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

Contents

1 Giới thiệu đề tài

1.1 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.

1.2 Giới thiệu về ứng dụng

1.2.1 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.

1.2.2 Các thành phần của ứng dụng

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

1.2.3 Các công nghệ được sử dụng

Các giao thức UDP, TCP, MQTT.

2 Phân tích và mô tả ứng dụng

2.1 Phân tích yêu cầu đặc điểm của ứng dụng

Từ kiến trúc tổng quát của mô hình IOT ứng dụng sẽ được chia thành 3 module chính:

- Clients: gồm các cảm biến GPS gắn trên các phương tiện nhằm thu thập dữ liệu GPS thời gian thực để gửi lên Server và ứng dụng Android trên các thiệt bị điện thoại dành cho người dùng sử dụng dịch vụ của hệ thống.
- Gateway: là các điểm thu thập dữ liệu từ các cảm biến GPS để gửi lên Server.
- Server (Analytic Application): nơi lưu trữ dữ liệu GPS, nhận các yêu cầu từ ứng dụng Android rồi xử lý dữ liệu trên database để trả kết quả về cho người dùng thông qua ứng dụng Android.

Về mặt giao tiếp giữa các modul:

- Clients (các cảm biến GPS) sẽ giao tiếp với Gateway thông qua giao thức truyền UDP.
- Gateway sẽ giao tiếp với Server thông qua hạ tầng dịch vụ của Cloud MQTT.

- Server sẽ giao tiếp với các Clients (ứng dụng Android) thông qua giao thức truyền TCP.
  Về trải nghiệm của người dùng (user experience):

- Giao diện thân thiện, dễ sử dụng.
- Thực hiện đầy đủ, chính xác các chức năng.
- Khi có lỗi, phải tiến hành xử lí và thông báo cho người dùng (hiển thị message …)

2.2 Chức năng cụ thể của ứng dụng

2.2.1 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.

2.2.2 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.

2.2.3 Cloud MQTT

Nhận dữ liệu từ các Gateway gửi về.

2.2.4 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.

2.3 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.

3 Thiết kế chi tiết ứng dụng

3.1 Thiết kế kiến trúc hệ thống

3.1.1 Sequence Diagram

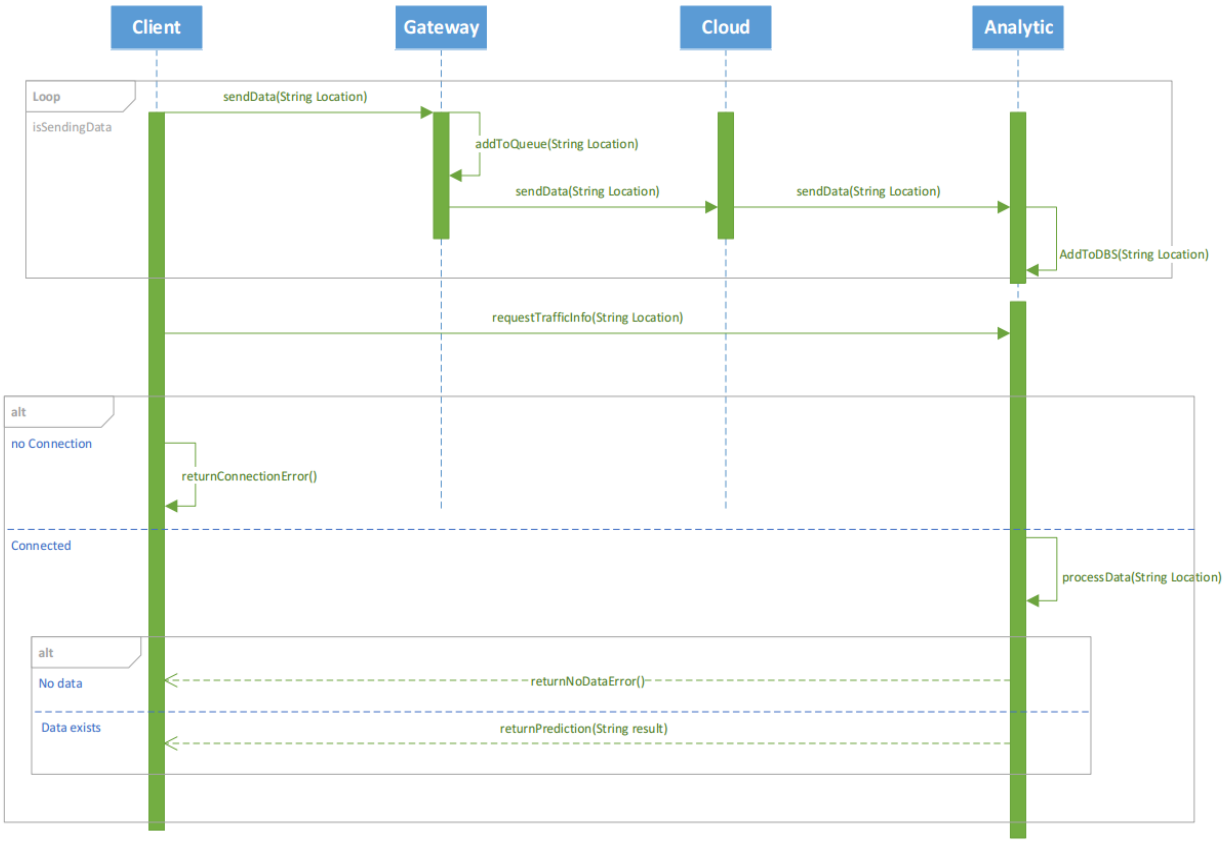

Diagram 1. Sequence Diagram

3.1.2 Architecture Diagram

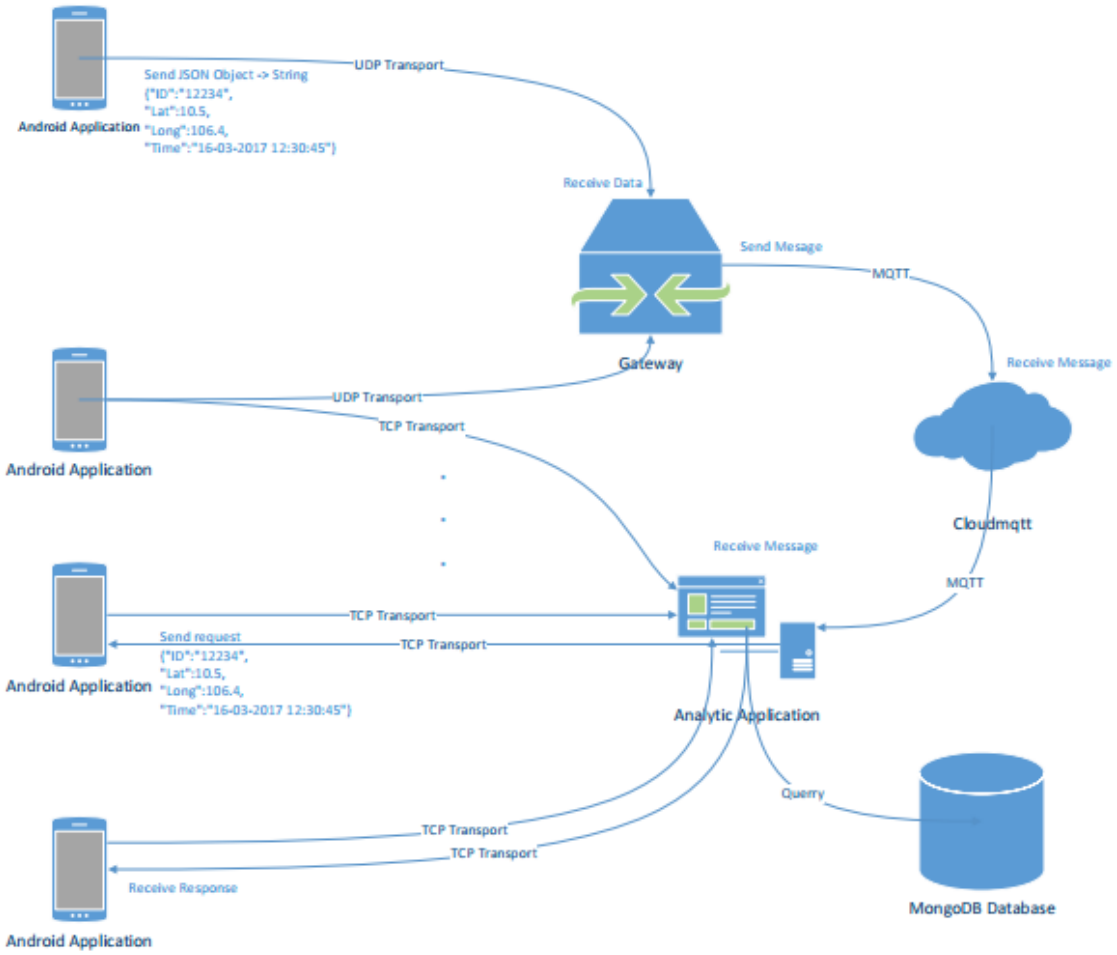

Diagram 2. Architecture Diagram

3.2 Thiết kế các bộ giao thức cho ứng dụng

3.2.1 Giao thức giữa Android Application và Gateway

Giao thức tầng transport: UDP

Dữ liệu được khởi tạo dưới dạng JSON object, bao gồm các trường:

- ID: (Type: String) là ID của thiết bị, nhằm phân biệt với các thiết bị khác.
- Lat (latitude): (Type: Double) là vĩ độ của địa điểm mà client chọn.
- Long (longitude): (Type: Double) là kinh độ của địa điểm mà client chọn.
- Time: (Type: Date, format: dd/MM/yyyy hh:mm:ss) là thời điểm tạo ra object

```
JSONObject obj = new JSONObject();
try {
    obj.put("ID", android_id);
    obj.put("Lat", myLatitude);
    obj.put("Long", myLongitude);
    obj.put("Time",currentTime);
}
catch (JSONException e){
    e.printStackTrace();
}
```

Chuyển JSON object về String bằng hàm toString()

```
obj.toString();
```

Sau đó chuỗi String sẽ được gửi qua UDP đến Gateway

```
String message = obj.toString().getBytes();

DatagramPacket dpSend = new
DatagramPacket(message.getBytes(),message.getBytes().length, ia, 1024);
socket.send(dpSend);
```

3.2.2 Giao thức giữa Gateway, Analytic Application với Cloudmqtt

Dữ liệu được truyền qua hạ tầng Cloud bằng messages là string có dạng JSON Object với cấu trúc:

- ID (Type: String): ID của client nhằm phân biệt với các client khác.
- Lat (Type: double): Vĩ độ vị trí của client tại thời điểm gửi dữ liệu.
- Long (Type: double): Kinh độ vị trí của client tại thời điểm gửi dữ liệu.
- Time (Type: Date, format: dd/MM/yyyy hh:mm:ss): thời điểm dữ liệu được gửi đi.

```
while (!Gateway.DataQueue.isEmpty()) {
    dataOut = Gateway.DataQueue.poll();
    Gateway.SendSocket.sendMessage(dataOut);
}
```

Hiện thực hàm sendMessage()

```
public void sendMessage(String payload) throws MqttException {
    MqttMessage message = new MqttMessage(payload.getBytes());
    message.setQos(qos);
    client.publish(topic, message); // Blocking publish
    System.out.println("Message sent to Cloud: "+message);
}
```

Dữ liệu sau đó vẫn ở dạng String messages từ Cloud gửi đến Server (Analytic Application) rồi được parse về lại JSON Object với cấu trúc tương tự.

3.2.3 Giao thức giữa Android Application và Analytic Application

Tương tự như với Gateway, Android App sẽ gửi request đến Analytic App với cùng cấu trúc dữ liệu, nhưng với kiểu kết nối TCP.

```
os = new BufferedWriter(new OutputStreamWriter(socketOfClient.getOutputStream()));
is = new BufferedReader(new InputStreamReader(socketOfClient.getInputStream()));
os.write(message);
os.newLine();
os.flush();
```

Android App sau đó sẽ đợi response từ Analytic App trả lại dự báo. Kiểu dữ liệu nhận về là String.

```
//check for response in 5 seconds
String responseLine;

if ((responseLine = is.readLine()) != null) {
    if (responseLine.equals("")) {
        MapsActivity.predictionStr = "Did not receive prediction";
    } else {
        MapsActivity.predictionStr = responseLine;
    }
} else {
    MapsActivity.predictionStr = "Did not receive prediction";
}
```

Như vậy, Android App sẽ gửi JSON object đã chuyển về dạng String và đợi String phản hồi. Trong khi đó Analytic App sẽ đợi request dưới dạng String và phản hồi cũng với dạng String.

3.3 Sơ đồ class

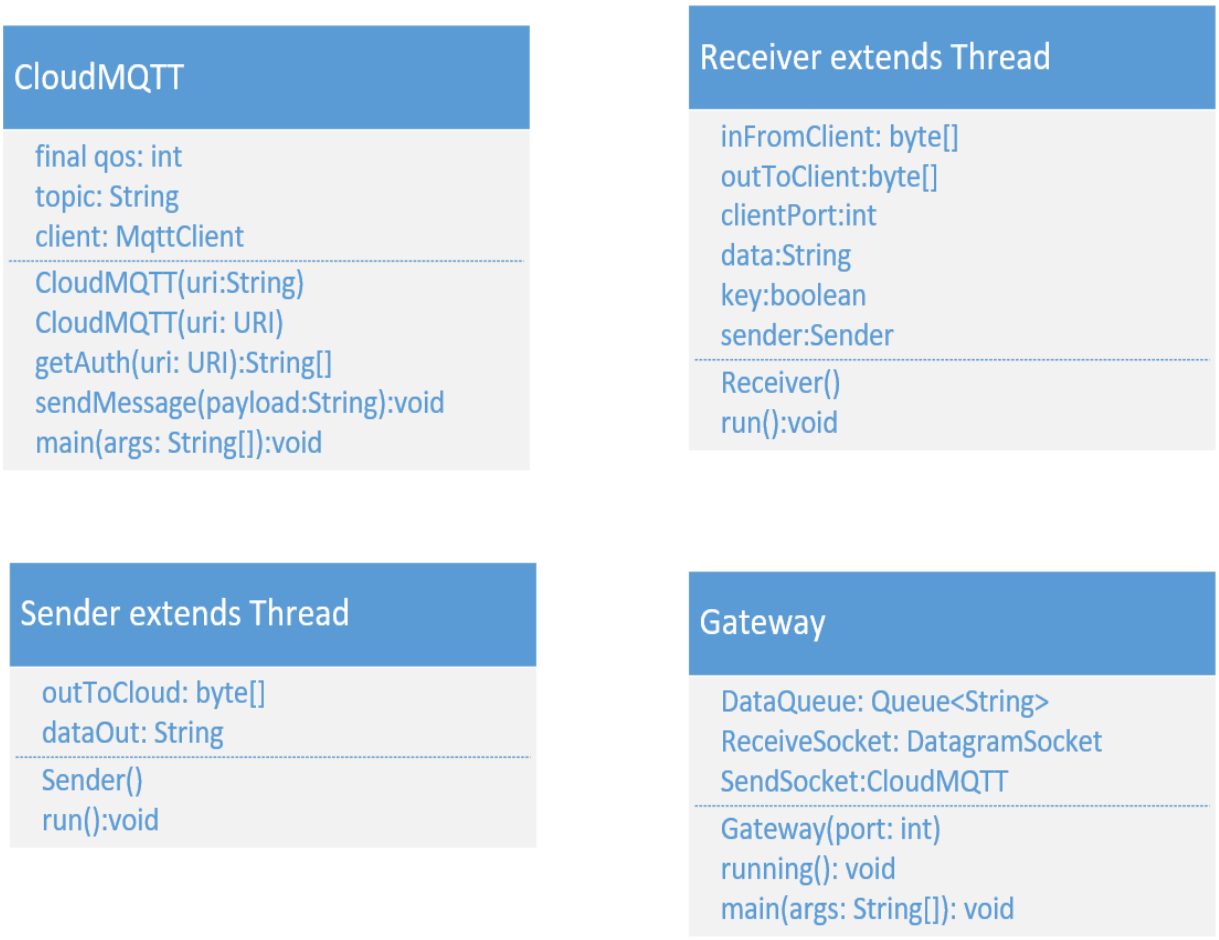

Diagram 3. Class diagram cho Gateway

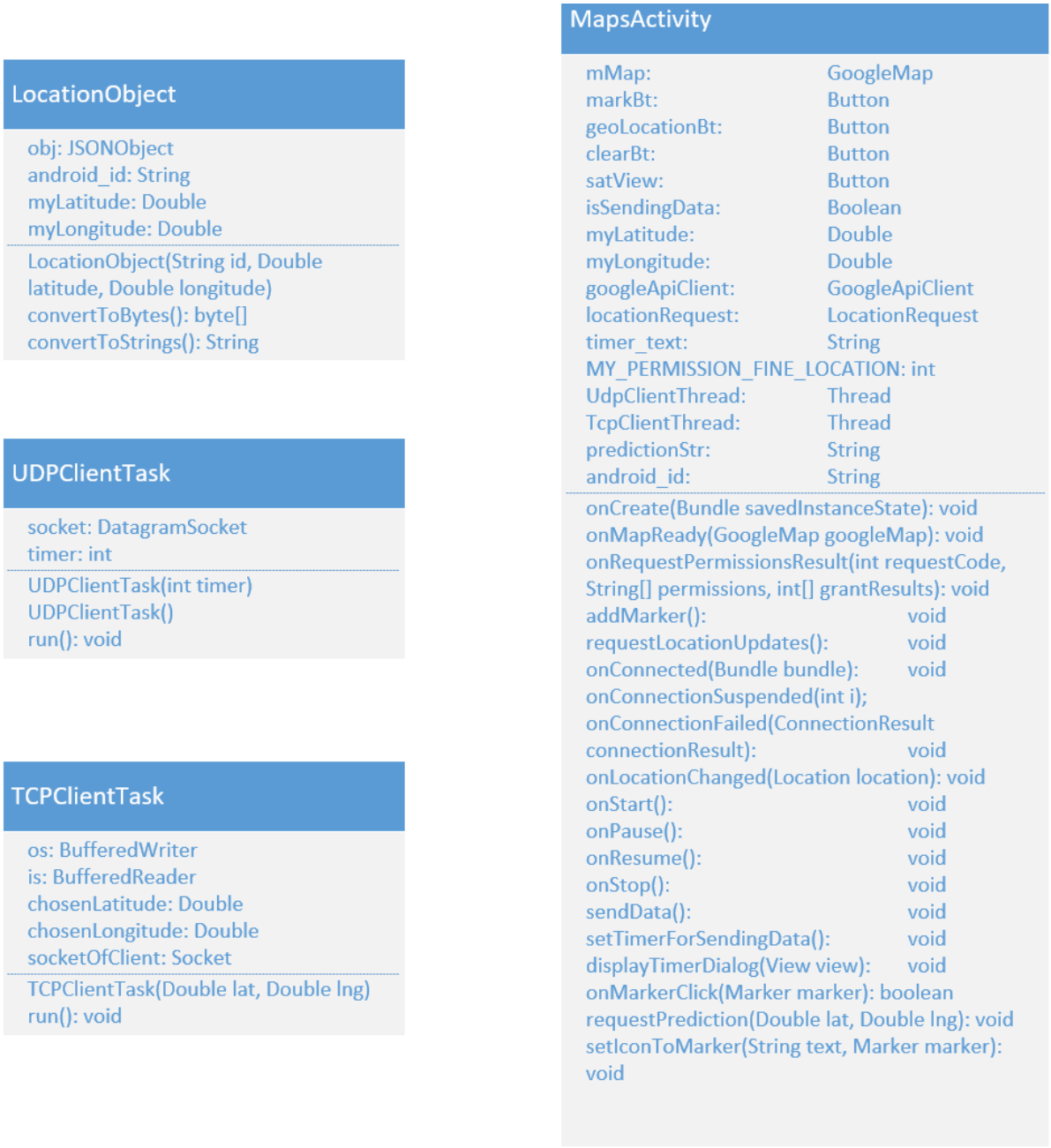

Diagram 4. Class diagram cho Analytic Application

Diagram 5. Class diagram cho Analytic Application

3.4 Đặc tả chi tiết từng thành phần trong ứng dụng

3.4.1 Class trong Android application (Client)

Dưới đây là danh sách các class trong Android Application:

- **MapsActivity:** class chính của Application, hiện thực chương trình và các giao diện UI trên thread chính của Application.
- **LocationObject:** class hiện thực kiểu dữ liệu được sử dụng trong ứng dụng, bao gồm JSON object và các tác vụ chuyển object về chuỗi String và chuỗi Byte
- **TCPClientTask:** class hiện thực thread chạy kết nối TCP với Analytic Application.
- **UDPClientTask:** class hiện thực thread chạy kết nối UDP với Gateway.

3.4.2 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.
- **Sender:** 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

3.4.3 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

3.4.4 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**.

3.4.5 Thuật toán xử lý dữ liệu

Trong xử lý dữ liệu, dữ liệu truy vấn được là dữ liệu GPS nên cần tính toán thành dữ liệu vận tốc. Sau đó, nhóm sử dụng các phương pháp thống kê cơ bản để tính toán. Cụ thể là 2 công thức tính trung bình (Mean) và phương sai (Variance).

3.4.5.1 Thuật toán tính vận tốc

$$A = sin(\frac{\Delta lat}{2})^2 + sin(\frac{\Delta long}{2})^2 \times cos(lat_1) \times cos(lat_2)$$

$$C = 2 \times arctan2(\sqrt{A}, \sqrt{1-A})$$

$$Distance = C \times r$$

$$Interval\ time = |Time_1 - Time_2|$$

$$Velocity = \frac{Distance}{Interval\ time}$$

- Δlat : trị tuyệt đối của hiệu 2 vĩ độ.
- $\Delta long$: trị tuyệt đối của hiệu 2 kinh độ.
- lat_1, lat_2 : vĩ độ của 2 vị trí GPS.
- r: bán kính Trái Đất = 6371 km.

3.4.5.2 Thuật toán tính vận tốc trung bình

$$Mean = \frac{1}{N}\sum_{i=1}^{N} Velocity_i \ (N \text{ là số lượng dữ liệu})$$

- Tính vận tốc trung bình của từng phương tiện.
- Tính vận tốc trung bình của các phương tiện.

3.4.5.3 Thuật toán tính phương sai

$$Variance = \frac{1}{N}\sum_{i=1}^{N}(Velocity_i - Mean)^2$$

- Tính phương sai trên dữ liệu là vận tốc trung bình của các phương tiện.

3.4.5.4 Thuật toán đưa ra cảnh báo

Thuật toán này rất đơn giản, chỉ cần so sánh vận tốc trung bình của các phương tiện và phương sai tính được với các ngưỡng cho trước và đưa ra kết luận.

If: Mean(Velocity) <= P và Variance(Velocity) <= Q.

Then: tắc đường.

Else: không tắc đường.

Ngưỡng P và Q do nhóm tự đặt ra theo cảm tính. Thật ra, các ngưỡng này có thể ước lượng được bằng các phương pháp thống kê nhưng nhóm không có dữ liệu để tính toán và ước lượng.

3.5 Phương án triển khai ứng dụng

3.5.1 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

3.5.2 Cloudmqtt

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

3.5.3 Cơ sở dữ liệu

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

4 Đánh giá kết quả đạt được

Nhìn chung, nhóm đã đạt được những mục tiêu cơ bản đã đề ra, bao gồm:

- Thiết kế ứng dụng Android với đủ các tính năng cần thiết như đã đề cập ở trên.
- Hoàn thiện các giao thức và các thao tác gửi nhận dữ liệu.
- Thực hiện xử lý được dữ liệu trên Analytic Application và trả về phía người dùng request.

Những yếu tố cần cải thiện:

- Về dữ liệu: phụ thuộc chặt chẽ vào số lượng người đi đường thực hiện việc gửi dữ liệu. Nếu số người đi đường quá ít, dự báo có thể sẽ kém chính xác. Dữ liệu cũng phụ thuộc vào hành vi của người đi đường, ví dụ trên đoạn đường không tắc nghẽn nhưng người dùng hay dừng nhiều (nghe điện thoại hoặc vào quán nhưng vẫn còn gửi dữ liệu), kết quả tính toán cũng bị ảnh hưởng.
- Dự báo chỉ dùng cho một địa điểm nhất định, thực tế trên một con đường sẽ có đoạn tắc và không tắc.
- Ứng dụng Android cần biết trước địa chỉ IP của server để có thể tiến hành kết nối. Do đó mỗi lần thay đổi kết nối mạng (wifi …) cần phải lấy lại IP của server để có thể chạy được.

5 Hướng dẫn sử dụng

5.1 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.

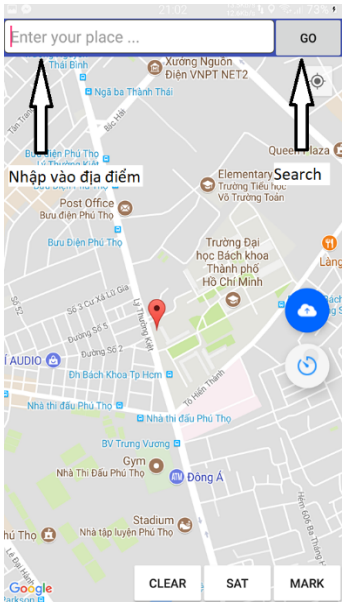

Figure 1. Search bar địa điểm

- 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.

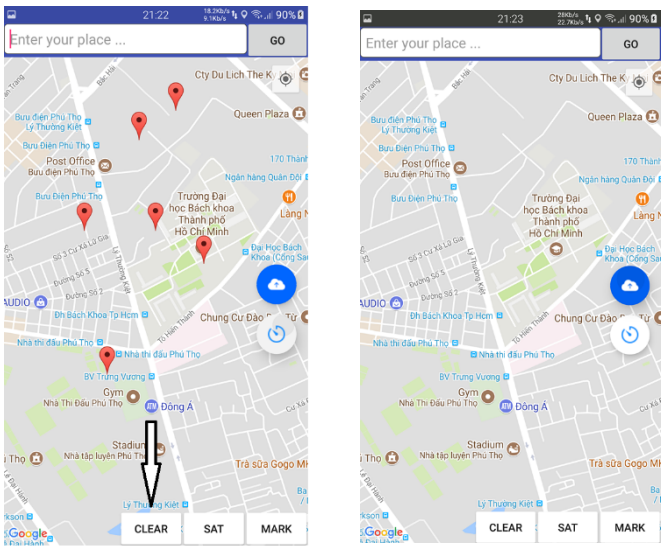

Figure 2. Trước và sau khi dùng Clear

- Sat button: chuyển đổi view nhìn bản đồ, từ view bình thường sang view vệ tinh (satellite)

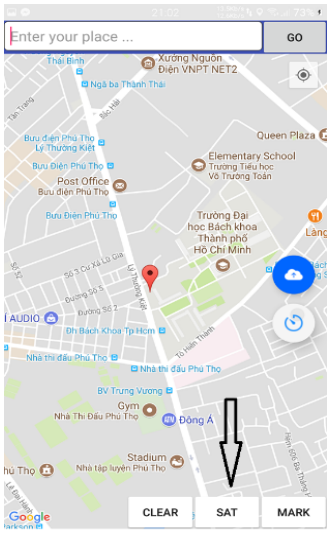

Figure 3. Thay đổi view bản đồ

- Mark button: gắn marker vào vị trí hiện tại của người dùng.

5.2 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.

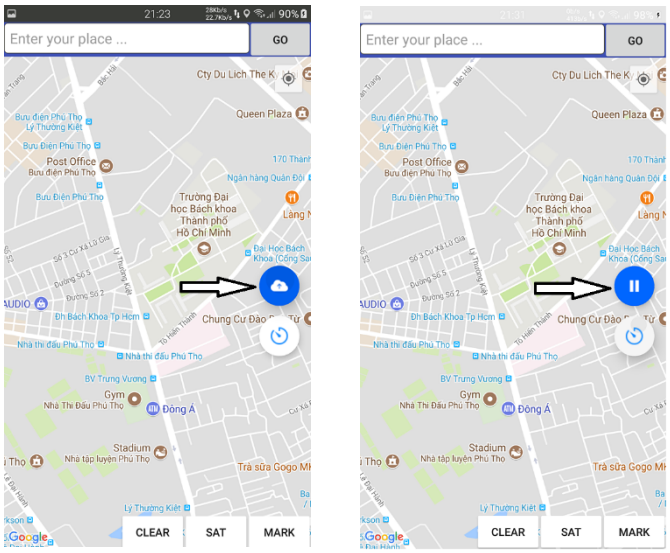

Figure 4. Bắt đầu và dừng gửi dữ liệu

Ngoài ra, nếu muốn cấu hình thời gian giữa mỗi lần gửi dữ liệu

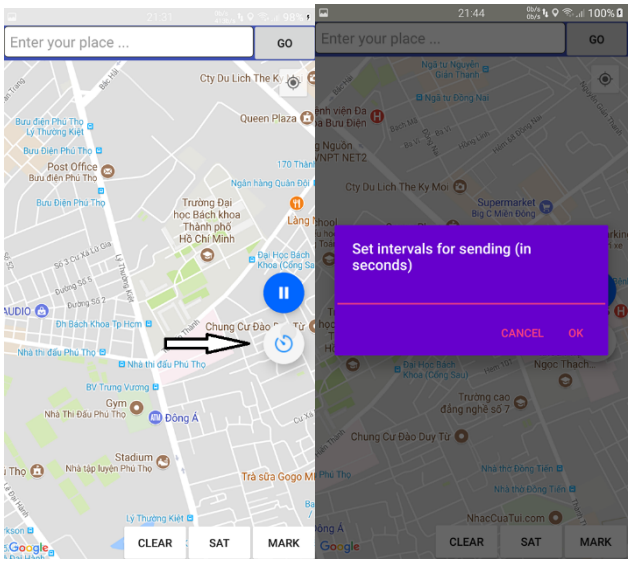

Figure 5. Cấu hình thời gian giữa mỗi lần gửi

- 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.

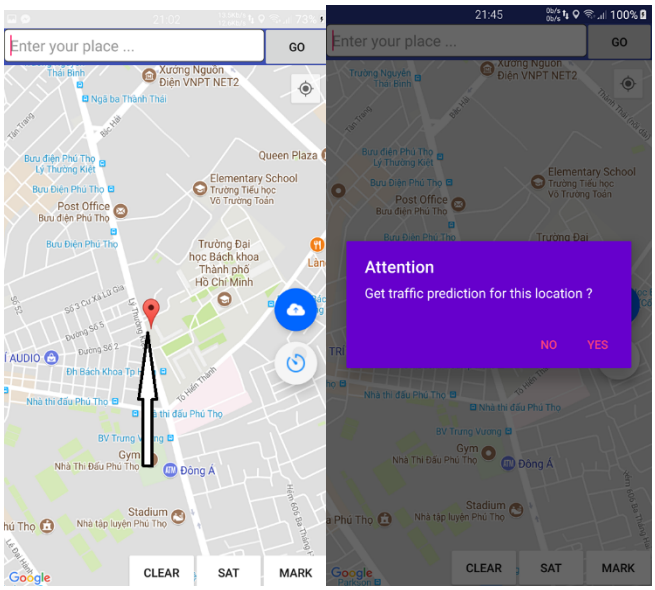

Figure 6. Request dự báo giao thông tại marker

6 Tham khảo

Tài liệu cho Java và Java Networking

- https://www.tutorialspoint.com/java/java_networking.htm
- https://www.tutorialspoint.com/java/

Hướng dẫn tổng quát về Android

- https://developer.android.com/index.html

Tài liệu cho thiết kế giao diện Android

- https://materialdoc.com/

Hướng dẫn cài đặt và sử dụng Google Maps API

- https://developers.google.com/maps/documentation/android-api/start
- https://www.youtube.com/watch?v=k2KXnT4ZecU

Tài liệu về Cloud MQTT

- https://www.cloudmqtt.com/docs.html
- https://www.cloudmqtt.com/docs-java.html