**BÁO CÁO ĐỀ TÀI IOT**

# 1. Thông tin chung

* Tên đề tài: Hệ thống IoT Chăm Sóc Thú Cưng Thông Minh – *Mô-đun Nước Uống Tự Động (Automatic Water Drinking System)*
* Thời gian thực hiện: 2 tuần (10h/tuần)
* Số thành viên nhóm: 5
* Tập trung vào mô-đun nước uống tự động trong hệ thống Smart Pet Care, có tích hợp Cloud MQTT và Node-RED Dashboard để mô phỏng toàn bộ quy trình IoT theo thời gian thực

# 2. Mục tiêu

Mục tiêu tổng thể của hệ thống Smart Pet Care

* Xây dựng hệ thống IoT chăm sóc thú cưng gồm 5 mô-đun: theo dõi chuyển động, camera giám sát, giám sát môi trường sống, nước uống tự động, và cho ăn tự động.
* Tự động hóa quy trình chăm sóc, tăng sự tiện nghi và đảm bảo sức khỏe cho thú cưng.
* Áp dụng công nghệ IoT hiện đại: MQTT, Dashboard, Cloud, cảm biến, AI

Mục tiêu cụ thể của mô-đun Nước Uống Tự Động

* Giám sát mực nước uống của thú cưng theo thời gian thực thông qua cảm biến mô phỏng.
* Mô phỏng thiết bị IoT (Python) gửi dữ liệu lên Cloud MQTT Broker.
* Thiết kế Dashboard trên Node-RED để hiển thị mực nước bằng Gauge và biểu đồ lịch sử.
* Xây dựng logic tự động bật/tắt bơm:

+ Nước < 30%: tự động bật bơm

+ Nước ≥ 90%: tự động tắt bơm.

* Cung cấp khả năng điều khiển thủ công qua giao diện người dùng (Manual Override).
* Thể hiện vòng điều khiển IoT khép kín:

*Thiết bị → Cloud → Xử lý → Phản hồi → Thiết bị.*

# 3. Nội dung thực hiện

## 3.1. Giới thiệu tổng quan hệ thống Smart Pet Care

Hệ thống Smart Pet Care là một mô hình ứng dụng IoT giúp tự động hóa quá trình chăm sóc thú cưng tại gia đình hoặc trong các cơ sở chăm nuôi. Hệ thống gồm 5 mô-đun chức năng chính, được kết nối thông qua mạng Internet và Cloud MQTT để truyền nhận dữ liệu thời gian thực.3.2. Phân tích và thiết kế hệ thống IoT

Các mô-đun chính gồm:

* Theo dõi chuyển động: phát hiện thú cưng di chuyển bất thường, cảnh báo nếu có nguy cơ.
* Camera giám sát: quan sát thú cưng từ xa, truyền hình ảnh/video về hệ thống.
* Theo dõi môi trường sống: cảm biến nhiệt độ, độ ẩm, chất lượng không khí.
* Nước uống tự động (trọng tâm báo cáo này): giám sát mực nước và tự động điều khiển bơm.
* Cho ăn tự động: định lượng khẩu phần ăn theo thời gian cài đặt sẵn.

Trong phạm vi báo cáo này, mô-đun Nước uống tự động được triển khai mô phỏng hoàn chỉnh, thể hiện rõ vòng điều khiển IoT khép kín.

## 3.2 Phân tích mô-đun Nước Uống Tự Động

### 3.2.1. Bài toán thực tế

Thú cưng cần được cung cấp nước sạch liên tục. Nếu nước cạn mà không được kiểm tra kịp thời, thú cưng có thể bị mất nước, giảm sức khỏe. Việc chăm sóc thủ công mất thời gian và dễ bỏ sót, đặc biệt khi chủ nuôi đi vắng.

### 3.2.2 Yêu cầu hệ thống

* Giám sát mực nước theo thời gian thực.
* Tự động bật bơm khi nước dưới ngưỡng an toàn.
* Tự động tắt bơm khi nước đầy để tránh tràn.
* Cho phép người dùng điều khiển thủ công thông qua giao diện Dashboard.
* Hệ thống hoạt động liên tục dựa trên công nghệ IoT và Cloud MQTT.

### 3.2.3. Kiến trúc hệ thống mô phỏng

Hệ thống được mô phỏng dựa trên vòng điều khiển IoT:

*Thiết bị (Python) → MQTT Cloud → Node-RED → Dashboard → Điều khiển → MQTT → Thiết bị*

Các thành phần chính

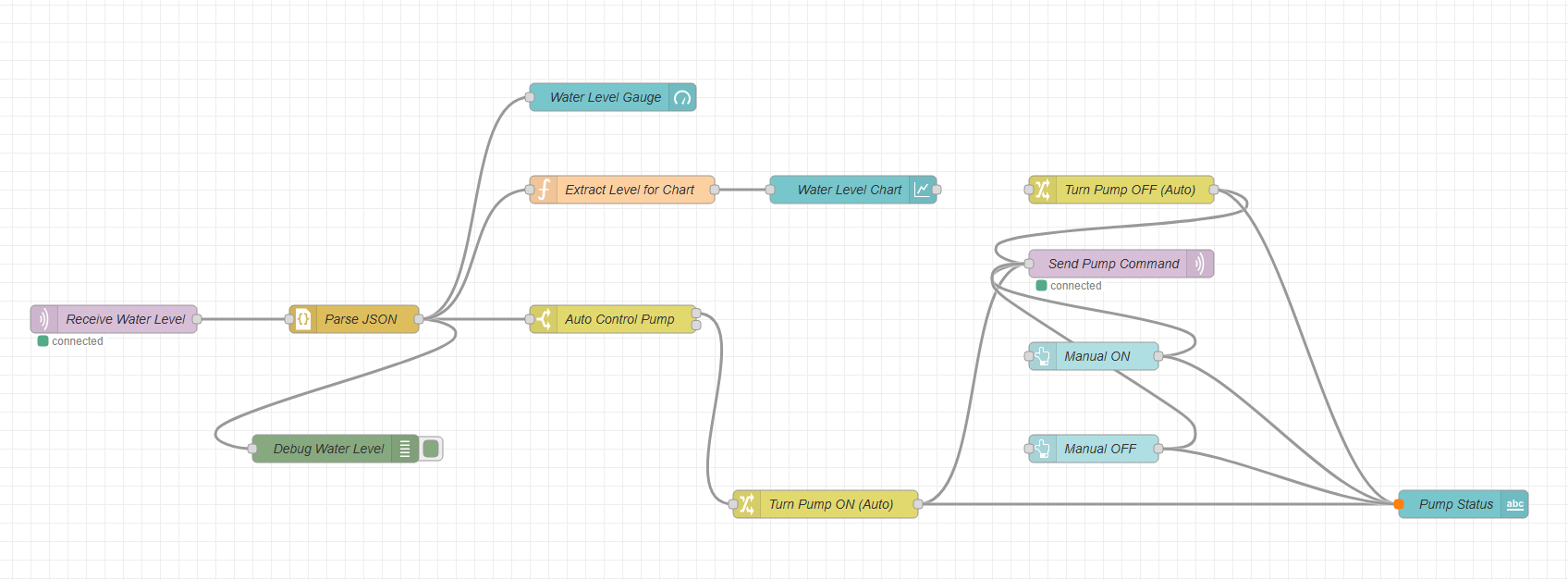
|  |  |  |
| --- | --- | --- |
| **Thành phần** | **Công nghệ sử dụng** | **Vai trò** |
| Thiết bị ảo | Python + Paho MQTT Client | |  | | --- | |  |   Giả lập cảm biến mực nước và máy bơm |
| Cloud MQTT | |  | | --- | |  |   HiveMQ Cloud | Kênh truyền dữ liệu Publish/Subscribe |
| IoT Gateway | Node-RED | |  | | --- | |  |   Xử lý dữ liệu, ra quyết định |
| Dashboard | |  | | --- | |  |   Node-RED Dashboard | Hiển thị mực nước và điều khiển bơm |
| Logic điều khiển | Node-RED Switch & Function | Tự động & thủ công |

### 3.2.4 Dòng dữ liệu (Data Flow)

1. Python gửi dữ liệu mực nước lên topic water/level
2. Node-RED nhận dữ liệu, phân tích:

* <30% → gửi tín hiệu bật bơm (water/pump = ON)
* ≥90% → gửi tín hiệu tắt bơm (water/pump = OFF)

1. Người dùng có thể gửi lệnh thủ công bằng Dashboard.
2. Python nhận lệnh qua topic water/pump và thay đổi mực nước tương ứng.



## 3.3. Xây dựng hệ thống

### 3.3.1. Kiến trúc hệ thống IoT

Hệ thống được xây dựng dựa trên mô hình Publish/Subscribe của giao thức MQTT. Dữ liệu được truyền giữa thiết bị IoT và máy chủ Cloud thông qua các topic, sau đó được Node-RED xử lý và hiển thị trên Dashboard.

Thiết bị (Python mô phỏng mực nước)

↓ Publish: water/level

MQTT Broker (HiveMQ Cloud)

↓ Subscribe

Node-RED (Gateway)

↓ Xử lý logic điều khiển

↓ Publish: water/pump

Thiết bị (Python nhận lệnh bơm)

### 3.3.2. Các topic trong hệ thống

|  |  |  |  |
| --- | --- | --- | --- |
| **Topic** | **Chức năng** | **Thiết bị gửi** | **Thiết bị nhận** |
| water/level | Gửi dữ liệu mức nước hiện tại | |  | | --- | |  |   Python/ESP32 | |  | | --- | |  |   Node-RED |
| |  | | --- | |  |   water/pump | |  | | --- | |  |   Điều khiển bật/tắt máy bơm | |  | | --- | |  |   Node-RED | Python/ESP32 |

### 3.3.3 Xây dựng mô phỏng thiết bị IoT bằng Python

Thiết bị ảo được lập trình để:

* Tạo dữ liệu mực nước thay đổi theo thời gian
* Kết nối tới Cloud MQTT thông qua bảo mật TLS
* Nhận lệnh bật/tắt bơm và thay đổi mực nước tương ứng

Mã nguồn mô phỏng chính (trích đoạn):

client = mqtt.Client(client\_id="water\_sensor")

client.username\_pw\_set("test-user", "Danggtuee@265")

def on\_connect(client, userdata, flags, rc):

client.subscribe("water/pump")

def on\_message(client, userdata, msg):

global pump\_status

pump\_status = msg.payload.decode()

client.connect(MQTT\_BROKER, MQTT\_PORT)

client.loop\_start()

while True:

if pump\_status == "ON" and water\_level < 100:

water\_level += random.randint(1, 5)

else:

water\_level -= random.randint(1, 3)

client.publish("water/level", json.dumps({"level": water\_level}))

time.sleep(2)

### 3.3.4. Thiết kế Node-RED Flow

Flow Node-RED được chia thành 4 nhóm chức năng:

|  |  |  |
| --- | --- | --- |
| **Nhóm chức năng** | **Thành phần Node-RED** | **Mô tả** |
| Nhận dữ liệu | mqtt in + json | |  | | --- | |  |   Nhận dữ liệu water/level từ cloud |
| Hiển thị dashboard | |  | | --- | |  |   ui\_gauge, ui\_chart, ui\_text | Gauge và biểu đồ lịch sử |
| Logic điều khiển tự động | switch, change | |  | | --- | |  |   Bật bơm khi <30%, tắt khi >=90% |
| Điều khiển thủ công | |  | | --- | |  |   ui\_button | Người dùng điều khiển trực tiếp |

### 3.3.5. Giao diện Dashboard

*Gauge*: hiển thị % mực nước theo thời gian thực

*Chart*: biểu đồ lịch sử giúp theo dõi xu hướng tiêu thụ nước

*Buttons*: ON/OFF thủ công

*Text*: hiển thị trạng thái máy bơm (ON/OFF)

Giao diện giúp người dùng theo dõi và tương tác trực tiếp với hệ thống IoT theo thời gian thực.

## 3.4. Quy trình hoạt động của hệ thống

Quy trình hoạt động của mô-đun nước uống tự động được xây dựng dựa trên kiến trúc IoT khép kín, trong đó dữ liệu cảm biến được thu thập liên tục, gửi lên nền tảng đám mây, xử lý logic và phản hồi điều khiển thiết bị. Quy trình có thể được mô tả theo các bước sau:

Bước 1: Thiết bị IoT gửi dữ liệu mực nước

* Thiết bị mô phỏng (Python hoặc ESP32 thực tế) đọc giá trị mực nước.
* Dữ liệu được đóng gói dưới dạng JSON:

{"level": 75}

* Thiết bị publish dữ liệu lên topic water/level của MQTT Cloud Broker.

Bước 2: MQTT Broker truyền dữ liệu

* HiveMQ Cloud tiếp nhận dữ liệu từ thiết bị.
* Node-RED subscribe topic water/level và nhận dữ liệu ngay khi có thay đổi.

Bước 3: Xử lý dữ liệu tại Node-RED

* Dữ liệu được parse từ JSON thành đối tượng.
* Hệ thống hiển thị giá trị trên Dashboard (Gauge và biểu đồ).
* Logic điều khiển được kích hoạt:
* Nếu giá trị mực nước < 30% → bật bơm.
* Nếu ≥ 90% → tắt bơm.

Bước 4: Điều khiển tự động hoặc thủ công

* Chế độ tự động: node switch xử lý và gửi tín hiệu ON/OFF qua topic water/pump.
* Chế độ thủ công: người dùng nhấn nút trên Dashboard để điều khiển trực tiếp.

Bước 5: Thiết bị nhận lệnh và phản hồi

* Thiết bị subscribe topic water/pump.
* Khi nhận lệnh "ON", thiết bị bật bơm và mức nước tăng dần.
* Khi nhận lệnh "OFF", bơm tắt và mức nước giảm tự nhiên (tiêu thụ bởi thú cưng).
* Thiết bị tiếp tục gửi dữ liệu mới lên để cập nhật vòng lặp.

# 4. Phân công công việc (vai trò)

|  |  |  |
| --- | --- | --- |
| **Thành viên** | **Vai trò** | **Công việc chính** |
| Hồ Đăng Tuệ – Nhóm trưởng | Quản lý & tích hợp hệ thống | Phân tích bài toán, thiết kế kiến trúc IoT, tích hợp các thành phần Python – MQTT – Node-RED – Dashboard |
| SV2 | Thiết bị mô phỏng | Viết mã Python mô phỏng cảm biến mực nước và máy bơm, xử lý dữ liệu đầu vào |
| SV3 | |  | | --- | |  |  |  | | --- | | Hệ thống truyền thông IoT | | Cấu hình MQTT Cloud, kiểm tra Publish/Subscribe, bảo mật TLS |
| SV4 | Xử lý logic & tự động hóa | Thiết kế flow xử lý trong Node-RED, cấu hình logic tự động bật/tắt bơm |
| SV5 | Giao diện & báo cáo | Thiết kế Dashboard UI thời gian thực, viết tài liệu và chuẩn bị slide thuyết trình |

# 5. Tiến độ (2 tuần – 20 giờ)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tuần** | **Nội dung** | **Thời gian (giờ)** | **Thành viên chính** |
| Tuần 1 | Xác định bài toán, chia vai trò, thiết kế kiến trúc hệ thống nước uống IoT | 4h | SV1 + cả nhóm |
| Tuần 1 | Viết chương trình Python mô phỏng cảm biến mực nước & máy bơm | 3h | SV4 |
| Tuần 1 | Cài đặt và cấu hình MQTT Cloud với TLS | 3h | SV3 |
| Tuần 2 | |  | | --- | |  |   Tạo flow trong Node-RED, kết nối MQTT In/Out, xử lý logic tự động | 5h | SV2 + SV3 + SV4 |
| Tuần 2 | Thiết kế dashboard UI, test hệ thống và chỉnh lỗi | 3h | SV5 + SV4 |
| Tuần 2 | Tổng hợp kết quả, viết báo cáo và slide thuyết trình | 2h | SV1 + SV5 |

# 6. Kết quả dự kiến

Mô-đun Nước Uống Tự Động đã được triển khai thành công dựa trên kiến trúc IoT thực tế.

Hệ thống hoạt động theo thời gian thực và hoàn toàn tự động dựa trên dữ liệu mực nước.

Tích hợp đầy đủ 3 chế độ:

* Giám sát
* Điều khiển tự động
* Điều khiển thủ công qua Dashboard

Kết quả mô phỏng thực tế:

|  |  |  |
| --- | --- | --- |
| **Chức năng** | **Trạng thái** | **Ghi chú** |
| Publish dữ liệu mực nước lên MQTT | Hoạt động ổn định | Python mô phỏng thiết bị |
| Dashboard hiển thị gauge & biểu đồ | Hoàn thiện | Cập nhật realtime |
| Tự động bật/tắt bơm | Chính xác theo ngưỡng | <30% bật / ≥90% tắt |
| Điều khiển thủ công | Hoạt động tốt | Người dùng bật/tắt trực tiếp từ giao diện |
| Vòng IoT khép kín | Hoạt động hoàn chỉnh | Thiết bị ↔ MQTT ↔ Node-RED |