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.

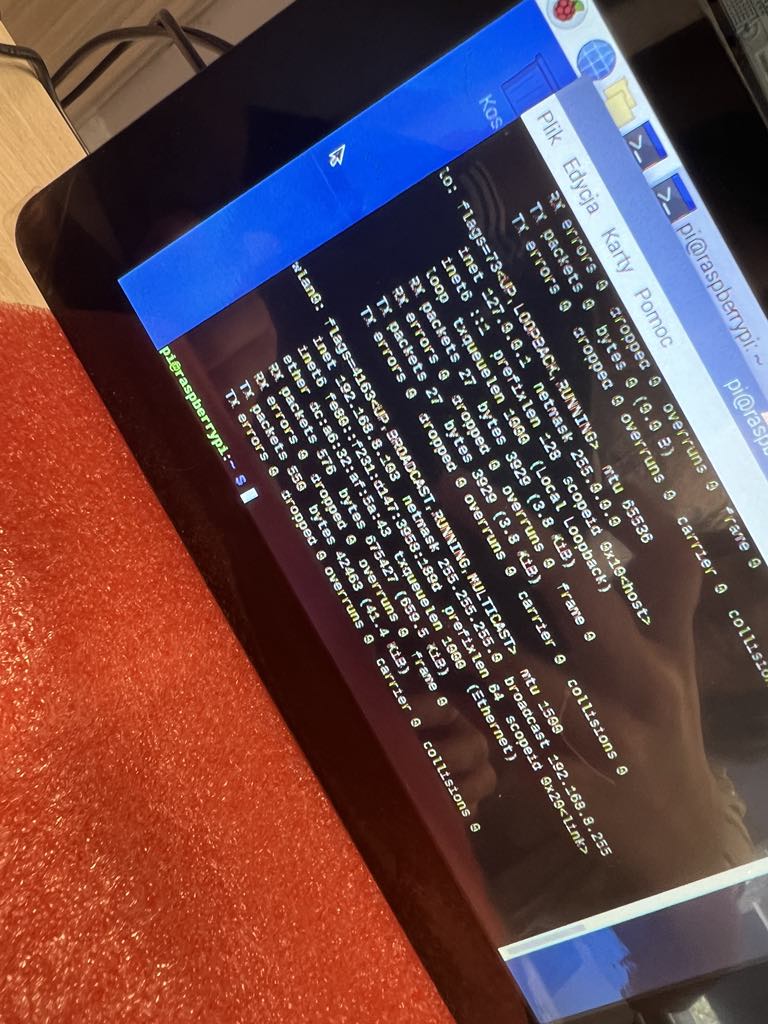
Trzecia i ostatnia część dotyczy odczytywania temperatury. Tutaj opisujemy, jak otrzymaliśmy czujnik, który pozwalał na odczyt temperatury z otoczenia, jak musieliśmy zaprogramować nasz ESP01, aby łączył się z siecią wifi, oraz jak musieliśmy połączyć kablami czujnik temperatury, programator i moduł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

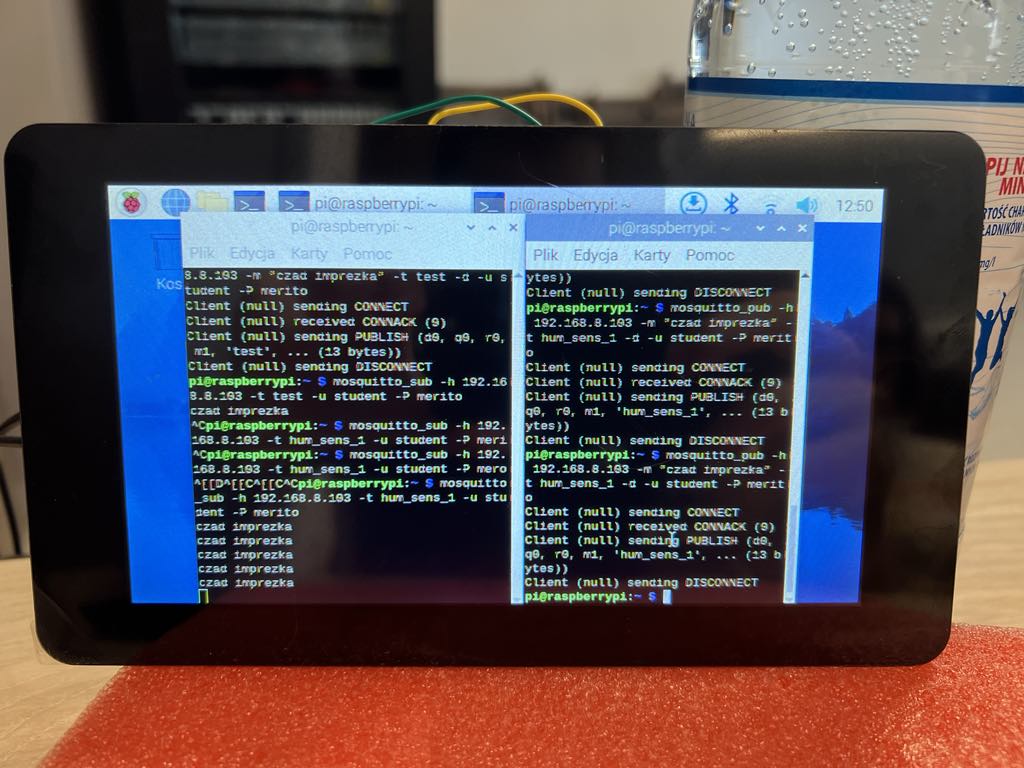
Podczas naszych prac nad Raspberry Pi, pierwszym krokiem, który podjęliśmy, było podłączenie urządzenia do źródła zasilania. W tym procesie napotkaliśmy pewne wyzwania, które wynikały z umiejscowienia gniazda zasilającego na spodzie urządzenia. Takie decyzje projektowe są często podyktowane ograniczeniami 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.



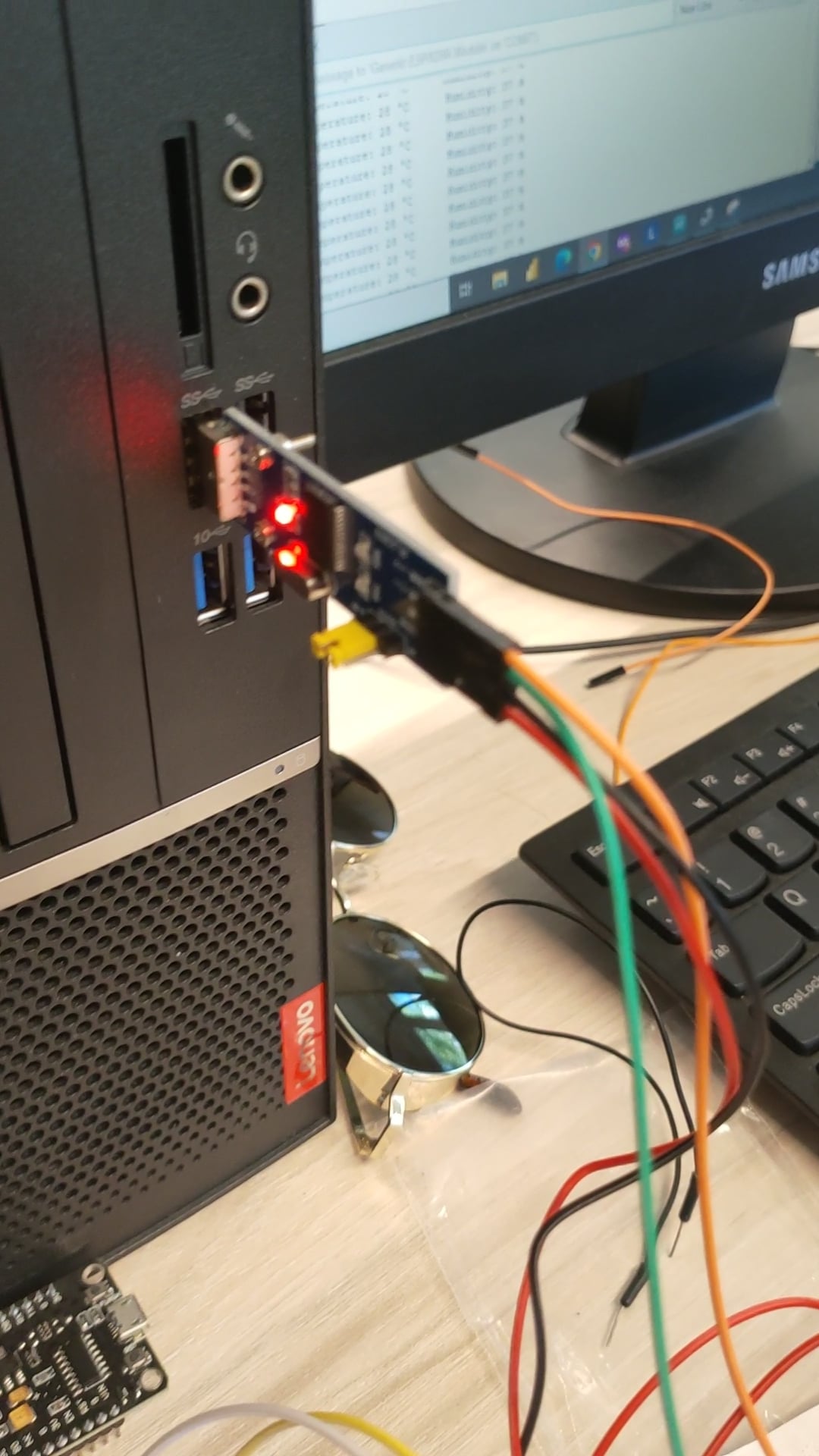
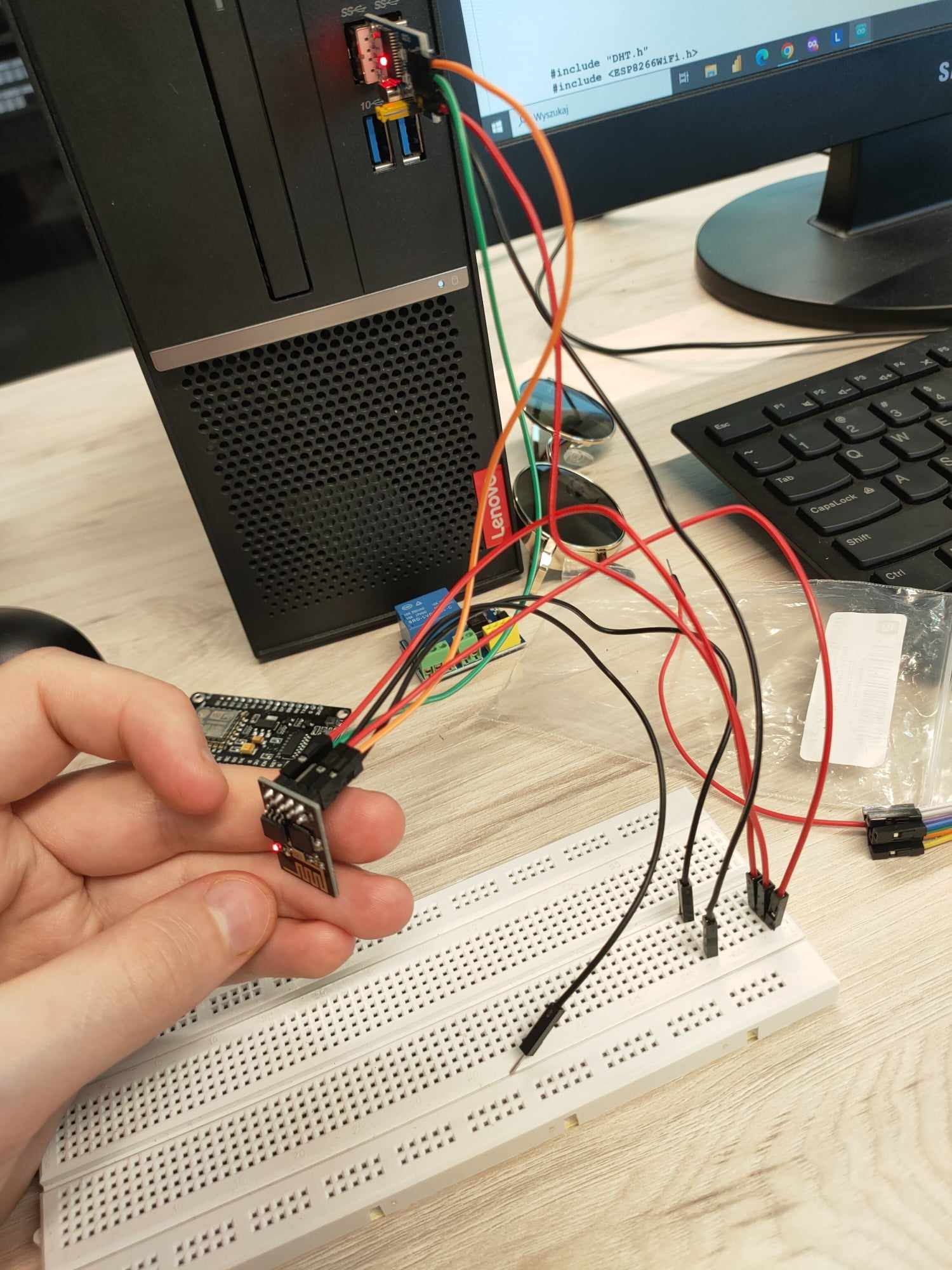
Rys. 2. Nasłuchiwane i wysyłane dane na topicu. Oba widoczne naraz w dwóch terminalach.

**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.

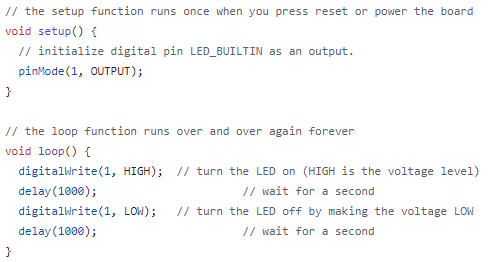
Dokładnie sprawdziliśmy poprawność połączeń między programatorem a modułem ESP01. Pomyłka w podłączeniu kabli mogłaby skutkować błędami w programowaniu lub nawet uszkodzeniem sprzętu. W związku z tym, upewniliśmy się, że każdy przewód jest podłączony do odpowiedniego pinu.



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

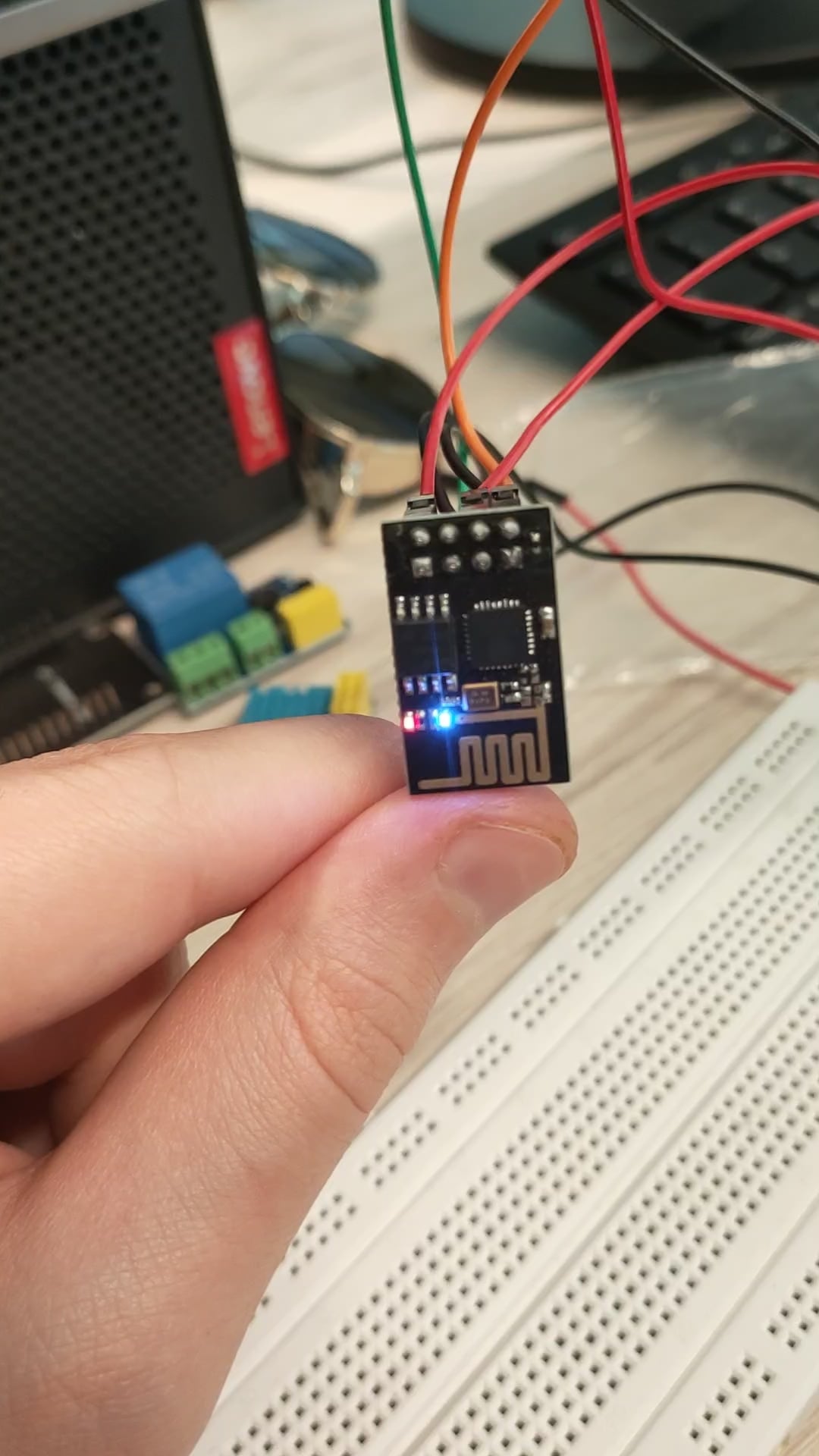
Po prawidłowym podłączeniu urządzeń przystąpiliśmy do wgrania szkicu na moduł ESP01. Użyliśmy do tego środowiska Arduino IDE, w którym przygotowaliśmy prosty szkic implementujący funkcję migania diody LED. Szkic ten został następnie skompilowany   
i wgrany na moduł ESP01.

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 szkic. W kodzie tego szkicu zawarliśmy informacje niezbędne do połączenia z siecią Wi-Fi, takie jak nazwa sieci (SSID) i hasło.

* Na następnym laboratorium otrzymaliśmy czujnik pozwalający na odczyt temperatury z otoczenia.
* Musieliśmy zaprogramować nasz ESP01 tak aby łączył się z siecią wifi, w tym celu ponownie wykorzystaliśmy programator, w kodzie szkicu podaliśmy dane które były wymagane aby podłączyć się do sieci, a następnie wgraliśmy szkic na płytkę. Ponownie musieliśmy chwilowo odpiąć zasilanie płytki aby nowy szkic został uruchomiony.
* Fakt czy urządzenie zostało podłączone mogliśmy monitorować na telefonie, poprzez zmianę liczby urządzeń podłączonych do sieci.
* Następnie musieliśmy poprawnie połączyć kablami czujnik temperatury, programator i moduł ESP01
* Wgrany przez nas szkic został zmodyfikowany aby oprócz łączenia się z siecią wifi, dodatkowo pobierał temperaturę i wilgotność powietrza z czujnika a następnie publikował ją wykorzystując protokół mqtt
* Po wgraniu tego szkicu mogliśmy zaobserwować w środowisku Arduino odczytywaną temperaturę i wilgotność powietrza, na drugim laboratorium nie mieliśm urządzeń Raspberry którymi moglibyśmy połączyć się jako subskrybent do tematu z temperaturą w celu odczytania danych z czujnika.