Project ID: 2023-1-HU01-KA210-VET-000156243

Implementation of the "Weather Station"

Project Title: Discover the Green Life with Robots **Project ID:** 2023-1-HU01-KA210-VET-000156243

Disclaimer: Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the Tempus Public Foundation. Neither the European Union nor the granting authority can be held responsible for them.

The following code (index.php) must be placed on a web server that can utilize an **SQL database**. An **ESP32 microcontroller** is placed anywhere in the world, to which a **DHT11 temperature and humidity sensor** is connected (The ESP32 code follows the PHP code). The ESP32 connects to the Internet after startup and begins functioning as a weather station. It begins sending the measured values to the server via the internet.

The page uses both the **POST and the GET method**.

I. Server-Side PHP Code Functionality (index.php)

```
<?php

$hostname = "localhost"; // The database server address

$username = "root"; // The database username

$password = "%Dht11_projekt"; // The database password

$database = "idojaras_esp32"; // The database name


// Creating a connection to the database

$conn = mysqli_connect($hostname, $username, $password, $database);

// Check if the connection was successful

if (!$conn) {

   die("Connection failed: " . mysqli_connect_error());
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
}
echo "The data connection was established with the server.<br>";
$refresh = false; // Variable for signaling automatic refresh

// Usage of POST method: Receiving data sent by the ESP32 microcontroller
if(isset($_POST["temperature"]) && isset($_POST["humidity"])) {
   $t = $_POST["temperature"];
   $h = $_POST["humidity"];

   // Check if there are already 10 records in the table
   $result = mysqli_query($conn, "SELECT COUNT(*) as count FROM dht11");
   $row = mysqli_fetch_assoc($result);
   if ($row['count'] >= 10) {
      // If there are 10 records, wait 10 seconds
      sleep(10);
      // Truncate the table and reset it
      mysqli_query($conn, "TRUNCATE TABLE dht11");
      echo "Table truncated and reset<br>";
      $refresh = true; // Set the refresh flag
   }

   // Insert the new data, the timestamp is automatically added
   $sql = "INSERT INTO dht11 (temperature, humidity) VALUES (".$t.", ".$h.")";
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
    if (mysqli_query($conn, $sql)) {
        echo "\nNew record created successfully";
        $refresh = true; // Set the refresh flag
    } else {
        echo "Error: " . $sql . "<br>" . mysqli_error($conn);
    }
}

// Usage of GET method: Querying data from the database for table display
$result = mysqli_query($conn, "SELECT * FROM dht11");
?>
```

II. Web Display (HTML/CSS/JavaScript)

```
<!DOCTYPE html>
<html>
<head>
<title>Sensor Data</title>
<style>
    body {
        background-image: url('background.jpg'); /* Setting background image */
        background-size: cover; /* Setting image size to cover the whole screen */
        background-position: center; /* Centering the image */
        background-repeat: no-repeat; /* Image should not repeat */
        font-family: Arial, sans-serif; /* Setting font type */
    }
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
    .center {
        display: flex;
        justify-content: center;
        align-items: center;
        height: 80vh;
    }
    table {
        border-collapse: collapse;
        width: 50%;
        background-color: rgba(255, 255, 255, 0.8); /* White background, slightly transparent */
    }
    th, td {
        border: 1px solid black;
        padding: 8px;
        text-align: center;
    }
    th {
        background-color: #f2f2f2;
    }
</style>
<script>
function autoRefresh() {
    setTimeout(function() {
        location.reload();
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
    }, 10000); // Reloads the page every 10,000 milliseconds (10 seconds)
}
<?php if ($refresh) { ?>
window.onload = function() {
    location.reload();
};
<?php } ?>
</script>
</head>
<body onload="autoRefresh()">
<h1>Széchenyi "Thermometer project"</h1>
<div class="center">
<table>
<tr>
    <th>ID</th>
    <th>Temperature (°C)</th>
    <th>Humidity (%)</th>
    <th>Measurement time</th>
</tr>
<?php
// Usage of GET method: Displaying data queried from the database in the table
while($row = mysqli_fetch_assoc($result)) {
    echo "<tr>";
    echo "<td>" . $row['id'] . "</td>";
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
    echo "<td>" . $row['temperature'] . "</td>";
    echo "<td>" . $row['humidity'] . "</td>";
    echo "<td>" . $row['datetime'] . "</td>";
    echo "</tr>";
}
?>
</table>
</div>
</body>
</html>
<?php
// Closing the connection to the database
mysqli_close($conn);
?>
```

In this code, the POST method is used for receiving data sent by the ESP32 microcontroller and uploading it to the database. The GET method is used for querying and displaying the data on the webpage.

III. Microcontroller Code: ESP32 (Wi-Fi Implementation)

```
#include <WiFi.h>      // Library required for WiFi connection
#include <HTTPClient.h> // Library required for handling HTTP requests
#include <DHT.h>      // Library belonging to the DHT sensor

#define DHTPIN 26     // Connection point of the DHT sensor (GPIO 26)
#define DHTTYPE DHT11 // Type of the DHT sensor (DHT11)
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
DHT dht11(DHTPIN, DHTTYPE); // Creating DHT sensor object

// The server URL where the data is sent
String URL = "https://lkcomputers.hu/esp32/";
// WiFi network SSID (network name) and password
const char* ssid = "Dualis-kepzes";
const char* password = "Tanulo123";
// Variables for storing temperature and humidity
int temperature = 0;
int humidity = 0;

void setup() {
  Serial.begin(115200); // Starting serial communication at 115200 baud speed
  dht11.begin(); // Initializing DHT sensor
  connectWiFi(); // WiFi connection
}

void loop() {
  if(WiFi.status() != WL_CONNECTED) {
    connectWiFi(); // WiFi reconnection if the connection is lost
  }

  Load_DHT11_Data(); // Reading data from the DHT sensor
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  // Formatting the data into an URL-encoded String
  String postData = "temperature=" + String(temperature) + "&humidity=" + String(humidity);


  HTTPClient http; // Creating HTTPClient object
  http.begin(URL); // Starting HTTP connection based on the URL
  http.addHeader("Content-Type", "application/x-www-form-urlencoded"); // Adding HTTP headers
  int httpCode = http.POST(postData); // Sending HTTP POST request with the String containing the data
  String payload = "";


  if(httpCode > 0) { // If HTTP response code is greater than 0, the request was successful
    if(httpCode == HTTP_CODE_OK) { // If HTTP response code is OK (200), the request was successfully processed
      payload = http.getString(); // Response message in String format
      Serial.println(payload); // Writing response message to the serial monitor
    } else {
      Serial.printf("[HTTP] GET... code: %d\n", httpCode); // Writing error code if the response is not OK
    }
  } else {
    Serial.printf("[HTTP] GET... failed, error: %s\n", http.errorToString(httpCode).c_str()); // Writing error code if the request failed
  }
  http.end();  // Closing HTTP connection
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  // Writing debug information to the serial monitor
  Serial.print("URL : "); Serial.println(URL);
  Serial.print("Data: "); Serial.println(postData);
  Serial.print("httpCode: "); Serial.println(httpCode);
  Serial.print("payload : "); Serial.println(payload);
  Serial.println("-------------------------------------------------");
  delay(5000); // Waiting for 5 seconds before the next data transmission
}

// Function for loading DHT11 sensor data
void Load_DHT11_Data() {
  temperature = dht11.readTemperature(); // Reading temperature in Celsius degrees
  humidity = dht11.readHumidity(); // Reading humidity in percentage

  // Check if data reading was successful
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!"); // Error notification if reading failed
    temperature = 0; // Setting temperature to default value
    humidity = 0; // Setting humidity to default value
  }

  // Writing data to the serial monitor
  Serial.printf("Temperature: %d °C\n", temperature);
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  Serial.printf("Humidity: %d %%\n", humidity);
}

// Function for handling WiFi connection
void connectWiFi() {
  WiFi.mode(WIFI_OFF); // Turning off WiFi
  delay(1000); // Waiting for 1 second
  WiFi.mode(WIFI_STA); // Turning on WiFi station mode (so that the ESP is not a visible hotspot)
  WiFi.begin(ssid, password); // Connecting to the specified SSID and password WiFi network
  Serial.println("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500); // Waiting for 0.5 seconds
    Serial.print("."); // Printing a dot to the serial monitor to indicate the connection process
  }

  // WiFi connection successful
  Serial.print("connected to : "); Serial.println(ssid);
  Serial.print("IP address: "); Serial.println(WiFi.localIP()); // Writing the obtained IP address to the serial monitor
}
```

IV. Implementation with Arduino UNO and ESP-01S

Project ID: 2023-1-HU01-KA210-VET-000156243

The project can also be implemented using **Arduino UNO and an ESP-01S module**. In this case, only the Arduino code needs to be adapted from the ESP32 code, and the wiring must be known. The website and database can be used without modification.

Hardware Wiring

1. **Connecting ESP-01S WiFi module to Arduino Uno:**
    - ESP-01S VCC -> 3.3V
    - ESP-01S GND -> GND
    - ESP-01S RX -> TX (D1)
    - ESP-01S TX -> RX (D0)
    - ESP-01S CH_PD -> 3.3V
2. **Connecting DHT11 sensor to Arduino Uno:**
    - DHT11 VCC -> 5V
    - DHT11 GND -> GND
    - DHT11 DATA -> D2

Arduino Uno Code (using SoftwareSerial and AT Commands)

```
#include <SoftwareSerial.h> // Library required for software serial communication
#include <DHT.h>            // Library belonging to the DHT sensor

#define DHTPIN 2            // Connection point of the DHT sensor (D2)
#define DHTTYPE DHT11       // Type of the DHT sensor (DHT11)
DHT dht11(DHTPIN, DHTTYPE); // Creating DHT sensor object
SoftwareSerial ESPserial(0, 1); // RX, TX - ESP-01S communication

const char* ssid = "Dualis-kepzes";
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
const char* password = "Tanulo123";

String URL = "https://lkcomputers.hu/esp32/";

// Variables for storing temperature and humidity

int temperature = 0;

int humidity = 0;


void setup() {

  Serial.begin(115200); // Starting serial communication at 115200 baud speed

  ESPserial.begin(115200); // Setting ESP-01S communication speed

  dht11.begin(); // Initializing DHT sensor

  connectWiFi(); // WiFi connection

}


void loop() {

  Load_DHT11_Data(); // Reading data from the DHT sensor

  String postData = "temperature=" + String(temperature) + "&humidity=" +
String(humidity);

  sendToServer(postData); // Sending data to the server

  delay(5000); // Waiting for 5 seconds before the next data transmission

}


// Function for loading DHT11 sensor data

void Load_DHT11_Data() {

  temperature = dht11.readTemperature(); // Reading temperature in Celsius degrees
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  humidity = dht11.readHumidity(); // Reading humidity in percentage


  // Check if data reading was successful
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!"); // Error notification if reading failed
    temperature = 0; // Setting temperature to default value
    humidity = 0; // Setting humidity to default value
  }


  // Writing data to the serial monitor
  Serial.print("Temperature: "); Serial.print(temperature); Serial.print(" °C\n");
  Serial.print("Humidity: "); Serial.print(humidity); Serial.println(" %\n");
}


// Function for handling WiFi connection
void connectWiFi() {
  ESPserial.println("AT"); // Sending AT command to the ESP-01S module
  delay(1000); // Waiting for 1 second
  ESPserial.println("AT+CWMODE=1"); // Setting Station Mode
  delay(1000); // Waiting for 1 second
  // Connecting to the specified SSID and password WiFi network
  ESPserial.print("AT+CWJAP=\"");
  ESPserial.print(ssid);
  ESPserial.print("\",\"");
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  ESPserial.print(password);
  ESPserial.println("\"");
  delay(10000); // Waiting 10 seconds for connection
}

// Function for sending data to the server
void sendToServer(String postData) {
  ESPserial.println("AT+CIPSTART=\"TCP\",\"lkcomputers.hu\",80"); // Starting TCP connection
  delay(2000); // Waiting for 2 seconds

  // Creating and sending HTTP request
  ESPserial.print("AT+CIPSEND=");
  ESPserial.println(postData.length() + 19); // Specifying the length of the data to be sent
  delay(2000); // Waiting for 2 seconds
  ESPserial.print("POST /esp32/ HTTP/1.1\r\n");
  ESPserial.print("Host: lkcomputers.hu\r\n");
  ESPserial.print("Content-Type: application/x-www-form-urlencoded\r\n");
  ESPserial.print("Content-Length: ");
  ESPserial.print(postData.length());
  ESPserial.print("\r\n\r\n");
  ESPserial.print(postData);
  delay(5000); // Waiting 5 seconds for the response
}
```

Project ID: 2023-1-HU01-KA210-VET-000156243

Performance Comparison (ESP32 vs. UNO + ESP-01S)

These modifications adapt to the limitations and capabilities of the Arduino Uno and the ESP-01S module. Now, the WiFi module can connect to the internet and send the data to the specified server.

The combination of the ESP-01S WiFi module and the Arduino Uno can generally be **slower** compared to the ESP32, as the ESP32 is a stronger microcontroller with more processing power and larger memory. However, the performance difference will usually not be significant.

Potential differences:

1. **Connection Time:** Setting up the connection between the Arduino Uno and the ESP-01S WiFi module and connecting to the internet may take more time.
2. **Data Transmission:** The data transmission speed may differ slightly, but this is usually not a significant difference for simple POST requests.
3. **Memory:** The Arduino Uno has less memory, which may limit the handling of more complex operations or larger amounts of data.
4. **Software Serial Communication:** Software serial communication between the ESP-01S and the Arduino Uno can sometimes be more sensitive to errors and delays, but this is generally manageable with appropriate programming.

Overall, if data needs to be sent at regular intervals, and the project does not require high processing power, the ESP-01S and Arduino Uno combination will work correctly.

V. Implementation with Arduino UNO (Wired Ethernet Connection)

The project implementation can be modified for **Arduino UNO with wired internet connection**. We modify the code to work with the Arduino UNO and the **Ethernet Shield W5100**. Since the Arduino UNO does not have built-in WiFi, we will connect to the internet via an Ethernet connection.

1. **Importing Libraries:** We will use the Ethernet.h and SPI.h libraries.
2. **Code Modification:** Using the Ethernet and EthernetClient libraries instead of WiFi connection.

```
#include <DHT.h>
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
#include <Ethernet.h>
#include <SPI.h>

// DHT sensor setup
#define DHTPIN 2 // DHT sensor connection pin
#define DHTTYPE DHT11 // DHT sensor type (DHT11)
DHT dht11(DHTPIN, DHTTYPE);

// Ethernet settings
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192, 168, 1, 177);
EthernetClient client;
const char* server = "example.com"; // Server address
const int port = 80; // Server port
float temperature;
float humidity;

void setup() {
  Serial.begin(9600); // Starting serial communication at 9600 baud speed
  dht11.begin(); // Initializing DHT sensor

  // Initializing Ethernet
  Serial.println("Initializing Ethernet...");
  if (Ethernet.begin(mac) == 0) {
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
    Serial.println("Failed to configure Ethernet using DHCP");
    // Try setting a static IP address
    Ethernet.begin(mac, ip);
  }
  delay(1000); // Waiting for Ethernet initialization
  Serial.println("Ethernet initialized");
}

void loop() {
  Load_DHT11_Data(); // Reading data from the DHT sensor
  String postData = "temperature=" + String(temperature) + "&humidity=" +
String(humidity);

  // Sending HTTP POST request
  if (client.connect(server, port)) {
    client.println("POST /post-data HTTP/1.1");
    client.println("Host: example.com");
    client.println("Content-Type: application/x-www-form-urlencoded");
    client.print("Content-Length: ");
    client.println(postData.length());
    client.println();
    client.println(postData);

    // Handling HTTP response
```

```
      String line; // Declaring the line variable
      while (client.connected()) {
        if (client.available()) {
          line = client.readStringUntil('\n');
          if (line == "\r") {
            break;
          }
        }
      }
      // Reading the response
      String response = client.readString();
      Serial.println("Response: ");
      Serial.println(response);
      client.stop();
    } else {
      Serial.println("Connection failed");
    }

    delay(5000); // Waiting for 5 seconds before the next data transmission
}

// Function for loading DHT11 sensor data
void Load_DHT11_Data() {
    temperature = dht11.readTemperature(); // Reading temperature in Celsius degrees
```

Project ID: 2023-1-HU01-KA210-VET-000156243


```
  humidity = dht11.readHumidity(); // Reading humidity in percentage


  // Check if data reading was successful
  if (isnan(temperature) || isnan(humidity)) {
    Serial.println("Failed to read from DHT sensor!"); // Error notification if reading failed
    temperature = 0; // Setting temperature to default value
    humidity = 0; // Setting humidity to default value
  }


  // Writing data to the serial monitor
  Serial.print("Temperature: "); Serial.print(temperature); Serial.println(" °C");
  Serial.print("Humidity: "); Serial.print(humidity); Serial.println(" %");
}
```

// This code connects to the internet using the Arduino UNO and the Ethernet Shield W5100, and sends the DHT11 sensor data to the specified server.