|  |  |
| --- | --- |
| **A picture containing text, clipart  Description automatically generated** | **DEPT OF COMPUTER AND COMMUNICATION ENGINEERING**  **Internet of Things: Foundations and Applications Lab**  **MMH: ITFL316064E** |
| Group:  Bùi Đức Hiền - 18119015  Phan Thị Mai Linh - 18119025  Phạm Minh Quân - 18161031 | |

**Project: Design and implement a product management system for a shop that provides VietSub service ROMs for Xiaomi phones.**

**TABLE OF CONTENTS**

[I. Introduction 1](#_Toc105948178)

[II Design and Implementation 2](#_Toc105948179)

[1 Hardware Devices 2](#_Toc105948180)

[1.1 Some components of hardware devices 2](#_Toc105948181)

[1.2 Design and Implementation 7](#_Toc105948182)

[2. Website 10](#_Toc105948183)

[2.1 HTML (tag, features), CSS (style), JavaScript. 10](#_Toc105948184)

[2.3 Interface and Function of website 25](#_Toc105948185)

[REFERENCE 33](#_Toc105948186)

**LIST OF PICTURES**

[Figure 1.1: Members of our group 1](#_Toc105952771)

[Figure 1.2: Module RFID RC522 3](#_Toc105952772)

[Figure 1.3:Module ESP32S NodeMCU LuaNode32 4](#_Toc105952773)

[Figure 1.4: Module DHT11 5](#_Toc105952774)

[Figure 1.5: Start Pulse of DHT11 5](#_Toc105952775)

[Figure 1.6: Response Pulse of DHT11 5](#_Toc105952776)

[Figure 1.7: Data Pulse of DHT11 6](#_Toc105952777)

[Figure 1.8: End Pulse of DHT11 6](#_Toc105952778)

[Figure 1.9: Module Relay 5V 7](#_Toc105952779)

[Figure 1.10: The Block Diagram of System 8](#_Toc105952780)

[Figure 1.11: The flowchart of system 9](#_Toc105952781)

[Figure 1.12: The hardware device of the system 10](#_Toc105952782)

[Figure 2.1: Display three elements on the same row 15](#_Toc105948345)

[Figure 2.2: Display two elements on the same row 16](#_Toc105948346)

[Figure 2.3: Step 1: In JavaScript, we want to use google firebase version 8, so we need to import necessary SDK for firebase 16](#_Toc105948347)

[Figure 2.4: Step 2: We must create project on google firebase. 16](#_Toc105948348)

[Figure 2.5: Step3: We set the name of a project and click continue button until the end. 17](#_Toc105948349)

[Figure 2. 6: Step 4: Select default Account for firebase and press continue. 17](#_Toc105948350)

[Figure 2.7: Step 5: Now, your project is created and click continue button. 17](#_Toc105948351)

[Figure 2.8: Step 6: Click the website icon on the screen and create a project to link between the website and google firebase. 18](#_Toc105948352)

[Figure 2.9: Step 7: Below is the code to link between the website and google firebase and website. 18](#_Toc105948353)

[Figure 2.10: Step 8: Next, we will create a Realtime database to read and write data to firebase. 19](#_Toc105948354)

[Figure 2.11: Step 9: Select test mode so that hardware devices and websites can read and write data on the database. 19](#_Toc105948355)

[Figure 2.12: Setting up Authentication 20](#_Toc105948356)

[Figure 2.13: Login Form 21](#_Toc105948357)

[Figure 2.14: Javascript code for login function 21](#_Toc105948358)

[Figure 2.15: Control hardware devices on website 22](#_Toc105948359)

[Figure 2.16: Javascript code for controlling devices 23](#_Toc105948360)

[Figure 2.17: Displaying temperature and humidity on website 23](#_Toc105948361)

[Figure 2.18: JavaScript code for displaying temperature and humidity 24](#_Toc105948362)

[Figure 2.19: Product information check function 24](#_Toc105948363)

[Figure 2.20: JavaScript code for production information check function 25](#_Toc105948364)

[Figure 2.21: The header and navigator interface 25](#_Toc105948365)

[Figure 2.22: The introduction interfaces 26](#_Toc105948366)

[Figure 2.23: The service interfaces 26](#_Toc105948367)

[Figure 2.24: The review interfaces 27](#_Toc105948368)

[Figure 2.25: About us interface 27](#_Toc105948369)

[Figure 2.26: The footer interfaces 28](#_Toc105948370)

[Figure 2.27: Login Interface 28](#_Toc105948371)

[Figure 2.28: The subpage interface 29](#_Toc105948372)

[Figure 2.29: Displaying Temperature and humidity and controlling the device 29](#_Toc105948373)

[Figure 2.30: Turning on the device 30](#_Toc105948374)

[Figure 2.31: Turning off the device 30](#_Toc105948375)

[Figure 2.32: Editing information of the product 30](#_Toc105948376)

[Figure 2.33: Checking information of the product 31](#_Toc105948377)

**LIST OF TABLES**

[Table 2.1: HTML tag 12](#_Toc105952793)

[Table 2.2: CSS attributes 14](#_Toc105952794)

[Table 2.3: JavaScript functions 15](#_Toc105952795)

# **I. Introduction**

In this system, we will implement a product management system consisting of two components: a hardware device checking information about products and a website with a function that displays product information such as the product's name, description, and price.

In the hardware device, we will check product information by integrating RFID tags on the product and proceed to read the product's information through the RFID card reader module. Then the hardware device will send the read information to the website to display product information to the store manager. In addition, we integrate additional functions for hardware devices such as measuring the temperature and humidity of the environment and connecting to peripherals such as lights and fans so that users can control devices on the device with just one click on the website.

We will build 1 main page and one subpage on the website's second component. The main page has the primary function of introducing information about the store and the products that the store serves to users. The subpage is designed for the store admins or employees to perform system functions such as checking the store's product information, displaying the temperature and humidity in the store, or adjusting the settings control peripherals connected to the hardware. In addition, this page can also change the information about the product on the main page. We will use the login function so that admins or store employees want to access the main page to the secondary page to perform the system's functions.

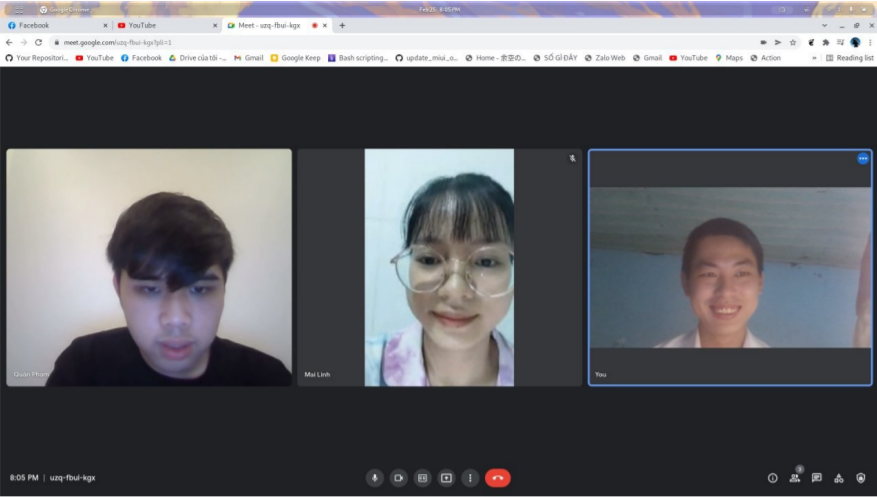


Figure 1.1: Members of our group

**II Design and Implementation**

## **1 Hardware Devices**

### **Some components of hardware devices**

**RFID RC522**

RFID (Radio Frequency Identification) is an automatic identification method based on the ability to store and receive data remotely using RFID tag devices. RFID tags are small and can be attached to products, people, and animals. An RFID tag contains silicon chips and antennas that allow receiving commands and responding to RF radio frequencies from an RFID transponder. It is a technology that confirms object data by radio waves to identify, track and store information in a tag (Tag). The Reader scans the card data and sends the information to the card’s data storage database.

RFID (Radio Frequency Identification) is an automatic identification method based on the ability to store and receive data remotely using RFID tag devices. RFID tags are small and can be attached to products, people, and animals. An RFID tag contains silicon chips and antennas that allow receiving commands and responding to RF radio frequencies from an RFID transponder. It is a technology that confirms object data by radio waves to identify, track and store information in a tag (Tag). The Reader scans the card data and sends the information to the card’s data storage database.

RC522 RFID module uses Phillip’s MFRC522 IC used to read and write data for 13.56MHZ NFC tags; with a low price and compact design, this module is the first choice for applications on reading and writing RFID tags...

The characteristics of RC 522:

* Power: 3.3VDC, 13 – 26mA
* Standby current: 10-13mA
* Line at rest:
* Carrier frequency: 13.56MHz
* Operating distance: 0～60mm（mifare1 card）
* Communication: SPI
* Data transfer rate: up to 10Mbit/s
* Supported RFID cards: mifare1 S50, mifare1 S70, mifare UltraLight, mifare

Pro, mifare Desfire

* Dimensions: 40mm×60mm



Figure 1.2: Module RFID RC522

**ESP 32**

ESP32 is a chip that integrates Wifi and Bluetooth technology with ultra-low power consumption technology. It provides a powerful integrated platform that meets the needs of best-in-class performance, flexibility, compact design, high performance, and reliability in a wide range of applications. The ESP32 chip family includes ESP32-D0WDQ6, ESP32-D0WD, ESP32-D2WD, and ESP32-S0WD. Espressif has designed and manufactured some modules to make it easier for users to access the ESP32 series of chips.

ESP32-WOOM-32 version is a Wifi microcontroller (MCU) module (Wireless Fidelity) – BT (Bluetooth) – BTE (Bluetooth Low Energy). The famous and powerful serve a variety of applications, from simple applications such as device control and sensor readings to tasks complex as voice encoding, streaming music, and MP3 decoding, ... This module is built with the core of ESP32-D0WDQ6 chip, which is designed to be scalable. There are 2 CPU cores (Central Processing Unit) that can be controlled separately, and the clock frequency ranges from 80MHz to 240MHz. Due to the integration of Bluetooth, Bluetooth LE, Wifi, many applications can be performed on the module. Using Wifi allows you to connect in a wide, unlimited range, and using Bluetooth allows an easy connection. The module also supports data transmission up to 150Mbps.

**A close-up of a computer chip

Description automatically generated with low confidence**

Figure 1.3:Module ESP32S NodeMCU LuaNode32

The characteristics of ESP32S NodeMCU LuaNode32:

* + **Wifi** : 802.11 b/g/n/e/I (Wi-Fi 2,4 GHz).
  + **Bluetooth**: v4.2 BR/EDR and BLE
  + **CPU**: is a 32-bit processor, Named “PRO\_CPU” and “APP\_CPU” Xtensa Dual-
  + Core LX6. When we use FreeRTOS, it will correspond to Core 0 and Core 1 (protocol CPU and application CPU), Processor speed 160MHZ ~ 240 MHZ.,Flash read clock speed 40MHZ ~ 80MHZ (customizable when programming), ROM: 449 Kbyte ROM, 4MB external FLASH and RAM: 520 Kbyte SRAM, 520KB SRAM on-chip.
  + **Peripheral**: 18 channels Analog-to-Digital Converter (ADC), 3xSPI, 3xUART, 2xI2C, 16 channels of PWM output, 2 Digital-to-Analog converters (DAC), 2xI2S, 10 GPIO capacitive sensors.

**Module DHT11**

DHT11 is a widely used temperature and humidity sensor with low cost and accessible data collection via a 1-wire communication standard. Has a fast response time and good anti-interference ability. At the same time, the sensor has a built-in signal preprocessor that helps the data be received accurately without any computation.

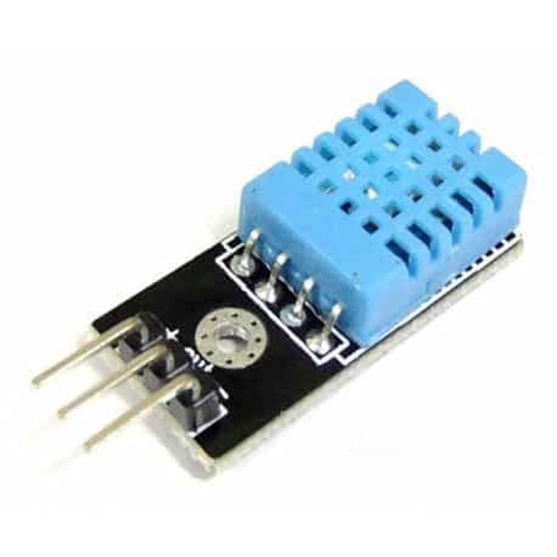


Figure 1.4: Module DHT11

The operating of DHT11:

DHT11 uses only one wire to communicate. The communication process is divided into three steps:

first, it sends a request to the sensor, then the sensor sends a feedback pulse, and then it starts sending data totaling 40 bits to the microcontroller.

To initiate communication with the DHT11, we first send a start pulse to the sensor. To provide a start pulse, pull the data pin low for a minimum of 18ms and then pull it high.

Shape, polygon

Description automatically generated

Figure 1.5: Start Pulse of DHT11

After receiving a start pulse, the sensor will send a feedback pulse to indicate that the DHT11 has received a start pulse. The feedback pulse goes low for a period of 54us, then goes high for 80us.

Diagram

Description automatically generated

Figure 1.6: Response Pulse of DHT11

After sending the feedback pulse, DHT11 will send data containing temperature and humidity values. The data frame is 40bit long, divided into five parts (bytes), and each part is 8bit. Of these five parts, the first two will contain the humidity value; the first 8 bits are the integer part value, and the remaining 8 bits contain the decimal value. The following two sections will contain the value temperature (℃) in decimal. The last part is 8bit to test for temperature and humidity measurement. After receiving data, the DHT11 pin will stay in low power consumption mode until the next start pulse.

Diagram

Description automatically generated

Figure 1.7: Data Pulse of DHT11

After sending 40bit data, DHT11 will go low 54us, then go high, and go to sleep. DHT11 is a widely used temperature and humidity sensor with low cost and accessible data collection via a 1-wire communication standard. Has a fast response time and good anti-interference ability. At the same time, the sensor has a built-in signal preprocessor that helps the data be received accurately without any computation.

Shape, rectangle

Description automatically generated

Figure 1.8: End Pulse of DHT11

The characteristics of DHT11:

* Source: 3 ~ 5 VDC.
* Current: 2.5mA max (when transmitting data).
* Humidity measurement range: 20%-90% RH (error 5%RH)
* Temperature measurement range: 0-50°C (2°C error)
* Maximum sampling frequency: 1Hz (1 second/time)
* Dimensions: 15mm x 12mm x 5.5mm.

**Module Relay 5V**

Module 1 Relay includes one relay operating at a voltage of 5VDC and withstands voltages up to 250VAC 10A. Module 1 low-level trigger relay is designed firmly, with good insulation. There is a relay trigger circuit on the module using transistor and optical isolation IC to completely isolate the control circuit (microcontroller) from the relay to ensure stable operation of the microcontroller. Available headers are convenient when connecting to microcontrollers. This module complies with international safety standards. The input area and the load area have isolating slots.

A picture containing electronics

Description automatically generated

Figure 1.9: Module Relay 5V

The characteristics of Module Relay 5V:

* Use DC 5V feed voltage.
* Each relay consumes about 80mA current.
* Maximum switching voltage: AC250V ~ 10A or DC30V ~ 10A.
* There is an indicator light on each relay.
* The trigger signal level 0 or 1 can be selected via the jumper.
* Dimensions: 1.97 in x 1.02 in x 0.75 in (5.0 cm x 2.6 cm x 1.9 cm)
* Weight: 17g

### **1.2 Design and Implementation**

With the project’s requirements to read product information by RFID tags and send that information to the cloud, we decided to use IC RC522 for the input data block and ESP 32 microcontroller for the

central controller to quickly bring data from input to the cloud via standard Wifi communication. Besides, we also actually read the temperature signal of the environment through the temperature sensor and send it to the website so that users can easily control home appliances such as fans or

heaters. We decided to divide my system into four main blocks: sensor block (using temperature sensor), input block (Module RC522), central processing block (ESP 32 Module), and output block (Relay module). Below is the block diagram of the system.

Diagram

Description automatically generated

Figure 1.10: The Block Diagram of System

The operating of the whole system: The microcontroller will read the temperature and humidity values from the sensor block, then process the data and send it to the cloud for users to observe. Besides, if the user wants to turn on/off the devices on the output block, the control signal will be transmitted from the cloud to the microcontroller to process and turn on / off the relay according to control signal. When the user wants the device to view product information from the RFID tag, the control signal will be transmitted from the cloud to the microcontroller. The microcontroller will read the RFID tag data through the RC522 module and send the information of the RFID tag. Back to the cloud.

The algorithm of the whole system:

After analyzing and selecting, and pairing suitable components for hardware devices, we will proceed to program the esp 32 microcontrollers to make the system function properly. Below is the flowchart of the system.

Diagram

Description automatically generated

Figure 1.11: The flowchart of system

In this flowchart, the system will initially check if there is an RFID read control signal; if yes, it will read data from the RFID tag and send it to firebase. Until there is no more signal to read the RFID tag, the system will read the temperature value from the temperature sensor and send it to firebase. At the same time, check if there is a signal to turn on the relay or not; if so, the system will turn on the relay; if not, the relay will be turned off and repeated until the power is cut off, the program will end.

After connecting all the components and uploading the code to the esp 32 microcontroller, below are the product results:

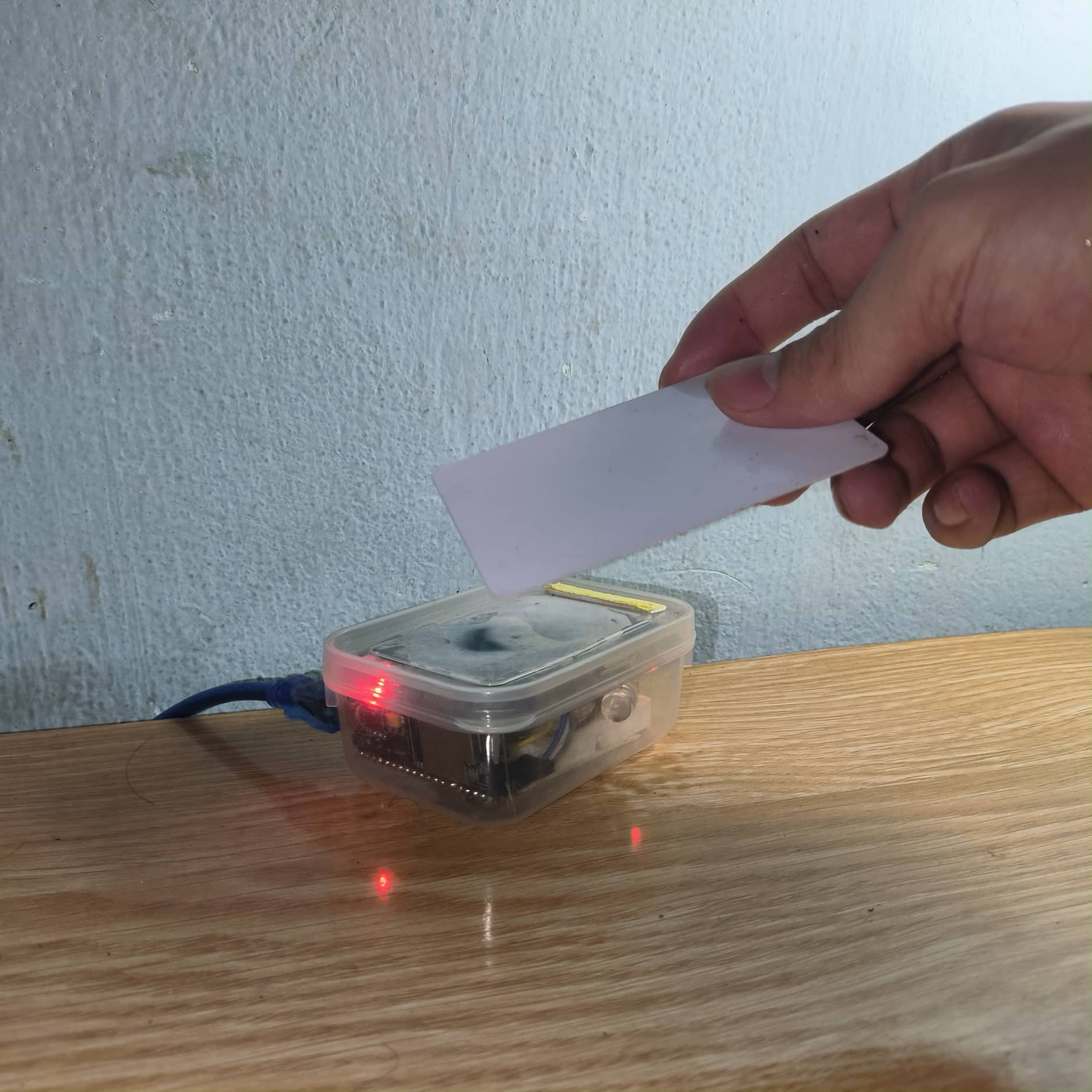


Figure 1.12: The hardware device of the system

Because during this time, it is challenging to buy some components, and this is just a project to implement the system’s functions, so we have changed some components in the hardware device, but the function is still the same. We decided to use the CPU temperature sensor inside esp 32 to replace the DHT 11 sensor in thermal measurement. Alternatively, we decided to use one led controller to model device control in device control.

Next, we will write a website so that users can observe the data from the RFID tag and control the devices by clicking on the website.

## **2. Website**

### **2.1 HTML (tag, features), CSS (style), JavaScript.**

**The definition of HTML:**

HTML (Hypertext Markup Language) is a markup language for the web that defines the structure of web pages. HTML is not a programming language; it is a markup language, which means it is a system for identifying and describing the various components of a document such as headings, paragraphs, and lists.

**Some common HTML tag:**

|  |  |  |
| --- | --- | --- |
| **HTML tag** | **Systax** | **Function** |
| 1. Doctype tag | <!DOCTYPE html> | Declaration defines that this document is an HTML5 document. |
| 2. Html tag | <html>...</html> | The html element is the root of an HTML page. |
| 3. Header tag | <head>...</head> | The head element is a container for metadata, such as title, content, keyword, language,... |
| 4. Title tag | <title>Page Name</title> | The title tag defines a title for the HTML page. |
| 5. link tag | <link rel= “stylesheet” href = “...” > | The link tag creates a link to external style sheet, or external libraries for css, html, javascript. |
| 6. Script tag | <script> ... </script> | The script tag contain a place for Javascript to excuted link, or link to external library for Javascript. |
| 7. Body tag | <body> ... </body> | The body tag is the document’s body, and a container for all the visible content, such as headings, paragraph, image,... |
| 8. Div tag | <div> ... </div> | The div tag is a division or a section in HTML document. It is used as a container for HTML elements – which is then styled with CSS or manipulated with JavaScript. |
| 9. Label tag | <label> ... </label> | The label tag defines several element. |
| 10. Button tag | <button type = “button” >... </button> | The button tag defines a clickable tag. |
| 11. Img tag | <img src = “...” alt = “...”> | The img tag used to embed an image in an HTML. |
| 12. Input tag | <input type = “...” > | The input tag specifies an input field where the user can enter data. It displays in several ways, depending on the type attribute such as email, pasword,... |
| 13. Ul & Li tag | <ul>  <li>...</li>  <li>...</li>  </ul> | The li tag defines a list item.  The ul tag is used inside unordered list. |
| 14. h3 tag | <h3>...</h3> | The h3 tag defines a large heading. |
| 15. p tag | <p> ... </p> | The p tag defines a paragraph. |
| 16. i tag | <i> ... </i> | The i tag specifies italic text without any extra importance. |
| 17. iframe tag | <iframe src = “...”> </iframe> | The iframe tag is used to embed another document within the current HTML document. |

Table 2.1: HTML tag

**The definition of CSS:**

Cascading Style Sheets (CSS) is a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects such as SVG, MathML or XHTML). CSS describes how elements should be rendered on screen, on paper, in speech, or on other media.

**The CSS systax:**

* The selector points to the HTML element you want to style.
* The declaration block contains one or more declarations separated by semicolons.
* Each declaration includes a CSS property name and a value, separated by a colon.
* Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.
* For example: p { color: #ccc; font-size: 16px;}
* **Some common CSS attributes:**

|  |  |
| --- | --- |
| **CSS selector** | **Function** |
| 1. Margin | The margin is used to create around elements, outside of any defined borders. |
| 2. Padding | The padding is used to create space around an elements’s content, inside of any defined borders. |
| 3. Box-sizing | The box-sizing is used to fix the padding and border in an element’s total width and height. |
| 4. Font-size | The font-size is used to set the size of the font. |
| 5. Font-family | The font-family is used to specify the font for an element. |
| 6. Width | The width is used to sets the width of an element |
| 7. Height | The height is used to set the height of an element |
| 8. Background-color | The background-color is used to specifies the background color of an element. |
| 9. Color | The color is used to sets the color of an element |
| 10. Background-image | The background-image is used to set the image for the background |
| 11. display | It specifies the display behavior (the type of rendering box) of an element. |
| 12. line-height | It sets the line height for different <div> elements |
| 13. float | It positions and formats content e.g. let an image float left to the text in a container. |
| 14. min-width | It defines the minimum width of an element. |
| 15. Border-radius | It defines the radius of the element's corners |
| 16. Border | It specifies the style, width, and color of an element's border. |
| 17. Transition | It changes property values smoothly, over a given duration. |
| 18. Overflow | It specifies what should happen if content overflows an element's box |
| 19. Top | It sets the top edge of the positioned. |
| 20. Left | It sets the left edge of the positioned |
| 21. Right | It sets the right edge of the positioned. |
| 22. Bottom | It sets the bottom edge of the positioned. |
| 23. Text-align | It sets the text alignment for different <div> elements. |
| 24. align-items | It specifies the default alignment for items inside the flexible container. |
| 25. justify-content | It aligns the flexible container's items when the items do not use all available space on the main-axis (horizontally). |
| 26. Text-decoration | It sets different text decorations for elements. |
| 27. Max width | It defines the maximum width of an element |
| 28. Z-index | It specifies the stack order of an element |
| 29. Transform | It applies a 2D or 3D transformation to an element. This property allows you to rotate, scale, move, skew, etc., elements. |
| 30. Text-transform | It transforms text in different <div>elements.content: property is used with the ::before and ::after pseudo-elements, to insert generated content. |
| 31 Transform | It applies a 2D or 3D transformation to an element. This property allows you to rotate, scale, move, skew, etc., elements |
| 15. Border-radius | It defines the radius of the element's corners |
| 32 @keyframe | The **@keyframes rule** specifies the animation code. The animation is created by gradually changing from one set of CSS styles to another. |
| 33 @media | The **@media rule** is used in media queries to apply different styles for different media types/devices. |

Table 2.2: CSS attributes

**The definition of Javascript**

JavaScript is a scripting language that adds interactivity and behaviors to web pages, including these:

* Checking form entries for valid entries.
* Swapping out styles for an element or an entire site.
* Loading scrolling feeds with more content automatically.
* Making the browser remember information about users.
* Building interface widgets, such as embedded video players or special form inputs.

**Several Javascript functions are used in websites.**

|  |  |
| --- | --- |
| **Name** | **Function** |
| document.querySelector(“.class”) or document.querySelector(“.classes”) | It gets the first element or all elements int the document with class “class” or “classes”. |
| document.getElementById(“ID”) | It gets an element with a specified value. |
| element.addEventListener(“action”, function) | The function will perform the function to listen to the event that occurs on the element and execute the corresponding function. |
| element.innerHTML | It sets or return the HTML content of an element. |
| event.stopPropagation() | It prevents propagation of the same event from being call. |
| element.classList.add(“class”) or element.classList.remove(“class”). | It adds or removes one or more token to the list. |
| document.createElement() | It creates a new HTML element. |

Table 2.3: JavaScript functions

In the website part of this project, we will use HTML and CSS to design the external interface of a website. In addition, we will use a JavaScript programming language to calculate and control the internal functions of the website, such as firebase links, and perform store product information checks. Besides, users can control the on and off of devices connected to the hardware.

2.2 **Function Explain**

* **Display multiple elements on the same row using the row and column method.**

This technique will display 2,3 or more content on the same row. We will use the tables' row and column method to perform this function. As shown below, there will be 1 row and three columns; we will use the float = left and width = 33.33% properties to display all three content on the same row. Next, we will implement CSS so that the distance between the three equally spaced elements is 8px. We will use Padding left and right for each column. The length is 8px, making the row margin with a value of -8px spaceless. Both external content. Using positive column padding and negative row margin will display three products on the same row at a constant distance from each other.

**Text

Description automatically generatedGraphical user interface

Description automatically generated with low confidence**

Figure 2.1: Display three elements on the same row

Use the same CSS with 1 row and two columns. We just need to change the width to equal 50%.

Text

Description automatically generatedShape, rectangle

Description automatically generated

Figure 2.2: Display two elements on the same row

* **Google Firebase Linking**

We will use google firebase to store the system's data. From there, we will read and write that data on the website and the hardware device.

Some steps to link google Firebase.

Figure 2.3: Step 1: In JavaScript, we want to use google firebase version 8, so we need to import necessary SDK for firebase

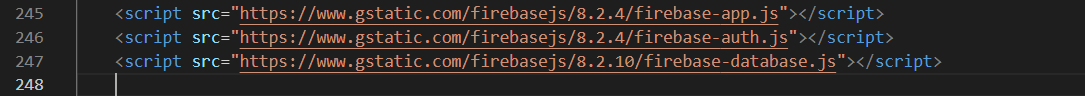


Figure 2.4: Step 2: We must create project on google firebase.

Graphical user interface, application

Description automatically generated

Figure 2.5: Step3: We set the name of a project and click continue button until the end.A picture containing text

Description automatically generated

Figure 2. 6: Step 4: Select default Account for firebase and press continue.

Graphical user interface, text, application

Description automatically generated

Figure 2.7: Step 5: Now, your project is created and click continue button.

A picture containing toy, aircraft, vector graphics

Description automatically generated

Figure 2.8: Step 6: Click the website icon on the screen and create a project to link between the website and google firebase.

Graphical user interface, website

Description automatically generated

Figure 2.9: Step 7: Below is the code to link between the website and google firebase and website.

Text

Description automatically generated

Figure 2.10: Step 8: Next, we will create a Realtime database to read and write data to firebase.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 2.11: Step 9: Select test mode so that hardware devices and websites can read and write data on the database.

Graphical user interface, application

Description automatically generated

Finally, we will paste the code copied in step 7 and import the necessary SDK for firebase in step 1. Then, we check that there is no error on the console screen, then the system has successfully connected between the website and firebase.

* **Login Function**

Firstly, we will set up the Authentication feature on firebase and choose the login method as email/password to set up accounts for users.

Graphical user interface, application, Word

Description automatically generated

Figure 2.12: Setting up Authentication

Secondly, we will create a login form with HTML, CSS JavaScript and perform the process of getting the account and password values entered by the user.

Graphical user interface, application

Description automatically generated

Figure 2.13: Login Form

Finally, we will use JavaScript to implement the login functionality, specifically after the user enters the account and password and presses the login button on the screen. The system will listen to the button click event from the user and will perform a check if the user's account and password match the account and password added on firebase. A move from the leading site to the secondary site will be performed if there is a match. If it is wrong, it will print out the wrong account or password on the screen.

Text

Description automatically generated

Figure 2.14: Javascript code for login function

* **Control hardware devices on website**

In this function, the Website will allow users to control the device on and off the hardware through clicking on the esp 32 button on the screen.

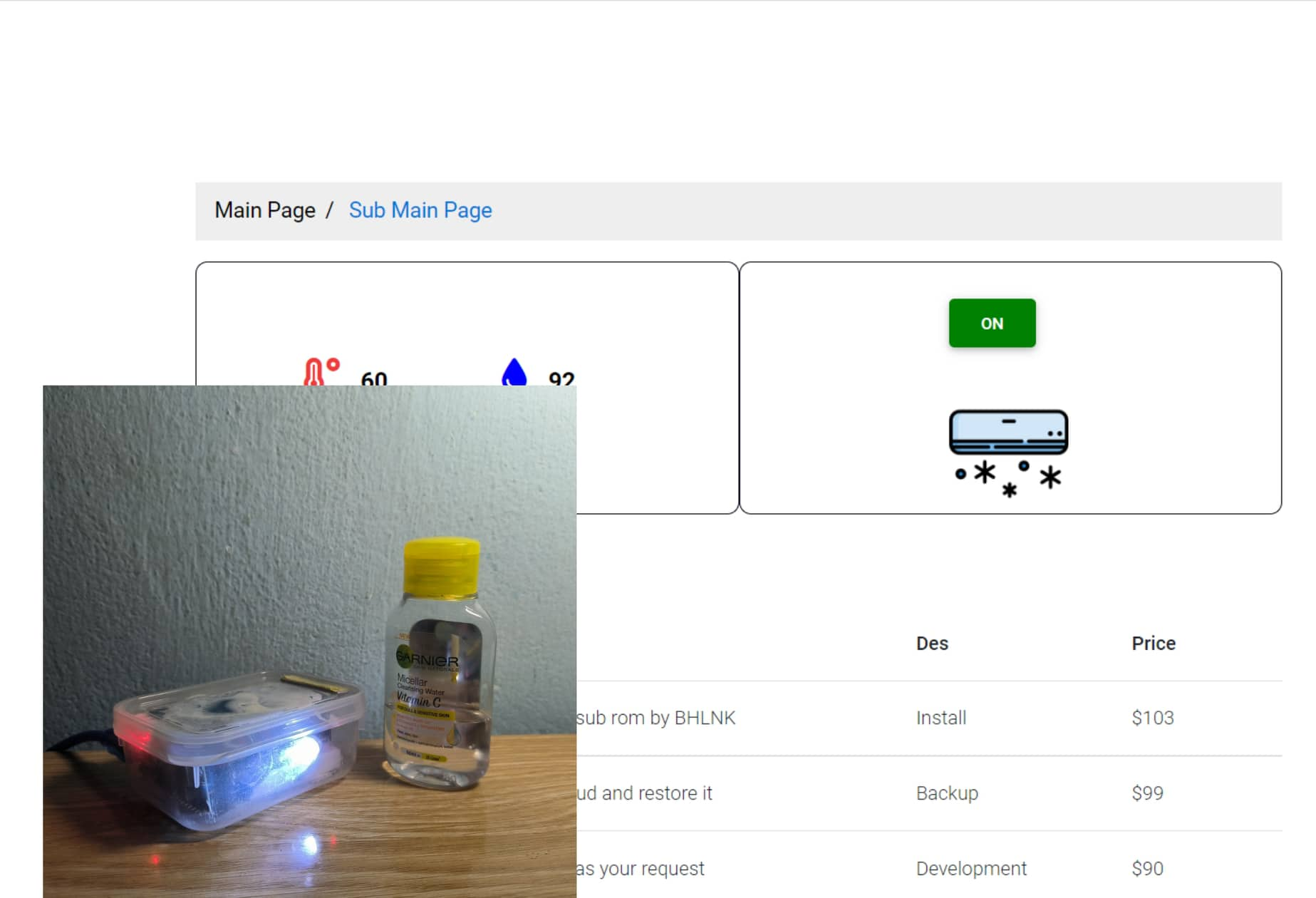


Figure 2.15: Control hardware devices on website

In the function, we use the add Event Listener process to listen for user events. When the user clicks, it will perform the corresponding function. Specifically, in this case, when the user clicks on the website, it will send a control signal to firebase, and the microcontroller will read that control signal and

perform the corresponding function. To Perform two functions on one button, we use a temporary variable to save the previous state from which to perform the next device stage.

**Text

Description automatically generated**

Figure 2.16: Javascript code for controlling devices

* **Display temperature and humidity on website.**

In this function, the website will display the temperature and humidity sent by esp32 to firebase.

Graphical user interface

Description automatically generated

Figure 2.17: Displaying temperature and humidity on website

In the function, we will use the id attribute to display the results from the hardware to the website. First, the system will take the data at the path on firebase that esp 32 has written to and assign that value to the corresponding id attribute. The website will display the value of the id attribute.

A screenshot of a computer

Description automatically generated with medium confidence

Figure 2.18: JavaScript code for displaying temperature and humidity

* **Product information check function**

In this function, the system will check the product information in the store, such as the name, description of the service, and service price, through data from the hardware device and sent to firebase. Then will display the information of that product if the information of that product is already saved on firebase's database.

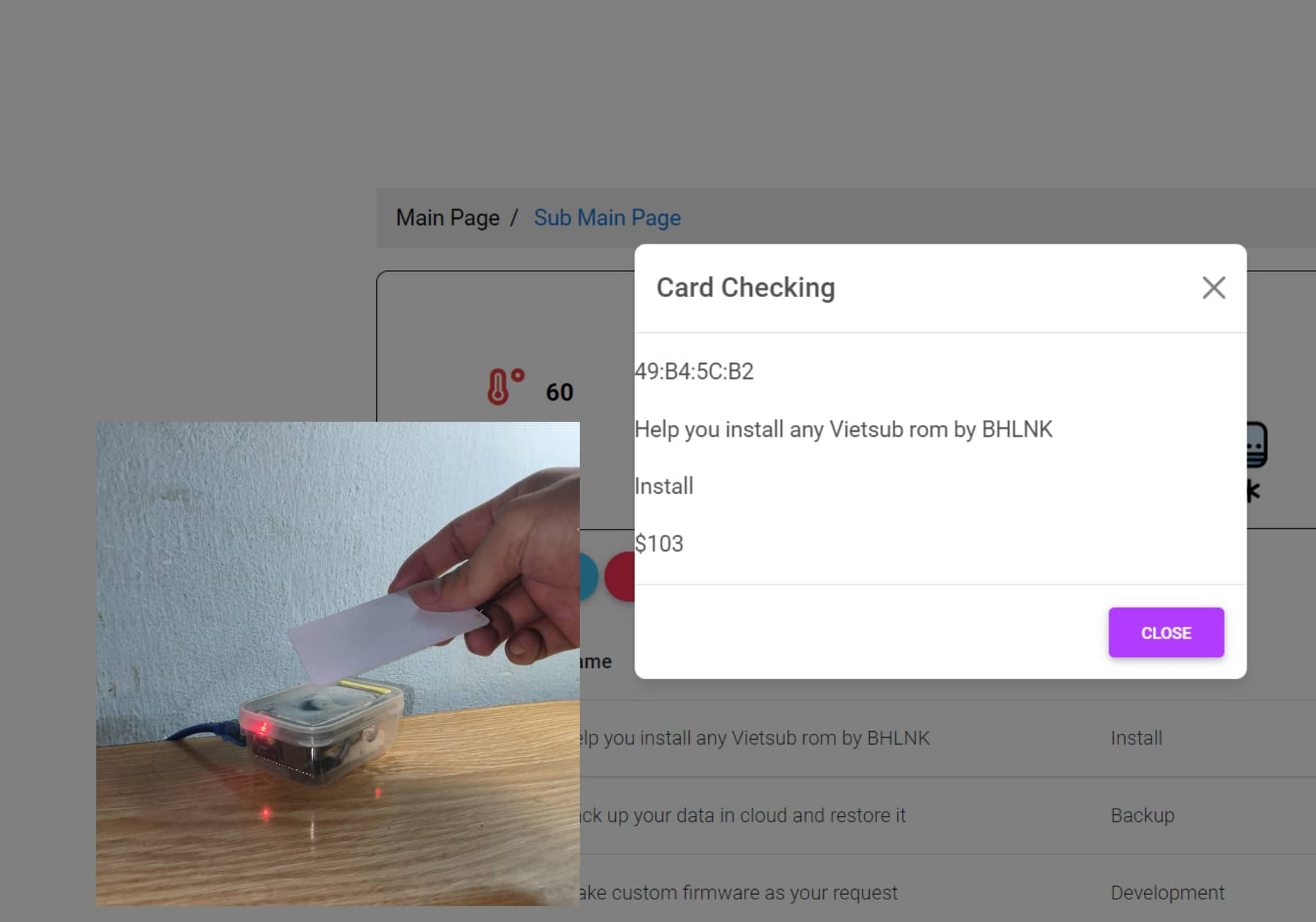


Figure 2.19: Product information check function

We will use JavaScript to implement this function. First, when the user clicks on the check button on the website, the system will allow the hardware device to read the product's information by reading the RFID tag. Then it sends the product's ID to the key\_temp key on the database. The system will take the value of that key\_temp and compare it with all the critical values of the store's products. If there is a product with a key matching key\_temp, it will print the information of that product on the screen.

**Text

Description automatically generated**

Figure 2.20: JavaScript code for production information check function

### **2.3 Interface and Function of website**

We will design two websites in the website section: the main page and the subpage. The main page has the purpose of introducing the store's information and services, ... The subsite has the primary purpose of performing the functions to help the admin, or the staff can check product information, control the store's equipment, monitor the store's temperature and humidity, and so on.

The first will come to the main website because this is a website used to introduce users to information about the store and the types of services that the store serves. The website will be divided into four sections: Header and Navigator, Introduction, Content, and Footer.

The first is the Header and Navigator section. This part of the interface will describe the logo and the buttons to link to other parts of the website related in the image below.

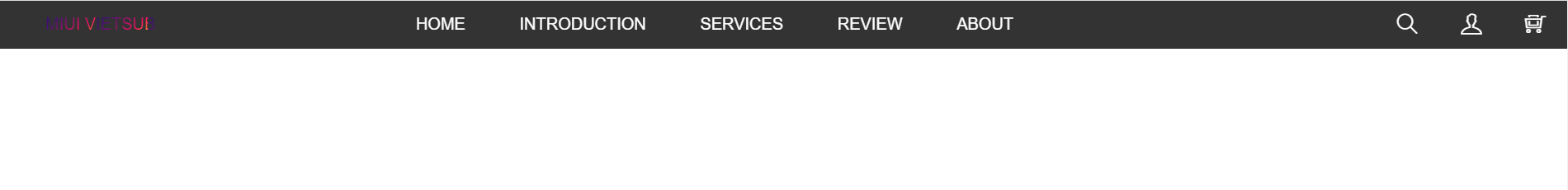


Figure 2.21: The header and navigator interface

The second is the Introduction section; this interface will describe the store's information and social networking pages, as shown in the figure below. On the left is a photo illustrating a phone we vietsub. On the right are our store's social media pages.

Graphical user interface, application

Description automatically generated

Figure 2.22: The introduction interfaces

Next is the Content section. This part of the interface will include information about the services the store serves, customer reviews for the store, and the store establishment leaders.

About the service section, the interface will describe all the services, and information on all the services of our store is shown in the figure below. In this section, all the information about the services, like name, description, and price, will be stored on firebase. This will make it easier for the owner to change the above information when the store wants to change the policy.

A picture containing timeline

Description automatically generated

Figure 2.23: The service interfaces

The review interface will describe the customer reviews for the store shown in the image below. Above is a video explaining how to install Viet sub-ROM for Xiaomi phones, below are comments about the store's service.

Graphical user interface, website

Description automatically generated

Figure 2.24: The review interfaces

As for About us, this section will introduce the members of the store described in the image below.

Diagram

Description automatically generated

Figure 2.25: About us interface

Finally, in the Footer section, the interface will describe the critical information about the store, such as the store's mail, address, and phone number, so that users can quickly contact as shown in the figure below.

Text

Description automatically generated

Figure 2.26: The footer interfaces

To transition from the main page to the subpage to execute system functions, we use the login function to implement this function. In the header and navigator on the right, there is a user icon to perform the login shown in Figure 10. The store owner or employee will log in to their account to complete the system's functions. When the user clicks on the user icon, The system will display a login form on the screen. The login form interface will allow users to enter their account and password to access the sub-site shown in the figure below. After the user enters the correct account and password and clicks the Login button in the form, the system will automatically switch from the main page to the secondary page to perform the system's functions.

Graphical user interface, website

Description automatically generated

Figure 2.27: Login Interface

After successful login, the main page interface will switch to the interface of the sub-site shown in the image below. To return to the main page, the user must click the Main Page button on the breadcrumb section.

Graphical user interface, application

Description automatically generated

Figure 2.28: The subpage interface

Next, the first two functions of the system are to monitor the temperature and humidity at the store and control the equipment at the store. The interface will divide this display into two sides, the right side displays the temperature and humidity at the store, and the left side allows users to control the device at the store with just one click shown at the bottom of the screen figure below. In the temperature and humidity measurement interface, the red icon on the right represents the temperature, and the blue icon on the right represents the humidity.

Chart

Description automatically generated with low confidence

Figure 2.29: Displaying Temperature and humidity and controlling the device

In the device control function, the interface will display a push button (ON/OFF) and an icon corresponding to turning on the device. This button will show 2 states when the user press the first time will turn on and press the second time will turn off. The content and icons will also change according to the device's operating mode as shown in the 2 images below.

Graphical user interface

Description automatically generated with medium confidence

Figure 2.30: Turning on the device

Graphical user interface, application

Description automatically generated

Figure 2.31: Turning off the device

Next are the remaining functions of the system. In the main interface image of the sub-website below, there are three buttons, Clear, Edit, and Check, corresponding to the remaining three functions of the system. The Edit function allows users to adjust the information about the store's services. When the user clicks the Edit button on the screen, the interface window will appear and allow the user to adjust the service's name, Description, and price according to the service's corresponding ID code. The figure below depicts the above function.

Graphical user interface, text, application

Description automatically generated

Figure 2.32: Editing information of the product

Finally, the function of checking information about the customer's product. When the user clicks the Check button on the screen, the system will send a control signal to the hardware device, and the hardware device will start reading the RFID tag attached to the product. Then the hardware device will send the product information to firebase. Finally, the website will display that information to the user. The image below depicts a product being read, and the website will display the product's data, such as the product's name, description, and price, on the screen.

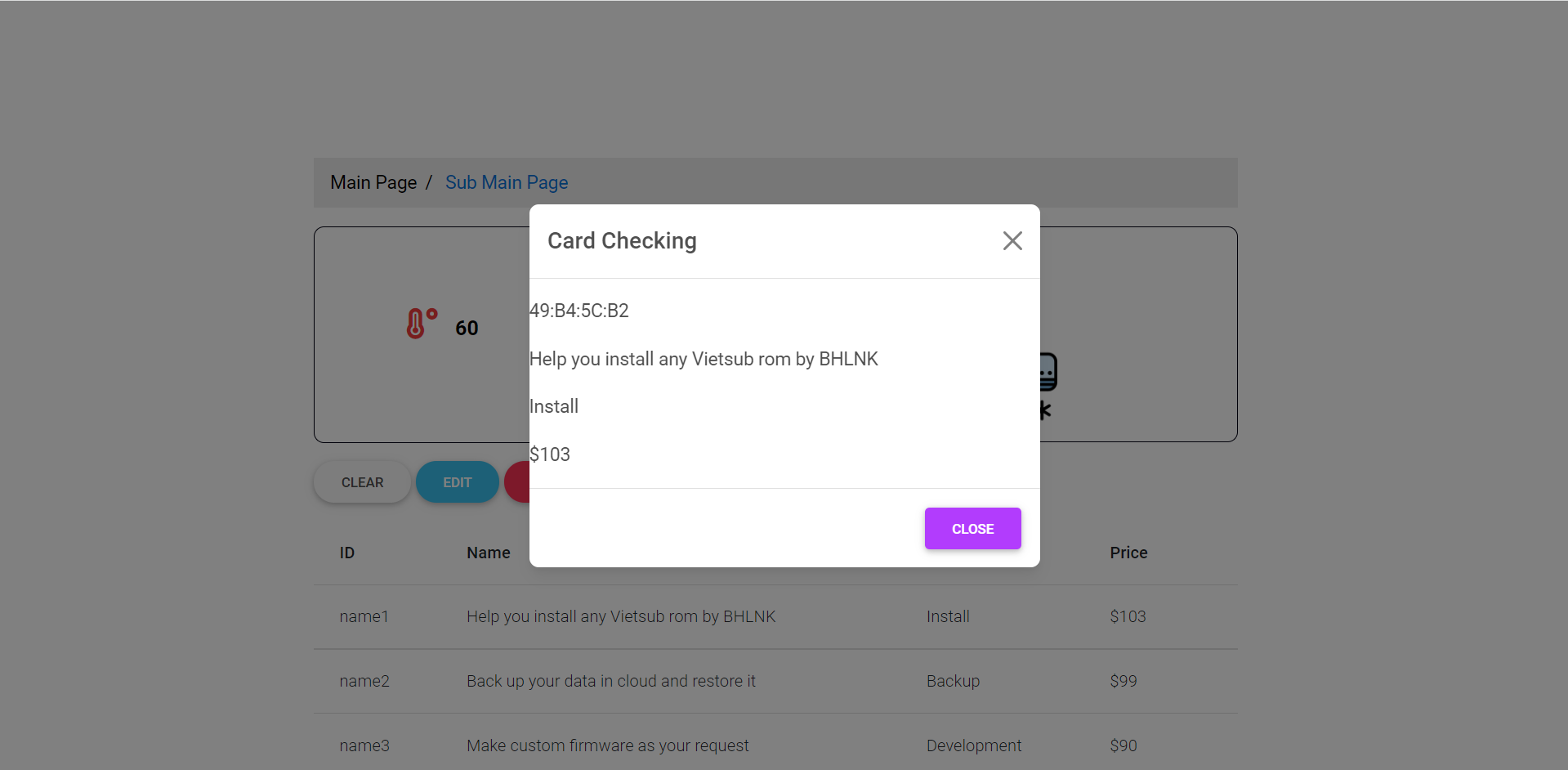


Figure 2.33: Checking information of the product

Clip demo:

Source code: <https://github.com/QuanVA03/iot_lab.git>

Domain: <https://quanva03.github.io/iot_lab/>

# **REFERENCE**

|  |  |
| --- | --- |
| [1] | "HTML CSS AND JAVASCRIPT Tutorial,"[Online]: http//www.w3schools.com. |
| [2] | SonDang, "CSS tutorial," [Online]: http//fullstack.edu.vn. |
| [3] | f. document, "Read and Write Data," [Online]. Available: https://firebase.google.com/docs/database/web/readand-write?authuser=0.. |