Martyna Filipiak,

Justyna Dołęga,

Krzysztof Hubert Kołodziejski

**Sprawozdanie z zajęć Internet Rzeczy**

Poniżej znajduje się szczegółowy opis trzech kluczowych etapów projektu związanego z konfiguracją i uruchomieniem Raspberry Pi, programowaniem modułu ESP01, a także odczytywaniem danych z czujnika temperatury. Wszystkie te etapy były realizowane   
w kontekście laboratorium technicznego i stanowiły unikalne wyzwania, które wymagały od nas zarówno teoretycznej wiedzy, jak i praktycznych umiejętności.

Pierwsza część opisuje proces uruchomienia i konfiguracji Raspberry Pi, poczynając od podłączenia urządzenia do prądu, aż do instalacji brokera MQTT - mosquitto. W tym etapie musieliśmy poradzić sobie z kilkoma trudnościami, w tym z problematycznym umiejscowieniem gniazda zasilającego.

Druga część skupia się na pracy z programatorem i diodą. Opisuje, jak otrzymaliśmy programator wraz z modułem ESP01 i jak musieliśmy zaprogramować moduł, aby dioda na nim zaczęła migać. Ten etap również wiązał się z kilkoma wyzwaniami, zwłaszcza   
w kontekście poprawnego podłączenia urządzeń.

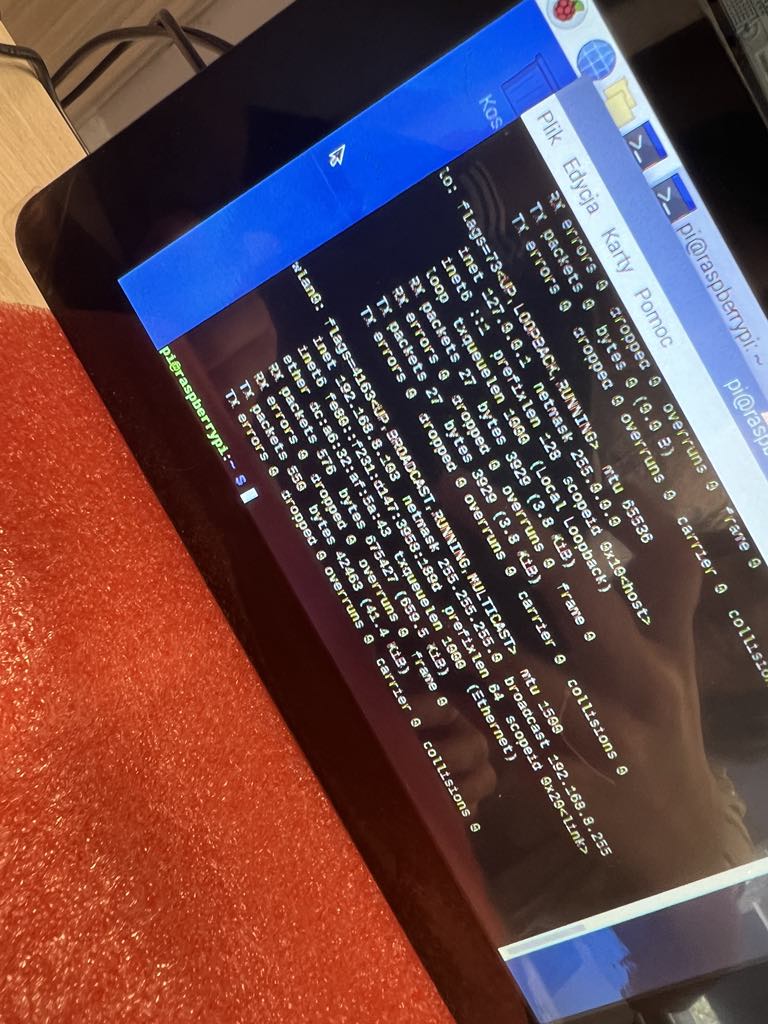
Trzecia i ostatnia część dotyczy odczytywania temperatury. Tutaj opisujemy, jak otrzymaliśmy czujnik, który pozwalał na odczyt temperatury z otoczenia, jak musieliśmy zaprogramować nasz ESP01, aby łączył się z siecią wifi, oraz jak musieliśmy połączyć kablami czujnik temperatury, programator i moduł ESP01.

Celem tego dokumentu jest nie tylko zarejestrowanie naszych doświadczeń   
i rozwiązań, ale także dostarczenie przyszłym użytkownikom Raspberry Pi cennych wskazówek i wskazówek, które mogą pomóc w ich własnych projektach.

**Uruchomienie i konfiguracja Raspberry Pi**

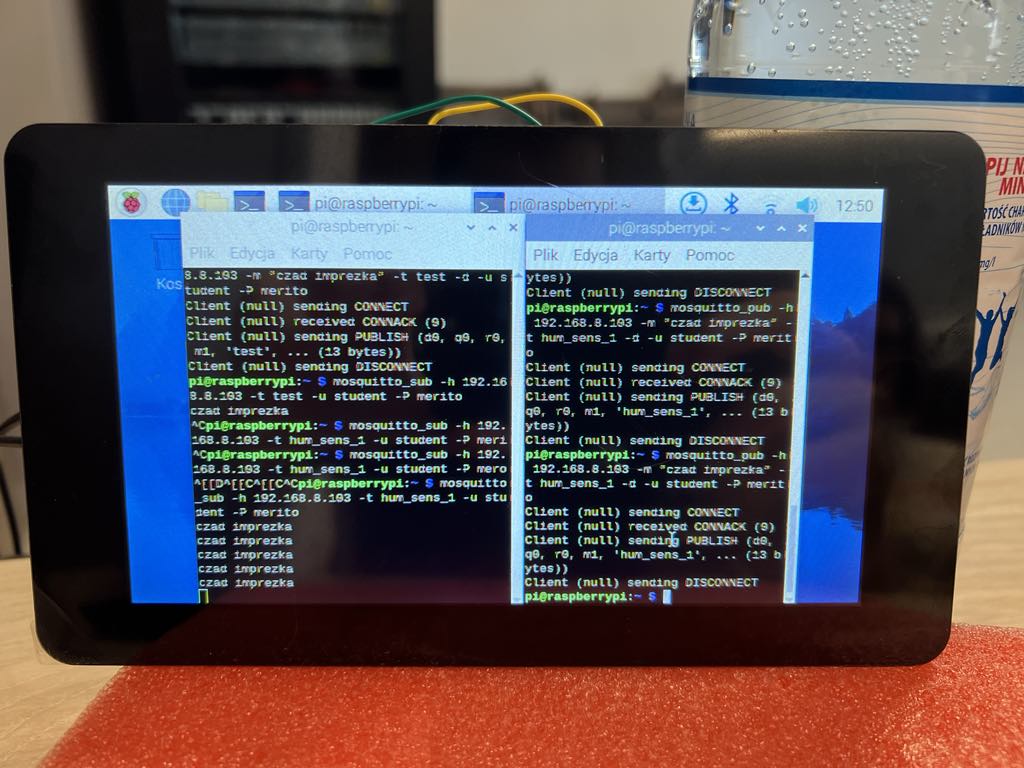
Podczas naszych prac nad Raspberry Pi, pierwszym krokiem, który podjęliśmy, było podłączenie urządzenia do źródła zasilania. W tym procesie napotkaliśmy pewne wyzwania, które wynikały z umiejscowienia gniazda zasilającego na spodzie urządzenia. Takie decyzje projektowe są często podyktowane ograniczeniami przestrzennymi i funkcjonalnymi urządzenia. Rozwiązaliśmy ten problem, konstruując specjalną podpórkę. Ta podpórka pozwoliła na swobodne zwisanie kabla pod ekranem urządzenia, eliminując tym samym kwestię niedogodnego umieszczenia gniazda zasilania.

Następnie, aby przystąpić do korzystania z Raspberry Pi, musieliśmy zainstalować system operacyjny. W tym celu do urządzenia włożyliśmy kartę microSD, na której uprzednio umieściliśmy obraz iso z dystrybucją Linuxa. Wybór padł na dystrybucję specjalnie zaprojektowaną i zoptymalizowaną pod kątem urządzeń Raspberry Pi. Dzięki temu mogliśmy liczyć na maksymalną wydajność i stabilność naszego systemu, co jest kluczowe dla realizacji złożonych projektów na tym urządzeniu.



Rys. 1. Działająca konsola na urządzeniu.

Po pomyślnej instalacji systemu operacyjnego, przystąpiliśmy do konfiguracji dodatkowego oprogramowania. Pierwszym z nich był mosquitto - broker MQTT. Jest to kluczowy element w systemach Internetu Rzeczy, umożliwiający komunikację między wieloma urządzeniami. Po zainstalowaniu i skonfigurowaniu mosquitto, połączyliśmy się   
z siecią udostępnioną przez prowadzącego. Dzięki temu mogliśmy zacząć publikować   
i nasłuchiwać wiadomości na wybranym przez nas topicu. To jest kluczowe dla naszej pracy, ponieważ umożliwia nam to interakcję i komunikację z różnorodnymi urządzeniami w sieci, które mogą publikować lub subskrybować dane z różnych tematów.



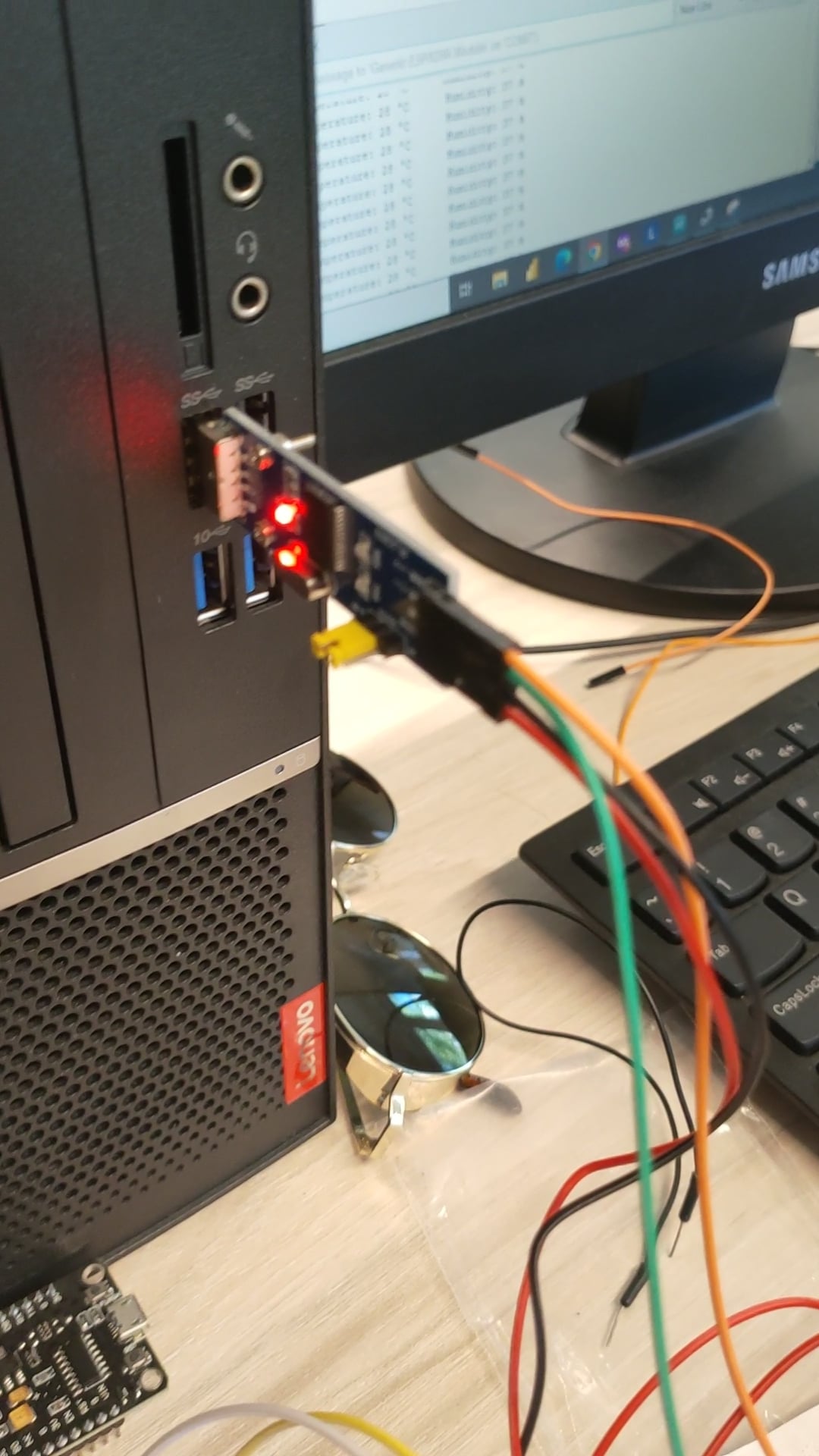
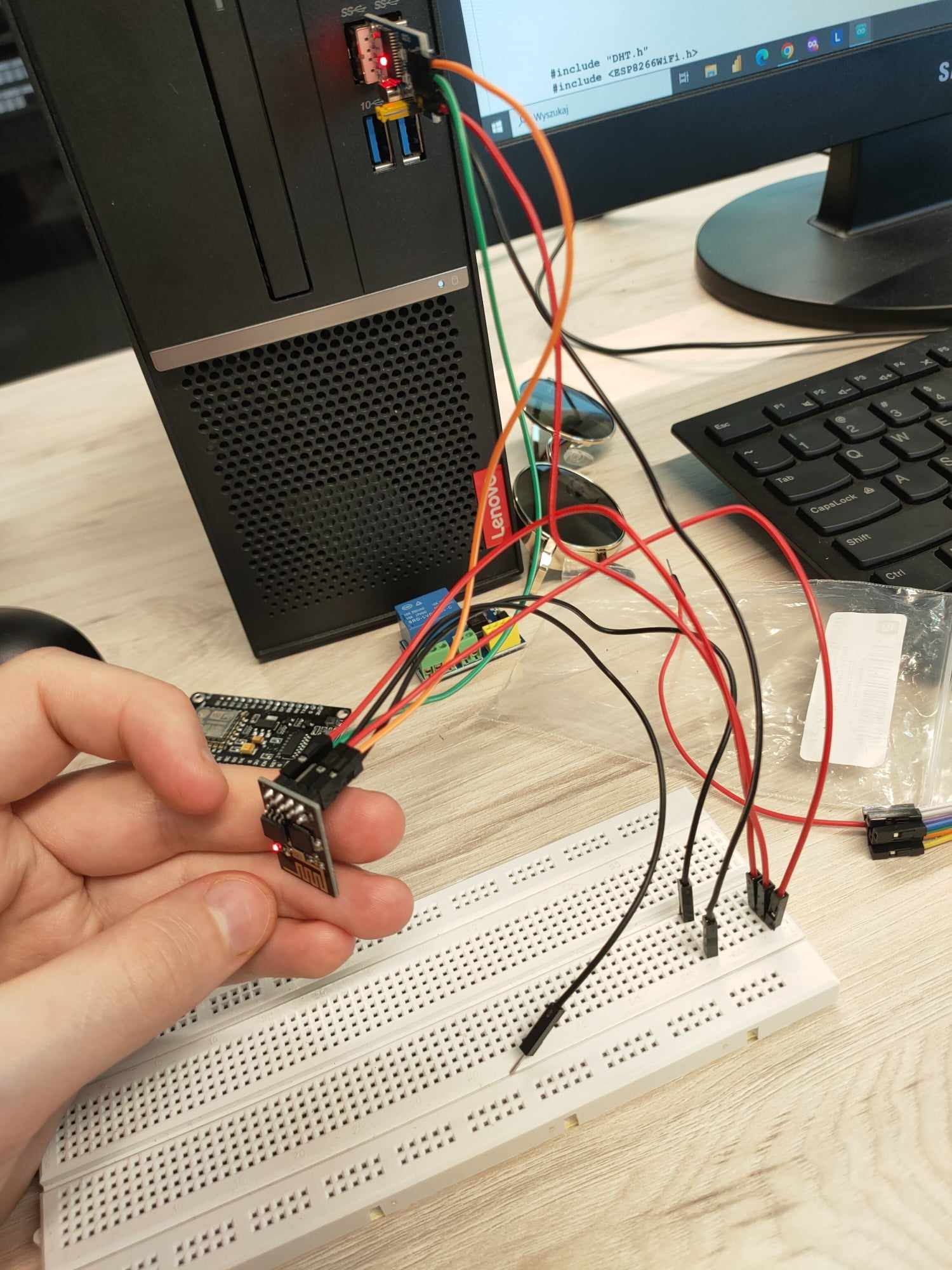
Rys. 2. Nasłuchiwane i wysyłane dane na topicu. Oba widoczne naraz w dwóch terminalach.

**Programator i dioda**

Po odpowiedniej konfiguracji Raspberry Pi, przystąpiliśmy do kolejnego etapu pracy, jakim było programowanie modułu ESP01 za pomocą dedykowanego programatora. Otrzymaliśmy zarówno programator, jak i moduł ESP01, a naszym celem było zaprogramowanie tego modułu w taki sposób, aby dioda LED zaczęła migać.

Programator został ustawiony na napięcie 3.3V, co jest kluczowe, ponieważ moduł ESP01 nie jest przystosowany do pracy na wyższych napięciach, takich jak 5V, które mogłyby go uszkodzić. Po ustawieniu odpowiedniego napięcia, programator został wpięty do komputera za pomocą portu USB. Następnie połączyliśmy programator z modułem ESP01 za pomocą odpowiednich kabli. W trakcie tego procesu musieliśmy zachować szczególną ostrożność, aby poprawnie podłączyć wszystkie przewody, ponieważ oznaczenia na samych urządzeniach nieco różniły się od tych przedstawionych na schematach, którymi się posługiwaliśmy.

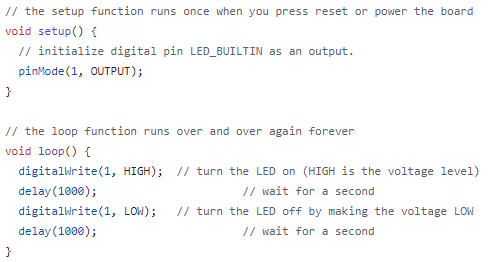
Dokładnie sprawdziliśmy poprawność połączeń między programatorem a modułem ESP01. Pomyłka w podłączeniu kabli mogłaby skutkować błędami w programowaniu lub nawet uszkodzeniem sprzętu. W związku z tym, upewniliśmy się, że każdy przewód jest podłączony do odpowiedniego pinu.



Rys. 4 i 5. Poprawnie podłączone urządzenia, gotowe do programowania.

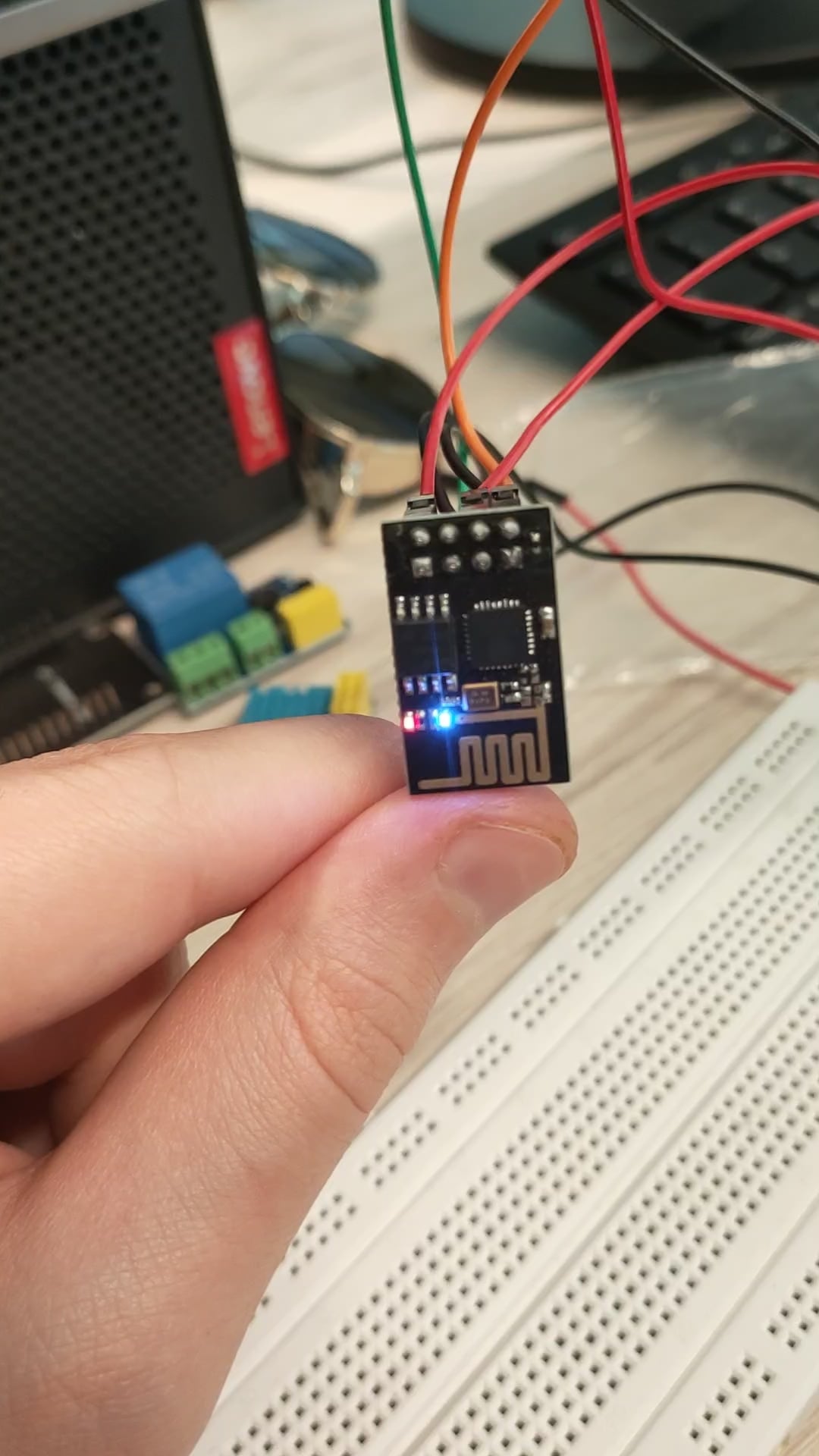
Po prawidłowym podłączeniu urządzeń przystąpiliśmy do wgrania szkicu na moduł ESP01. Użyliśmy do tego środowiska Arduino IDE, w którym przygotowaliśmy prosty szkic implementujący funkcję migania diody LED. Szkic ten został następnie skompilowany   
i wgrany na moduł ESP01.

Po zakończeniu procesu wgrywania szkicu, musieliśmy na chwilę odłączyć moduł ESP01 od zasilania. Zrobiliśmy to poprzez odłączenie kabla łączącego ESP01 z programatorem. Jest to konieczne, aby zresetować moduł i zapewnić prawidłowe działanie nowo wgranego programu. Po krótkiej chwili ponownie podłączyliśmy zasilanie.



Rys. 6. Skrypt z funkcją migania diody LED.

Po ponownym podłączeniu zasilania, dioda LED na płytce ESP01 zaczęła migać, co było oznaką poprawnego wgrania programu. Migająca dioda potwierdziła, że nasz proces konfiguracji i programowania przebiegł pomyślnie, a moduł ESP01 działa zgodnie   
z oczekiwaniami.



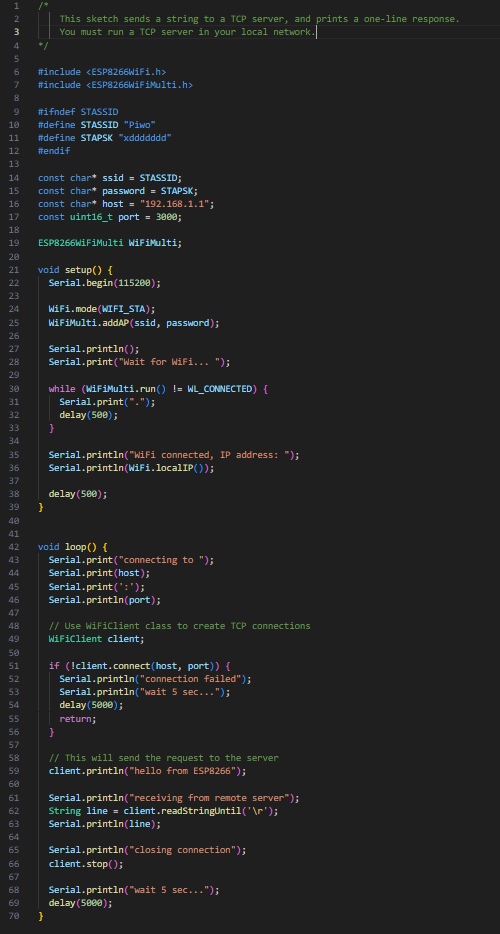
Rys. 7. Migająca dioda LED.

Cały proces programowania modułu ESP01, od ustawienia programatora, przez sprawdzenie połączeń, aż po wgranie szkicu i przetestowanie działania, wymagał dokładności i precyzji. Dzięki przestrzeganiu wszystkich kroków i weryfikacji poprawności połączeń, osiągnęliśmy zamierzony cel – dioda LED na module ESP01 zaczęła migać, co oznaczało, że moduł został prawidłowo zaprogramowany i działa zgodnie z naszymi oczekiwaniami.

**Odczytywanie temperatury**

Podczas kolejnych zajęć laboratoryjnych otrzymaliśmy nowy komponent - czujnik temperatury. Ten czujnik miał za zadanie odczytywać temperaturę z otoczenia, co stanowiło kluczowy element naszego projektu IoT.

Aby nasz moduł ESP01 mógł komunikować się z siecią Wi-Fi, musieliśmy go odpowiednio zaprogramować. Wykorzystaliśmy do tego celu programator, którego używaliśmy wcześniej, ustawiony na napięcie 3.3V. Podłączyliśmy programator do komputera za pomocą kabla USB. Następnie, za pomocą przewodów, połączyliśmy programator   
z modułem ESP01, upewniając się, że wszystkie połączenia są poprawne. W środowisku Arduino IDE stworzyliśmy nowy skrypt. W kodzie zawarliśmy informacje niezbędne do połączenia z siecią Wi-Fi, takie jak nazwa sieci (SSID) i hasło.

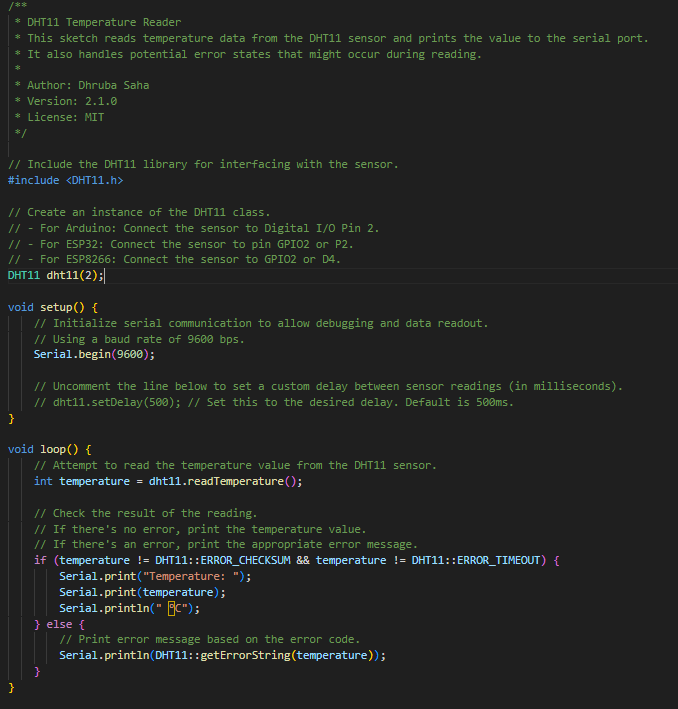


Rys. 8. Skrypt łączący moduł ESP01 z siecią Wi-Fi.

Po przygotowaniu kodu wgraliśmy szkic na moduł ESP01. Proces wgrywania przebiegł podobnie jak poprzednio – po jego zakończeniu na chwilę odłączyliśmy zasilanie modułu, aby nowy szkic mógł się uruchomić. Aby sprawdzić, czy moduł ESP01 poprawnie połączył się   
z siecią Wi-Fi, monitorowaliśmy liczbę urządzeń podłączonych do sieci za pomocą aplikacji na telefonie.

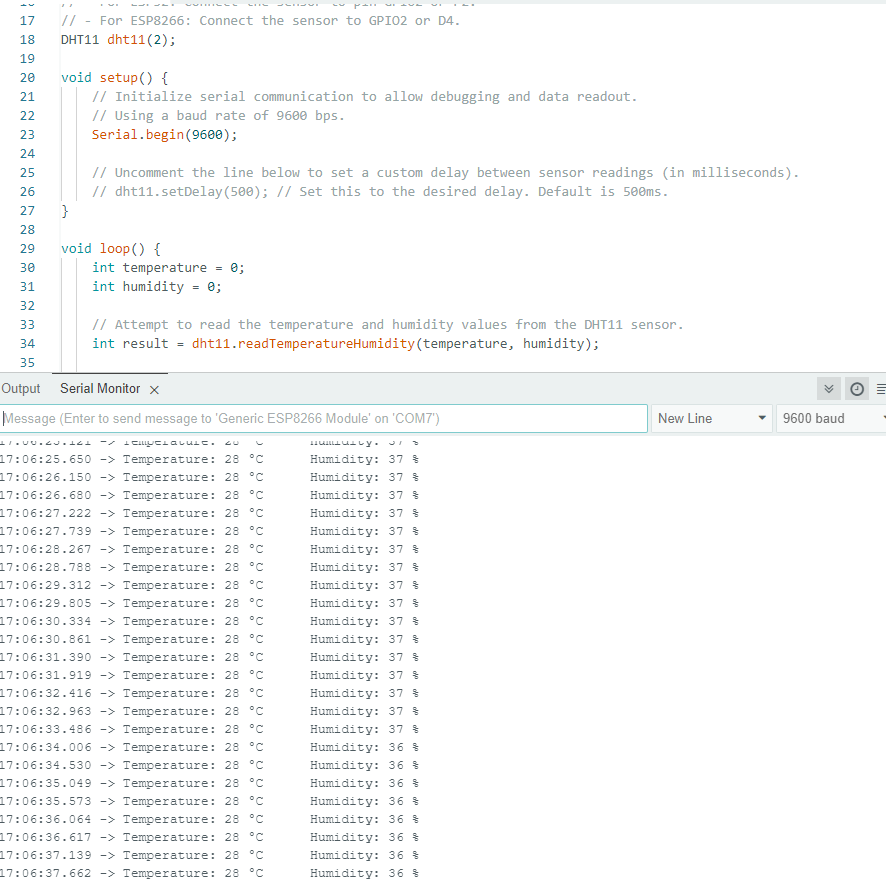
Po potwierdzeniu, że moduł ESP01 prawidłowo łączy się z siecią Wi-Fi, przystąpiliśmy do podłączenia czujnika temperatury. Był to czujnik cyfrowy DHT, który wymagał odpowiedniego połączenia z modułem ESP01 i programatorem. Użyliśmy odpowiednich przewodów, aby połączyć czujnik temperatury z modułem ESP01. Czujnik miał trzy piny: zasilania, masy oraz sygnałowy, które połączyliśmy z odpowiednimi pinami na ESP01. Dokładnie sprawdziliśmy, czy wszystkie przewody są prawidłowo podłączone, aby uniknąć błędów w odczycie danych.

Następnie zmodyfikowaliśmy nasz skrypt, aby moduł ESP01 mógł nie tylko łączyć się z siecią Wi-Fi, ale także odczytywać dane z czujnika temperatury i publikować je za pomocą protokołu MQTT. W pierwszej kolejności w ramach testów wgraliśmy kod, który wypisuje w konsoli temperaturę i wilgotność.



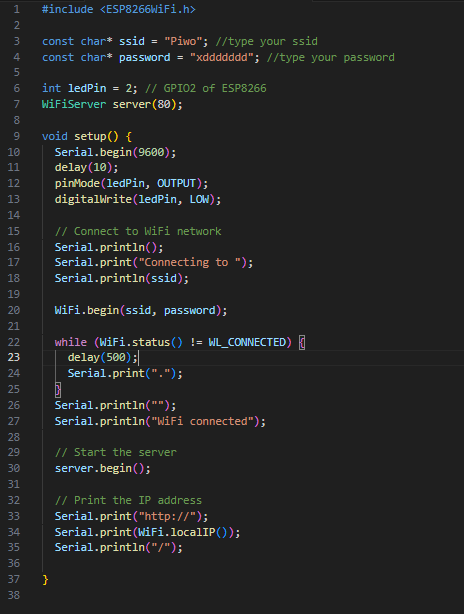
Rys 9. Kod wypisujący w konsoli temperaturę i wilgotność po pobraniu jej z czujnika.

Po wgraniu zmodyfikowanego szkicu, mogliśmy zaobserwować odczyty temperatury   
i wilgotności powietrza w środowisku Arduino IDE. Jednak podczas laboratorium nie mieliśmy dostępu do urządzeń Raspberry Pi, które mogłyby pełnić rolę subskrybentów   
w protokole MQTT, aby bezpośrednio odczytywać te dane.



Rys. 10. Logi zwracane przez kod z rys. 9.

Kolejnym krokiem było stworzenie serwera oraz połączenie skryptów z rysunku   
nr 8 i 9. Celem tego kroku było stworzenie połączenia z siecią Wi-Fi i możliwość publikacji informacji o wilgotności i temperaturze bezpośrednio z czujnika.



Rys. 11. Skrypt służący do stworzenia serwera.

Po poprawnym postawieniu serwera udało się nam wejść na stronę, która była podłączona do sieci urządzeń. Zabraklo nam jednak czasu na wyświetlenie informacji   
o temperaturze i wilgotności na stronie.

**Podsumowanie**

Projekt realizowany podczas zajęć Internet Rzeczy obejmował trzy kluczowe etapy związane z konfiguracją i uruchomieniem Raspberry Pi, programowaniem modułu ESP01 oraz odczytywaniem danych z czujnika temperatury.

Pierwszy etap skupiał się na uruchomieniu i konfiguracji Raspberry Pi. Wymagało to podłączenia urządzenia do źródła zasilania, co okazało się wyzwaniem ze względu na umiejscowienie gniazda zasilającego. Po rozwiązaniu problemu, przystąpiliśmy do instalacji systemu operacyjnego na urządzeniu za pomocą karty microSD. Wybraliśmy dystrybucję Linuxa specjalnie zaprojektowaną i zoptymalizowaną pod kątem urządzeń Raspberry Pi. Następnie przystąpiliśmy do konfiguracji dodatkowego oprogramowania - brokera MQTT, mosquitto, który jest kluczowym elementem w systemach Internetu Rzeczy.

Drugi etap dotyczył pracy z programatorem i diodą. Otrzymaliśmy programator wraz z modułem ESP01 i musieliśmy zaprogramować moduł, aby dioda na nim zaczęła migać. Programator został ustawiony na napięcie 3.3V i podłączony do komputera za pomocą portu USB. Następnie połączyliśmy programator z modułem ESP01 i wgraliśmy na niego szkic implementujący funkcję migania diody LED.

Trzeci i ostatni etap dotyczył odczytywania temperatury za pomocą czujnika. Musieliśmy zaprogramować nasz ESP01, aby łączył się z siecią Wi-Fi, oraz połączyć kablami czujnik temperatury, programator i moduł ESP01. Po potwierdzeniu, że moduł ESP01 prawidłowo łączy się z siecią Wi-Fi, przystąpiliśmy do podłączenia czujnika temperatury. Po prawidłowym podłączeniu urządzeń, wgraliśmy na moduł ESP01 szkic, który pozwalał na odczytywanie danych z czujnika temperatury i publikowanie ich za pomocą protokołu MQTT.

Pomimo różnych wyzwań napotkanych w trakcie realizacji tego projektu, nabyliśmy cenną wiedzę i umiejętności związane z konfiguracją i programowaniem Raspberry Pi   
i modułu ESP01, a także z odczytywaniem danych z czujnika temperatury. Wszystkie te doświadczenia stanowią solidną podstawę do przyszłych projektów związanych z Internetem Rzeczy.Projekt ten pokazał nam również, jak ważne jest dokładne planowanie i przemyślane podejście do problemów. Każdy z trzech etapów projektu wymagał od nas zarówno teoretycznej wiedzy, jak i praktycznych umiejętności, a także zdolności do rozwiązywania problemów i adaptacji do nieprzewidzianych sytuacji.