

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO ĐỒ ÁN PROJECT 3

THIẾT KẾ VÀ XÂY DỰNG HỆ THỐNG IOT THU THẬP THỜI QUEN LÁI XE VÀ CẢNH BÁO AN TOÀN

Sinh viên thực hiện: Đỗ Đình Hoàng

MSSV: 20225445

Giảng viên hướng dẫn: Nguyễn Đức Tiến

Hà Nội, Tháng 2 Năm 2026

Mục lục

1	GIỚI THIỆU CHUNG	3
1.1	Lý do chọn đề tài	3
1.2	Mục tiêu sản phẩm	3
1.3	Hướng tiếp cận và Công nghệ	3
2	KIẾN TRÚC HỆ THỐNG VÀ THIẾT BỊ	4
2.1	Tổng quan thiết bị sử dụng	4
2.2	Danh sách thiết bị phần cứng (Bill of Materials)	4
2.3	Mô tả chi tiết các cảm biến	4
2.3.1	Cảm biến ánh sáng (Quang trở – LDR)	4
2.3.2	Cảm biến vật cản hồng ngoại	5
2.3.3	Cảm biến rung SW520D	5
2.4	Nhận xét về lựa chọn thiết bị	5
3	CẤU HÌNH VÀ TRIỂN KHAI PHẦN CỨNG	6
3.1	Sơ đồ kết nối (Wiring Diagram)	6
3.2	Cấu hình môi trường phần mềm	6
4	THUẬT TOÁN VÀ LOGIC ĐIỀU KHIỂN	7
4.1	Kiến trúc xử lý tổng thể	7
4.2	Khởi tạo và cấu hình phần cứng	7
4.3	Thuật toán đọc cảm biến	8
4.4	Thuật toán cảnh báo thời gian thực	8
4.5	Đóng gói và truyền dữ liệu MQTT	8
5	THIẾT KẾ HỆ THỐNG XỬ LÝ DỮ LIỆU VÀ GIAO DIỆN	10
5.1	Tổng quan luồng dữ liệu hệ thống	10
5.2	Giao thức MQTT và cơ chế hoạt động	10
5.2.1	Giới thiệu giao thức MQTT	10
5.2.2	Mô hình Producer – Broker – Consumer	11
5.2.3	Cấu trúc topic MQTT	11
5.3	Thiết kế và hoạt động của Consumer (Backend)	11
5.3.1	Kiến trúc Backend	11
5.3.2	Cơ chế subscribe và xử lý dữ liệu	12
5.3.3	Thiết kế cơ sở dữ liệu	12
5.4	Thiết kế giao diện người dùng (Dashboard)	13
5.4.1	Mục tiêu thiết kế giao diện	13
5.4.2	Các thành phần chính của Dashboard	13

5.4.3	Cơ chế cập nhật dữ liệu thời gian thực	14
5.5	Đánh giá thiết kế chương 5	14
6	KẾT QUẢ VÀ ĐÁNH GIÁ	15
6.1	Kết quả đạt được	15
6.1.1	Kết quả về chức năng phần cứng	15
6.1.2	Kết quả về thời gian phản hồi và độ ổn định	15
6.1.3	Kết quả về hệ thống xử lý dữ liệu và Dashboard	16
6.2	Đánh giá hệ thống	16
6.2.1	Ưu điểm của hệ thống	16
6.2.2	Hạn chế còn tồn tại	16
6.3	Hướng phát triển trong tương lai	17

Chương 1

GIỚI THIỆU CHUNG

1.1 Lý do chọn đề tài

Trong bối cảnh giao thông đô thị ngày càng phức tạp, việc giám sát thói quen lái xe không chỉ giúp người điều khiển nhận biết được các nguy cơ tiềm ẩn mà còn cung cấp dữ liệu quý giá cho việc phân tích an toàn giao thông. Các hệ thống hiện có thường chỉ tập trung vào định vị GPS mà chưa đi sâu vào việc cảm nhận môi trường vật lý xung quanh phương tiện như độ rung lắc hay vật cản thời gian thực.

1.2 Mục tiêu sản phẩm

Mục tiêu cốt lõi là xây dựng một thiết bị IoT nhỏ gọn, lắp đặt trên xe máy nhằm:

- Thu thập dữ liệu môi trường (ánh sáng, vật cản) và trạng thái xe (rung lắc).
- Đưa ra cảnh báo tức thời bằng còi và đèn LED khi phát hiện nguy hiểm.
- Truyền dữ liệu qua Wi-Fi lên máy chủ để lưu trữ và hiển thị trên Dashboard trực tuyến.

1.3 Hướng tiếp cận và Công nghệ

Hệ thống sử dụng vi điều khiển ESP8266 làm trung tâm xử lý tại biên (Edge Computing). Dữ liệu từ cảm biến được đọc liên tục với tần suất cao (100ms/lần) và gửi lên máy chủ qua giao thức MQTT nhẹ nhàng, phù hợp với các thiết bị IoT di động.

Chương 2

KIẾN TRÚC HỆ THỐNG VÀ THIẾT BỊ

2.1 Tổng quan thiết bị sử dụng

Hệ thống được thiết kế nhằm thu thập dữ liệu thói quen lái xe thông qua các cảm biến môi trường và trạng thái chuyển động của phương tiện. Các thiết bị được lựa chọn đều hoạt động ổn định ở điện áp thấp (3.3V–5V), phù hợp với vi điều khiển ESP8266 và điều kiện triển khai thực tế trên xe máy.

2.2 Danh sách thiết bị phần cứng (Bill of Materials)

Danh sách các thiết bị phần cứng sử dụng trong hệ thống

STT	Thiết bị	Điện áp	Đầu ra	Chức năng
-----	----------	---------	--------	-----------

1	ESP8266 NodeMCU	3.3V	Digital / Wi-Fi	Điều khiển trung tâm
2	Cảm biến ánh sáng (LDR)	3.3V–5V	AO / DI	Đo cường độ ánh sáng
3	Cảm biến vật cản hồng ngoại	3.3V–5V	DI (TTL)	Phát hiện vật cản
4	Cảm biến rung SW520D	3.3V–5V	DI (TTL)	Phát hiện rung lắc
5	Buzzer	3.3V	Digital	Cảnh báo âm thanh
6	LED	3.3V	Digital	Cảnh báo trực quan

2.3 Mô tả chi tiết các cảm biến

2.3.1 Cảm biến ánh sáng (Quang trở – LDR)

Cảm biến ánh sáng được sử dụng trong hệ thống là module quang trở (LDR), cho phép đo cường độ ánh sáng môi trường xung quanh phương tiện.

Thông số kỹ thuật chính:

- Điện áp hoạt động: 3.3V – 5V
- Ngõ ra:

- AO (Analog Output): 0 – 5V
- DI (Digital Output)

Trong đồ án này, cảm biến được kết nối vào chân Analog A0 của ESP8266 để thu thập giá trị liên tục trong dải 0–1024. Giá trị ánh sáng này được sử dụng để xác định điều kiện môi trường (ban ngày hoặc ban đêm), từ đó phục vụ phân tích mức độ rủi ro khi di chuyển.

2.3.2 Cảm biến vật cản hồng ngoại

Cảm biến vật cản hồng ngoại hoạt động dựa trên nguyên lý phát và thu tia hồng ngoại. Khi có vật thể ở phía trước, tia hồng ngoại phản xạ trở lại làm thay đổi trạng thái tín hiệu đầu ra.

Thông số kỹ thuật chính:

- Điện áp hoạt động: 3.3V – 5V
- Ngõ ra: Digital (TTL)

Cảm biến được sử dụng để phát hiện vật cản ở khoảng cách gần, giúp hệ thống đưa ra cảnh báo kịp thời nhằm hạn chế nguy cơ va chạm khi tham gia giao thông.

2.3.3 Cảm biến rung SW520D

Cảm biến rung SW520D là cảm biến dạng công tắc rung, hoạt động dựa trên nguyên lý đóng/ngắt mạch khi có tác động rung hoặc va chạm cơ học.

Thông số kỹ thuật chính:

- Điện áp hoạt động: 3.3V – 5V
- Ngõ ra: Digital (TTL)

Trong hệ thống, cảm biến rung được sử dụng để ghi nhận các rung lắc bất thường của xe, chẳng hạn như khi di chuyển trên mặt đường xấu hoặc xảy ra va chạm nhẹ. Dữ liệu rung lắc đóng vai trò quan trọng trong việc phân tích thói quen lái xe và đánh giá mức độ an toàn của hành trình.

2.4 Nhận xét về lựa chọn thiết bị

Các thiết bị và cảm biến được lựa chọn đều có ưu điểm:

- Dễ tích hợp với ESP8266
- Giá thành thấp, phổ biến trên thị trường
- Phù hợp với các hệ thống IoT di động

Sự kết hợp giữa cảm biến ánh sáng, cảm biến vật cản và cảm biến rung cho phép hệ thống thu thập dữ liệu đa chiều, tạo nền tảng cho việc phân tích thói quen lái xe và xây dựng các thuật toán cảnh báo an toàn.

Chương 3

CẤU HÌNH VÀ TRIỂN KHAI PHẦN CỨNG

3.1 Sơ đồ kết nối (Wiring Diagram)

Hệ thống phần cứng được thiết kế xoay quanh vi điều khiển ESP8266 NodeMCU, đóng vai trò trung tâm trong việc thu thập dữ liệu từ các cảm biến và điều khiển các thiết bị cảnh báo.

Cấu hình chân kết nối cụ thể như sau:

- Cảm biến vật cản hồng ngoại được kết nối với chân D5 (GPIO 14).
- Cảm biến rung SW520D được kết nối với chân D1 (GPIO 5).
- Cảm biến ánh sáng (LDR) được kết nối với chân Analog A0.
- Còi báo (Buzzer) được điều khiển bởi chân D6 (GPIO 12).
- Đèn LED cảnh báo được điều khiển bởi chân D7 (GPIO 13).

Việc tách riêng các chân Digital và Analog giúp hệ thống hoạt động ổn định, giảm nhiễu và thuận tiện cho việc mở rộng trong tương lai.

3.2 Cấu hình môi trường phần mềm

Môi trường phát triển cho ESP8266 được xây dựng dựa trên Arduino IDE với các bước chính:

1. Cài đặt Arduino IDE.
2. Thêm Board ESP8266 thông qua Board Manager.
3. Cài đặt thư viện PubSubClient để hỗ trợ giao thức MQTT.
4. Cấu hình các tham số mạng như SSID, mật khẩu Wi-Fi và địa chỉ MQTT Broker.

Sau khi hoàn tất cấu hình, ESP8266 có thể kết nối Wi-Fi và sẵn sàng truyền dữ liệu lên máy chủ.

Chương 4

THUẬT TOÁN VÀ LOGIC ĐIỀU KHIỂN

4.1 Kiến trúc xử lý tổng thể

Chương trình được xây dựng theo mô hình vòng lặp không chặn (non-blocking loop), sử dụng hàm `millis()` để quản lý thời gian thay cho `delay()`. Cách tiếp cận này giúp hệ thống duy trì kết nối MQTT ổn định và phản hồi nhanh với các sự kiện nguy hiểm.

Hai chu kỳ chính được sử dụng trong hệ thống:

- Chu kỳ đọc cảm biến: 100ms.
- Chu kỳ gửi dữ liệu MQTT: 1000ms.

4.2 Khởi tạo và cấu hình phần cứng

Các chân GPIO được định nghĩa tương ứng với từng cảm biến và thiết bị đầu ra như thể hiện trong đoạn mã sau:

```
#define OBSTACLE_PIN 14
#define VIBRATION_PIN 5
#define LIGHT_PIN A0

#define BUZZER_PIN 12
#define LED_PIN 13
```

Listing 4.1: Định nghĩa chân kết nối phần cứng

Trong hàm `setup()`, các chân được cấu hình chế độ hoạt động:

```
pinMode(OBSTACLE_PIN, INPUT);
pinMode(VIBRATION_PIN, INPUT);
pinMode(BUZZER_PIN, OUTPUT);
pinMode(LED_PIN, OUTPUT);
```

Listing 4.2: Cấu hình chế độ GPIO

4.3 Thuật toán đọc cảm biến

Cứ mỗi 100ms, hệ thống tiến hành đọc dữ liệu từ các cảm biến:

```
obstacleState = digitalRead(OBSTACLE_PIN);
vibrationState = digitalRead(VIBRATION_PIN);
lightValue = analogRead(LIGHT_PIN);
```

Listing 4.3: Đọc dữ liệu cảm biến

Cảm biến vật cản và rung trả về tín hiệu Digital (HIGH/LOW), trong khi cảm biến ánh sáng trả về giá trị Analog trong khoảng từ 0 đến 1024.

4.4 Thuật toán cảnh báo thời gian thực

Thuật toán cảnh báo được xây dựng theo nguyên tắc logic OR. Chỉ cần một trong hai điều kiện nguy hiểm xảy ra (phát hiện vật cản hoặc rung lắc mạnh), hệ thống sẽ kích hoạt cảnh báo ngay lập tức.

```
if (obstacleState == LOW || vibrationState == LOW) {
    buzzerState = HIGH;
    ledState = HIGH;
} else {
    buzzerState = LOW;
    ledState = LOW;
}
```

Listing 4.4: Logic điều khiển cảnh báo

Sau khi xác định trạng thái, tín hiệu được xuất ra còi và đèn LED:

```
digitalWrite(BUZZER_PIN, buzzerState);
digitalWrite(LED_PIN, ledState);
```

Listing 4.5: Xuất tín hiệu cảnh báo

4.5 Đóng gói và truyền dữ liệu MQTT

Dữ liệu cảm biến và trạng thái thiết bị được đóng gói dưới dạng JSON để gửi lên máy chủ thông qua MQTT.

```
String json = "{";
json += "\"deviceId\":\"" + deviceId + "\",";
json += "\"timestamp\":\"" + String(millis()) + ",";
json += "\"sensors\":{\"";
json += "\"obstacle\":\"" + String(obstacleState) + ",";
json += "\"vibration\":\"" + String(vibrationState) + ",";
json += "\"light\":0";
json += "},";
json += "\"outputs\":{\"";
json += "\"buzzer\":\"" + String(buzzerState) + ",";
json += "\"led\":0";
json += "}"
```

```
json += "}";
```

Listing 4.6: Đóng gói dữ liệu JSON

Chu kỳ gửi dữ liệu được kiểm soát bằng hàm `millis()` nhằm tránh gửi quá dày:

```
if (millis() - lastMqttPublish >= mqttInterval) {  
    lastMqttPublish = millis();  
    publishSensorData();  
}
```

Listing 4.7: Chu kỳ gửi MQTT

Chương 5

THIẾT KẾ HỆ THỐNG XỬ LÝ DỮ LIỆU VÀ GIAO DIỆN

5.1 Tổng quan luồng dữ liệu hệ thống

Hệ thống được xây dựng theo mô hình IoT chuẩn gồm ba thành phần chính: thiết bị IoT (Producer), MQTT Broker và hệ thống Backend (Consumer). Dữ liệu được truyền theo thời gian thực từ thiết bị gắn trên xe máy lên máy chủ và hiển thị trên giao diện người dùng.

Luồng dữ liệu bao gồm các bước:

- ESP8266 thu thập dữ liệu cảm biến
 - Dữ liệu được publish lên MQTT Broker
 - Backend subscribe và xử lý dữ liệu
 - Dữ liệu được lưu trữ và hiển thị trên Dashboard
-

5.2 Giao thức MQTT và cơ chế hoạt động

5.2.1 Giới thiệu giao thức MQTT

MQTT (Message Queuing Telemetry Transport) là giao thức truyền thông nhẹ, hoạt động theo mô hình publish/subscribe, phù hợp với các thiết bị IoT có tài nguyên hạn chế và môi trường mạng không ổn định.

So với mô hình HTTP truyền thống, MQTT có các ưu điểm:

- Giảm băng thông truyền tải
 - Độ trễ thấp
 - Hỗ trợ giao tiếp thời gian thực
-

5.2.2 Mô hình Producer – Broker – Consumer

Producer (ESP8266) Thiết bị ESP8266 đóng vai trò là Producer, chịu trách nhiệm thu thập dữ liệu cảm biến và gửi dữ liệu lên MQTT Broker thông qua các topic đã định nghĩa trước.

Broker (Mosquitto) MQTT Broker đóng vai trò trung gian, nhận thông điệp từ Producer và phân phối lại cho các Consumer đã subscribe. Broker không xử lý nội dung dữ liệu mà chỉ đảm nhiệm việc định tuyến thông điệp.

Consumer (Backend Server) Backend đóng vai trò Consumer, subscribe các topic liên quan và xử lý dữ liệu nhận được theo logic nghiệp vụ của hệ thống.

5.2.3 Cấu trúc topic MQTT

Hệ thống sử dụng cấu trúc topic phân cấp như sau:

```
iot/vehicle/{deviceId}/data  
iot/vehicle/{deviceId}/status
```

Trong đó:

- **deviceId**: mã định danh duy nhất của thiết bị ESP8266
- **data**: topic chứa dữ liệu cảm biến
- **status**: topic chứa trạng thái kết nối của thiết bị

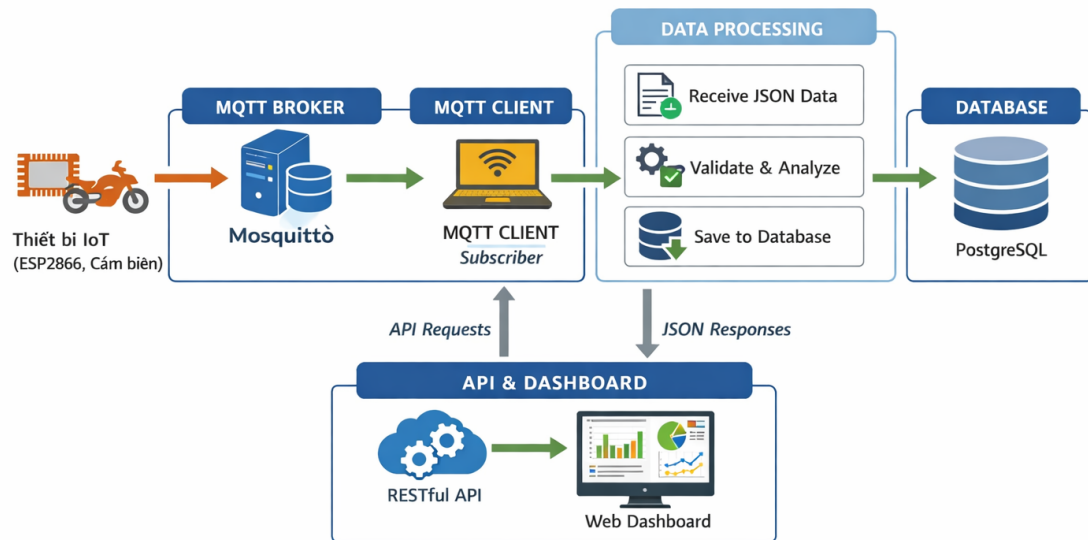
Cách tổ chức này cho phép hệ thống mở rộng nhiều thiết bị mà không cần thay đổi kiến trúc tổng thể.

5.3 Thiết kế và hoạt động của Consumer (Backend)

5.3.1 Kiến trúc Backend

Backend được xây dựng theo mô hình dịch vụ, chịu trách nhiệm:

- Nhận dữ liệu MQTT
 - Phân tích và kiểm tra dữ liệu
 - Lưu trữ dữ liệu vào cơ sở dữ liệu
 - Cung cấp API cho giao diện Dashboard
-



Hình 5.1: Sơ đồ kiến trúc backend

5.3.2 Cơ chế subscribe và xử lý dữ liệu

Backend sử dụng MQTT Client để subscribe các topic dữ liệu. Khi một thông điệp mới được gửi đến, hàm callback sẽ được kích hoạt để xử lý dữ liệu.

Quy trình xử lý gồm các bước:

1. Nhận thông điệp JSON từ MQTT Broker
2. Parse dữ liệu cảm biến
3. Kiểm tra tính hợp lệ của dữ liệu
4. Lưu dữ liệu vào cơ sở dữ liệu PostgreSQL

Việc xử lý này được thực hiện theo cơ chế bất đồng bộ, đảm bảo hệ thống có thể xử lý nhiều thiết bị cùng lúc mà không gây nghẽn tài nguyên.

5.3.3 Thiết kế cơ sở dữ liệu

Dữ liệu cảm biến được lưu trữ trong bảng `sensor_logs` với cấu trúc như sau:

- `log_id`: khóa chính, tự tăng
- `timestamp`: thời điểm ghi nhận dữ liệu
- `device_id`: định danh thiết bị

- vibration_val: trạng thái rung
- obstacle_val: trạng thái vật cản
- light_level: cường độ ánh sáng

Cấu trúc này cho phép truy vấn dữ liệu theo thời gian, theo thiết bị và phục vụ phân tích hành vi lái xe.

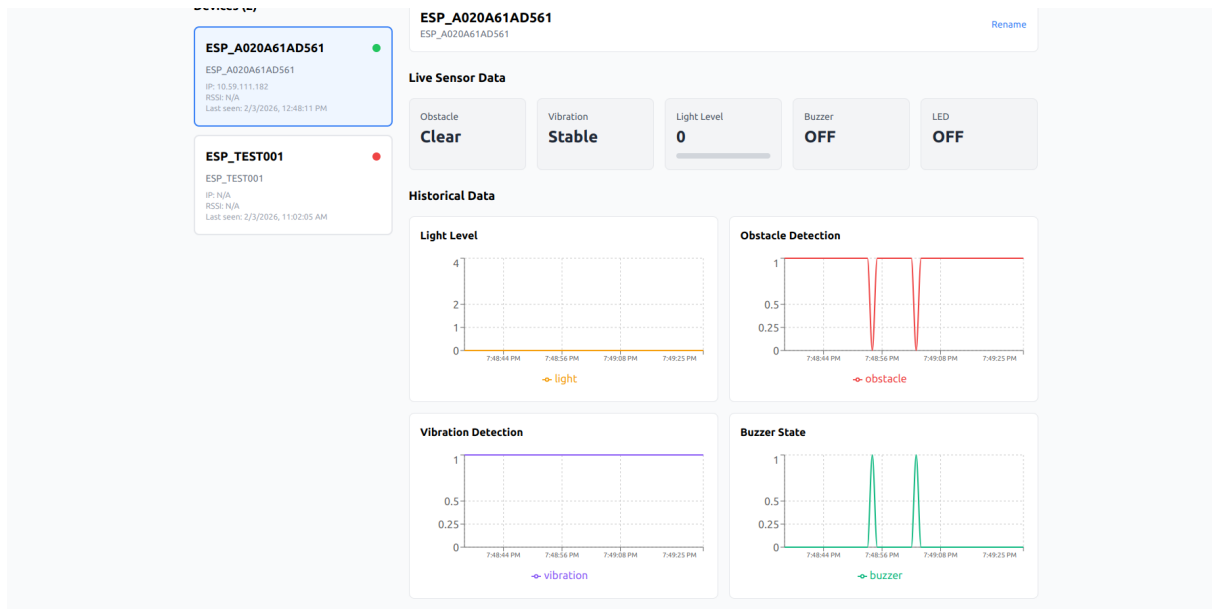
5.4 Thiết kế giao diện người dùng (Dashboard)

5.4.1 Mục tiêu thiết kế giao diện

Giao diện Dashboard được thiết kế nhằm:

- Hiển thị dữ liệu cảm biến theo thời gian thực
- Cung cấp biểu đồ trực quan
- Giúp người dùng dễ dàng theo dõi trạng thái xe

5.4.2 Các thành phần chính của Dashboard



Hình 5.2: Dashboard hiện thị

Dashboard bao gồm các thành phần chính:

- Bảng trạng thái thiết bị (Online/Offline)
- Biểu đồ cường độ ánh sáng theo thời gian

- Biểu đồ tần suất rung lắc
 - Cảnh báo khi phát hiện nguy hiểm
-

5.4.3 Cơ chế cập nhật dữ liệu thời gian thực

Giao diện sử dụng cơ chế cập nhật dữ liệu theo thời gian thực thông qua WebSocket hoặc API polling định kỳ. Khi Backend nhận dữ liệu mới từ MQTT, thông tin sẽ được đẩy ngay lên giao diện giúp người dùng theo dõi tình trạng xe gần như tức thời.

5.5 Đánh giá thiết kế chương 5

Thiết kế hệ thống xử lý dữ liệu và giao diện đáp ứng tốt yêu cầu của một hệ thống IoT thời gian thực:

- Độ trễ thấp
- Dễ mở rộng
- Giao diện trực quan, dễ sử dụng

Chương này đóng vai trò cầu nối giữa phần cứng IoT và trải nghiệm người dùng cuối.

Chương 6

KẾT QUẢ VÀ ĐÁNH GIÁ

6.1 Kết quả đạt được

Sau quá trình thiết kế, triển khai và thử nghiệm, hệ thống IoT thu thập thói quen lái xe và cảnh báo an toàn đã đạt được các kết quả chính như sau:

6.1.1 Kết quả về chức năng phần cứng

Thiết bị phần cứng gắn trên xe máy hoạt động ổn định trong điều kiện thực tế. Các cảm biến ánh sáng, vật cản hồng ngoại và rung SW520D đều phản hồi đúng với các thay đổi của môi trường xung quanh phương tiện.

Cụ thể:

- Cảm biến ánh sáng (LDR) cho phép phân biệt rõ điều kiện ban ngày và ban đêm, phục vụ phân tích bối cảnh lái xe.
- Cảm biến vật cản hồng ngoại phát hiện vật cản ở khoảng cách gần với độ trễ thấp, phù hợp cho mục đích cảnh báo tức thời.
- Cảm biến rung SW520D ghi nhận hiệu quả các rung lắc bất thường của xe khi đi qua đoạn đường xấu hoặc xảy ra va chạm nhẹ.

Các thiết bị cảnh báo gồm còi (buzzer) và đèn LED hoạt động chính xác theo logic điều khiển đã thiết kế, giúp người lái nhận biết nguy hiểm ngay lập tức.

6.1.2 Kết quả về thời gian phản hồi và độ ổn định

Qua quá trình thử nghiệm, hệ thống cho thấy khả năng phản hồi nhanh:

- Thời gian từ lúc phát hiện sự kiện nguy hiểm đến khi kích hoạt cảnh báo nhỏ hơn 150 ms.
- Chu kỳ đọc cảm biến 100 ms đảm bảo không bỏ sót các sự kiện ngắn.
- Dữ liệu MQTT được gửi ổn định với chu kỳ 1 giây mà không gây nghẽn hệ thống.

Trong điều kiện kết nối Wi-Fi ổn định, hệ thống không ghi nhận hiện tượng mất gói dữ liệu, đảm bảo tính toàn vẹn của thông tin truyền về máy chủ.

6.1.3 Kết quả về hệ thống xử lý dữ liệu và Dashboard

Hệ thống Backend và Dashboard hoạt động đúng theo thiết kế:

- Backend nhận và xử lý dữ liệu MQTT theo thời gian thực.
- Dữ liệu được lưu trữ đầy đủ trong cơ sở dữ liệu PostgreSQL.
- Dashboard hiển thị trực quan các biểu đồ về cường độ ánh sáng, tần suất rung lắc và số lần phát hiện vật cản.

Thông qua Dashboard, người dùng có thể theo dõi tình trạng phương tiện và thói quen lái xe theo từng khoảng thời gian, từ đó nâng cao nhận thức về an toàn khi tham gia giao thông.

6.2 Đánh giá hệ thống

6.2.1 Ưu điểm của hệ thống

Hệ thống đạt được nhiều ưu điểm nổi bật:

- Kiến trúc IoT rõ ràng, tuân theo mô hình Producer – Broker – Consumer.
- Chi phí phần cứng thấp, dễ triển khai và mở rộng.
- Giao thức MQTT nhẹ, phù hợp với thiết bị di động và mạng không ổn định.
- Khả năng cảnh báo thời gian thực giúp tăng mức độ an toàn cho người lái.

Ngoài ra, việc tách biệt rõ ràng giữa phần cứng IoT, Backend và giao diện người dùng giúp hệ thống dễ bảo trì và nâng cấp trong tương lai.

6.2.2 Hạn chế còn tồn tại

Bên cạnh các kết quả đạt được, hệ thống vẫn tồn tại một số hạn chế:

- Hệ thống phụ thuộc vào kết nối Wi-Fi, chưa phù hợp khi di chuyển trên quãng đường dài hoặc khu vực không có mạng.
- Cảm biến vật cản hồng ngoại chỉ hoạt động hiệu quả trong khoảng cách ngắn và bị ảnh hưởng bởi ánh sáng môi trường.
- Chưa có thuật toán phân tích nâng cao để đánh giá mức độ nguy hiểm một cách định lượng.

Những hạn chế này chủ yếu xuất phát từ phạm vi và thời gian thực hiện đồ án, tuy nhiên không ảnh hưởng đến mục tiêu chính là xây dựng một hệ thống IoT hoàn chỉnh và có khả năng hoạt động thực tế.

6.3 Hướng phát triển trong tương lai

Trong tương lai, hệ thống có thể được mở rộng và nâng cấp theo các hướng sau:

- Tích hợp module GPS để thu thập dữ liệu vị trí, từ đó xây dựng bản đồ nhiệt các khu vực nguy hiểm.
- Sử dụng module SIM 4G/5G nhằm đảm bảo kết nối liên tục khi phương tiện di chuyển ngoài vùng phủ sóng Wi-Fi.
- Áp dụng các thuật toán học máy (Machine Learning) để phân tích thói quen lái xe và dự đoán nguy cơ tai nạn.
- Mở rộng Dashboard với các chức năng thống kê theo ngày, tuần, tháng nhằm hỗ trợ đánh giá hành vi lái xe dài hạn.

Với các hướng phát triển trên, hệ thống có tiềm năng trở thành một nền tảng hỗ trợ an toàn giao thông thông minh, góp phần giảm thiểu rủi ro và nâng cao ý thức của người tham gia giao thông.