United International University

Course Code: CSE 4326
Section: B

Course Name
Microprocessors and Microcontrollers Laboratory

Experiment No: 02

Experiment Name
Wifi Communication and building IoT-based systems
Using XAMPP and ESP32

Submitted by (Group 4):

Muhammad Saadman Sakib Hossain - 0112230100
Jahidul Islam - 0112230654
Abrar Habib Adib - 0112230312
A. M. Zayed Abdullah - 011221127
Samira Bintey Haque - 011221128

Date of Performance: 29/07/2025
Date of Submission: 12/08/2025

1. Objective:

In this lab, we will interface a **DHT11 sensor** with an **ESP32** to monitor temperature and humidity. The collected data will be sent to a local **XAMPP** server, stored in a database, and displayed on a web page in real-time. We will also extend the setup to monitor gas data and control an LED. Furthermore, we will connect the ESP32 to the Arduino IoT Cloud to monitor gas concentration and remotely control the LED, learning how to integrate sensors, local servers, and cloud platforms to build an IoT-based system.

Final outcomes of this experiment will be:

1. Proficient ESP32 setup for gas sensor monitoring.
2. Secure IoT data transmission using XAMPP and Arduino Cloud.
3. Get a basic understanding of how databases work using PHP.
4. How to prepare a simple webpage to show sensor data.
5. Effective use of virtual "Thing" for data exchange in Arduino Cloud.
6. Competent programming for sensor data processing and LED control.
7. User-friendly IoT dashboard design for real-time data visualization.

Expected Outcomes:

1. Ability to set up and configure ESP32 for gas sensor monitoring.
2. Understanding of secure IoT data transmission using both XAMPP (local).
3. Skills to develop a simple webpage for displaying real-time sensor data.

2. Apparatus:

Hardware:

1. ESP-32 Development Board
2. DHT11 Sensor
3. MQ-2 Gas Sensor
4. LED
5. Breadboard
6. Jumper Wires
7. 10k Ohm Resistor

Software:

1. Arduino IDE
2. Proteus Design Suite (for simulation)
3. XAMPP
4. Arduino Cloud IoT

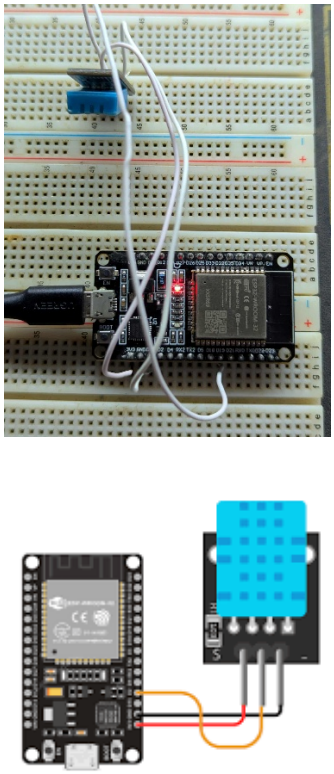
3. Theory:

1. The **ESP32** is a widely-used System on a Chip (SoC) that plays a key role in developing Internet of Things (IoT) devices. It boasts a range of features that make it a preferred choice for various IoT applications.

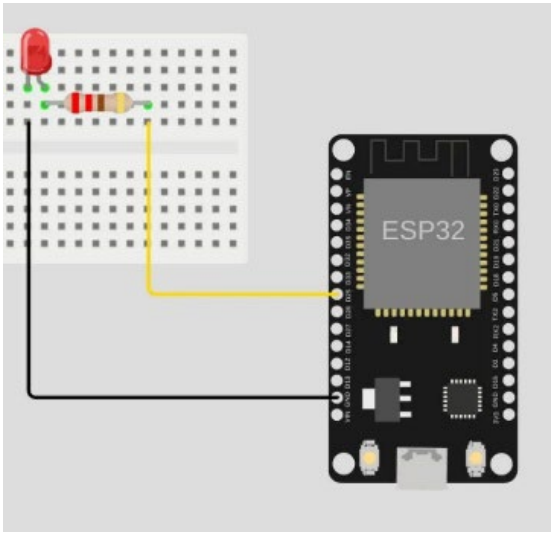
2. The **DHT11** is a basic, low-cost digital sensor used to measure temperature and humidity. It has a single-wire digital interface to communicate with microcontrollers like Arduino or ESP32. The sensor provides real-time environmental data and is commonly used in weather stations, home automation, and IoT projects due to its simplicity and ease of use.

3. The **MQ-2** gas sensor is a metal oxide semiconductor (MOS) sensor that detects combustible gases like LPG, smoke, CO, and methane. It changes its resistance when exposed to gas, and this change is read as analog voltage using Arduino's analog pins.

4. A **LED** (Light Emitting Diode) is a semiconductor device that emits light when an electric current passes through it. LEDs are energy-efficient, long-lasting, and commonly used as indicators, displays, or for illumination in electronic circuits and IoT projects.



DHT11 and ESP32 Wire Connection



LED with ESP32

4. Circuit Description:

The system includes **two (02)** main components:

1. DHT11 Sensor Connection to ESP32

- VCC → 3V3
- GND → GND
- SIG → D19 (Digital signal to ESP32)

2. LED Connection to ESP32

- Positive leg → D13
- Other leg → GND

The components are powered by ESP32 and mounted on a breadboard. This creates a compact, reliable, and modular circuit.

5. Procedure:

1. Connect the DHT11 sensor to the ESP32 as per the circuit description:
   - VCC to 3.3V, GND to GND, digital output to D19.

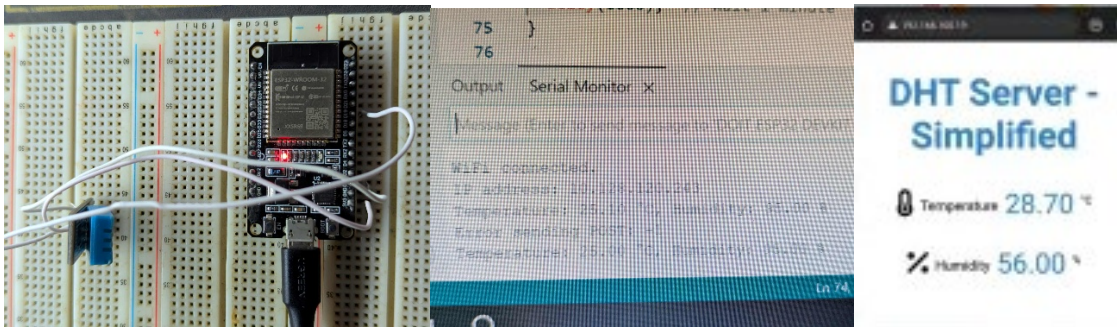
2. Connect LED to ESP32 pins:
   - Positive to D13, GND to GND

3. Connect the XAMPP to ESP32.

4. Write and upload the Arduino code that
   - Reads temperature and humidity data from the DHT11 sensor.
   - Sends the data to the XAMPP server to store in a database.
   - Controls the LED based on specific conditions (e.g., temperature or humidity thresholds).
   - Updates the local webpage or Arduino Cloud dashboard with real-time sensor data.

5. Verify the setup:
   - Check the OLED or webpage to ensure temperature and humidity values are displayed correctly.
   - Test LED control by simulating conditions in the code.

6. Final Setup:



Final setup & Result

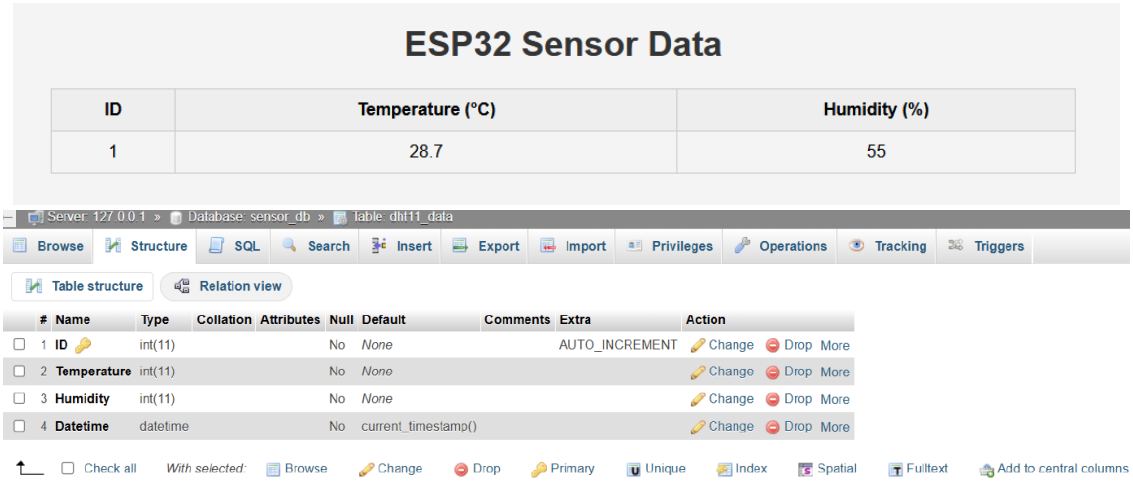
7. Program Code:

🔗 **GitHub Repository:** [Experiment No. 02]

8. Result & Observations:

The **DHT11** sensor successfully measured temperature and humidity, and the data was displayed on the **XAMPP** webpage in real time. The LED responded correctly to programmed conditions, and the data was stored in the XAMPP database for monitoring and retrieval.

- The DHT11 sensor successfully measured **temperature and humidity** and transmitted the data to the ESP32.
- Real-time sensor data was displayed correctly on the **local XAMPP webpage**.
- The **LED** responded accurately to the programmed conditions, turning ON or OFF based on sensor readings.
- Data storage in the **XAMPP database** worked as expected, allowing retrieval and visualization.
- It reflected real-time data, enabling remote monitoring and control.



9. Discussion:

This experiment successfully demonstrated how to establish Wi-Fi communication with an ESP32 and build an IoT-based system using XAMPP as a local server. The system enabled real-time data transfer between the ESP32 and the server, where sensor readings were stored and visualized through a web interface. This reflects the core concept of IoT applications, where smart devices collect, transmit, and present data for monitoring and decision-making.

Some key takeaways from this experiment include:

1. Understanding how to connect the ESP32 to a Wi-Fi network and establish client-server communication.
2. Practical use of XAMPP to host a local MySQL database and PHP scripts for data storage and retrieval.
3. Designing a simple web dashboard to display real-time sensor data stored in the database.
4. Learning how IoT devices interact with cloud/local servers to enable remote monitoring and control.

During the experiment, minor challenges included:

1. Configuring the correct Wi-Fi credentials and ensuring stable connectivity between ESP32 and the router.
2. Setting up XAMPP properly, including Apache and MySQL services, and avoiding port conflicts.
3. Managing the correct PHP-ESP32 interaction for GET/POST requests to ensure data was inserted into the database without errors.
4. Handling response delays due to network fluctuations and optimizing request intervals for efficient communication.

Despite these challenges, the final implementation worked as expected. The ESP32 successfully transmitted data to the server, and the stored values could be accessed and displayed via the web interface. This experiment reinforced essential IoT skills, including Wi-Fi communication, client-server interaction, database integration, and real-time data visualization, forming the foundation for building scalable IoT-based systems.

10. Conclusion:

The experiment was successful in achieving the following:

- ESP32 successfully connected to Wi-Fi for wireless communication.
- Sensor data was transmitted and stored in a XAMPP-based server.
- Data visualization on a web interface demonstrated real-time monitoring.
- Challenges included Wi-Fi stability, server setup, and database handling.
- The experiment built a strong foundation for IoT system development.

11. References:

- ESP32 Documentation: ESP32
- Adafruit DHT11 Sensor Library: Library of Adafruit_DHT11
- GitHub Code Repository: **GitHub (Micro Lab)**