**TRƯỜNG ĐẠI HỌC BÁCH KHOA - ĐH QUỐC GIA TP. HỒ CHÍ MINH**

**KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**

****

**BÁO CÁO BÀI TẬP LỚN 1**

**Xây dựng ứng dụng Internet of Things (IoT) sử dụng thiết bị di động và các dịch vụ trên Internet**

**MÔN HỌC: MẠNG MÁY TÍNH**

**GVHD: PGS. TS. Phạm Trần Vũ**

**ThS. Nguyễn Hồng Nam**

**Sinh viên thực hiện:**

1. **Văn Minh Hào 1510901**
2. **Lê Phước Lộc 1511842**
3. **Nguyễn Thành Phương 1512591**

# Giới thiệu đề tài

## Tính cấp thiết của đề tài

Hiện nay, với sự phát triển của khoa học kĩ thuật đặc biệt là điện tử và công nghệ thông tin và sự mở rộng của Internet, việc ứng dụng công nghệ vào đời sống ngày càng trở nên phổ biến. Dễ thấy nhất là sự phát triển của smartphone, tablet, … và không thể không nhắc đến các hệ thống thông minh Internet of Things trong đời sống con người. Chúng ngày càng trở nên phổ biến, mang nhiều ứng dụng hữu ích và thực tế không chỉ cho công nghiệp, dịch vụ mà ngay trong đời sống hằng ngày.

Từ những yếu tố trên, nhóm chúng tôi đã có ý tưởng phát triển một hệ thống IoT dựa trên nền tảng Android có khả năng định vị, tính toán và cung cấp thông tin về tình trạng giao thông.

## Giới thiệu về ứng dụng

### Mục đích của ứng dụng

Cho phép người dùng có khả năng kiểm tra tình hình giao thông tại tuyến đường mình đang muốn đi. Nói cách khác, trước khi tham gia giao thông, người dùng có thể xác định tình trạng tắc đường để có thể lựa chọn tuyến đường phù hợp.

### Các thành phần của ứng dụng

* Client application được xây dựng trên Android
* Gateway
* CloudMQTT
* Analytic Application

### Các công nghệ được sử dụng

Các giao thức UDP, TCP, MQTT.

# Phân tích và mô tả ứng dụng

## Phân tích yêu cầu đặc điểm của ứng dụng

## Chức năng cụ thể của ứng dụng

### Android Application

Đây là ứng dụng tương tác trực tiếp với người dùng, bao gồm các chức năng sau:

* Các tác vụ cơ bản của Google Maps: zoom, rotate map, xác định vị trí người dùng, di chuyển đến các địa điểm, thành phố được nhập vào thanh tìm kiếm.
* Gửi data lên Gateway: vì đây là ứng dụng dự báo tắc đường, muốn có sự chính xác và kịp thời nhất, app có chức năng gửi vị trí và thời điểm hiện tại đến gateway. Ngoài ra người dùng có thể cấu hình khoảng thời gian giữa mỗi lần gửi.
* Nhận dự báo từ Analytic Application: từ một vị trí đã chọn sẵn trên Map, app sẽ gửi request đến Analytic Application và nhận về kết quả thông báo tắc đường tại địa điểm đã chọn. Request bao gồm địa điểm được chọn và thời điểm đưa ra request.

### Gateway

Điểm thu thập dữ liệu. Các clients sẽ gửi dữ liệu đến Gateway trực tiếp quản lý thông qua giao thức UDP. Dữ liệu sau đó được Gateway tải lên Cloud thông qua giao thức MQTT.

### Cloud MQTT

Nhận dữ liệu từ các Gateway gửi về.

### Analytic Application

* Nhận yêu cầu và phản hồi kết quả từ ứng dụng Clients thông qua giao thức TCP.
* Phân tích các yêu cầu, phân tích dữ liệu được lấy từ Cloudmqtt sau đó xử lý và trả về kết quả cho ứng dụng.

## Cách thức hoạt động của ứng dụng

* Người dùng có thể gửi data liên tục tùy theo ý muốn, nói cách khác, chỉ với một thao tác chạm màn hình, người dùng có thể gửi data về địa điểm hiện tại của mình lên server. Và cũng chỉ với thao tác đó, người dùng có thể dừng hoạt động gửi dữ liệu của mình.
* Khi gateway nhận được data từ client, gateway sẽ tiến hành gửi dữ liệu lên cloud và cloud sẽ tiếp tục chuyển dữ liệu về database được tích hợp ở Analytic Application
* Khi Analytic Application nhận được request về tình trạng tắc đường tại một địa điểm, phần mềm này sẽ tiến hành query từ database những data có liên quan. Trước hết phần mềm sẽ xác định những thiết bị ở xung quanh điểm cần dự báo trong một khoảng bán kính nhất định. Tiếp theo phần mềm sẽ truy ngược lại các vị trí của nhóm thiết bị này trước đó một khoảng thời gian. Có được khoảng cách (được tính từ các vị trí) cùng với thời gian (Hiệu các thời điểm) ta sẽ lấy được vận tốc trung bình của người dùng khi đi trên đoạn đường đó. Dựa vào ngưỡng đặt ra cho vận tốc, Analytic Application sẽ trả về dự báo cho người dùng. Ví dụ, vận tốc trung bình thấp hơn ngưỡng thì thông báo tắc đường và ngược lại.

# Thiết kế chi tiết ứng dụng

## Thiết kế kiến trúc hệ thống

## Thiết kế các bộ giao thức cho ứng dụng

## Sơ đồ class

## Đặc tả chi tiết từng thành phần trong ứng dụng

### Class trong Android application (Client)

### Class trong Gateway

* **Receiver**: là class nhằm tạo ra đối tượng có vai trò nhận dữ liệu từ Client thông qua giao thức truyền UDP, giao thức trao đổi dữ liệu được mô tả chi tiết ở mục 4, dữ liệu nhận từ Client sẽ được đẩy vào DataQueue. Vì việc truyền và gửi dữ liệu sẽ hoạt động tách biệt và cùng lúc với nhau nên class Receiver sẽ thừa kế class Thread.
* **DataQueue**: là một Queue hiện thực bởi LinkedList nhằm mục đích chứa các dữ liệu gửi lên từ Client trước khi được gửi lên Cloud.
* **Sende**r: là class nhằm tạo ra đối tượng có vai trò gửi dữ liệu từ Gateway lên Cloud. Dữ liệu sẽ được lấy ra từ DataQueue sau đó đưa lên Cloudmqtt thông qua giao thức MQTT. Nhóm sử dụng thư viện Paho để hiện thực giao thức MQTT này. Tương tự như Receiver, Sender cũng được thừa kế từ class Thread để có thể chạy đa luồng với Receiver.
* **CloudMQTT**: là class nhằm tạo đối tượng kết nối Gateway với Cloudmqtt

### Cloud MQTT và phương thức kết nối

Cloudmqtt là một server cung cấp nền tảng truyền tin cho các ứng dụng IoT thông qua giao thức MQTT, chi tiết: cloudmqtt.com

### Class trong Analytic Application

* **Server**: chứa hàm main để chạy hệ thống.
* **Database**: chứa các phương thức để kết nối với Database.
* **CloudMQTT**: chứa các phương thức kết nồi với Cloud để nhận dữ liệu từ các Gateway để lưu trữ và Database.
* **QueueRequests**: là một hàng đợi để chứa các Socket mang theo Request được gửi từ các Client.
* **ServiceThread**: chứa các phương thức để lấy các Socket từ **QueueRequests** rồi đẩy vào Thread để xử lý.
* **Query**: chứa các phương thức sử dựng thông tin Request của các Socket để truy vấn dữ liệu từ Database.
* **Computing**: chứa các phương thức để xử lý Request và tính toán dữ liệu truy vấn được từ class **Query** rồi trả về kết quả.
* **GPS**: cấu trúc dữ liệu để tính toán, lưu trữ và làm giao thức truyền dữ liệu trong hệ thống.
* **Dict**: cấu trúc dữ liệu phục vụ quá trình tính toán của class **Computing**.

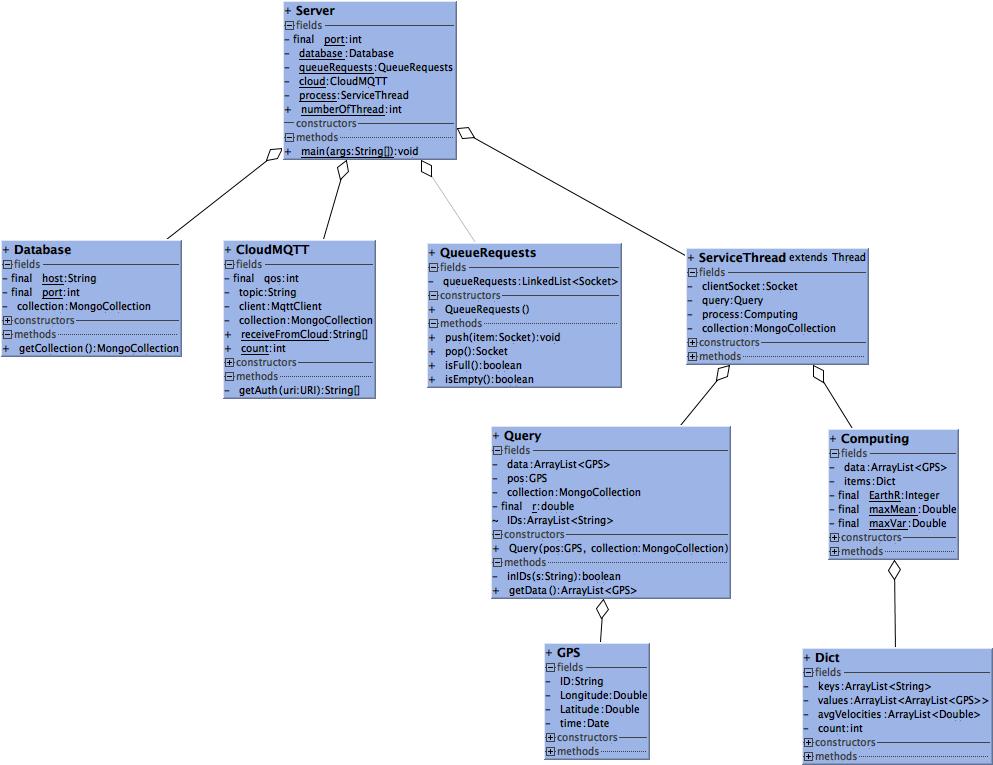


Figure : Class diagram của Analytic Application

## Phương án triển khai ứng dụng

### Ngôn ngữ sử dụng

* Hiện thực Client application: ứng dụng nền tàng Android sử dụng ngôn ngữ Java
* Hiện thực Gateway, Analytic Application: Ngôn ngữ Java

### Cloudmqtt

Sử dụng thư viện Paho cho Java để kết nối đến Cloud bằng giao thức MQTT.

### Cơ sở dữ liệu

Sử dụng hệ cơ sở dữ liệu MongoDB để lưu trữ dữ liệu dạng JSON tại Analytic Application.

# Đánh giá kết quả đạt được

# Hướng dẫn sử dụng

## Các nút chức năng trên ứng dụng

* Thanh tìm kiếm: nằm phía trên cùng của app, bao gồm phần nhập chữ (TextView) và nút (Button). Người dùng sẽ nhập vào địa điểm mình mong muốn tìm kiếm (Ví dụ: London, New York, Hanoi, v.v) và ấn nút Go, camera của bản đồ sẽ di chuyển đến địa điểm được chọn.
* Clear button: khi trên bản đồ có quá nhiều marker, việc nhấn nút Clear sẽ giúp người dùng xóa bỏ hết tất cả các marker đánh dấu đó.
* Sat button: chuyển đổi view nhìn bản đồ, từ view bình thường sang view vệ tinh (satellite)
* Mark button: gắn marker vào vị trí hiện tại của người dùng.

## Gửi và nhận dữ liệu:

* Gửi dữ liệu: khi muốn gởi dữ liệu lên gateway, người dùng chỉ việc bấm nút nằm ở giữa bên phải của màn hình. Muốn dừng quá trình gửi lại, người dùng sẽ bấm vào nút đó một lần nữa. Icon của nút bấm sẽ thay đổi tùy theo trạng thái đang gửi hoặc đã ngừng gửi.
* Request dự báo tắc đường: khi người dùng chạm vào một vị trí trên bản đồ, một marker sẽ được đặt ở vị trí đó. Nếu muốn nhận dự báo cho vị trí này, ta chỉ cần chạm vào marker đó, sẽ có một prompt hiện ra và yêu cầu xác nhận việc nhận request. Nếu chọn Yes, app sẽ tự động gửi yêu cầu tới Analytic Application và hiển thị dự báo lên màn hình.