Politechnika Gdańska
Wydział Elektroniki, Telekomunikacji i Informatyki

Sprawozdanie z przedmiotu:
Projektowanie Oprogramowania Systemów - 2025
Prowadzący: Bartłomiej Dec

Temat projektu: Bezprzewodowa automatyczna stacja pogodowa, oparta o ESP32

Autorzy:
Maciej Politowski
Nr indeksu: 188658
E-mail s188658@student.pg.edu.pl

Damian Holk
Nr indeksu: 188750
E-mail: s188750@student.pg.edu.pl

Streszczenie:
Projekt "Bezprzewodowa automatyczna stacja pogodowa, oparta o ESP32 miał na celu stworzenie kompletnego systemu pomiarowego do monitorowania warunków atmosferycznych w czasie rzeczywistym. Urządzenie oparte zostało na mikrokontrolerze ESP32-C3, który współpracuje z zestawem czujników środowiskowych:

- BME280 (do pomiaru temperatury, wilgotności i ciśnienia)
- BH1750 (do pomiaru natężenia światła)
- YL-83 (do wykrywania opadów deszczu)
- PlanTower PMSA003-A (do pomiaru czystości powietrza).

Pomiar danych odbywa się cyklicznie, co 30 sekund, a zebrane informacje są przesyłane bezprzewodowo za pomocą WiFi do komputera. Urządzenie może działać zarówno bezprzewodowo (dzięki zasilaniu z baterii) jak i być zasilanym za pomocą zasilacza 5V.
Na komputerze użytkownika działa dedykowana aplikacja stworzona w języku Python, która odbiera dane z urządzenia i prezentuje je za pomocą przejrzystego interfejsu graficznego. Urządzenie posiada także wewnętrzną pamięć i umożliwia przesyłanie historii do komputera, który zapisuje ją w formacie .csv. Urządzenie zostało zaprojektowane z myślą o prostocie obsługi oraz możliwości łatwego rozszerzenia o kolejne moduły pomiarowe lub integrację z platformami IoT.

Gdańsk, czerwiec 2025

1.Identyfikacja, specyfikacja, opis

1.1 Klient i użytkownicy końcowi

- Właściciele domów jednorodzinnych, a także mieszkań z możliwością montażu urządzenia na balkonie lub na zewnątrz okna
- Domownicy wykorzystujący zebrane informacje do wyboru odpowiedniego ubioru i dbania o roślinność

1.2 Wymagania funkcjonalne

- Automatyczny, cykliczny odczyt wszystkich dostepnych czujników
- Wizualizacja danych aktualnych (z ostatnich 24 godzin) w formie wykresów
- Powiadomienia informujące o możliwym wystąpieniu awarii czujników/problemów z ich odczytem
- Powiadomienia o wystąpieniu nieprzychylnych warunków pogodowych (niska temperatura, opady deszczu, bardzo niskie lub bardzo wysokie ciśnienie, wysokie stężenie zanieczyszczenia powietrza)
- Możliwość włączania i wyłączania systemu
- Zapis danych (historii) w pamięci wewnętrznej urządzenia
- Cykliczne przesyłanie zebranej historii do komputera
- Odebranie i zapis przesłanej historii do plików csv

1.3 Wymagania poza funkcjonalne

- System powinien działać 24/7
- Możliwość zwiększenia ilość czujników/ich łatwa wymiana w wypadku uszkodzenia
- Bardzo niskie zużycie energii (system powinien przez 5 lat działać na dwóch bateriach AAA)

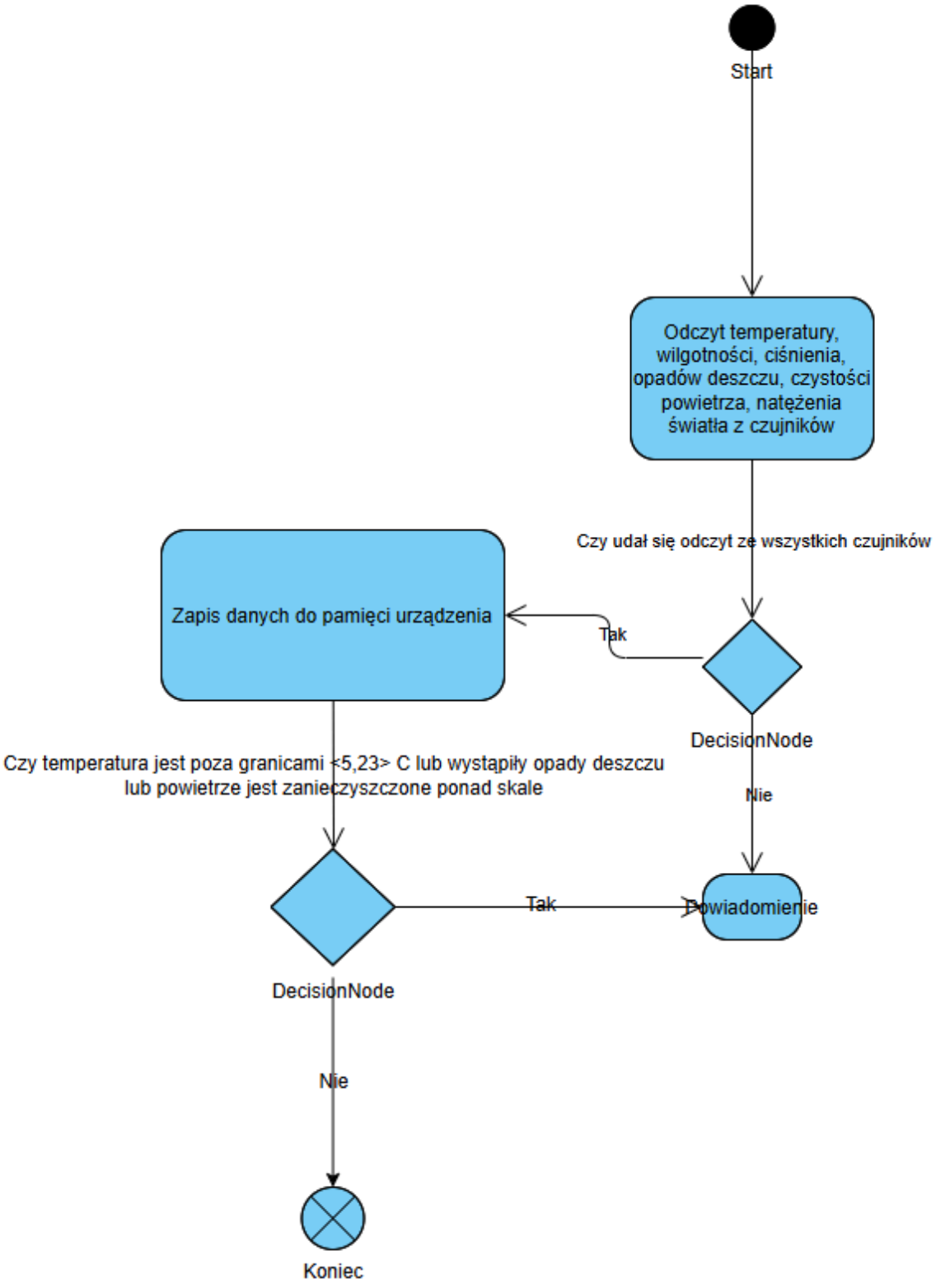
1.4 Scenariusze użycia

- Powiadomienia: Alert przy wykryciu błędów czujników lub ekstremalnych wartości.
- Podgląd wykresów: Użytkownik przegląda historię pomiarów z wybranego zakresu czasu.
- Pomiar cykliczny: System co 30 sekund zapisuje aktualne dane pogodowe.
  Włącznie/wyłączanie: Użytkownik może sterować aktywnością urządzenia

1.5 Specyfikacja funkcjonalna

- Odczyt temperatury, wilgotności i ciśnienia z czujnika BME280
- Odczyt natężenia światła z BH1750
- Sprawdzenie czy pada deszcz przy użyciu YL-83
- Pomiar czystości powietrza przez czujnik PlanTower PMSA003-A
- Aplikacja w języku Python działająca na komputerze
- Powiadomienia do aplikacji

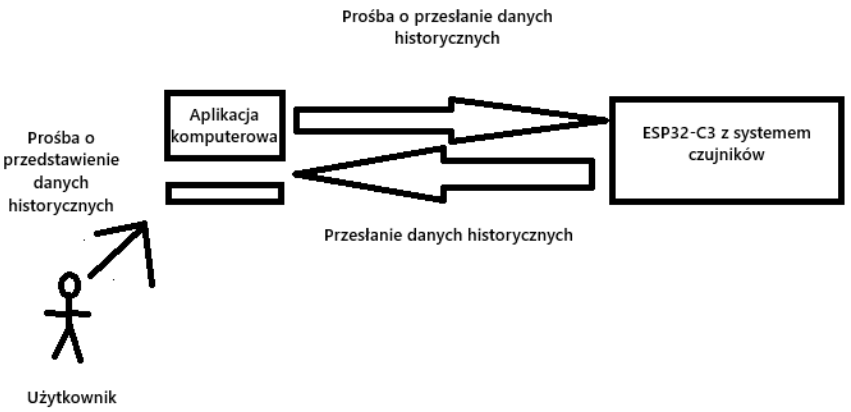
1.6 Diagram UML aktywności



Opis: Diagram przedstawia sposób funkcjonowania systemu. Część główna (po lewej), wykonywana cyklicznie (zaczynająca się od czarnej kropki) najpierw

wykonuje odczyt ze wszystkich czujników:
-BME280 (do pomiaru temperatury, wilgotności i ciśnienia)
-BH1750 (do pomiaru natężenia światła)
-YL-83 (do wykrywania opadów deszczu)
-PlanTower PMSA003-A (do pomiaru czystości powietrza)
Odczyt jest wykonywany przez mikrokontroler ESP32-C3, do którego dane czujniki są podłączone. W przypadku gdy nie powiódł się odczyt chociaż jednego z czujników, to mikrokontroler wysyła powiadomienie, o możliwym uszkodzeniu czujnika. Jeśli mikrokontroler dostał właściwe informacje zwrotne od wszystkich czujników, to dane te są zapisywane. Następnie otrzymane wartości są sprawdzane, pod względem tego czy wystąpiły opady, czy temperatura jest poza ustalonymi granicami (bardzo zimno lub bardzo ciepło) lub czy powietrze stało się nagle mocno zanieczyszczone. Jeśli którykolwiek z tych warunków zostanie spełniony, system wysyła odpowiednie powiadomienie informujące użytkownika o wystąpieniu zmian pogodowych, które mogą mu pokrzyżować plany. Jeżeli żaden z warunków nie zostanie spełniony to system kończy cykl i zaczyna oczekiwanie na rozpoczęcie kolejnego cyklu.

1.7 Diagram UML przypadków użycia



Opis: W wypadku wywołania przez użytkownika prośby o przesłanie informacji historycznych z poziomu interfejsu aplikacji komputerowej, mikrokontroler szyfruje trzymane dane historyczne i wysyła je na komputer, na którym dane są odszyfrowane i zapisywane, następnie komputer wysyła potwierdzenie otrzymania informacji i właściwego ich odczytania. Jeśli komputer odbierze dane niekompletne, albo nie będzie mógł ich odczytać, poprosi kolejny raz o przesłanie informacji.

2.Repozytorium, wykorzystane technologie, środowiska

2.1 Repozytorium

-Repozytorium: GitHub, projekt: Automatyczna-stacja-pogodowa
-Osoba odpowiedzialna za repozytorium: Maciej Politowski

2.2 Struktura repozytorium:

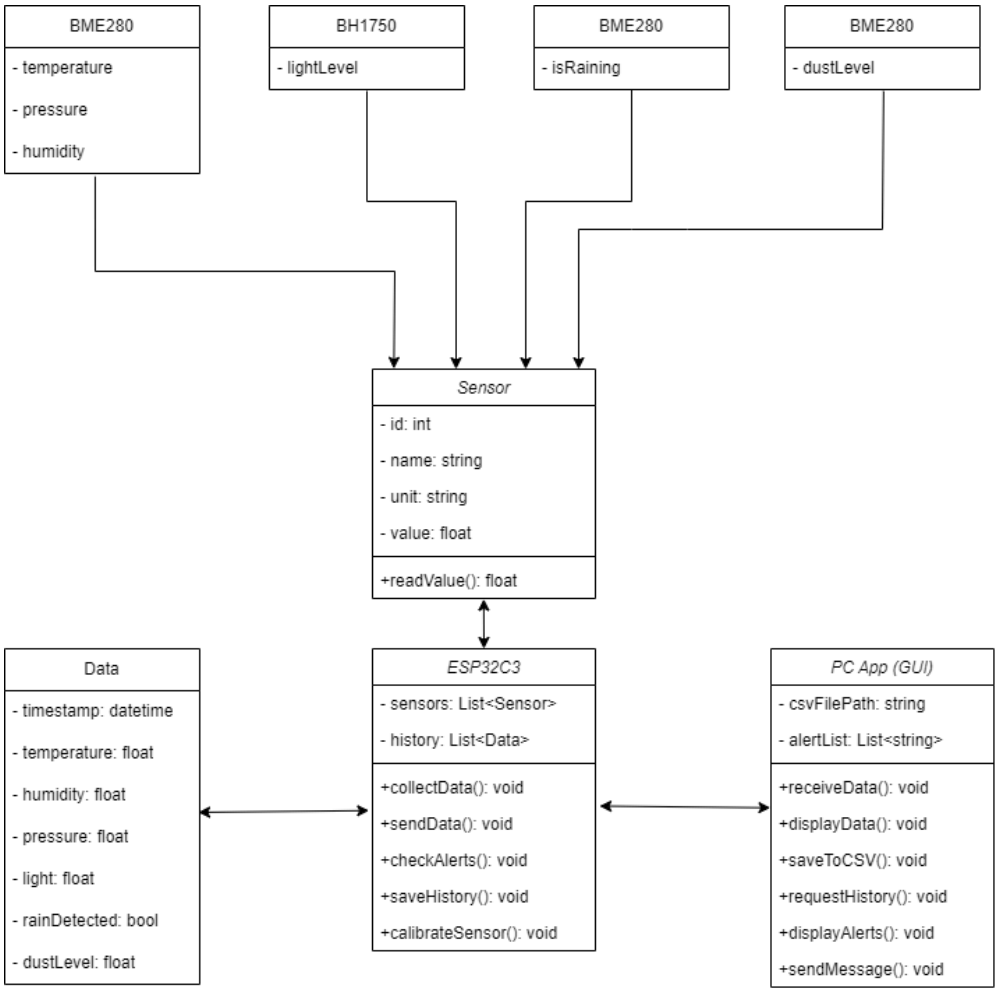
-src - kod źródłowy (Arduino IDE)
-PC App - frontend (python3)
-UML diagrams - diagramy UML
-docs - dokumentacja

2.3 Wykorzystane technologie

-Arduino IDE 2.3.6
-Python 3.14
-BME280, BH1750, YL-83, PlanTower PMSA003-A (czujniki)

3. Model UML aplikacji, biblioteki, specyfikacja techniczna, harmonogram prac

3.1. Model UML aplikacji



3.2 Wykorzystanie bibliotek, aplikacji, osprzętu

Biblioteki w Arduino IDE:

- Adafruit BME280 (obsługa sensora bme280)
- Adafruit Sensor (obsługa wielu sensorów firmy Adafruit)
- BH1750 (obsługa sensora BH1750)
- PMSA003 (obsługa sensora PMSA003)
- Serial1 (komunikacja szeregowa UART)

Biblioteki w Python:

- Flask (obsługa komunikacji po WiFi z urządzeniem)
- tkinter (interfejs graficzny)
- pandas (obsługa i zapis danych)
- matplotlib (wizualizacja danych)

3.3 Specyfikacja techniczna funkcji

- Funkcja odczytu temperatury, wilgotności i ciśnienia: cyklicznie co 30 sekund
- Funkcja odczytu natężenia światła: cyklicznie co 30 sekund
- Funkcja odczytu czystości powietrza: cyklicznie co 30 sekund
- Funkcja odczytu natężenia opadów: cyklicznie co 30 sekund
- Funkcja wysyłania danych do komputera: cyklicznie co 30 sekund
- Funkcja zapisu historii odczytów: cyklicznie co 1 dzień
- Funkcja powiadamiania o opadach: wysyła komunikat do komputera przy wykryciu opadu atmosferycznego
- Funkcja odświeżania ekranu w aplikacji: po odbiorze danych z urządzenia
- Funkcja odczytów alertów/błędów z sensorów

3.4 Harmonogram prac

Etap	**Osoba**	**Data rozpoczęcia**	**Data zakończenia**
Analiza wymagań	Zespół	20.05.2025	20.05.2025
Projektowanie UML	Zespół	21.05.2025	24.05.2025
Projektowanie Hardware	Maciej Politowski	24.05.2025	31.05.2025
Implementacja Backendu (komunikacja)	Damian Holk	24.05.2025	31.05.2025
Implementacja Fronendu (aplikacja)	Zespół	31.05.2025	06.06.2025
Integracja i testy	Zespół	06.06.2025	13.06.2025

Poprawki w projekcie	Zespół	13.06.2025	20.06.2025
Dokumentacja i raport	Zespół	20.06.2025	24.06.2025

5. Testowanie systemu

5.1 Test podstawowego działania funkcji
-Test poprawności odczytu każdego z czujników
-Test reakcji systemu na spełnienie warunków wymagających wysłania powiadomienia
-Test reakcji systemu na otrzymanie prośby o przesłanie danych historycznych
-Test zapisu otrzymanych danych historycznych do pliku
-Test poboru prądu
5.2 Testy symulacji problemów systemu
-Symulacja awarii czujników osobno, a także razem
-Symulacja braku odpowiedzi od mikrokontrolera przy próbie uzyskania danych historycznych poprzez aplikację
-Symulacja braku uzyskania potwierdzenia przez mikrokontroler na wysłane dane historyczne lub powiadomienia
5.3 Testy jednostkowe
-Testy odczytu czujnika deszczu w celu kalibracji progu stwierdzającego deszcz (wilgoć może spowodować błędny odczyt)
-Test wyświetlania pomiarów w GUI
6. Zdjęcia prototypu, zrzut ekranu GUI aplikacji, raport testowy
-Zrzut ekrany z wyglądu aplikacji komputerowej
-Raport testu (zawarty w pliku o nazwie “Testy_raport”)