
Zachodniopomorska
Szkoła Biznesu
w Szczecinie

**Wydział w Szczecinie**

**Kierunek INFORMATYKA**

# Praca inżynierska

**Igor Oskar Maculewicz**

**Nr albumu 27703**

---

## USPRAWNIENIE OBSŁUGI UPRAWY HYDROPONICZNEJ ZA POMOCĄ MIKROKOMPUTERA RASPBERRY PI I OPROGRAMOWANIA STERUJĄCEGO W JĘZYKU PYTHON

Praca inżynierska napisana na specjalności
Aplikacje internetowe i systemy mobilne
pod kierunkiem dr. Pawła Włocha

Szczecin 2022

Oświadczam, że niniejszą pracę przygotowałem/-łam samodzielnie. Wszystkie dane, istotne myśli i sformułowania pochodzące z literatury (przytoczone dosłownie lub niedosłownie) są opatrzone odpowiednimi odsyłaczami. Praca ta nie była w części tej ani podobnej formie przez nikogo przedłożona do żadnej oceny i nie była publikowana. Jednocześnie przyjmuję do wiadomości, że gdyby powyższe oświadczenie okazało się nieprawdziwe, decyzja o wydaniu dyplomu ukończenia studiów zostanie cofnięta.

Oświadczam również, że wersja papierowa i elektroniczna pracy są tożsame. Na mocy art. 76. Ust. 4 Ustawy Prawo o szkolnictwie wyższym i nauce z dnia 20 lipca 2018 r. z późn. zm., wyrażam zgodę na poddanie mojej pracy weryfikacji systemie JEDNOLITY SYSTEM ANTYPLAGIATOWY.

Przyjmuję do wiadomości, że na mocy Art. 347 Ustawy Prawo o szkolnictwie wyższym i nauce z dnia 20 lipca 2018 r. z późn. zm., obroniona praca dyplomowa zostaje umieszczona w OGÓLNOPOLSKIM REPOZYTORIUM PISEMNYCH PRAC DYPLOMOWYCH.

Wyrażam zgodę / ~~Nie wyrażam zgody~~ *(* niepotrzebne skreślić)* na udostępnienie mojej pracy osobom trzecim (niezwiązanym z procedurą dyplomową w ZPSB), w tym do wglądu w bibliotece.

Szczecin, dnia ..... Podpis dyplomanta .....

# SPIS TREŚCI

# Wstęp

W dzisiejszych czasach, konwencjonalne uprawy w ziemi przysparzają rolnikom coraz więcej problemów. Do uprawy, niezbędne jest posiadanie ziemi oraz czasu do jej uprawy. Uprawa na świeżym powietrzu jest bardzo podatna na wszelkie ekstremalne warunki atmosferyczne takie jak nadmierne nasłonecznienie lub podtopienia upraw spowodowane nadmiernymi opadami. To wszystko sprawia że rolnicy coraz częściej rezygnują z upraw w glebie na rzecz upraw hydroponicznych. Uprawa hydroponiczna posiada szereg zalet, oraz eliminuje prawdopodobnie wszystkie wymienione mankamenty uprawy w ziemi. Jedną z większych zalet uprawy hydroponicznej jest to że może być zbudowana praktycznie wszędzie gdzie istnieje dostęp do energii. Ponadto uprawę tą można w pełni kontrolować a co za tym idzie zautomatyzować procesy którymi się rządzi. Istnieje wiele rozwiązań które oferuje dzisiejszy rynek, które pozwalają na zbudowanie autonomicznej uprawy która z nasiona dostarcza nam gotowy produkt.

Celem tej pracy jest próba zaprojektowania i zbudowania podobnego rozwiązania. Rozwiązanie składać będzie się z urządzenia które zarządzać będzie uprawą, oprogramowaniem które sterować będzie urządzeniem, a także aplikacją na smartfona która będzie umożliwiało zdalne zarządzanie uprawą.

Praca podzielona została na pięć rozdziałów.  W pierwszym rozdziale znaleźć można postawiony został cel niniejszej pracy a także opisane zostały założenia oraz ogólny sposób ich realizacji. W drugim rozdziale zamieszczony został kosztorys sprzętu, który jest niezbędny do wykonania postawionych założeń. W trzecim rozdziale zamieszczony został szczegółowy opis projektu oraz budowy warstwy sprzętowej, która spełnia pierwsze z założeń. W czwartym rozdziale zamieszczony został opis projektu oraz implementacji warstwy logicznej, która spełnia drugie z założeń. W piątym rozdziale zamieszczony został projekt oraz opis poszczególnych komponentów z których składa się warstwa prezentacji.
Na końcu pracy znaleźć można wnioski które powstały w wyniku rozwiązywania poniższego problemu, a także bibliografię.

# I. Rozdział

**Problematyka oraz przegląd istniejących rozwiązań**

# 1. Przedstawienie problemu

Od zarania dziejów ludzkość szukała najbardziej efektywnych sposobów prowadzenia upraw rolnych. Pierwsze wzmianki o początkach rolnictwa datuje się na IX tysiąclecie p.n.e. To właśnie wtedy na bliskim wschodzie rozwinęła się kopieniacza uprawa gleby. Stosowane w niej najprostsze narzędzia wykonane z drewna były już swego rodzaju usprawnieniem.

Na przestrzeni dziejów, rolnictwo ewoluowało a wraz z nią pojawiały się nowe metody upraw. Powstawały także coraz to nowe narzędzia usprawniające procesy rolnicze. Zaczynając od narzędzi wykonanych z brązu(kosy, sierpy, haczki, grabie), poprzez wszelkie narzędzia napędzane siłą żywą(pługi, kultywatory, siewniki) aż po współczesne napędzane mechanicznie(kombajn, opryskiwacz, sieczkarnia). To wszystko skupiało się na uprawie roli z wykorzystaniem gleby.

W XVII wieku pojawił się alternatywny sposób pielęgnacji roślin. Został wynaleziony hydroponiczny sposób uprawy, którego głównym założeniem jest brak gleby w procesach rolnych. Odkryto wtedy, że rośliny absorbują niezbędne do własnego rozwoju składniki mineralne w postaci jonów nieorganicznych. Wystarczy stworzyć środowisko, w którym składniki z gleby rozpuszczą się w wodzie, a ziemia nie jest już potrzebna, aby korzenie mogły je wchłaniać.

Sposób ten wymaga jednak bardzo restrykcyjnych warunków. Należy zapewnić odpowiedni przepływ wody a także odpowiednie jej pH. Oprócz tego woda powinna być napowietrzana. Na koniec należy odpowiednio dozować wszelkiego rodzaju nawozy, które dostarczają roślinom wartości odżywcze. Niestety, ze względu na tak dużą ilość zmiennych czynników, bardzo ciężko jest wykonywać to wszystko ręcznie.

Stąd też pomysł na stworzenie urządzenia, które automatycznie zarządzać będzie wybranymi wyżej wymienionymi czynnikami. Genezą pomysłu jest chęć usprawnienia działania istniejącej już manualnej uprawy hydroponicznej. Metoda tejże uprawy nazywana jest metodą przepływową. Polega to na tym, że przez koryto w którym znajdują się sitka wypełnione keramzytem, przepływa woda która jest odpowiednio napowietrzona i nawieziona. W keramzycie umieszcza się wcześniej przygotowane nasionko, które przed tym musi najpierw wykiełkować w nawilżonej wacie. Każda roślina posiada swój własny profil

uprawy. Jest wiele opracowań naukowych, które traktują o tym jakie parametry powinno się dobrać aby uprawa była najefektywniejsza.

Cała zebrana wiedza, pozwala na zaprojektowanie oraz wdrożenie urządzenia wraz z aplikacją która pozwoli na automatyczne bądź zdalne manualne zarządzanie uprawą.

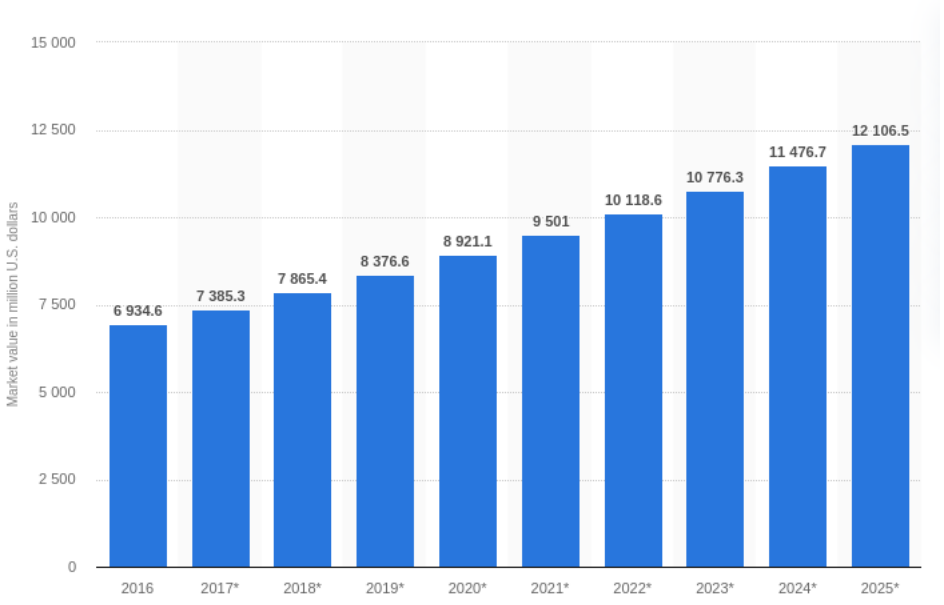
# 2. Hydroponika

Uprawa hydroponiczna jest uprawą w pełni kontrolowaną. Bardzo precyzyjnie można określać poziom czynników wpływających na wzrost rośliny, między innymi poziom oświetlenia, poziom napowietrzenia, częstotliwość przepływu wody oraz ilość nawozu. Dlatego też można stosować wiele mechanizmów wspierających uprawę. Zaliczają się do nich wszelkie czujniki takie jak pH-metr oraz EC-metr, precyzyjne pompy(perystaltyczne) do dokładnego dozowania nawozów, specjalnie przygotowane taśmy led zapewniające odpowiednie oświetlenie(np. diody czerwone 580-660 nm oraz niebieskie 390-460 nm w proporcjach 5:1) a także wszelkie moduły wspierające obsługę wyżej wymienionych mechanizmów, takie jak na przykład mikrokomputer Raspberry PI lub mikrokontroler Arduino.

Ponadto uprawa hydroponiczna może stać się uprawą w pełni ekologiczną przy zastosowaniu odnawialnych źródeł energii takich jak na przykład farma fotowoltaiczna lub przydomowa elektrownia wiatrowa lub wodna.

# 3. Popularność upraw hydroponicznych

Na przestrzeni lat bardzo popularnym trendem stało się stosowanie upraw hydroponicznych. Dowodem na to jest poniższy wykres obrazujący wzrost udziału w rynku upraw hydroponicznych.




*Rys. 1 - Wykres wzrostu udziału w rynku w latach 2016-2025*

Na wykresie z poprzedniej strony możemy zaobserwować prawie dwukrotny wzrost popularności tego typu upraw w przeciągu dziewięciu lat. Wzrost spowodowany jest wieloma zaletami które płyną z uprawy hydroponicznej. Uprawa hydroponiczna jest o wiele łatwiejsza w zarządzaniu, bardziej elastyczna oraz bardziej wydajniejsza od konwencjonalnych upraw z wykorzystaniem ziemi. Nie zajmuje ona także wielkiej powierzchni i może być zaimplementowana w dowolnym miejscu. Uprawę możemy zabezpieczyć przed skrajnymi warunkami atmosferycznymi takimi jak susza lub podtopienia spowodowane deszczem. Jedną z ważniejszych zalet jest to, że przy zastosowaniu ekologicznych źródeł energii takich jak panele fotowoltaiczne lub farma wiatrowa, nasza uprawa staje się samowystarczalna.

# 4. Rozwiązania dostępne na rynku

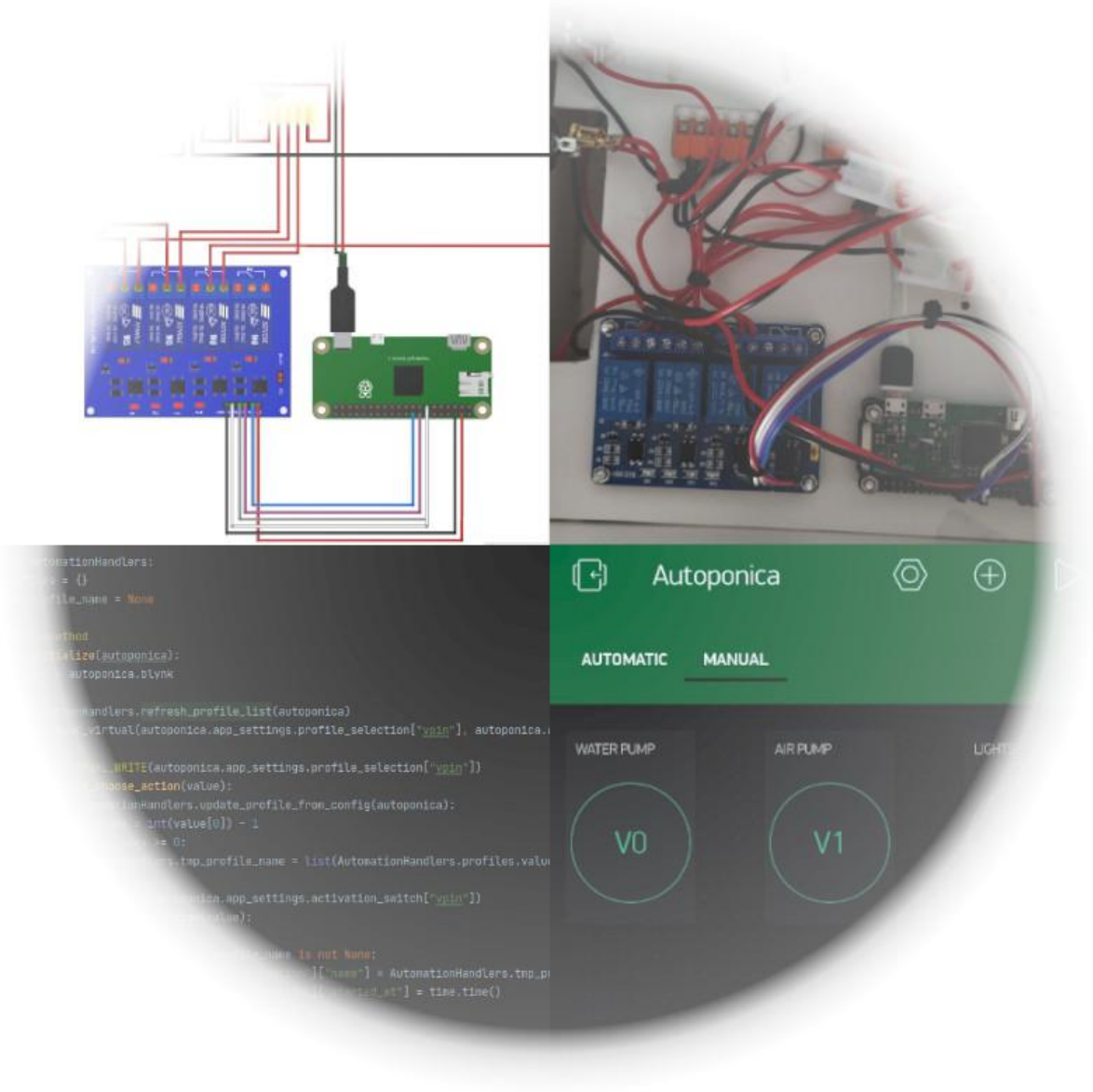
Na rynku istnieje niewiele komercyjnych technologii automatyzacji upraw hydroponicznych. Zaliczają się do nich projekty:

- **Jalkrishi** - Automatyczny, przemysłowy system uprawy sterowany przez jednostkę stacjonarną komputera. Posiada wszelkie mechanizmy dozowania nawozu za pomocą pomp, zbiornik na świeża oraz zużytą wodę.
- **Viscon Hydroponics** - Automatyczny bądź półautomatyczny, przemysłowy system uprawy. Wymaga bardzo dużej powierzchni magazynowej ze względu na poziom skomplikowania. System ten jest systemem kompleksowym gdyż automatyzuje proces od wykiełkowania sadzonki aż po otrzymanie gotowego produktu.
- **Seedo** - Jest to domowy system uprawy hydroponicznej. Zapewnia wszystkie niezbędne założenia uprawy hydroponicznej, takiej jak obieg powietrza, wody, oświetlenie oraz odpowiednie nawożenie. Sterowanie odbywa się za pomocą smartfona.

Wszystkie wymienione systemy wiążą się albo z wysokim kosztem zakupu systemu(Przydomowy seedo to koszt 3000$) lub z obowiązkiem posiadania bardzo dużych przestrzeni użytkowych. Wiążą się także z ograniczoną skalowalnością gdyż są to rozwiązania dedykowane, niepodatne na rozszerzanie.

Rozwiązanie opisane w niniejszej pracy, jest rozwiązaniem stosunkowo tanim(urządzenie z podstawowymi funkcjami opisane zostało w kosztorysie, dodatkowo można je rozszerzyć o czujniki pH oraz EC i pompki dozowania nawozu co wiąże się z dodatkowym kosztem około 200zł).

Nie wymaga on także posiadania dużych powierzchni użytkowych. To użytkownik końcowy sam bądź z pomocą eksperta w danej dziedzinie, może dobrać odpowiednią wielkość uprawy wedle jego potrzeb.




# II. Rozdział

**Projekt usprawnienia uprawy hydroponicznej za pomocą mikrokomputera raspberry pi i oprogramowania sterującego w języku python**

# 1. Cel i założenia pracy

Celem niniejszej pracy inżynierskiej jest usprawnienie obsługi nawadniania, napowietrzania oraz doświetlania uprawy hydroponicznej. Aby zrealizować cel niniejszej pracy, należy zaprojektować i zaimplementować rozwiązanie spełniające trzy założenia.

Pierwszym z nich jest zaprojektowanie oraz stworzenie warstwy sprzętowej którą będzie sterowniki, kontrolujący uprawę. Warstwa sprzętowa powinna realizować następujące założenia: zapewnienie ciągłego obiegu wody, zapewnienie ciągłego napowietrzania wody oraz zapewnienie sterowania oświetleniem.

Drugim założeniem jest zaprojektowanie i stworzenie warstwy logicznej, czyli aplikacji która posadowiona będzie na mikrokomputerze. Warstwa logiczna powinna udostępniać odpowiedni interfejs, który pozwala sterować urządzeniami w warstwie sprzętowej oraz umożliwiać zarządzanie szablonami uprawy.

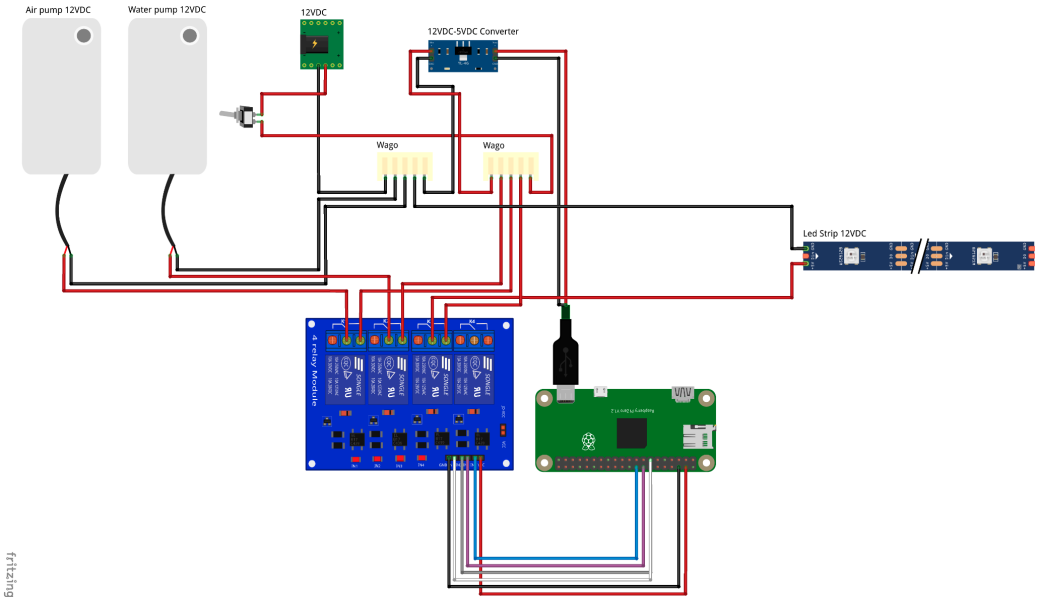
Ostatnim, trzecim założeniem jest zaprojektowanie i stworzenie warstwy prezentacji którą będzie aplikacja na smartfona. Warstwa prezentacji powinna realizować wizualną reprezentację aktualnego stanu warstwy sprzętowej oraz umożliwiać poprzez warstwę logiczną, sterowanie warstwą sprzętową.

## 1.2. Warstwa sprzętowa

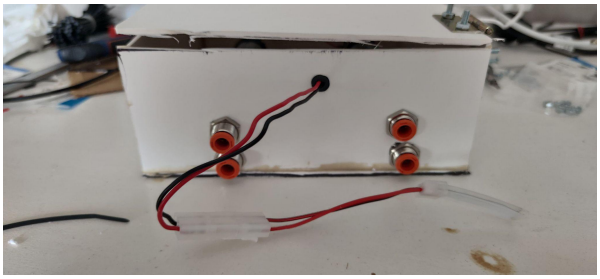
W tym podrozdziale opisane zostało założenie utworzenia struktury warstwy sprzętowej. Warstwa sprzętowa powinna dzielić się na dwie warstwy:

- Warstwa robocza - będą znajdować się w niej komponenty robocze(pompa wody i pompa powietrza)
- Warstwa sterująca - będą znajdować się w niej wszystkie komponenty odpowiedzialne za sterowanie komponentami roboczymi.

Oprócz powyższych warstw całość urządzenia zamknięta powinna zostać w obudowie z tworzywa sztucznego. Ponadto powinna zostać zaprojektowana w sposób, który umożliwia łatwy demontaż warstwy sterującej, aby uzyskać dostęp do warstwy roboczej. Dodatkowo na obudowie znajdować się włącznik główny a także gniazdo 12VDC.




*Rys. 2 - Schemat urządzenia*



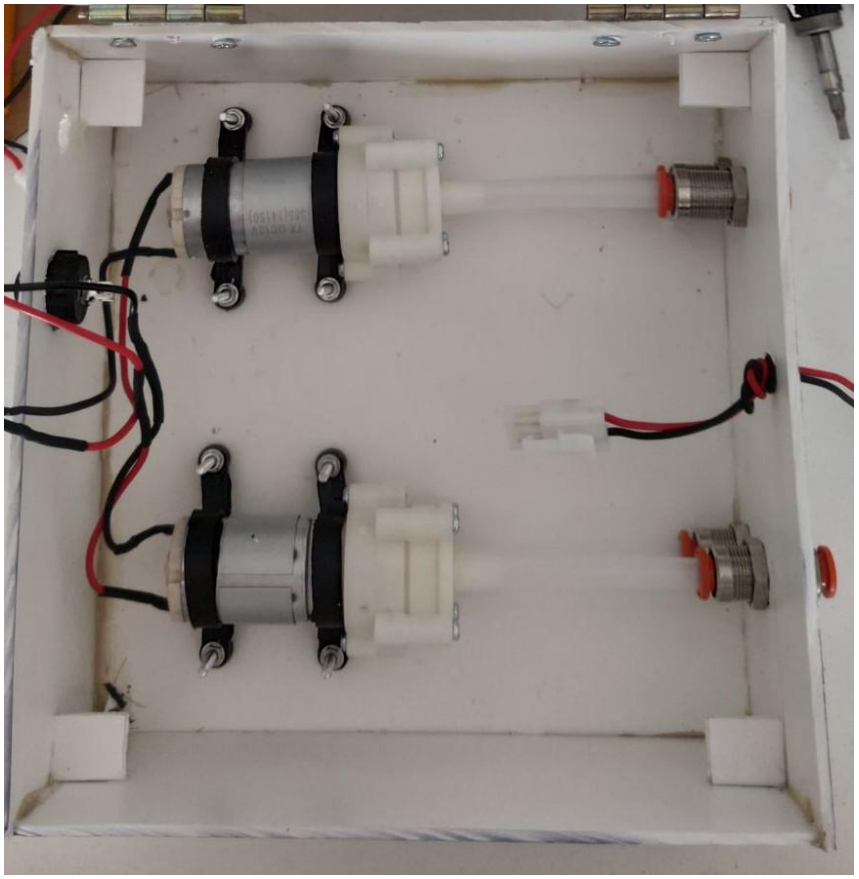
*Rys. 3 - Przednia część obudowy*

*Rys. 4 - Tylna część obudowy*

## Warstwa robocza

W warstwie roboczej powinny elementy robocze które odpowiadają za dostarczanie do uprawy niezbędnych składników takich jak woda czy odpowiednia ilość powietrza. Powinny znaleźć się tu takie elementy jak:

- Pompa wodny
- Pompa powietrza
- Wyprowadzenia kablowe w odpowiednim kolorze
- Rurki z PVC łączące obudowę z pompami
- Złącza tablicowe w obudowie

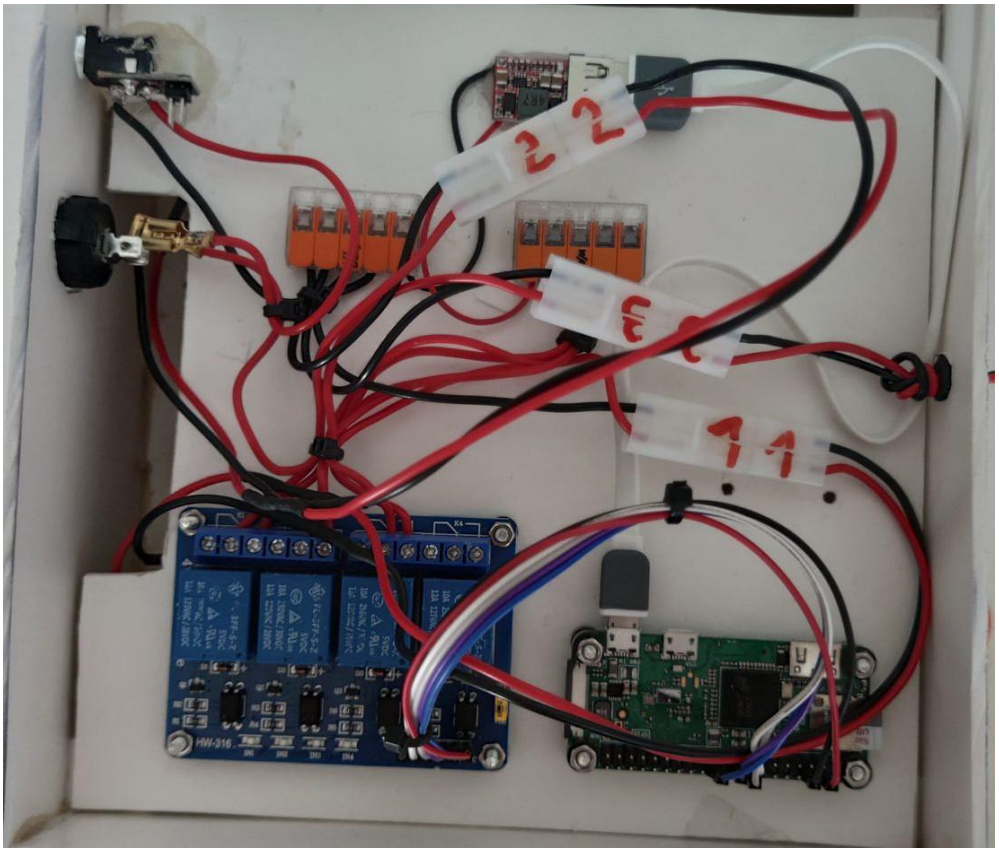


*Rys. 5 - warstwa robocza urządzenia*

## Warstwa sterująca

W warstwie sterującej powinny znaleźć się podzespoły oraz połączenia które realizują sterowanie urządzeniami w warstwie roboczej. Powinny znaleźć się tutaj takie elementy jak:

- Mikrokomputer Raspberry PI
- Zespół przekaźników które sterują wyższym napięciem(12VDC) w warstwie roboczej,
- Przetwornica z 12VDC na 5VDC
- Dwie złączki wago które łączą linię napięciową 12VDC
- Wyprowadzenie gniazda oświetlenia
- Włącznik urządzenia
- Gniazdo 12VDC

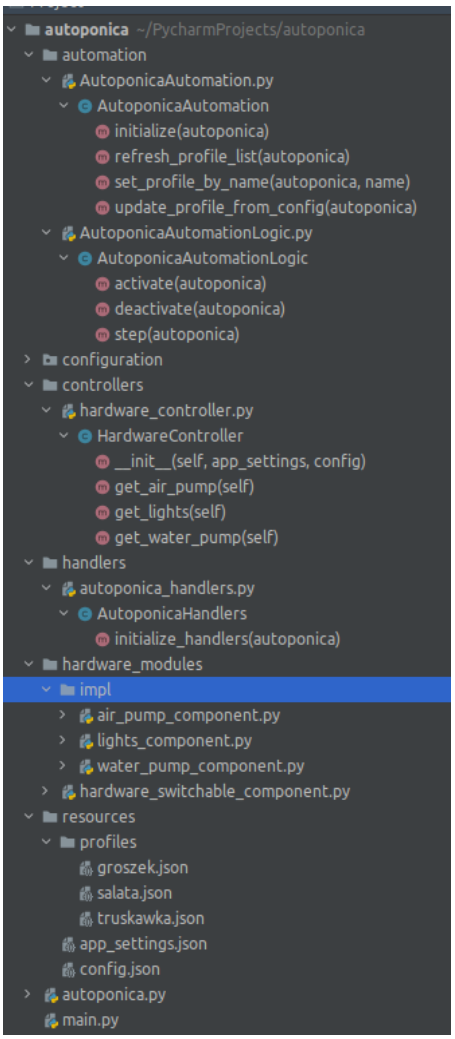


*Rys. 6 - warstwa robocza urządzenia*

## 1.3. Warstwa logiczna

Warstwa logiczna powinna być aplikacją napisaną w języku Python, która posadowiona będzie na mikrokomputerze Raspberry Pi umieszczonym w warstwie sterującej urządzenia.

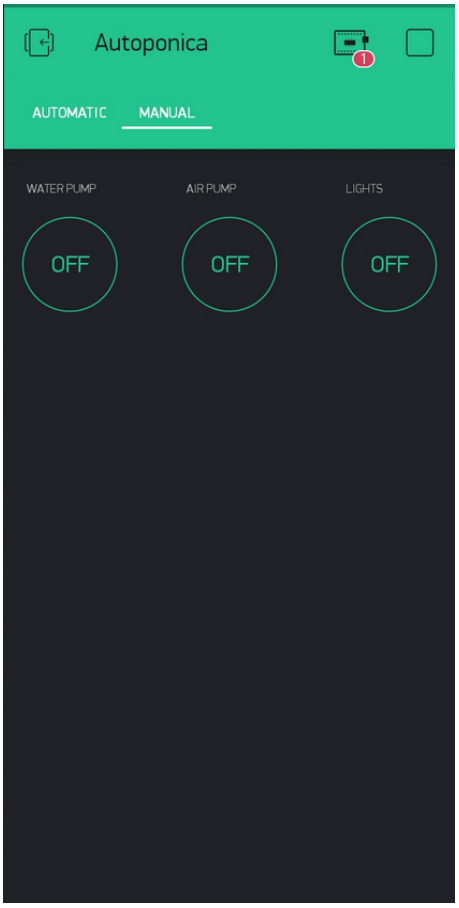
Aplikacja realizować powinna komunikację dwustronną poprzez serwer(blynk.io), który pośredniczyć będzie w komunikacji z warstwą prezentacji. W warstwie logicznej zawarta powinna zostać logika urządzenia, która łączyć będzie akcje wykonane w warstwie prezentacji i przypisane odpowiedniej akcji zdarzenie w warstwie sprzętowej. Warstwa logiczna powinna także realizować zarządzanie szablonami upraw oraz automatyzację uprawy wedle wybranego szablonu.

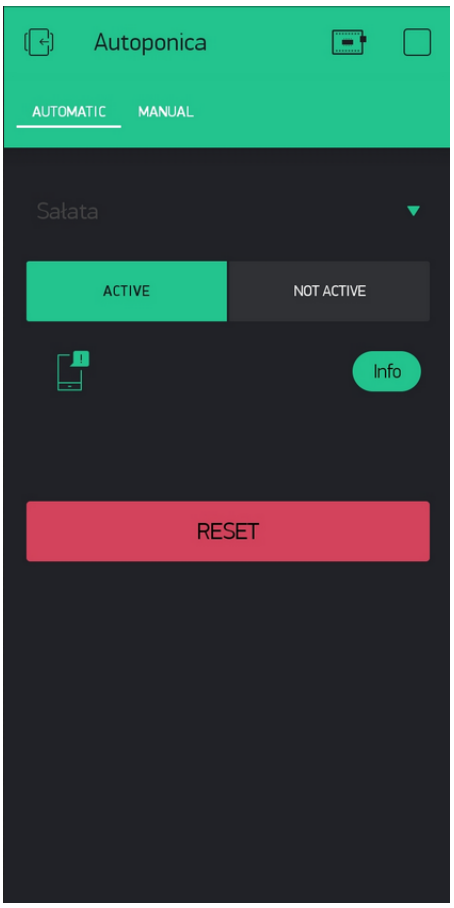



*Rys. 7 - struktura warstwy logicznej*

## 1.4. Warstwa prezentacji

Warstwę prezentacji powinna być reprezentowana przez aplikację stworzona w kreatorze aplikacji dla platformy blynk.io. Aplikacja ta powinna realizować funkcjonalności manualnego sterowania urządzeniem oraz zarządzania opcjami trybu automatycznego, w którym uprawą zarządza samo urządzenie.




*Rys. 8 - Ekran sterowania manualnego* *Rys. 9 - Ekran sterowania automatycznego*

# 2. Kosztorys

Poniżej przedstawiony został orientacyjny kosztorys wszystkich elementów które niezbędne są do zrealizowania założeń opisanych w powyższych rozdziałach. Są to ceny orientacyjne, części które wybierane były w procesie projektowania urządzenia.

| Nazwa pozycji | Cena | Ilość |
|---|---|---|
| Raspberry PI Zero WH+ | 79zł | 1 |
| Przetwornica 12VDC-5VDC | 10zł | 1 |
| Moduł 4-kanałowy przekaźnik 5V 10A | 20zł | 1 |
| Wago 5-cio złączowe | 4,50zł | 2 |
| Przewód instalacyjny 1x0,5 H05V-K - czerwony - rolka 100m | 40zł | 1 |
| Przewód instalacyjny 1x0,5 H05V-K - czarny - rolka 100m | 40zł | 1 |
| Przewody kable zworki F-F 40 szt 20cm - żeńsko-żeńskie | 4zł | 1 |
| Rurka PCV 8mm 20m | 50zl | 1 |
| Pompa membranowa 12VDC | 14.50zł | 2 |
| Złączka dwustronna grodziowa 8mm | 25zł | 2 |
| Kabel USB na micro USB | 5zł | 1 |
| Włącznik ON/OFF | 2zł | 1 |
| Gniazdo DC | 0.40zł | 1 |
| Zestaw śrub mocujących 4mm | 5zł | 1 |
| Zestaw nakrętek mocujących 4mm | 3.50zł | 1 |
| Zestaw podkładek 4mm | 4zł | 1 |
| Płyta z PCV spienionego 3 mm 100 x 50 cm biała | 60zł | 1 |
| **Razem: 362 zł 40 gr.** | | |

*Tabela 1 - Przybliżony kosztorys komponentów urządzenia*

# 3. Projekt warstwy sprzętowej

W tym rozdziale opisana została warstwa sprzętowa całego projektu. Warstwa sprzętowa odpowiada za wykonywanie wszelkich fizycznych czynności wspomagających uprawę. Wybrane funkcje to:

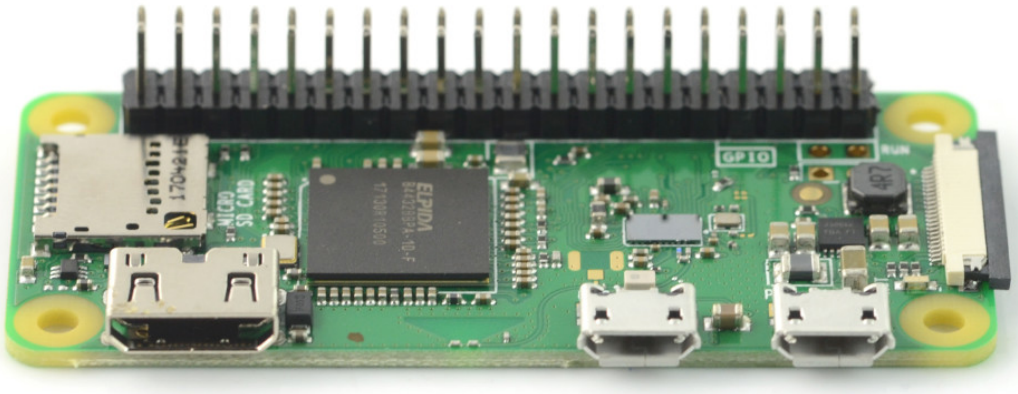
- Zapewnienie ciągłego obiegu wody
- Zapewnienie ciągłego napowietrzania wody
- Zapewnienie sterowania oświetleniem
- Automatyczne zarządzanie uprawą

Warstwa sprzętowa dzieli się na dwie warstwy: warstwę roboczą, w której znajdują się wszelkie urządzenia wykonujące czynności wspomagające uprawę oraz warstwę sterującą, która odpowiada za sterowanie urządzeniami w warstwie roboczej.

## 3.1. Omówienie komponentów

W tym podrozdziale omówione zostały wszystkie komponenty użyte w obu warstwach urządzenia.

### Raspberry PI

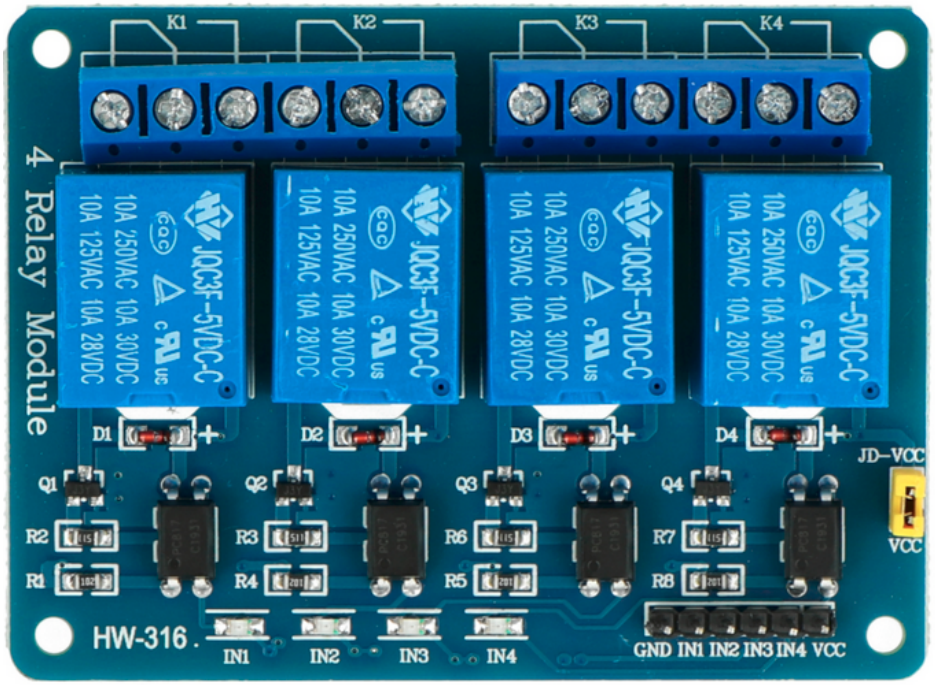



*Rys. 10 - Raspberry Pi Zero WH*

**Raspberry Pi** – Platforma komputerowa stworzona przez Raspberry Pi Foundation. Urządzenie składa się z pojedynczego obwodu drukowanego i zostało wymyślone, by wspierać naukę podstaw informatyki. Jego premiera miała miejsce 29 lutego 2012 roku.

Występuje w różnych konfiguracjach ze względu na ilość ramu, procesor a także warianty komponentowe(Bluetooth, WIFI, dodatkowe złącza). Wariant użyty w projekcie to Raspberry Pi Zero WH 512MB RAM - WiFi + BT 4.1.

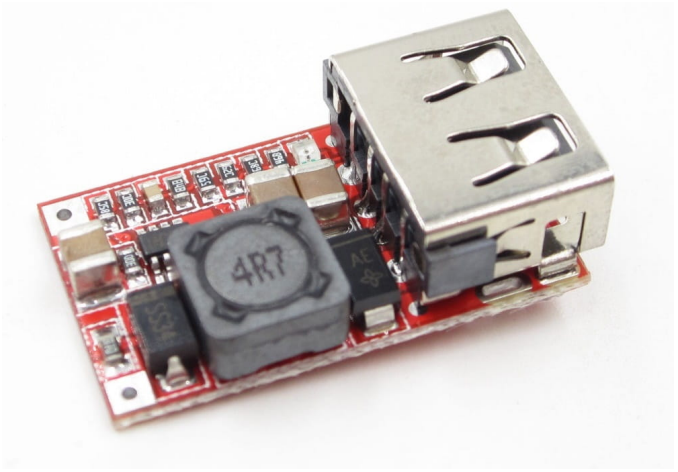
## Czterokanałowy moduł przekaźników




*Rys. 11 - Czterokanałowy moduł przekaźników 5VDC 10A*

**Czterokanałowy moduł przekaźników** – Moduł pozwala na sterowanie elementami wykonawczymi pobierającymi prąd do 10 A przy pomocy portów mikrokontrolera lub dowolnego zestawu uruchomieniowego. Optoizolacja separuje sygnał sterujący od części związanej z zasilaniem przekaźnika, dzięki czemu zapewnia bezpieczeństwo pracy układu sterującego.

## Przetwornica 5V Step down 12-24V 3A




*Rys. 12 - Przetwornica 5V Step down 12-24V 3A*

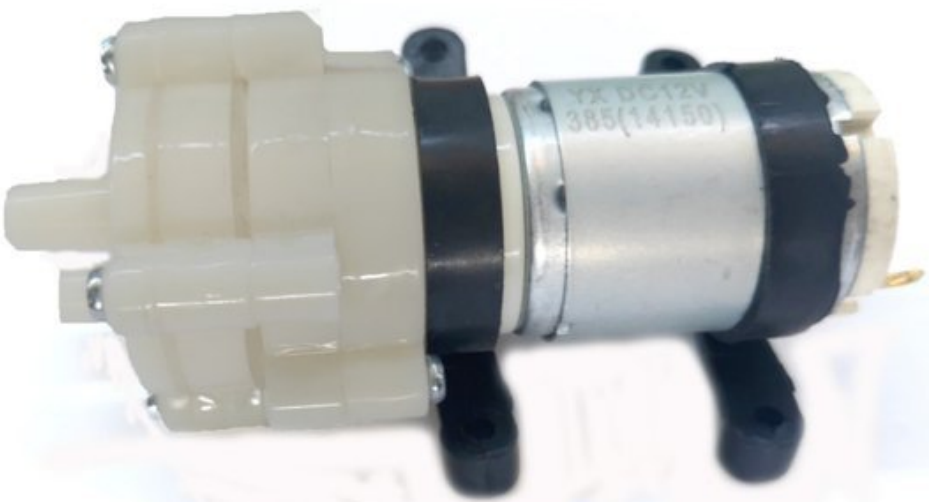
**Przetwornica 5V Step down 12-24V 3A** – Przetwornica obniżająca napięcie wejściowe z zakresu od 6V do 24V na napięcie 5V na złączu USB typu A (klasyczne).

## Złącza wago


*Rys. 13 - Złączka wago*

**Złącza wago** – Złącza zaciskowe pozwalają na szybki montaż przewodów w instalacjach elektrycznych. Dzięki szerokiemu asortymentowi złączy możliwy jest dobór odpowiednich modeli o najbardziej odpowiedniej liczbie podłączeń. W projekcie został zastosowany model 5-pinowy.

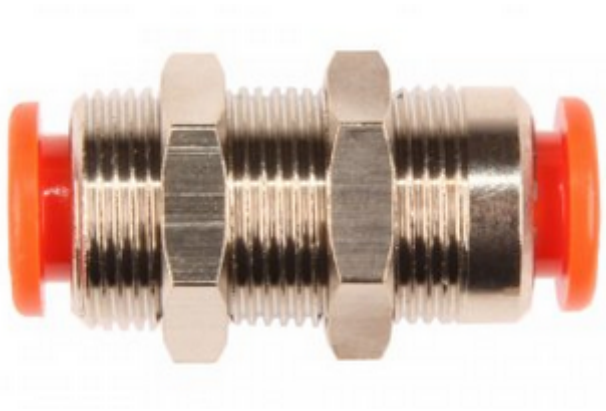
## Pompa membranowa- 12V - 3W



*Rys. 14 - Pompa membranowa 12V*

**Pompa membranowa 12V 3W** – Pompa membranowa o mocy 3W z silnikiem R385+, zasilanym napięciem 12V. Pompa pracująca na niskie napięcie jest bezpieczna, a także dzięki małemu poboru prądu jej użytkowanie nie wiąże się z dużymi kosztami.

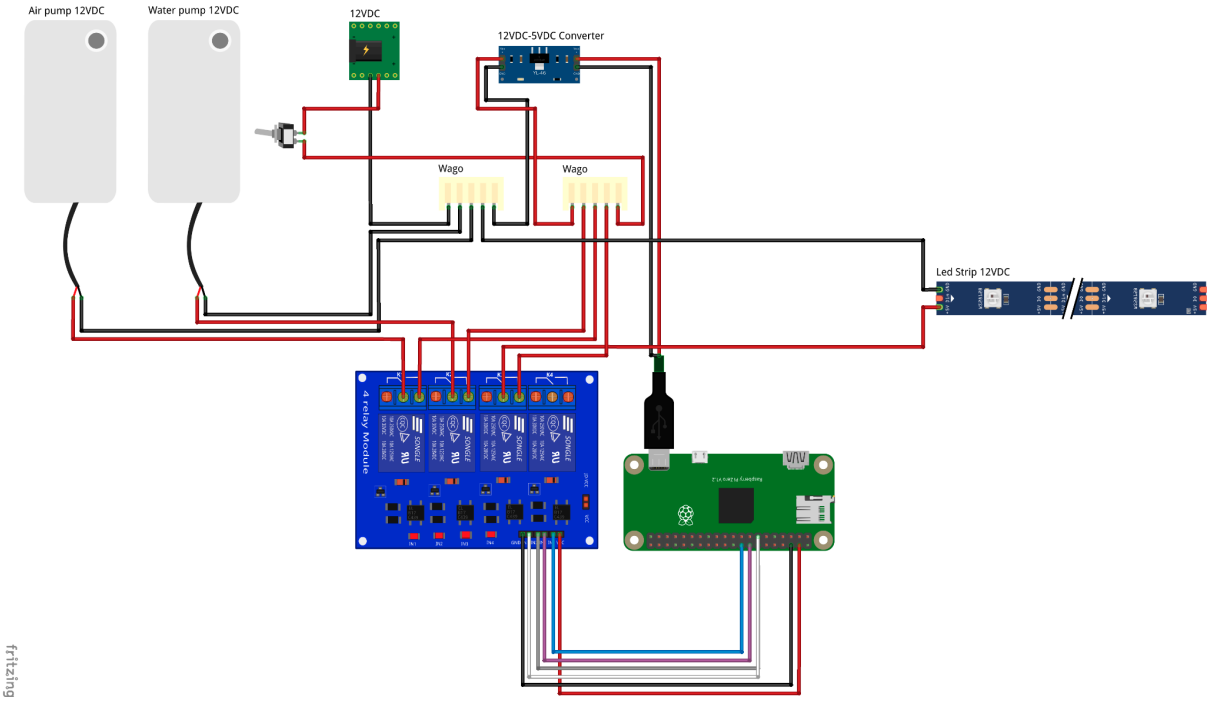
## Złączka grodziowa wtykowa prosta, typ R10



*Rys. 15 - Pompa membranowa 12V*

**Złączka grodziowa wtykowa prosta, typ R10** – Złączka przeznaczona do połączeń przewodów i innych elementów w układach pneumatycznych. Łatwe i szybkie w użyciu do wielokrotnego montażu bez utraty szczelności po stronie mechanicznej i pneumatycznej. Materiał: mosiądz niklowany. Ciśnienie robocze: 16 bar. Medium robocze: próżnia, powietrze. Temperatura pracy: od -20°C do +80°C. Uszczelnienie: NBR.

## 3.2. Opis schematu połączeń




*Rys. 16 - Schemat połączeń*

Poprzez zasilacz podawane jest napięcie 12VDC. Podawane jest ono w dwa miejsca:

- Do przetwornicy która konwertuje to napięcie na 5VDC
- Do linii zasilania 12VDC.

Z przetwornicy napięcie po przetworzeniu, podawane jest do Raspberry Pi, do gniazda zasilania mikrokomputera.

Z linii 12VDC zasilane są komponenty robocze(pompy oraz oświetlenie).

To czy napięcie jest podawane lub nie na komponenty robocze, określa program komputerowy, który poprzez piny sterujące, podaje stan wysoki lub niski do odpowiedniego przekaźnika.

# 4. Projekt warstwy logicznej

W rozdziale tym opisana została szczegółowo architektura aplikacji, jej najważniejsze komponenty oraz wymagania. Aplikacja została zrealizowana w języku Python oraz przy użyciu platformy Blynk.io. Program ten osadzony jest na urządzeniu i jest uruchamiany przy jego starcie. Program ma za zadanie poprzez serwer pośredniczący, realizować obustronną komunikację z warstwą prezentacji i reagować na wszelkie akcje wykonywane na tejże warstwie. Każda akcja ma przypisane odpowiednie zdarzenie które zachodzi w urządzeniu.
Cała aplikacja oparta jest o platformę blynk.io, która usprawnia wytwarzanie tego typu rozwiązań. Platforma udostępnia gotowy serwer pośredniczący, oraz kreator aplikacji na smartfona.

## 4.1. Wymagania

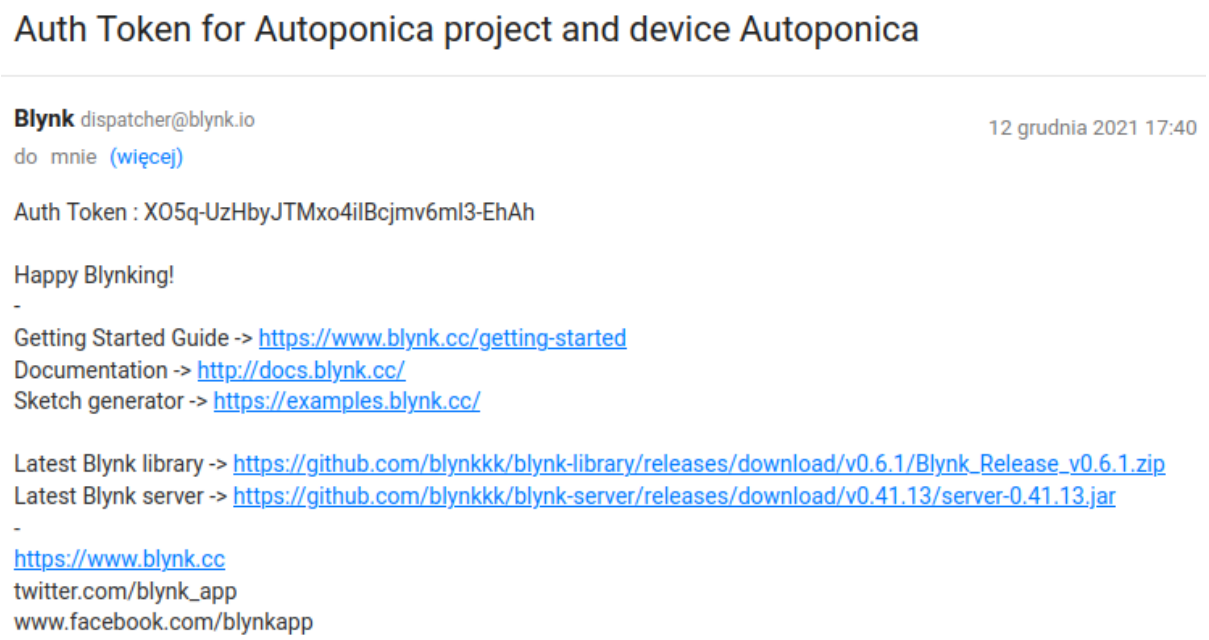
| Nazwa | Wersja |
|---|---|
| System Microsoft Windows lub Linux | Dowolna obsługująca technologię Pythona w wersji 3 lub wyższej |
| Python | 3.* |
| Pakiet pip installer | 3.* |

## 4.2. Platforma blynk.io

Blynk jest to platforma IoT(Internet of Things). Umożliwia ona zdalną kontrolę nad modułami sprzętowymi oraz urządzeniami. Jest w stanie odczytywać dane z czujników i wizualizować je w formie wygodnej dla użytkownika. Może także przechowywać dane lub integrować się z wieloma znanymi serwisami(Twitter, Facebook).

## Konfiguracja projektu

Na początku należy zarejestrować się w na stronie blynk.io. Po zalogowaniu się na zarejestrowane konto, należy utworzyć nowy projekt. Po utworzeniu nowego projektu na adres email podany w procesie rejestracji wysłany zostanie token dostępu, którego użyty zostanie do autoryzacji z poziomu aplikacji.


Auth Token for Autoponica project and device Autoponica

**Blynk** dispatcher@blynk.io 12 grudnia 2021 17:40
do mnie (więcej)

Auth Token : XO5q-UzHbyJTMxo4ilBcjmv6ml3-EhAh

Happy Blynking!
-
Getting Started Guide -> https://www.blynk.cc/getting-started
Documentation -> http://docs.blynk.cc/
Sketch generator -> https://examples.blynk.cc/

Latest Blynk library -> https://github.com/blynkkk/blynk-library/releases/download/v0.6.1/Blynk_Release_v0.6.1.zip
Latest Blynk server -> https://github.com/blynkkk/blynk-server/releases/download/v0.41.13/server-0.41.13.jar
-
https://www.blynk.cc
twitter.com/blynk_app
www.facebook.com/blynkapp

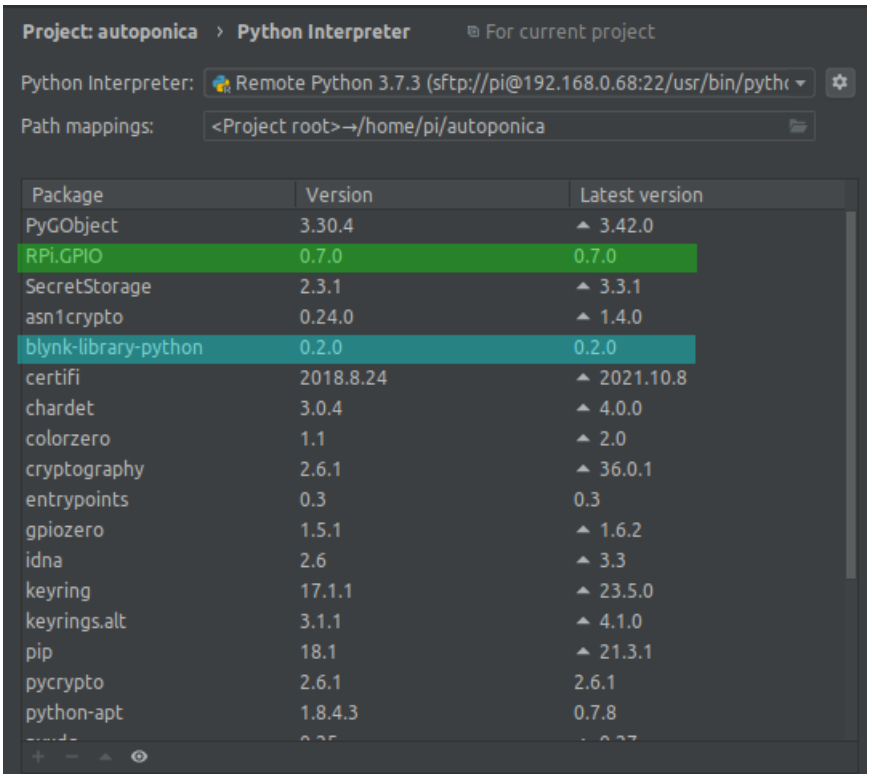
*Rys. 17 - Dane dostępowe do projektu*

Następnie, należy utworzyć nowy projekt w języku Python. Po utworzeniu nowego projektu należy zainstalować bibliotekę która umożliwi komunikację z serwerem pośredniczącym, oraz bibliotekę do komunikacji z pinami w mikrokomputerze Raspberry Pi.

Aby to osiągnąć należy wykonać następujące polecenia:

***pip install GPIO.RPi***

***pip install blynk-library-python***




*Rys. 18 - Biblioteki niezbędne do uruchomienia projektu*

Aby połączyć się z projektem na serwerze pośredniczącym, należy podać token dostępowy otrzymany w wiadomości email a następnie użyć poniższego kodu:

```
import blynklib
# import blynklib_mp as blynklib # micropython import

BLYNK_AUTH = 'XO5q-UzHbyJTMxo4ilBcjmv6ml3-EhAh' #insert your Auth Token here
# base lib init
blynk = blynklib.Blynk(BLYNK_AUTH)
```

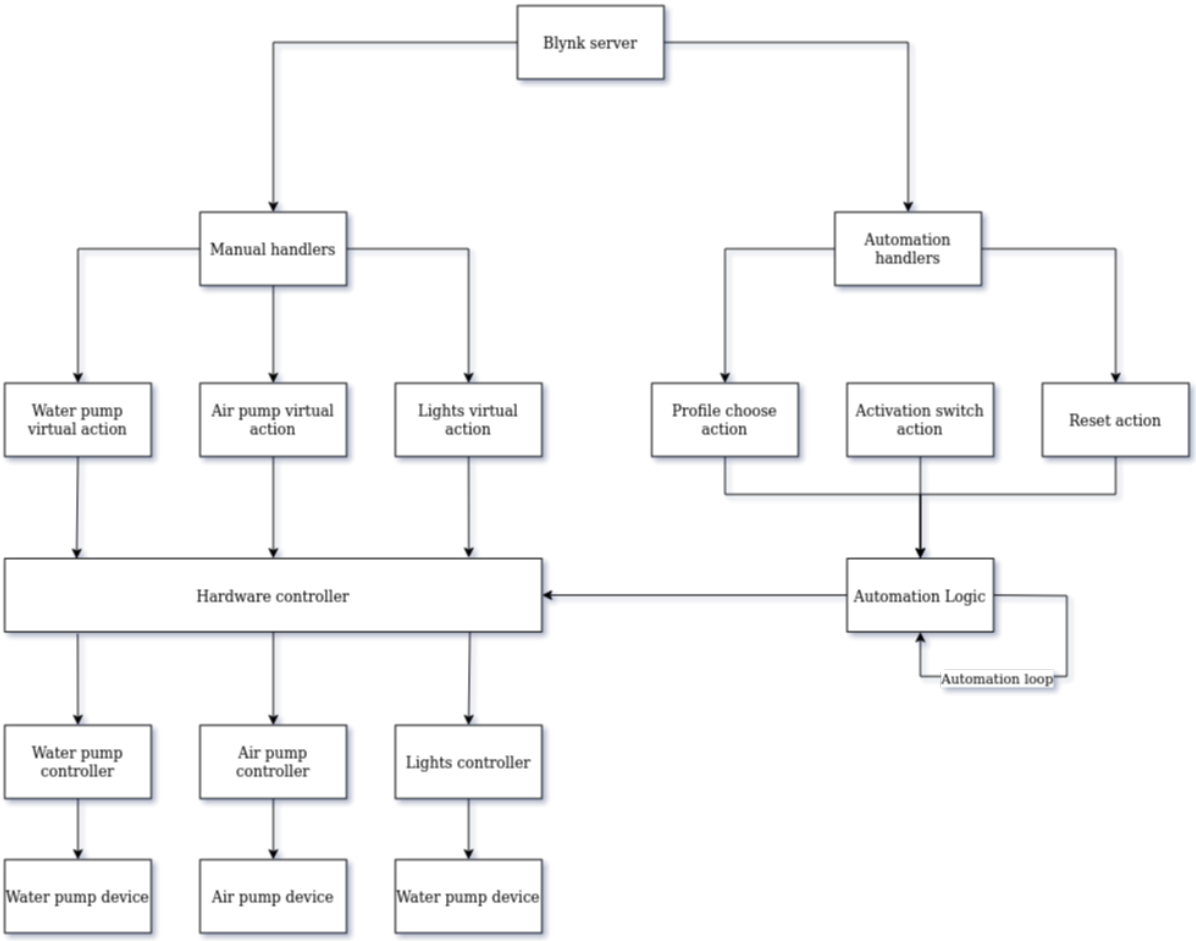
*Rys. 19 - Sposób łączenia się z serwerem pośredniczącym platformy Blynk*

## 4.2. Architektura aplikacji

W tym podrozdziale została rozpisana architektura aplikacji która osadzona jest na urządzeniu.

Architektura aplikacji dzieli się na dwie główne warstwy:

- Warstwę zarządzania manualnego która, poprzez warstwę kontroli komponentów sprzętowych pozwala na sterowanie każdym z podłączonych komponentów w urządzeniu.
- Warstwę zarządzania automatycznego która skupia się na automatyzacji uprawy za pomocą profili. Logika automatyzacji steruje urządzeniami poprzez warstwę kontroli komponentów sprzętowych.




*Rys. 20 - Architektura aplikacji*

## Blynk server

Jest to serwer do którego aplikacja na smartfona wysyła wszelkie żądania o wykonanie akcji w urządzeniu. Serwer występuje w dwóch wariantach:

- Chmurowym, który za każdy wykorzystany komponent w warstwie prezentacji pobiera opłatę w postaci punktów energii które możemy doładowywać za pomocą realnej waluty.
- Standalone, który możemy posadowić na dowolnej maszynie wspierającej technologię Java. Rozwiązanie to jest bardziej budżetowe gdyż nie wymaga kupna energii.

Komunikacja z serwerem odbywa się za pomocą protokołu TCP/IP. Serwer przekazuje wiadomość do odpowiedniego urządzenia także za pomocą tego protokołu.

Opis protokołu wygląda następująco:

**blynkkk/blynk-server/blob/master**
**/docs/README_FOR_APP_DEVS.md#protocol-messages** 124

```
# Protocol messages

Every message consists of 2 parts.

+ Header :
    + Protocol command (1 byte);
    + MessageId (2 bytes);
    + Body message length (2 bytes);

+ Body : string (could be up to 2^15 bytes).

Blynk transfers binary messages with the following structure:

| Command       | Message Id    | Length/Status   | Body     |
|:-------------:|:-------------:|:---------------:|:--------:|
| 1 byte        | 2 bytes       | 2 bytes         | Variable |

So, message is always "1 byte + 2 bytes + 2 bytes + messageBody.length".

### Command field
```

*Rys. 21 - Architektura aplikacji*

## Warstwa manual handlers

Jest to warstwa z którą komunikuje się serwer blynk i przekazuje mu wszystkie akcje manualnego sterowania urządzeniem. Wirtualne piny, które później używane są przy projektowaniu aplikacji na smartfona odczytywane są z pliku konfiguracyjnego. W tej warstwie wyróżnia się 3 akcje:

- włącz/wyłącz pompę wody
- włącz/wyłącz pompę powietrza
- włącz/wyłącz oświetlenie

Przy użyciu jakiejkolwiek z manualnych akcji, automatyzacja uprawy jest dezaktywowana.

```python
class ManualHandlers:
  @staticmethod
  def initialize_handlers(autoponica):
    blynk = autoponica.blynk
    hardware_controller = HardwareController(autoponica.app_settings, autoponica.config)

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.water_pump["vpin"])
    def water_pump_action(value):
      if value[0] == '1':
        AutoponicaAutomationLogic.deactivate(autoponica)
      hardware_controller.get_water_pump().handle(value)

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.air_pump["vpin"])
    def air_pump_action(value):
      if value[0] == '1':
        AutoponicaAutomationLogic.deactivate(autoponica)
      hardware_controller.get_air_pump().handle(value)

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.lights["vpin"])
    def lights_action(value):
      if value[0] == '1':
        AutoponicaAutomationLogic.deactivate(autoponica)
      hardware_controller.get_lights().handle(value)
```

*Rys. 22 - Implementacja warstwy manual handlers*

```
  "water_pump": {
    "gpio": 22,
    "vpin": 0
  },
  "air_pump": {
    "gpio": 23,
    "vpin": 1
  },
  "lights": {
    "gpio": 24,
    "vpin": 2
  },
```

*Rys. 23 - Konfiguracja wirtualnych pinów*

## Warstwa automation handlers

Jest to warstwa z którą komunikuje się serwer blynk i przekazuje mu wszystkie akcje automatycznego sterowania urządzeniem. Wirtualne piny, których później używa się przy projektowaniu aplikacji na smartfona sczytywane są z pliku konfiguracyjnego.

W tej warstwie wyróżnia się 4 akcje:

- wybór profilu uprawy
- aktywacja automatycznej uprawy
- reset uprawy
- pobranie informacji o statusie uprawy

W czasie trwania pętli automatyzacji, dane na temat przebiegu uprawy zapisywane są na bieżąco w pliku konfiguracyjnym. Dane te później można odczytać z poziomu aplikacji.

```
class AutomationHandlers:
  profiles = {}
  tmp_profile_name = None

  @staticmethod
  def initialize(autoponica):
    blynk = autoponica.blynk

    AutomationHandlers.refresh_profile_list(autoponica)
    blynk.sync_virtual(autoponica.app_settings.profile_selection["vpin"], autoponica.app_settings.activation_switch["vpin"])

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.profile_selection["vpin"])
    def profile_choose_action(value):
      if not AutomationHandlers.update_profile_from_config(autoponica):
        profile_index = int(value[0]) - 1
        if profile_index >= 0:
          AutomationHandlers.tmp_profile_name = list(AutomationHandlers.profiles.values())[profile_index]

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.activation_switch["vpin"])
    def activation_switch_action(value):
      if "1" in value:
        if AutomationHandlers.tmp_profile_name is not None:
          autoponica.config.settings["automation"]["name"] = AutomationHandlers.tmp_profile_name
          autoponica.config.settings["automation"]["started_at"] = time.time()
          AutomationHandlers.tmp_profile_name = None

        if autoponica.config.settings["automation"]["name"] is None:
          blynk.notify("Select profile first!")
          blynk.virtual_write(autoponica.app_settings.activation_switch["vpin"], 2)
          return
        AutoponicaAutomationLogic.activate(autoponica)
      else:
        AutoponicaAutomationLogic.deactivate(autoponica)
      autoponica.config.save()

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.reset_button["vpin"])
    def reset_button_action(value):
      autoponica.config.settings["automation"]["name"] = None
      autoponica.config.settings["automation"]["active"] = False
      AutoponicaAutomationLogic.deactivate(autoponica)
      autoponica.config.save()

      AutomationHandlers.refresh_profile_list(autoponica)

    @blynk.VIRTUAL_WRITE(autoponica.app_settings.info_button["vpin"])
    def info_event(value):
      if autoponica.config.settings["automation"]["name"] is not None:
        blynk.notify(
          f'Profile name: {autoponica.config.settings["automation"]["name"]}\nStarted at:    {autoponica.config.settings["automation"]["started_at"]}')
      else:
        blynk.notify("Cannot get info because cycle not started!")
```

*Rys. 24 - Implementacja warstwy automation handlers*


*Rys. 25 - Konfiguracja wirtualnych pinów*

## Warstwa hardware controller

Jest to warstwa która odpowiada za komunikację z fizycznymi pinami w mikrokomputerze, do których podłączone są odpowiednie komponenty urządzenia.

Ponadto, dla każdego komponentu, stworzone zostały abstrakcyjne kontrolery które umożliwiają zarządzanie poszczególnym komponentem.

```
class HardwareController:

  def __init__(self, app_settings, config: RefreshingConfig):
    self.__water_pump_component = WaterPumpController(app_settings.water_pump["gpio"], config)
    self.__air_pump_component = AirPumpController(app_settings.air_pump["gpio"], config)
    self.__lights_component = LightsController(app_settings.lights["gpio"], config)

  def get_water_pump(self):
    return self.__water_pump_component

  def get_air_pump(self):
    return self.__air_pump_component

  def get_lights(self):
    return self.__lights_component
```

*Rys. 26 - Implementacja warstwy hardware controller*

```
class AirPumpController(HardwareSwitchableComponent):

  def handle(self, value):
    print("Air pump event")
    if value[0] == '1':
      self._component.on()
    else:
      self._component.off()
```

*Rys. 27 - Implementacja kontrolera dla pompy powietrza*

## Warstwa automation logic

Warstwa ta odpowiada za całą logikę związaną z automatycznym zarządzaniem uprawą. Na podstawie wczytanego profilu wykonywane są wszystkie kroki określone w danym profilu uprawy. Profil podzielony jest na na okresy. W każdym okresie, można określić odpowiednio interwały przepływu wody, napowietrzania oraz czas w którym oświetlenie powinno być włączone. Metoda step wywoływana jest w pętli automatyzacji.

```python
@staticmethod
def step(autoponica):
    if AutoponicaAutomationLogic.profile is not None and AutoponicaAutomationLogic.last_refresh + 1 < time.time():
        for phase in AutoponicaAutomationLogic.profile["phases"]:
            if autoponica.config["started_at"] + phase["since"] > datetime.now():
                AutoponicaAutomationLogic.current_phase = phase

    if AutoponicaAutomationLogic.current_phase["water_pump"]["when"] > datetime.now().time().hour > \
        AutoponicaAutomationLogic.current_phase["water_pump"]["when"] + \
        AutoponicaAutomationLogic.current_phase["water_pump"]["duration"]:
        ManualHandlers.hardware_controller.get_water_pump().on()
    else:
        ManualHandlers.hardware_controller.get_water_pump().off()

    if AutoponicaAutomationLogic.current_phase["air_pump"]["when"] > datetime.now().time().hour > \
        AutoponicaAutomationLogic.current_phase["air_pump"]["when"] + \
        AutoponicaAutomationLogic.current_phase["air_pump"]["duration"]:
        ManualHandlers.hardware_controller.get_air_pump().on()
    else:
        ManualHandlers.hardware_controller.get_air_pump().off()

    if AutoponicaAutomationLogic.current_phase["lights"]["when"] > datetime.now().time().hour > \
        AutoponicaAutomationLogic.current_phase["lights"]["when"] + \
        AutoponicaAutomationLogic.current_phase["lights"]["duration"]:
        ManualHandlers.hardware_controller.get_lights().on()
    else:
        ManualHandlers.hardware_controller.get_lights().off()

    AutoponicaAutomationLogic.last_refresh = time.time()
```

*Rys. 28 - Implementacja logiki obsługi profilu*

```json
{
  "name": "Groszek",
  "phases": [
    {
      "since": "0 day",
      "water_pump": {
        "when": "12:00",
        "duration": "5 hour"
      },
      "air_pump": {
        "when": "12:00",
        "duration": "5 hour"
      },
      "lights": {
        "when": "08:00",
        "duration": "12 hour"
      }
    },
    {
      "since": "3 days",
      "water_pump": {
        "when": "08:00",
        "duration": "12 hour"
      },
      "air_pump": {
        "when": "08:00",
        "duration": "12 hour"
      },
      "lights": {
        "when": "06:00",
        "duration": "15 hour"
      }
    },
```

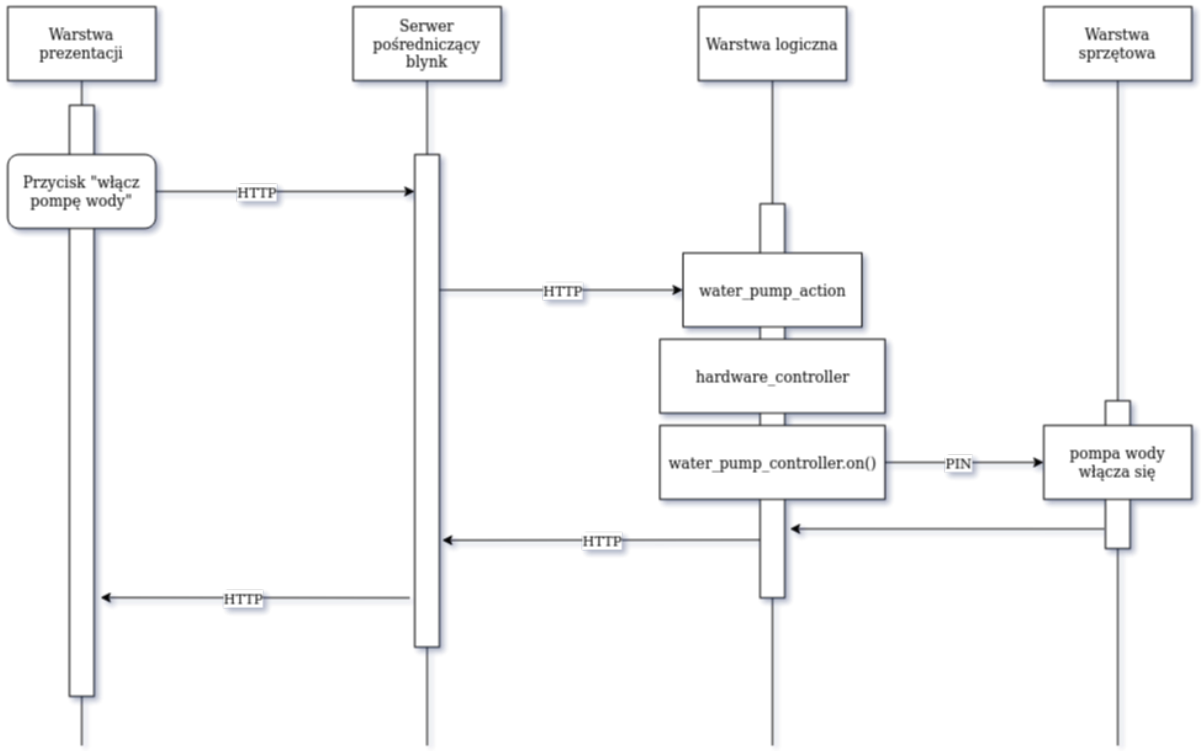
*Rys. 29 - Fragment profilu uprawy*

## 4.3. Diagramy sekwencji

W podrozdziale tym przedstawione zostały diagramy sekwencji jednej wybranej akcji w trybie manualnym oraz automatycznym.

### Diagram sekwencji trybu manualnego

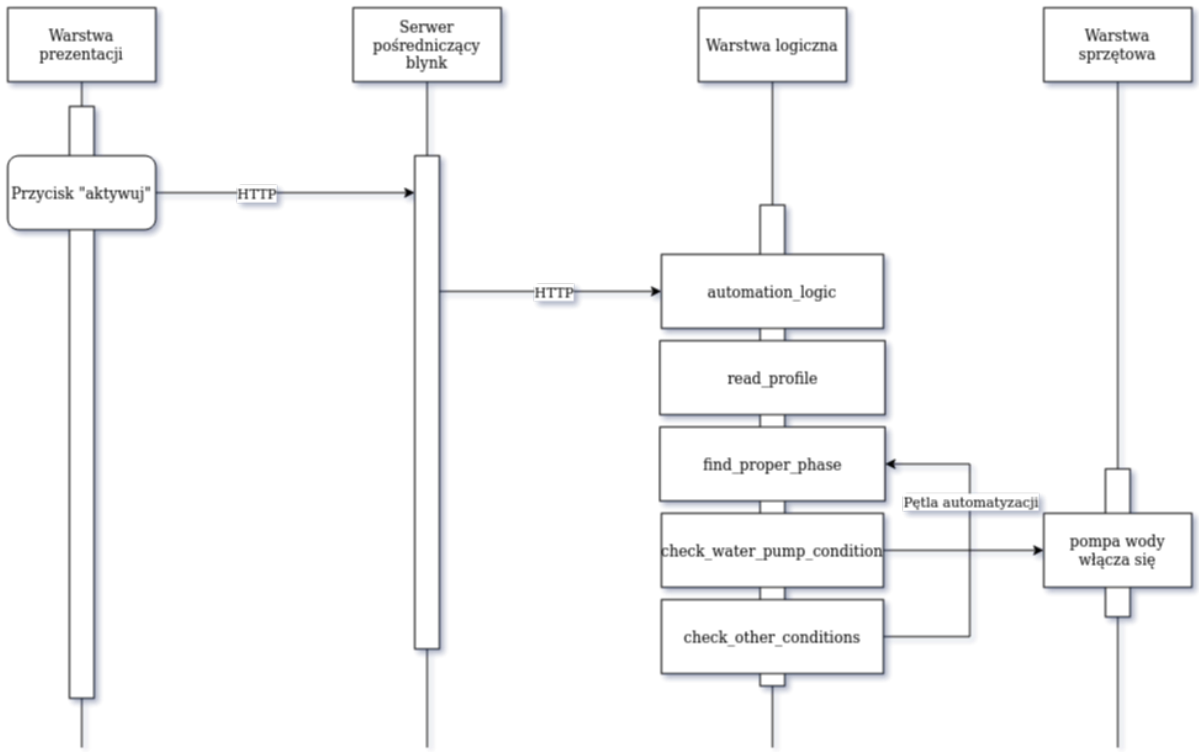
W aplikacji która pełni rolę warstwy prezentacji, po wciśnięciu guzika który reprezentuje akcję włączenia pompy wody, wysyłane jest żądanie protokołem HTTP do serwera pośredniczącego. Serwer pośredniczący posiada informację o tym gdzie znajduje się warstwa logiczna i za pomocą protokołu HTTP wysyła żądanie o wysłanie sygnału HIGH na określony wirtualny pin przypisany do przycisku. Warstwa logiczna posiada informację który kontroler przypisany jest do danego wirtualnego pinu i za jego pośrednictwem wysyła sygnał HIGH na pin w mikrokomputerze. Pompa podłączona poprzez przekaźnik do danego pinu włącza się.




*Rys. 30 - Diagram sekwencji trybu manualnego*

## Diagram sekwencji trybu automatycznego

W aplikacji która pełni rolę warstwy prezentacji, po wciśnięciu guzika który reprezentuje akcję aktywacji trybu automatycznego powinien zostać wczytany wybrany wcześniej profil z listy dostępnych. Następnie z profilu powinna zostać wybrana odpowiednia faza wzrostu rośliny na podstawie aktualnego dnia uprawy. Z wybranej fazy wzrostu powinien zostać odczytany przedział godzinowy dla każdego urządzenia. Gdy aktualna godzina znajdzie się w przedziale godzinowym, urządzenie przypisane do tego przedziału powinno się aktywować. Jeżeli godzina wyjdzie poza przedział, urządzenie powinno zostać wyłączone.




*Rys. 31 - Diagram sekwencji trybu automatycznego*

# 5. Projekt warstwy prezentacji

W tym rozdziale opisany został projekt warstwy prezentacji. Warstwa ta reprezentowana jest przez aplikację stworzoną w kreatorze platformy blynk.io. Kreator udostępnia wiele możliwości, począwszy od prostych akcji włącz/wyłącz, poprzez prezentację wyników w formie wykresów, aż po integrację z wieloma platformami lub serwisami mediów społecznościowych.

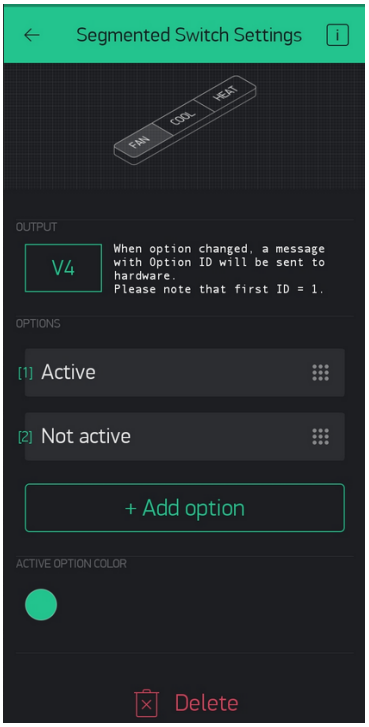
## 5.1. Kreator aplikacji platformy blynk.io

W tym podrozdziale opisany został kreator aplikacji platformy blynk.io. Zaprezentowane zostały opisane wybrane formatki oraz sposób ich konfiguracji.

### Segmented switch

Segmented switch jest to formatka reprezentująca przełącznik. Dla tej formatki można zdefiniować dowolną ilość stanów, ich opis a także kolor dla aktualnego stanu. Można ją przypisać do wirtualnego pinu. W tym przypadku pinu “V4”. Na ten pin po wykonaniu akcji zmiany stanu formatki zostanie wysłana informacja na temat tego jaki stan aktualnie ona przyjęła.


*Rys. 32 - Formatka segmented switch*

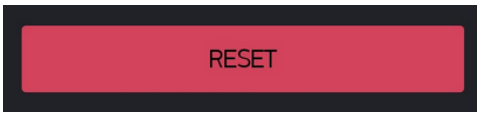



*Rys. 33 - Ekran edycji atrybutów formatki segmented switch*

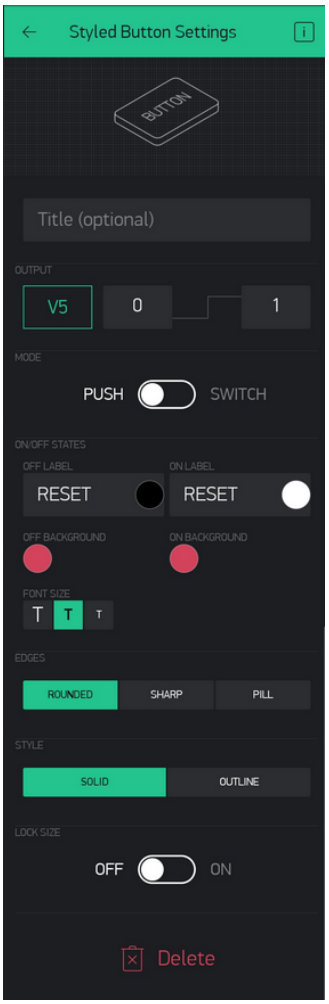
## Styled button

Styled button jest to formatka reprezentująca przycisk, któremu można ustawić określony styl. Przycisk ten może dwa stany. Można określić wartość dla każdego ze stanu, a także nazwę widoczną na przycisku. Przycisk może działać w jednym z dwóch trybów: Push(naciśnij/puść) lub switch(on/off).

W przycisku można ustalić parametry stylu takie jak: tło, wielkość czcionki, kształt. Przypisać go można do wirtualnego pinu. W tym przypadku pinu "V5". Na ten pin po wykonaniu akcji zmiany stanu formatki zostanie wysłana informacja na temat tego jaki stan aktualnie ona przyjęła.




*Rys. 34 - Formatka styled button*

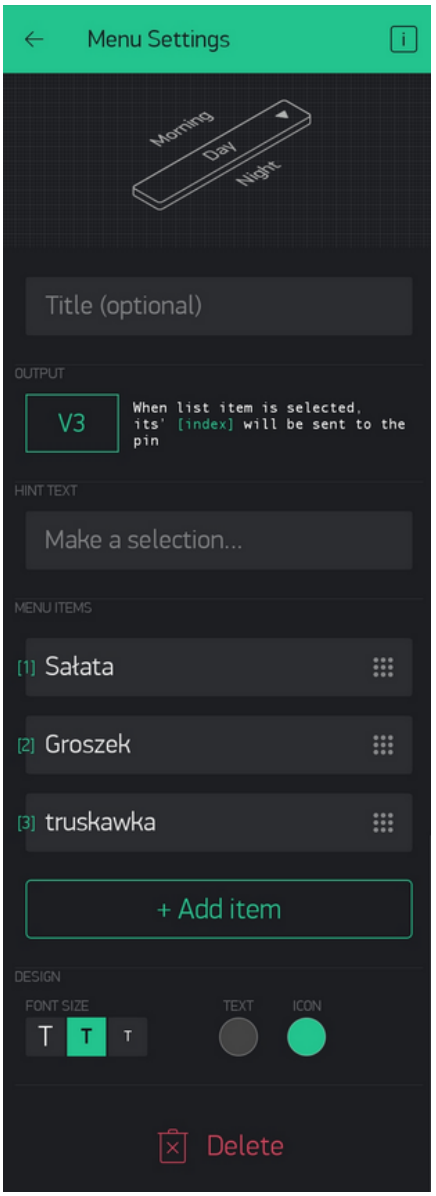



*Rys. 35 - Ekran edycji atrybutów formatki styled button*

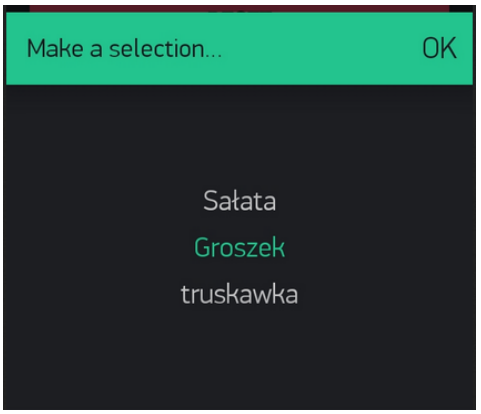
## Menu

Menu jest to formatka reprezentująca listę rozwijaną. Dla tej formatki można zdefiniować dowolną ilość stanów, ich opis a także kolor formatki. Przypisać ją można do wirtualnego pinu. W tym przypadku pinu “V3”. Na ten pin po wykonaniu akcji zmiany stanu formatki zostanie wysłana informacja na temat tego jaki stan aktualnie ona przyjęła.


*Rys. 36 - Formatka menu*




*Rys. 37 - Ekran edycji atrybutów formatki menu*



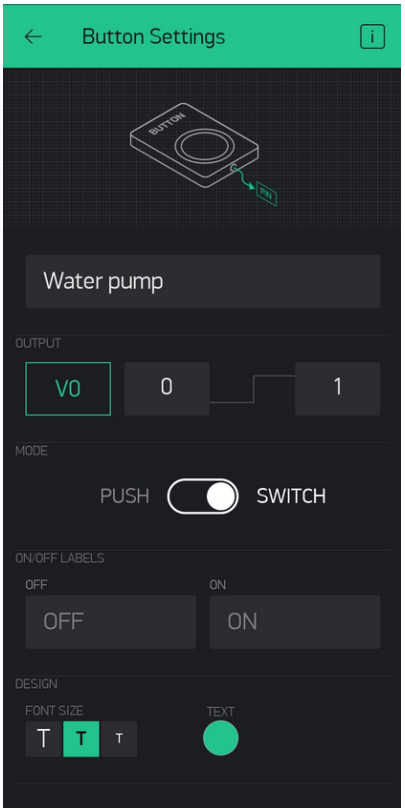

*Rys. 38 - Rozwinięta lista*

## Button

Button jest to formatka reprezentująca przycisk. Przycisk ten może dwa stany. Można określić wartość dla każdego ze stanu, a także nazwę widoczną przycisku. Przycisk może działać w jednym z dwóch trybów: Push(naciśnij/puść) lub switch(on/off).

W przycisku można ustawić tło oraz wielkość czcionki. Przypisać go można do wirtualnego pinu. W tym przypadku pinu "V0". Na ten pin po wykonaniu akcji zmiany stanu formatki zostanie wysłana informacja na temat tego jaki stan aktualnie ona przyjęła.


*Rys. 39 - Formatki button w obu stanach*

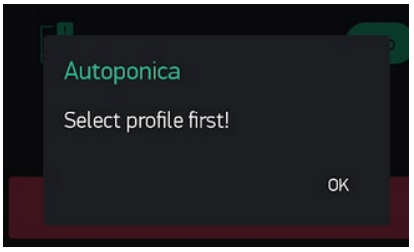



*Rys. 40 - Ekran edycji atrybutów formatki button*

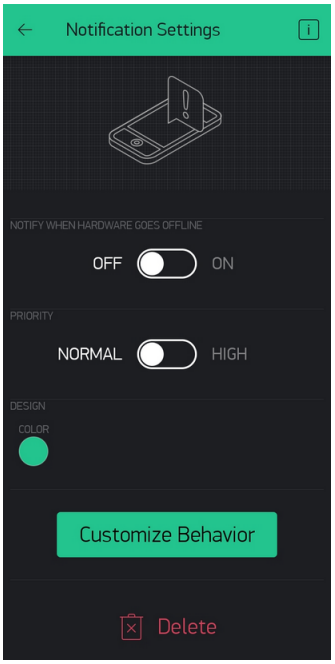
## Notification

Notification jest to formatka odpowiedzialna za odbieranie notyfikacji z warstwy logicznej do warstwy prezentacji. Z warstwy logicznej można wysłać notyfikację o dowolnej treści. Wyświetli się ona w warstwie prezentacji. Dla tej formatki można ustawić czy powiadamiać o wyłączeniu urządzenia i priorytet notyfikacji, a także jej kolor.

*Rys. 41 - Formatka notification*



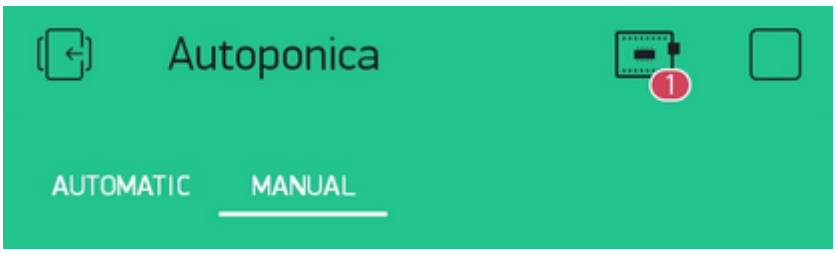

*Rys. 42 - Poglądowe powiadomienie z warstwy logicznej*



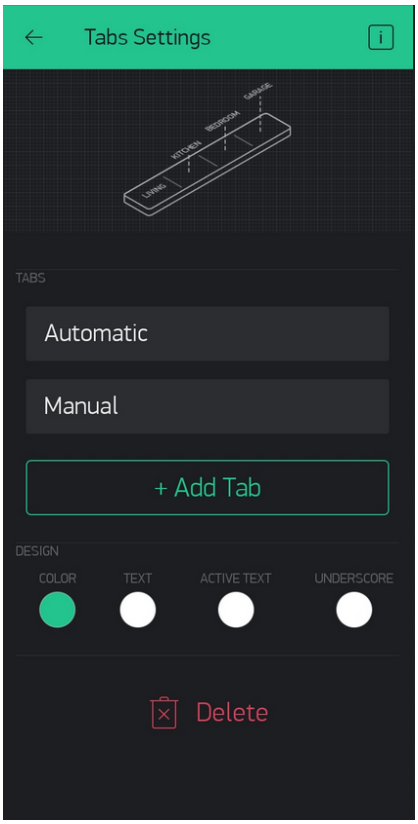

*Rys. 43 - Ekran edycji atrybutów formatki notification*

## Tabs

Tabs jest to formatka która pozwala się przełączać pomiędzy zakładkami. Za jej pomocą można stworzyć kilka różnych ekranów, gdyż umożliwia ona bardzo łatwe przełączanie się pomiędzy nimi. Można w niej dodać nieskończoną liczbę zakładek. Każda z zakładek reprezentuje osobny ekran.




*Rys. 44 - Formatka tabs*




*Rys. 45 - Ekran edycji atrybutów formatki tabs*

## 5.2. Warstwa prezentacji

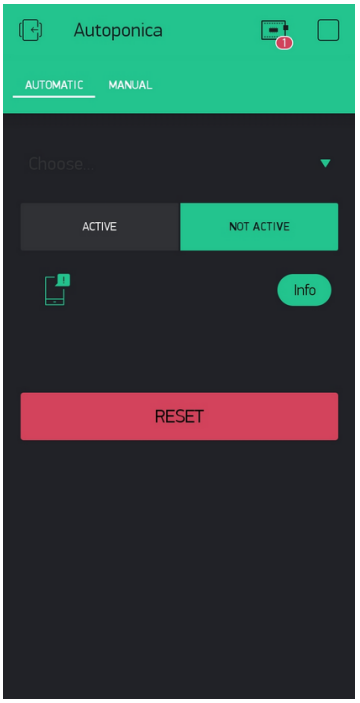
W tym podrozdziale przedstawiona i opisana została warstwa prezentacji którą reprezentuje aplikacja stworzona w kreatorze opisany w podrozdziale 5.1. Aplikacja posiada dwa ekrany, które można przełączać za pomocą formatki tabs. Na pierwszym ekranie zostały umieszczone wszystkie funkcjonalności związane z automatyzacją działania urządzenia. Na drugim ekranie umieszczone zostały wszystkie funkcjonalności związane z manualnym sterowaniem urządzeniem.

### Ekran automatycznego sterowania urządzeniem

Na ekranie tym umieszczone zostały wszystkie funkcjonalności związane z automatyzacją działania urządzenia.

Można wyróżnić takie funkcjonalności jak:

- Wybór profilu uprawy z listy rozwijanej
- Aktywacja bądź dezaktywacja trybu automatycznego
- Pobranie informacji na temat stanu uprawy
- Całkowity reset cyklu uprawy

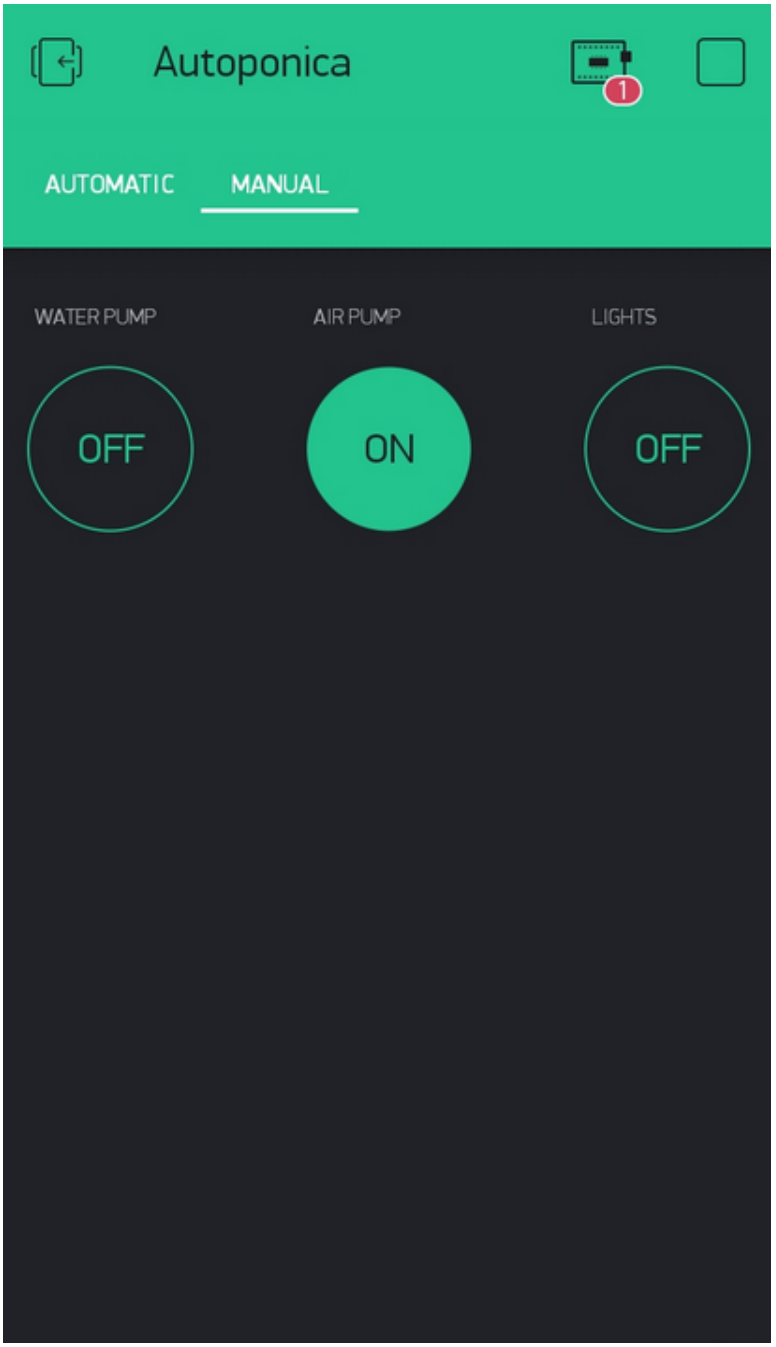



*Rys. 46 - Ekran automatycznego sterowania urządzeniem*

## Ekran manualnego sterowania urządzeniem

Na ekranie tym umieszczone zostały wszystkie funkcjonalności związane z manualnym sterowaniem urządzeniem. Można wyróżnić takie funkcjonalności jak:

- Włącz/wyłącz pompę wody
- Włącz/wyłącz pompę powietrza
- Włącz wyłącz oświetlenie




*Rys. 47 - Ekran manualnego sterowania urządzeniem*

# 6. Testy

W rozdziale tym zostały opisane testy dwóch trybów w których jest w stanie działać urządzenie. Oprócz komponentów które znalazły się w urządzeniu została także podłączona taśma led o długości 4 metrów. Taśma o takiej długości pobiera 56W mocy. Urządzenie więc potrzebuje zasilacza który jest w stanie dać prąd który jest sumą poniższych komponentów:

| **Nazwa komponentu** | **Pobór prądu** |
|---|---|
| Pompa wody | 8W |
| Pompa powietrza | 8W |
| Raspberry PI WH Zero | 1W |
| Zespół przekaźników | 2W |
| Przetwornica 12VDC-5VDC | 0.5W |
| Taśma LED 4m | 57W |
| **Suma: 76,5W** | |

*Tabela 2 - Przybliżony pobór prądu poszczególnych komponentów*

Dla takiego poboru mocy zastosowany został zasilacz o wydajności prądowej **8A** i napięciu **12V**. Wybór wyżej wymienionego zasilacza jest wynikową zastosowania prostego działania:

**76.5W / 12V = 6,3A+20% tolerancji zasilacza = 7.65A**

## 6.1. Testy trybu manualnego

W tym podrozdziale zamieszczony został opis testów trybu manualnego. Test trwał **24 godziny**. Test polegał na uruchomieniu wszystkich komponentów urządzenia i sprawdzeniu jaki jest pobór mocy. Pomiar został wykonany za pomocą watomierza **GB-202**. Pobór prądu dla tego okresu wyniósł **1.91 kWh** przy aktywowanych wszystkich komponentach w urządzeniu.

## 6.1. Testy trybu automatycznego

W tym podrozdziale zamieszczony został opis testów trybu automatycznego. Test trwał **7 dni**. Test polegał na stworzeniu przykładowego profilu uprawy który zakładał:

- Nieprzerwane działanie pompy wody w całym okresie trwania testu
- Aktywowanie pompy wody co pół godziny
- Aktywne oświetlenie od godziny 8 rano do godziny 20.

Pomiar został wykonany za pomocą watomierza **GB-202**. Pobór prądu dla tego okresu wyniósł **8.12 kWh** przy powyższych ustawieniach.


*Rys. 48 - Watomierz GB-202*

# 7. Wnioski końcowe

Celem niniejszej pracy inżynierskiej było stworzenie kompleksowego rozwiązania, które usprawni obsługę uprawy hydroponicznej. Aby zrealizować cel wykonany został szereg prac związanych z zaprojektowaniem a później wdrożeniem rozwiązania.

Rozwiązanie składa się z trzech głównych elementów takie jak:

- Warstwa sprzętowa(Fizyczne urządzenie połączone z uprawą)
- Warstwa logiczna(Oprogramowanie na urządzeniu realizujące logikę związaną z manualnym oraz automatycznym zarządzaniem uprawą)
- Warstwa prezentacji(Aplikacja pozwalająca realizować wszystkie funkcjonalności zaprogramowane w warstwie logicznej)

Projektowanie rozwiązania problemu poparte zostało dogłębną analizą z zakresu, elektroniki, pneumatyki, hydrauliki oraz wytwarzania oprogramowania łącząc tym samym ze sobą wiele dziedzin z którymi spotykamy się na co dzień. Cały system sprawdzi się wyśmienicie w gospodarstwie domowym, ale także w rozwiązaniach przemysłowych, gdzie zastąpi wiele prac związanych z uprawą i pielęgnacją roślin.
Rozwiązanie zostało oparte o platformę IoT(Internet of things) co pozwala zarządzać uprawą z dowolnego miejsca na świecie.
Rozwiązanie problemu cechuje się także stosunkowo niską ceną wykonania co obrazuje kosztorys zawarty w niniejszej pracy.

Zrealizowany projekt inżynierski podyktowany był zwiększającą się popularnością upraw hydroponicznych. Niestety, większość z tych rozwiązań nie oferują kompaktowych rozmiarów do zastosowania w gospodarstwie domowym, lub są zwyczajnie za drogie. Praca ta nie wyczerpuje tematu tego typu urządzeń.
W trakcie analizy powstał także szereg nowych pomysłów które mogłyby usprawnić opisane wyżej rozwiązanie.
Zaliczają się do nich:

- Monitorowanie wzrostu za pomocą kamer
- Monitorowanie poziomu nawozu w wodzie
- Automatyczne dozowanie nawozu na podstawie odczytów
- Automatyczne dostosowywanie wysokości oświetlenia do fazy wzrostu

# Literatura

**[1] Kopieniactwo – Wikipedia, wolna encyklopedia**
https://pl.wikipedia.org/wiki/Kopieniactwo

**[2] Maszyna rolnicza – Wikipedia, wolna encyklopedia**
https://pl.wikipedia.org/wiki/Maszyna_rolnicza

**[3] Rolnictwo — najważniejsze daty z historii - Encyklopedia PWN - źródło wiarygodnej i rzetelnej wiedzy**
https://encyklopedia.pwn.pl/haslo/Rolnictwo-najwazniejsze-daty-z-historii;2233707.html

**[4] Hydroponika - nowatorski i kontrowersyjny sposób uprawy roślin doniczkowych**
https://fajnyogrod.pl/porady/hydroponika-nowatorski-i-kontrowersyjny-sposob-uprawy-roslin-doniczkowych/

**[5] Raspberry Pi: historia minikomputera**
https://www.komputerswiat.pl/poradniki/sprzet/raspberry-pi-historia-minikomputera/3wrd656

**[6] Złącza Wago - poznaj asortyment złączy przemysłowych**
https://www.conrad.pl/artykuly/guides/z%C5%82acza-wago

**[7] Getting started with Raspberry Pi - Introduction | Raspberry Pi Projects**
https://projects.raspberrypi.org/en/projects/raspberry-pi-getting-started

**[8] Przetwornica USB 5V ładowarka Step down 12-24V 3A - Sklep internetowy AGD i RTV - Allegro.pl**
https://sklep-elektronik.pl/zasilacze-i-ladowarki-samochodowe/zasilacz-5v-3a-usb-montazowy-przetwornica-dc-dc-12-24v.html

**[9] Pompa membranowa - silnik R385+ - 12V - 3W - mini pompa wody**
https://abc-rc.pl/pl/products/pompa-membranowa-12v-3w-silnik-r385-mini-pompa-wody-9788.html

**[10] Moduł przekaźników 4 kanały z optoizolacją - styki 10A/250VAC - cewka 5V Botland - Sklep dla robotyków**
https://botland.com.pl/przekazniki-przekazniki-arduino/2579-modul-przekaznikow-4-kanaly-z-optoizolacja-styki-10a-250vac-cewka-5v-5904422330996.html

**[11] Blynk IoT platform: for businesses and developers**
https://blynk.io/

**[12] Build Physical Projects With Python on the Raspberry Pi – Real Python**
https://realpython.com/python-raspberry-pi/

**[13] Garden Guides | The Advantages of Hydroponics Over Conventional Farming**
https://www.gardenguides.com/98252-advantages-hydroponics-over-conventional-farming.html

**[14] Jalkrishi Automated Hydroponics System**
https://www.cdac.in/index.aspx?id=pe_aee_AutomatedHydroponicsSystems

**[15] Deep water cultivation - Fruit & Vegetables - Viscon Group**
https://viscongroup.eu/expertise/deep-water-cultivation/

**[16] Automation and Robotics Used in Hydroponic System | IntechOpen**
https://www.intechopen.com/chapters/70662

**[17] Fully Automated Hydroponic Indoor Grow Box | Seedo**
https://www.seedolab.com/#how_it_works

**[18] Hydroponics: forecasted market value worldwide 2016-2025**
https://www.statista.com/statistics/879946/global-hydroponics-market-value/

**Igor Maculewicz**
The West Pomeranian Business School, Poland

***Improvement of hydroponic cultivation with use of Raspberry PI microcomputer and control software written in Python language***

The main goal of the engineers thesis is to discuss the issues that occur in hydroponic cultivation. The work describes the problems associated with manual hydroponic cultivation. It also includes an analysis of current solutions that improve hydroponic farming and the concept of a project that implements automatization to this type of cultivation. The thesis contains a detailed description of the device design, that is based on Raspberry PI microcomputer and describes the working concept of software that automatically supports the cultivation processes managed by smartphone application.