NEW TECHNOLOGIES IN THE IMPLEMENTATION OF AIR QUALITY MEASUREMENT SYSTEMS

Adis Rahmanović[1], Muzafer Saračević[2], Lejla Pezo[3]

1. Faculty of Technical Studies, University of Travnik, Travnik, Coal mine Banovići, Bosnia and Herzegovina
2. International University Novi Pazar, Serbia
3. Public Institution Secondary School Centre "Nedžad Ibrišimović" Ilijaš, Bosnia and Herzegovina

ABSTRACT

This article presents the implementation of an air quality measurement system based on an Arduino microcontroller. The goal of the system is to demonstrate the collection of data on temperature, air humidity and concentrations of PM2.5 and PM10 particles, which are then wirelessly sent to a remote server via a Wi-Fi module, and then stored in a MySQL database and displayed via a web application in the form of an interactive table with time stamps. This work contributes to the development of IoT devices for environmental monitoring, providing a low-cost and portable solution for monitoring air pollution.

Keywords: new technologies for air quality measurement systems, air quality, smart solution

INTRODUCTION

Air quality is becoming an increasingly important issue in urban environments around the world. Ecological parameters are key indicators of the state of the environment that reflect the quality of air, water and soil. Monitoring of these parameters plays a vital role in the preservation of natural resources, the protection of biological diversity, and ensuring the health and safety of people and ecosystems.

The article describes the development of a system for measuring air quality, mobile technology, as well as smart technologies. It is also necessary to ensure that the collected data are available in real time, and that they are easily accessible and understandable for analysis and interpretation. Existing air quality systems are expensive, difficult to implement and maintain, and therefore a major challenge is how to develop a simple, inexpensive and simultaneously reliable air quality monitoring system that can be implemented in different environments, including urban areas, industrial zones, as well as indoor rooms. Considering the progress of technology, the Arduino platform offers an affordable and adaptable solution for building such a system. The combination of Arduino with appropriate sensors enables the construction of a compact device that can measure various parameters of air quality. Wi-Fi network integration enables real-time data transmission, while the web interface enables viewing.

Maintaining optimal humidity levels can improve air humidity monitoring, and it is an important aspect of maintaining a healthy and pleasant indoor environment. Hypothesis: The implementation of advanced technologies can ensure the monitoring and improvement of ecological parameters.

NEW TECHNOLOGIES IN THE IMPLEMENTATION OF AIR QUALITY MEASUREMENT SYSTEMS

The introduction of advanced functionalities in air quality monitoring systems represents an important step towards the creation of integrated, efficient and responsible ICT solutions. Connecting hardware components with software analysis, supported by artificial intelligence and data security, contributes not only to better environmental monitoring, but also to the development of sustainable urban and rural communities. Such systems can become an important tool in the creation of public policies, infrastructure planning, but also in the everyday life of citizens who want to be informed about the quality of the air they breathe.

The system consists of several components: a microcontroller (Arduino), sensors, a Wi-Fi module for wireless data transmission, a database for storing results, and a web application for displaying measured values, a web server, AI tools for analysis, as well as included security submodules.

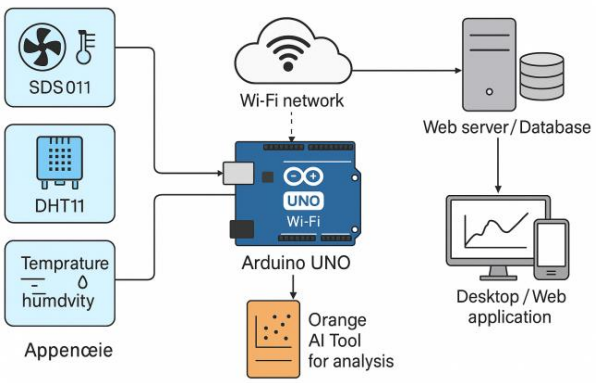

Figure 1. Schematic representation of the system

The picture shows a finished prototype of an air quality monitoring system based on the Arduino UNO platform with integrated Wi-Fi functionality. Using the SDS011 sensor, the concentration of PM2.5 and PM10 particles is measured, while the DHT11 sensor reads the temperature and air humidity. The components are connected by wire to the microcontroller, and the data is sent in real time to a remote server for processing and display via a Wi-Fi connection. The system enables automatic monitoring of the air quality without the need for manual reading.

The Wi-Fi network

The Wi-Fi network is a key element of the communication platform, enabling wireless communication between the Arduino platform and the database on the server. A Wi-Fi module (such as ESP8266 or ESP32) allows the Arduino to connect to the local network and send data to the server or directly to the application. The use of a Wi-Fi network reduces the need for physical connection and enables remote monitoring of air quality from any device connected to the Internet.

Wi-Fi network connection procedure:

1. Setting Wi-Fi data
SSID (network name) and password are entered in a special file called arduino_secrets.h. This approach helps protect sensitive information from public display in code.

2. Checking the Wi-Fi module
Before connecting, Arduino checks if the Wi-Fi module is properly connected and functional.

3. Successful connection
If the module is configured correctly, the Arduino connects to the network and retrieves the IP address of the device.

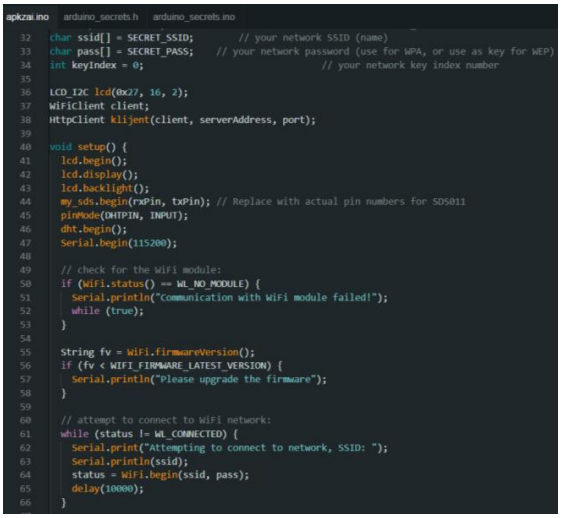

Figure 2. Code for connecting to a Wi-Fi network

The code checks whether the Wi-Fi module is properly connected and functional. Then the function WiFi.begin(ssid, pass) tries to connect the device to the given network. After a successful connection, the Arduino retrieves and displays its IP address.

Air quality measurement sensors

Sensors for measuring air quality represent a necessary element in systems for monitoring and analysing the state of the environment.

PM (particulate matter) particles represent suspended particles in the air that can have a significant impact on human health. To measure PM particles, sensors based on the principles of optical measurement are used.

• SDS011: This sensor is one of the most commonly used for measuring PM2.5 and PM10 particles. It uses laser light scattering to detect the presence and concentration of particles in the air. Its advantages include high precision, fast data processing and the ability to measure very low particle concentrations.

• PMS5003: Another popular sensor that offers similar features to the SDS011, but comes in a more compact format. It is ideal for integration into smaller devices.

• Sharp GP2Y1010AU0F: A simple and affordable sensor that measures dust and particles in the air. It is used in applications where basic air quality monitoring is required.

PM sensors: Use a laser light source and photodetectors. When particles pass through a laser beam, light is scattered. The scattering intensity allows the determination of the size and concentration of the particles.

To monitor temperature and air humidity, reliable and easy-to-use sensors are deployed.

• DHT11: One of the most commonly used sensors in projects like this. The DHT11 offers basic precision and reliability for measuring temperature (0–50°C) and humidity (20–90% RH). The advantage of this sensor is its accessibility and ease of use.

• DHT22: A more advanced version of the DHT11 sensor, with a larger range and better precision. It is used when more precise measurements are required.

• BME280: Multifunctional sensor that can measure air pressure in addition to temperature and humidity. Ideal for more complex air quality measurement systems.

Program code

The main code for the microcontroller (Arduino) is responsible for connecting to the Wi-Fi network, reading data from the temperature and humidity sensor (DHT11), and measuring the concentration of PM2.5 and PM10 particles using the SDS011 sensor. The read values are displayed on the LCD screen and periodically sent by HTTP GET request to the web server, where the data is stored.

The code uses the following libraries: WiFiNINA, DHT, SDS011, Arduino Http Client, LCD_I2C, Software Serial, as well as its own arduino_secrets.h file that contains information such as the SSID and password for the Wi-Fi network.

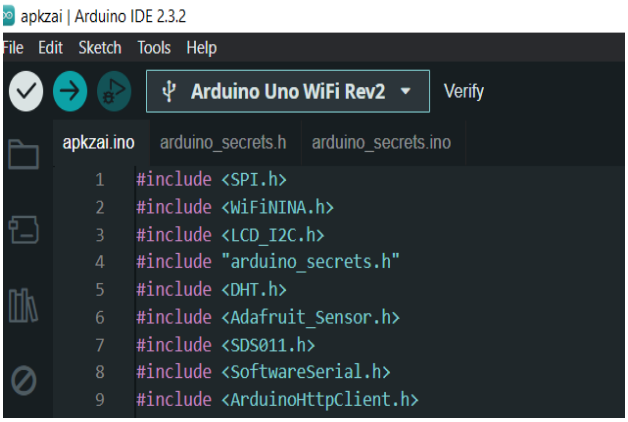

Figure 3. Libraries

```
#define DHTPIN 2
#define DHTTYPE DHT11

char serverAddress[] = "apkz.scilijas.com.ba";  // server address
int port = 80;
int rxPin = 10;  // Pin za RX senzor
int txPin = 11;  // Pin za TX senzor

DHT dht(DHTPIN, DHTTYPE);
float p10, p25;
int error;
SDS011 my_sds;
String postData;
String postVariable = "temp=";
int status = WL_IDLE_STATUS;

unsigned long lastTime = 0;
unsigned long timerDelay = 10000;
char server[] = "apkz.scilijas.com.ba";
String url = "/add_data.php?";


char ssid[] = SECRET_SSID;
char pass[] = SECRET_PASS;
int keyIndex = 0;

LCD_I2C lcd(0x27, 16, 2);
WiFiClient client;
HttpClient klijent(client, serverAddress, port);


SoftwareSerial mySerial(rxPin, txPin);

void setup() {
  lcd.begin();
  lcd.display();
  lcd.backlight();


  my_sds.begin(rxPin, txPin);

  pinMode(DHTPIN, INPUT);
  dht.begin();
  Serial.begin(115200);
  if (WiFi.status() == WL_NO_MODULE) {
    Serial.println("Communication with WiFi module failed!");
    while (true);
  }

  String fv = WiFi.firmwareVersion();
  if (fv < WIFI_FIRMWARE_LATEST_VERSION) {
    Serial.println("Please upgrade the firmware");
  }
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to network, SSID: ");
    Serial.println(ssid);
    status = WiFi.begin(ssid, pass);
    delay(10000);
  }

  Serial.println("You're connected to the network");
  printCurrentNet();
  printWifiData();
}
```

```
void loop() {
  if (millis() - lastTime > timerDelay) {
    lastTime = millis();

    float temperatureReading = dht.readTemperature();
    float humidityReading = dht.readHumidity();
    error = my_sds.read(&p25, &p10);

    Serial.println("Pozdrav, ovo je moj prvi ispis u Serial Monitoru!");

    if (isnan(temperatureReading) || isnan(humidityReading)) {
      Serial.println("Failed to read from DHT sensor!");
      return;
    }
    Serial.print("SDS011 greška: ");
    Serial.println(error);
    Serial.print("Temperatura: ");
    Serial.println(temperatureReading);
    Serial.print("Vlaznost: ");
    Serial.println(humidityReading);

    String data = "Senzor=R1D1SES";
    data += "&Temperatura=" + String(temperatureReading);
    data += "&Vlaznost=" + String(humidityReading);
    if (!error) {
      data += "&pm25=" + String(p25);
      data += "&pm10=" + String(p10);
    } else {
      data += "&pm25=0";
      data += "&pm10=0";
    }
      String fullUrl = url + data;
      Serial.println("Requesting URL: " + fullUrl);

      klijent.get(fullUrl);

      int statusCode = klijent.responseStatusCode();
      String response = klijent.responseBody();

      Serial.print("Status code: ");
      Serial.println(statusCode);
      Serial.print("Response: ");
      Serial.println(response);

      if (statusCode == 302) {
        Serial.println("The request was redirected.");
      }

      client.stop();
      delay(100000);
    }
}

void printWifiData() {
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  byte mac[6];
  WiFi.macAddress(mac);
  Serial.print("MAC address: ");
  printMacAddress(mac);
}


void printCurrentNet() {
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());

  byte bssid[6];
  WiFi.BSSID(bssid);
  Serial.print("BSSID: ");
  printMacAddress(bssid);

  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.println(rssi);

  byte encryption = WiFi.encryptionType();
  Serial.print("Encryption Type:");
  Serial.println(encryption, HEX);
  Serial.println();
}

void printMacAddress(byte mac[]) {
  for (int i = 5; i >= 0; i--) {
    if (mac[i] < 16) {
      Serial.print("0");
    }
    Serial.print(mac[i], HEX);
    if (i > 0) {
      Serial.print(":");
    }
  }
  Serial.println();
}
```

Figure 4. Parts of Arduino code

After defining all necessary libraries, all variables are defined:

• Pins to which sensors are connected (e.g., DHTPIN on pin 2, RX and TX pins for SDS011).

• Server address and path to which data is sent (/ add_data.php).

• Name and password of the Wi-Fi network, which are located in a special file arduino_secrets.h for security.

The setup () function initialises the LCD screen and turns on the backlight, starts communication with DHT11 and SDS011 sensors, and starts serial communication for printing data in the Serial Monitor. After that, we establish a connection with the Wi-Fi network. If the connection is not successful, the microcontroller tries again until it establishes a connection. After connecting to the network, basic network information is displayed: SSID, IP address, MAC address and signal strength (RSSI), which is useful for diagnostics.

The function loop () executes the following tasks in an infinite loop every 10 seconds:

• Reads the current temperature and humidity from the DHT11 sensor.

• Detects PM2.5 and PM10 values from the SDS011 sensor.

• Displays data in the Serial Monitor and LCD screen.
• Prepares an HTTP GET request that includes all measured values.

• Sends data to a remote server where it will be stored in a database.

In case there is an error when reading from the sensor, the system will register it and set the PM2.5 and PM10 values to 0 so that the database remains consistent.

The server's response is displayed in the Serial Monitor, including the HTTP status code and any return message.

The code also includes the following functions:

printWifiData() – displays the current IP address and MAC address of the device.

printCurrentNet() – displays information about the currently active network (SSID, BSSID, signal strength, and encryption type).

printMacAddress() – prints the MAC address of the device in readable format.

The implemented Arduino code is a key component of the air quality measurement system, enabling reliable data collection and transmission in real time. By integrating multiple sensors and using wireless

communication, a functional and efficient connection between the physical environment and the digital platform for storing and displaying data has been achieved. This approach enables continuous monitoring of environmental parameters without the need for manual intervention, thus facilitating the analysis and monitoring of trends in air pollution. The code is written in a modular and adaptable manner, which enables future system upgrades or the inclusion of additional sensors and functionality as needed.

Database implementation

In order to efficiently store and later process the measured values of temperature, humidity and PM2.5 and PM10 particles, a simple relational database was developed using the MySQL database management system. The database is placed on a remote web server, and access to the data is done via a PHP script that receives data via HTTP GET requests.

The standard mysqli_connect() function is used to connect the PHP script to the MySQL database. Below is an example of PHP code that establishes a connection to the database:

Figure 5. Connection to the database

After the connection is established, other scripts that communicate with the database can include this file using the include or require function, thus avoiding code duplication.

The database contains one main table for storing all relevant information from the sensors. The name of the table is, for example, measurements, and it contains the following fields:

Field name Data type Description
id INT (AUTO_INCREMENT) Primary key
time TIMESTAMP Date and time of measurement
sensor VARCHAR (20) Sensor identifier
temperature FLOAT Temperature in °C
humidity FLOAT Humidity in %
pm25 FLOAT PM2.5 concentration (μg/m^3)
pm10 FLOAT PM10 concentration (μg/m^3)

The table can be easily expanded with new columns if additional sensors or parameters are added in the future.

Data is entered into the database via the add_data.php script, which receives data from GET requests sent by Arduino. The script processes the data and writes it into the corresponding fields of the database.

Example of a call that Arduino generates:

http://apkz.scilijas.com.ba/add_data.php?Senzor=R1D1SES&Temperature=23.5&Vlaznost= 45.1&pm25=12.3&pm10=18.7

The PHP script uses the MySQL library to connect to the database and execute SQL queries. Before entering the data, a check is made to see whether all parameters have been received in order to avoid incomplete records.

The database is administered via the phpMyAdmin tool, which allows for visual viewing and data management. This allows for manual viewing of measurement results, filtering data by date or value, and basic analysis. A web interface has been implemented that displays database data to the end user in tabular form.

Implementation of the application

The web application developed as part of this project represents a simple solution for displaying air quality data collected using an Arduino-based sensor system. Although it is not visually and structurally complex, the application successfully fulfils its basic function - it enables the user to quickly and easily view the readings of temperature, humidity and concentration of floating particles (PM2.5 and PM10).

The application is built using a combination of front-end and back-end technologies. HTML and CSS were used to create the user interface, PHP for server-side logic and communication with the database, while JavaScript and additional libraries, such as Data Tables and Chart.js, were used to improve interactivity and data visualisation.

Figure 6. PHP code snippet

In addition to the table, users are also provided with a visual representation of temperature changes over time. Using the Chart.js library, temperature data is displayed as a line graph. The PHP script extracts the last ten entries from the database, and they are transferred to the graph

in real time using JavaScript. In this way, the user can easily observe trends and oscillations in the measured temperature values.

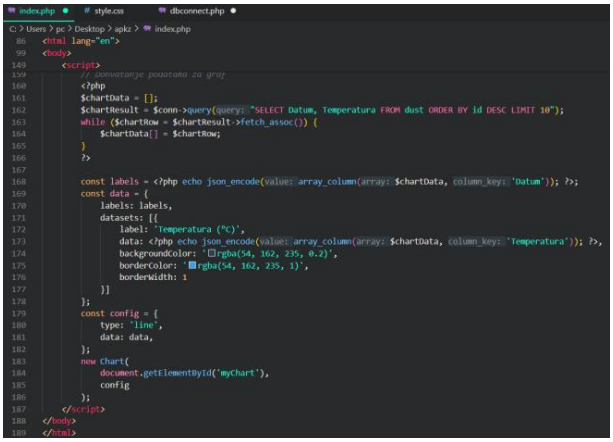

Figure 7. Graphical representation of temperature

The appearance of the application is minimalistic, but functional. Responsive design was achieved by using the Bootstrap framework, which means that the page automatically adapts to different screen sizes - from desktop monitors to mobile devices. Navigation and layout of elements are simple, with a focus on readability and clarity of display.

The application contains a basic navigation bar, a title and a short description of functionality, followed by a table with read values and a graph that visualises temperature changes. All components are carefully organised so that they are easily accessible and understandable to users.

CONCLUSION

Through the article, we presented the implementation of the project, confirming the hypothesis that the implementation of appropriate ICT technologies - specifically Arduino microcontrollers, sensors for measuring air quality and temperature/humidity, Wi-Fi modules, Web servers, software, and security software and databases with adequate programming - can enable efficient monitoring and analysis of environmental parameters.

The results of the implementation research show that this type of air quality monitoring system has a wide range of applications and upgrades, especially in urban and industrial environments where the monitoring of environmental parameters is of key importance for public health and environmental protection.

Thanks to the use of open platforms, such as the Arduino system, and the flexibility of wireless communication via the Wi-Fi module, the system can be easily upgraded and adapted to specific user requirements. Additional functionalities, such as expansion to more sensors (e.g., for measuring CO_2, noise or UV radiation), GPS location tracking or even automated alarms based on exceeding defined limit values, can be implemented relatively easily.

REFERENCES

1. Arduino. (2024). Arduino Reference – Libraries. https://www.arduino.cc/reference/en/libraries/
2. Arduino official website. (2025). https://www.arduino.cc/
3. Arduino Store. (2023). Arduino Sensor Kit – Base. https://store.arduino.cc/products/arduino-sensor-kit-base
4. Orange Data Mining. (2024). Orange Data Mining Documentation. https://orangedatamining.com/docs/
5. Blum, J. (2020). Exploring Arduino: Tools and techniques for engineering wizardry. John Wiley and Sons, Inc, Canada.
6. Chen, W. (2023). Database design and implementation. ATU Faculty Open Educational Resources, 2. https://orc.library.atu.edu/atu_oer/2
7. Davila, S. Postupci i metode za procjenu i upravljanje kvalitetom zraka. https://www.researchgate.net/profile/Silvije-Davila/publication/265070893_Postupci_i_metode_za_procjenu_i_upravljanje_kvalitetom_zraka/links/5507d2860cf27e990e084649/Postupci-i-metode-za-procjenu-i-upravljanje-kvalitetom-zraka.pdf
8. Dhawan, S. (2007). Analogy of promising wireless technologies on different frequencies: Bluetooth, Wi-Fi, and WiMAX. IEEE, Sydney, Australia. https://ieeexplore.ieee.org/abstract/document/4299663/authors
9. Karan, T., Aggarwal, A., & Masih, D. (2018). Wi-Fi Technology. Scribd. https://www.scribd.com/document/373130860/WiFi-Technology-Bss
10. Komlen Lalović, I., & Živić, I. (2022). ANDROID – JAVA mobile application for presenting fingerprint scanner results in YU Info IT conference 2022, Serbia. https://www.researchgate.net/publication/359685263_JAVA_MOBILE_APPLICATION_Android_FOR_PRESENTING_FINGERPRINT_SCANNER_RESULTS
11. Krstulović, D. (2022). Application for collection and analysis of air quality data. Final paper. University of Split, University Department of Professional Studies, Department of Electrical Engineering. https://urn.nsk.hr/urn:nbn:hr:228:643448
12. Sufyan bin Uzayr. (2023). Mastering SQL. CRC Press, Boca Raton. https://api.taylorfrancis.com/content/books/mono/download?identifierName=doi&identifierValue=10.1201/9781003358435&type=googlepdf
13. Vicić, P. (2023). Wireless SMART sensors principles and applications. Final paper. Osijek. Josip Juraj Strossmayer University of Osijek. Faculty of Electrical Engineering, Computer Science and Information Technology Osijek. https://urn.nsk.hr/urn:nbn:hr:200:811068

NOVE TEHNOLOGIJE U IMPLEMENTACIJI SISTEMA ZA MJERENJE KVALITETA ZRAKA

SAŽETAK
Ovaj rad predstavlja implementaciju sistema za mjerenje kvaliteta zraka zasnovanog na Arduino mikrokontroleru. Cilj sistema je prikazati prikupljanje podataka o temperaturi, vlažnosti zraka i koncentracijama PM2.5 i PM10 čestica, koji se zatim bežično šalju na udaljeni server putem Wi-Fi modula, a nakon toga pohranjuju u MySQL bazu podataka i prikazuju putem web aplikacije u obliku interaktivne tabele sa vremenskim oznakama. Ovaj rad doprinosi razvoju IoT uređaja za praćenje okoliša, pružajući jeftino i prenosivo rješenje za praćenje zagađenja zraka.

Ključne riječi: nove tehnologije za sisteme mjerenja kvaliteta zraka, kvalitet zraka, pametno rješenje

Correspondence to: Adis Rahmanović
Faculty of Technical Studies, University of Travnik, Travnik, Coal mine Banovići, Bosnia and Herzegovina
E-mail: cefes06@gmail.com