ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH



**BÁO CÁO**

**ĐỒ ÁN MÔN HỌC THIẾT KẾ LUẬN LÝ (CO3091)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**HỆ THỐNG HỖ TRỢ CHĂM SÓC CHẬU CÂY THÔNG MINH**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Ngành: Kỹ thuật máy tính**

**HỘI ĐỒNG:** ĐỒ ÁN THIẾT KẾ LUẬN LÝ 4 **GIÁO VIÊN HƯỚNG DẪN:** Huỳnh Hoàng Kha

**THƯ KÝ HỘI ĐỒNG:** Huỳnh Phúc Nghị

**DANH SÁCH THÀNH VIÊN:** Nguyễn Trọng Vinh - 2015070

Nguyễn Hồng Phát - 2014082

Nguyễn Duy Hòa - 2010276

**Tp.Hồ Chí Minh, Tháng 12/2022**

# MỤC LỤC

[**1. ĐÁNH GIÁ MỨC ĐỘ ĐÓNG GÓP**](#_5wh5ia5c7dyp)2

[**2. TỔNG QUAN ĐỒ ÁN**](#_e2e3twk75nr1)3

[2.1. Vấn đề đặt ra - mục tiêu của sản phẩm đồ án](#_ad7ypahzzcoy) 3

[2.2. Lý do chọn giải pháp thiết kế](#_e0w99ytyp961) 4

[2.3. Tính năng - phi tính năng sản phẩm](#_ox0qiy7tk1bg) 5

[2.4. Quá trình xây dựng giải pháp để giải quyết thực tiễn](#_w92lvf7ywwrm) 6

[2.5. Những tiêu chí chính đánh giá độ hoàn thiện sản phẩm](#_ys22hlst95ab) 7

[2.6. Giới hạn về khối lượng công việc và chất lượng sản phẩm](#_ydxahnwtf4cj) 7

[2.7. Kịch bản và phương pháp kiểm thử sản phẩm](#_62jbjk5hh84w) 8

[**3. THIẾT KẾ VÀ HIỆN THỰC**](#_y7aiyzjeb4ks)10

[3.1. Quá trình thiết kế](#_ut5mcpyz50mg) 10

[3.1.1. Phần cứng - Linh kiện sử dụng:](#_ap6zq2e72itz) 10

[3.1.2. Phần mềm sử dụng:](#_n46da699s20x) 17

[3.2. Quá trình hiện thực](#_zc8o5yr665zg) 17

[**4. KIỂM THỬ**](#_j89m1m85walv)25

[4.1. Chức năng chính](#_q3kzx0o4iolg) 25

[4.2. Kịch bản và cách thức kiểm thử](#_g5gcnx5iq013) 25

[4.3. Kết quả](#_t4exz1ynkft0) 27

[**5. SẢN PHẨM**](#_lsqwpx10wbwm)28

[5.1. Sơ đồ khối của mạch](#_4lte14l4kqnu) 28

[5.2. Source code sản phẩm](#_rdxlr8u2w80e) 28

[5.3. Schematic](#_xyzwiouuhn5a) 37

[5.4. Giao diện trên ứng dụng Blynk](#_lq0gg149veh6) 38

[**6. TỔNG KẾT**](#_4l1asft014vi)39

[6.1. Tổng kết và đánh giá sản phẩm](#_cb24jfea0t25) 39

[6.2. Hướng phát triển sản phẩm](#_d5oz8k6ijrfj) 39

[6.3. Bài học kinh nghiệm](#_vyexw6xtbdp1) 39

[**7. TÀI LIỆU THAM KHẢO**](#_yjw4rucdsi8h)40

# ĐÁNH GIÁ MỨC ĐỘ ĐÓNG GÓP

**BẢNG PHÂN CÔNG CÔNG VIỆC ĐỒ ÁN THIẾT KẾ LUẬN LÝ**

| **No.** | **Họ Tên** | **MSSV** | **Nhiệm vụ** | **Mức độ đóng góp** | **Hoàn thành** |
| --- | --- | --- | --- | --- | --- |
| 1 | **Nguyễn Trọng Vinh** | 2015070 | Tìm ý tưởng đồ án, thực hiện Proposal, lắp ráp linh kiện phần cứng, hoàn thiện Report báo cáo | **90%** | **100%** |
| 2 | **Nguyễn Hồng Phát** | 2014802 | Thảo luận tìm ý tưởng đồ án, hiện thực code phần mềm, lắp ráp hoàn thiện phần cứng sản phẩm | **100%** | **100%** |
| 3 | **Nguyễn Duy Hòa** | 2010276 | Thảo luận tìm ý tưởng đồ án, viết slide Proposal, Report; Lắp ráp phần cứng sản phẩm; hoàn thiện Report báo cáo | **90%** | **100%** |

# 

# TỔNG QUAN ĐỒ ÁN

## **Vấn đề đặt ra - mục tiêu của sản phẩm đồ án**

Ngày nay, khi mà công nghệ ngày càng phát triển thì việc ứng dụng các công nghệ điều khiển và quản lý vào trong sản phẩm là rất nhiều và cần thiết trong các ngành nghề và không ngoại trừ lĩnh vực nông nghiệp. Trong nông nghiệp, nhờ ứng dụng các công nghệ chăm sóc và điều khiển hiện đại, thông minh mà chất lượng sống cây trồng đã được tăng lên một cách đáng kể mà thời gian bỏ công chăm sóc từ con người đã giảm đi một cách rõ rệt.

Thực tế trong cuộc sống đô thị bận rộn ngày nay, nhiều vẫn có thú vui là trồng những cây cảnh, hay vườn rau vào những không gian trống được tận dụng trong ngôi nhà như: sân thượng, ban công,... Tuy nhiên, với nhiều người thì việc bỏ hàng giờ mỗi ngày hay đều đặn các buổi trong ngày để chăm sóc để chăm sóc “khu vườn” này không phải là điều dễ dàng, thì khi đó những chậu cây cảnh hay vườn rau đó sẽ không được tưới tiêu, chăm sóc đầy đủ; và khi phải sinh sống trong một mô đất nhỏ, dinh dưỡng và nguồn nước hạn hẹp như chậu cảnh mà không được theo dõi chăm sóc đúng thời điểm thì việc cây trồng bị kém phát triển và dẫn đến bị chết là điều rất dễ xảy ra.

Đứng trước yêu cầu thực tiễn này, nhóm em đã tiến hành thiết kế và hiện thực đề tài ***“Hệ thống chăm sóc chậu cây thông minh”*** trong môn học đồ án lần này. Sản phẩm đồ án ra đời nhằm giúp theo dõi và quản lí các chỉ số về độ ẩm đất, không khí và nhiệt độ môi trường xung quanh cây trồng được theo dõi, sau đó gửi các dữ liệu đo được này lên phần mềm được theo dõi bởi người dùng để từ đó giúp người dùng đưa ra quyết định có nên hay không việc tưới nước cho hệ thống cây trồng từ xa; ngoài ra, sản phẩm còn có thể tự động tưới nước cho cây trồng nếu độ ẩm môi trường đất của cây trồng bị thấp hơn ngưỡng cho phép của người dùng mà không cần sự thiệp của con người cho đến khi độ ẩm đất đạt đến ngưỡng mong muốn của chúng ta.

## Lý do chọn giải pháp thiết kế

Trước yêu cầu về việc cần được chăm sóc, tưới tiêu và theo dõi quá trình sinh trưởng phát triển của hệ thống cây trồng trong chậu. Ngoài việc **chăm sóc thủ công** yêu cầu chúng ta cần bỏ thời gian để theo dõi chăm sóc cây trồng vào mỗi ngày, một phương án tốn nhiều thời gian công sức của ta mà hiệu quả về sinh trưởng mang lại cho cây trồng chưa chắc đã cao.

Thì trên thị trường ngày nay, đã phát triển và cho ra đời khá nhiều hệ thống tưới tự động, tiêu biểu bao gồm một vài **hệ thống tưới tự động** nổi bật và ưa chuộng như:

* Hệ thống tưới tự động nhỏ giọt (Drip Systems)
* Hệ thống tưới cây phun mưa (Microsplay systems)
* Hệ thống tưới tự động pop - up (Pop-up Sprinklers)



Cả 3 hệ thống trên đều cung cấp khả năng tưới tự động cho cây trồng bằng nhiều hình thức khác nhau thông qua việc thiết lập hẹn giờ tưới từ đó giúp chúng ta giảm thiểu bớt thời gian cần bỏ ra cho việc chăm sóc cây trồng.

Tuy nhiên, nhược điểm của những hệ thống trên là: *việc lắp đặt, vận hành chúng đòi hỏi người dùng cần có những kiến thức nhất định về từng hệ thống* thì khi đó việc vận hành và sử dụng mới mang lại hiệu quả tốt nhất; ngoài ra những hệ thống trên thực tế *vẫn chưa được tích hợp tính năng tự động tưới theo yêu cầu về độ ẩm đất mà người dùng mong muốn, mà chỉ đảm bảo tự động tưới vào khoảng thời gian mong muốn người dùng*. Không những vậy thì các hệ thống này hầu như *chỉ phù hợp cho những mô hình chăm sóc vườn cây diện tích lớn, với số lượng mật độ cây trồng cao* do chi phí lắp đặt và vận hành là khá cao, không phù hợp với nhu cầu sử dụng ở những mô hình chăm sóc cây trồng trong chậu hộ gia đình.

Chính vì vậy mà nhóm em đã tiến hành thiết kế mô hình chăm sóc thông minh cho cây trồng trong chậu ở quy mô gia đình ít cây trồng sao cho chi phí thực hiện sản phẩm là thấp nhất mà vẫn đáp ứng được những yêu cầu về kỹ thuật được nêu ra ở phần **“mục tiêu sản phẩm đồ án”** nhằm khắc phục được những nhược điểm mà các sản phẩm hay giải pháp đã nêu ở trên vẫn còn tồn đọng.

## Tính năng - phi tính năng sản phẩm

Sản phẩm được thiết kế trong đồ án bao gồm những **đặc điểm về** **tính năng** như:

* **Thu thập các thông số** về nhiệt độ môi trường, độ ẩm không khí và đất thông qua cảm biến và **hiển thị dữ liệu** trên lên ứng dụng được cài đặt cho người dùng
* Được **tối đa hóa các chế độ tưới cho cây trồng** bao gồm: **tưới tự động** dựa vào mức độ ẩm đất ở thời gian hiện tại có đang thấp hơn mức chấp nhận của ta hay không, các yếu tố về thời điểm tưới trong ngày, và dự báo thời tiết; và **chế độ điều khiển tưới thủ công** từ xa bởi người dùng thông qua các thông số về nhiệt độ, độ gửi về từ hệ thống.
* **Gửi cảnh báo** cho người dùng khi các thông số về **nhiệt độ ngoài trời hoặc độ ẩm đất** vượt quá ngưỡng cho phép của người dùng, để giúp chúng ta có các biện pháp xử chăm sóc kịp thời cho cây trồng.

Ngoài ra, bên cạnh những tính năng kể trên, thì sản phẩm cũng bao gồm những **đặc điểm về phi tính năng** cụ thể như:

* Dữ liệu sẽ được cập nhật mỗi 1 phút một lần cho người dùng giúp chúng ta kịp thời xử lý khi có sự cố về độ ẩm hoặc nhiệt độ nằm ngoài cho phép trong thời gian thực; giao diện ứng dụng hiển thị đơn giản, dễ sử dụng.
* Có thể điều khiển giám sát ở bất kỳ nơi nào đảm bảo kết nối Internet.
* Phạm vi giá trị về độ ẩm đất : **0 - 100%**RH, sai số **±3%**RH.
* Phạm vi giá trị về độ ẩm không khí: **20% - 90%** RH, sai số **±5%**RH
* Phạm vi nhiệt độ cho phép là: **0°C ~ 50°C**, sai số **±2°C**

## Quá trình xây dựng giải pháp để giải quyết thực tiễn

Để cho ra được giải pháp cuối cùng là sản phẩm mà nhóm em mong muốn hiện thực, thì nhóm đã làm việc thảo luận dựa trên những cơ sở:

* **Thứ nhất**, nhóm nghiên cứu thực tiễn của vấn đề đang tồn đọng những khó khăn gì cần nhóm em phải giải quyết.
* Tham khảo thực tế thị trường đã có những sản phẩm, giải pháp nào đi vào giải quyết vấn đề này chưa, và đã giải quyết được bao nhiêu % vấn đề (nói cách khác là triệt để chưa), từ đó đặt ra những yêu cầu cho sản phẩm mà nhóm cần hiện thực phải cố gắng hoàn thiện được những mặt còn chưa làm được trên.
* Thảo luận, xây dựng bản thiết kế từ tổng quát đến chi tiết (sơ đồ khối, mạch điện sản phẩm, linh kiện cần thiết,...) cho sản phẩm mà nhóm đặt ra để từ đó làm tiền đề cho việc hoàn thiện sản phẩm.
* **Cuối cùng,** đưa ra những tiêu chí để đánh giá mức độ hoàn thiện sản phẩm (dựa trên những yêu cầu đặt ra ban đầu ).

## Những tiêu chí chính đánh giá độ hoàn thiện sản phẩm

Sản phẩm thiết kế mà nhóm thực hiện là ***“hệ thống chăm sóc thông minh cho cây trồng trong chậu”*** , thì để đánh giá được sản phẩm sau cùng mà nhóm làm ra có đạt được yêu cầu hay chưa, nhóm dựa trên những tiêu chí sau để từ đó, nếu hệ thống chưa đáp ứng được sẽ thực hiện việc thay đổi và hoàn thiện:

* **Đảm bảo cung cấp đủ các dữ liệu về thông số độ ẩm đất lẫn không khí và nhiệt độ môi trường** từ cảm biến đồng thời phải hiển thị chính xác giá trị đo được lên ứng dụng Blynk phía người dùng.
* **Thực hiện ngay chế độ tự động tưới** khi có sự hạ thấp vượt mức giá trị độ ẩm đất mà người dùng đã setup trước vào hệ thống.
* **Cho phép người dùng điều khiển từ xa** hệ thống tưới cho cây đến khi đạt mức độ ẩm đất mong muốn của người dùng thì dừng.
* Và cuối cùng, **hệ thống phải thực hiện gửi cảnh báo** ngay khi các giá trị về độ ẩm đất và nhiệt độ môi trường nằm ngoài khoảng mong muốn để người dùng.

## Giới hạn về khối lượng công việc và chất lượng sản phẩm

Để đảm bảo hoàn thành được sản phẩm đồ án đúng tiến độ cũng như chất lượng đã đặt ra ở mục tiêu đồ án, thì mỗi thành viên của nhóm em dự kiến cần đảm bảo được **tối thiểu 8h** làm việc, nghiên cứu cho sản phẩm đồ án trên mỗi tuần; và cả nhóm cần ít nhất **10 tuần** để thiết kế và hoàn thiện sản phẩm. Ngoài ra thì khối lượng công việc cần giải quyết của nhóm cho từng giai đoạn đồ án **tối thiểu phải đạt được theo timeline mà giảng viên đã nêu ra** từ đầu môn học, nếu như nhóm không thể vượt tiến độ công việc.

Bên cạnh đó, thì **về mặt chất lượng**, sản phẩm hoàn thiện sau cùng của nhóm em, dự kiến phải đạt tối thiểu là **đảm bảo pass toàn bộ các tính năng cơ bản mà nhóm đã đặt ra ở phần mục tiêu đồ án** bao gồm:

* Đảm bảo thu thập đầy đủ dữ liệu và hiển thị rõ ràng ra ứng dụng theo dõi người dùng
* Thực hiện chế độ tự động tưới kịp thời khi gặp vấn đề về độ ẩm như nhóm đã đề ra, và cho phép điều khiển tưới từ xa.
* Gửi cảnh báo ngay khi các giá trị về độ ẩm và nhiệt độ không như mong muốn mà người dùng đã thiết lập.

## Kịch bản và phương pháp kiểm thử sản phẩm

Để đánh giá sản phẩm làm ra có đảm bảo các yêu cầu về mặt kỹ thuật và tính năng như ban đầu mà nhóm đã đặt ra hay không, thì nhóm em đã xây dựng những kịch bản và phương pháp kiểm thử sản phẩm tương ứng cho mỗi kịch bản bao gồm:

* **Kiểm thử cảm biến và hiển thị ra ứng dụng**
  + Kiểm tra giá trị của cảm biến nhiệt độ và độ ẩm không khí ở 2 buổi trong ngày là: vào sáng sớm và buổi trưa. Riêng cảm biến về độ ẩm đất, ta so sánh sự thay đổi giá trị trong 2 chậu đất khô và ướt.
  + In giá trị thông qua serial port của phần mềm Arduino IDE, và quan sát sự khác biệt
  + Đồng thời dùng dữ liệu thu được gửi lên server, mở ứng dụng kiểm tra các giá trị so sánh với giá trị được in trên serial port.
* **Kiểm thử chế độ tự động tưới**
  + Đưa cảm biến vào đất khô có độ ẩm thấp
  + Đặt trong môi trường có nhiệt độ thấp (vào buổi sáng sớm hoặc tối), giả định sẽ không có mưa trong ngày tới. Sau đó thay đổi giả định sẽ có mưa
  + Đặt cảm biến trong môi trường nhiệt độ cao như buổi trưa
  + Tương tự, đưa cảm biến vào đất ẩm và kiểm tra phản hồi của hệ thống.
* **Kiểm thử tính năng cảnh báo nhiệt độ và độ ẩm**
  + Làm cho nhiệt độ tăng cao vượt mức tối đa cho phép bằng việc hơ nóng cảm biến.
  + Giảm nhiệt độ xuống thấp hơn mức tối thiểu cho phép bằng nước đá và xem phản hồi của hệ thống.
  + Đưa cảm biến độ ẩm đất vào đất ngập nước trong 30phut và xem thông báo của hệ thống ra ứng dụng.

# **THIẾT KẾ VÀ HIỆN THỰC**

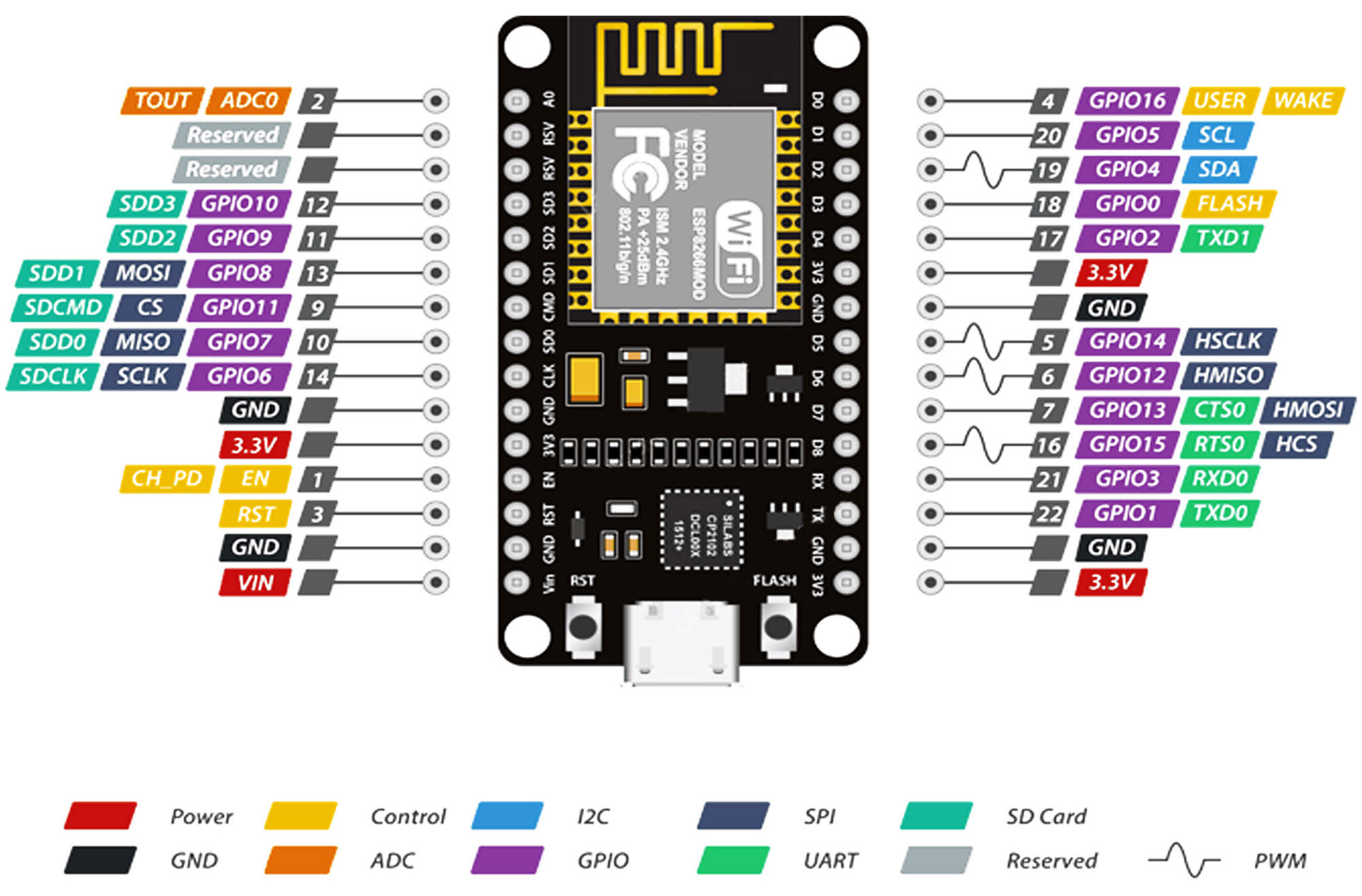
## **Quá trình thiết kế**

* **Chức năng hiển thị dữ liệu**: Thu thập các dữ liệu về nhiệt độ, độ ẩm không khí, độ ẩm đất thông qua các cảm biến, sau đó thực hiện gửi dữ liệu lên server và hiển thị dữ liệu lên giao diện của ứng dụng, cho người dùng có thể quản lý các chỉ số.
* **Chức năng tự động tưới:** Đọc các giá trị như độ ẩm đất mong muốn, tối đa, nhiệt độ tối thiểu, tối đa từ ứng dụng do người dùng nhập. Kết hợp với dữ liệu về thời điểm hiện tại trong ngày (ngoài khung giờ từ 10h-15h, khi nhiệt độ trong ngày quá cao), và dự báo về thời tiết. Sau khi thu được dữ liệu từ thực tế và do người dùng cài đặt, thực hiện so sánh dữ liệu thu được, kết hợp với các yếu tố khác như thời điểm thích hợp trong ngày, dự báo thời tiết, qua đó ra quyết định hệ thống có tự động tưới nước cho cây.
* **Chức năng gửi cảnh báo:** So sánh dữ liệu thu được từ thực tế với khoảng giá trị phù hợp cho cây do người dùng cài đặt. Nếu giá trị thực tế của nhiệt độ môi trường, độ ẩm đất cao vượt ngưỡng cho phép, ứng dụng gửi cảnh báo cho người dùng qua ứng dụng và email.

## Phần cứng - Linh kiện sử dụng:

### 

* **ESP8266 CP2102 NodeMCU:**
  + **Sơ đồ:**

****

* **Chức năng:**

**ESP8266** có thể được dùng làm module Wifi bên ngoài, sử dụng firmware tập lệnh AT tiêu chuẩn bằng cách kết nối nó với bất kỳ bộ vi điều khiển nào sử dụng UART nối tiếp hoặc trực tiếp làm bộ vi điều khiển hỗ trợ Wifi, bằng cách lập trình một chương trình cơ sở mới sử dụng SDK được cung cấp. Hay được sử dụng trong các ứng dụng liên quan đến IoT.

* **Lý do chọn linh kiện:**
  + Dễ sử dụng, có thể được lập trình thông qua Arduino IDE
  + Kết nối được Wifi
  + Giá thành rẻ, phù hợp với quy mô của đồ án môn học.
* **Module 1 Relay kích mức thấp 5V DC**
  + **Chức năng:**

Được sử dụng để đóng ngắt nguồn điện công suất cao AC hoặc DC, có thể chọn đóng khi kích mức cao hoặc mức thấp bằng cách điều chỉnh Jumper, tiếp điểm đóng ngắt gồm 3 tiếp điểm: **NC** (normally close) tức thường đóng, **NO** (normally open) tức thường mở và **COM** (chân chung).

* + **Thông số kỹ thuật:**

**Điện áp hoạt động:** DC5V

**Ngõ ra tiếp điểm:** NC-NO-COM

**Relay kích mức thấp**

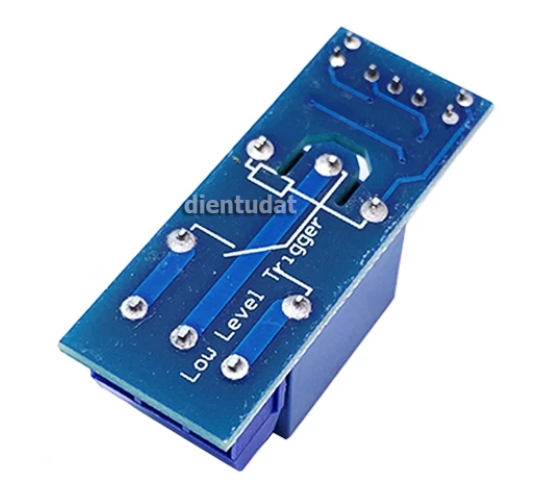
**Công suất tiếp điểm Relay:** 250VAC/10A, 30VDC/10A

**Led báo trạng thái:** có

**Kích thước:** 43 x 17 x 17mm

* + **Hình ảnh:**

****

****

* **Cảm biến nhiệt độ - độ ẩm không khí DHT11:**
  + **Giới thiệu:**

Cảm biến độ ẩm và nhiệt độ **DHT11 Temperature Humidity Sensor** là cảm biến rất thông dụng hiện nay vì chi phí rẻ và rất dễ lấy dữ liệu thông qua giao tiếp **1 wire** (giao tiếp digital 1 dây truyền dữ liệu duy nhất). Bộ tiền xử lý tín hiệu tích hợp trong cảm biến giúp bạn có được dữ liệu chính xác mà không phải qua bất kỳ tính toán nào.

* + **Thông số kỹ thuật:**

**Nguồn sử dụng:** 3 - 5 VDC.

**Dòng sử dụng:** 2.5mA max (khi truyền dữ liệu).

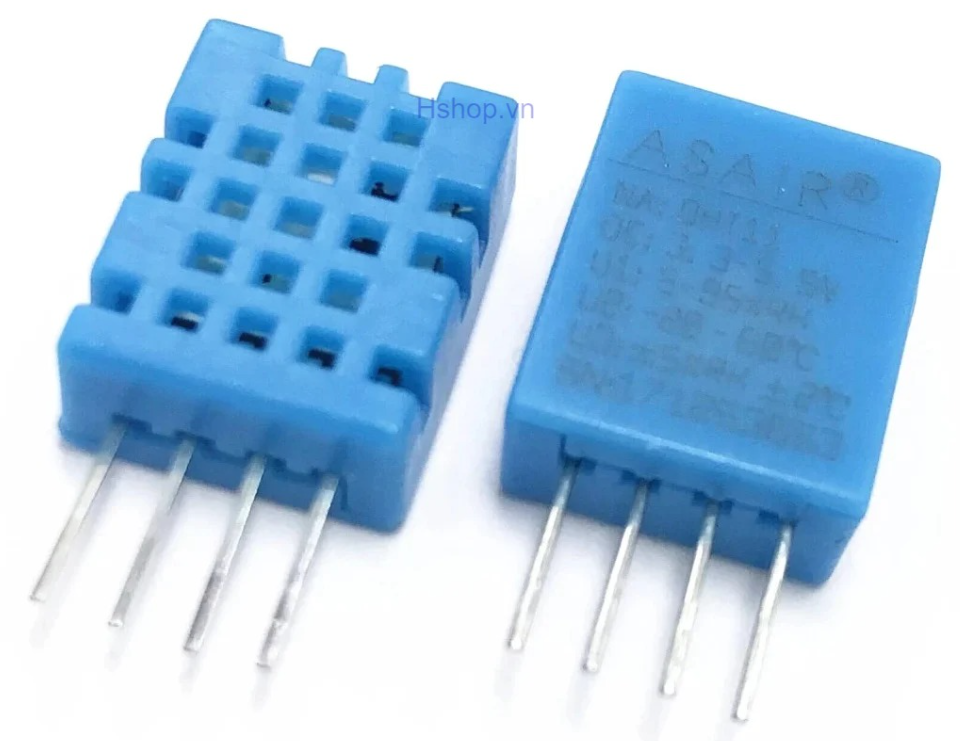
**Khoảng đo độ ẩm:** 20%-90% RH (sai số 5%RH)

**Khoảng đo nhiệt độ:** 0-50°C (sai số 2°C)

**Tần số lấy mẫu tối đa:** 1Hz (1 giây / lần)

**Kích thước:** 15mm x 12mm x 5.5mm.

* + **Hình ảnh**

****

****

* **Cảm biến độ ẩm đất - Soil Moisture Sensor**
  + **Giới thiệu:**

Cảm biến độ ẩm đất Soil Moisture Sensor thường được sử dụng trong các mô hình tưới nước tự động, vườn thông minh,..., cảm biến giúp xác định độ ẩm của đất qua đầu dò và trả về giá trị Analog, trạng thái đầu ra mức thấp **(0V)**, khi đất thiếu nước đầu ra sẽ là mức cao **(5V)**, độ nhạy cao chúng ta có thể điều chỉnh được bằng biến trở.

* + **Thông số kỹ thuật:**

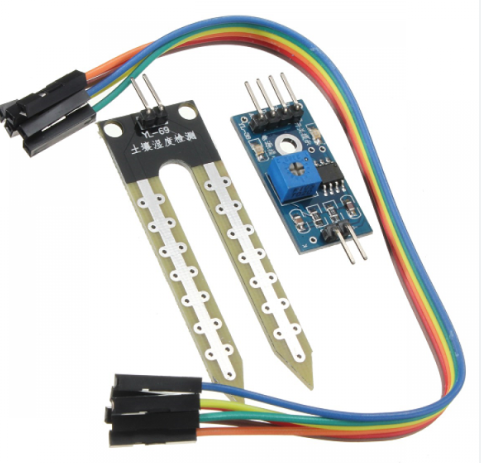
**Điện áp hoạt động:** 3.3 ~ 5VDC

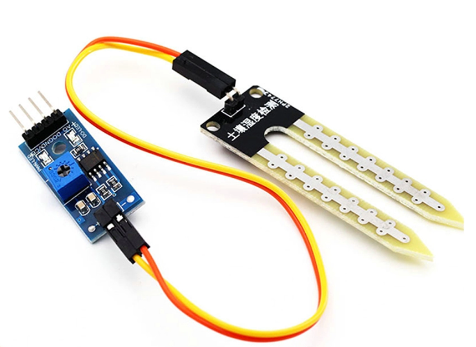
**Tín hiệu đầu ra:**

* **Analog:** theo điện áp cấp nguồn tương ứng.
* **Digital:** High hoặc Low, có thể điều chỉnh độ ẩm mong muốn bằng biến trở thông qua mạch so sánh LM393 tích hợp.

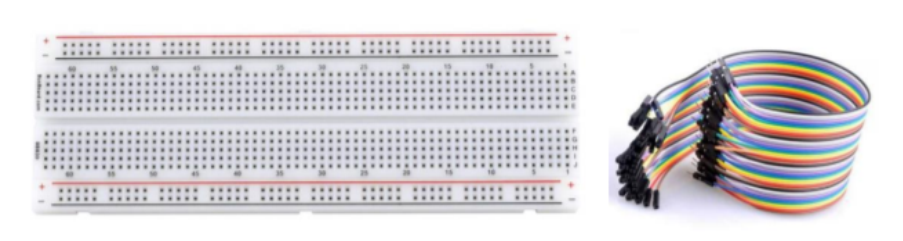
**Kích thước**: 3 x 1.6cm.

* + **Hình ảnh:**

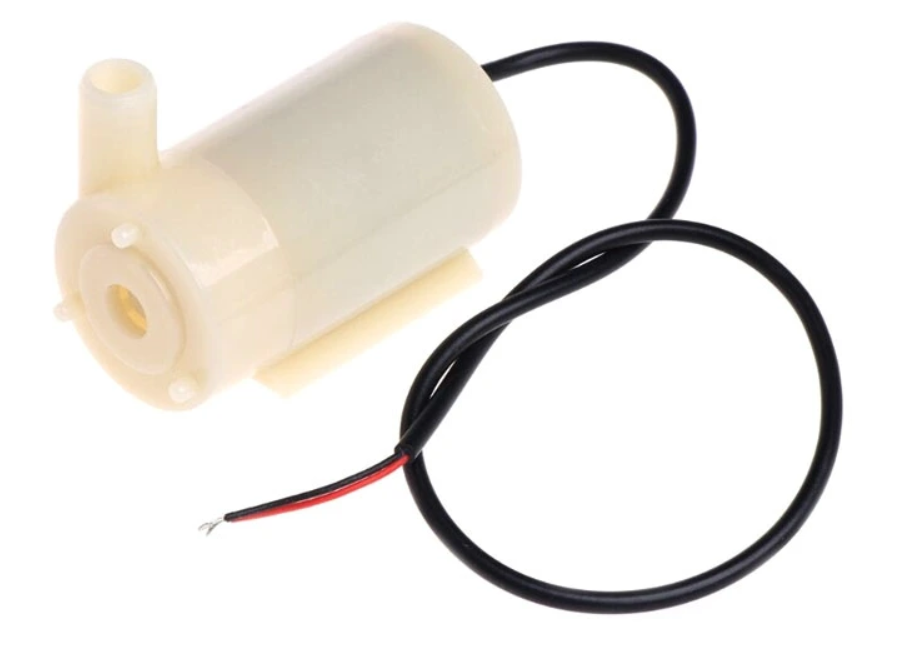
****

****

* **Breadboard và dây cắm**
  + Dùng để kết nối các linh kiện được sử dụng lại với nhau.
  + Hình ảnh:



* **Máy bơm nước Mini - DC 5V**
  + Dùng để thực hiện chế độ tự động tưới và tưới thủ công của người dùng qua hệ thống điều khiển từ xa.



* + **Thông số kỹ thuật:**
    - Điện áp: 3-5VDC.
    - Dòng điện: 100-200mA
    - Mức đẩy nước tối đa: 0,3-0,8 mét.
    - Khả năng bơm: 1.2-1.6L / phút.
    - Trọng lượng: 28g.

## Phần mềm sử dụng:

## 

### Lựa chọn phần mềm Blynk để hiển thị kết quả đo ra giao diện cho người dùng, vì ứng dụng miễn phí, dễ sử dụng, giao diện đơn giản. Sau đó thiết kế giao diện cho người dùng có thể quản lý, cài đặt các giá trị cần thiết cho môi trường. Cuối cùng thực hiện các kết nối cần thiết để ứng dụng có thể gửi và nhận dữ liệu từ phần cứng.

* Sử dụng phần mềm Arduino IDE để hiện thực source code phía phần mềm cho các tác vụ thu thập dữ liệu cảm biến, đưa ra cảnh báo và hiện thực cơ chế tưới tự động và thủ công thông qua linh kiện trung tâm là: ESP8266 NodeMCU

## Quá trình hiện thực

* **Thiết lập phần cứng, phần mềm:**
  + Thiết kế sơ đồ kết nối cho các phần cứng, sau đó thực hiện theo thiết kế bao gồm: **cảm biến nhiệt độ**, **độ ẩm không khí** **(DHT11)**, **và độ ẩm đất**, **NodeMCU**.
  + Trên ứng dụng Blynk, đăng ký tài khoản nhận token, thực hiện tạo dự án. Trong dự án, thiết kế các giao diện hiển thị **nhiệt độ**, **độ ẩm không khí**, **độ ẩm đất**, thanh điều chỉnh cho người dùng cài đặt các giá trị mong muốn bao gồm độ ẩm đất mong muốn cho cây, độ ẩm tối đa, khoảng nhiệt độ an toàn cho cây ( tối thiểu, tối đa). Và cài đặt các chân ảo tương ứng cho từng giá trị để giao tiếp trong lúc lập trình. Chân **V0** là nút nhấn điều khiển tưới qua ứng dụng, các chân **V4, V5, V6, V7** tương ứng với độ ẩm đất tối đa, nhiệt độ tới thiểu, tối đa, độ ẩm đất mong muốn.
  + Thực hiện các khai báo, cài đặt thư viện cần thiết trên Arduino IDE để kết nối wifi cho module NodeMcu. Sử dụng lệnh **Blynk.begin** truyền vào token, id và password wifi để kết nối đến ứng dụng. Sử dụng **BlynkTimer** để gọi các hàm trong những khoảng thời gian khác nhau, tránh dùng delay.

| #define BLYNK\_TEMPLATE\_ID "TMPLhK0cTMZy" #define BLYNK\_DEVICE\_NAME "esp" #define BLYNK\_AUTH\_TOKEN "1keelibMrIg7szjPDh19NuHL0bLXLNTt" #define BLYNK\_PRINT Serial  #include <ESP8266WiFi.h> #include <BlynkSimpleEsp8266.h> char auth[] = BLYNK\_AUTH\_TOKEN; // Your WiFi credentials. char ssid[] = ""; char pass[] = "";  BlynkTimer timer; void loop() {  Blynk.run();  timer.run(); }  void setup(){  // connect to blynk Blynk.begin(auth, ssid, pass); } |
| --- |

* **Thu thập dữ liệu:** 
  + Đặt các cảm biến vào môi trường không khí, đất của chậu cây. Trên Arduino IDE, cài đặt các thư viện cần thiết để đọc dữ liệu từ cảm biến, khai báo các chân. Trong hàm **SensorData()** khởi tạo **DHTesp** là **dht**, sau đó gọi các phương thức từ DHTesp để đọc các giá trị từ cảm biến DHT11. Đọc giá trị tín hiệu analog từ cảm biến độ ẩm đất sau đó chuyển đổi giá trị về khoảng **0-100** và thực hiện in ra serial monitor. Hàm **SensorData()** sẽ được gọi mỗi giây 1 lần thông qua phương thức **setInterval** của **BlynkTimer**.

| #include <DHTesp.h>  DHTesp dht;  void SensorData()  {  humidity = dht.getHumidity();  temperature = dht.getTemperature();  Serial.print("humidity ");  Serial.println(humidity);  Serial.print("temperature ");  Serial.println(temperature);  sensorData = analogRead(sensorpin);  moisture = map(sensorData,0,1023,100,0);  Serial.print("moisture ");  Serial.println(moisture);  // send data to blynk  Blynk.virtualWrite(V1, temperature);  Blynk.virtualWrite(V2, humidity);  Blynk.virtualWrite(V3, moisture);  }  void setup(){  timer.setInterval(1000,SensorData);  } |
| --- |

* + Trong hàm **SensorData()**, sau khi dữ liệu được đọc vào, sử dụng lệnh **Blynk.virtualWrite** để gửi những dữ liệu này lên server qua các chân ảo của ứng dụng blynk, hiển thị cho người dùng theo giao diện đã được thiết lập.
* **Chức năng tự động tưới**
  + Trên IDE, trong hàm **BLYNK\_WRITE(V7)** đọc giá trị độ ẩm đất mong muốn do người dùng nhập từ chân V7, sử dụng phương thức **asDouble()** để đọc giá trị từ ứng dụng. Sau đó, lấy dữ liệu đọc được từ cảm biến độ ẩm đất và giá trị của người dùng mong muốn. Và thực hiện so sánh 2 giá trị.
  + Lấy dữ liệu về thời tiết, trên Openweathermap, đăng ký tài khoản và lấy API Key. Khởi tạo 1 chuỗi chứa url trỏ tới server của Openmap, hàm **getWeatherData()** sẽ được gọi mỗi **10p**, mỗi khi gọi chúng ta sẽ sử dụng phương thức get với URL để gửi request, sau đó đọc kết quả trả về. Kết quả thu được là kết quả dự báo thời tiết ở mỗi thời điểm cách nhau **3h** kể từ lần gửi request. Sau đó kiểm tra trong vòng 8 khoảng thời gian **(24h)**, nếu kết quả có mưa sẽ gán biến **notRain = false**.

| #include <ArduinoJson.h>  #define JSON\_BUFF\_DIMENSION 2500  WiFiClient client;  const char server[] = "api.openweathermap.org";  String apiKey = "6323acdc1da35c67469fbb7e3de6cd76";  String forecastTitle[10];  String text;  int jsonend = 0;  void getWeatherData() {  Serial.println("\nStarting connection to server...");   if (client.connect(server, 80)) {   Serial.println("connected to server");   // Make a HTTP request:   client.print("GET /data/2.5/forecast?lat=10.750&lon=106.667");   client.print("&appid=6323acdc1da35c67469fbb7e3de6cd76");   client.print("&cnt=8");   client.println("&units=metric");   client.println("Host: api.openweathermap.org");   client.println("Connection: close");   client.println();   } else {   Serial.println("unable to connect");   }   String line = "";   while (client.available()) {  line = client.readStringUntil('\n');   StaticJsonBuffer<30000> jsonBuffer;   JsonObject& root = jsonBuffer.parseObject(line);   for(int i=0;i<8;i++){  forecastTitle[i] = root["list"][i]["weather"][0]["main"].as<String>();  }  notRain = true;  for(int i=0;i<8;i++){  Serial.println(forecastTitle[i]);  }  for(int i=0;i<8;i++){  if(forecastTitle[i]=="Rain"){  notRain = false;  break;  }  }  Serial.println(notRain);  } } |
| --- |

* + Cài đặt thư viện NTPClient (**NTPClient.h**), thư viện này kết nối **WiFi ESP8266** với máy chủ thời gian, máy chủ này gửi thông tin thời gian đến module NTP. Thông qua module NTP có thể lấy được các giá trị về thời gian hiện tại. Hàm getHour() lấy kết quả trả về và so sánh khoảng thời gian, biến ***hour sai*** nếu nằm trong khoảng **10h - 15h.**

| #include <NTPClient.h>  #include <WiFiUdp.h>  WiFiUDP ntpUDP;  NTPClient n(ntpUDP, "3.vn.pool.ntp.org",7\*60\*60 );  void getHour(){  n.update();  if(n.getHours()>=10 && n.getHours() <= 15){  hour = false;  }  else{  hour = true;  }  Serial.println(n.getHours()); }  void setup(){  n.begin();  WiFi.begin(ssid,pass);  } |
| --- |

* + Hàm **Water()** sẽ kiểm tra người dùng có bật chế độ tự động tưới hay không, nếu có sẽ kiểm tra nếu thỏa mãn các điều kiện như độ ẩm đất thấp hơn mức người dùng mong muốn, thời điểm thích hợp trong ngày (***ngoài khung giờ 10h-15h***), trong **24h** tiếp theo sẽ không có mưa, sẽ gửi tín hiệu bật công tắc máy bơm để tưới cây, cho đến khi độ ẩm đất thu vào xấp xỉ so với mức mong muốn sẽ dừng tưới.

| void Water(){  // user control  if(userWater == 1){  digitalWrite(waterPin,1);  water == true;  }  else{  digitalWrite(waterPin,0);  water == false;  }  bool hourDemo = true;  bool notRainDemo = true;    // automatic control  if(moisture < soilMoisture && water == false && hour ==true && userWater == 0 && notRain == true){  digitalWrite(waterPin,1);  water == true;  }  else if(moisture >= soilMoisture && userWater == 0){  digitalWrite(waterPin,0);  water = false;  } } |
| --- |

* **Chức năng gửi cảnh báo:**
  + Trên ứng dụng đã được thiết lập phần thanh trượt để người dùng nhập các giá trị độ ẩm mong muốn, khoảng nhiệt độ thích hợp cho cây, tương ứng mỗi giá trị được ghi vào một virtual pin. Thiết lập các sự kiện gửi cảnh báo cho người dùng như sự kiện độ ẩm đất vượt quá mức, nhiệt độ quá cao hoặc quá thấp.
  + Trên IDE, đọc lần lượt các giá trị từ virtual pin, sau đó so sánh với các giá trị từ thực tế đã được thu thập từ bước trước. Nếu các giá trị trên vượt qua ngưỡng người dùng mong muốn và các sự kiện đang không được kích hoạt, thì các sự kiện gửi thông báo tương ứng đã được thiết lập trên ứng dụng sẽ được kích hoạt.

| void sendNotify(){  // send notify if moisture > max moisture  if(moisture > maxSoilMoisture && eventTrigger == false){  Serial.println("max mois");  eventTrigger = true;  Blynk.logEvent("maxmoisture","Soild Moisture is higher max");  }  else if(moisture <= maxSoilMoisture){  eventTrigger = false;  }  // send notify if temperature < min temperature  if(temperature< minTemp && eventTriggerTemp==false){  eventTriggerTemp = true;  Serial.println("min temp");  Blynk.logEvent("mintemp","Temperatur is lower min");  }  // send notify if temperature > max temperature  else if(temperature > maxTemp && eventTriggerTemp == false){  eventTriggerTemp = true;  Serial.println("max temp");  Blynk.logEvent("maxtemp","Temperatur is higher max");  }  else if(temperature<= maxTemp && temperature>=minTemp){  eventTriggerTemp = false;  } } |
| --- |

# **KIỂM THỬ**

## Chức năng chính

* Thu thập dữ liệu và hiển thị trên ứng dụng, lấy dữ liệu từ người dùng
* Chức năng tự động tưới
* Chức năng gửi cảnh báo

## Kịch bản và cách thức kiểm thử

**Cảm biến và ứng dụng**

* Đưa cảm biến vào các điều kiện môi trường khác nhau ( buổi sáng: nhiệt độ, độ ẩm thấp; buổi trưa: nhiệt độ, độ ẩm cao; chậu đất khô và ướt), kiểm tra sự thay đổi của các giá trị thông qua monitor port của phần mềm Arduino IDE. So sánh nhiệt độ, độ ẩm không khí với giá trị thực tế, đối với độ ẩm đất quan sát sự thay đổi khi đưa vào các môi trường đất khác nhau.
* Thực hiện đưa dữ liệu thu được lên server của ứng dụng và hiển thị lên giao diện người dùng. Kiểm tra so sánh các giá trị tương ứng bao gồm nhiệt độ, độ ẩm không khí, độ ẩm đất trên monitor port.
* Kiểm tra chức năng nhập dữ liệu từ người dùng. Nhập các giá trị bao gồm độ ẩm đất mong muốn, tối đa, nhiệt độ tối thiểu, tối đa từ ứng dụng, đọc giá trị từ các chân ảo đã được cài đặt, sau đó in ra monitor port của Arduino IDE để kiểm tra sự chính xác.

**Chức năng tự động tưới**

* Đặt cảm biến vào đất khô, có độ ẩm thấp hơn mức mong muốn. Đặt trong môi trường có nhiệt độ thấp (vào buổi sáng sớm hoặc tối, ngoài khung giờ 10-15h), giả định sẽ không có mưa trong 24h tới. Kiểm tra kết quả. Sau đó thay đổi giả định trời sẽ có mưa. Tương tự kiểm tra kết quả khi đặt vào đất khô, nhiệt độ cao, có mưa và ngược lại. Đặt cảm biến vào đất ẩm, nhiệt độ thấp, giả định không mưa, kiểm tra kết quả và trường hợp ngược lại.

**Chức năng gửi cảnh báo**

* Tăng nhiệt độ môi trường lên cao, vượt qua mức tối đa do người dùng đã cài đặt qua ứng dụng, tương tự giảm nhiệt độ môi trường xuống thấp hơn mức tối thiểu. Sau đó kiểm tra thông báo từ ứng dụng và email.
* Sau đó đưa cảm biến độ ẩm đất vào đất ẩm ngập nước, có độ ẩm vượt mức tối đa do người dùng cài đặt, sau đó kiểm tra cảnh báo được gửi qua ứng dụng và email.

## Kết quả

**Cảm biến và ứng dụng**

* Khi đưa các cảm biến vào các điều kiện môi trường khác nhau, sau đó in các giá trị thu được ra monitor port của Arduino IDE, kiểm tra nhiệt độ, độ ẩm của môi trường tương đối chính xác so với thực tế, kiểm tra giá trị độ ẩm đất thu được thay đổi một cách hợp lý khi thay đổi môi trường đất.
* Các giá trị tương ứng thu được hiển thị chính xác trên giao diện của ứng dụng so với các giá trị được in ra trên monitor của IDE.
* Các giá trị thu được do người dùng nhập từ ứng dụng thông qua các pin ảo được in ra monitor port chính xác so với giá trị trên ứng dụng.

**Chức năng tự động tưới**

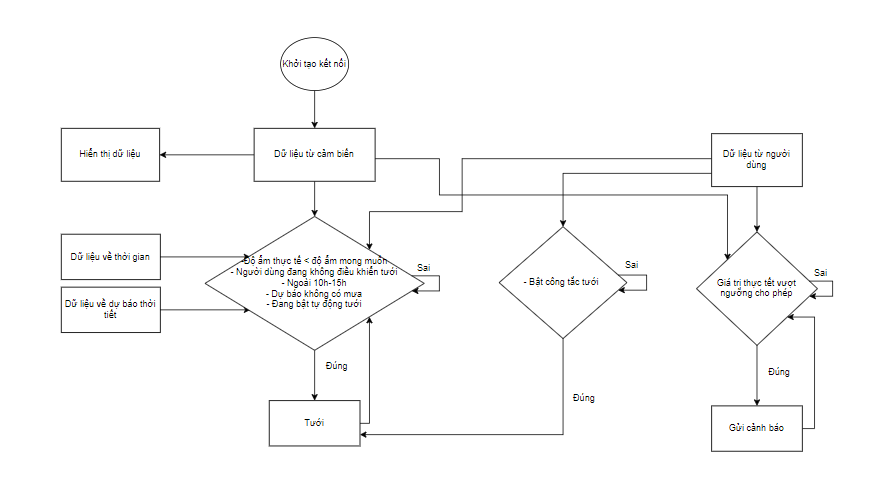
* Kết quả thu được, chỉ khi độ ẩm đất xuống thấp hơn mức mong muốn, thời điểm xét ngoài khoảng 10-15h khi nhiệt độ của môi trường không quá cao và trong ngày tiếp theo dự báo sẽ không có mưa, hệ thống mới bật công tắc tưới, cho đến khi độ ẩm đất xấp xỉ mức người dùng mong muốn sẽ dừng lại. Ngoài ra không có trường hợp nào hệ thống tự động tưới.

**Chức năng gửi cảnh báo**

* Khi cho các giá trị vượt ngưỡng mong muốn của người dùng, cảnh báo tương ứng sẽ được gửi cho người dùng trên ứng dụng và email.

# **SẢN PHẨM**

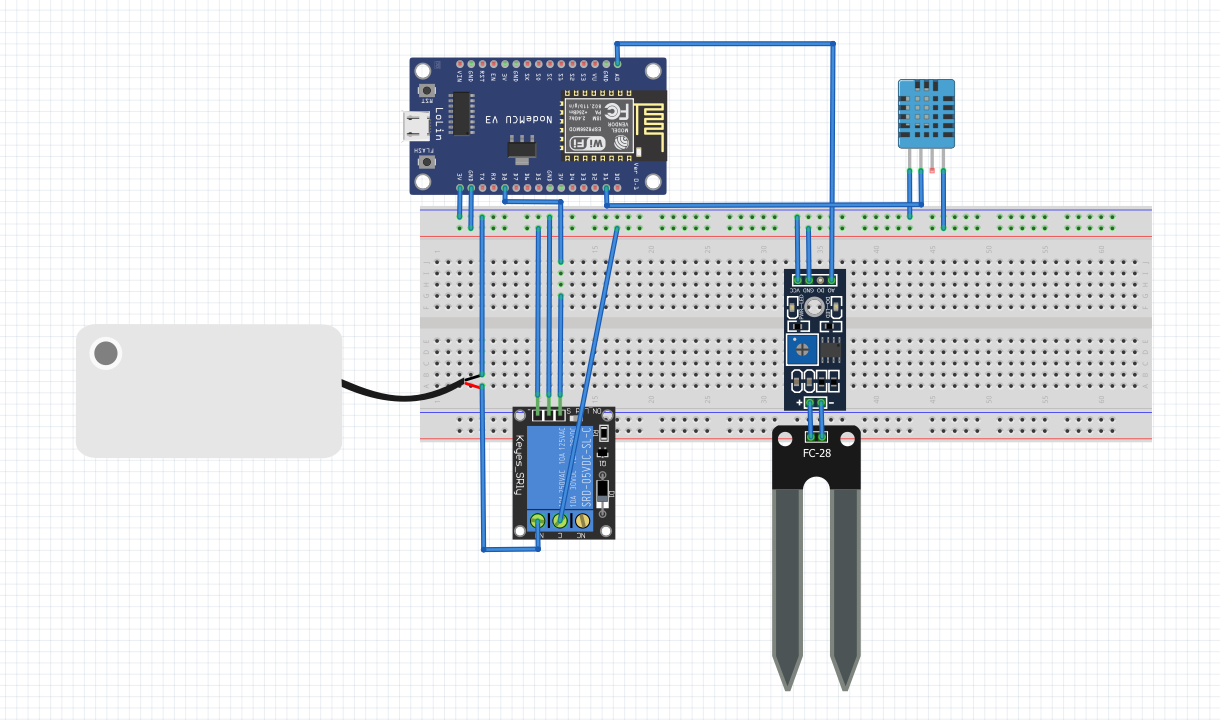
## Sơ đồ khối của **mạch**



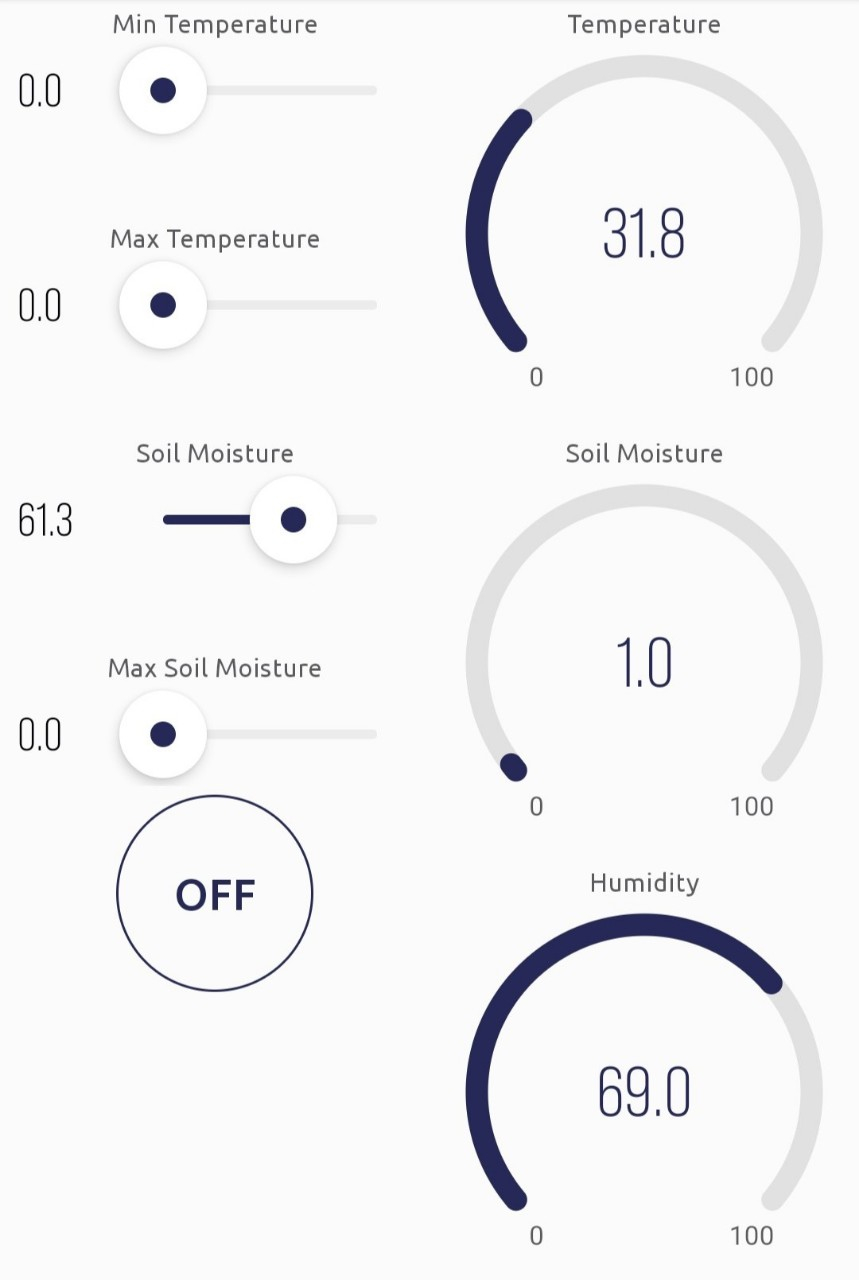
## Source code sản phẩm

| #define BLYNK\_TEMPLATE\_ID "TMPLhK0cTMZy" #define BLYNK\_DEVICE\_NAME "esp" #define BLYNK\_AUTH\_TOKEN "1keelibMrIg7szjPDh19NuHL0bLXLNTt" #define BLYNK\_PRINT Serial  #include <NTPClient.h>  #include <ArduinoJson.h> #include <WiFiClient.h> #include <Adafruit\_Sensor.h> #include <ESP8266WiFi.h> #include <BlynkSimpleEsp8266.h> #include <DHTesp.h> #include <Arduino.h> #include <WiFiUdp.h> char auth[] = BLYNK\_AUTH\_TOKEN;  // Your WiFi credentials. char ssid[] = "MANG DAY KTX H1-1004"; char pass[] = "123456789A"; BlynkTimer timer; DHTesp dht; WiFiUDP ntpUDP; NTPClient n(ntpUDP, "3.vn.pool.ntp.org",7\*60\*60 ); // constant int waterPin = D8;  int sensorpin= A0; bool water = false; int sensorData; int output; int var ; bool eventTrigger = false; bool eventTriggerTemp = false; double humidity; double temperature; double moisture;  // User's input double soilMoisture; double maxSoilMoisture=100; double minTemp =0; double maxTemp =100; // current hour bool hour; // forecast  bool notRain; bool userWater; //////// WiFiClient client; const char server[] = "api.openweathermap.org"; String apiKey = "6323acdc1da35c67469fbb7e3de6cd76";  String forecastTitle[10]; String text; int jsonend = 0; #define JSON\_BUFF\_DIMENSION 2500  // Read user input from blynk app // user control BLYNK\_WRITE(V0) {  userWater = param.asInt(); } // read max soil moisture BLYNK\_WRITE(V4) {  maxSoilMoisture = param.asDouble();  Serial.print("maxSoilMoisture: ");  Serial.println(maxSoilMoisture); } // read min temperature BLYNK\_WRITE(V5) {  minTemp = param.asDouble();  Serial.print("min Temperature: ");  Serial.println(minTemp); } // read max temperature BLYNK\_WRITE(V6) {  maxTemp = param.asDouble();  Serial.print("max Temperature: ");  Serial.println(maxTemp); } // read soil moisture BLYNK\_WRITE(V7) {  soilMoisture = param.asDouble();  Serial.print("Soid Moisture : ");  Serial.println(soilMoisture); }  // This function, read data(humidity, temperature, soil moisture) from sensor, then sends to blynk void SensorData() {  humidity = dht.getHumidity();  temperature = dht.getTemperature();  Serial.print("humidity ");  Serial.println(humidity);  Serial.print("temperature ");  Serial.println(temperature);   sensorData = analogRead(sensorpin);  moisture = map(sensorData,0,1023,100,0);  Serial.print("moisture ");  Serial.println(moisture);  // send data to blynk  Blynk.virtualWrite(V1, temperature);  Blynk.virtualWrite(V2, humidity);  Blynk.virtualWrite(V3, moisture); } // this function control watering  void Water(){  // user control  if(userWater == 1){  digitalWrite(waterPin,1);  water == true;  }  else{  digitalWrite(waterPin,0);  water == false;  }  bool hourDemo = true;  bool notRainDemo = true;    // automatic control  if(moisture < soilMoisture && water == false && hour ==true && userWater == 0 && notRain == true){  digitalWrite(waterPin,1);  water == true;  }  else if(moisture >= soilMoisture && userWater == 0){  digitalWrite(waterPin,0);  water = false;  } } void sendNotify(){  // send notify if moisture > max moisture  if(moisture > maxSoilMoisture && eventTrigger == false){  Serial.println("max mois");  eventTrigger = true;  Blynk.logEvent("maxmoisture","Soild Moisture is higher max");  }  else if(moisture <= maxSoilMoisture){  eventTrigger = false;  }  // send notify if temperature < min temperature  if(temperature< minTemp && eventTriggerTemp==false){  eventTriggerTemp = true;  Serial.println("min temp");  Blynk.logEvent("mintemp","Temperatur is lower min");  }  // send notify if temperature > max temperature  else if(temperature > maxTemp && eventTriggerTemp == false){  eventTriggerTemp = true;  Serial.println("max temp");  Blynk.logEvent("maxtemp","Temperatur is higher max");  }  else if(temperature<= maxTemp && temperature>=minTemp){  eventTriggerTemp = false;  } } // get forecast data void getWeatherData() {  Serial.println("\nStarting connection to server...");   if (client.connect(server, 80)) {   Serial.println("connected to server");   // Make a HTTP request:   client.print("GET /data/2.5/forecast?lat=10.750&lon=106.667");   client.print("&appid=6323acdc1da35c67469fbb7e3de6cd76");   client.print("&cnt=8");   client.println("&units=metric");   client.println("Host: api.openweathermap.org");   client.println("Connection: close");   client.println();   } else {   Serial.println("unable to connect");   }   String line = "";   while (client.available()) {  line = client.readStringUntil('\n');   StaticJsonBuffer<30000> jsonBuffer;   JsonObject& root = jsonBuffer.parseObject(line);   for(int i=0;i<8;i++){  forecastTitle[i] = root["list"][i]["weather"][0]["main"].as<String>();  }  notRain = true;  for(int i=0;i<8;i++){  Serial.println(forecastTitle[i]);  }  for(int i=0;i<8;i++){  if(forecastTitle[i]=="Rain"){  notRain = false;  break;  }  }  Serial.println(notRain);  } } // get current hour void getHour(){  n.update();  if(n.getHours()>=10 && n.getHours() <= 15){  hour = false;  }  else{  hour = true;  }  Serial.println(n.getHours()); } void loop() {  Blynk.run();  timer.run(); } void setup() {  // Debug console  Serial.begin(9600);  // connect to blynk  Blynk.begin(auth, ssid, pass);  // call function every timer (1000 = 1s)  timer.setInterval(1000,sendNotify);  timer.setInterval(600000,getWeatherData);  timer.setInterval(1000,SensorData);  timer.setInterval(1000,Water);  timer.setInterval(1000,getHour);  pinMode(waterPin,OUTPUT);   dht.setup(5, DHTesp::DHT11);  // connect to WIFI  WiFi.begin(ssid,pass);  Serial.println("connecting");  while (WiFi.status() != WL\_CONNECTED) {  delay(500);  Serial.print(".");  }  Serial.println("WiFi Connected");  n.begin(); } |
| --- |

## Schematic

****

## **Giao diện trên ứng dụng Blynk**



# **TỔNG KẾT**

## Tổng kết và đánh giá sản phẩm

* Sản phẩm hoàn thiện được các chức năng chính như cung cấp giao diện hiển thị các giá trị nhiệt độ, độ ẩm không khí, môi trường. Chức năng tự động tưới cho cây trong điều kiện thích hợp. Và gửi cảnh báo cho người dùng khi môi trường bất lợi cho cây phát triển.
* **Ưu điểm:**
* Sản phẩm với thiết kế đơn giản, giá thành rẻ, giao diện đơn dễ sử dụng, giúp người dùng dễ dàng hơn trong việc chăm sóc chậu cây.
* **Nhược điểm:**
* Sản phẩm chưa có chức năng lưu lại lịch sử thay đổi của nhiệt độ, độ ẩm, và lịch sử tự động tưới cho cây
* Chỉ đo được 1 vùng đất nhỏ của chậu

## Hướng phát triển sản phẩm

* Nhóm sẽ phát triển thêm một số tính năng cho sản phẩm, giúp người dùng có thể theo dõi nhiều chỉ số khác của môi trường như: nồng độ các chất dinh dưỡng có lợi cho cây trong đất vào từng thời điểm,...
* Hoàn thiện về phần cứng giúp sản phẩm gọn gàng, thẩm mỹ hơn.

## Bài học kinh nghiệm

* Cần thiết kế lịch trình làm việc, lên kế hoạch và phân chia công việc chi tiết hơn
* Các thành viên cần có sự chuẩn bị trước về ý tưởng một cách phong phú từ sớm để sản phẩm đồ án làm ra đạt được chất lượng và đáp ứng được yêu cầu từ phía giảng viên tốt nhất.
* Nhóm làm việc cần cố gắng giữ đúng tiến độ sản phẩm theo như kế hoạch đã đề ra.

# TÀI LIỆU THAM KHẢO

[1] Tưới tự động Maka, *“Hệ thống tưới cây thông minh”.* Địa chỉ: <https://maka.vn/blogs/giai-phap-tuoi-tu-dong/he-thong-tuoi-cay-thong-minh>

[2] *“Hệ thống nông nghiệp thông minh phiên bản ESP8266”.* Địa chỉ:

<https://hocarm.org/tu-lam-he-thong-nong-nghiep-thong-minh-voi-esp8266/>

[3] *“Cộng đồng Arduino Việt Nam”*. Địa chỉ: <http://arduino.vn/>