1. Giới thiệu về ứng dụng:
   1. Mục đích 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.
   2. Các thành phần của ứng dụng:
      * Application được xây dựng trên Android.
      * Gateway.
      * Cloud MQTT
      * Analytic Application.
   3. Các công nghệ được sử dụng: giao thức TCP, UDP, MQTT.
2. Các chức năng cụ thể của ứng dụng:
   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. 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.
   3. Cloud MQTT: Nhận dữ liệu từ các Gateway gửi về.
   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.
3. Thiết kế chi tiết kiến trúc hệ thống:
   1. Android application (Client):
   2. Gateway: Trong ứng dụng Gateway có 2 phần nhiệm vụ chính là nhận dữ liệu từ Client gửi lên và đưa dữ liệu lên Cloud thông qua giao thức MQTT, cụ thể:
      * 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. Cloudmqtt: là một server cung cấp nền tảng truyền tin cho các ứng dụng IoT thông qua giao thức MQTT, chi tiết: cloudmqtt.com
   4. Analytic Application:
4. Thiết kế chi tiết các bộ giao thức được phát triển bởi ứng dụng:
   1. Giao thức trao đổi dữ liệu giữa Client – Gateway:
   2. Giao thức trao đổi dữ liệu giữa Client – Analytic Application:
5. Sơ đồ class và phương án triển khai ứng dụng:
6. 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.

1. Các chức năng mở rộng hệ thống ngoài các yêu cầu được qui định:
2. Hướng dẫn sử dụng ứng dụng:

Các tác vụ cơ bản

* 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.

Bộ giao thức được phát triển:

* Giữa Client- Gateway: UDP.
* Giữa Gateway – Cloud và Cloud – Analytic Application: MQTT.
* Giữa Analytic Application và Client: TCP.

Dữ liệu: được lưu 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 tung độ của địa điểm mà client chọn.
* Time: (Type: Date, format: dd/MM/yyyy hh:mm:ss) là thời điểm tạo ra object.

Cách tạo một 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();  
}

Cách gửi dữ liệu:

* TCP: Sử dụng hàm toString() của JSON object, trả về dạng String của Object. String này sẽ được gửi thông qua giao thức TCP.

obj.toString()

* UDP: tương tự như trên, từ String của JSON object, ta chuyển sang một byte array và gửi thông qua giao thức UDP

obj.toString().getBytes()

1. 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.

Hướng dẫn sử dụng app: