**Slide 1: Tiêu đề và Giới thiệu**

Tiêu đề: "Hệ thống IoT Giám sát và Điều khiển Nông trại với ESP32 và Raspberry Pi"

Thông tin:

Họ tên sinh viên, mã số sinh viên.

Lớp và môn học: "IoT".

Giảng viên hướng dẫn.

**Slide 2: Tóm tắt nội dung**

Nội dung chính: Tóm tắt mục tiêu của đồ án:

Thiết kế hệ thống IoT giám sát và điều khiển môi trường nông trại nhà kính sử dụng mạng cục bộ.

Thu thập dữ liệu từ các cảm biến và lưu trữ trên Raspberry Pi.

Điều khiển thiết bị qua internet.

**Slide 3: Mục tiêu đồ án**

Các mục tiêu chính:

Xây dựng hệ thống giám sát nhiệt độ, độ ẩm, chất lượng không khí và mực nước.

Điều khiển thiết bị từ xa (quạt, bơm nước).

Gửi dữ liệu cảm biến tới server và lưu trữ để phân tích.

Hiển thị thông tin thời gian thực qua LCD.

Cung cấp API RESTful cho việc tích hợp và điều khiển.

**Slide 4: Kiến trúc tổng quan hệ thống**

Hình vẽ kiến trúc:

ESP32:

Đọc dữ liệu cảm biến (DHT11, MQ135, cảm biến mực nước).

Điều khiển relay.

Gửi dữ liệu đến server.

Raspberry Pi:

Server nhận dữ liệu từ ESP32.

Lưu dữ liệu vào SQLite.

Hiển thị thông tin lên LCD.

Điều khiển relay qua API.

Các thành phần giao tiếp:

Wi-Fi kết nối ESP32 và Raspberry Pi.

HTTP POST/GET để truyền dữ liệu.

**Slide 5: Phần cứng**

Các thành phần phần cứng sử dụng:

ESP32 (chip IoT chính).

Raspberry Pi 3/4 (server lưu trữ và xử lý dữ liệu).

Cảm biến:

DHT11: Đo nhiệt độ và độ ẩm.

MQ135: Đo chất lượng không khí.

Cảm biến mực nước: Giám sát trạng thái mực nước.

Relay (điều khiển thiết bị).

LCD 1602 (hiển thị dữ liệu).

Nguồn cấp và dây nối.

**Slide 6: Phần mềm**

ESP32:

Lập trình bằng Arduino IDE.

Sử dụng thư viện:

WiFi.h: Kết nối Wi-Fi.

DHT.h: Đọc dữ liệu từ cảm biến DHT11.

ArduinoJson: Xử lý JSON.

HTTPClient.h: Gửi HTTP POST đến server.

Raspberry Pi:

Lập trình bằng C.

Thư viện:

Mongoose: Tạo HTTP server.

cJSON: Xử lý JSON.

sqlite3.h: Lưu dữ liệu vào SQLite.

lcd1602.h: Điều khiển LCD.

**Slide 7: Chức năng ESP32**

Kết nối Wi-Fi:

ESP32 kết nối với mạng Wi-Fi, lấy địa chỉ IP.

Đọc dữ liệu cảm biến:

DHT11 (nhiệt độ, độ ẩm), MQ135 (ppm), cảm biến mực nước.

Gửi dữ liệu tới SEVER:

HTTP POST với payload JSON chứa:

json

Sao chép mã

{

"temperature": 30.5,

"humidity": 60.2,

"airQuality": 200,

"waterLevel": 1

}

Điều khiển relay: Qua API /control.

**Slide 8: Chức năng Raspberry Pi**

Server HTTP:

Endpoint /data: Nhận dữ liệu từ ESP32.

Endpoint /data\_endpoint: Trả về dữ liệu cảm biến mới nhất.

Xử lý và lưu trữ dữ liệu:

Lưu dữ liệu vào SQLite.

Hiển thị thông tin lên LCD:

Dữ liệu cảm biến được hiển thị real-time.

Điều khiển relay qua API.

**Slide 9: Tích hợp mô hình OSI**

Mô hình OSI trong dự án:

Tầng 1: Sóng Wi-Fi truyền dữ liệu.

Tầng 3: Định tuyến qua địa chỉ IP.

Tầng 4: Giao tiếp qua TCP.

Tầng 6: Dữ liệu JSON được trình bày.

Tầng 7: REST API được cung cấp bởi server.

**Slide 10: Mô tả giao tiếp REST API**

Các endpoint API:

/data (POST): ESP32 gửi dữ liệu.

/data\_endpoint (GET): Lấy dữ liệu mới nhất.

/control?state=on|off (GET): Điều khiển relay.

Định dạng JSON:

Gửi dữ liệu từ ESP32:

json

Sao chép mã

{

"temperature": 25.4,

"humidity": 70.5,

"airQuality": 180,

"waterLevel": 1

}

Phản hồi từ server:

json

Sao chép mã

{

"status": "success"

}

**Slide 11: Cơ sở dữ liệu**

Cấu trúc SQLite:

Tạo bảng environment:

sql

Sao chép mã

CREATE TABLE environment (

id INTEGER PRIMARY KEY AUTOINCREMENT,

temperature REAL,

humidity REAL,

air\_quality INTEGER,

water\_level INTEGER,

timestamp DATETIME DEFAULT CURRENT\_TIMESTAMP

);

Ví dụ dữ liệu:

ID Temperature Humidity Air Quality Water Level Timestamp

1 30.5 60.2 200 1 2024-12-06 12:00:00

Slide 11: Kết quả đạt được

ESP32:

Đọc dữ liệu cảm biến chính xác.

Gửi dữ liệu tới server thành công.

Raspberry Pi:

Lưu trữ dữ liệu ổn định trong SQLite.

Điều khiển relay theo yêu cầu.

Hiển thị dữ liệu cảm biến trên LCD.

**Slide 12: Thử nghiệm thực tế**

Demo hệ thống:

ESP32 đọc cảm biến và gửi dữ liệu.

Raspberry Pi nhận dữ liệu, hiển thị trên LCD.

Điều khiển relay qua API /control.

**Slide 13: Những khó khăn**

Kỹ thuật:

Kết nối Wi-Fi không ổn định.

Đọc giá trị từ MQ135 cần hiệu chỉnh kỹ lưỡng.

Tích hợp:

Định dạng JSON đúng chuẩn để tránh lỗi giao tiếp.

Thời gian:

Lập trình và kiểm thử các thành phần tốn nhiều thời gian.

**Slide 14: Hướng phát triển**

Nâng cấp phần cứng:

Sử dụng cảm biến chính xác hơn (DHT22, cảm biến siêu âm).

Mở rộng chức năng:

Tích hợp đám mây (Firebase, AWS IoT).

Gửi cảnh báo qua SMS hoặc email khi vượt ngưỡng.

Bảo mật:

Thêm xác thực API và mã hóa HTTPS.

**Slide 15: Kết luận**

Kết quả:

Hệ thống hoạt động ổn định, đáp ứng yêu cầu giám sát và điều khiển.

Ứng dụng thực tế:

Nông nghiệp thông minh, tự động hóa.

Mở rộng cho các hệ thống IoT khác.

**Mẹo trình bày**

Slide demo thực tế:

Chuẩn bị một video ngắn mô tả hệ thống hoạt động (Mai lên quay).

Trình diễn trực tiếp nếu có điều kiện.

Tương tác:

Mời giảng viên và các bạn điều khiển relay qua API để tăng tính hấp dẫn.

Thời gian:

Dành 15 phút thuyết trình, 5 phút trả lời câu hỏi.