vào FirebaseDatabase. |
| createUser() | Tạo tài khoản mới, lưu thông tin vào hệ thống xác thực FirebaseAuth, lưu thông tin  người dùng vào hệ thống FirebaseDatabase và FirebaseStorage. |
| showDisaster() | Lấy thông tin về thiên tai, dịch bệnh từ FirebaseDatabase sau đó hiển thị lên trang chủ Website. |
| showMaps() | Lấy dữ liệu chi phí đi lại từ FirebaseDatabase của một người dùng và vẽ lại thành lộ trình mà người đó đã đi theo ngày trên Google Maps. |
| showProfile() | Lấy dữ liệu thông tin về các nhân sau đó hiển thị lên trang chủ. Người quản trị có thể thay đổi quyền hạn của người dùng. |

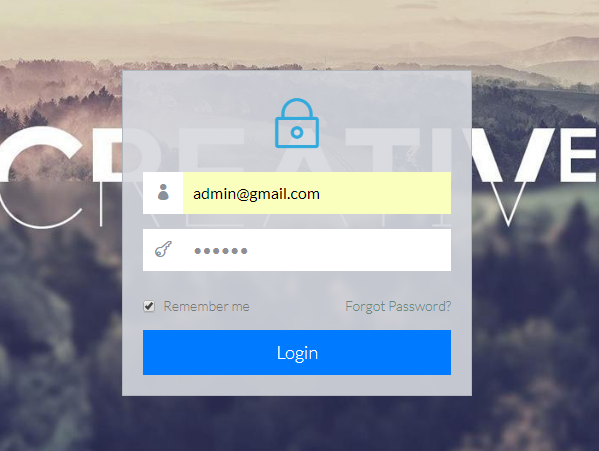
* ***Charts***

Khối này có nhiệm vụ liên kết với cơ sở dữ liệu và tiến hành thống kê theo thao tác của người quản trị.

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| getUsersForSelect() | Chọn đối tượng để xem thống kê. Có thể là toàn bộ nhân viên hoặc một nhân viên cụ thể nào đó. |
| drawChartRangeDay() | Chọn mốc thời gian và tiến hành vẽ lại biểu đồ thông qua dữ liệu thu thập trước đó của một nhân viên cụ thể được cập nhật lên FirebaseDatabase và dùng GoogleChart để tiến hành vẽ biểu đồ thống kê. |

* + 1. ***Giao diện.***
* Trang đăng nhập.

Người quản trị vào website và tiến hành đăng nhập bằng tài khoản của người quản trị. Trường hợp người quản trị muốn lưu lại tài khoản và mật khẩu cho lần sau thì chọn “Remember me”.

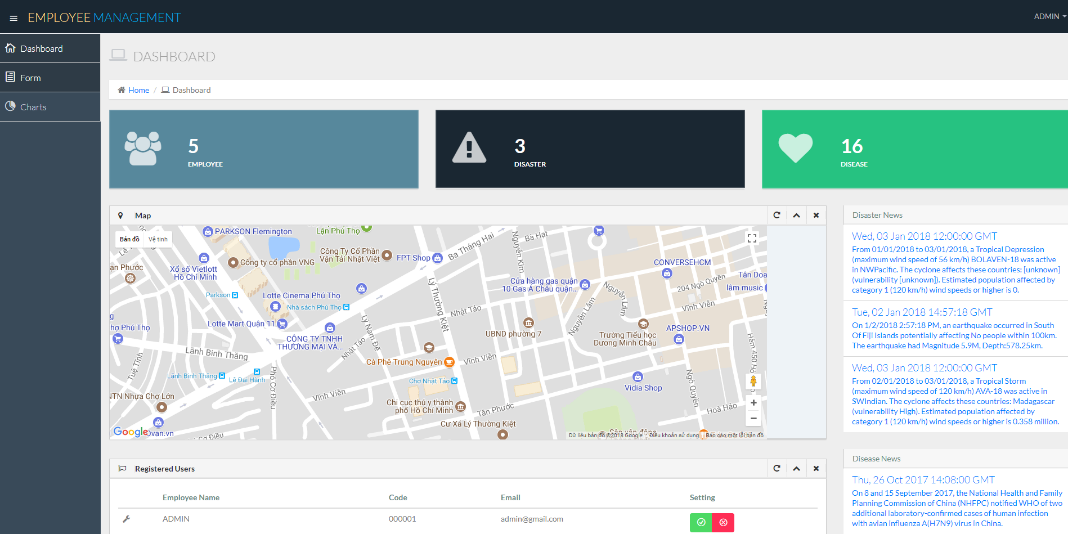


Hình 3.3 3: Giao diện đăng nhập của người quản trị hệ thống.

* Trang chủ.

Sau khi đăng nhập thành công vào hệ thống. Trang chủ của người quản trị sẽ được hiển thị rất nhiều mục với những chức năng khác nhau

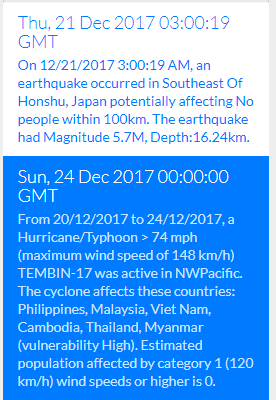
* Hiển thị số lượng nhân viên.



Hình 3.3 4: Trang chủ website hệ thống

* Hiển thị những tin tức dịch bệnh và thiên tai.

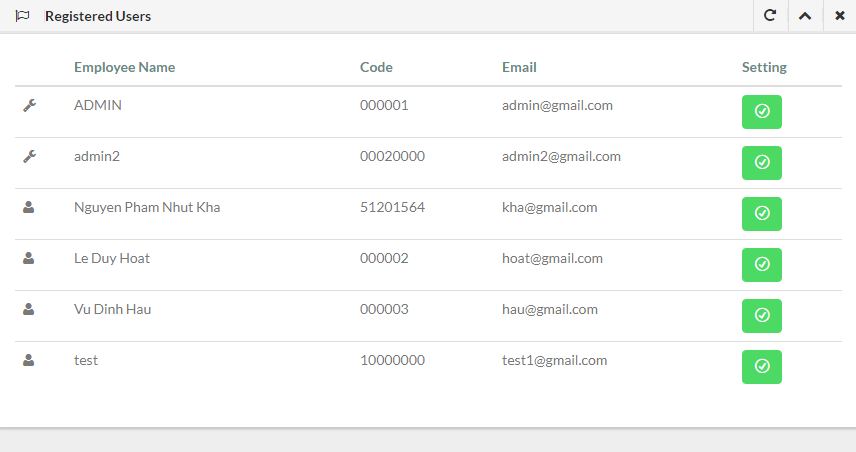
Người quản trị có thể chọn thông tin và truy cập vào để xem rõ hơn về thiên tai, dịch bệnh.



Hình 3.3 5: Trang chủ của website hệ thống – phần thông báo thiên tai

* Hiển thị danh sách nhân viên.

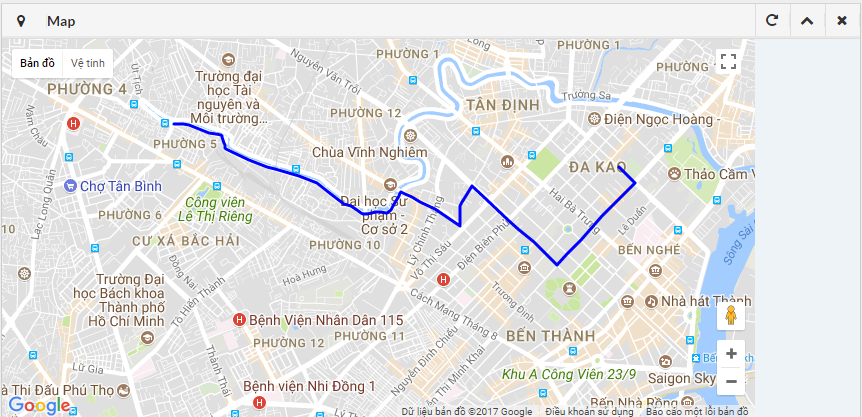
Tại đây người quản trị có thể xem danh sách tài khoản các thành viên trong công ty (kể cả người quản trị), người quản trị có thể cài đặt thông tin, xóa và xem lộ trình của nhân viên trên GoogleMaps



Hình 3.3 6: Trang chủ website của hệ thống - phần xem danh sách nhân viên.

* Hiển thị di chuyển của nhân viên cụ thể trên bản đồ.

Chọn nhân viên bạn muốn hiển thị ở danh sách trên bằng cách chọn vào phần “setting”, trang web sẽ chuyển đến bản đồ và hiển thị lộ trình của nhân viên bạn muốn xem.



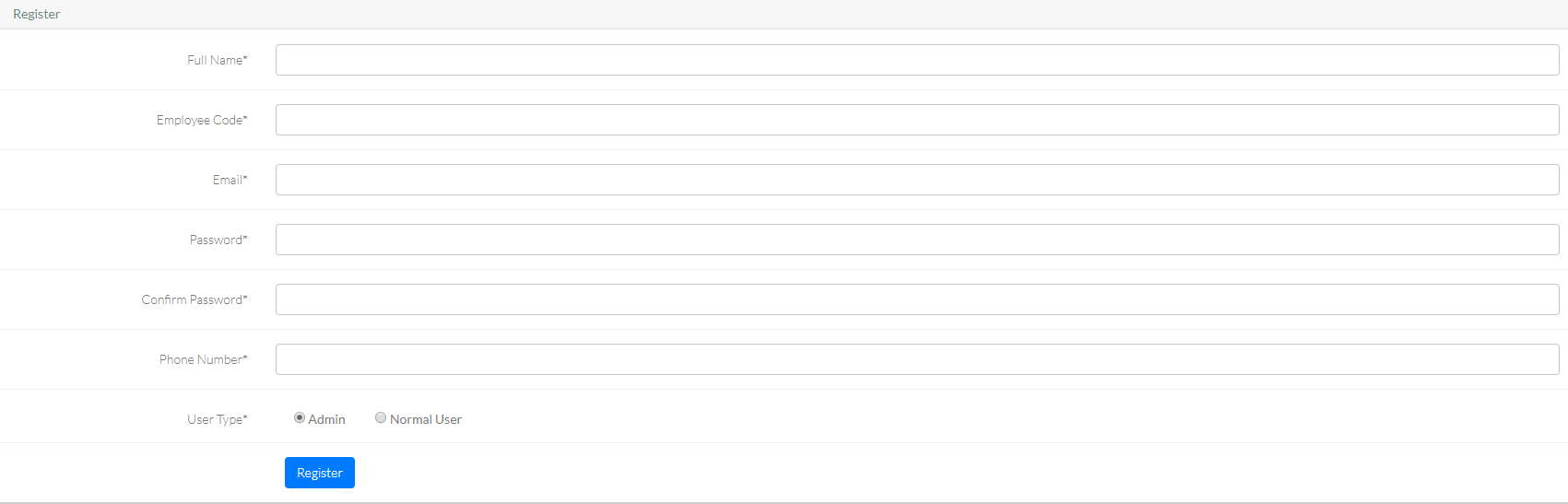
Hình 3.3 7: Bản đồ di chuyển trong ngày của một nhân viên cụ thể.

* Trang quản lý.

Trên thanh công cụ của website quản lý. Người quản lý chọn mục “Form” để tiến hành các thao tác như: thêm thành viên, cập nhật công thức tính toán chi phí.

* Thêm thành viên.

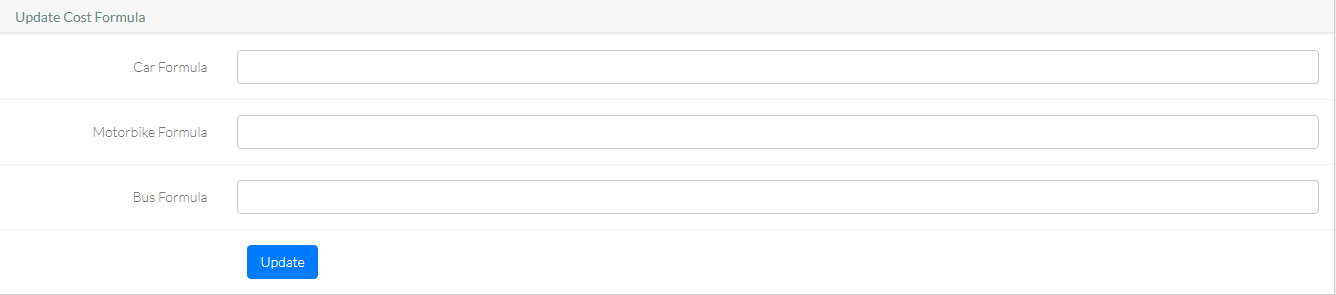
Người quản trị sẽ phải điền vào một mẫu sẵn có để khởi tạo thành viên, tại đây người quản trị có thể cài đặt quyền hạn của tài khoản bằng cách chọn vào “User Type”. Hoàn thành xong bấm “Register” để đăng ký.



Hình 3.3 8: Trang thêm thành viên.

* Cập nhật công thức tính toán chi phí.

Tùy theo chính sách của công ty mình mà người quản trị sẽ phải cập nhật công thức tính toán chi phí bằng hình thức điền vào mẫu theo định dạng cho sẵn sau đó chọn “Update”



Hình 3.3 9: Trang cập nhật công thức tính toán chi phí.

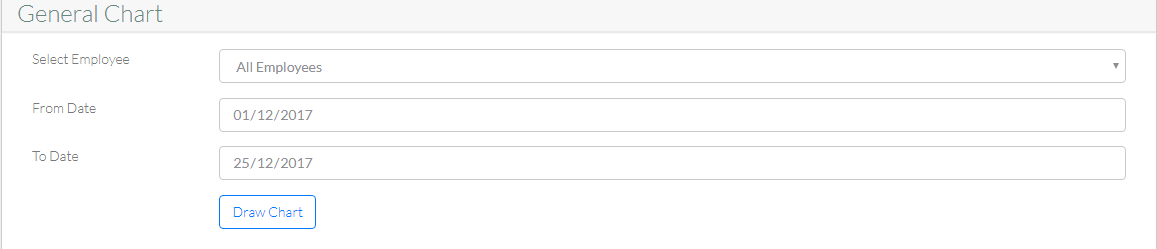
* Trang xem thống kê.

Trên thanh công cụ có phần “Charts”. Đây chính là phần để người quản trị xem thống kê của cả công ty cũng như một nhân viên cụ thể theo thời gian.

* Xem thống kê của cả công ty theo thời gian.

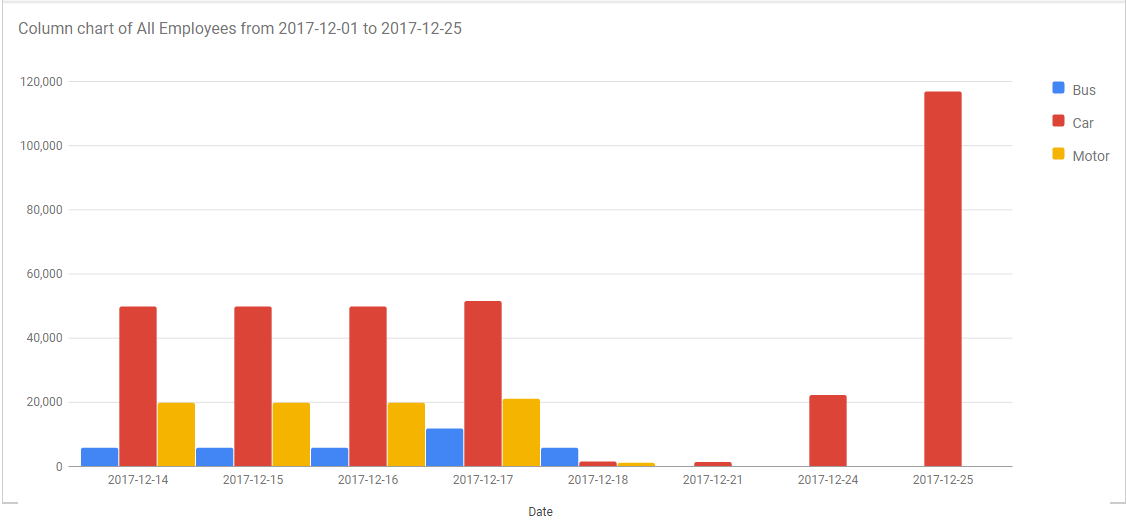
Để xem thống kê của cả công ty trong tháng 12 ấn lựa chọn

“All Employees” và tiến hành chọn ngày sau đó ấn “DrawChart”.

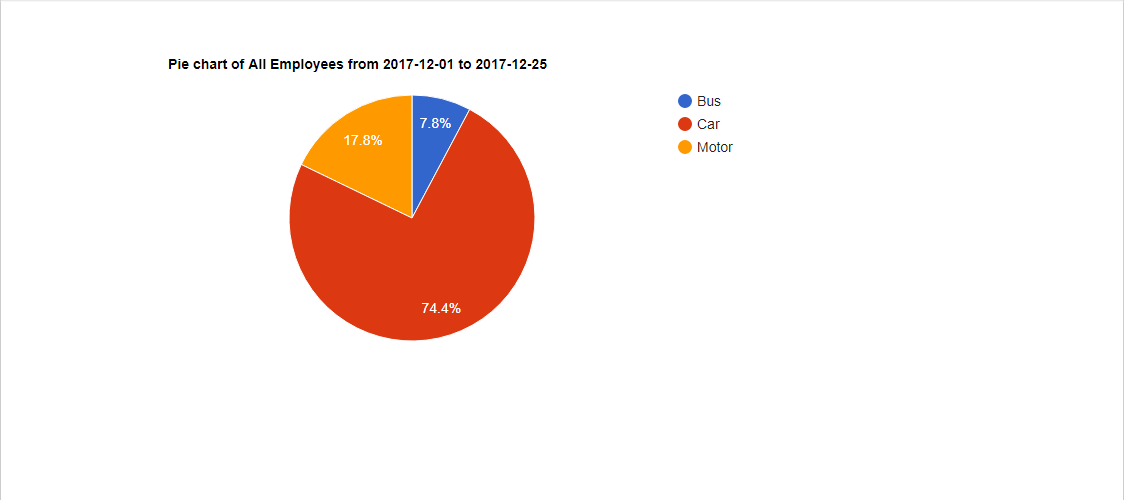


Hình 3.3 10: Trang lựa chọn đối tượng xem thống kê.

Và kết quả sẽ hiển thị bên dưới là các biểu đồ.



Hình 3.3 11: Biểu đồ cột của tất cả nhân viên từ 01/12/2017 – 25/12/2017.

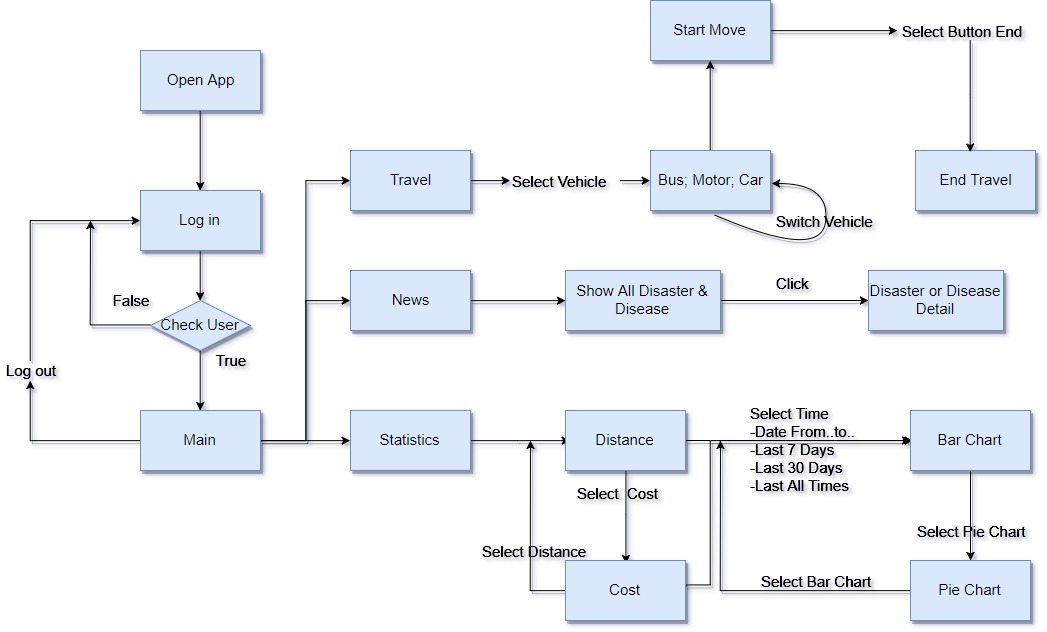


Hình 3.3 12: Biểu đồ tròn của tất cả nhân viên từ 01/12/2017 – 25/12/2017.

* Xem thống kê của một nhân viên cụ thể theo thời gian.

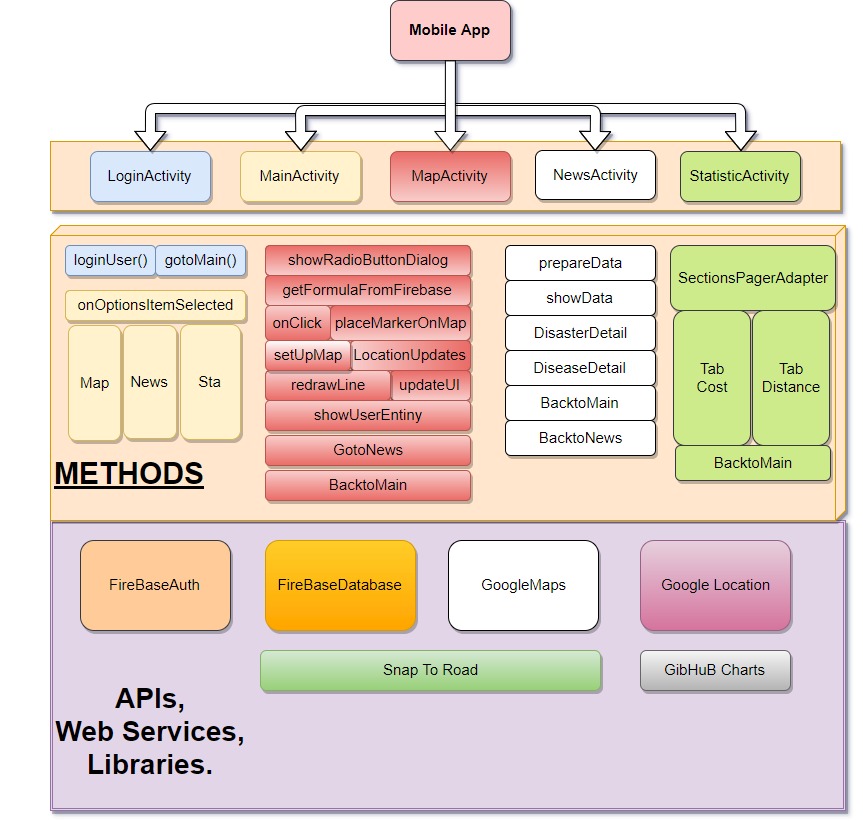
Tương tự như xem thống kê của công ty, người quản trị sẽ phải lựa chọn thành viên cụ thể, chọn mốc thời gian và tiến hành “Draw Chart”.

* 1. ***Ứng dụng di động.***
     1. ***WorkFlow.***



Hình 3.3 13 WorkFlow của ứng dụng di động.

* + 1. ***Sơ đồ kiến trúc***



Hình 3.3 14: Sơ đồ kiến trúc của ứng dụng di động.

* ***Các khối chức năng chính.***
* ***LoginActivity.***

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| loginUser | Kiểm tra thông tin đăng nhập của người dùng thông qua FirebaseAuth, nếu đúng thì cho phép đăng nhập vào hệ thống. |
| gotoMain | Sau khi đăng nhập thành công. Lớp login sẽ chuyển đến lớp “MainActivity”. |

* ***MainActivity.***

Tại lớp này. Màn hình chính sẽ hiển thị và cho phép người dùng lựa chọn chức năng kế tiếp

* ***MapsActivity.***

MapsActivity là lớp chuyển tiếp từ MainActivity khi người dùng lựa chọn vào “Travel”. Lớp này có nhiệm vụ thu thập thông tin về lộ trình di chuyển của người dùng, vẽ đường đi, tính toán chi phí

và khoảng cách di chuyển rồi lưu vào Firebase Database và hiển thị cho người dùng xem lộ trình, chi phí của chính mình. Xem thông báo khi có thiên tai, dịch bệnh.

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| showRadioButtonDialog | Cho phép người dùng chọn phương tiện di chuyển. Nhận thông tin này gửi dữ liệu lên FireBaseDatabase. |
| getFormulaFromFirebase | Sau khi gửi dữ liệu về phương tiện lên FireBaseDatabase. Ứng dụng sẽ được trả về công thức tính toán chi phí phù hợp với phương tiện đó |
| onClick | Người dùng tiến hành chọn các hành động trên ứng dụng di động của mình như: “START”, “END”, xem thông báo về thiên tai, dịch bệnh. |
| placeMarkerOnMap | Sau khi người dùng thực hiện “onClick” thì hệ thống bắt đầu đánh dấu địa điểm của người dùng để bắt đầu quản lý. |
| setUpMap | Hiển thị vị trí hiện tại của người dùng thông qua Google Maps API. |
| LocationUpdates | Liên tục nhận về các tọa độ mà người dùng đi qua từ Google Location API, sử dụng Snap to Roads để tăng độ chính xác của các dữ liệu này nhằm phục vụ cho việc vẽ đường. |
| redrawLine | Vẽ lại đường đi của người dùng. |
| updateUI | Cập nhật chi phí đi lại của người dùng lên FirebaseDatabase đồng thời hiển thị lên màn hình ứng dụng của người dùng. |
| showUserEntity | Hiển thị lên màn hình ứng dụng về khoảng cách di chuyển (km), chi chí (VNĐ), và bản đồ di chuyển. Các dữ liệu này được lấy từ FirebaseDatabase. |
| gotoNews | Bởi vì những thông tin về thiên tai và dịch bệnh luôn là quan trọng nên tại màn hình nào của ứng dụng cũng phải có chức năng thông báo. Mọi thông tin được nhận từ database. |
| BacktoMain | Quay lại màn hình chính của ứng dụng để xem thêm về biểu đồ cá nhân, thông tin khác. |

* ***NewsActivity.***

Lớp này chuyển tiếp từ lớp “MainActivity”. Lớp này có nhiệm vụ nhận thông tin từ FireBaseDatabase về những thông tin về thiên tai, dịch bệnh. Các dữ liệu này được trích xuất liên tục để so sánh, đối chiếu tránh sự trùng lặp. Sau đó hiển thị cho người dùng.

|  |  |
| --- | --- |
| Phương thức | Nhiệm vụ |
| prepareData | Thu nhận những thông tin từ Database đồng thời so sánh xem những thông tin này đã hiển thị hay chưa để tránh sự trùng lặp. |
| showData | Hiển thị toàn bộ thông tin về thiên tai, dịch bệnh sau khi đã lọc. |
| DisasterDetail | Hiển thị cụ thể thông tin thiên tai. |
| DiseaseDetail | Hiển thị cụ thể thông tin dịch bệnh. |
| BacktoNews | Quay lại trang thông tin. |
| BacktoMain | Quay lại màn hình chính. |

* ***StatisticActivity.***

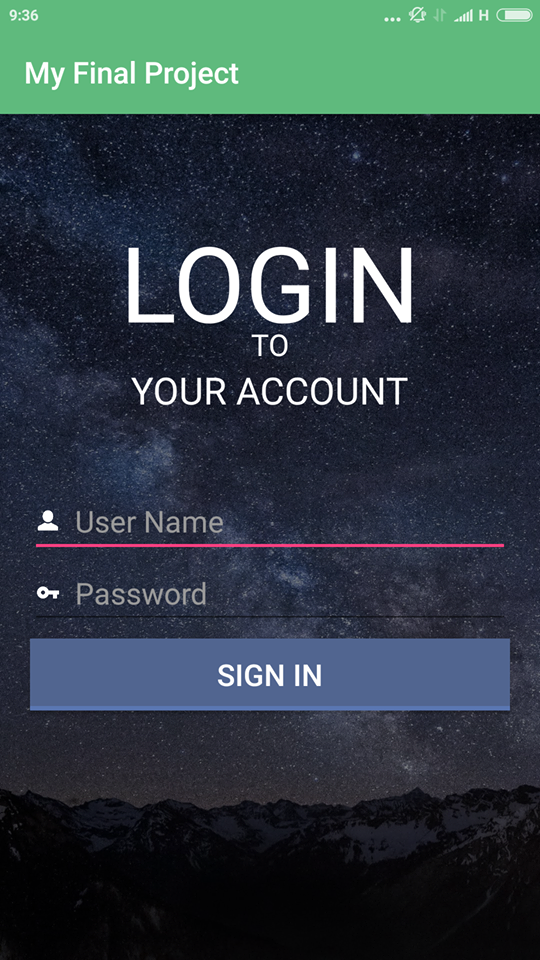
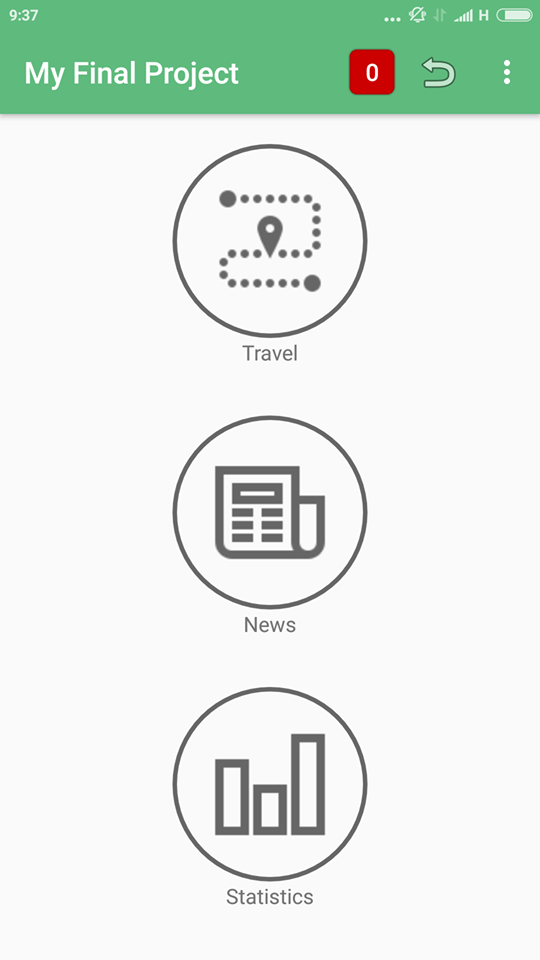
Lớp này chuyển tiếp từ lớp “MainActivity”. Lớp này có nhiệm vụ nhận dữ liệu từ database, nhận lệnh từ “onClick” của người dùng để tiến hành vẽ lại biểu đồ thống kế theo ngày, theo khoảng thời gian đặc biệt theo biểu đồ hình trụ và hình tròn thông qua MPAndroid Charts.

* + 1. ***Giao diện***

Lần đầu bật ứng dụng lên. Ứng dụng sẽ yêu cầu đăng nhập và điều bắt buộc là phải có kết nối mạng và mở chức năng định vị GPS.

* Màn hình đăng nhập - màn hình chính.

Sau khi đăng nhập thành công. Màn hình chính sẽ xuất hiện như hình dưới:

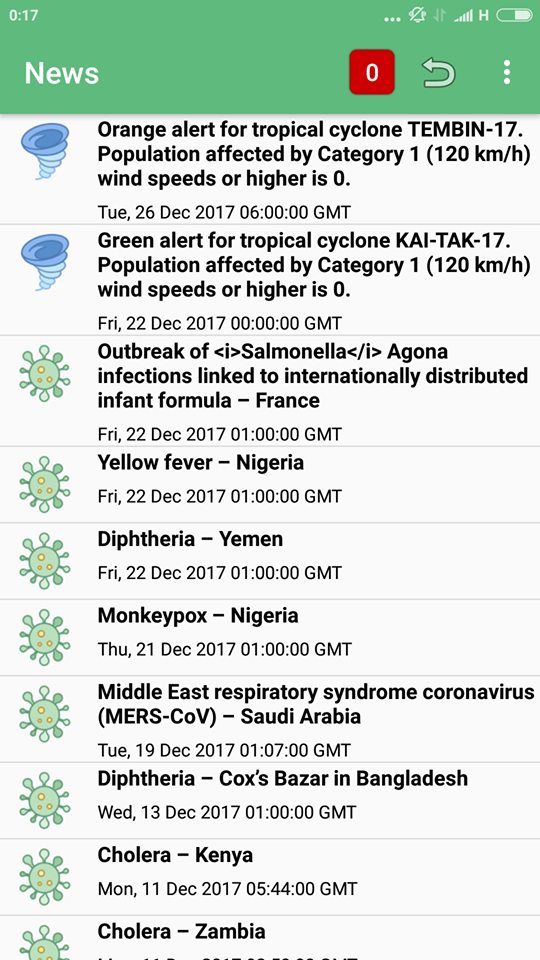
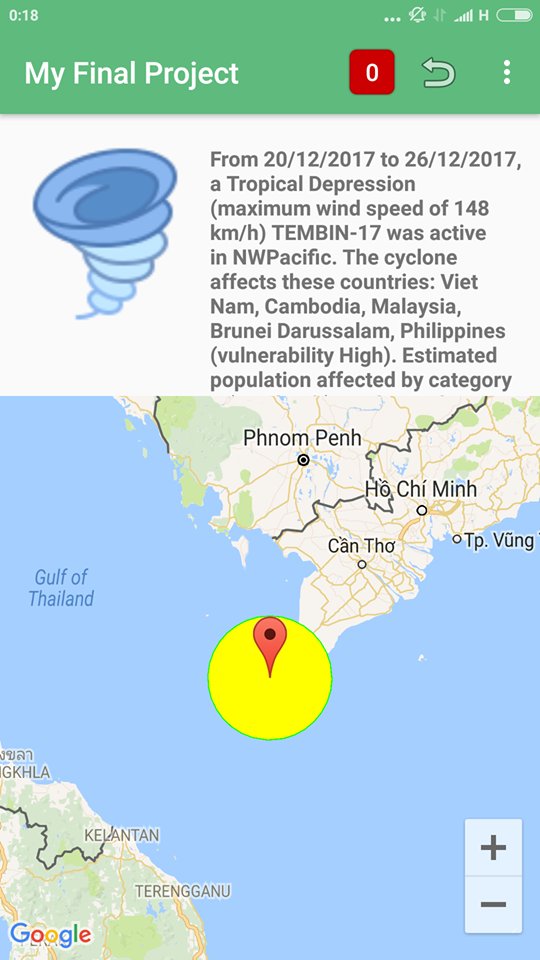
Hình 3.3 15: Giao diện ứng dụng di động từ “log in” đến màn hình chính.

* Màn hình di chuyển (Travel).

Nhân viên bắt đầu đi lại thì phải chọn vào nút Travel để hệ thống xác nhận định vị. Sau khi ấn vào Travel thì một giao diện mới sẽ hiện ra cho phép nhân viên chọn phương tiện đi lại là: Xe Bus, Car và Motorbike. Nhân viên bắt đầu di chuyển và bấm Start. Trên màn hình sẽ hiện lên bản đồ, khoảng cách và chi phí đi lại.

* Màn hình đọc thông báo (News).

Tại màn hình xem thông báo (News), người dùng có thể chọn xem cụ thể một thông báo với những thông số rõ ràng, sau đó có thể chọn quay lại bằng nút “quay lại”

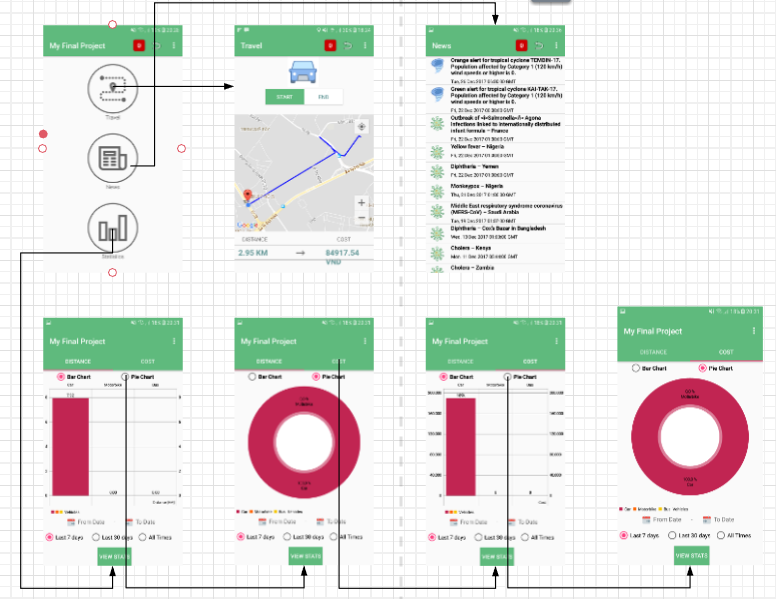
 

Hình 3.3 16: Xem cụ thể một thiên tai (Bão TEMBIN).

* Màn hình xem thống kê (Statistics).

Tại màn hình xem thống kê. Người dùng có thể linh hoạt chọn giữa xem thống kê theo từng loại biểu đồ khác nhau.

Các chức năng của ứng dụng và các chuyển tiếp giao diện cụ thể như hình bên dưới:



Hình 3.3 17: Giao diện chuyển tiếp giữa các chức năng.

1. **Kết quả.**
2. **Kiểm thử hệ thống.**
   1. ***Tiêu chí kiểm thử.***

Kiểm thử hệ thống quản lý chi phí đi lại của nhân viên trong công ty mong muốn đạt được những tiêu chí sau đây:

* Độ chính xác về GPS của thiết bị
* Độ chính xác của Realtime Database
* Hiệu năng của hệ thống (pin của thiết bị…)
* Chức năng của ứng dụng bao gồm cả Website quản lý và ứng dụng di động.
* Thời gian cảnh báo về thiên tai, dịch bệnh.
  1. ***Kế hoạch kiểm thử.***
     1. ***Kiểm thử các khối chức năng chính của hệ thống.***

Nhóm sẽ thực hiện kiểm thử các chức năng chính của hệ thống (website và ứng dụng di động). Tất cả chức năng chính của hệ thống sẽ được kiểm thử kỹ lưỡng bao gồm:

* Website:
* Đăng nhập.
* Xem thống kê của nhân viên.
* Tạo tài khoản.
* Thiết lập công thức tính toán chi phí.
* Xem thống kê công ty.
* Ứng dụng di động:
* Đăng nhập.
* Chọn phương tiện và bắt đầu di chuyển.
* Quản lý lộ trình và chi phí đi lại.
* Xem thông báo về dịch bệnh và thiên tai.
* Xem thống kê của cá nhân.
  + 1. ***Kiểm thử về hiệu năng của hệ thống (độ hao pin của thiết bị).***

Nhóm sẽ thực hiện kiểm thử mức độ hao pin của thiết bị di động khi sử dụng ứng dụng di động của đề tài bằng phương pháp sử dụng ứng dụng di động của đề tài trong thực tế.

Việc kiểm thử về độ hao pin của thiết bị bao gồm các thông số:

* + Dung lượng pin ban đầu: được tính bằng phần trăm (%) pin của thiết bị.
  + Dung lượng pin sau lúc kiểm thử sau: được tính bằng phần trăm (%) pin của thiết bị.
  + Dung lượng pin tiêu tốn khi sử dụng ứng dụng: đơn vị tính là miliampe giờ.
  + Dung lượng dữ liệu di động (3G) tiêu tốn khi sử dụng ứng dụng: đơn vị tính là megabyte (MB).
    1. ***Kiểm thử lộ trình.***

Nhóm thực hiện kiểm thử độ chính xác của lộ trình do ứng dụng di động của hệ thống định vị bằng phương pháp thực nghiệm, sử dụng ứng dụng ngoài thực tế, đi qua một số tuyến đường cụ thể trong thành phố Hồ Chí Minh. Sau đó, nhóm sẽ thống kê và đưa ra đánh giá về độ sai lệch của lộ trình do hệ thống định vị so với lộ trình thực tế.

* + 1. ***Kiểm thử về dịch bệnh và thiên tai.***

Nhóm thực hiện kiểm thử độ chính xác của thông báo về dịch bệnh và thiên tai bằng thực nghiệm cụ thể là mỗi khi có thông báo về thiên tai, dịch bệnh hiện trên ứng dụng thì tiến hành xác thực qua các kênh thông tin truyền thông về dịch bệnh và thiên tai cụ thể là các trang:

* + who.int/en/
  + gdacs.org
  + Công cụ tìm kiếm Google
  1. ***Quá trình kiểm thử.***
     1. ***Kiểm thử các khối chức năng của hệ thống.***
        1. ***Admin website***
* Đăng nhập

|  |  |  |
| --- | --- | --- |
| Người quản trị đăng nhập | Xử lý | Kết quả |
| Nhập đúng | Nhập chính xác email và mật khẩu. | Đăng nhập thành công |
| Nhập sai | Nhập sai email hoặc mật khẩu | Báo lỗi (Yêu cầu đăng nhập lại) |

* Xem thông tin cá nhân.

|  |  |  |
| --- | --- | --- |
| Chọn xem thông tin cá nhân | Xử lý | Kết quả |
| Thực hiện đúng | Click Dashboard -> Chọn tài khoản bạn muốn xem | Hiển thị toàn bộ thông tin về tài khoản. |
| Thực hiện sai | Chọn tài khoản đã bị xóa. | Không hiển thị gì. |

* Tạo tài khoản mới

|  |  |  |
| --- | --- | --- |
| Tạo tài khoản mới | Xử lý | Kết quả |
| Nhập đúng | Click Form -> Nhập đầy đủ thông tin theo mẫu. | Tài khoản mới được tạo. |
| Nhập sai | Tạo tài khoản đã tồn tại hoặc tạo tài khoản không đầy đủ thông tin bắt buộc. | Không tạo được tài khoản mới, |

* Xem thống kê công ty

|  |  |
| --- | --- |
| Bar Chart | Pie Chart |
| Hiển thị toàn bộ chi phí của một hoặc tất các nhân viên trong công ty. | Hiển thị thị phần về chi phí đi lại của một hoặc tất cả nhân viên trong công ty (dựa trên phương tiện). |

* + - 1. ***Android App***
* Đăng nhập

|  |  |  |  |
| --- | --- | --- | --- |
| **Mô tả testcase** | **Cách thực hiện** | **Kết quả mong muốn** | **Kết quả** |
| Đăng nhập với tài khoản user hợp lệ | - Nhập email và password của tài khoản user hợp lệ.  - Nhấn “Login” | User login thành công và chuyển đến màn hình chính của ứng dụng | Đạt |
| Đăng nhập với tài khoản user không hợp lệ | - Nhập email sai  - Nhập password đúng  - Nhấn “Login”  - Nhập lại email đúng  - Nhập lại password sai  - Nhấn “Login” | Cả 2 lần login đều hiển thị thông báo cho user là email hoặc password không đúng | Đạt |

* Xem thông tin thiên tai và dịch bệnh

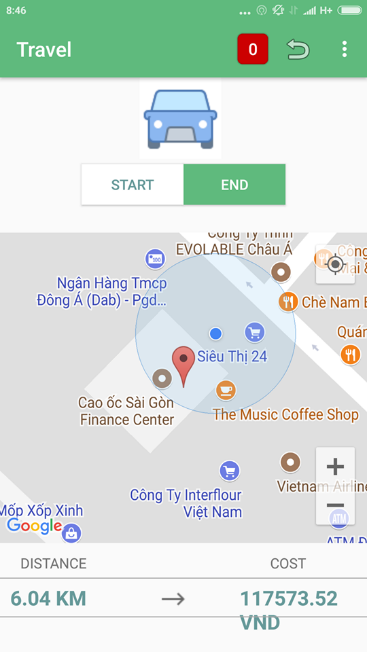
|  |  |  |  |
| --- | --- | --- | --- |
| **Mô tả testcase** | **Cách thực hiện** | **Kết quả mong muốn** | **Kết quả** |
| Xem chi tiết thiên tai và dịch bệnh | - Từ màn hình chính, nhấn “News”  - Ở màn hình tiếp theo, chọn dịch bệnh hoặc thiên tai cần xem | - Ở màn hình chi tiết thiên tai, hiển thị đầy đủ thông tin và vị trí của thiên tai  - Ở màn hình chi tiết dịch bệnh, hiển thị đầy đủ thông tin và cách phòng tránh | Đạt |

* Xem thống kê cá nhân

|  |  |  |  |
| --- | --- | --- | --- |
| **Mô tả testcase** | **Cách thực hiện** | **Kết quả mong muốn** | **Kết quả** |
| Xem thống kê về đường đi và chi phí dựa trên loại phương tiện | - Từ màn hình chính, nhấn “Statistics”  - Chọn khoảng thời gian mong muốn và loại Chart  - Nhấn “View stats” | - Chart hiển thị đúng và đủ thông tin dựa trên database | Đạt |

* + 1. ***Kiểm thử về hiệu năng của hệ thống (độ hao pin của thiết bị).***

Nhóm sử dụng điện thoại Xiaomi Redmi 4 với dung lượng pin 4100mAhđể đi từ địa chỉ 184/44 Phạm Văn Hai đến địa chỉ số 9 Đinh Tiên Hoàng. Kết quả được ghi nhận như sau

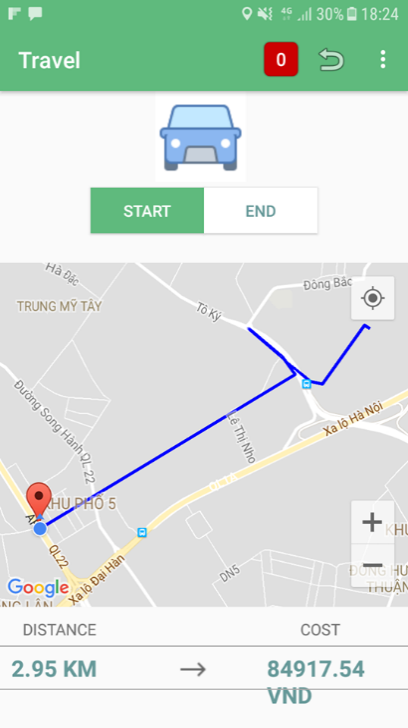


Hình 4.1 1 Vị trí của mình hiện tại hiển thị trên ứng dụng.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Khoảng cách (Km) | Thời gian (phút) | Dung lượng pin bắt đầu (%) | Dung lượng pin kết thúc (%) | Năng lượng tiêu hao (mAh) | Dung lượng 3G sử dụng (MB) |
| 6.04 | 35 | 100 | 92 | 328 | 8 |

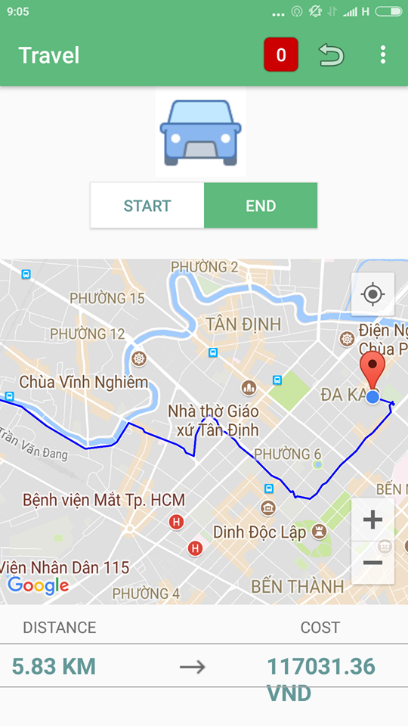
Vậy App có thể hoạt động liên tục với khoảng thời gian tối đa là 7 giờ, dung lượng 3G cần thiết là 100MB trong 1 ngày. Theo nhóm đây là một kết quả chấp nhận được.

* + 1. ***Kiểm thử về lộ trình.***
       1. ***Lộ trình 1.***



Hình 4.1 2: Hình ảnh ứng dụng sau khi di chuyển được một quãng

* + Mô tả lộ trình: Từ công viên phần mềm Quang Trung đến ngã tư An Sương bằng xe bus.
  + Thời tiết: Trời quang đãng ít mây.
  + Khoảng cách theo ước lượng của Google Map: 3.02 Km
  + Khoảng cách đo được của App: 2.95 Km
  + Sai số tương đối: 2.32%
    - 1. ***Lộ trình 2.***

******

Hình 4.1 3: Thông tin lộ trình được hiển thị.

* Mô tả lộ trình: Từ 184/44 Phạm Văn Hai đến sân vận động Hoa Lư.
* Thời tiết: Trời âm u, nhiều mây, mưa ít.
* Khoảng cách theo ước lượng của Google Map: 6.08 Km
* Khoảng cách đo được của App: 5.83 Km
* Sai số tương đối: 4.11%
  + 1. ***Kiểm thử về thông báo dịch bệnh, thiên tai.***

Nhóm chạy chương trình RSS Feed parser và Web crawler để lấy dữ liệu trong 3 ngày 22/12, 24/12, 25/12, mỗi ngày 8 giờ.

* + - 1. ***Thông báo dịch bệnh.***

******

Hình 4.1 4: Thông báo dịch bệnh từ WHO trả về định dạng RSS.

Vào ngày 22/12, WHO có public thông tin về 3 loại dịch bệnh mới. Một thông báo được gởi đến tất cả user.

* + - 1. ***Thông báo thiên tai.***

Trong 3 ngày trên, các thông tin mới nhất về bão TEMBIN-17 được GDACS cập nhật cũng được gởi thông báo đến tất cả user.

1. **Đánh giá hệ thống.**
   1. ***Ưu điểm.***

* Hoàn chỉnh các chức năng cơ bản, đạt được các mục tiêu ban đầu đề ra.
* Xây dựng được hệ thống với giao diện thân thiện, dễ sử dụng.
* Dung lượng 3G mà ứng dụng di động sử dụng có thể chấp nhận được.
* Android App có thể hoạt động ngay cả khi mất kết nối internet.
* Giao diện website tương thích tốt trên nhiều trình duyệt khác nhau.
* Cảnh báo thiên tai dịch bệnh gần như Realtime, có thể xác định vị trí của nhân viên trong vùng thiên tai.
  1. ***Nhược điểm.***
* Khả năng tương thích trên nhiều thiết bị di động khác nhau của ứng dụng di động chưa tốt.
* Phương pháp xử lý sai số tín hiệu GPS, sai lệch lộ trình chưa tốt.
* Mức độ tiêu hao pin của ứng dụng di động còn cao.
* Sử dụng nhiều dịch vụ của bên thứ ba.

1. **Kết luận.**
2. **Kết luận.**

Về cơ bản, nhóm đã hoàn thành được mục tiêu ban đầu đặt ra là xây dựng hệ thống quản lý chi phí đi lại của nhân viên bán hàng trong công ty với giao diện thân thiện, dễ dùng, bao gồm 2 thành phần là website và ứng dụng di động.

Tuy nhiên, hệ thống của nhóm vẫn còn một số nhược điểm như đã trình bày ở phần trước. Nhưng, hệ thống cũng đã đáp ứng được những tính năng cơ bản nhất là hỗ trợ công ty quản lý chi phí đi lại của nhân viên một cách hệ thống, tự động.

Hệ thống hiện tại có thể được áp dụng cho các công ty vừa và nhỏ (do hạn chế của các dịch vụ của Google như đã trình bày ở các phần trước). Muốn được áp dụng cho những công ty có quy mô nhân sự lớn, hệ thống cần có nhiều sự cải tiến, mở rộng.

1. **Hướng phát triển.**

Để ứng dụng có thể hoạt động hiệu quả hơn cần phải

* Cải thiện ứng dụng.
* Cải thiện về mức sai lệch trong định vị GPS và sử dụng bản đồ số Google Maps.
* Cải thiện về hiệu năng hệ thống về mức độ sử dụng dữ liệu và độ hao pin.
* Thông báo cụ thể tới những thiết bị nằm trong khu vực xảy ra thiên tai và dịch bệnh.
* Hạn chế sử dụng dịch vụ bên thứ ba.
* Việc sử dụng nhiều dịch vụ của bên thứ ba (Google Maps, Google Maps Road API, Firebase Realtime Database, Firebase Authentication, Firebase Cloud Storage, Firebase Hosting) gây ra sự lệ thuộc cho hệ thống, dẫn đến trường hợp hệ thống không thể đáp ứng nhu cầu của công ty và nhân viên khi các dịch vụ đó gặp vấn đề. Bên cạnh đó, các dịch vụ này còn thu phí khi lưu lượng sử dụng vượt quá giới hạn được miễn phí. Do đó, cần có kế hoạch để thay thế bằng các dịch vụ hoàn toàn miễn phí hoặc tự phát triển, bổ sung những dịch vụ này vào hệ thống tùy theo nhu cầu và quy mô nhân sự của mỗi công ty.
* Thêm các chức năng hỗ trợ người dùng.
* Việc sử dụng dịch vụ chỉ hỗ trợ người dùng đối với hệ điều hành Android, cần mở rộng ứng dụng thêm cho hệ điều hành iOS do các điện thoại thông minh của Apple đang được sử dụng rộng rãi
* Thêm một số tính năng đã đưa ra như là nhận diện giọng nói, điều khiển bằng giọng nói,…

# **TÀI LIỆU THAM KHẢO**

|  |  |
| --- | --- |
| [1] | "GPS Overview," [Online]. Available: https://www.gps.gov/systems/gps/. |
| [2] | "Android (Hệ Điều Hành)," [Online]. Available: https://vi.wikipedia.org/wiki/Android\_(hệ\_điều\_hành). |
| [3] | Google, "Dashboards | Android Developers," Google, 11 Dec 2017. [Online]. Available: https://developer.android.com/about/dashboards/index.html. |
| [4] | "Google Maps," [Online]. Available: https://vi.wikipedia.org/wiki/Google\_Maps. |
| [5] | "Pricing And Plans," [Online]. Available: https://developers.google.com/maps/pricing-and-plans/#details. [Accessed 05 Dec 2017]. |
| [6] | A. Danilo, "Introduction to the Google Maps Roads API," [Online]. Available: https://developers.google.com/maps/documentation/roads/intro?hl=vi#overview. [Accessed 07 Dec 2017]. |
| [7] | "Firebase Realtime Database," [Online]. Available: https://firebase.google.com/products/realtime-database/. [Accessed 08 Dec 2017]. |
| [8] | "FireBase Authentication," [Online]. Available: https://firebase.google.com/products/auth/. [Accessed 12 Dec 2017]. |
| [9] | "Firebase Hosting," [Online]. Available: https://firebase.google.com/products/hosting/. [Accessed 12 Dec 2017]. |
| [10] | "RSS (Định dạng tập tin)," [Online]. Available: https://vi.wikipedia.org/wiki/RSS\_(định\_dạng\_tập\_tin). [Accessed 13 Dec 2017]. |
| [11] | "Web Clrawler," [Online]. Available: https://en.wikipedia.org/wiki/Web\_crawler. [Accessed 13 Dec 2017]. |