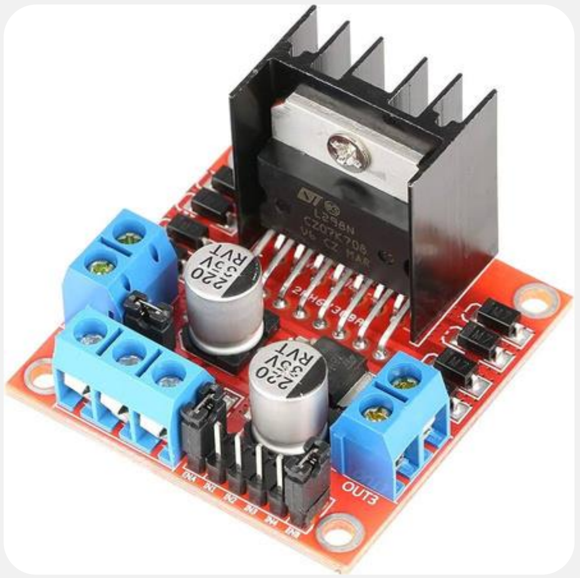
UNIVERSITÀ
DEGLI STUDI
DI SALERNO

A.A. 2023/2024

DRONE PROTOTYPE

TEAM:
ALESSANDRO RICCHETTI
ALESSANDRO MACARO


HARDWARE

- **Arduino** is essential for fundamental control tasks of the drone like stabilization, motor control, and handling flight sensors

- The **ESP32-CAM** module facilitates real-time video transmission over Wi-Fi, enabling live visual feedback by integrating a camera into the drone

- **DC motors** are employed for various drone movements such as propulsion and balance control

- The **DC Motor Controller L298N** serves as an H-bridge, enabling control over motor speed and direction based on signals from Arduino

SOFTWARE

ARDUINO IDE

NODE-RED

PYTHON

ARDUINO IDE

The entire control system was developed in Arduino IDE.
In the main loop of the program, the following operations are performed:

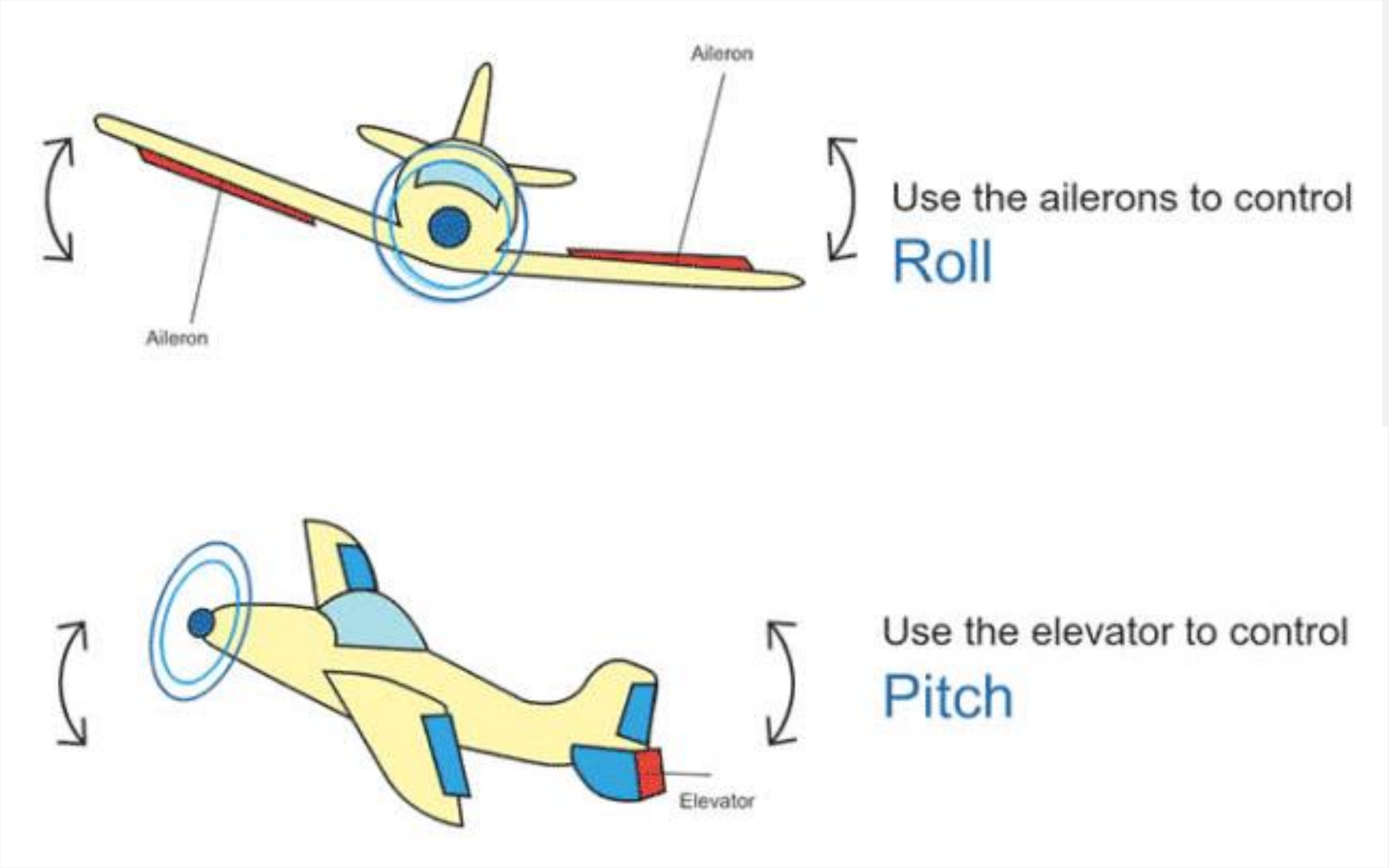
- The drone's motors are activated to enable takeoff
- Data from the inertial sensors (accelerometer and gyroscope) are read out
- Control signals for the drone's pitch and roll are calculated using the PID controllers
- A data packet containing the sensor readings and control signals is sent to the client connected via Wi-Fi

```
if (IMU.gyroscopeAvailable() && IMU.accelerationAvailable() && client) {

    IMU.readGyroscope(x_gyro, y_gyro, z_gyro);
    IMU.readAcceleration(x_acc, y_acc, z_acc);
    filter.updateIMU(x_gyro, y_gyro, z_gyro, x_acc, y_acc, z_acc);
      t = (IMU.readTemperature(t)/512.0)+23;

    String rollInputString = (dtostrf(filter.getRoll(), 4, 0, buffer));
    String pitchInputString = (dtostrf(filter.getPitch(), 4, 0, buffer));

    rollInput = atof(rollInputString.c_str());
    pitchInput = atof(pitchInputString.c_str());

    pitchPID.Compute();
    rollPID.Compute();
    int actSpeed[4];
    stabilise (targetSpeed, actSpeed, rollOutput, pitchOutput);

    // Invio dei valori
    wifiServer.print(String(x_acc)+";"+String(y_acc)+";"+String(z_acc)+";"
                     +String(x_gyro)+";"+String(y_gyro)+";"+String(z_gyro)+";"+String(t)+";"
                     +String(actSpeed[0])+";"+String(actSpeed[1])+";"
                     +String(actSpeed[2])+";"+String(actSpeed[3]) + ";"
                     +String(rollInput) + ";" + String(pitchInput) + "a");

    }
```

STABILISE

The `**stabilise**` function ensures drone stability by adjusting motor speeds according to pitch and roll angles from inertial sensors.
It computes motor speeds based on these angles, representing drone tilt. Differences between measured and desired angles determine necessary motor speed adjustments for tilt correction.
Upon calculating new motor speeds, the function directly updates PWM values of motor control pins (EN).

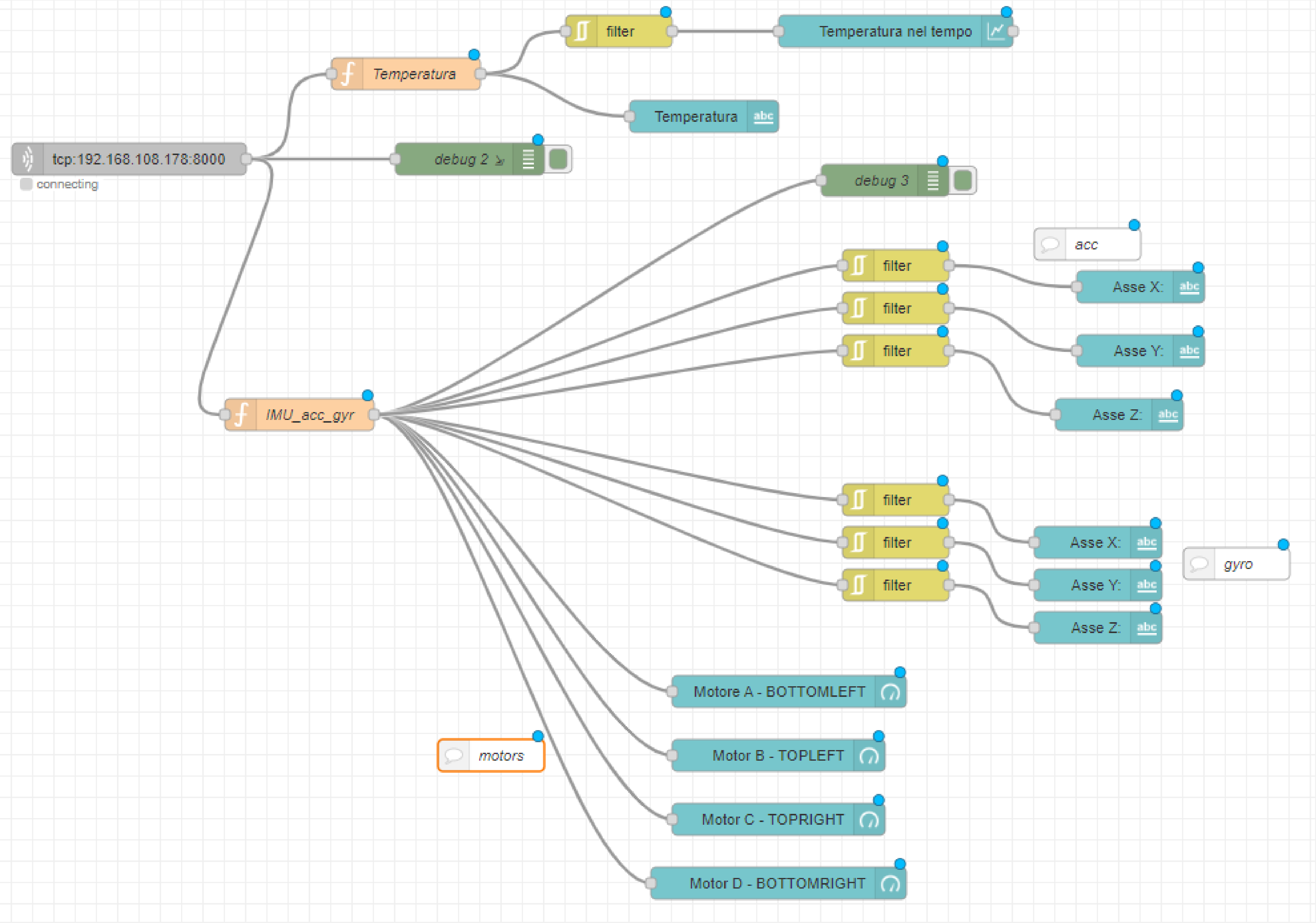
```
void stabilise (int* currSpeed, int* actSpeed, float rollDiff, float pitchDiff) {
  actSpeed[0] = (int) currSpeed[0] - (rollDiff / 2) - (pitchDiff / 2);
  actSpeed[1] = (int) currSpeed[1] - (rollDiff / 2) + (pitchDiff / 2);
  actSpeed[2] = (int) currSpeed[2] + (rollDiff / 2) - (pitchDiff / 2);
  actSpeed[3] = (int) currSpeed[3] + (rollDiff / 2) + (pitchDiff / 2);

  analogWrite(enA, actSpeed[0]);
  analogWrite(enB, actSpeed[1]);
  analogWrite(enC, actSpeed[2]);
  analogWrite(enD, actSpeed[3]);

  for (int i = 0; i < 4; i ++) {
    if (actSpeed[i] < 0)
      actSpeed[i] = 0;
  }
}
```



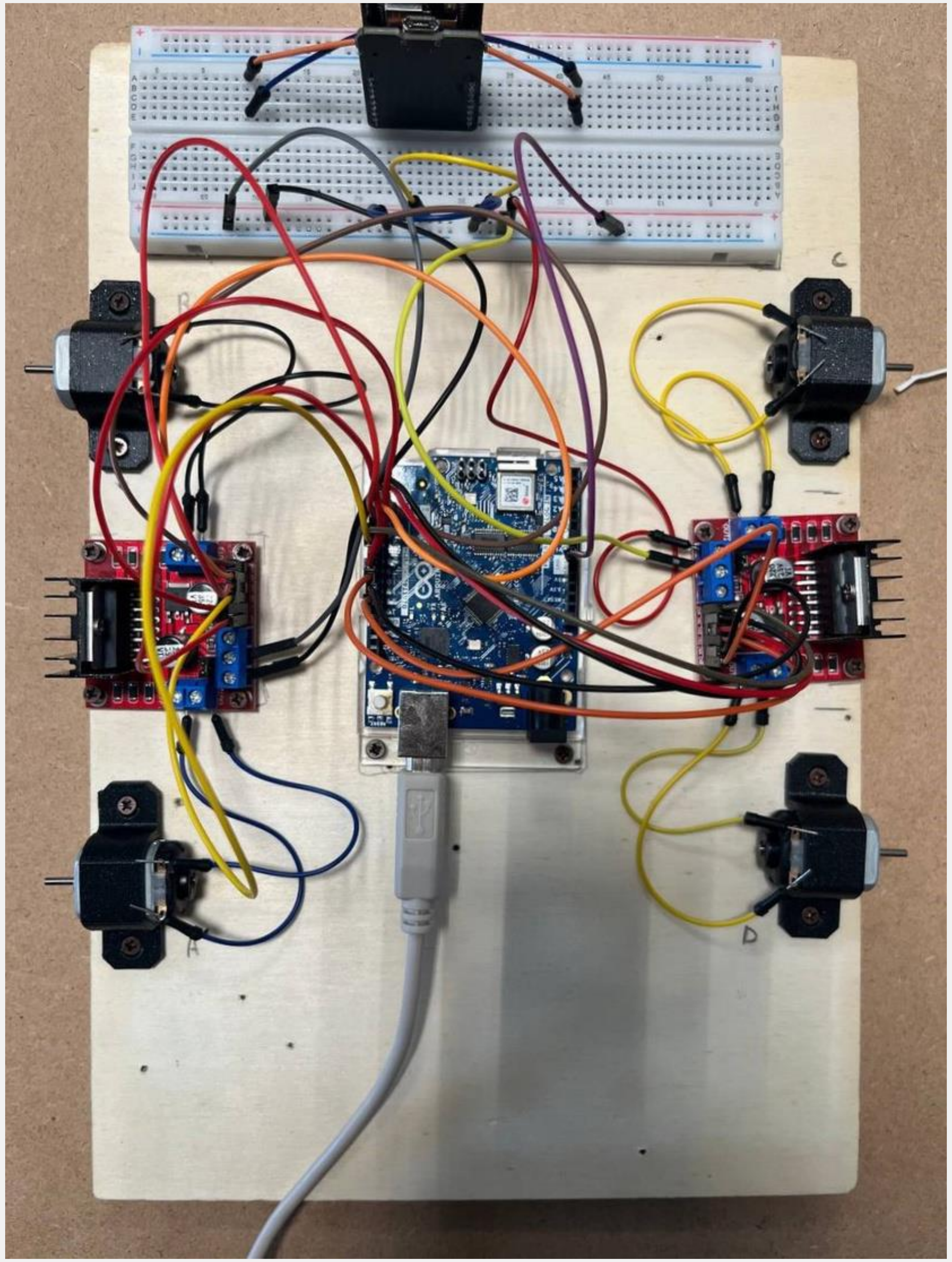
NODE-RED

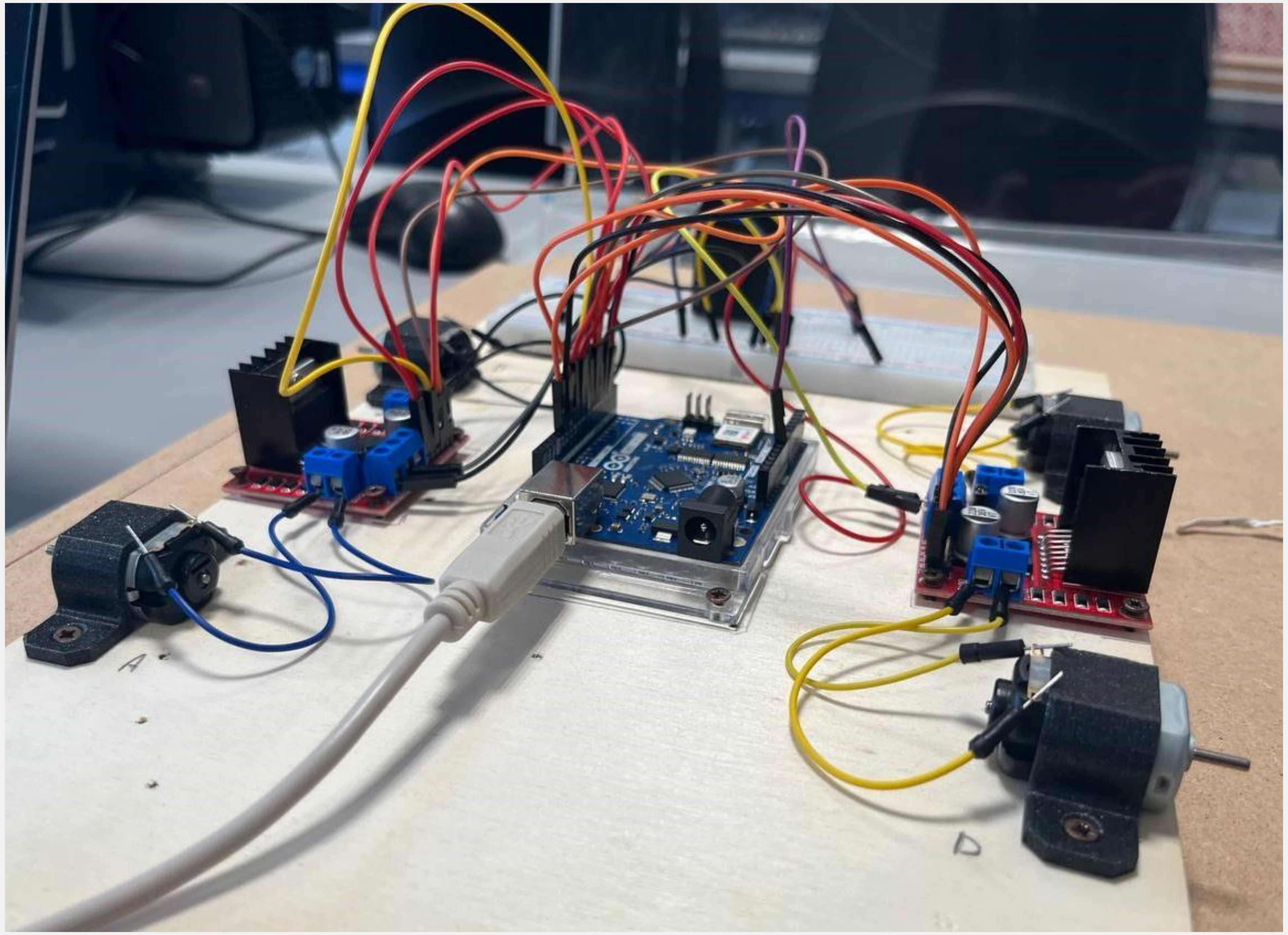
Node-RED is one of the most well-known flow-based programming tools for the Internet of Things. It was created with the goal of giving everyone the ability to connect different devices together in order to be able to build highly integrated and complex systems in a completely simple and intuitive way.

In our context it was used to graphically represent (via **dashboard**) all the values being received from Arduino.
More specifically, the values of the Accelerometer, Gyroscope, temperature sensor, and the powers of the four motors.



IMPLEMENTATION





THANKS FOR YOUR ATTENTION!