

POLITECHNIKA ŚWIĘTOKRZYSKA
Wydział Elektrotechniki, Automatyki i Informatyki

Patryk Piotr Niziołek
Numer albumu: 089028

**Opracowanie i implementacja inteligentnego systemu
do zdalnego nawadniania pól uprawnych**

**Praca dyplomowa
na studiach I-go stopnia
na kierunku Informatyka**

Opiekun pracy dyplomowej:
dr inż. Karol Wieczorek
Katedra Systemów Informatycznych

Kielce, 2022

POLITECHNIKA ŚWIĘTOKRZYSKA
WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI

Studia stacjonarne
Kierunek: INFORMATYKA

Zatwierdzam: **PRODZIEKAN**
ds. Studenckich i Dydaktyki
Wydziału Elektrotechniki, Automatyki i Informatyki

Rok akademicki: 2021/2022

..... dr inż. Barbara Łukawska
Prodziekan ds. studenckich i dydaktyki

ZADANIE NA PRACĘ DYPLOMOWĄ STUDIÓW PIERWSZEGO STOPNIA

Wydano dla studenta:

Patryk Niziołek

I. Temat pracy:

Opracowanie i implementacja inteligentnego systemu do zdalnego nawadniania pól uprawnych
The development and the implementation of an intelligent remote irrigation system for farmland

II. Plan pracy:

1. Projekt układu urządzenia nawadniającego
2. Opis wykonanego urządzenia
3. Założenia funkcjonalne aplikacji do zarządzania systemem nawadniającym
4. Implementacja aplikacji według założeń z pkt. 3
5. Opracowanie i wykonanie testów
6. Wnioski i plan dalszego rozwoju

III. Cel pracy:

Celem pracy jest projekt i implementacja urządzenia nawadniającego wraz z aplikacją do zarządzania systemem zdalnych nawadniaczy. Technologie wykonania są w kwestii Dyplomanta.

IV. Uwagi dotyczące pracy:

V. Termin oddania pracy: 2 lutego 2022 r.

Opiekun promotoryczny
Katedry Systemów Informatycznych
Wydziału Elektrotechniki, Automatyki i Informatyki
..... dr hab. inż. Roman Stanisław Deniziak, prof. PŚK
(pieczęć i podpis)

Promotor pracy dyplomowej

..... Karol Wierzech
dr inż. Karol Wierzech

Temat pracy dyplomowej celem jej wykonania otrzymałem(am):

Kielce, dnia 2021-12-21 r. Niziołek Patryk
czytelny podpis studenta



Politechnika Świętokrzyska

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI I INFORMATYKI

Kielce, dnia 24.01.2022r.

Imię i nazwisko studenta: WioJek Patryk nr albumu: 089028

Adres zamieszkania: Łaz 12 Starachowice

Kierunek, specjalność, rok studiów, rodzaj studiów (stacjonarne, niestacjonarne): Informatyka 4 rok stacjonarne

Opiekun pracy dyplomowej inżynierskiej/magisterskiej: dr inż. Karol Wieczorek

OŚWIADCZENIE

Przedkładając w roku akademickim 2021/2022 opiekunowi pracy dyplomowej inżynierskiej/magisterskiej, powołanemu przez Dziekana Wydziału Elektrotechniki Automatyki i Informatyki Politechniki Świętokrzyskiej, pracę dyplomową inżynierską/magisterską* pod tytułem:

Opracowanie i implementacja inteligentnego systemu do automatycznego nadzoru i sterowania procesami produkcyjnymi

oświadczam, że:

- 1) przedstawiona praca dyplomowa inżynierska/magisterska* została opracowana przeze mnie samodzielnie, stosownie do wskazań merytorycznych opiekuna pracy,
- 2) przy wykonywaniu pracy dyplomowej inżynierskiej/magisterskiej* wykorzystano materiały źródłowe, w granicach dozwolonego użytku wymieniając autora, tytuł pozycji i miejsce jej publikacji,
- 3) praca dyplomowa inżynierska/magisterska* nie zawiera żadnych danych, informacji i materiałów, których publikacja nie jest prawnie dozwolona,
- 4) przedstawiona praca nie była wcześniej przedmiotem procedur związanych z uzyskaniem stopnia zawodowego/naukowego w wyższej uczelni,
- 5) niniejsza wersja pracy jest identyczna z załączoną treścią elektroniczną (na CD i w systemie Archiwum Prac Dyplomowych).

Przyjmuję do wiadomości, iż w przypadku ujawnienia naruszenia przepisów ustawy o prawie autorskim i prawach pokrewnych, praca dyplomowa inżynierska/magisterska* może być unieważniona przez Uczelnię, nawet po przeprowadzeniu obrony pracy.

Zostałem uprzedzony:

- 1) o odpowiedzialności karnej wynikającej z art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (Dz. U. 1994 Nr 24, poz. 83, t.j. Dz. U. 2018 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”,
- 2) o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 i 2 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668, z późn. zm.): „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta. Student może być ukarany przez rektora lub komisję dyscyplinarną”.

Prawdziwość powyższego oświadczenia potwierdzam własnoręcznym podpisem.

WioJek Patryk
czytelny podpis studenta

*) niepotrzebne skreślić

Opracowanie i implementacja inteligentnego systemu do zdalnego nawadniania pól uprawnych

Cel pracy to wykonanie oprogramowania do zdalnego sterowania opracowanego układu urządzenia nawadniającego. Ideą stworzenie takiego rozwiązania jest optymalizacja zużycia wody przez spryskiwacze na polu uprawnym oraz automatyzacja samego procesu podlewania. Użytkownik końcowy za pomocą platformy internetowej ma możliwość sterowania urządzeniami podłączonymi za pomocą sieci Wifi. Urządzenie jest również w stanie pracować w pełni autonomicznie, ponieważ w realnym czasie analizuje dane z podłączonych czujników. Głównym czynnikiem decyzyjnym pracy spryskiwaczy jest aktualna wartość wilgotności gleby. W pracy uwzględnione zostały również zabezpieczenia przed nieoczekiwanymi awariami lub niesprzyjającymi warunkami pogodowymi, które miałyby by negatywny wpływ na prace spryskiwacza.

Słowa kluczowe: Thingworx, Arduino, Spryskiwacz, Nawadnianie

The development and the implementation of an intelligent remote irrigation system for farmland

The goal of the work is to develop software for remote control of the developed system of irrigation equipment. The idea of creating such a solution is to optimize the use of water by sprinklers in the field and to automate the process of watering. The end user has the ability to control devices connected via Wi-Fi using a web platform. The device is also able to work fully autonomously because it analyzes data from connected sensors in real time. The main decision factor of sprinkler operation is the current value of soil moisture. In the work what is also taken into account is the protection against unexpected failures or unfavorable weather conditions, which would have a negative impact on the work of the sprinklers.

Keywords: Thingworx, Arduino, Sprinkler, Irrigation

SPIS TREŚCI

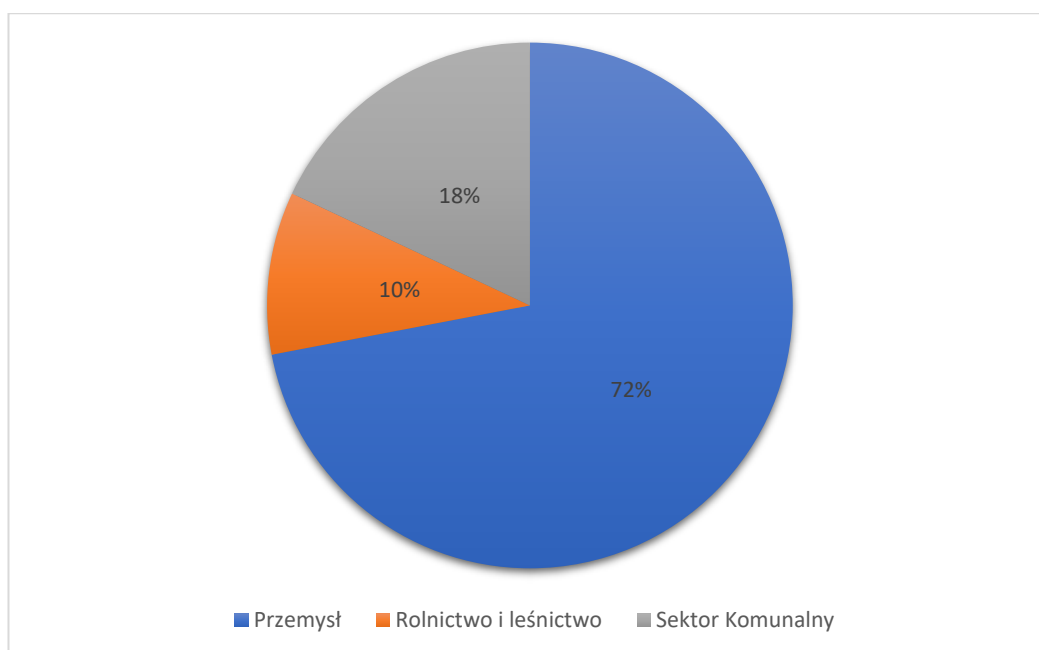
SPIS TREŚCI	9
1. WSTĘP	11
2. CEL I ZAKRES PRACY	13
3. ZASADY DZIAŁANIA SYSTEMU	14
3.1. Korzyści płynące z IoT.....	14
3.2. Użyte technologie	14
3.2.1. Platforma Thingworx.....	14
3.2.2. Thingworx kompozytor.....	15
3.2.3. Thingworx mashup.....	16
3.2.4. Zintegrowane środowisko programistyczne arduino.....	17
3.3. Założenia systemu	18
4. PROJEKT PROTOTYPU SPRYSKIWACZA	19
4.1. Działanie prototypu spryskiwacza.....	19
4.2. Reakcja na wybór użytkownika.....	20
4.3. Podzespoły Arduino	21
4.3.1. Mikrokontroler	21
4.3.2. Moduł ESP32.....	23
4.3.3. Czujnik poziomu wody.....	25
4.3.4. Fotorезystor.....	25
4.3.5. Czujnik temperatury i wilgotności powietrza.....	26
4.3.6. Czujnik wilgotności gleby.....	26
4.3.7. Pompa wody 5V i przełącznik 1-kanalowy	27
4.3.8. Ekran OLED	29
4.4. Schemat Arduino.....	30
4.5. Zapytania HTTPS.....	31
5. PROJEKT PLATFORMY ZBUDOWANEJ W THINGWORX	33
5.1. Interfejs użytkownika.....	33
5.2. Zabezpieczenia przed awarią.....	37
5.3. Monitorowanie pogody.....	38
6. WERYFIKACJA ORAZ TESTOWANIE APLIKACJI	39
6.1. Weryfikacja działania czujnika wilgotności gleby.....	39
6.2. Test automatycznego podlewania.....	40
6.3. Weryfikacja czujnika temperatury.....	41
6.4. Weryfikacja rezystora wrażliwego na światło.....	43
6.5. Weryfikacja czujnika poziomu wody w zbiorniku.....	44
6.6. Weryfikacja ekranu OLED.....	45
7. MOŻLIWOŚĆ ROZWOJU	46
7.1. Rozwój prototypu spryskiwacz	46
7.2. Rozwój platformy.....	46
8. PODSUMOWANIE	47
LITERATURA	48
Spis rysunków	50
Dodatek	53

1.Wstęp

Współcześnie, gdy priorytet to innowacyjność jest bardzo duże prawdopodobieństwo spotkania się z pojęciem Internetu rzeczy. W dzisiejszych czasach ten trend jest prężnie rozwijany i technologia oparta na Internecie rzeczy mocno zakorzenia się na rynku. Jej filozofia oparta jest na automatyzacji procesów, które do niedawna wykonywał człowiek. Zadaniem takiego oprogramowania jest odciążenie człowieka od obowiązków monitorowania parametrów oraz rozszerzeniem funkcjonalności urządzeń o umiejętność samodzielnego podejmowania decyzji, kiedy jest to wymagane. Zastosowanie takich rozwiązań w urządzeniach jest nie ocenione w procesach produkcyjnych, które znacząco można w ten sposób zoptymalizować. Koncepcja Internetu rzeczy ma zastosowanie również w produkcji i powoli staje się normą, dzięki której produkty są atrakcyjne dla potencjalnego kupującego. Podążając za duchem czasu wielu producentów artykułów gospodarstwa domowego oraz elektroniki użytkowej również w taki sposób rozszerza funkcjonalność swoich urządzeń. Inteligentne rozwiązania jesteśmy w stanie dostrzec w urządzeniach takich jak pralki, lodówki oraz można tu podać wiele innych przykładów urządzeń ułatwiających życie człowiekowi.

Niestety na rynku brakuje rozwiązań pomagających zadbać o wilgotność gleby i jej nawadnianie. Dostrzegając tę lukę idealnym rozwiązaniem jest stworzenie automatycznie podlewającego spryskiwacza, który samodzielnie ustala, kiedy włączyć natrysk. Aplikacja zależnie od problemu wybiera odpowiednie rozwiązanie według wcześniej ustalonych warunków przez człowieka. Czynniki decyzyjne oparte są o zmienne czynniki środowiskowe, które są monitorowane przez czujniki i po analizie tych parametrów ustalany jest wynik w jaki sposób zadziała spryskiwacz. System do zdalnego nawadniania jest elastyczny i nadaje się do pielęgnowania roślin domowych jednak jego zastosowanie nie ogranicza się do małych powierzchni. Potencjalne zastosowanie inteligentnego spryskiwacza to kontrolowanie rozległych pól uprawnych. Sieć połączonych ze sobą spryskiwaczy, możliwość podglądu ich parametrów i sterowania nimi z jednego miejsca gdy mamy dostęp do Internetu znacznie ułatwiłaby kontrolę nad stanem roślin. Stała weryfikacja i kontrola danych bez angażowania człowieka zapewnia całodobową ochronę w przypadku niechcianych awarii i daje możliwość uniknięcia całkowitego zepsucia sprzętu. Długoterminowe monitorowanie danych może przyczynić się również do optymalizacji zużycia wody.

W związku z dostrzeganym deficytem wód nadających się do podlewania roślinności konieczne jest oszczędzenie każdej kropli wody co przyczynia się do ochrony naszej planety. Bez wody słodkiej gatunek ludzki nie byłby w stanie przetrwać, dlatego uwzględnienie w oprogramowaniu ekologicznych funkcjonalności to obowiązek. Jeśli nie zmienimy swojego nastawienia do oszczędzania zasobów naturalnych problem niedostatku wody może wystąpić już za kilkadziesiąt lat. Według Organizacji Narodów Zjednoczonych świat czeka globalny kryzys wodny. Z analizy ekspertów do 2050 roku „Co najmniej co czwarta osoba na świecie będzie mieszkać w kraju dotkniętym chronicznym lub okresowym deficytem wody pitnej” [1]. W Polsce według raportu danych [2] Głównego Urzędu Statystycznego z 2020 roku wiodącym źródłem zaopatrzenia gospodarki narodowej są wody powierzchniowe. Ich pobór w 2019 roku wyniósł 7,4km³ i pokrył 80% potrzeb. Raport pokazuje, że sektor, w którym zawarte jest rolnictwo na który miałyby najbardziej wpływać aplikacja wynosi aż 10% zapotrzebowania wody.



Rysunek 1.1. Diagram przedstawiający zużycie wody w Polsce w 2019r.

Przy rozwijaniu krajowego rolnictwa powinniśmy zwrócić uwagę na ekologię. Racjonalnie wykorzystywać dane nam zasoby i patrzeć w przyszłość, aby nie doprowadzić do suszy i jeszcze większych zmian klimatycznych. Susza mogłaby doprowadzić do głodu co byłoby destrukcyjne w skutkach i doprowadziłaby do fali zgonów ludzi z powodu niedożywienia. Dodatkowym aspektem jest pobór prądu potrzebny do zasilenia pompy wody. Im mniej będzie się wykorzystywać niepotrzebnie spryskiwacze tym bardziej zaoszczędzi się energię elektryczną.

2.Cel i zakres pracy

Celem pracy jest zaprojektowanie systemu, którego interfejs graficzny jest łatwy oraz przejrzysty dla użytkownika końcowego. Zadaniem systemu jest sterowanie zaprojektowanym spryskiwaczem, którego oprogramowanie umożliwi w rzeczywistym czasie komunikowanie się ze stroną internetową na której jest możliwość wyświetlania stanu wejść analogowych dostarczających dane. Użytkownik będzie w stanie sterować natryskami za pomocą dedykowanych do tego przycisków. System ma obowiązek reagowania w przypadku wystąpienia awarii oraz natychmiastowego powiadomienia o tym drogą mailową właściciela sprzętu. Oprócz podstawowych parametrów spryskiwaczowi będzie przypisywana wartość zależna od aktualnej sytuacji czy występuje jakaś awaria. Jeśli istnieje to jaki jest jej powód i dodatkowo będzie dostępny pisemny opis zaistniałego problemu. Priorytetem jest również analiza danych, które dostarczają nam czujniki odpowiedzialne za sprawdzanie wilgotności gleby, temperatury powietrza, wilgotności powietrza, ilości wody w zbiorniku oraz intensywności światła na zewnątrz. Oprogramowanie odpowiedzialne za wymianę danych pomiędzy platformą a urządzeniem musi zadbać o ich bezpieczeństwo i poufność oraz stabilność dostarczania bez względu na ich ilość. Dane powinny być przechowywane na platformie oraz wyświetlane na czytelnych wykresach w przestrzeni czasu, która interesuje użytkownika. Platforma powinna również dawać możliwość wyboru sposobu nawadniania, aby dopasować procent wilgotności gleby do roślinności, która jest w niej hodowana, aby miała idealne warunki rośnięcia. Drugim celem jest skonstruowanie prototypu spryskiwacza, który będzie się komunikować z platformą oraz pobierać dane z czujników. Mikrokontroler sterujący układem oraz jego wszystkie podzespoły będą mogły być zasilane ze źródła napięcia stałego do 5V. Pompa wody powinna mieć osobne zasilanie oraz układ musi być łatwo modyfikowalny, aby móc w prosty sposób zmieniać pompę oraz jej zasilanie bez ingerencji w pozostałą część komponentów i przy takiej konfiguracji zestawu nie potrzebna będzie zmiana kodu programu, który jest wgrany w mikrokontroler. Sieć ma za zadanie automatycznie wykrywać istniejące lub nowe urządzenia oraz nie powinno to wymagać ponownej konfiguracji sieci. Urządzenie powinno mieć unikalny identyfikator, po którym można go rozpoznać na platformie. Zakres pracy również obejmuje testowanie działania spryskiwacza oraz funkcjonalności platformy. Aby użytkownik łatwo mógł zacząć korzystać z platformy została przygotowana instrukcja w jaki sposób ją zainstalować oraz wgrać oprogramowanie do mikrokontrolera.

3. Zasady działania systemu

Moduł, który wykorzystuje układ ESP32 steruje czujnikami, pompą wody oraz wyświetlaczem OLED. Umożliwia to zaprojektowanie urządzeń, które mają możliwość komunikowania się przez sieci bezprzewodowe WiFi oraz Bluetooth. Jest to nowsza wersja układu ESP8266. Nowszy moduł zaopatrzony jest w szybszy mikrokontroler. Dzięki swojej wydajności jeszcze lepiej wpasowuje się w trend szybkiego przesyłania danych w urządzeniach skonstruowanych z myślą o Internecie rzeczy. Czujniki przesyłają dane do modułu, który następnie analizuje wartość współczynnika wilgotności gleby i na podstawie wartości tego parametru ustala czy wymagane jest automatyczne podlewanie. Następnie moduł wysyła te dane za pomocą zapytań HTTPS do platformy Thingworx, która sprawdza, czy dane mieszczą się w normie. Interfejs użytkownika umożliwia sterowanie podłączonymi urządzeniami czy wybranie profilu podlewania. Dane przechowywane są w systemie oraz jest możliwość wyświetlenia ich za pomocą wykresów liniowych.

3.1. Korzyści płynące z Internetu rzeczy

Internet rzeczy(ang. *Internet of Things, IoT*) [3] to system powiązanych ze sobą urządzeń komputerowych, przedmiotów, maszyn mechanicznych i cyfrowych, zwierząt lub ludzi, które wyposażone są w unikalne identyfikatory i mają możliwość przesyłania danych bez konieczności komunikacji człowiek-komputer lub człowiek-człowiek. Technologia ta umożliwia redukcję kosztów, ułatwia człowiekowi pracę na produkcji oraz wspomaga proces monitorowania uzyskanych danych. Dzięki takim danym można tworzyć kluczowe wskaźniki efektywności, które pokazują precyzyjnie informacje o produktywności maszyn lub pracowników. Znając takie dane można starać się zwiększać wydajność w poszczególnych punktach procesu, gdzie najbardziej wymagana jest optymalizacja.

3.2. Użyte technologie

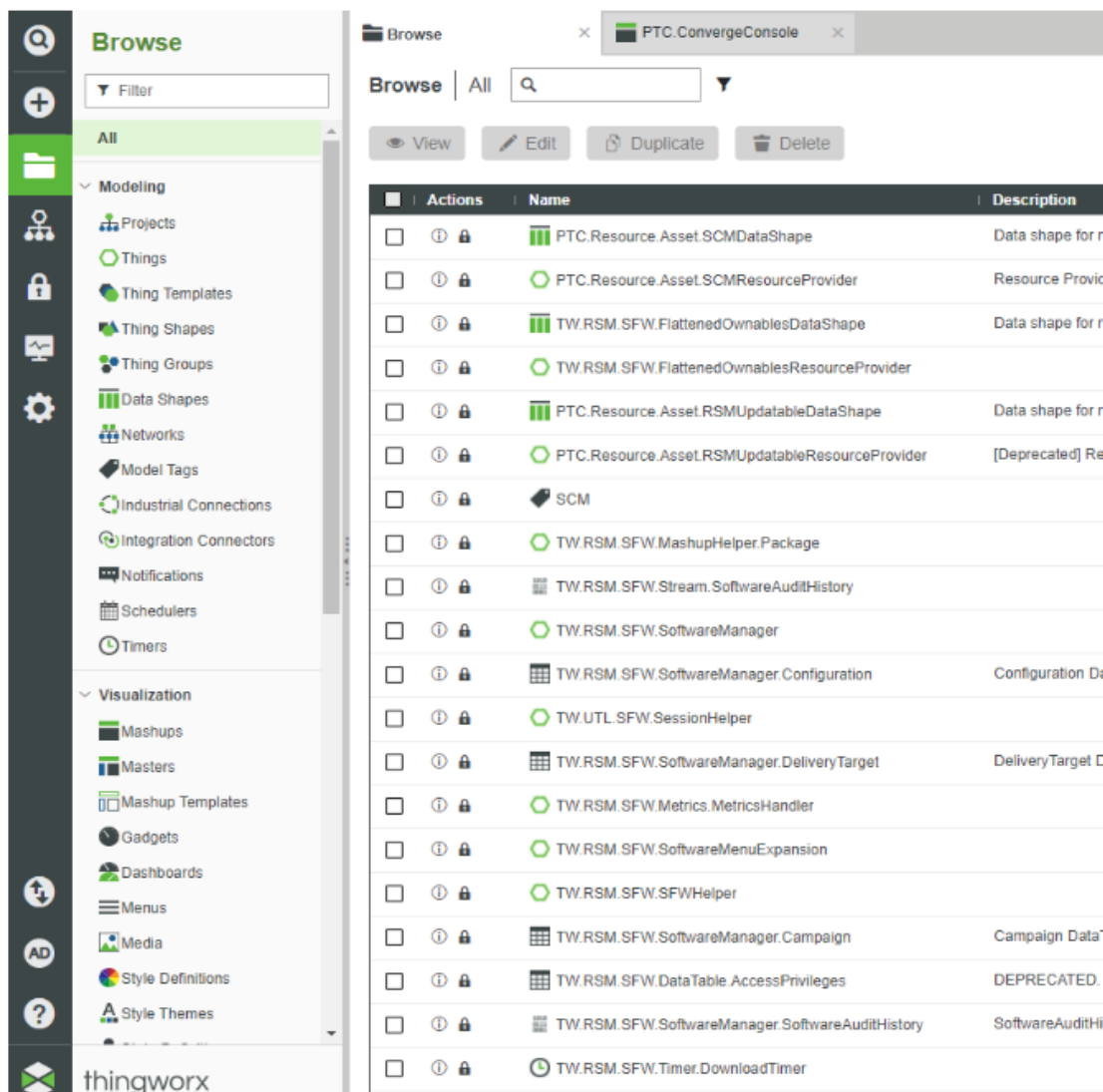
3.2.1. Platforma Thingworx

Thingworx[4] to dedykowana platforma do tworzenia aplikacji dla Internetu rzeczy za pomocą przeznaczonych do tego narzędzi. Jest to środowisko technologiczne dla koncernów, które umożliwia szybko opracować innowacyjne projekty i wdrażać inteligentne rozwiązania do produkcji. Obecnie jej właścicielem jest firma PTC [5] zajmująca się oprogramowaniem i usługami komputerowymi. Platforma zawiera

kompletny zestaw zintegrowanych narzędzi programistycznych i funkcji Internetu rzeczy, co sprawia, że tworzone rozwiązania są proste, szybko wprowadzane na rynek i bardziej atrakcyjne dla użytkownika końcowego. Ideą platformy jest abstrakcja. Jest to sposób na ujednolicenie specyfikacji wszystkich czujników czy sieci i przekształcenie ich w proste interfejsy z którymi można łatwo się zintegrować. Otwarte i łatwo rozszerzalne rozwiązanie umożliwia przedsiębiorcom łatwość rozwijania połączeń pomiędzy urządzeniami, a technologiami chmurowymi czy bazami danych SQL. Dzięki możliwości klasyfikowania danych z czujników oraz konwertowania ich w zrozumiałe informacje oferowanej przez platformę Thingworx dane na niej przechowywane mogą być analizowane np. za pomocą uczenia maszynowego oraz wyświetlane dla użytkownika w czytelnej i zrozumiałej formie. Platforma pomaga skalować systemy przy zachowaniu niezawodności podczas dodawania nowych typów urządzeń, nowych sieci, nowych użytkowników. Dodatkowo zapewnia bezpieczeństwo, autoryzację użytkownika. Spowodowane jest to wymuszeniem środków bezpieczeństwa na każdym poziomie, od urządzenia do aplikacji. Zawiera analizę danych, diagnozowanie problemów, prognozy i planowanie [6].

3.2.2. Thingworx kompozytor

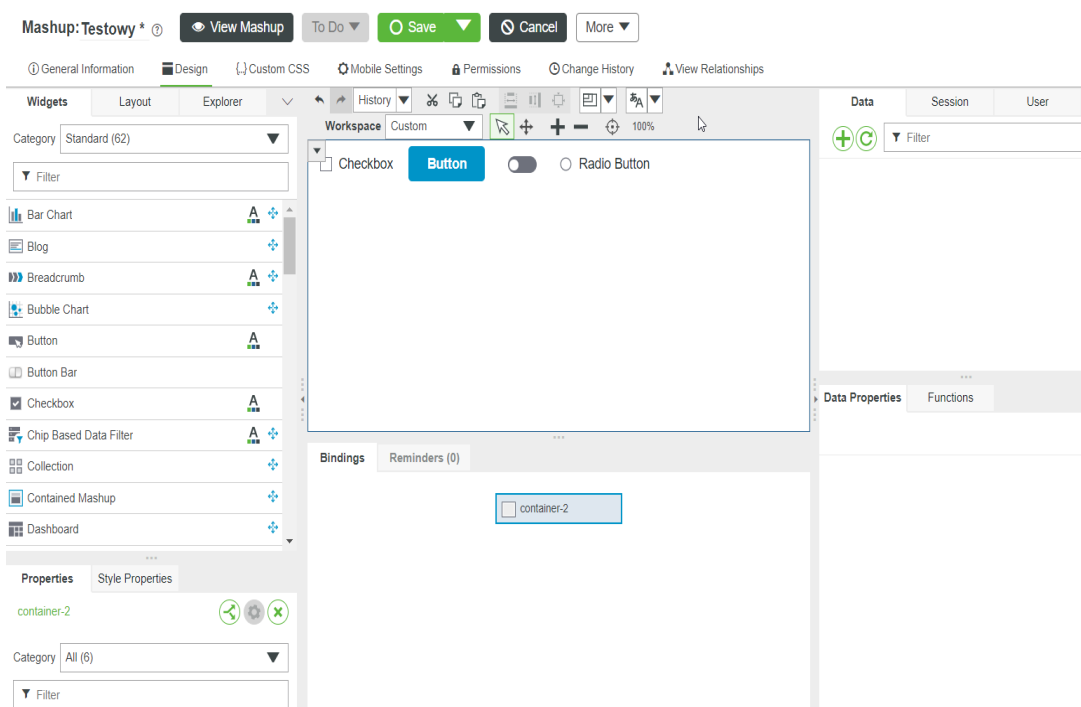
Kompozytor [7] to strona internetowa, na której komponuje się rozwiązania IoT. Prawie wszystko jest uznane za byt, który ma np. odzwierciedlenie w rzeczywistości lub ma swoją funkcjonalność. Byt w IoT to rzecz (zasób, urządzenie), który można również nazywać jednostką lub obiektem fizycznym, który jest jednoznacznie zidentyfikowany i można go gromadzić, przetwarzać i analizować. Komponenty mają swoje funkcje, tabele danych oraz parametry, które mają określone typy. Bytom można zdefiniować zdarzenia, które wywołują subskrypcje.



Rysunek 3.1. Thingworx kompozytor

3.2.3 Thingworx Mashup

Mashup [8] to strona internetowa opracowana w kompozytorze Thingworx. Zawiera widżety (elementy wyglądu na przykład różnego rodzaju przyciski). Mashup umożliwia wykonywanie serwisów, w których ustalone jest zachowanie aplikacji. W większości przypadków „Mashups” komunikują się z Thingworx za pomocą REST API. Wyjątkiem są funkcje, które pobierają wartości właściwości urządzeń i te, które je ustawiają. Te funkcje używają do komunikacji websocketów. W przypadku aplikacji mashupy odpowiedzialne są za interakcje z użytkownikiem.



Rysunek 3.2. Thingworx Mashup

3.2.4. Zintegrowane środowisko programistyczne Arduino

Zintegrowane środowisko programistyczne Arduino (*ang. Arduino IDE-Arduino Integrated Development Environment*) [9] to wieloplatformowa aplikacja napisana w funkcjach z C i C++. Do wykorzystania wszystkich funkcjonalności Arduino wystarczy posiadać podstawowe umiejętności w pisaniu programów w języku C. Składnia kodu jest wnikliwie uregulowana, a każdy błąd niepożądanie będzie wytknięty przez kompilator. Służy do pisania i przesyłania programów na płyty Arduino oraz na kompatybilne płyty programistyczne innych producentów. Szkice, które tworzy się w środowisku bardzo przypominają dokumenty tekstowe w zwykłym edytorze tekstowym jednak w tworzeniu kodu pomagają kolory tekstu, które zależą od tego jaki kod implementujemy. Arduino IDE ma wbudowaną bibliotekę oprogramowania z projektu Wiring, która udostępnia wiele typowych procedur wyjścia i wejścia. Kod napisany przez użytkownika wymaga tylko dwóch podstawowych funkcji, szkicu do uruchomienia programu w postaci funkcji `setup()`, która wykona się tylko raz i głównej pętli `loop()` wykonywanej za każdym razem. Funkcje są kompilowane i łączone za pomocą skrótu programu `main()` w wykonywanym cyklicznym programie wykonawczym z łańcuchem narzędzi GNU, również zawartym w dystrybucji IDE [10].



Rysunek 3.3. Środowisko Arduino IDE

3.3. Założenia systemu

Użytkownik posiada możliwość podejrzenia wartości z czujników na ekranie lub zdalnie za pomocą strony internetowej, jeśli ma dostęp do Internetu. Dane takie jak temperatura, muszą być przedstawione w odpowiednich jednostkach natomiast reszta wartości w skali od 0 do 100%. Szybki czas pętli programu daje wrażenie płynności więc maksymalny czas sprawdzania wartości i wysłania ich na platformę nie może być większy niż 10s. System w tym czasie ma obowiązek również odbierać dane od platformy i w przypadku jakiejś zmiany wprowadzonej przez użytkownika reagować w odpowiedni sposób. System powinien mieć dwa tryby. Pierwszy automatyczny, który działa na zasadzie, że w pierwszej kolejności sprawdza jaki profil podlewania jest włączony. W profilu koniecznie musi być uwzględnione od jakiego procentu wilgotności gleby powinna włączyć się pompa wody. Minimalna ilość takich profilów to trzy, aby użytkownik mógł dopasować wilgotność gleby do hodowanych roślin. Jeśli wartość wilgotności spadnie poniżej wymaganej spryskiwacz będzie podlewał, dopóki nie wykryje wilgotności o 10% wyższej niż minimum. Takie rozwiązanie pozwoli uniknąć sytuacji, gdzie wartość wilgotności jest progowa co powodowałoby, że pompa włącza i wyłącza się w bardzo krótkich odstępach czasu. Takie zachowanie mogłoby doprowadzić do zniszczenia pompy. Drugim trybem jest tryb manualny. W tym trybie użytkownik online może samodzielnie ingerować w stan pompy za pomocą przełącznika. Dodatkową funkcjonalnością jest tryb zależny od pogody, który po wcześniejszym pobraniu danych z platformy ustala czy potrzebne będzie podlewanie. Jeśli warunki pogodowe wymagają dodatkowego podlania gleby to urządzenie włącza na określony czas spryskiwacz.

4. Projekt prototypu spryskiwacza

Zadaniem oprogramowania jest stworzenie modelu, z którym można zintegrować zależnie od potrzeb dowolnego rodzaju pompę i jej zasilanie od 5V do 230V. Jest możliwość dołożenia modułu GSM, który pozwalałby na sterowanie spryskiwaczami na dużych odległościach, aby użytkownik mógł zautomatyzować nawadnianie np. pól uprawnych. Na rynku można znaleźć takie produkty jak „TinySline GSM shield” [11]. Płytką tą daje możliwość połączenia się do sieci GSM oraz używać funkcjonalność GPRS. General Packet Radio Service zapewnia przesyłanie danych w zakresie 30-80 kb na sekundę [12]. Jest to wystarczające dla tego projektu, ponieważ potrzebne jest jedynie wysyłać i odbierać małe ilości danych. Takie technologie jak SMS opierają się na GPRS [13]. Daje to możliwość wysyłania SMS w przypadku awarii. Dzięki GSM możliwe jest również wykorzystanie transmisji danych w celu uzyskania dostępu do Internetu. Dzięki tej płytce również można korzystać z zapytań HTTPS więc nie zmniejszyłoby to funkcjonalności, która daje nam sieci WiFi. Zastosowanie takiej płytki jest jednak kosztowne i wymaga wykupienia pakietu u operatora sieciowego więc prototyp spryskiwacza komunikuje się z platformą po WiFi jednak system jest tak skonstruowany, aby w przyszłości jego modyfikacja była możliwa.

4.1. Działanie prototypu spryskiwacza

setup()

- Połączenie się z wifi
- ustawienie pinów sygnałowych na output i pinów zasilających na „low”
- Włączenie wyświetlacza i wyświetlenie logo

Loop()

- Pobranie wartości z czujników, mapowanie i wyświetlenie ich na wyświetlaczu
- Sprawdzenie istnienia urządzenia na platformie, jeśli nie dodanie go oraz routera
- Wysyłanie wartości z czujników na platformę za pomocą zapytania HTTPS PUT
- Pobranie wartości z platformy informujących jak użytkownik chce nawadniać w formacie Json
- Deserializacja Json
- Logika uruchamiająca pompę na podstawie wyborów użytkownika
- Sprawdzenie czy nie ma awarii
- Wysyłanie informacji jaki jest stan pompy

4.2 Reakcja na wybór użytkownika

Logika odpowiedzialna za stan nawadniania jest uzależniona od wyboru opcji przez użytkownika na platformie. Pierwszy parametr dostarczany w zapytaniu do platformy jest związany z stanem opcji włączenia automatyzacji. Odpowiada on za to czy spryskiwacz ma autonomicznie działać lub być uzależniony od drugiego parametru jakim jest status podlewania. Jeśli automatyzacja podlewania jest wyłączona, wtedy spryskiwacz będzie działał, gdy użytkownik włączy opcje podlewania. Trzecią wartością pobraną z platformy jest stan włączenia funkcjonalności pogodowej oraz zwracanej przez nią informacji. Odpowiada ona za wyłączenie podlewania, gdy warunki pogodowe nie uzasadniają jego zastosowania. Natomiast jeśli warunki atmosferyczne stwarzają potrzebę nawadniania pompa włączana na jeden cykl trwający określoną ilość czasu.

```
if(automationStatus==false)
{
    if(irrigationStatus==true)
    {
        irrigationStatus=true;
        pompActive();
    }
    else if(irrigationStatus==false)
    {
        irrigationStatus=false;
        pompDeactive();
    }
}
else if (automationStatus==true){
    irrigationStatus = automationIrrigation(soilMoisturePercent,selectedPrifle);
}
putToThing(sprinklerName,"irrigationStatus","{\"irrigationStatus\" : \"\"+(String)irrigationStatus+\"\"}");
if (weatherAutomationStatus==true)
{
    if(weatherAutomationIrrigation = true){
        pompActive();
        irrigationStatus=true;
        putToThing(sprinklerName,"irrigationStatus","{\"irrigationStatus\" : \"\"+(String)irrigationStatus+\"\"}");
        delay(5000);
        pompDeactive();
        irrigationStatus=false;
        putToThing(sprinklerName,"irrigationStatus","{\"irrigationStatus\" : \"\"+(String)irrigationStatus+\"\"}");
    }
}
```

Rysunek 4.1. Logika uruchamiająca pompę na podstawie wyborów użytkownika

4.3. Podzespoły Arduino

Arduino to przyjazne środowisko, które ma dostęp do pomocnych materiałów open-source oraz szeroki wybór dodatków umożliwiający łatwe tworzenie prototypów własnych rozwiązań. Siłowniki, moduły sprawdzające parametry są nieodłączną częścią prawie każdego produktu Internetu rzeczy. Czujniki mają za zadanie zbierać wartości z otoczenia i zamienić je na sygnały elektryczne, które później przetwarza mikrokontroler. Siłowniki mają odmienną zasadę działania. Elementy odbierają sygnały elektryczne i po nich wiedzą w jaki sposób pracować. Moduły pozwalają korzystać z podstawowych elementów elektronicznych takich jak rezystory, fotorezystory, diody czy kondensatory, ale również z rozbudowanych modułów, które sterują pracą czujników, silników krokowych czy pomp wodnych. Wymagane jest jedynie prawidłowe podłączenie układu i od razu można go programować w środowisku Arduino.



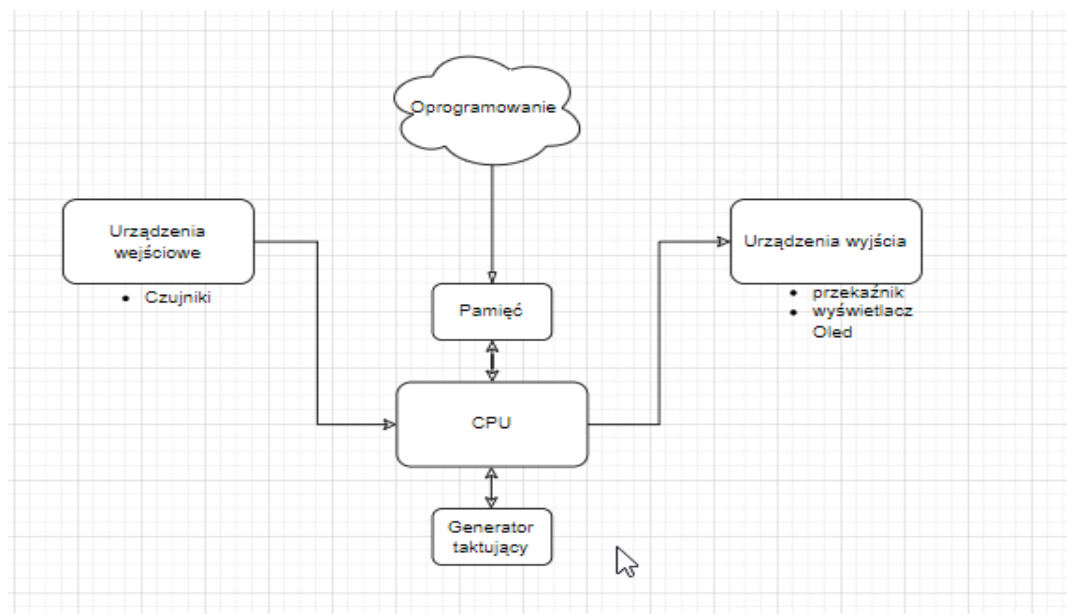
Rysunek 4.2. Podzespoły Arduino [14]

4.3.1. Mikrokontroler

„Mikrokontroler(*MCU, $\mu C, uC$, ang. *Microcontroller**) to scalony system mikroprocesorowy, który jest zrealizowany w postaci pojedynczego układu scalonego.” [15] Głównym zadaniem mikrokontrolera jest sterowanie układami, w których skład wchodzi urządzenia elektroniczne. Jego zastosowanie jest bardzo obszerne np. masowo korzysta się z niego do tworzenia sprzętu AGD i RTV. Mikroprocesor zawiera jednostkę centralną CPU oraz jednostki obliczeniowe ALU, które wbudowane są w mikrokontrolery. ALU mają przeważnie 8bitów, 16bitów, 32bitów lub 64bity. Obowiązkowe jest również posiadanie generatora taktującego. Mikrokontrolery cechuje również rozwinięty układ wejścia-wyjścia. Następnym blokiem funkcjonalnym jest pamięć RAM.



Rysunek 4.3. Przykładowy mikrokontroler PIC18F8720 firmy Microchip Technology [15]



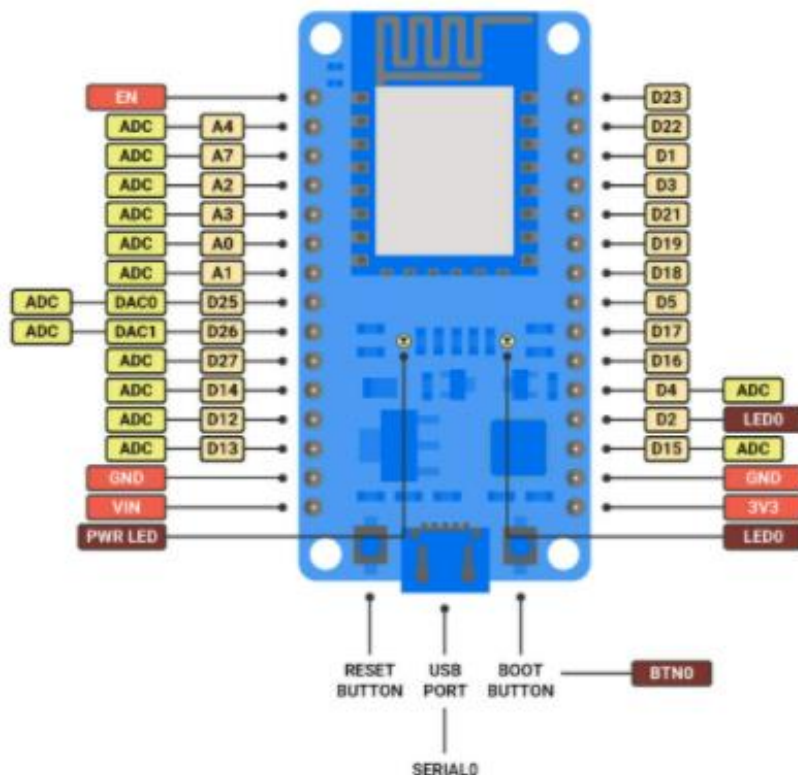
Rysunek 4.4. Schemat układu mikrokontrolera wykonany na stronie [16]

Przedstawiony schemat przybliża poglądowo w jaki sposób działał prototyp spryskiwacza. Urządzeniami wejścia w tym przypadku są czujniki temperatury, wilgotności gleby i tym podobne.

Natomiast urządzeniami wyjścia będzie przekaźnik podłączony z pompą i jej zasilaniem oraz wyświetlacz OLED, na którym będą wyświetlane podglądowe dane. W pamięci zawarta będzie logika programu i sposób obsługi urządzenia wejścia-wyjścia.

4.3.2. Moduł ESP32

To wydajny, tani oraz o niskim poborze energii układ pozwalający na budowę urządzeń komunikujących się przez dwutrybowe 2,4Ghz WiFi oraz Bluetooth. Popularność tego modułu z biegiem czasu rośnie również z powodu małych rozmiarów, które wynoszą 55x28x8mm. ESP-WROOM-32 [17] to układ rodzaju SoC(*ang. System on a chip*), a jego producent to firma z chińskimi korzeniami o nazwie Espressif Systems. Ma 2-rdzeniowy procesor taktowany z częstotliwością do 240 Mhz. Ma 38 pinów w tym 18 pinów, które są potrzebne do mierzenia napięcia, aby odczytywać wartości z czujników. Z powodu ilości analogowych pinów ESP32 ma dużą przewagę nad ESP8266, dlatego ten moduł został wybrany do projektu. Ilość czujników wymaga minimum trzech analogowych pinów. Do tego wyróżnia się zawartością Bluetooth 4.2 i wsparciem dla BLE (*ang. Bluetooth Low Energy*). Model, który został wykorzystany to DevKit v1. „Wewnętrzna pamięć Flash modułu esp32 jest zorganizowana w pojedynczym obszarze Flash zawierającym strony po 4096bajtów każda. Dostępne są wersje 4MB, 8MB oraz 16MB. Flash zaczyna się pod adresem 0x00000, ale wiele obszarów jest zarezerwowanych dla pakietu SDK IDF ESP32. Istnieją dwa różne układy oparte na obecności obsługi BLE” [18]. Zasilac ESP32 DevKit v1 można za pomocą złącza USC Micro B lub prosto przez pin VIN. Preferowane zasilanie zaczyna się od 7 wolt a kończy na 12 woltach. Płytkę może działać na zasilaniu do 20 wolt jednak grozi to przegrzaniem regulatora napięcia i nieodwracalnym popsuciem urządzenia. Programowanie umożliwia układ szeregowy USB, za pomocą którego możliwe jest również otwieranie UART i odpowiedzialny jest za to konwerter. PWM jest dostępny na każdym cyfrowym pinie. Mikrokontroler nie wspiera ICU. Moduł ma dwie diody. Pierwsza sygnalizuje czy płytkę podłączona jest do zasilania a drugą możemy uruchamiać za pomocą oprogramowania. Płytki zależnie od wersji mają wbudowane anteny w postaci ścieżki lub mają złącze antenowe. Męskie piny zlokalizowane są na bokach płytki i ustawione są szeregowo. Z dodatkowych funkcji, które ma płytkę to czujnik Halla oraz obsługa interfejsu dotykowego. Dostępne zabezpieczenia Wifi to WEP, WPA/WPA2, PSK/Enterprise, AES/SHA2/Elliptical Curve Cryptography/RSA-4096 [19].



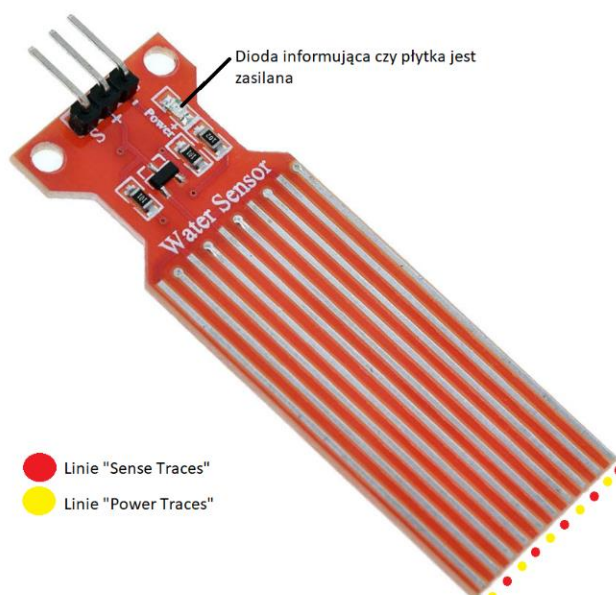
Rysunek 4.6. Schemat pinów oraz zastosowanie przycisków ESP32 DevKit v1 [18]

Tabela 4.1. Porównanie ESP32 do ESP8266 [20]

	ESP32	ESP8266
Procesor CPU	Xtensa single/dual core 32-bit LX6, taktowany do 240MHz	Tensilica L106 32-bit, taktowany do 160MHz
Pamięć RAM	520 kb SRAM + 16kB SRAM in RTC	150kB w tym 8kB user-data RAM
Pamięć ROM	448 KB + dodatkowa pamięć programu w postaci zewnętrznej pamięci SPI Flash, do 16 MB	Brak pamięci wbudowanej, pamięci programu w postaci zewnętrznej pamięci SPI flash, do 16MB
Porty GPIO	34	
Wi-Fi	802.11 b/g/n up to 150Mbps	802.11 b/g/n up to 72,2 Mbps
SPI/I ² C/I ² S/UART/PWM	Tak	Tak
Dodatkowe peryferia	ADC 12-bitowy, 18 kanałów, DAC 10-bitowy, 2 kanały, Touch Sensor 10-kanałowy, Wbudowany moduł Bluetooth (wsparcie dla BLE), Blok ULP – Ultra Low Power	ADC 10-bitowy, 1 kanał

4.3.3. Czujnik poziomu wody

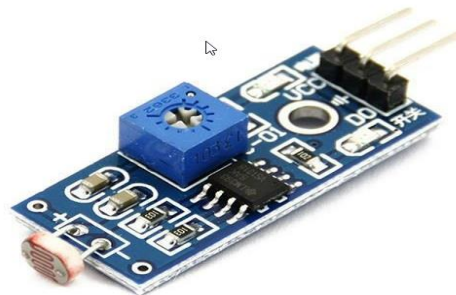
Czujnik poziomu wody ma dziesięć odsłoniętych przeplatających się miedzianych linii. Pięć z nich to linie „Sense Traces” a następne pięć to „Power Traces” [21]. Na płycie znajduje się również dioda LED, która świeci się, gdy sensor jest zasilany. Szereg odsłoniętych równoległych przewodników działa razem jak rezystor zmienny, którego rezystancja zmienia się w zależności od poziomu wody w zbiorniku. Im więcej zanurzony jest czujnik tym lepsze będzie przewodnictwo i niższy opór. Czujnik wytwarza napięcie wyjściowe zgodnie z rezystancją. Podsumowując, opór jest odwrotnie proporcjonalny do wysokości wody [22].



Rysunek 4.7. Czujnik poziomu wody [23]

4.3.4. Fotorezystor

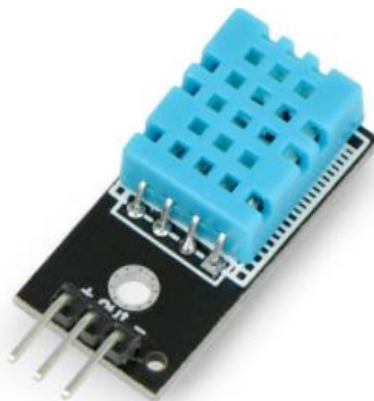
Moduł cyfrowego czujnika światła jest wyczulony na zmianę natężenia światła otoczenia, w którym się znajduje. Fotorezystor znany jest również pod nazwą Light Dependent Resistor (LDR) [24]. Używany do wykrywania jasności i natężenia światła. Za pomocą potencjometru możliwe jest regulowanie wrażliwości detekcji sygnału. Posiada cztery piny i 2 diody LED, które pokazują stan zasilania i wyjścia cyfrowego. Czujnik jest zasilany napięciem od 3,3V do 5V .



Rysunek 4.8. Fotorezystor [25]

4.3.5. Czujnik temperatury i wilgotności powietrza

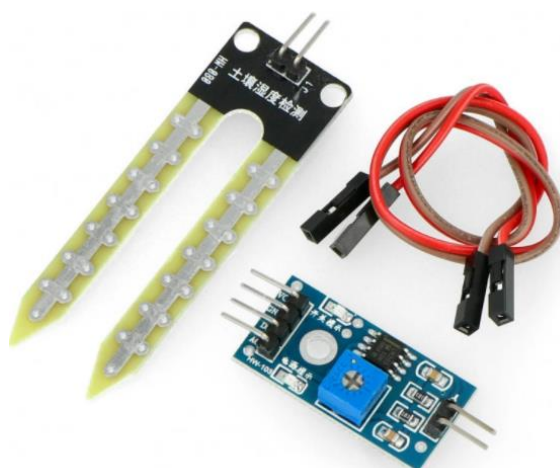
W DHT11 pomiar wilgotności powietrza jest możliwy dzięki dwóm wbudowanym elektrodom i powierzchni pomiędzy nimi zatrzymującej wilgoć [26]. Zmiana rezystancji między dwiema elektrodami jest odwrotnie proporcjonalna do wilgotności względnej. Oszacowanie temperatury jest natomiast możliwe dzięki rezystorowi termicznemu, który zmienia swoją rezystancję wraz z temperaturą.



Rysunek 4.9. Czujnik temperatury i wilgotności powietrza [27]

4.3.6. Czujnik wilgotności gleby

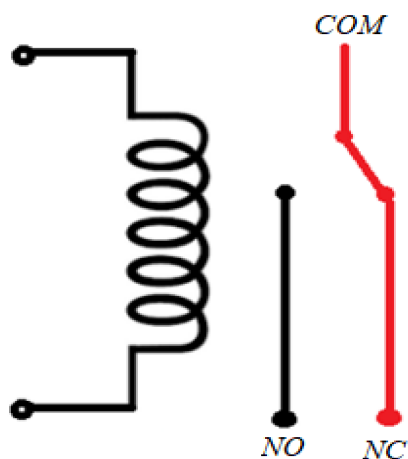
Sonda widelkowa z dwoma odsłoniętymi przewodami działa jak rezystor zmienny, którego rezystancja zmienia się w zależności od zawartości wody w glebie [28]. Opór jest odwrotnie proporcjonalny do wilgotności gleby. Im więcej wody w glebie tym lepsza przewodność i niższy opór. Czujnik zawiera również moduł, który wytwarza napięcie wyjściowe zgodnie z rezystancją, dzięki niemu możemy określić poziom wilgotności.



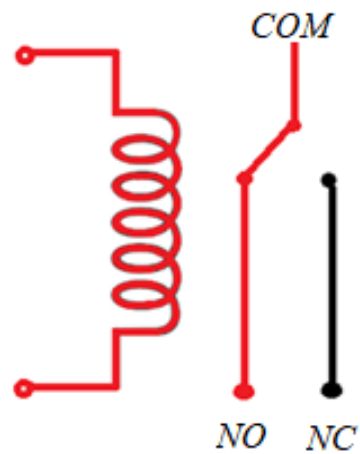
Rysunek 4.10. Czujnik wilgotności gleby [29]

4.3.7. Pompa wody 5V i przekaźnik 1-kanalowy

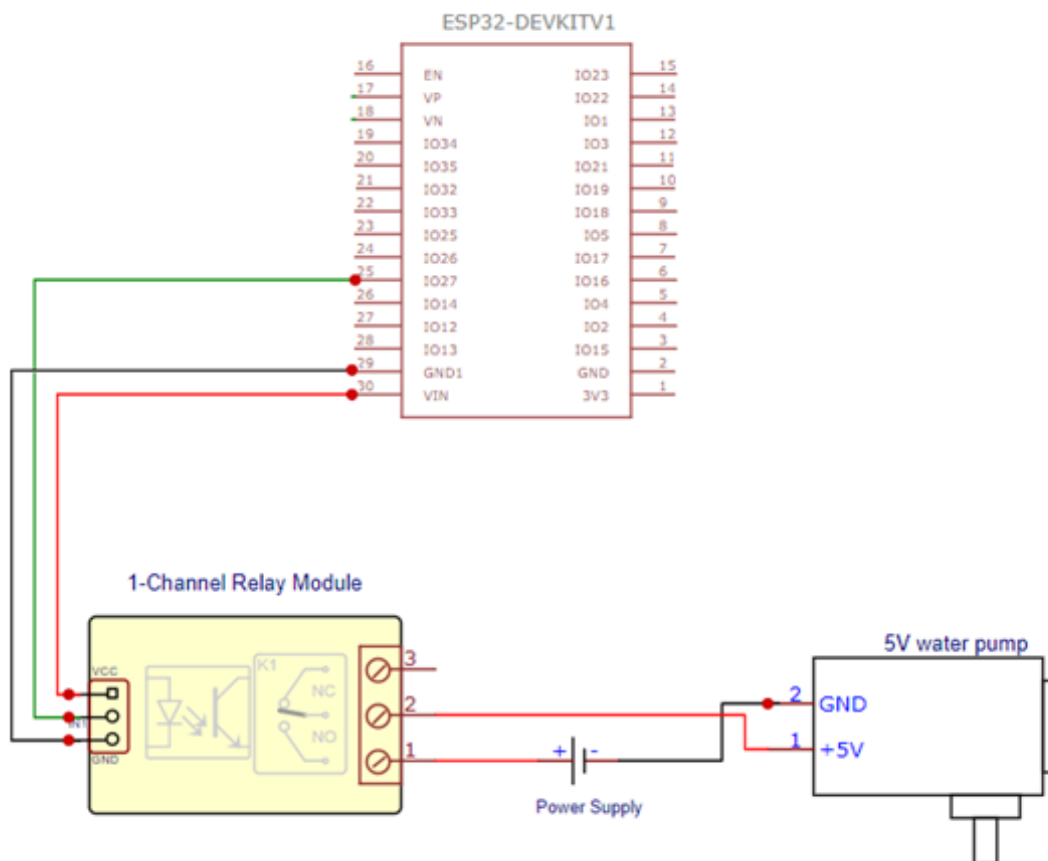
Jeśli podłączy się do pompy zasilanie 5V, pompa zacznie działać. Można programowo sterować pompą za pomocą ESP32 natomiast w tym celu potrzebny jest przekaźnik pomiędzy płytką, a pompą. To rozwiązanie daje możliwość zasilania pompy niezależnie od zasilania płytki. Użyty przekaźnik ma w sobie cewkę. Jej działanie przybliżone jest do działania elektromagnesu, który przyciąga do siebie styk w drugim obwodzie co umożliwia przepływ znacznie większego prądu przed drugi obwód. Aby aktywować pole magnetyczne, które wytwarza elektromagnes musi przepłynąć przez pierwszy obwód mały prąd. Mały prąd w pierwszym obwodzie idzie od ESP32. W momencie, gdy prąd kończy przepływać przez cewkę styk ponownie łączy się z zaciskiem normalnie zamkniętym. Ponieważ w układzie przewidziana jest tylko jedna pompa wody wystarczy przekaźnik o jednym kanale natomiast na rynku można spotkać moduły z większą ilością kanałów.



Rysunek 4.11. Elektromagnes nie został aktywowany



Rysunek 4.12. Prąd przepływa przez cewkę i elektromagnes został aktywowany



Rysunek 4.13. Układ związany z pompą wody wykonany przy użyciu portalu Easyseda [30]



Rysunek 4.14. Pompa wody 5V [31]

Przełącznik ma dwie diody LED pokazujące stan przełącznika. Aby dioda zasilania zaświeciła się potrzebny jest sygnał zasilający z ESP32, natomiast dioda stanu zaświeci się w przypadku, gdy przełącznik zostanie aktywowany.



Rysunek 4.15. Przełącznik 5V jednokanałowy [32]

4.3.8 Ekran OLED

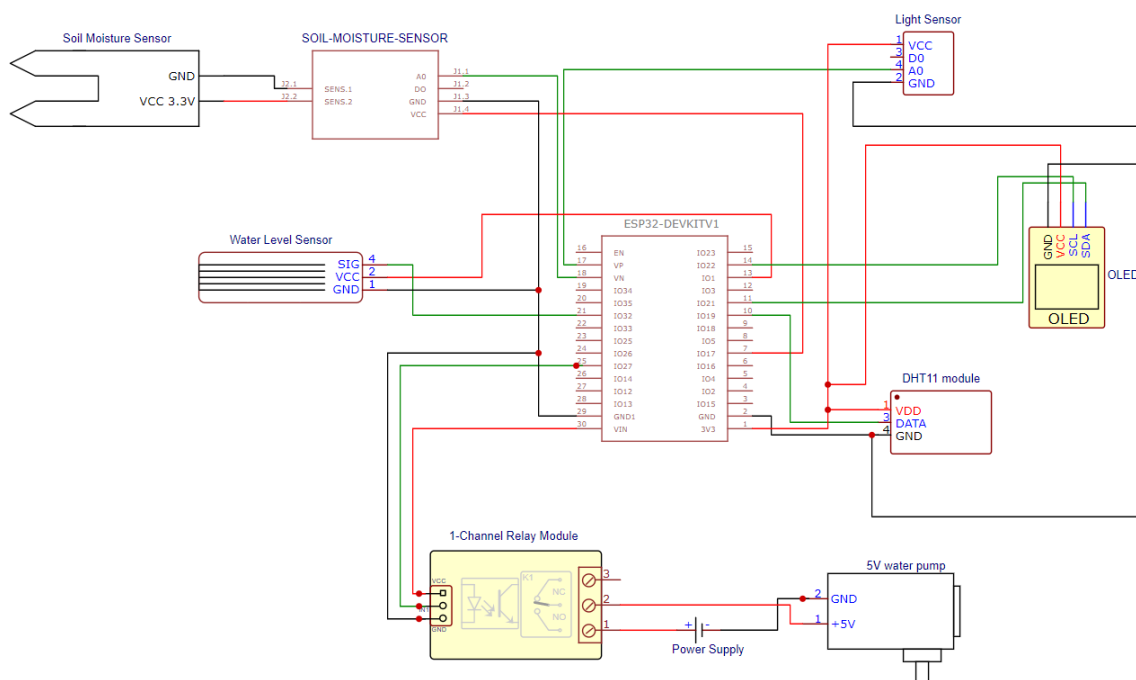
Sercem modułu jest jednokładowy sterownik CMOS OLED [33]. Sterownik użyty do tego ekranu to SH1106 lub SSD1306. Może komunikować się z mikrokontrolerem na wiele sposobów, w tym I2C i SPI. Jego rozmiar to 0,96" a rozdzielczość wynosi 128x64. Biblioteka użyta do oprogramowania tego wyświetlacza, aby wyświetlał dane z czujników

to Adafruit. Maksymalny pobór mocy wynosi 0,08W. Wymagane zasilanie jest w zakresie 3-5V. Dzięki technologii OLED pola na wyświetlaczu pełniące rolę tła są intensywnie czarne, natomiast piksele wyświetlające zawartość są koloru niebieskiego. Na rynku można również spotkać wersje z białymi pikselami. Wymiary ekranu to 26x24 mm a płytki to 28x27mm. Pin GND to masa, VCC odpowiedzialny jest za zasilanie modułu(3.3/5V),SCK to linia zegarowa natomiast SDA to linia danych.



Rysunek 4.16. Wyświetlacz OLED 0,96" [34]

4.4. Schemat Arduino



Rysunek 4.17. Schemat układu z ESP32 wykonany przy użyciu portalu Easyeda [30]

4.5. Zapytania HTTPS

Za pomocą funkcji, które wykorzystują zapytania HTTPS wysyłane są do platformy i odbierane z niej dane. Dzięki HTTPS POST uruchamiane są wbudowane w platformę funkcje. Pierwsza z nich to „CreateThing”, która jest odpowiedzialna za tworzenie na platformie „Thingów” spryskiwacza i routera, które mają realne odzwierciedlenie w rzeczywistości. Nazwa tworzonych spryskiwaczy jest unikalna, ponieważ jest tworzona na podstawie trzech ostatnich znaków adresu MAC płyty ESP32. Używane są również „EnableThing” i „RestartThing” umożliwiające w pełni uruchomienie synchronizacji pomiędzy platformą a urządzeniem, które wysyła zapytania. HTTPS POST umożliwia również uruchamianie funkcji, które zostały napisane w szablonach funkcji, odpowiedzialnych za ustawienie lokalizacji, nazwy miasta, w którym ma się znajdować spryskiwacz. W ten sam sposób pobierane są dane z platformy, gdy wywołuje się funkcja zwracająca treść w postaci JSON z informacjami jaki stan jest przycisków w panelu użytkownika. HTTPS PUT daje możliwość ustawiania wartości parametrów „Thingów”, które później wyświetlane są w interfejsie użytkownika i analizowane pod kontem awarii. Informacje w jaki sposób napisać kod i jakie typy powinny być wybrane uzyskano na stronie [35].

```
#define WL_MAC_ADDR_LENGTH 6
String getUniqueDeviceName(String device){
    String uniqueDeviceName;
    uint8_t mac[WL_MAC_ADDR_LENGTH];
    WiFi.macAddress(mac);
    String smallMacAddress = String(mac[WL_MAC_ADDR_LENGTH - 2], HEX) +
                             String(mac[WL_MAC_ADDR_LENGTH - 1], HEX)+ "Pro";
    smallMacAddress.toUpperCase();
    if(device=="Sprinkler"){
        uniqueDeviceName = "Sprinkler-" + smallMacAddress;
    }
    else if(device=="Router") {
        uniqueDeviceName = "Router-" + smallMacAddress;
    }
    Serial.println("Device-" + uniqueDeviceName);
    return uniqueDeviceName;
}
```

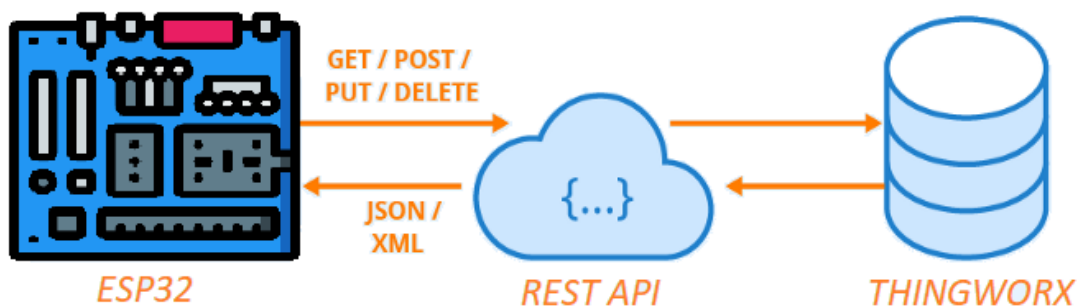
Rysunek 4.18. Funkcja ustawiająca unikalną nazwę spryskiwacza lub routera

```

////////////////////////////////////
// make HTTP PUT to ThingWorx server Thing service    //
// nameOfThing - Name of Thing                        //
// propertyName - Name of property                   //
// putBody - Body of PUT to send to ThingWorx platform //
// returns HTTP response code from server             //
////////////////////////////////////
int putToThing(String nameOfThing, String propertyName, String putBody){
    WiFiClientSecure *client = new WiFiClientSecure;
    int httpCode= -1;
    if(client) {
        client -> setCACert(rootCACertificate);
        {
            HTTPClient https;
            String fullRequestURL = String(TWPlatformBaseURL) + "/Thingworx/Things/" + nameOfThing +
            "/Properties/" + propertyName + "?appKey=" + String(appKey);
            if (https.begin(*client,fullRequestURL )) {
                https.addHeader("Accept",ACCEPT_TYPE,false,false);
                https.addHeader("Content-Type",CONTENT_TYPE,false,false);
                Serial.print("[putToThing] PUT body>");
                Serial.println(putBody);
                httpCode = https.PUT(putBody);
                if (httpCode > 0) {
                    Serial.printf("[HTTPS] PUT... code: %d\n", httpCode);
                    if (httpCode == HTTP_CODE_OK || httpCode == HTTP_CODE_MOVED_PERMANENTLY) {
                        String payload = https.getString();
                        Serial.println(payload);
                    }
                } else {
                    Serial.println(httpCode);
                    Serial.printf("[HTTPS] PUT... failed, error: %s\n", https.errorToString(httpCode).c_str());
                }
                https.end();
            } else {
                Serial.printf("[HTTPS] Unable to connect\n");
            }
        }
        delete client;
    } else {
        Serial.println("Unable to create client");
    }
    return httpCode;
}

```

Rysunek 4.19. Funkcja, która ustawią za pomocą HTTP PUT wartości parametrów



Rysunek 4.20. Komunikacja ESP32 z Thingworx[36][37]

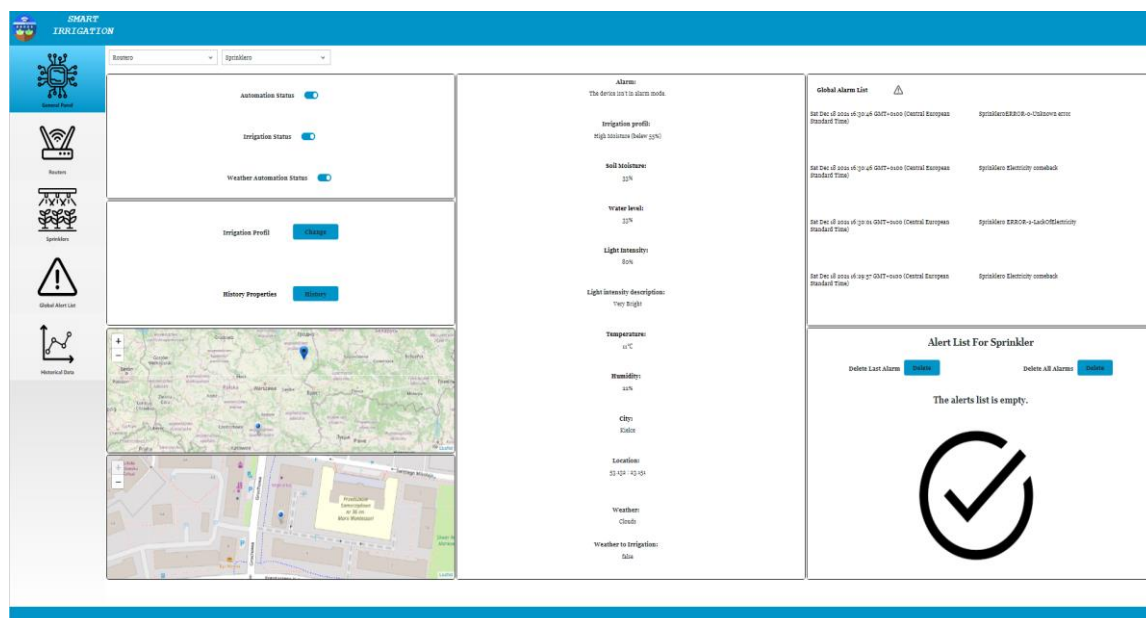
5. Projekt platformy zbudowanej w Thingworx

Założeniem platformy Thingworx to udostępnienie gotowych funkcjonalności do wykonania możliwie najlepiej skalującej się i stabilnej aplikacji. Aplikacja odpowiedzialna za sterowanie spryskiwaczami musi mieć możliwość dodania dużej ilości spryskiwaczy w różnych miejscach na ziemi. Spryskiwacze powinny być uporządkowane w listach bazujących na lokalizacji lub przypisane do routerów, które umożliwiają im połączenie z Internetem.

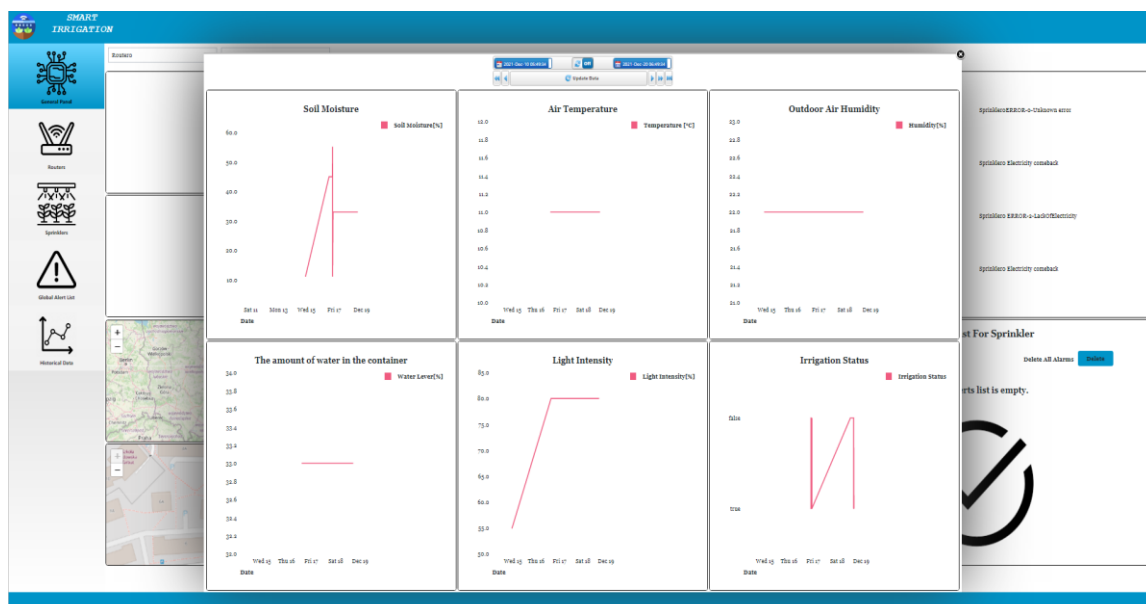
5.1. Interfejs użytkownika

Do nawigowania po stronie zaprojektowane zostało menu główne. Cztery kafelki po naciśnięciu przenoszą użytkownika do preferowanych podstron. W głównym panelu zawarta jest lista, gdzie wyświetlają się nazwy dodanych na platformę routerów i po wybraniu jednej z dostępnych pozycji w drugiej liście obok wyświetlane są przyporządkowane do routera spryskiwacze. Dane w mniejszych oknach pobierane są z spryskiwacza, który został wybrany w rozwijanej liście. W pierwszym oknie pod rozwijanymi listami zawarte są trzy przyciski ustawiające stany czy ma być włączone nawadnianie lub automatyzacja podlewania oraz funkcjonalność sprawdzania pogody, która jest ustalana po nazwie miejscowości opartej na lokalizacji. Drugie okno zawiera dwa przyciski, które odpowiedzialne są za włączenie podokien, w których ustalany jest profil nawadniania lub jest dostęp do historii danych w wybranym okresie czasowym. Pod drugim oknem ukazuje się mapa, gdzie wyświetlają się pinezki w miejscach lokalizacji routerów oraz po wybraniu jednej z nich druga mapa skaluje się do miejsca, gdzie są spryskiwacze dla danego routera. Środkowy panel ukazuje parametry spryskiwacza. Dostępne są tam takie informacje jak: obecny stan wilgotności gleby, temperatura powietrza, obecna pogoda w danym mieście, lokalizacja. Ostatnie widniejące dwa parametry informują czy potrzebne jest nawadnianie po uwzględnieniu pogody. Zawarte są również dwa panele odpowiedzialne za wyświetlanie globalnej listy alarmów oraz listy alarmów dla danego spryskiwacza. W liście alarmów obecnie analizowanego spryskiwacza jest możliwość usunięcia ostatniego wiersza lub zresetowania całej listy. Gdy lista jest pusta, użytkownik widzi stosowny komunikat i ma możliwość korzystania z wszystkich funkcji. W przypadku gdy pojawi się alarm automatycznie wszystkie przyciski od sterowania spryskiwaczem są dezaktywowane, ich kolor jest w odcieniach szarości i wyłączane jest nawadnianie. Po usunięciu alarmu przyciski ponownie są aktywowane i użytkownik ma

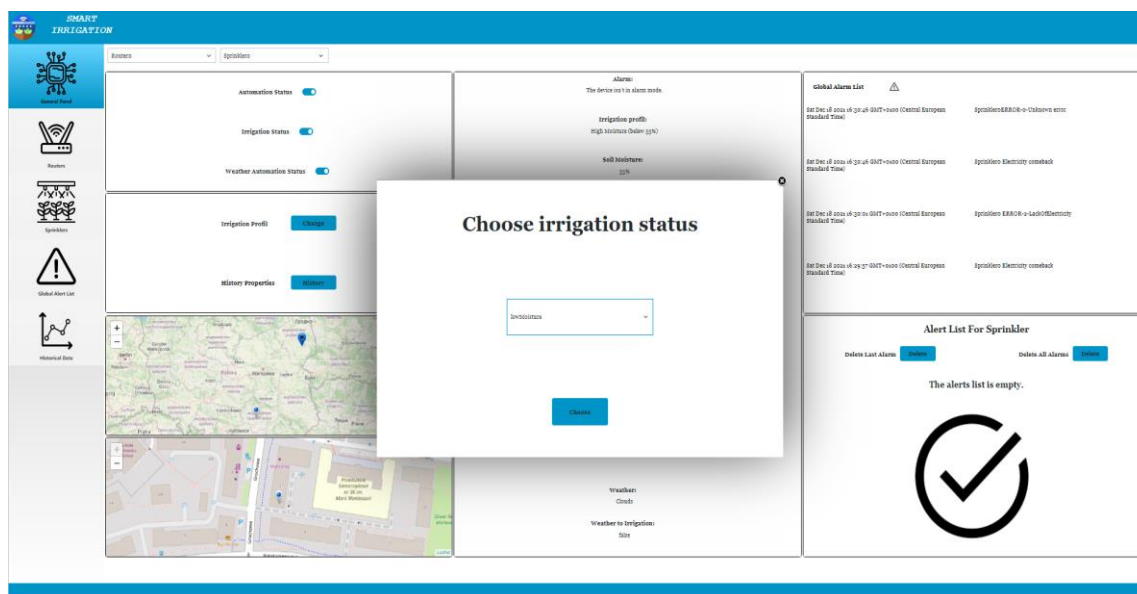
możliwość sterowaniem ich stanem. Źródło ikon wykorzystanych do tworzenia interfejsu użytkownika to strona internetowa [38]. Wykorzystane zostały dwie zewnętrzne biblioteki (Mail-Extensions, OpenStreetMap) od PTC [5]. Dostępne są po założeniu konta. Były potrzebne do zrobienia funkcjonalności wysyłania mailów oraz tworzenia map.



Rysunek 5.1. Główny panel systemu

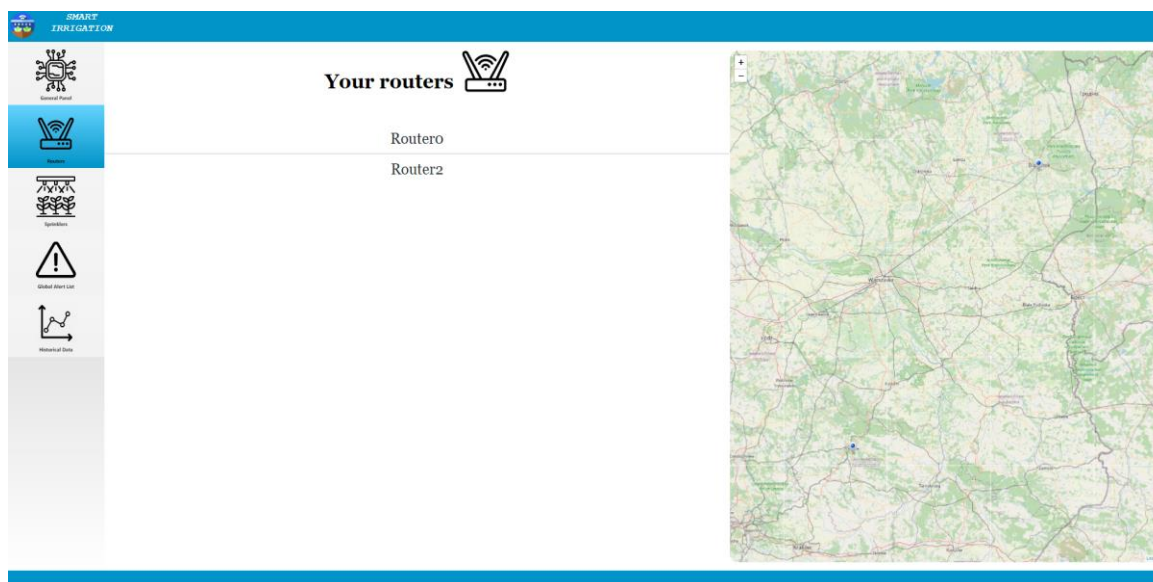


Rysunek 5.2. Okno ukazujące się po naciśnięciu przycisku „History”



Rysunek 5.3. Okno ukazujące się po naciśnięciu przycisku „Change”

Na Panelu zawarta jest lista dostępnych routerów z funkcją przeniesienia do panelu przyporządkowanych do nich spryskiwaczy. Taką samą funkcjonalność przeniesienia ma mapa z pinezkami umieszczonymi według lokalizacji routerów.

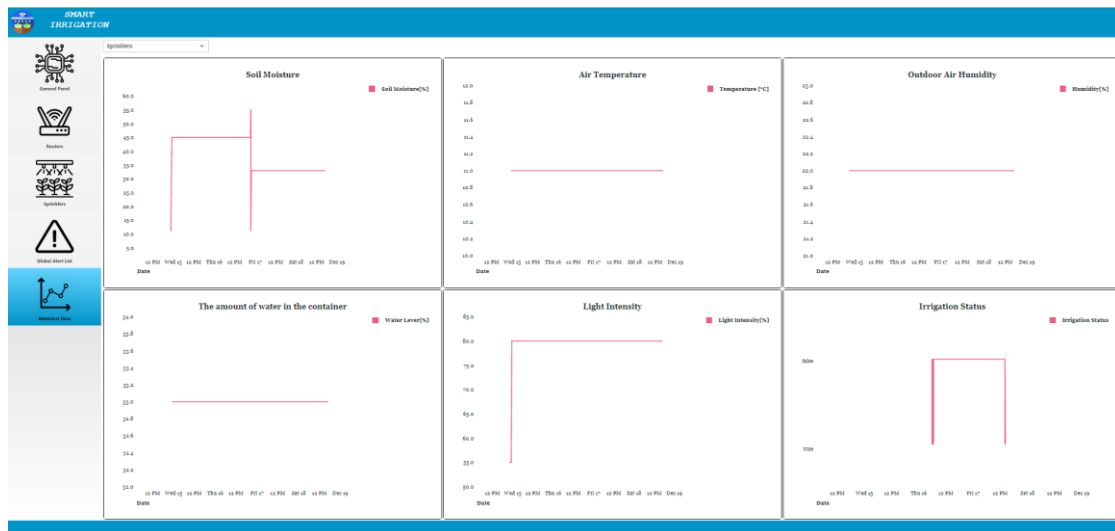


Rysunek 5.4. Panel Routerów

- wilgotność gleby,
- ilość wody w zbiorniku,
- temperatura i wilgotność powietrza, aby użytkownik miał dostęp do tych danych po najmniejszej ilości kliknięć myszki.

[illegible]

Rysunek 5.6. Panel globalnej listy alarmów



5.2. Zabezpieczenia przed awarią

Na platformie Thingworx zaimplementowane zostały subskrypcje, które w realnym czasie sprawdzają parametry urządzeń. W przypadku wystąpienia progowych wartości uruchamiają alarm i wyłączają możliwość uruchomienia spryskiwaczy. Automatycznie wysyłana jest również wiadomość mailowa z informacją, które urządzenie się popsło oraz kod błędu. Warunki wystąpienia alarmu:

- brak elektryczności,
- brak wody z zbiorniku,
- wilgotność gleby powyżej 98%,
- temperatura powietrza poniżej 0°C.

Warning !

Irrigation System Alert

Hi ,

Your undefined has problem with Sprinkler0 : ERROR-2-LackOfElectricity

[Check on platform](#)

We hope it's nothing serious.

Sprinklers technologies

Rysunek 5.8 Powiadomienie o awarii

5.3. Monitorowanie pogody

Cyklicznie co godzinę włączany jest serwis pogodowy pobierający dane z zewnętrznego API [39]. Harmonogram ustalany jest po Cron, programie opartym na czasie i służy do uruchamiania serwisów w odpowiednich odstępach czasowych. Podając lokalizację jako parametr zapytanie zwraca nam aktualną pogodę w danym mieście. Jeśli wynik to:

- deszczowo,
- ulewa,
- śnieg,
- burza z piorunami,

to serwis ustala, iż nie jest wymagane nawadnianie.

```
2 let params = {
3   url: "http://api.openweathermap.org/data/2.5/weather?q="+city+"&appid=ff 6969ad8 9d 1019450 c363 cc98a" /* STRING */;
4 };
5 let result = Resources["ContentLoaderFunctions"].getJSON(params);
6 result=result.weather[0].main;
7
```

Rysunek 5.9. Zapytanie do serwisu pogodowego

```
2 var result1= me.WeatherApi({city:me.city});
3 me.weather=result1;
4 var result= false;
5 if( result1 == "Rain" || result1=="Shower rain" || result1 == "Snow" || result1=="Thunderstorm"){
6   me.weatherToIrrigation=false;
7   result= me.weatherToIrrigation;
8 }
9 else {
10   me.weatherToIrrigation=true;
11   result= me.weatherToIrrigation;
12 }
```

Rysunek 5.10. Serwis pogodowy

Seconds	<input type="text" value="0"/>	allowed values: 0-59, -, *, /
Minutes	<input type="text" value="0/1"/>	allowed values: 0-59, -, *, /
Hours	<input type="text" value="1"/>	allowed values: 0-23, -, *, /
Day of Month	<input type="text" value="*"/>	allowed values: 1-31, -, * ?, / L W
Month	<input type="text" value="*"/>	allowed values: 1-12 or JAN-DEC, -, *, /
Day of Week	<input type="text" value="?"/>	allowed values: 1-7 or SUN-SAT, -, * ?, / L #
Year	<input type="text"/>	allowed values: empty or 1970-2099, -, *, /

CRON String
0 0/1 1 * * ?

Rysunek 5.11. Harmonogram Cron

6. Weryfikacja oraz testowanie aplikacji

Weryfikacja i testowanie aplikacji to zbiór zdjęć ukazujących sposób symulacji oraz zrzuty ekranu z „Serial Monitor” w Arduino IDE, które potwierdzają zmianę wartości parametrów i stanu podlewania.

6.1. Weryfikacja działania czujnika wilgotności gleby

Symulacja działania czujnika wilgotności gleby została przeprowadzona poprzez zanurzenie go w pojemniku z wodą. Im głębiej zostawał zanurzonym tym wartość napięcia wyjściowego zgodnie z rezystancją była mniejsza. Za pomocą mapowania oprogramowanie konwertowało tą wartość na procenty.



Rysunek 6.1. Czujnik wilgotności wyjęty z wody



Rysunek 6.2. Czujnik wilgotności włożony do wody


```
int soilMoisture = readSoilMoisureSensor();
int soilMoisturePercent = map(soilMoisture, 4095, 250, 0, 100);
```

Rysunek 6.3. Wartość z czujnika wilgotności mapowana, aby wyświetlić ją w procentach

```
Serial.print("Soil Moisture Sensor Value: ");
Serial.println(soilMoisture);
Serial.print("Soil Moisture Percent Map: ");
Serial.print(soilMoisturePercent);
Serial.println("%");
```

Rysunek 6.4. Kod odpowiedzialny za wyświetlanie wartości z czujnika wilgotności w serial monitorze

```
Soil Moisture Sensor Value: 4095
Soil Moisture Percent Map: 0%
```

Rysunek 6.5. Komunikat w serial monitorze, gdy czujnik wilgotności był wyjęty z wody

```
Soil Moisture Sensor Value: 1195
Soil Moisture Percent Map: 75%
```

Rysunek 6.6. Komunikat w serial monitorze, gdy czujnik wilgotności był włożony do wody

6.2. Test automatycznego podlewania

Gdy na platformie włączona jest opcja automatycznego podlewania i spryskiwacz wysłał informację, że wilgotność wynosi około 75% przycisk informujący o stanie podlewania jest wyłączony.



Rysunek 6.7. Panel użytkownika z włączoną automatyzacją

W tym samym czasie platforma wysłała informację, że opcja automatyzacji jest włączona i spryskiwacz po ustaleniu, że wilgotność jest ponad wartością minimalną analizuje, że pompa wody jest wyłączona i obecne warunki nie wymagają jej uruchomienia. Spryskiwacz następnie wysłał zwrótną informację na platformę.


```
[HTTPS] GET... code: 200
{"weatherAutomationStatus":false,"irrigationStatus":false,"profileStatus":"mediumMoisture"}
Soil Moisture: 52%
Pomp deactivated...
[putToThing] PUT body>{"irrigationStatus" : "0"}
[HTTPS] PUT... code: 200
```

Rysunek 6.8. Komunikat o wyłączonej pompie w serial monitor

Gdy przyjdzie informacja na platformę, że wilgotność spadła poniżej progowej wartości ustalonej w profilu podlewania i pompa wody jest włączona to przycisk związany z podlewaniem zmienia kolor na niebieski i uruchamia się. W przedstawionym przypadku minimalną wartością było 45% wilgotności gleby i pompa działała aż wartość wzrośnie do 55%.



Rysunek 6.9. Panel użytkownika, gdy włączy się pompa

Gdy czujnik wilgotności gleby wykrył, że wartość wynosi 11% automatycznie włączył pompę wody i poinformował o tym platformę za pomocą HTTPS PUT do odpowiadającego parametru na platformie.

```
[HTTPS] GET... code: 200
{"weatherAutomationStatus":false,"irrigationStatus":true,"profileStatus":"mediumMoisture"}
Soil Moisture: 11%
Pomp activated...
[putToThing] PUT body>{"irrigationStatus" : "1"}
[HTTPS] PUT... code: 200
```

Rysunek 6.10. Komunikat o włączonej pompie w serial monitor

6.3. Weryfikacja czujnika temperatury

Pierwsza sprawdzana wartość czujnika to temperatura otoczenia. Sprawdzenie czy sensor reaguje na zmianę warunków polegało na przyłożeniu do niego obiektu, uprzednio schłodzonego w zamrażarce.



Rysunek 6.11. Symulacja zmniejszenia temperatury



Rysunek 6.12. Czujnik temperatury DHT 11 znajdujący się w obudowie, aby sprawdzać temperaturę otoczenia.

```
float tempC = dht_sensor.readTemperature();
```

Rysunek 6.13. Funkcja odpowiedzialna za odczytanie wartości z czujnika temperatury

```
Serial.print("Temperature: ");  
Serial.print(tempC);  
Serial.println("°C");
```

Rysunek 6.14. Kod odpowiedzialny za wyświetlanie wartości z czujnika temperatury w serial monitorze

Temperature: 23.80°C

Rysunek 6.15. Wartość z czujnika temperatury otoczenia w serial monitorze

Temperature: 12.20°C

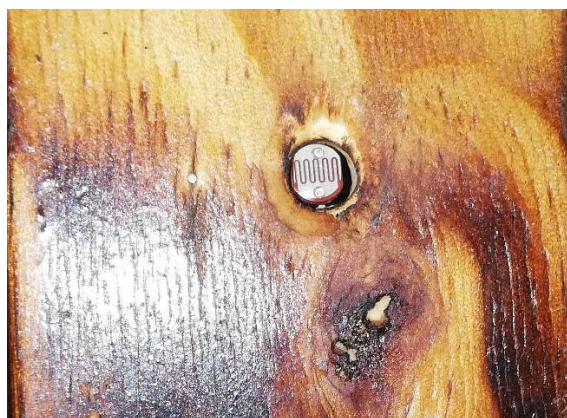
Rysunek 6.16. Wartość z czujnika temperatury w serial monitorze po przyłożeniu zamrożonego obiektu

6.4. Weryfikacja rezystora wrażliwego na światło

Sprawdzanie diody LDR polegało na sprawdzeniu wartości intensywności światła z przyłożoną do niej wyłączoną i włączoną latarką.



Rysunek 6.17. Czujnik LDR bez symulowania jaśniejszego otoczenia



Rysunek 6.18. Czujnik LDR podczas symulowania jaśniejszego otoczenia za pomocą latarki

```
int lightIntensity = analogRead(LIGHT_SENSOR_PIN);
int lightIntensityPercent = map(lightIntensity, 4095, 0, 0, 100);
```

Rysunek 6.19. Wartość z czujnika LDR mapowana, aby wyświetlić ją w procentach

```
Serial.print("LDR Sensor Value: ");
Serial.println(lightIntensity);
Serial.print("Light Intensity Percent Map:");
Serial.print(lightIntensityPercent);
Serial.println("%");
```

Rysunek 6.20. Kod odpowiedzialny za wyświetlanie wartości z czujnika LDR w serial monitorze

```
LDR Sensor Value: 2585
Light Intensity Percent Map:37%
```

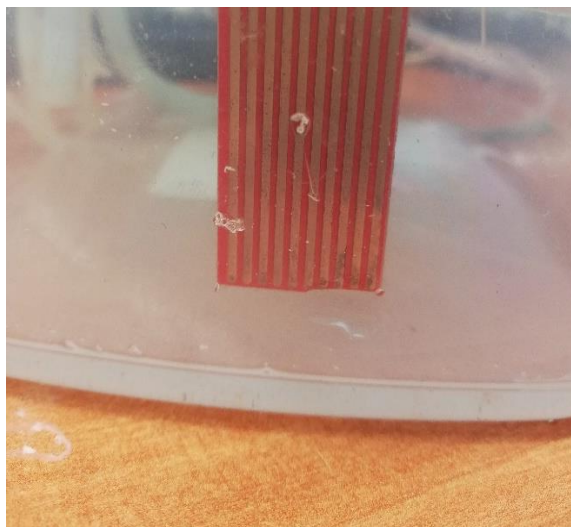
Rysunek 6.21. Wartość z czujnika LRD bez symulowania intensywniejszego światła latarką

```
LDR Sensor Value: 65
Light Intensity Percent Map:98%
```

Rysunek 6.22. Wartość z czujnika LRD z włączoną latarką

6.5. Weryfikacja czujnika poziomu wody w zbiorniku

Czujnik poziomu wody znajduje się w szczelnym zbiorniku. Testowanie polegało na sprawdzeniu wartości rezystancji, gdy pojemnik jest pusty oraz gdy wlana zostanie do niego woda, która będzie na takim poziomie, aby mieć styczność z czujnikiem.



Rysunek 6.33. Brak wody w pojemniku



Rysunek 6.34. Pojemnik wypełniony wodą

```
int waterLevel = readWaterLevelSensor();
int waterLevelPercent = map(waterLevel, 0, 1100, 0, 100);
```

Rysunek 6.35. Wartość z czujnika poziomu wody mapowana, aby wyświetlić ją w procentach

```
Serial.print("Water Level Sensor Value: ");
Serial.println(waterLevel);
Serial.print("Water Level Percent Map: ");
Serial.print(waterLevelPercent);
Serial.println("%");
```

Rysunek 6.36 Kod odpowiedzialny za wyświetlanie wartości z czujnika poziomu wody w serial monitorze

```
Water Level Sensor Value: 0
Water Level Percent Map: 0%
```

Rysunek 6.37. Wartość z czujnika w serial monitorze, gdy pojemnik nie był wypełniony wodą

```
Water Level Sensor Value: 216
Water Level Percent Map: 20%
```

Rysunek 6.38. Wartość z czujnika w serial monitorze, gdy do pojemnika była wlana woda

6.6. Weryfikacja ekranu OLED

Ekran poprawnie wyświetla dane pobrane przez czujniki, które są przekształcone na skalę procentową z wyjątkiem temperatury powietrza.

```
Soil M:22%
Light:100%
Temp:20.20°C
Humidity:36.00%
Water lv:0%
```

Rysunek 6.39. Obraz z ekranu OLED

7. Możliwość rozwoju

Platforma przystosowana jest do możliwości łatwego rozszerzania jej funkcjonalności. Dobrym pomysłem będzie oddzielenie urządzeń do zastosowania domowego oraz do uprawy pól uprawnych do osobnych kategorii. Idąc tym tokiem spryskiwacz do użytku domowego nie musiałby wtedy mieć funkcji sprawdzania pogody czy diody LDR co przyczyniłoby się do oszczędności energii. Jego rozmiary powinny być jak najbardziej kompaktowe. Płytką powinna być zespolona z wszystkimi podzespołami tworząc z nimi jedną całość bez potrzeby kablowego połączenia.

7.1. Rozwój prototypu spryskiwacza

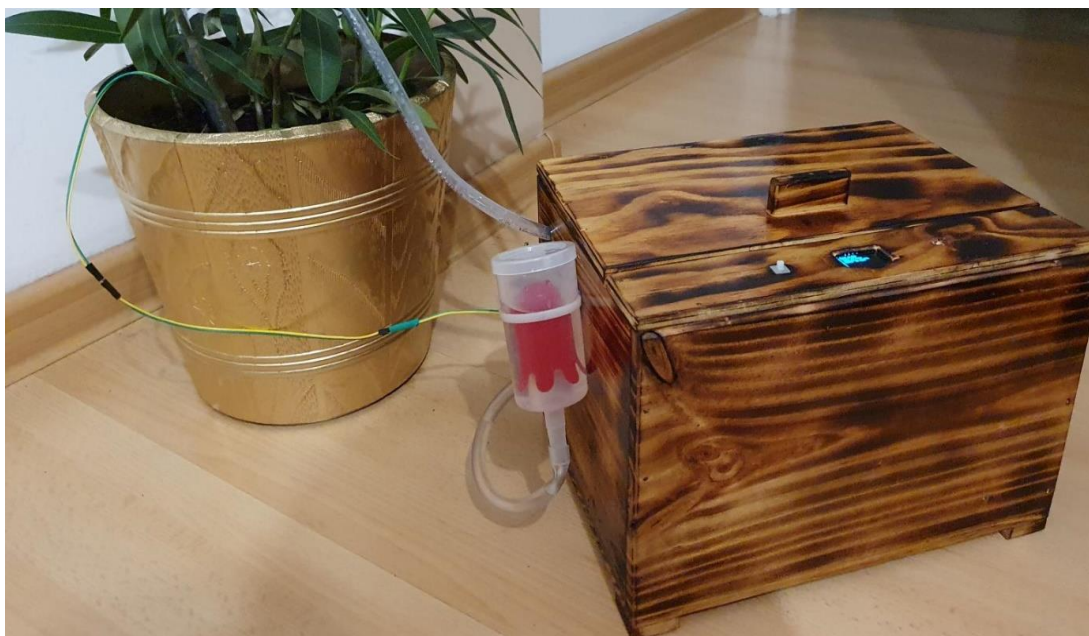
Prototyp spryskiwacza potrzebuje dodatkowego modułu GSM, aby móc komunikować się z routerami na dużych odległościach oraz modułu GPS, aby automatycznie ustalać swoje położenie, po tej wartości ustalać miasto i przekazywać tą wartość do zewnętrznego API sprawdzającego pogodę. Wszystkie czujniki powinny być zastąpione na lepsze jakościowo, aby poprawić dokładność pomiarów. Zastosowany czujnik poziomu wody powinien być odporny na długotrwały kontakt z wodą, ponieważ obecny śniedzieje po pewnym czasie. W oprogramowaniu ESP32 powinno również być uwzględnione włączanie trybu oszczędzania energii. Z powodu braku środków również jego obudowa jest wykonana z drewna. W przypadku chęci poprawy wyglądu urządzenia, aby dawało odczucie bardziej nowoczesnego i innowacyjnego obowiązkowe będzie zaprojektowanie modelu obudowy, który mogłaby wydrukować drukarka 3D.

7.2. Rozwój platformy

Pożądane jest dodanie dodatkowych zabezpieczeń związanych z bezpieczeństwem. Aplikacja powinna brać pod uwagę każdą okoliczność, gdy może wystąpić awaria i skutecznie zareagować. Dodatkowym krokiem ku ulepszeniu platformy będzie uruchomienie serwera z pasującą nazwą domeny, aby użytkownicy mogli łatwo wyszukać ją w swoich przeglądarkach. Aplikacja wymaga dodania możliwości personalizacji okien, aby użytkownik mógł jeszcze dokładniej dopasować wygląd do swoich preferencji. Z myślą o ewentualnym rozszerzeniu rynku aplikacji na inne kraje można wprowadzić obsługę innych języków, które byłyby ustawiane automatycznie w oparciu o lokalizację użytkownika.

8. Podsumowanie

Efekt końcowy w postaci działającej aplikacji jest gotowym rozwiązaniem. Można go spersonalizować pod pożądane zastosowania. Również prototyp inteligentnego spryskiwacza to produkt, którego brakuje w sprzedaży producentów urządzeń IoT. Dzięki kompatybilności płytki ESP użytkownik miałby możliwość dodania dodatkowych potrzebnych czujników, które monitorowałyby następne parametry oprócz wbudowanych. Pomysł jest innowacyjny i jeśli przy jego rozwoju znalazłby się środki w postaci większego kapitału pieniężnego na pewno mógłby się rozwinąć. Jego zastosowanie w nawadnianiu pól uprawnych byłoby nieocenione pod względem oszczędności wody i tym samym ochrony naszej planety co jest istotne biorąc pod uwagę zagrożenie jakie niesie ocieplenie klimatu co może być przyczyną powstania suszy.



Rysunek 8.1. Prototyp spryskiwacza

LITERATURA

- [1] <https://www.un.org.pl/cel6> [online, dostęp 6.12.2021r.]
- [2] <https://raportsdg.stat.gov.pl/2020/cel6.html> [online, dostęp 6.12.2021r.]
- [3] https://pl.wikipedia.org/wiki/Internet_rzeczy [online, dostęp 6.12.2021r.]
- [4] <https://support.ptc.com/help/thingworx/platform/r9/en/index.html#page/ThingWorx/Welcome.html#> [online, dostęp 6.12.2021r.]
- [5] <https://www.ptc.com/> [online, dostęp 6.12.2021r.]
- [6] G.Blokdyk „ThingWorx Third Edition” 5STARCook 2018r. [online, dostęp 7.12.2021r.]
- [7] https://support.ptc.com/help/thingworx_hc/thingworx_8_hc/en/index.html#page/ThingWorx/Help/Getting_Started/MenuInComposer/NewComposer.html [online, dostęp 7.12.2021r.]
- [8] https://support.ptc.com/help/thingworx_hc/thingworx_8_hc/en/index.html#page/ThingWorx/Help/Best_Practices_for_Developing_Applications/visualization_mashups_masters_in_applications.html [online, dostęp 8.12.2021r.]
- [9] <https://docs.arduino.cc/> [online, dostęp 10.12.2021r.]
- [10] S. Monk „Arduino dla początkujących. Podstawy i szkice”, Helion 2014r.
- [11] <https://www.tinyosshop.com/arduino-gsm-shield> [online, dostęp 14.12.2021r.]
- [12] https://pl.wikipedia.org/wiki/General_Packet_Radio_Service [online, dostęp 14.12.2021r.]
- [13] <https://www.arduino.cc/en/Guide/ArduinoGSMShield> [online, dostęp 14.12.2021r.]
- [14] P.Hoddie, L.Prader „IoT Development for ESP32 and ESP8266 with JavaScript A Practical Guide to XS and the Moddable SDK”, Apress 2020r.
- [15] <https://pl.wikipedia.org/wiki/Mikrokontroler> [online, dostęp 19.12.2021r.]
- [16] <https://app.diagrams.net/> [online, dostęp 19.12.2021r.]
- [17] <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/> [online, dostęp 20.12.2021r.]
- [18] <https://grobotronics.com/esp32-development-board-devkit-v1.html> [online, dostęp 20.12.2021r.]
- [19] <https://botland.com.pl/moduly-wifi-i-bt-esp32/8893-esp32-wifi-bt-42-platforma-z-modulem-esp-wroom-32-zgodny-z-esp32-devkit-5904422337438.html> [online, dostęp 20.12.2021r.]
- [20] <https://ep.com.pl/projekty/moduly-w-aplikacjach/12827-esp32-modul-doiot-przykladowa-aplikacja-wstacji-pogodowej> [online, dostęp 20.12.2021r.]
- [21] <https://arduinogetstarted.com/tutorials/arduino-water-sensor> [online, dostęp 22.12.2021r.]

- [22] <https://lastminuteengineers.com/water-level-sensor-arduino-tutorial/> [online, dostęp 29.12.2021r.]
- [23] <https://www.gotronik.pl/czujnik-poziomu-wody-water-sensor-p-7855.html>
[online, dostęp 23.12.2021r.]
- [24] <https://www.electroduino.com/ldr-sensor-module-how-ldr-sensor-works/> [online, dostęp 23.12.2021r.]
- [25] http://m.jselectronics.com.my/index.php?ws=showproducts&products_id=1963634
[online, dostęp 23.12.2021r.]
- [26] <https://www.instructables.com/Measuring-Humidity-Using-Sensor-DHT11/>
[online, dostęp 25.12.2021r.]
- [27] <https://botland.com.pl/czujniki-multifunkcyjne/1886-czujnik-temperatury-i-wilgotnosci-dht11-modul-przewody-5903351242448.html> [online, dostęp 25.12.2021r.]
- [28] <https://lastminuteengineers.com/soil-moisture-sensor-arduino-tutorial/> [online, dostęp 26.12.2021r.]
- [29] <https://botland.com.pl/czujniki-wilgotnosci/1588-czujnik-wilgotnosci-gleby-5904422368289.html>
[online, dostęp 27.12.2021r.]
- [30] <https://easyeda.com/> [online, dostęp 27.12.2021r.]
- [31] <https://abc-rc.pl/product-pol-11984-Mini-Pompa-do-wody-DC-3V-6V-1W-100l-h-pompka-cieczypionowa.html> [online, dostęp 27.12.2021r.]
- [32] <https://automatyka.elstat.com.pl/p34995.modul-z-1-przekaznikiem-5vdc-wyzwalnie-wysoki-poziom.html> [online, dostęp 27.12.2021r.]
- [33] <https://lastminuteengineers.com/oled-display-arduino-tutorial/> [online, dostęp 28.12.2021r.]
- [34] <https://yaboo.pl/product-pol-2351-Wyswietlacz-OLED-0-96-128x64-px-niebieski.html>
[online, dostęp 28.12.2021r.]
- [35] <https://developer.thingworx.com/en> [online, dostęp 29.12.2021r.]
- [36] <https://ichi.pro/pl/co-to-jest-restful-api-jak-tworzyc-i-uzywac-restful-api-61928290583564>
[online, dostęp 15.01.2022r.]
- [37] <https://www.flaticon.com/free-> [online, dostęp 15.01.2022r.]
- [38] <https://www.flaticon.es/> [online, dostęp 29.12.2021r.]
- [39] <http://api.openweathermap.org> [online, dostęp 29.12.2021r.]

Spis rysunków

Rysunek 1.1. Diagram przedstawiający zużycie wody w Polsce w 2019r.

Rysunek 3.1. Thingworx kompozytor

Rysunek 3.2. Thingworx Mashup

Rysunek 3.3. Środowisko Arduino IDE

Rysunek 4.1. Logika uruchamiająca pompę na podstawie wyborów użytkownika

Rysunek 4.2. Podzespoły Arduino [14]

Rysunek 4.3. Przykładowy mikrokontroler PIC18F8720 firmy Microchip Technology [15]

Rysunek 4.4. Schemat układu mikrokontrolera wykonany na stronie [16]

Rysunek 4.6. Schemat pinów oraz zastosowanie przycisków ESP32 DevKit v1 [18]

Rysunek 4.7. Czujnik poziomu wody [23]

Rysunek 4.8. Fotorezystor [25]

Rysunek 4.9. Czujnik temperatury i wilgotność powietrza [27]

Rysunek 4.10. Czujnik wilgotności gleby [29]

Rysunek 4.11. Elektromagnes nie został aktywowany

Rysunek 4.12. Prąd przepływa przez cewkę i elektromagnes został aktywowany

Rysunek 4.13. Układ związany z pompą wody wykonany przy użyciu portalu Easyeda [30]

Rysunek 4.14. Pompa wody 5V [31]

Rysunek 4.15. Przekaznik 5V jednokanałowy [32]

Rysunek 4.16. Wyświetlacz OLED 0,96" [34]

Rysunek 4.17. Schemat układu z ESP32 wykonany przy użyciu portalu Easyeda [30]

Rysunek 4.18. Funkcja ustawiająca unikalną nazwę spryskiwacza lub routera

Rysunek 4.19. Funkcja, która ustawia za pomocą HTTP PUT wartości parametrów

Rysunek 4.20. Komunikacja ESP32 z Thingworx[36][37]

Rysunek 5.1. Główny panel systemu

Rysunek 5.2. Okno ukazujące się po naciśnięciu przycisku „History”

Rysunek 5.3. Okno ukazujące się po naciśnięciu przycisku „Change”

Rysunek 5.4. Panel Routerów

Rysunek 5.5. Panel spryskiwaczy

Rysunek 5.6. Panel globalnej listy alarmów

Rysunek 5.7. Panel linowych wykresów danych

Rysunek 5.8 Powiadomienie o awarii

Rysunek 5.9. Zapytanie do serwisu pogodowego

Rysunek 5.10. Serwis pogodowy

Rysunek 5.11. Harmonogram Cron

Rysunek 6.1. Czujnik wilgotności wyjęty z wody

Rysunek 6.2. Czujnik wilgotności włożony do wody

Rysunek 6.3. Wartość z czujnika wilgotności mapowana, aby wyświetlić ją w procentach

Rysunek 6.4. Kod odpowiedzialny za wyświetlanie wartości z czujnika wilgotności w serial monitorze

Rysunek 6.5. Komunikat w serial monitorze, gdy czujnik wilgotności był wyjęty z wody

Rysunek 6.6. Komunikat w serial monitorze, gdy czujnik wilgotności był włożony do wody

Rysunek 6.7. Panel użytkownika z włączoną automatyzacją

Rysunek 6.8. Komunikat o wyłączonej pompie w serial monitor

Rysunek 6.9. Panel użytkownika, gdy włączy się pompa

Rysunek 6.10. Komunikat o włączonej pompie w serial monitor

Rysunek 6.11. Symulacja zmniejszenia temperatury

Rysunek 6.12. Czujnik temperatury DHT 11 znajdujący się w obudowie, aby sprawdzać temperaturę otoczenia.

Rysunek 6.13. Funkcja odpowiedzialna za odczytanie wartości z czujnika temperatury

Rysunek 6.14. Kod odpowiedzialny za wyświetlanie wartości z czujnika temperatury w serial monitorze

Rysunek 6.15. Wartość z czujnika temperatury otoczenia w serial monitorze

Rysunek 6.16. Wartość z czujnika temperatury w serial monitorze po przyłożeniu zamrożonego obiektu

Rysunek 6.17. Czujnik LDR bez symulowania jaśniejszego otoczenia

Rysunek 6.18. Czujnik LDR podczas symulowania jaśniejszego otoczenia za pomocą latarki

Rysunek 6.19. Wartość z czujnika LDR mapowana, aby wyświetlić ją w procentach

Rysunek 6.20. Kod odpowiedzialny za wyświetlanie wartości z czujnika LDR w serial monitorze

Rysunek 6.21. Wartość z czujnika LRD bez symulowania intensywniejszego światła latarką

Rysunek 6.22. Wartość z czujnika LRD z włączoną latarką

Rysunek 6.33. Brak wody w pojemniku

Rysunek 6.34. Pojemnik wypełniony wodą

Rysunek 6.35. Wartość z czujnika poziomu wody mapowana, aby wyświetlić

ją w procentach

Rysunek 6.36 Kod odpowiedzialny za wyświetlanie wartości z czujnika poziomu wody

w serial monitorze

Rysunek 6.37. Wartość z czujnika w serial monitorze, gdy pojemnik nie był wypełniony wodą

Rysunek 6.38. Wartość z czujnika w serial monitorze, gdy do pojemnika była wlewana woda


Rysunek 6.39. Obraz z ekranu OLED


Rysunek 8.1. Prototyp spryskiwacza

Dodatek

1. Na stronie <https://www.ptc.com/en/support>

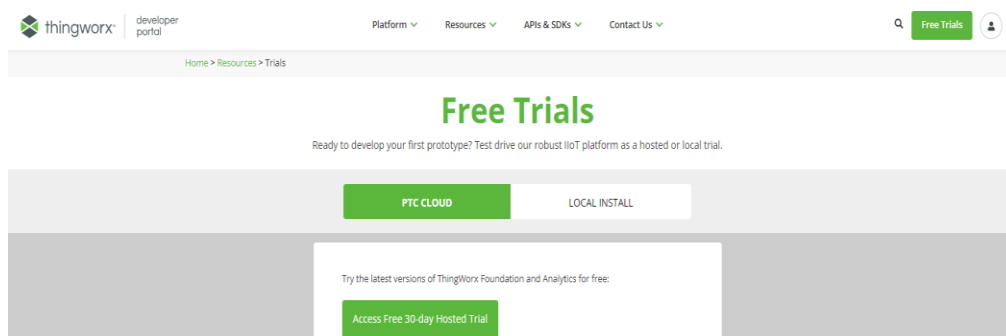
Załącz konto i pobierz dwie zewnętrzne biblioteki.

 MED-61337-CD-091_F000_Mail-Extensions




 OpenStreetMap_Extension_V1.1_TWX8.5

2. Na stronie <https://developer.thingworx.com/en/resources/trials>

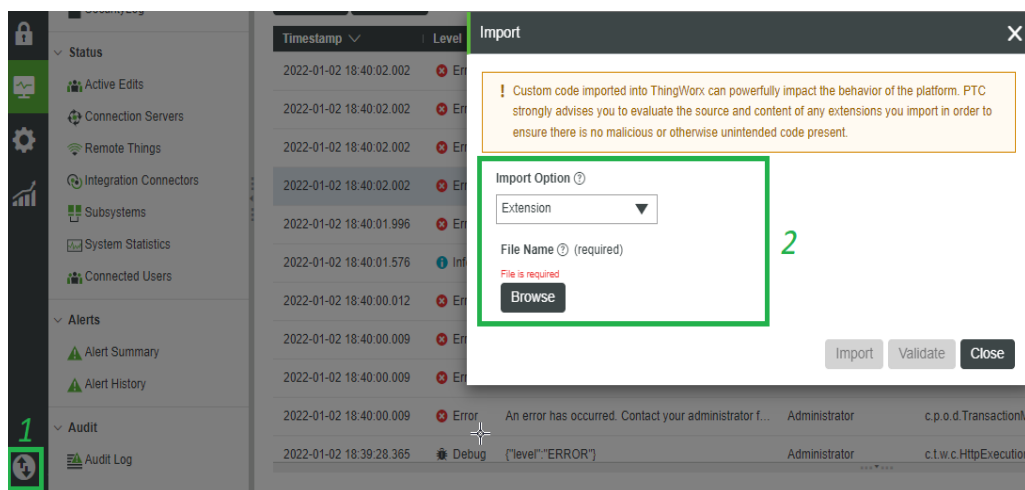
załącz darmowe konto próbne i stwórz „hosted trial” serwer na okres 30dni.



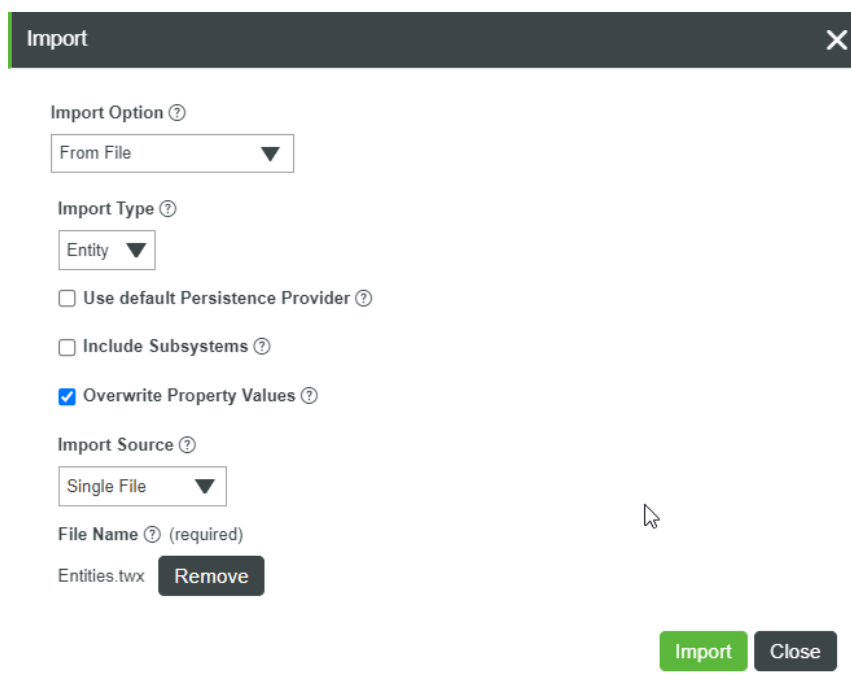
3. Uruchomienie serwera klikając przycisk „Launch”

Trial	Days Remaining	Details	Server Status
 Foundation & Analytics 9.2	28 Days	Server Status: running Hostname: https://PP-21123116135Z.portal.ptc.io/Thingworx Server Type: 9.2 Expiration: 2022-01-30 Username: Administrator Password: Copy to clipboard	<div><div>Launch</div><div>Stop</div></div> <div>Export</div> <div><div>Data </div><div>Entities </div></div>

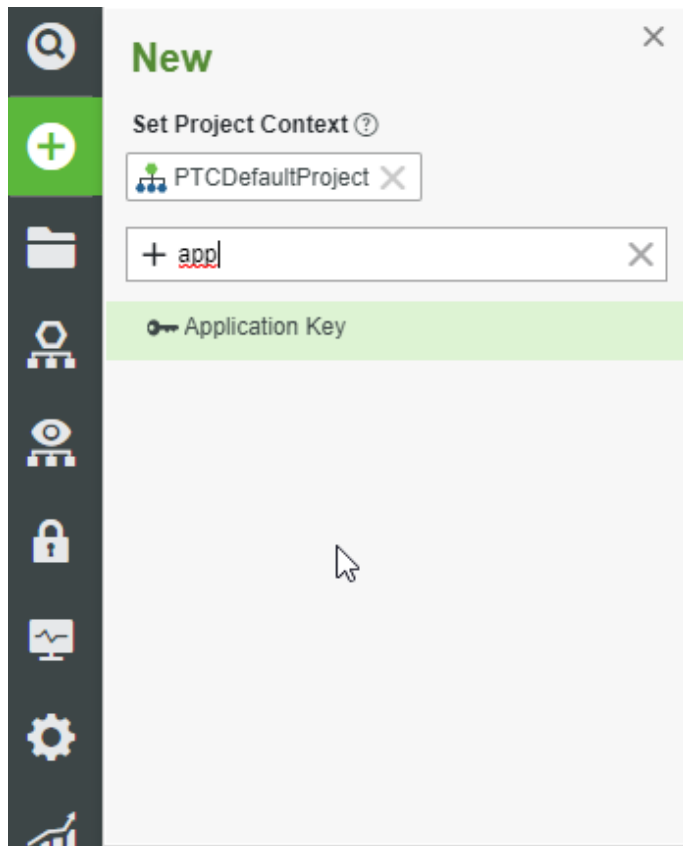
4. Naciśnij przycisk znajdujący się w menu w lewym dolnym rogu (Na rysunku zaznaczony pierwszym zielonym kwadratem). Następnie w oknie, które się pojawiło wybierz Import Option jako „Extension” i importuj dwa pobrane wcześniej pliki.



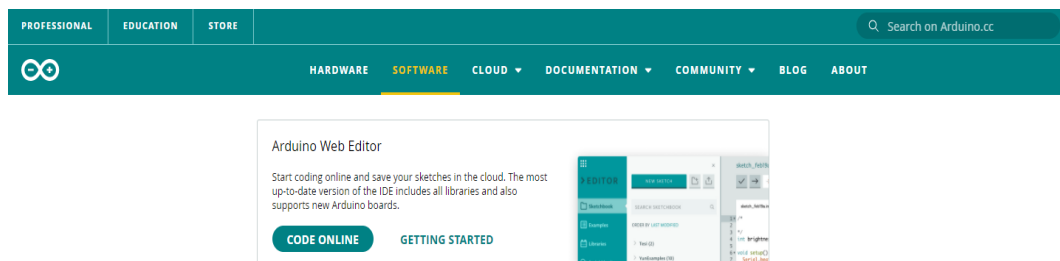
5. Następnie importuj cały projekt. W tym przypadku wybierz Import Option jako „From File” i wybierz plik Entities.twx



6. Stwórz również „Applikation Key” oraz ustaw date wygaśnięcia do dnia wygaśnięcia serwera.



7. Następnym krokiem będzie wgranie oprogramowanie na ESP32 w środowisku Arduino IDE, które znajduje się na stronie <https://www.arduino.cc/en/software>



Downloads



8. Po otworzeniu przejdź do File → Preferences i wklej http://dl.espressif.com/dl/package_esp32_index.json w okienko „Additional Boards Manager URLs”.

Board: "ESP32 Wrover Module"
Upload Speed: "921600"
Flash Frequency: "80MHz"
Flash Mode: "QIO"
Partition Scheme: "Default 4MB with spiiffs (1.2MB APP/1.5MB SPIFFS)"
Core Debug Level: "None"

16. W zakładce Tools wybierz również port USB w którym wpięta jest płytką ESP32. Możesz to sprawdzić włączając panel sterownia. Następnie w managerze urządzeń w zakładce Ports (COM & LPT) będzie informacja pod którym portem znajduje się płytką.
17. Po wszystkich krokach uruchom „Upload”



18. Po wgraniu oprogramowania ESP powinno zacząć komunikować się z platformą.