fit@hcmus

Báo cáo Đồ án cuối kỳ

Đề tài: Smart Lighting System (Phiên bản cải tiến)

Sinh viên thực hiện:

Trần Hiếu Tâm (21127423)

Giảng viên hướng dẫn:

TS. Nguyễn Đức Hoàng Hạ

ThS. Đỗ Thị Thanh Hà

TP. Hồ Chí Minh, tháng 12 năm 2024

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN CUỐI KỲ

Đề tài: Smart Lighting System (Phiên bản cải tiến)

Học phần: Nhập môn lập trình kết nối vạn vật - 21KHDL

Sinh viên thực hiện: Trần Hiếu Tâm (21127423)
Giảng viên hướng dẫn: TS. Nguyễn Đức Hoàng Hạ
ThS. Đỗ Thị Thanh Hà

TP. Hồ Chí Minh, tháng 12 năm 2024

Lời mở đầu

Đồ án này là một phần của học phần *Nhập môn lập trình kết nối vạn vật* - lớp 21KHDL, được tổ chức tại Khoa Công nghệ Thông tin, Trường Đại học Khoa học Tự nhiên, ĐHQG-HCM nhằm mục tiêu học tập, ứng dụng kiến thức IoT với tính chất là một đồ án phi thương mại.

Em xin chân thành cảm ơn TS. Nguyễn Đức Hoàng Hạ, ThS. Đỗ Thị Thanh Hà, giảng viên Khoa Công nghệ Thông tin đã tận tình tư vấn, hỗ trợ về mặt kiến thức và định hướng, giúp cá nhân em hiểu rõ hơn về các nguyên lý và kỹ thuật cần thiết để hoàn thành đồ án này. Đồng thời xin cảm ơn bạn Trần Lê Kim Ngân, sinh viên Khoa Điện tử Viễn thông, đã hỗ trợ nhiệt tình các linh kiện và vật liệu dựng mô hình cần thiết, góp phần quan trọng vào thành công của đồ án.

Sự hỗ trợ quý báu này là động lực quan trọng để bản thân em có thể hoàn thành dự án một cách tốt nhất. Mọi góp ý xin liên hệ qua email thtam21@clc.fitus.edu.vn.

Trân trọng,

Trần Hiếu Tâm

Mục lục

1 Đặc tả yêu cầu **1**

2 Thiết kế hệ thống **3**

2.1 Thiết kế tổng quát 3

2.2 Thiết kế chi tiết 5

2.3 Sơ đồ mạch điện đề xuất 5

2.4 Sơ đồ máy trạng thái hữu hạn (FSM - Finite-state Machine) 6

2.5 Sơ đồ truyền nhận dữ liệu 9

2.5.1 Truyền dữ liệu, tín hiệu giữa Controller và các thiết bị 10

2.5.2 Truyền dữ liệu, tín hiệu từ Controller đến Web Service 11

2.5.3 Truyền dữ liệu, tín hiệu từ Web Service đến Controller 12

3 Đề xuất các giải pháp **13**

4 Kế hoạch kiểm thử **16**

4.1 Kiểm tra từng thành phần 16

4.2 Kiểm tra tích hợp 17

5 Kết quả triển khai **18**

5.1 Giả lập hệ thống trên Wokwi 18

5.2 Triển khai hệ thống trên thiết bị thật 19

5.3 Giới thiệu chức năng website 20

5.4 Release 22

Tài liệu tham khảo **26**

Danh sách bảng

2 Bảng chuyển đổi trạng thái của đèn LED trong hệ thống 8
3 Các thiết bị truyền dữ liệu, tín hiệu đến Controller 10
4 Các thiết bị nhận tín hiệu từ Controller 10
5 Danh sách đề xuất các giải pháp cho hệ thống 15
1 Danh sách các thành phần của hệ thống 23
6 Kiểm tra các thành phần hệ thống 24
7 Các bước tích hợp và kiểm tra hệ thống 25

Danh sách hình vẽ

1 System Design 3
2 Sơ đồ mạch điện dự kiến triển khai hệ thống 5
3 Sơ đồ máy trạng thái hữu hạn của đèn LED trong hệ thống 6
4 Sơ đồ truyền nhận dữ liệu giữa các thiết bị, Controller và Web Server 9
5 Triển khai mạch ESP32 trên Wokwi 18
6 Triển khai hệ thống thực tế 19
7 Giao diện website 20
8 Giao diện trang web theo dõi MQTT 21
9 Cấu trúc JSON lưu trữ lịch sử nhắn tin 21

1 Đặc tả yêu cầu

Hệ thống điều khiển ánh sáng thông minh (tiếng Anh: Smart Lighting Control System) là một hệ thống tự động điều chỉnh đèn dựa trên điều kiện ánh sáng xung quanh và hỗ trợ người dùng điều khiển thủ công khi cần thiết nhằm tiết kiệm năng lượng điện cũng như có thể vận hành hiệu quả, chính xác mà không cần sự can thiệp của con người.

Yêu cầu:

- Hệ thống gồm 01 đèn LED, 01 cảm biến ánh sáng dùng để điều khiển tự động đèn LED theo điều kiện ánh sáng xung quanh (Auto Mode), 01 nút bấm dùng để điều khiển thủ công (Manual Mode) và 01 bộ vi điều khiển trên chip nhằm kết nối và điều khiển các thiết bị trên cũng như kết nối Internet.

- Hệ thống vận hành theo 2 chế độ (mode): Auto Mode và Manual Mode. Khi bắt đầu chạy thì hệ thống sẽ chạy theo Auto Mode: Đèn LED sẽ tự động tắt (Off) khi trời sáng và sẽ tự động sáng nhấp nháy (Blinking) khi trời tối. Khi nhấn vào nút bấm thì hệ thống sẽ chuyển từ đổi từ Auto Mode sang Manual Mode: Đèn sẽ tạm thời không vận hành tự động (Auto Mode) theo điều kiện ánh sáng môi trường. Theo đó, nếu người dùng nhấn nút bấm thì đèn sẽ tắt nếu đèn đang sáng và ngược lại, nếu người dùng nhấn nút bấm thì đèn sẽ bật nếu hiện tại đèn đang tắt. Nếu người dùng nhấn nút khi đèn đang blinking thì đèn sẽ tắt. Nếu hệ thống đang trong Manual Mode, khi có sự chuyển đổi ánh sáng ngày đêm thì hệ thống sẽ mặc định chuyển về Auto Mode và đèn LED sẽ vận hành tự động theo điều kiện ánh sáng môi trường. Ngoài ra, khi hệ thống đang trong Manual Mode mà sau N giây người dùng không nhấn nút bấm điều khiển bật tắt đèn thủ công thì hệ thống sẽ tự động chuyển sang Auto Mode.

- Hệ thống sẽ hỗ trợ kết nối Internet qua kết nối không dây WiFi để điều khiển từ xa thông qua website để người dùng theo dõi theo thời gian thực trạng thái đèn hiện tại (Bật, Tắt, Blinking), mode (Auto Mode hay Manual Mode), điều kiện ánh sáng hiện tại (Giá trị hiện tại của cảm biến ánh sáng để xác định Light hay Night), điều kiện thời gian N để chuyển mode cho hệ thống. Ngoài ra, người dùng có thể bật, tắt điều khiển đèn từ xa thông qua Website này bằng cách nhập văn bản hoặc bằng giọng nói yêu cầu cho hệ thống và các tin nhắn sẽ được lưu lại kèm thời gian nhắn của tin nhắn. Giao tiếp giữa hệ thống kết nối không dây WiFi và website thông qua giao thức MQTT.

Ràng buộc:

- Hệ thống mô phỏng chỉ được thử nghiệm trên nguồn điện thấp (5-10V), không sử dụng điện dân dụng (220V) để thử nghiệm và vận hành hệ thống này.
- Hệ thống không được sử dụng các loại cảm biến thu thập dữ liệu riêng tư của con người dưới mọi hình thức.
- Hệ thống luôn phải được cung cấp điện liên tục khi vận hành và nên được kết nối Internet liên tục để có thể theo dõi trạng thái và điều khiển từ xa thông qua website.
- Thiết bị và linh kiện nên phổ biến ở thị trường Việt Nam, có thể dễ dàng tìm kiếm trên địa bàn Thành phố Hồ Chí Minh. Đồng thời ưu tiên sử dụng các thiết bị và linh kiện mà Wokwi có hỗ trợ mô phỏng.
- Chi phí không vượt quá 200.000 VNĐ (bằng chữ: Hai trăm nghìn Việt Nam Đồng chẵn)

2 Thiết kế hệ thống

2.1 Thiết kế tổng quát

Hình 1: System Design

Block Diagram của hệ thống được mô tả qua hình 1 cho biết thiết kế tổng quát của đồ án này. Theo đó, hệ thống gồm các thành phần sau: (1) Các thiết bị Input là Pushbutton (tên tiếng Việt: nút nhấn) để nhận tín hiệu bật tắt đèn của người dùng và Photoresistor (tên tiếng Việt: quang trở) để đo lường độ sáng của môi trường xung quanh hệ thống; (2) Thiết bị Output là đèn LED; (3) Controller đóng vai trò là thiết bị để nhận dữ liệu từ thiết bị Input, điều khiển thiết bị Output cũng như truyền và nhận không dây dữ liệu với Web Service; (4) Web Service là một ứng dụng web được triển khai nhằm mục đích cho người dùng biết được trạng thái hiện tại của đèn LED cũng như điều khiển đèn LED từ xa thông qua nút trên Website; (5) MQTT có vai trò là một giao thức

trung gian giữa Controller và Website (cùng các thao tác, dịch vụ khác trên Internet nếu có); (6) LLMs (Generative AI) đóng vai trò là công cụ hỗ trợ để phân tích yêu cầu của người dùng thành lệnh điều khiển hệ thống và phản hồi người dùng. Các thành phần trên sẽ được mô tả chi tiết tại bảng 1.

2.2 Thiết kế chi tiết

Bảng 1 cho biết mô tả chi tiết về các thành phần Controller, Web Service, Pushbutton, Photoresistor, đèn LED, MQTT, LLMs (Generative AI) của hệ thống đã được thiết kế trên System Design ở hình 1. Theo đó, bảng 1 đưa ra đề xuất thiết bị, công nghệ phù hợp cũng như cách thiết lập các thiết bị đó để có cơ sở thiết kế chi tiết và triển khai ở các giai đoạn sau (như thiết kế sơ đồ mạch điện, FSM, sơ đồ truyền nhận dữ liệu, triển khai giả lập và triển khai trên thiết bị thật).

2.3 Sơ đồ mạch điện đề xuất

Hình 2: Sơ đồ mạch điện dự kiến triển khai hệ thống

Sơ đồ mạch điện trên hình 2 mô tả một mạch bao gồm một nút nhấn (Pushbutton), một quang trở (Photoresistor) và một đèn LED được kết nối với các chân của vi điều khiển Kit Wifi BLE ESP32 NodeMCU-32S CH340 Ai-Thinker (sau đây gọi tắt là ESP32). Các chân ESP32 được sử dụng bao gồm: chân nguồn 5V và GND; các chân esp:14, esp:34, esp:35. Nguồn điện 5V được cung cấp cho

mạch qua chân 5V và chân GND. Khi nút nhấn được bấm, mạch cho phép dòng điện đi qua các điện trở và quang trở đến chân esp:35 để đo ánh sáng từ quang trở. Chân esp:34 được sử dụng để theo dõi trạng thái của nút nhấn và chân esp:14 được sử dụng để điều khiển đèn LED. Đèn LED, quang trở và nút nhấn được mắc nối tiếp qua các điện trở để hạn chế dòng điện nhằm bảo vệ các thiết bị trên khỏi dòng điện quá lớn và đảm bảo mạch hoạt động ổn định.

2.4 Sơ đồ máy trạng thái hữu hạn (FSM - Finite-state Machine)

Hình 3: Sơ đồ máy trạng thái hữu hạn của đèn LED trong hệ thống

Sơ đồ FSM (Finite State Machine) ở hình 3 mô tả hệ thống điều khiển đèn với hai chế độ: AutoMode và ManualMode. Khi ESP32 được nạp điện (Power ON), chế độ AutoMode được kích hoạt. Trong AutoMode, nếu là ban ngày (Day), đèn sẽ ở trạng thái OFF; nếu chuyển sang ban đêm (Night), đèn sẽ chuyển sang trạng thái BLINKING. Khi trời sáng trở lại (Night sang Day), đèn sẽ quay về trạng thái OFF và ngược lại, khi trời chuyển từ sáng sang tối (Day sang Night) thì đèn sẽ sang trạng thái BLINKING. Khi người dùng nhấn nút, hệ thống chuyển sang ManualMode.

Trong ManualMode, đèn có thể được bật (ON) hoặc tắt (OFF) thông qua nút nhấn. Ở trạng thái

ON, nếu nhấn nút lần nữa, đèn sẽ chuyển sang OFF và ngược lại. Hệ thống sẽ quay lại AutoMode khi có sự chuyển đổi từ ngày sang đêm, đêm sang ngày hoặc khi hết thời gian quy định (timeout). Ở bất kỳ trạng thái nào, hệ thống sẽ kết thúc hoạt động khi bị ngắt nguồn (tức là Shut Down).

Sự chuyển đổi trạng thái của sơ đồ FSM ở hình 3 được mô tả thông qua bảng 2: Bảng chuyển đổi trạng thái của đèn LED trong hệ thống.

Current State	**Input**	**Condition**	**Next State**	**Output**
Initial State	Power ON	Current Condition == Day	AutoMode - OFF	Start AutoMode
Initial State	Power ON	Current Condition == Night	AutoMode - BLINKING	Start blinking LED with AutoMode
AutoMode - OFF	Day → Night Transition	-	AutoMode - BLINKING	Start blinking LED
AutoMode - BLINKING	Night → Day Transition	-	AutoMode - OFF	Turn off LED
AutoMode - OFF	Press Button	-	ManualMode - ON	Switch to ManualMode and turn on LED
AutoMode - BLINKING	Press Button	-	ManualMode - OFF	Switch to ManualMode and turn off LED
ManualMode - ON	Press Button	-	ManualMode - OFF	Turn off LED
ManualMode - OFF	Press Button	-	ManualMode - ON	Turn on LED
ManualMode - ON	Day-Night Transition	Current Condition == Day	AutoMode - OFF	Switch to AutoMode and turn off LED
ManualMode - ON	Day-Night Transition	Current Condition == Night	AutoMode - BLINKING	Switch to AutoMode and start blinking LED
ManualMode - ON	Timeout	Current Condition == Day	AutoMode - OFF	Switch to AutoMode and turn off LED
ManualMode - ON	Timeout	Current Condition == Night	AutoMode - BLINKING	Switch to AutoMode and start blinking LED
ManualMode - OFF	Day-Night Transition	Current Condition == Day	AutoMode - OFF	Switch to AutoMode and turn off LED
ManualMode - OFF	Day-Night Transition	Current Condition == Night	AutoMode - BLINKING	Switch to AutoMode and start blinking LED
ManualMode - OFF	Timeout	Current Condition == Day	AutoMode - OFF	Switch to AutoMode and turn off LED
ManualMode - OFF	Timeout	Current Condition == Night	AutoMode - BLINKING	Switch to AutoMode and start blinking LED
AutoMode - OFF	Shut Down	-	Shutdown State	Shut Down System
AutoMode - BLINKING	Shut Down	-	Shutdown State	Shut Down System and turn off LED
ManualMode - ON	Shut Down	-	Shutdown State	Shut Down System and turn off LED
ManualMode - OFF	Shut Down	-	Shutdown State	Shut Down System

Bảng 2: Bảng chuyển đổi trạng thái của đèn LED trong hệ thống

Căn cứ FSM trên hình 3, bảng 2 ở trên mô tả chuyển đổi trạng thái của đèn LED trong hệ thống.

2.5 Sơ đồ truyền nhận dữ liệu

Hình 4: Sơ đồ truyền nhận dữ liệu giữa các thiết bị, Controller và Web Server

Sơ đồ tại hình 4 mô tả quá trình truyền nhận dữ liệu giữa ESP32 và Web Service qua giao thức MQTT bằng truyền dữ liệu không dây WiFi. ESP32 đọc giá trị từ quang trở (qua cổng esp:35), nút nhấn (qua cổng esp:34) và căn cứ vào các dữ liệu đó để điều khiển đèn LED (qua cổng esp:14). Trong quá trình xử lý, ESP32 gửi liên tục các thông tin hiện thời về thời gian timeout (đơn vị: second), chế độ đang vận hành hệ thống (AutoMode hay ManualMode), điều kiện ánh sáng môi trường (chỉ số quang trở tương ứng Day hay Night) và trạng thái đèn (đang tắt - OFF hay đang sáng - ON hay đèn có đang Blinking không) đến Web Service thông qua giao thức MQTT (Topic: esp32/data) qua truyền không dây WiFi dưới dạng 1 JSON. Ở chiều ngược lại, Web Service cũng có thể gửi lệnh điều khiển dưới hình thức người dùng thao tác trên Website (nhập text hoặc voice) và Web Service sẽ gọi LLMs (Generative AI) để xử lý thành lệnh và phản hồi người dùng và gửi tới ESP32 (vẫn thông qua topic esp32/data của MQTT) để điều khiển hệ thống. Việc truyền nhận dữ liệu giữa ESP32 và Web Service thông qua MQTT và WiFi được diễn ra liên tục nhằm cập nhật chính xác nhất trạng thái hiện thời của hệ thống.

Sơ đồ tại hình 4 được mô tả chi tiết và rõ ràng hơn thông qua các phần 2.5.1, 2.5.2, 2.5.3.

2.5.1 Truyền dữ liệu, tín hiệu giữa Controller và các thiết bị

Thiết bị	**Pin giao tiếp**	**Điện thế**	**Giá trị**	**Ý nghĩa**
Photoresistor	esp:35	0V đến 5V	0 đến 1023	Cường độ ánh sáng của môi trường xung quanh hệ thống
Pushbutton	esp:34	0V	LOW	Released
		5V	HIGH	Pressed

Bảng 3: Các thiết bị truyền dữ liệu, tín hiệu đến Controller

Thiết bị	**Pin giao tiếp**	**Điện thế**	**Giá trị**	**Ý nghĩa**
LED	esp:14	5V	HIGH	ON: tức bật sáng đèn LED
		0V	LOW	OFF: tức tắt đèn LED

Bảng 4: Các thiết bị nhận tín hiệu từ Controller

Người đọc có thể tham khảo sơ đồ mạch điện trên hình 2, sơ đồ truyện nhận dữ liệu trên hình 4 và mạch giả lập hệ thống trên Wokwi tại hình 5 để có góc nhìn rộng hơn về các thiết bị này và các pin liên quan được đề cập trong bảng 3 và bảng 4.

2.5.2 Truyền dữ liệu, tín hiệu từ Controller đến Web Service

Cấu trúc JSON:

```
{"delay_duration":<int>,"is_day":<bool>,"is_auto_mode":<bool>,"led_state":<int
    >,"light_sensor":<int>,"ldr_threshold":<int>}
```

Trường JSON được gửi liên tục đến Topic esp32/data của HiveMQ MQTT bao gồm các thông tin cần thiết để hệ thống điều khiển và giám sát trạng thái hoạt động của đèn LED thông minh:

- `delay_duration`: Thời gian chờ trước khi thực hiện thay đổi trạng thái đèn hoặc chuyển chế độ từ Manual Mode sang Auto Mode nếu không có thao tác từ người dùng. Giá trị này được gửi bằng đơn vị giây (giá trị nguyên).
- `is_day`: Xác định điều kiện ánh sáng môi trường xung quanh (sáng hoặc tối). Giá trị true biểu thị ban ngày, và false biểu thị ban đêm.
- `is_auto_mode`: Chế độ hiện tại của hệ thống. Giá trị true biểu thị chế độ tự động (Auto Mode), và false biểu thị chế độ thủ công (Manual Mode).
- `led_state`: Trạng thái hiện tại của đèn LED. Giá trị 0 biểu thị đèn tắt (Off), 1 biểu thị đèn sáng (On), và 2 biểu thị đèn nhấp nháy (Blinking).
- `light_sensor`: Giá trị ánh sáng đọc được từ cảm biến, giúp hệ thống phân biệt giữa ngày và đêm.
- `ldr_threshold`: Giá trị ngưỡng ánh sáng cài đặt để xác định ngày hay đêm. Nếu giá trị ánh sáng từ cảm biến lớn hơn hoặc bằng ngưỡng này, hệ thống xác định là ban đêm; ngược lại là ban ngày.

2.5.3 Truyền dữ liệu, tín hiệu từ Web Service đến Controller

Cấu trúc JSON:

```
{"push_button":<bool>}
```

Trường JSON này được gửi liên tục đến Topic esp32/data của HiveMQ MQTT. Theo đó dữ liệu này được phân tích dựa trên user_request (dạng text hoặc text sau khi Speech to Text voice của người dùng) mà LLMs phân tích. LLMs sẽ gửi xem là yêu cầu của người dùng cần nhấn nút hay không, cũng như tạo response để phản hồi cho người dùng.

3 Đề xuất các giải pháp

Component /Module	**Giải pháp 1**	**Giải pháp 2**	**Giải pháp 3**	**Quyết định giải pháp**
Controller	**Arduino Uno R3 ATmega328** - Chi phí: ∼100.000 VNĐ - Dễ lập trình và mở rộng - Phù hợp cho các hệ thống cơ bản	**Raspberry Pi 3B+** - Chi phí: >1.890.000 VNĐ - Có khả năng xử lý cao hơn - Hỗ trợ hệ điều hành - Phù hợp cho các ứng dụng phức tạp	**Kit Wifi BLE ESP32 NodeMCU-32S CH340 Ai-Thinker** - Chi phí: ∼190.000 VNĐ - Tích hợp sẵn Wifi và Bluetooth - Phù hợp cho IoT và hệ thống có kết nối mạng	**Kit Wifi BLE ESP32 NodeMCU-32S CH340 Ai-Thinker** Đáp ứng đủ yêu cầu, chi phí hợp lý, dễ sử dụng.
Web Service	**Local Host (Panel Python)** - Triển khai cục bộ trên máy tính - Chi phí: Miễn phí - Thích hợp cho giai đoạn thử nghiệm	**Cloud Hosting** - Dùng nền tảng đám mây (AWS, Azure) - Chi phí: phí duy trì hàng tháng - Khả năng mở rộng tốt - Truy cập từ xa tiện lợi	**Raspberry Pi làm Server** - Kết nối trực tiếp với cảm biến - Chi phí thiết lập cao hơn - Phù hợp cho các hệ thống nhỏ gọn, tự chủ	**Local Host (Panel Python)** Chi phí thấp và đáp ứng được yêu cầu cơ bản của đồ án.

Component /Module	**Giải pháp 1**	**Giải pháp 2**	**Giải pháp 3**	**Quyết định giải pháp**
Pushbutton	**6x6x5 mm Tactile Push Button** - Chi phí thấp - Đơn giản, dễ lắp đặt	**Công tắc cảm ứng chạm** - Chi phí cao hơn (khoảng 20.000 VNĐ) - Tạo trải nghiệm hiện đại - Có thể nhúng vào thiết kế đẹp mắt	**Công tắc nhấn IP67** - Chống nước và bụi, phù hợp môi trường ngoài trời - Chi phí cao hơn (khoảng 20.000 VNĐ) - Độ bền cao	**6x6x5 mm Tactile Push Button** Đơn giản, chi phí thấp, dễ sử dụng.
Photoresistor	**Lm393 Optical Photosensitive LDR** - Chi phí thấp (khoảng 20.000 VNĐ) - Độ nhạy cao với ánh sáng - Dễ kết nối với Arduino	**Cảm biến ánh sáng BH1750** - Chi phí cao hơn (∼40.000 VNĐ) - Độ chính xác cao hơn - Kết nối qua giao thức I2C	**Cảm biến TSL2561** - Chi phí khoảng 140.000 VNĐ - Đo ánh sáng với độ chính xác cao, dải rộng - Phù hợp cho các dự án cần đo ánh sáng chính xác	**Lm393 Optical Photosensitive LDR** Đáp ứng yêu cầu chi phí thấp và dễ sử dụng trong môi trường ánh sáng đơn giản.

Component /Module	**Giải pháp 1**	**Giải pháp 2**	**Giải pháp 3**	**Quyết định giải pháp**
LED	**5mm Color LED** - Đơn giản, dễ lắp đặt - Chi phí thấp (∼9.000 VNĐ/30 đèn)	**LED RGB** - Đổi màu, tạo hiệu ứng - Chi phí cao hơn (∼16.000 VNĐ) - Phù hợp nếu cần thay đổi màu sắc	**Grove - LED Button** - Có tích hợp sẵn kèm theo với Push-button. - Tuổi thọ sử dụng cao hơn (100.000 lần bật tắt) - Chi phí cao hơn (∼66.000 VNĐ)	**5mm Color LED** Chi phí thấp, đáp ứng yêu cầu chiếu sáng cơ bản.

Bảng 5: Danh sách đề xuất các giải pháp cho hệ thống

Bảng 5 đưa ra mô tả các giải pháp khả thi cho các Component/Module của hệ thống. Việc đa dạng hóa các giải pháp giúp hệ thống có nhiều phương án dự phòng, nhiều phương án phát triển hệ thống theo các hướng đi mới hiệu quả hơn và có khả năng khắc phục được những hạn chế còn tồn đọng (nếu có). Cần căn cứ vào Đặc tả yêu cầu và những ràng buộc, hạn chế cũng như bản thiết kế hệ thống để có thể đưa ra được *Quyết định giải pháp* hợp lý nhất cho quá trình phát triển hệ thống ở thời điểm hiện tại. Các giải pháp có chi phí thấp, dễ mua, dễ thiết lập và sử dụng, có phiên bản giả lập, đáp ứng đầy đủ ở mức cơ bản mà đặc tả yêu cầu đề ra là những giải pháp được chọn hiện tại dựa trên cơ sở như còn gặp nhiều hạn chế về kinh phí và trình độ hiện tại của lập trình viên hệ thống.

4 Kế hoạch kiểm thử

4.1 Kiểm tra từng thành phần

Quy trình kiểm tra các thành phần hệ thống được mô tả ở bảng 6 phản ánh sự cần thiết trong việc kiểm thử độc lập từng thiết bị để xác định khả năng sử dụng hiện tại của các thiết bị cũng như tính đúng đắn của dữ liệu mà các thiết bị này trả về. Đây là bước quan trọng trong việc xác định lỗi (nếu có) đến từ thiết bị nào để có phương án khắc phục phù hợp về phần mềm và phần cứng. Ngoài ra, việc kiểm thử này cũng giúp lập trình viên hệ thống có thể cập nhật thông số môi trường sẽ vận hành hệ thống để có sự hiệu chỉnh thông số trong phần mềm một cách phù hợp nhất. Việc kiểm tra từng thành phần này sẽ cần được triển khai cho tất cả phiên bản của hệ thống (trong trường hợp này là phiên bản giả lập và phiên bản triển khai thực tế).

4.2 Kiểm tra tích hợp

Quy trình kiểm tra tích hợp hệ thống thể hiện qua bảng 7 nhằm kiểm tra xem việc vận hành giữa các thành phần với nhau đúng với đặc tả yêu cầu và mô tả hệ thống đặt ra. Quy trình này đảm bảo rằng tất cả kịch bản kiểm thử sẽ bao phủ càng toàn diện càng tốt các yêu cầu mà đặc tả đã đề cập đến. Việc kiểm tra tích hợp này sẽ cần được triển khai cho tất cả phiên bản của hệ thống (trong trường hợp này là phiên bản giả lập và phiên bản triển khai thực tế).

5 Kết quả triển khai

5.1 Giả lập hệ thống trên Wokwi

Hình 5: Triển khai mạch ESP32 trên Wokwi

Hệ thống được giả lập trên Wokwi được mô tả như hình 5 này sử dụng ESP32 và bao gồm các thành phần: cảm biến ánh sáng (LDR), đèn LED và nút nhấn. Cảm biến ánh sáng được kết nối với cổng esp:35 của ESP32 thông qua một điện trở cho phép ESP32 đo mức điện áp thay đổi theo cường độ ánh sáng xung quanh. Đèn LED được điều khiển thông qua cổng esp:14 của ESP32, có thể phản hồi lại theo tín hiệu từ cảm biến ánh sáng hoặc tín hiệu từ nút nhấn. Nút nhấn cho phép người dùng chuyển chế độ vận hành hệ thống từ AutoMode sang ManualMode và điều khiển đèn LED bật/tắt một cách thủ công. Thông qua đó ta thấy hệ thống phiên bản giả lập trên Wokwi giải quyết tương đối đầy đủ những yêu cầu mà đặc tả và bản thiết kế hệ thống đã đề ra. Ngoài ra, người dùng có thể kết nối với WiFi giả lập của Wokwi để kết nối MQTT và truyền dữ liệu lên Internet.

5.2 Triển khai hệ thống trên thiết bị thật

Hình 6: Triển khai hệ thống thực tế

Hệ thống thực tế trên hình 6 được bố trí và lắp đặt tương tự như triển khai giả lập trên Wokwi ở hình 5. Mạch được cung cấp nguồn và nạp code bằng USB Port nối với Laptop và sẽ truyền nhận dữ liệu giữa mạch và Laptop thông qua kết nối không dây WiFi đến MQTT.

5.3 Giới thiệu chức năng website

Website gồm có các chức năng sau:

- Theo dõi trạng thái đèn bao gồm Off, On và Blinking.
- Theo dõi chế độ vận hành hệ thống hiện tại (Auto Mode hay Manual Mode thông qua màu sắc của Delay Duration), trạng thái ánh sáng môi trường xung quanh hệ thống (Day hay Night thông qua màu sắc của Light Sensor), timeout từ Manual Mode sang Auto Mode bao nhiêu Millisecond (nếu hiển thị 0 thì hệ thống sẽ bỏ qua điều kiện timeout này khi vận hành)
- Người dùng có thể dùng lệnh (bằng text hoặc voice) để điều khiển hệ thống từ xa, thông qua giao tiếp MQTT, truyền dữ liệu kết nối không dây WiFi và LLMs (Generative AI) để phân tích yêu cầu người dùng.

Hình 7: Giao diện website

Hình 8: Giao diện trang web theo dõi MQTT

Hình 9: Cấu trúc JSON lưu trữ lịch sử nhắn tin

5.4 Release

```
21127423,Tran Hieu Tam,
https://youtube.com/playlist?list=PL49PFd0rcrSMamwSboGjRe7nyyKdWxkfafeature=shared,
https://drive.google.com/drive/folders/1Dzda6UcX8GyJiqW1w3eEMZk_fvy09f99?usp = sharing
```

Project Information

- **Release Version:** 1.0
- **Release Date:** December 30, 2024
- **Author:** Trần Hiếu Tâm
- **Contact Email:** thtam21@clc.fitus.edu.vn
- **Location:** Ho Chi Minh City, Viet Nam
- **GitHub Repository:** https://github.com/HieuTam/HCMUS-IoT-SmartLightingControlSystem

Copyright

Faculty of Information Technology, VNUHCM-University of Science

TT	Thành phần	Tên thiết bị/Công nghệ đề xuất	Cách thiết lập
1	Controller	Kit Wifi BLE ESP32 NodeMCU-32S CH340 Ai-Thinker	- Nguồn và GND: Nối chân 5V và GND của ESP32 với nguồn điện và GND cho các thiết bị. - Các Pin của ESP32: Nối với các thiết bị như LED và Pushbutton để truyền/nhận tín hiệu điều khiển và nối với cảm biến ánh sáng để đọc tín hiệu ánh sáng.
2	Web Service	Local Host, triển khai bằng Python (Panel)	- Triển khai trên Local Host và kết nối với Controller qua WiFi và trung gian là MQTT. - Gửi/Nhận Dữ liệu: Thiết lập kết nối giữa ESP32 và Web Service thông qua kết nối không dây WiFi với trung gian là giao thức MQTT để trao đổi dữ liệu điều khiển và giám sát trạng thái đèn LED từ Web Service đến ESP32 và ngược lại.
3	Pushbutton	6x6x5 mm Miniature Micro Momentary Tactile Tact Touch Push Button	- Kết nối với ESP32: Nối một chân của Pushbutton với chân có thể đọc tín hiệu Digital của ESP32 và chân còn lại với GND. - Điện trở: Thêm điện trở giữa chân Digital và GND để ổn định tín hiệu, tránh nhiễu. ESP32 sẽ đọc tín hiệu từ Pushbutton để điều khiển đèn LED.
4	Photoresistor	Lm393 Optical Photosensitive Ldr Light Sensor Module	- Kết nối với ESP32: Nối chân tín hiệu của module cảm biến ánh sáng (LDR) với chân có thể đọc tín hiệu Analog In của ESP32. Nối chân VCC và GND của module với chân 5V và GND của ESP32. - Đọc cường độ ánh sáng: ESP32 sẽ đọc giá trị từ LDR để xác định trạng thái sáng tối của môi trường xung quanh.
5	LED	5mm Color Led	- Kết nối với ESP32: Nối chân dương của LED với chân có thể ghi tín hiệu Digital của ESP32 qua một điện trở để bảo vệ LED. Nối chân âm của LED với GND. - Điều khiển Bật/Tắt: ESP32 sẽ điều khiển LED dựa trên tín hiệu từ cảm biến ánh sáng hoặc Pushbutton hoặc lệnh bật tắt từ Website.
6	MQTT	HiveMQ	- Có nhiệm vụ là trung gian nhận tín hiệu từ hệ thống IoT để các dịch vụ khác có thể theo dõi được dữ liệu các thiết bị và gửi tín hiệu điều khiển của người dùng đến hệ thống IoT. - Cần đăng ký tài khoản và topic để tăng tính bảo mật cho dữ liệu.
7	LLMs (Generative AI)	Các mô hình ngôn ngữ lớn	- Có nhiệm vụ tiếp nhận yêu cầu của người dùng (dưới dạng văn bản), phân tích thành lệnh điều khiển thiết bị IoT và tạo văn bản hồi đáp/trả lời người dùng. - Cần đăng ký tài khoản và tạo API key để sử dụng. Cũng lưu ý rằng đối với tài khoản miễn phí thì API key có số quota khá hạn chế nếu có nhu cầu dùng nhiều.

Bảng 1: Danh sách các thành phần của hệ thống

Thành phần	**Kiểm tra**
Controller (Kit Wifi BLE ESP32 NodeMCU-32S CH340 Ai-Thinker)	- Kiểm tra Controller khởi tạo đúng và tất cả các chân (pins) được cấu hình theo thiết kế. - Kiểm tra giao tiếp nối tiếp dữ liệu thông qua kết nối không dây WiFi để đảm bảo khả năng truyền dữ liệu chính xác.
Web Service (Local Host với Panel của Python)	- Kiểm tra các thành phần của chương trình xử lý yêu cầu và trả về kết quả như mong đợi. - Giả lập dữ liệu gửi từ ESP32 và kiểm tra dữ liệu được nhận và hiển thị chính xác. Đồng thời xuất kết quả gửi từ chương trình trên terminal và MQTT để đảm bảo tính chính xác của dữ liệu gửi đến ESP32.
Pushbutton (6x6x5 mm Miniature Micro Momentary Tactile Tact Push Button)	- Xác nhận nút nhấn ghi nhận trạng thái thay đổi (nhấn và thả) và gửi dữ liệu đến Controller.
Photoresistor (Lm393 Optical Photosensitive LDR Light Sensor Module)	- Đo độ nhạy sáng và đảm bảo đọc giá trị analog chính xác. - Kiểm tra khả năng của quang trở để phân biệt giữa ngưỡng sáng ngày và đêm. (Đặc biệt tại môi trường dự kiến triển khai hệ thống)
LED (5mm Color LED)	- Kiểm tra đèn LED phản hồi tín hiệu kỹ thuật số cho các trạng thái BẬT, TẮT và NHẤP NHÁY. - Kiểm tra độ sáng/độ rõ của đèn LED trong điều kiện bình thường.

Bảng 6: Kiểm tra các thành phần hệ thống

Bước	**Tích hợp**	**Kiểm tra**
1	ESP32 + MQTT	- Kiểm tra phản hồi của MQTT khi ESP32 gửi dữ liệu giả lập từ nút nhấn, cảm biến ánh sáng và các dữ liệu kèm theo khác.
2	Web Service + MQTT	- Kiểm tra phản hồi của MQTT khi Web Service gửi dữ liệu giả lập từ nút nhấn, cảm biến ánh sáng và các dữ liệu kèm theo khác.
3	ESP32 + MQTT + Web Service + Quang trở	- Thay đổi điều kiện ánh sáng xung quanh để kích hoạt quang trở và kiểm tra sự thay đổi trạng thái trên MQTT và dịch vụ web.
4	ESP32 + MQTT + Web Service + Quang trở + LED	- Kiểm tra đèn LED BẬT, TẮT hoặc NHẤP NHÁY dựa trên đầu vào từ cảm biến ánh sáng, được ghi nhận trên MQTT và dịch vụ web.
5	ESP32 + MQTT + Web Service + Quang trở + LED + Nút nhấn	- Kiểm tra đèn LED BẬT, TẮT hoặc NHẤP NHÁY dựa trên đầu vào từ nút nhấn, được ghi nhận trên dịch vụ web.
6	Tích hợp toàn bộ hệ thống	- Thực hiện kiểm tra toàn diện cho cả hệ thống khi tất cả các thành phần hoạt động, đảm bảo hành vi FSM khớp với thiết kế (TẮT, BẬT, NHẤP NHÁY). - Xác nhận trạng thái của đèn LED thay đổi chính xác dựa trên cả nút nhấn vật lý và lệnh trên Website cũng như dựa trên đầu vào từ cảm biến ánh sáng, và tất cả dữ liệu được ghi lại trên dịch vụ web. - Xác nhận các thông tin khác hiển thị trên Website được nhận từ ESP32 khớp với thực tế.

Bảng 7: Các bước tích hợp và kiểm tra hệ thống

Tài liệu

[1] Arduino. Arduino Home Page, 2024. https://www.arduino.cc/.

[2] Holoviz Contributors. Build a Monitoring Dashboard with Panel, 2024. https://panel.holoviz.org/tutorials/build_monitoring_dashboard.html.

[3] Holoviz Contributors. Build a Streaming Chat Bot with Panel, 2024. https://panel.holoviz.org/tutorials/build_chat_bot.html.

[4] Holoviz contributors. Panel v1.5.5 documentation, 2024. https://panel.holoviz.org.

[5] krshelar. Light Sensor using Arduino, 2020. https://www.tinkercad.com/things/ek9dwFldgEI-light-sensor-using-arduino.

[6] Pallets. Flask User's Guide, 2010. https://flask.palletsprojects.com/.

[7] Công ty cổ phần công nghệ giáo dục F8. HTML CSS từ Zero đến Hero, 2024. https://fullstack.edu.vn/courses/html-css.

[8] W3Schools. CSS Tutorial, 2024. https://www.w3schools.com/css/.

[9] W3Schools. HTML Tutorial, 2024. https://www.w3schools.com/html/.

[10] Nguyễn Đức Hoàng Hạ. An Introduction to Programming the Internet of Things (IoT) Lecturer Slides, 2024. https://drive.google.com/drive/folders/1Vo7YCZSkvnj8ns3Y5iACijcrUE82hDja.