POLITECHNIKA KRAKOWSKA IM.
TADEUSZA KOŚCIUSZKI
WYDZIAŁ FIZYKI MATEMATYKI
I INFORMATYKI
KIERUNEK FIZYKA TECHNICZNA

MACIEJ BOROWIEC

INTERFEJS WEBOWY DRUKARKI 3D REPRAP PRUSA I3 OPARTY NA SERWERZE NODE.JS I MIKROKOMPUTERZE RASPBERRY PI 3

Web of interface for 3D Printer RepRap Prusa i3 based on NodeJS server and Raspberry Pi 3

PRACA MAGISTERSKA

STUDIA STACJONARNE

Ocena:

Podpis promotora:

Promotor: *Dr Radosław Kycia*

Kraków 2018

Składam serdeczne podziękowania dla:

Promotora:Pana Dr Radosława Kyci

Pana Prof. Andrzeja Woszczyny

Pana Prof. Wojciecha Otowskiego

Spis treści

Streszczenie

W niniejszej pracy opisano proces konstrukcji zdalnego interfejsu sterowania drukarką 3D. Wykorzystana drukarka to urządzenie Prusa i3 zmontowane i uruchomione w ramach mojej inżynierskiej pracy dyplomowej.

Do realizacji interfejsu zdalnego sterowania wykorzystano szereg technologii webowych oraz mikrokomputer Raspberry Pi 3 B+. Głównym modułem projektu jest aplikacja oparta o nowoczesną technologię NodeJS. Moduł ten służy do kontrolowania komunikacji z drukarką. W wewnętrznym procesie tej aplikacji uruchomiona jest darmowa biblioteka Pronsole, odpowiedzialna za nawiązanie połączenie i wymianę danych z drukarką 3D. Dane do aplikacji dostarczane są poprzez API w technologii RestAPI. Na zdalnym serwerze umieszczony jest system CMS, który komunikuje się aplikacją NodeJS.

Nieodłącznym elementem projektowanego interfejsu jest przestawka do sterowania zasilaniem drukarki. Umożliwia ona, odłączenie lub podłączenia zasilania do drukarki poprzez akcję wywołaną ze zdalnego panelu.

W pracy opisano sposób działania całego sytemu jak i poszczególnych modułów. Spisano cele i założenia projektowanej aplikacji. Na końcu, w krótkim podsumowaniu, spisano wady, zalety i otrzymane wnioski.

Linki do projektu:

https://github.com/mb92/RRWI.git

https://github.com/mb92/RRWI---App-NodeJS-.git

Obraz skonfigurowanego Raspbian (wraz z plikami aplikacji i szkicem układu elektronicznego):

https://mega.nz/#F!11wB0K4a

Wstęp

Druk przestrzenny to ostatnimi laty bardzo popularna, dynamicznie rozwijana zarówno przez hobbystów jak i profesjonalistów dziedzina technologii. Powstają coraz to nowsze konstrukcje drukarek, głowic drukujących, a także materiałów drukarskich. Nieustannie dąży się do uzyskania jak najlepszych efektów, przy możliwie niskim koszcie i oszczędności czasu.

Innymi szybko zmieniającymi się dziedzinami technologii jest elektronika i branża IT (ang. Internet Technology). Połączenie obu daje olbrzymie możliwości przy tworzeniu wszelkich projektów, a przede wszystkim nowych technologii – wydajniejsze maszyny mogą sprostać większym wymaganiom aplikacji. Korzystając z ogólnodostępnych w obecnej chwili technologii i urządzeń można tworzyć ciekawe i wymagające systemy, które jeszcze kilka lat temu były by nie do zrealizowania, lub ich budowa była by kompletnie nieopłacalna. Idea polega na zdalnym nadzorze urządzeń lub wymianie danych zbieranych z sensorów przez globalną sieć Internet. Ta obecnie modna dziedzina nazywa się Internetem Rzeczy (ang. IoT – Internet of Things).

Przykładem takiego przedsięwzięcia jest projekt opisany w niniejszej pracy dyplomowej, stanowiący połączenie użycia różnych technologii webowych, w tym głównie NodeJS, pracującej na nowoczesnym i wydajnym mikrokomputerze Raspberry Pi 3 B+. Opisany dalej projekt realizowany jest z myślą o druku 3D, w celu stworzenia zdalnego interfejsu umożliwiającego kontrolowanie i monitorowanie pracy drukarki 3D.

Cel i zakres pracy

Celem pracy jest stworzenie aplikacji internetowych umożliwiających zdalne kontrolowanie działania drukarki 3D z wykorzystaniem mikrokomputera Raspberry Pi 3B +.

Praca jest o charakterze badawczo - aplikacyjnym.

Zakres pracy obejmuje:

- uruchomienie mikrokomputera Raspberry Pi,
- instalacja i uruchomienie potrzebnego oprogramowania i bibliotek,
- uruchomienie kamerki dedykowanej do Raspberry,
- analiza dostępnego w sieci kodu podobnym interfejsem[1],
- stworzenie aplikacji w technologii NodeJS[2] umożliwiającej sterowanie drukarką poprzez API[3],
- stworzenie prostego systemu CMS[4] do sterowania drukarką z wykorzystaniem języków programowania PHP i JavaScript,
- zaprojektowanie, montaż oraz integracja z powstałym systemem zewnętrznej przystawki do sterowania zasilaniem drukarki,
- przeprowadzenie testów systemu na poziomie sieci lokalnej.

[1] Kod udostępniony na otwartej licencji, odnośniki w bibliografii [1] i [2]

[2] NodeJS - środowisko uruchomieniowe, technologia oparta na języku JavaScript. Więcej informacji w rozdziale 4.3.5

[3] API - ang. Appication Interface

[4] CMS - ang. Content Manager System, aplikacja sieciowa do zarządzania treścią na stronie internetowej.

CZĘŚĆ TEORETYCZNA

1. Druk 3D - zarys technologii

W rozdziale pierwszym opisano ogólnie pojęcie druku przestrzennego. Znajdują się tu podstawowe informacje na temat powstania samej koncepcji, materiałów do druku oraz zasady działania drukarki pracującej w technologii FDM, na przykładzie drukarki RepRap Prusa i3.

1.1. Na czym polega druk 3D?

Historia druku 3D sięga swoją historią roku 1984. Wynalazcą technologii druku przestrzennego jest *Charls Hull* [3]. Opatentował on swoją technologię w 1986 roku. Przez ostatnie kilka lat obserwujemy dynamiczny rozwój tej dziedziny, spowodowany wygaśnięciem patentu i rozwijaniem technologii przez hobbystów. W 2006 roku stworzono koncepcję otwartego projektu RepRap. Pomysłodawcą był *Adrian Bawyer* [3]. Ideą projektu jest budowa samoreplikujących się urządzeń do druku 3D i darmowym udostępnianiu tych projektów w społeczności drukarzy amatorów [3].

Urządzenia RepRap pracują w technologii osadzania topionego materiału - FDM[5]. Materiał to żyłka wykonana z tworzywa sztucznego (filament[6]),

[5] FDM - ang. *Fused Deposition Modeling*, spotyka się również oznaczenie FFF od ang. Fused Filament Fabrication [4].

[6] filament - nazwa materiału drukarskiego w postaci żyłki (włókna) z tworzywa sztucznego [4].

na przykład z ABS[7], PLA[8] lub nylonu. Zostaje roztopiona w odpowiednio rozgrzanej głowicy i wyciśnięta przez niewielką dyszę.
Temperatura głowicy zależy od użytego materiału. Przykładowo dla ABS jest to zakres temperatury od 230 ^{o}C do 250 ^{o}C. [3, 4]

Model do wydruku musi zostać „pocięty" na cienkie warstwy przy wykorzystaniu odpowiedniego oprogramowania (np. *Slicer*, *Cura*). Następnie generowany jest kod maszynowy (gcode) zrozumiały dla drukarki. Zawiera on zestaw poleceń potrzebny do wydrukowania modelu. Proces drukowania jest czasochłonny i nie zawsze opłacalny [4]. Mimo to druk 3D znajduje szerokie zastosowanie w przemyśle, jako metoda szybkiego prototypowania. Często jest to praktyka bardziej opłacalna niż przestrajanie maszyn do seryjnej produkcji [4, 6].

Więcej informacji na temat druku 3D można znaleźć w mojej pracy dyplomowej [3].

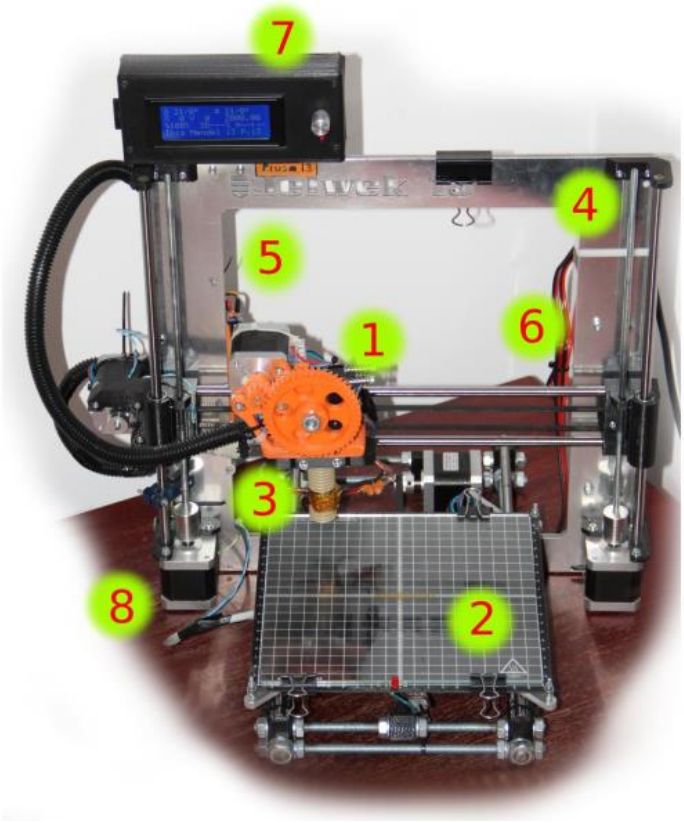
[7] ABS - ang. *Acrylonitrile Butadiene Styrene*, jeden z najbardziej popularnych materiałów wykorzystywany w druku przestrzennym. Składa się z akrylonitrylu, butadienu i styrenu. Jest toksyczny [4, 5].

[8] PLA - ang. *polyactic acid* - poliaktyd, popularne włókno wykonane na bazie kwasu mlekowego, biodegradowalny [4, 5].

1.2. Zasada działania drukarki 3D na przykładzie *RepRap Prusa i3*

W opracowaniu wykorzystano drukarkę zbudowaną podczas pracy inżynierskiej.

Jest to Prusa i3 widoczny na zdjęciu poniżej:



Rysunek 1: Drukarka RepRap Prusa i3 wykorzystywana w badaniach [3].

Drukarka składa się z:

1 - wózek z głowicą i ekstruderem[9] poruszającym się wzdłuż osi X

2 - stół grzewczy na którym powstaje wydruk, pracuje wzdłuż osi Y

3 - głowica drukująca, wersja *7h7 mod. E*[10]

[9] ekstruder - mechanizm do podawania (wciskania) filamentu do rozgrzanej głowicy [3, 6].

[10] 7h7 mod. E - polska głowica drukująca [4].

4 - mechanizmy prowadzenia elementów os X wzdłuż osi Z

5 - moduł sterowania drukarki *RAMPS 1.4*[11]

6 - zasilacz

7 - panel sterowania wraz z czytnikiem kart pamięci SD

8 - silniki krokowe NEMA 17

[3]

Pliki z kodem maszynowym można przesyłać na bieżąco z komputera poprzez złącze USB lub zapisać je na karcie pamięci, a następnie uruchomić poprzez panel sterowania drukarki. To drugie rozwiązanie jest zalecane, ponieważ drukarka 3D działa niezależnie od zewnętrznych urządzeń, co zmniejsza ryzyko błędnego odczytu pliku [3, 7].

Po dostarczeniu pliku z kodem wydruku następuje rozgrzewanie stołu i głowicy. Wszystkie osie są sprowadzane do pozycji początkowej. Po wykonaniu tych czynności drukarka przystępuje do wydruku, zgodnie z ustawieniami dla danego modelu. Po zakończonym wydruku urządzenie jest automatycznie zatrzymywane i wychładzane [3, 7, 8].

[11] RAMPS 1.4 - ang. *RepRap Arduino Mega Pololu Shiel*, wersja 1.4, dedykowany moduł sterowania do drukarek 3D oparty o kontroler *Arduino MEGA* [3, 4].

Rysunek 2: Zestaw części do drukarki 3D wydrukowanych na innym urządzeniu tego typu [8].

Więcej informacji na temat budowy, zasady działania i konfiguracji drukarki 3D można znaleźć w opracowaniu wymienionym w punkcie 1.1, pozycji [3].

2. Charakterystyka mikrokomputera Raspberry Pi 3

W tym rozdziale opisano mikrokomputer Raspberry Pi 3. Poruszona została kwestia ogólnej budowy i zasady działania tegoż urządzenia. Pokazano kilka przykładowych zastosowań Raspberry w życiu codziennym, co obrazuje ogromy potencjał tego niewielkiego komputera.

2.1. Co to jest Raspberry Pi?

Raspberry Pi to mikrokomputer o szerokim spektrum zastosowań. Umożliwia podłączenie monitora, myszki, klawiatury i innych urządzeń peryferyjnych. Obsługiwany jest przez system Linux [9]. Dla początkujących użytkowników zalecana jest specjalna dystrybucja Raspbian, bazująca na *Debianie*[12], zoptymalizowana pod Raspberry Pi. Wersja ta jest nieco bardziej okrojona niż oryginalny *Debian*, ale zawiera potrzebne oprogramowanie taki jak: przeglądarka internetowa, narzędzia do programowania w *Pythonie*, *Mathematica*, GUI[13], terminal, menadżer plików, przeglądarka obrazów i wiele innych. [10]. System operacyjny instalowany jest na zewnętrznej karcie pamięci SD[14], co pozwala na szybką reinstalację systemu lub posiadanie kilku wersji Raspbiana i sprawne posługiwanie sie nimi [9]. Kompilacja Linuksa dla Raspberry zawiera domyślnie powłokę graficzną LXDE[15]. Jest to dość atrakcyjna i estetyczna powłoka graficzna oferująca interfejs sterowania touchpadem lub myszką. Oprócz preinstalowanego oprogramowania, można instalować dodatkowe aplikacje i biblioteki, w zależności od potrzeby. Służy do tego linuksowy menadżer pakietów - apt [10].

Rysunek 3: Raspberry Pi 3 B

Raspberry posiada również piny GPIO[16] - piny wejścia i wyjścia, do których można podłączać różne czujniki i elementy elektroniczne. Zachowanie złącza *GPIO* można

[12] Debian - jedna z najstarszych i najmniej wymagających dystrybucji Linux [10]. Link do oficjalnej strony systemu w pozycji [11].

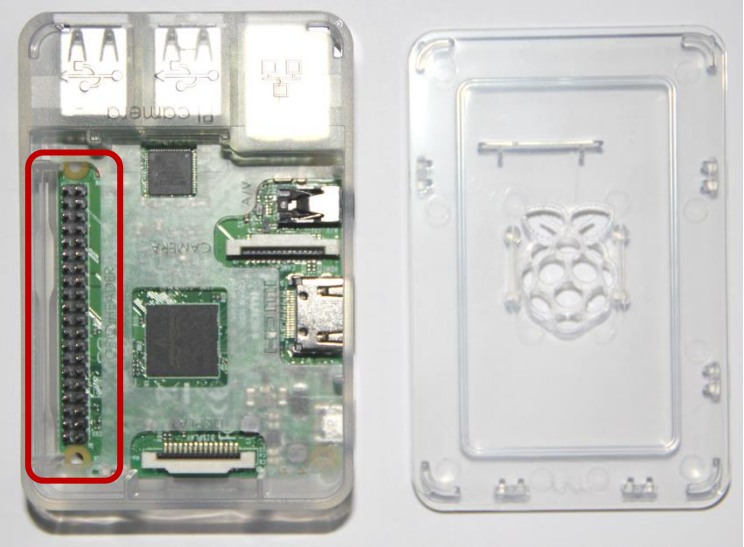
[13] GUI - ang. Graphical User Interface, graficzny interfejs użytkownika.

[14] Karta pamięci SD - w zależności od wersji jest to standardowa karta SD lub microSD.

[15] LXDE – ang. Lightweight X11 Desktop Environment.

[16] GPIO - ang. General Purpose Input/Output.

dowolnie oprogramowywać własnymi skryptami *Pythona* czy *C* [9] i dzięki temu wykorzystywać je do komunikacji z zewnętrznymi urządzeniami elektronicznymi.



Rysunek 4: Raspberry Pi B3 z widocznymi po lewej pinami GPIO

2.2. Specyfikacja Raspberry Pi 3 B

Procesor	BROADCOM BCM2387 1.2 GHz Quad Core (4 rdzenie) 64-bitowy ARM-8 Cortex-A53
Pamięć RAM	1 GB
Karta pamięci	Micro SD
GPIO	40 pinów
USB	4 gniazda
Złącza graficzne	HDMI[17] - gniazdo monitora LCD[18] - złącze ekranu Kamera - złącze dedykowanej kamerki
Komunikacja	WiFI - moduł wbudowany Bluetooth 4.1 - moduł wbudowany Ethernet - RJ 45 I2C[19] - magistrala synchroniczna UART[20] - magistrala asynchroniczna SPI[21] - interfejs szeregowy
Audio	Mini jack
Zasilanie	Mikro USB, 5V 1A DC
Obsługiwane systemy operacyjne	Linux, Android, Windows
Otwory montażowe	4 otwory

Tabela 1: Specyfikacja mikrokomputera Raspberry PI 3B [10, 11, 12, 13]

[17] HDMI - ang. *High Definition Multimedia Interface* [10].

[18] LCD - ang. *Liquid Crystal Display*, wyświetlacz ciekłokrystaliczny.

[19] I2C - ang. *Inter-Integrated Circuit*, umożliwia komunikację między kilkoma układami scalonymi, w Raspberry Pi dostępna na pinach numer 3 i 5 [10].

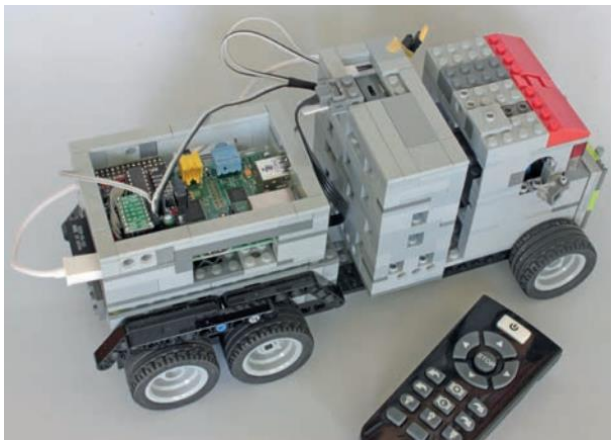
[20] UART - ang. *Universal Asynchronus Receiver/Transmitter*, prosty, dwuliniowy interfejs do komunikacji szeregowej. Umożliwia m.in. śledzenie komunikatów jądra systemu Linuks. W razie awarii może być pomocnym narzędziem diagnostycznym [10].

[21] SPI - ang. *Serial Peripheral Interface*, cztero-liniowa magistrala szeregowa stworzona do programowania mikrokontrolerów. Umożliwia komunikację z wieloma urządzeniami. W RPi dostępna jest poprzez piny 19, 21, 23, 24 i 26 [10].

Rysunek 5: Płyta główna RPi 3B [12]

2.3. Przykłady zastosowań Raspberry Pi

Mikrokomputer cieszy się sporą popularnością ludzi w różnym wieku. Może stanowić centrum multimedialne w obrębie domowej sieci lokalnej lub ze zwykłego telewizora zrobić zaawansowany "SMART TV" z dostępem do Internetu [9]. Równie dobrze może być komputerem produkcyjnym z aplikacjami użytkowymi, pełnić funkcję serwera stron internetowych czy po prostu działać jako mikrokontroler [10].



Rysunek 6: Samochód ciężarowy wykonany z klocków LEGO, sterowny przez Raspberry Pi [9]

3. Sieć globalna - Internet

W tym rozdziale omówiono historyczne aspekty Internetu. Nakreślono również współczesne standardy obowiązujące w wirtualnym świecie w odniesieniu do standardów z lat poprzednich.

3.1. Internet na świecie

Internet to akronim pochodzący od angielskiego sformułowania ***INTER****national* ***NET****work*. Jest to globalna, rozproszona sieć komputerowa, łącząca ze sobą inne sieci. Komputery budujące tą sieć rozlokowane są na całym świecie. Określenie sieci rozproszonej nawiązuje do tego, że nie można w niej wskazać centralnego punktu. IP (ang. Internet Protocol) to protokół, na którym oparty jest system adresowania węzłów (urządzeń podłączonych do sieci) w tej ogromnej sieci [15]. Z kolei program na danym komputerze, aby połączyć się z siecią podłącza się do gniazda (ang. sockets), które stanowi adres IP oraz numer portu (zakres 1 – 5535), który identyfikuje aplikację na danym komputerze. W ten sposób aplikacja z jednego węzła przez podanie IP i portu może komunikować się z aplikacją na innym komputerze.

Internet nie ma właściciela i zarządcy. Istnieje organizacja *interNIC*[22], która gospodaruje adresami IP - przydziela je operatorom Internetu. Operatorzy przydzielają je dalej komputerom w swoich sieciach [15, 16].

Historia Internetu zaczyna się w latach 60-tych XX wieku, kiedy to w USA powstał *ARPANet*[23]. Na początku sieć była przeznaczona wyłącznie na użytek

[22] InterNIC - ang. *Internet Network Information Center* [16].

[23] ARPANet - sieć komputerowa stworzona przez organizację ARPA (ang. *Advanced Reaserch Projects Agency*), zajmującą się unowocześnianiem armii USA. Jej celem było stworzenie systemu odpornego na uszkodzenia, który pracowałby nawet po zniszczeniu jego części [15, 16].

wojskowy. Była ona projektowana w taki sposób, aby wskutek ataku nuklearnego i zniszczenia kilku węzłów sieć mogła dalej działać. W 1970 roku odtajniono protokół adresowania TCP/IP i przyłączono niektóre uczelnie w USA do *ARPANet*- u. Trzy lata później, zaczęto wysyłać pierwsze wiadomości email, powstał protokół FTP i Telnet. Sieć globalna sięga poza Atlantyk, co umożliwiło przyłączenie Wielkiej Brytanii oraz Norwegii. Na początku lat 80-tych organizacja *ARPA* rozdzieliła część wojskową sieci od części akademickiej, która nazwana została Internetem. Od tej pory Internet zaczyna rozwijać się niezależnie [15].

W 1989 roku w ***CERNIE***[24] powstaje protokół hipertekstowy będący fundamentem stron www[25] i służył do wymiany danych pomiędzy naukowcami [15, 17]. W 1990 roku powstała pierwsza strona internetowa [18].

3.2. Internet w Polsce

W dniu 19 listopada 1990 roku Departament Obrony USA nadał Polsce pierwszy numer IP: 192.86.14.0 (obecnie ten adres posiada Akademickie Centrum Superkomputerowe Cyfronet AGH), który umożliwiał podłączenie maksymalnie 255 komputerów. 20 listopada zakończono pomyślnie instalację tegoż adresu. Tego samego dnia miedzy 10:57 a 13:25 nadano pierwszą wiadomość email *CERN*-u. Nadawcami emaila byli **dr Grzegorz Polok** i **mgr Paweł Jałocha**. Adresatem natomiast był **mgr inż. Andrzej Sobala** z Instytutu Fizyki Jądrowej Polskiej Akademii Nauk [19].

Rysunek 7: Dr Grzegorz Polok z Instytutu Fizyki Jądrowej [19]

[24] Europejska Organizacja Badań Jądrowych CERN (fr. Organisation Européenne pour la Recherche Nucléaire).

[25] www – ang. World Wide Web [16].

Wiosną 1991 roku powstała organizacja NASK[26], Polska otrzymała klasę adresową 148.81.0.0 - klasa B[27]. W 1991 roku, z instytutu Fizyki Uniwersytetu Warszawskiego nawiązano połączenie z Centrum Komputerowym Uniwersytetu w Kopenhadze [15, 19].

4. Wybrane technologie webowe

Rozdział ten zawiera podstawowe informacje na temat najbardziej popularnych technologii internetowych. Obejmuje swym zakresem informacje na temat serwera stron www i bazy danych oraz języków programowania wykorzystanych przy tworzeniu aplikacji internetowych. Spośród ogromu zagadnień wybrano kilka, tych potrzebnych do zrozumienia działania dalszej części dysertacji bądź bezpośrednio wykorzystane w części praktycznej.

[26] NASK - Naukowa i Akademicka Sieć Komputerowa założona w 1991 roku na Uniwersytecie Warszawskim. Obecnie NASK prowadzi krajowy rejestr nazw internetowych w domenie ".pl" i formalnie nadal pozostaje placówką naukowo-badawczą [15].

[27] Wyróżnia się pięć klas adresów IP: A, B, C, D, E. Adres składa się z adresu sieci o adresu hosta. Wymienione klasy prezentują liczby obsługiwanych sieci oraz hostów, Każdy adres klasy B może obsłużyć 65534 unikalnych adresów hostów [16].

4.1. Serwer stron internetowych

4.1.1. Serwer stron www - Apache

Serwer Apache HTTP jest to darmowy serwer dostępny dla szeregu systemów operacyjnych (Microsoft Windows, OS X, Linux itp.). Serwer ten cechuje się bezpieczeństwem, jest wielowątkowy i wysoce skalowalny. Ponadto oferuje szerokie możliwości konfiguracyjne. Apache jest najczęściej wybieranym serwerem stron [20].

Rysunek 8: Logo APACHE [21]

Apache napisany został przez *Roba McCoola*. W kwietniu 1995 roku ukazała się pierwsza oficjalna wersja. Konfiguracja serwera znajduje się w pliku *httpd.conf*. Podstawowy plik konfiguracyjny zawiera szereg zaawansowanych ustawień takich np. jak: środowisko serwera, parametry sieciowe, lista dołączanych modułów, aliasy, ścieżki do katalogów i wiele innych [20, 21].

4.1.2. Raspberry Pi jako serwer stron www

RPi[28] doskonale nadaje się na domowy lub firmowy serwer plików i stron www. Oczywiście nie można porównywać serwera Raspberry z superkomputerami w centrach danych, ale poprzez niski pobór prądu i cichą pracę jest doskonałą maszyną na niewielki prywatny serwer [10].

[28] Rpi – skrót od nazwy Raspberry Pi.

Do publikowania zasobów sam serwer Apache nie wystarczy. Potrzebny jest zestaw podstawowego oprogramowania do obsługi języka PHP[29] i bazy danych MySQL[30] [10]. Wyżej wymienione pojęcia zostaną omówione w kolejnym punkcie 4.2.

Podsumowując, do poprawnego i użytecznego działania serwera www potrzebny jest tzw. *stos LAMP*[31] *(Linux Apache MySQL PHP)*. Instalacja *LAMP* na Raspberry Pi jest stosunkowo prosta. Wykonuje się ją np. poprzez terminal systemowy, wpisując poniższe komendy [22]:

1. Aktualizacja listy pakietów	`apt-get update && apt-get upgrade`
2. Instalacja apache	`apt-get install apache2`
3. Instalacja MySQL (w trakcie instalacji zostanie wyświetlona prośba o ustawienie hasła dla użytkownika root do serwera MySQL)	`apt-get install mysql-server mysql-client`
4. Instalacja interpretera PHP:	`apt-get install php5 php5-mysql libapache2-mod-php5`

Tabela 2: Zestaw poleceń do instalacji pakietu LAMP na Raspberry Pi [22]

[29] PHP - skryptowy język programowania, wykonywany po stronie serwera. Więcej informacji w punkcie 4.2 [10].

[30] MySQL - system zarządzania bazami danych. Więcej informacji w punkcie 4.2 [10].

[31] Wersja dla Windows – WAMP.

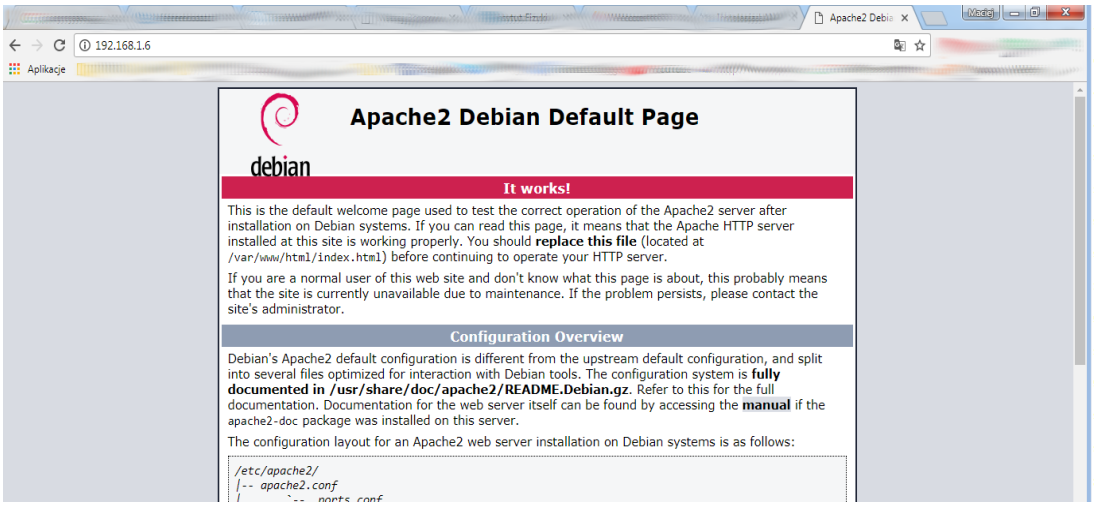
Instalacja zajmuje kilka minut. Efekty można zobaczyć wpisując w okno przeglądarki Raspberry adres 127.0.0.1[32] lub adres IP Malinki[33] przypisany w sieci lokalnej, w przeglądarce innego hosta tej samej sieci [10]. Pliki strony www domyślnie znajdują się w katalogu */var/www/html/*, do którego możemy dostać się wpisując w terminalu:

```
cd /var/www/html
```

Pliki i katalogi znajdujące się w tym folderze, powinny mieć uprawnienia użytkownika *www-data*[34]. Nadaje się je (rekurencyjnie dla katalogu i wszystkich jego podkatalogów – opcja R) używając poniższego polecenia:

```
sudo chown -R www-data /var/www/html/<katalog>
```

[10]



Rysunek 9: Zrzut ekranu uruchomionego serwera na Raspberry - widok z hosta w sieci lokalnej. W tym przypadku serwer Apache na Raspberry dostępny jest pod adresem IP: 192.168.1.6

[32] Jest to tak zwany „loopback" - pętla zwrotna służąca do wymiany danych wewnątrz komputera. Dzięki temu, żadne pakiety nie wydostają się na zewnątrz, co jest dobrym rozwiązaniem do testowania aplikacji.

[33] Malinka – potoczna nazwa Raspberry Pi.

[34] www-data - konto, z którego korzysta linuxowy Apache [10].

Serwer obsługujemy poniższym zestawem poleceń:

Aktualny stan serwera Apache (status)	`sudo service apache2 status`

Restart, zatrzymanie i uruchomienie Apache:	`sudo service apache2 restart`
	`sudo service apache2 stop`
	`sudo service apache2 start`

Analogicznie postępujemy w przypadku serwera bazy danych:	`sudo service mysql status`
	`sudo service mysql restart`
	`sudo service mysql stop`
	`sudo service mysql start`

Tabela 3: Zestaw komend przydatnych w pracy z serwerami

Zagadnienia związane z serwerami internetowymi to bardzo rozległa dziedzina. Powyższy opis to tylko prosty przykład tego, jak szybko udostępnić stronę w sieci.

4.2. Języki programowania aplikacji internetowych

4.2.1. HTML, XHTML i CSS

Rysunek 10: Oficjalne logo HTML 5 - www.w3c.org

Skrót HTML pochodzi od angielskiej nazwy *Hypertext Markup Language*, czyli język znakowania hipertekstu. Jest on podstawą każdej strony internetowej. Opisuje on tylko układ/kompozycję strony. Strona internetowa (najprostsza) jest zazwyczaj plikiem tekstowym napisanym w języku HTML o rozszerzeniu *html* lub *htm* [24]. W 1997 roku HTML 4 był oficjalnie rekomendowany przez organizację W3C[35]. Obecnie powszechnie stosowany jest HTML 5.

[35] W3C - *http://www.w3.org*, organizacja określająca standardy tworzenia stron internetowych pod kątem jakości kody. Teoretycznie kod strony spełniającej warunki W3C będzie poprawnie interpretowany przez wszystkie przeglądarki internetowe.

Celem wersji piątej było:

- wprowadzenie nowych elementów interaktywnych na stronę www,
- wprowadzenie nowych znaczników przy zachowaniu kompatybilności wstecz (dodano m.in. sekcję nagłówka, stopki, artykułu, nawigacji, odtwarzacze audio i video itp.),
- bardziej szczegółowe określenie sposobu obsługi błędów,
- interakcja z rozszerzeniami dodanych niegdyś przez producentów przeglądarek do ich produktów.

[25]

Język XHTML to rozszerzona wersja HTML. Skrót pochodzi od angielskiego zwrotu *Extended HTML*[24]. Rozszerza on standardy zwykłego HTML, tak aby były zgodne z XML[36]. Dokumenty XHTML-owe są zgodne ze składnią XML, co daje możliwość prostego przeglądania i walidowania kodu przez narzędzia XML [25]. Wedle obowiązujących standardów, w dokumencie XHTML mogą występować wyłącznie elementy składające się ze znacznika otwarcia i zamknięcia [24].

Poniżej przykład:

```
<body>
    <p>Lorem ipsum</p>
</body>
```

[36] XML - ang. Extensible Markup Language, czyli rozszerzalny język oznaczania [25].

HTML daje niewielkie możliwości co do zmiany wyglądu poszczególnych elementów strony internetowej. Z pomocą przychodzą tutaj style CSS, czyli kaskadowe arkusze stylów (ang. Cascade Style Sheets). Wprowadzenia arkuszy stylów ujednoliciło po raz kolejny standardy tworzenia stron www i dało szerokie pole manewru ich twórcom [24].

Rysunek 11: Logo CSS 3 - www.w3c.org

CSS to język dedykowany do formatowania elementów strony internetowej. Mimo, że stworzono odpowiednie standardy, nie wszystkie przeglądarki potrafią poprawnie interpretować atrybuty css-owe. W praktyce zaleca się testowanie kodu na różnych przeglądarkach. Głównym celem kaskadowych arkuszy jest oddzielenie warstwy strukturalnej od warstwy prezentacji [25].

Style CSS można wykorzystać na trzy sposoby:

- styl wpisany (tzw. "inline'owy") - umieszczany bezpośrednio w tagu html, niezalecany,
- styl osadzony - formatowanie definiowane w postaci klasy, definicje umieszczone w pliku html,
- arkusz zewnętrzny - klasy formatowania umieszczone w zewnętrznym pliku css, dołączane w nagłówku pliku html, zalecane.

[24, 25]

Pozostaje jeszcze niewyjaśnione pojęcie "kaskadowe". Otóż wynika to z zasady ich działania - każdy element podrzędny dziedziczy formatowanie elementu nadrzędnego [25].

4.2.2. JavaScript

Rysunek 12: Logo JS - www.javascript.com

JavaScript (JS) to stosunkowo prosty język stworzony przez firmę *NetScape* z myślą o przeglądarkach internetowych [24]. Wykorzystywany jest do tworzenia elementów aktywnych na stronach internetowych, czyli takich, które reagują na zdarzenia generowane przez użytkownika (naciśnięcie przycisku, wpisanie wartości, przesunięcie kursora). Kod w JavaScript załączany jest do pliku html i wykonywany przez przeglądarkę (po stronie klienta). Kod JavaSctipt lub lokalizacja pliku tekstowego z kodem umieszczona jest w tagach `<script></script>` [26].

JavaScript często nazywany jest jako Java. Otóż jest to niepoprawne stwierdzenie, gdyż Java to odrębny język programowania rozwijany przez firmę *Sun Microsystems*, służący do tworzenia aplikacji [26]. Był to chwyt marketingowy mający na celu spopularyzowanie tego języka. Podobnie jest z JScript. Został opracowany przez *Microsoft* i jest rozszerzeniem międzynarodowego standardu opartego o JavaScript. Można go uruchamiać wyłącznie w przeglądarce *Internet Explorer* i *Windows Scripting Host* dostępna na systemach Windows [26, 27].

4.2.3. PHP

Rysunek 13: Oficjalne logo języka PHP [29]

PHP oficjalnie oznacza ang. Hypertext Preprocessor, a w oryginale brzmiał ang. Personal Home Page - osobista strona domowa [28]. Autorem pierwszej wersji PHP z 1994 roku był *Rasmus Lerdorf* [29]. Kolejne wersje tego języka rozwijane były przez Izraelczyków. *Zeev Suraski i Andi Gutmans* zainteresowali się

projektem w 1997 roku. Zdecydowali przepisać kod od nowa, gdyż pierwotna wersja miała niewielki możliwości użytkowe [29].

Najnowszą wersją standardu języka jest wersja 7.2. Jest to najczęściej stosowany język do tworzenia skryptów wykonujących się po stronie serwera [29]. Wykorzystując język PHP, strona staje się "żywa", jej treść jest dynamiczna. Przez to, że skrypty PHP wykonywane są całkowicie po stronie serwera, jest on całkowicie niewidoczny dla klienta, który tylko wysyła żądanie i otrzymuje odpowiedź w postaci gotowego, wygenerowanego pliku html [30]. Dzięki swej modularnej budowie możliwa jest budowa samodzielnych aplikacji z interfejsem graficznym [29].

PHP jest całkowicie darmowy i dostępny na większości systemów operacyjnych. Jest kompatybilny z wieloma serwerami [30]. W porównaniu ze swymi konkurentami (Perl, Microsoft ASP.NET, Ruby, JSP oraz Cold Fusion) język PHP jest bardzo wydajny, skalowalny, posiada interfejsy do wielu systemów bazodanowych. Posiada znaczną ilość wbudowanych bibliotek przydatnych do wielu zadań. Jest językiem zorientowanym obiektowo [31].

Dodatkowym atutem jest bogata dokumentacja, dostępna pod adresem [32].

4.2.4. Python

Jest to interaktywny, interpretowany język programowania wysokopoziomowego. Twórcą był Guido van Rossuma w 1990 roku [10, 30]. Od samego początku Python

Rysunek 14: Oficjalne Logo języka Python - www.python.org

cieszył się dużą popularnością [10]. Dysponuje w pełni dynamicznym systemem typów i zautomatyzowanym zarządzaniem pamięcią [32].

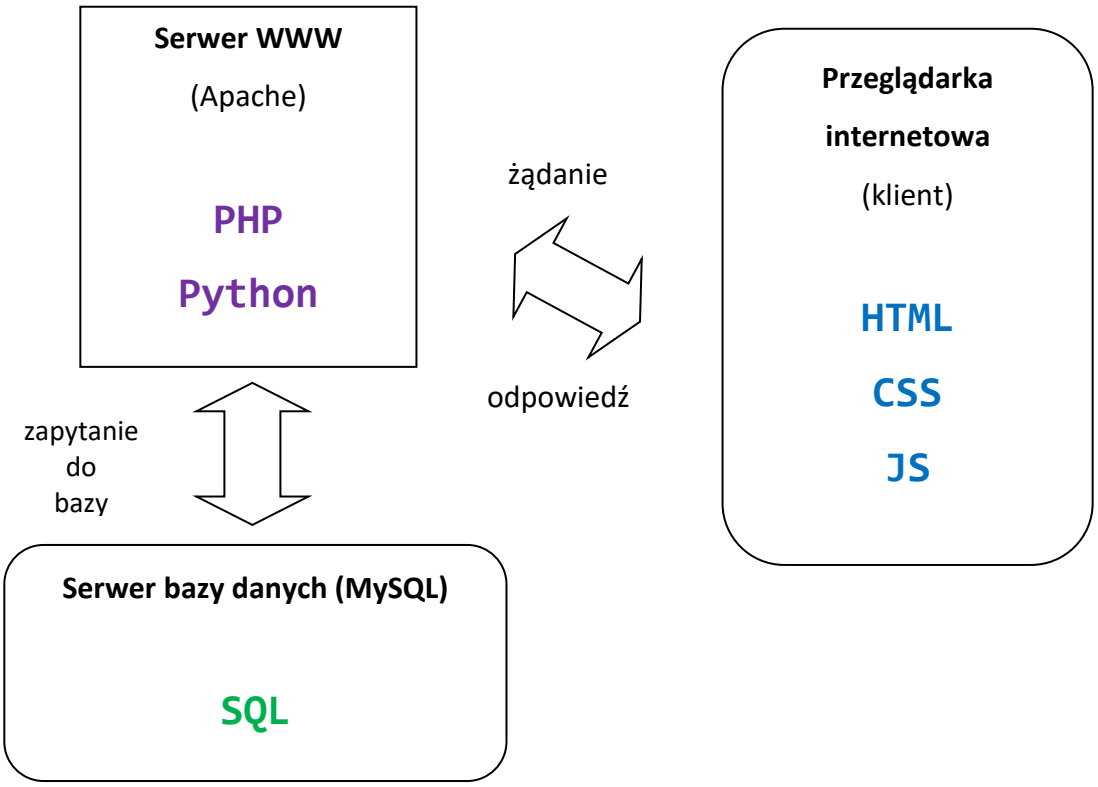
Kod Pythona jest łatwy do pisania i czytania oraz intuicyjny. Miejscami przypomina zdania w języku angielskim. Język ten jest oficjalnie polecanym przez fundację *Raspberry Pi Foundation* do nauki programowania [10].

Python jest dostępny za darmo, na zasadach licencji *Open Source*[37]. Przez to że jest multiplatformowy, kod napisany na maszynie o określonej konfiguracji *isystemie* operacyjnym będzie mógł być uruchomiony na maszynie z inną konfiguracją [10, 33].

4.2.5. Podsumowanie

Na poniższym schemacie zobrazowano w/w technologie i sposób ich współdziałania:

[37] Open Source - licencja, na mocy której za darmo rozpowszechniany jest kod aplikacji.



[35]

4.3. Wybrane technologie tworzenia aplikacji webowych

4.3.1. Co to jest framework?

Framework, inaczej zwany platformą programistyczną to swego rodzaju szkielet aplikacji. Definiuje wewnętrzną strukturę aplikacji wraz z zarysem mechanizmu przepływu informacji pomiędzy komponentami. Zawiera zestaw użytecznych komponentów i bibliotek ogólnego stosowania. Programista w trakcie tworzenia aplikacji rozbudowuje i modyfikuje komponenty dostarczone przez framework oraz dodaje własne komponenty i moduły [41].

Praca z wykorzystaniem frameworka (w szczególności praca w dużym zespole programistów) jest wygodnym rozwiązaniem, ponieważ wszyscy pracują na jednym zbiorze pewnych trywialnych, powtarzających się rozwiązań. Obowiązują pewne standardy, chociażby związane z architekturą aplikacji. Każdy

z programistów stosuje podobny schemat działania przy tworzeniu nowych modułów [41].

Poniżej zestawiono zalety i wady frameworku, jako odrębnego oprogramowania:

Zalety	**Wady**
Efektywność *(mniejsza ilość kodu do napisania)*	Złożoność *(poprzez elastyczność mają dość złożoną budowę)*
Lepsza jakość kodu *(są elastyczne, a przez to posiadają dobrą logikę wewnętrzną)*	Wydajność *(elastyczność kodu obniża jego wydajność)*
Niezawodność *(są dobrze zaprojektowane, przetestowane i odpowiednio zabezpieczone)*	Nie zawsze posiadają dobrą dokumentację.

Reasumując:

FRAMEWORK	=	Rozwiązania powtarzających się problemów	+	Wzór na budowanie nowych funkcjonalności

[42]

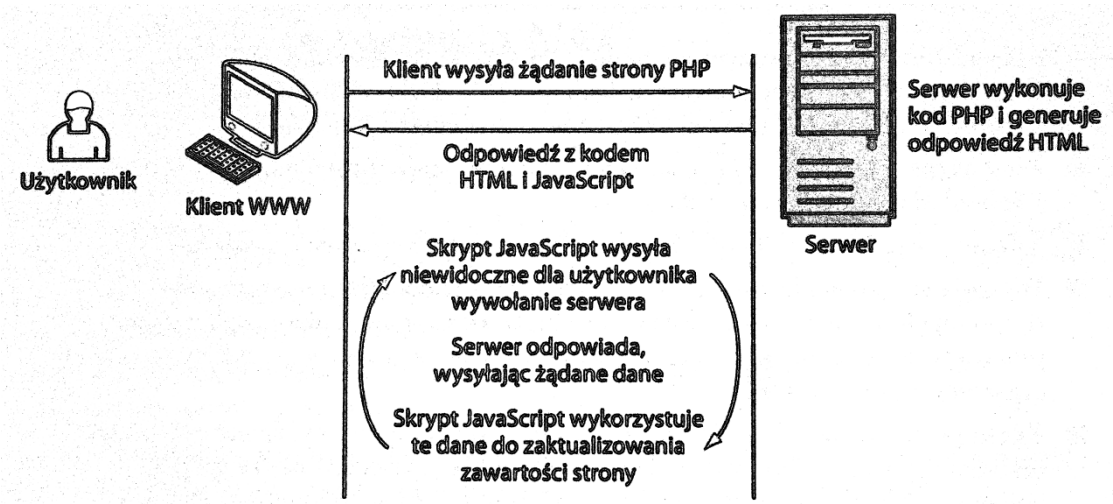
4.3.2. jQuery

jQuery to szybka, niewielka i bogata w funkcje biblioteka, napisana w języku JavaScript [43]. Pierwsza stabilna wersja została wydana w sierpniu 2006 roku [44]. Obsługa zdarzeń, manipulowanie elementami HTML, animacje oraz żądania Ajax są dużo prostsze w obsłudze, niż w sposób klasyczny. Oprócz tego jQuery obsługuje selektory CSS3. Interfejs jQuery jest bardzo łatwy w obsłudze i kompatybilny z wieloma przeglądarkami [43]. Niewątpliwym atutem jest jej niewielki rozmiar, gdyż wersja produkcyjna osiąga rozmiar około 80 kB.

Rysunek 15: Logo jQuery

4.3.3. Ajax

Ajax to skrót pochodzący od angielskiego zwrotu: Asynchronous JavaScript and XML, czyli asynchroniczny JavaScript i XML [26]. Jest to narzędzie umożliwiające wysłanie przy pomocy skryptu JS zapytania do serwera, z którego pobierana jest strona internetowa. Dane otrzymane z serwera przesyłane są w ten sam sposób, czyli asynchronicznie. Zaletą takiego rozwiązania jest możliwość aktualizacji wybranych fragmentów strony www bez konieczności jej przeładowania (odświeżenia) [45]. Poniżej, na schemacie przedstawiono ideę tej technologii:



Rysunek 16: Standardowe wywołanie AJAX [45]

Przy zastosowaniu synchronicznego przesyłania zapytania, klient korzystając z przeglądarki internetowej wysyła żądanie do serwera www. Następnie serwer wykonuje kod PHP i wysyła odpowiedź w postaci kodu strony, na podstawie, którego przeglądarka generuje widok strony wynikowej – następuje przeładowanie strony. Natomiast w przypadku asynchronicznej transakcji, skrypty JavaScript działając w tle, wysyłają w niewidoczny dla użytkownika sposób potrzebne zapytania. Po otrzymaniu odpowiedzi, odpowiednie skrypty JavaScript zmieniają tylko te elementy strony, dla których odnosi się otrzymana odpowiedź ze zdalnego serwera – brak przeładowania strony (zmiana tylko dla wybranych elementów strony).

Na AJAX składają się jak już wcześniej wspomniano JS i XML oraz technologie działające po stronie serwera, więc użytkownik ze swej strony nie musi instalować żadnego dodatkowego oprogramowania [45]. Wymiana danych pomiędzy klientem i serwerem odbywa się poprzez obiekt XMLHttpReques, pozwalający na wysyłanie danych parami: nazwa - wartość. Przesyłanie odbywa się metodą GET lub POST. Serwer Odpowiada wykorzystując protokół HTTP ale w takiej formie, aby kod JS go zinterpretował. Najczęściej jest to JSON lub XML [31, 45].

Wady i zalety AJAX:

Zalety	Wady
Możliwość projektowania interaktywnych aplikacji.	Istnieje ryzyko negatywnego wpływu na jakość pracy z aplikacją (spowolnienie systemu, utrudnione pozycjonowanie w wyszukiwarkach).
Wymusza tworzenie wzorców projektowych i schematów minimalizujących czas wykonania zadań.	Klient może mieć wyłączoną obsługę skryptów JavaScript co uniemożliwi działanie Ajaxa.
Wykorzystuje popularne technologie do działania.	Nie zawsze da się dodać stronę do zakładek, gdyż podczas przeładowania strony adres strony się nie zmienia bo treść podmieniana jest dynamicznie.
Nie wymusza na programiście korzystania z nowych narzędzi.	Nie działają poprawnie przyciski w przeglądarce "Poprzednia/Następna strona".

Tabela 4: Zalety i wady asynchronicznej technologii przesyłania danych [45].

4.3.4. Bootstrap

Bootstrap jest otwartym narzędziem (frameworkiem CSS) do tworzenia kodu HTML, CSS i JS. W szybki i łatwy sposób można zbudować estetyczną i responsywną (skalowalna do urządzeń mobilnych) stronę www [46, 47]. Został stworzony na potrzeby Twittera [47].

Rysunek 17: Logo Bootstrap [46]

Bootstrap bazuje na na strukturze dwunastu kolumn. Szerokość strony określana jest za pomocą klasy container oraz container-fluid [47].

4.3.5. NodeJS

Rysunek 18: Logo NodeJS

NodeJS jest to popularna platforma programistyczna (framework JS), dedykowany do łatwego tworzenia szybkich i skalowalnych aplikacji sieciowych. NodeJS korzysta z opartego na zdarzeniach, nieblokującego modelu wejścia/wyjścia, co poprawia efektywność i rozmiar aplikacji. Jest doskonałym rozwiązaniem dla aplikacji działających w czasie rzeczywistym, na wielu urządzeniach jednocześnie [38]. Nadaje się również do kompilacji zasobów, monitorowania, wykonywania skryptów czy do przygotowywania serwerów WWW. Okazało się bowiem, że JavaScript doskonale nadaje się do wykonywania po stronie serwera [48]. Framework pracuje na jednym wątku [49].

Do programowania po stronie serwera Node wykorzystuje silnik V8, będący dziełem firmy Google. Jest to maszyna wirtualna JS stosowana w przeglądarce Google Chrome. Dzięki kompilacji z wykorzystaniem V8, Node jest bardzo wydajny, ponieważ kompilacja zachodzi bezpośrednio na kodzie rodzimym[38] [38].

Z omawianym środowiskiem wiąże się spora społeczność programistów. Nieustannie powstają nowe pakiety i moduły rozszerzające możliwości tego frameworka. Można je pobrać przy pomocy pakietu NPM omawianego w punkcie 4.3.3. [49].

[38] Kod napisany, jest gotowy do wykonania.

4.3.6. Yii2

Rysunek 19: Logo frameworka - www.yiiframework.com

Yii2 pochodzi od angielskiego "Yes, it is!", co stanowi, według twórców, „odpowiedź na wszelkie pytania dotyczących tego frameworka”[39]. Yii2 jest stworzony w języku PHP, dedykowany do tworzenia zaawansowanych aplikacji sieciowych o wielkim zasięgu i dużym obciążeniu [50]. Yii2 jest platformą bazującą na architekturze/wzorcu projektowym MVC - Model- Widok-Kontroler (ang. Model - View - Controller) [51]. Podejście tego typu pozwala oddzielić od siebie logikę aplikacji, warstwę prezentacji danych oraz warstwę sterowania. Model dotyczy logiki aplikacji, umożliwia dostęp do danych (np. z bazy danych). Widok reprezentuje warstwę odpowiedzialną za wyświetlenie strony przy pomocy wygenerowanego szablonu. Kontroler natomiast należy do warstwy sterowania - wczytuje dane wejściowe użytkownika i odpowiednio na nie reaguje (generalnie wywołuje odpowiednią akcje lub widok) [50].

Yii2 posiada wiele dodatkowych rozszerzeń oraz wtyczek. Zarówno framework jak i potrzebny (i dodatkowe) biblioteki można sprawnie zainstalować przy użyciu Composer’a[40] [50, 51].

4.3.7. RESTful API

REST to akronim od angielskiego REpresentational State Transfer, czyli przesyłanie stanu reprezentacyjnego. Jest to bardziej zbiór zasad niż protokół,

[39] Czy jest szybki? bezpieczny? wydajny? czy nadaje się do mojego nowego projektu? - "Tak, jest! (Yes, it is!)".

[40] Composer – menadżer pakietów.

przesyłania danych pomiędzy przeglądarką internetową a zdalnym serwerem [52]. RESTful API (RESTful Web Api) jest usługą sieciową stworzoną na protokole HTTP [53]. Wzorzec REST-owy zapoczątkował w 2000 roku *Roy Fielding* (pracował nad specyfikacjami protokołu HTTP 1.0 i 1.1). W celu stworzenia serwera zgodnego z wzorcem REST należy zaimplementować metody HTTP, takie jak:

- POST - utworzenie dowolnego elementu,
- GET - wyświetlenie listy elementów,
- DELETE - usunięcie elementu,
- PUT - aktualizacja istniejącego elementu.

[38]

Wyżej wymienione metody to tzw. operacje CRUD[41]. Są to cztery podstawowe konstrukcje/komendy [52]. Usługi typu RESTful-owego przesyłają reprezentację zasobów. Reprezentacja ta może być w formie pliku JSON, XML lub innym. Przesyłanym zasobem może być w zasadzie wszystko (rekord danych, zakodowany obraz itp.) [52, 53].

W usłudze RESTful charakterystyczne są tzw. "eleganckie adresy URL". Analizując przykładowy adres URL można domyślać się za co są odpowiedzialne. Ułatwia to interpretację treści i usprawnia poruszanie się po API. Zawierają one tylko informacje o zasobach - bez czasownika. Cechą dobrze zaprojektowanego API jest to, że można łatwo zgadnąć jakiej metody używa dane żądanie. Poniżej przykład linków API serwisu GitHub: *https://api.github.com/users/lornajane*

[41] CRUD - ang Create, Read, Update, Delete - Utwórz, odczytaj, modyfikuj, usuń [52].

```
{
  "login": "lornajane",
  "id": 172607,
  "avatar_url": "https://avatars3.githubusercontent.com/u/172607?v=4",
  "gravatar_id": "",
  "url": "https://api.github.com/users/lornajane",
  "html_url": "https://github.com/lornajane",
  "followers_url": "https://api.github.com/users/lornajane/followers",
  "following_url": "https://api.github.com/users/lornajane/following{/other_user}",
  "gists_url": "https://api.github.com/users/lornajane/gists{/gist_id}",
  "starred_url": "https://api.github.com/users/lornajane/starred{/owner}{/repo}",
  "subscriptions_url": "https://api.github.com/users/lornajane/subscriptions",
  "organizations_url": "https://api.github.com/users/lornajane/orgs",
  "repos_url": "https://api.github.com/users/lornajane/repos",
  "events_url": "https://api.github.com/users/lornajane/events{/privacy}",
  "received_events_url": "https://api.github.com/users/lornajane/received_events",
  "type": "User",
  "site_admin": false,
  "name": "Lorna Jane Mitchell",
  "company": null,
  "blog": "http://www.lornajane.net",
  "location": null,
  "email": null,
  "hireable": true,
  "bio": null,
  "public_repos": 130,
  "public_gists": 40,
  "followers": 306,
  "following": 0,
  "created_at": "2009-12-27T15:19:20Z",
  "updated_at": "2017-09-11T19:48:45Z"
}
```

▼ General
Request URL: https://api.github.com/users/lornajane
Request Method: GET
Status Code: 304 Not Modified
Remote Address: 192.30.253.117:443

Rysunek 20: Odpowiedź API serwisu GitHub. Wyświetlono dane użytkownika *lornajane* w formacie JSON [52]

Kolejna część pracy zawiera opis wykorzystania omówionych technologii do budowy interfejsu webowego do sterowania drukarką 3D.

CZĘŚĆ PRAKTYCZNA

5. Założenia i wymagania projektu

W niniejszym rozdziale przedstawiono założenia projektowe wraz z podstawowymi wymaganiami i funkcjonalnościami interfejsu internetowego do zdalnego sterowania drukarką 3D. Kolejno omówione zostały: ogólne reguły biznesowe aplikacji , założenia niefunkcjonalne i funkcjonalne.

5.1. Ogólne założenia i cele aplikacji

Interfejs webowy (Internetowy, zdalny) ma służyć do zdalnej kontroli drukarki 3D. Nazwa projektu to „RRWI" – RepRap Web Interface. Językiem interfejsu jest język angielski. Użytkownik po zalogowaniu do interfejsu ma mieć możliwość kontrolowania samej drukarki, jak i procesu wydruku.

5.2. Wymagania niefunkcjonalne[42]

Aplikacja:

- opiera się na darmowych, ogólnodostępnych technologiach i urządzeniach,
- jest wieloplatformowa, skalowalna, niskobudżetowa,
- działa na popularnych przeglądarkach internetowych:
  - Chrome w wersji: >= 69.0.3497.81,
  - Mozilla Firefox w wersji: >= 62.0,
  - Safari w wersji: >= 11.0.2,
- bezpieczna pod względem programowymi i sprzętowym.

[42] Wymagania niefunkcjonalne – wymagania aplikacji, które są warunkowane głównie przez użytkowników oraz osoby odpowiedzialne za prowadzenie i utrzymanie kodu aplikacji. Są to przeróżne ograniczenia dla aplikacji, które nie dotyczą bezpośrednio jej funkcjonalności [54].

5.3. Wymagania funkcjonalne[43]

Aplikacja umożliwia:

- Logowanie do interfejsu za pomocą adresu email oraz domyślnego hasła. Po poprawnym logowaniu – zmianę domyślnego adresu email oraz hasła.
- Połączenie z drukarką i kontrolę nad jej pracą:
    - Poruszanie głowicą oraz stołem,
    - Ustawienie temperatury głowicy i stołu,
    - Przesłanie pliku do wydruku,
    - Uruchomienie i zatrzymanie wydruku,
    - Podgląd w czasie rzeczywistym z kamery skierowanej na drukarkę,
    - Włączenie/Wyłączenie zasilania drukarki, z wykorzystaniem dwóch różnych sposobów, działających niezależnie od siebie - przycisk zwykły i wyłączanie awaryjne,
    - Wznowienie połączenia,
    - Przetrzymywanie i zarządzanie plikami do druku z poziomu aplikacji – wysłanie na serwer, zlecenie wydruku, usunięcie,
    - Monitorowanie aktualnej temperatury stołu grzewczego i głowicy oraz stanu uruchomionej aplikacji NodeJS.
- Umożliwienie konfiguracji podstawowych parametrów potrzebnych do poprawnej pracy całego projektu.

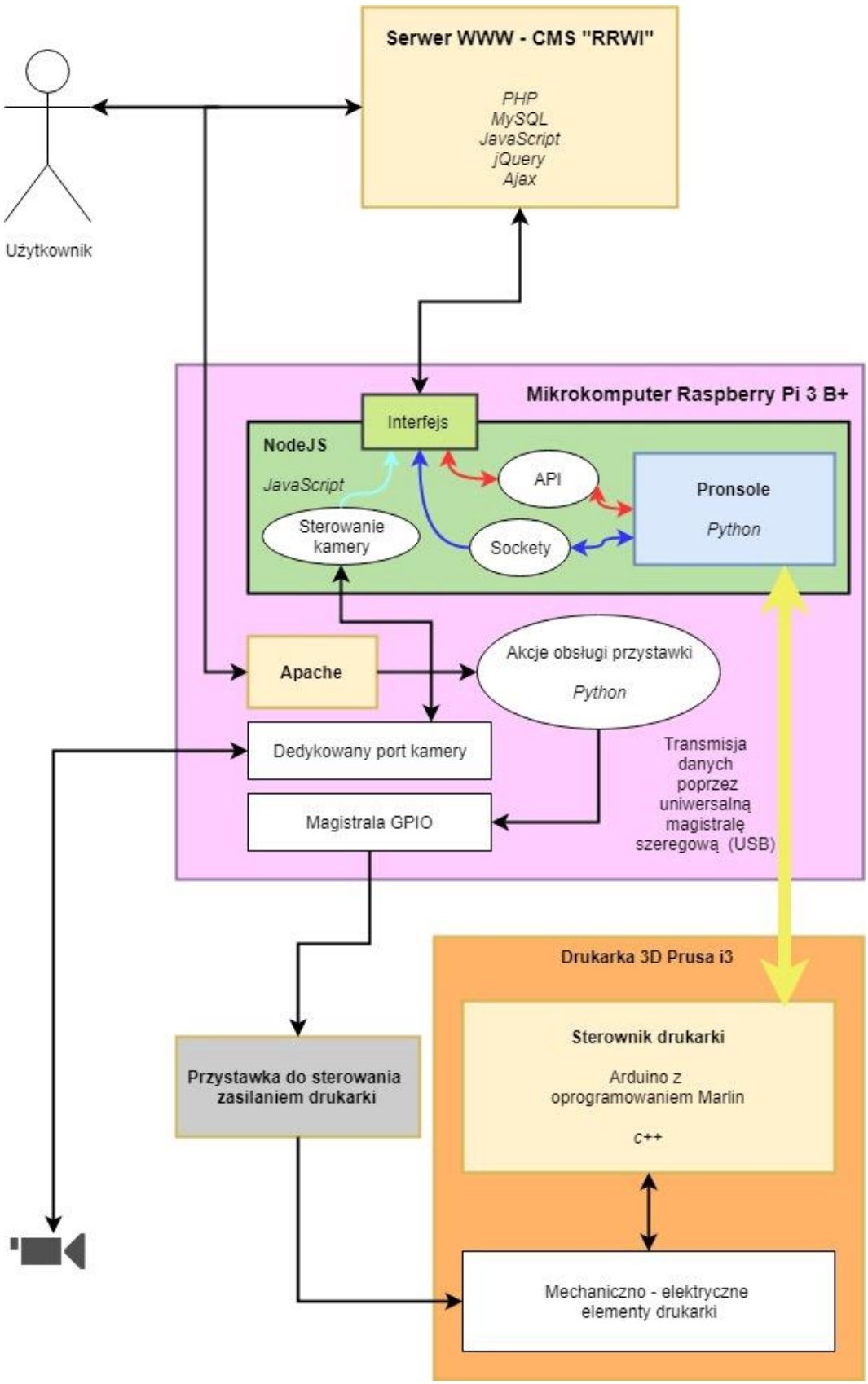
[43] Wymagania funkcjonalne – dotyczą opisu docelowego zachowania projektowanego systemu – jakie usługi ma oferować, czasem czego nie powinien robić [55].

6. Opis stworzonego interfejsu

Niniejszy rozdział zawiera opis struktury i sposobu działania całego systemu z uwzględnieniem wszystkich jego części programowych i sprzętowych. W opracowaniu użyto stosowne diagramy wraz z komentarzem.

6.1. Ogólna struktura całego systemu

Stworzony system to integracja kilku wybranych technologii i wymaganych urządzeń elektronicznych. W połączeniu tworzą projektowany interfejs webowy do sterowania drukarką 3D (w pracy użyto wykonanej w ramach inżynierskiej pracy dyplomowej drukarki Prusa i3). Na obrazie 21 przedstawiono ogólną strukturę powstałego systemu:



Rysunek 21: Struktura całego interfejsu z ogólnym uwzględnieniem jego części

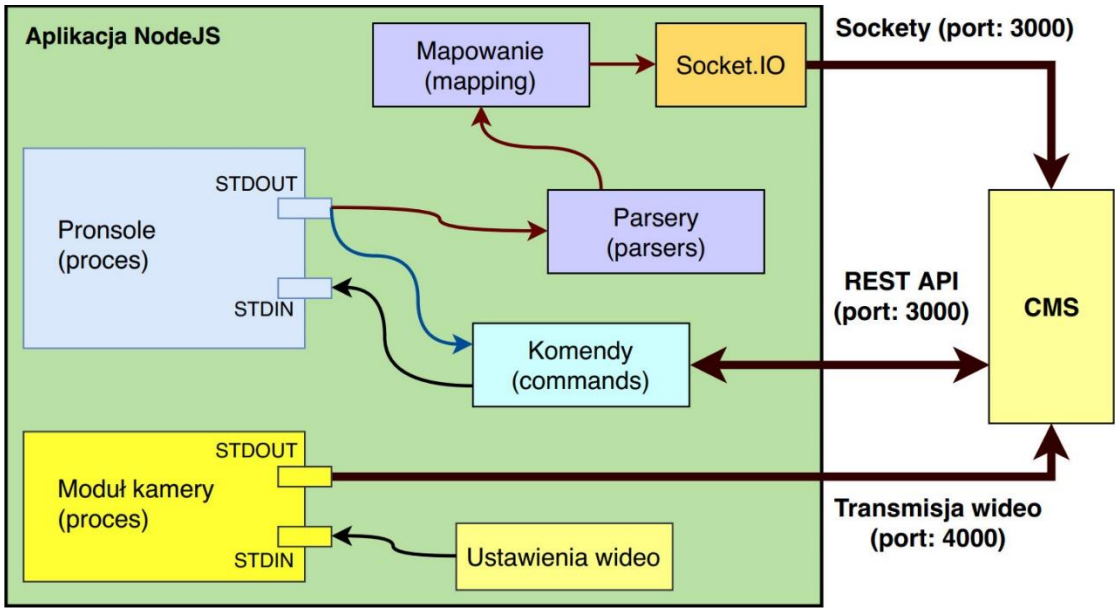
W systemie zintegrowano ze sobą serwer Apache i elementy kontrolne drukarki 3D za pośrednictwem mikrokomputera RaspberryPI 3 B+ oraz kamerkę. Każde z urządzeń wykorzystuje odpowiednie oprogramowanie potrzebne do poprawnej pracy i wzajemnej komunikacji:

1. Użytkownik, chcąc kontrolować drukarkę loguje się do systemu CMS umieszczonego na zdalnym serwerze WWW. Jest to wygodne rozwiązanie, ponieważ z każdego miejsca z dostępem do sieci Internet, użytkownik może zdalnie kontrolować i monitorować pracę drukarki 3D. System CMS umożliwia podgląd aktualnego stanu drukarki, przesłanie stosownych komend sterujących oraz zmianę parametrów połączenia. Więcej informacji na temat systemu CMS w punkcie 6.4.
2. Wyżej wymieniony system łączy się z serwerem NodeJS, pracującym na Raspberry Pi. Aplikacja ta, synchronizuje ze sobą wiele podprogramów:
    a. Wewnątrz aplikacji uruchomiony jest wewnętrzny proces – *Pronsole.py*. *Biblioteka* umożliwia ustanowienie połączenia z drukarką poprzez port USB i dość stabilną komunikację z oprogramowaniem drukarki – *Marlin.*
    b. Poprzez API do procesu *Pronsole* przekazywane są odpowiednio przygotowane komendy, tak, aby były zrozumiałe dla użytej biblioteki sterującej. Użytkownik wyzwalając odpowiednią akcję w panelu sterowania, przesyła informacje do API NodeJS, wyzwalając odpowiednią komendę. Więcej informacji na temat przesyłania komend znajduje się w punkcie 6.7.1.
    c. Ta sama aplikacja NodeJS odpowiada również za nadawanie informacji na temat aktualnej temperatury i stanu procesu *PronsoleI* wykorzystując do tego celu Web Sockety.
    d. NodeJS przejmuje także kontrolę nad kamerką dedykowaną do mikrokomputera RPi.

3. Oprócz aplikacji NodeJS, na mikrokomputerze został uruchomiony serwer Apache. Za jego pośrednictwem istnieje możliwość sterowania akcjami zdalnego włączania i wyłączania zasilania drukarki – fizycznie realizowane przez dedykowaną, samodzielnie zaprojektowaną przystawkę. Szczegóły dotyczące tego urządzenia znajdują się w punkcie 6.3.

6.2. Struktura aplikacji NodeJS

Na poniższym szkicu przedstawiono wewnętrzną strukturę aplikacji NodeJS:



Rysunek 22: Struktura aplikacji NodeJS

Jak wspomniano wcześniej, aplikacja zawiera uruchomione wewnątrz procesy i moduły Node. Najważniejsze z nich to:

a. RestAPI umożliwiające komunikację aplikacji NodeJS z dowolną zewnętrzną aplikacją – w tym przypadku jest to dedykowany system CMS. Domyślnym portem dla API jest port 3000.

b. *Pronsole* – biblioteka do komunikacji z drukarką.

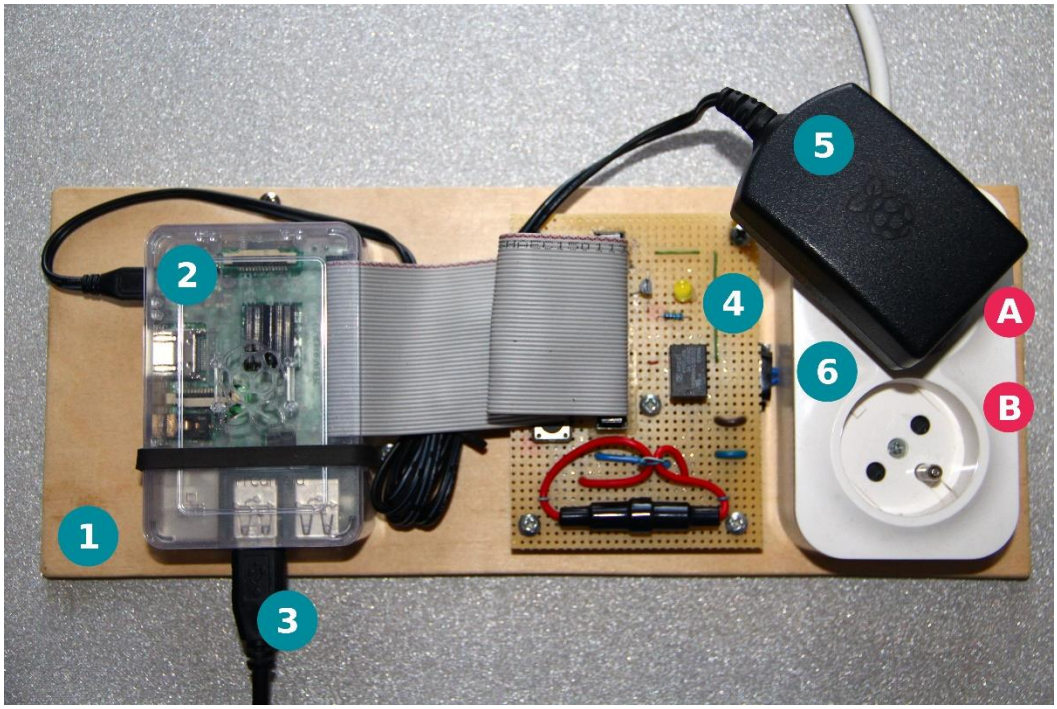
Pozostałe elementy aplikacji to:

a. Moduł kamery – oprogramowanie NodeJS do sterowania kamerką Raspberry Pi. Odpowiada z jej uruchomienie, sterowanie i konfigurację oraz transmisję sygnału wideo (domyślnie na porcie 4000).
b. Sockety (SocketIO) – moduł NodeJS, w tym przypadku użyty do emitowania cyklicznych sygnałów (co jedną sekundę) z informacjami o aktualnej temperaturze stołu, głowicy oraz parametrami z konsoli procesu *Pronsole*. Informacje przed wysłaniem muszą być odpowiednio parsowane do przyjaznego formatu, a następnie mapowane i emitowane. Gotowy sygnał jest odbierany po stronie serwera z systemem CMS i wyświetlany w odpowiednim miejscu w panelu kontrolnym.

6.3. Przystawka do sterowania zasilaniem drukarki

Tak zwana „przystawka”, to dodatkowy element wyposażenia interfejsu, zaprojektowany z myślą o bezpieczeństwie przy zdalnej pracy z drukarką 3D. Celem wykonania tego elementu było stworzenie możliwość zdalnego włączenie i wyłączenia zasilania drukarki. Dodatkowo w panelu sterowania został uwzględniony przycisk awaryjnego wyłączania zasilania drukarki na wypadek błędnego działania aplikacji NodeJS. Akcje te muszą być wyzwolone przez użytkownika. Szczegółowy opis działania poszczególnych funkcji sterowania przystawką został przedstawiony na rysunkach 23 i 24 w dalszej części tego rozdziału.

Na zdjęciu poniżej, pokazano przystawkę wraz z oznaczeniem poszczególnych elementów:



Rysunek 23: Fotografia przedstawiająca komputer Raspberry Pi w raz z przystawką do sterowania zasilaniem drukarki 3D

Opis elementów zaznaczonych na zdjęciu:

1 – Sklejka montażowa o wymiarach 30,0 cm x 12,0 mm x 0,5 cm

2 – Mikrokomputer Raspberry Pi 3 B+

3 – Kabel USB drukarki 3D

4 – Uniwersalna płytka PCB wraz z odpowiednio połączonymi elementami elektronicznymi

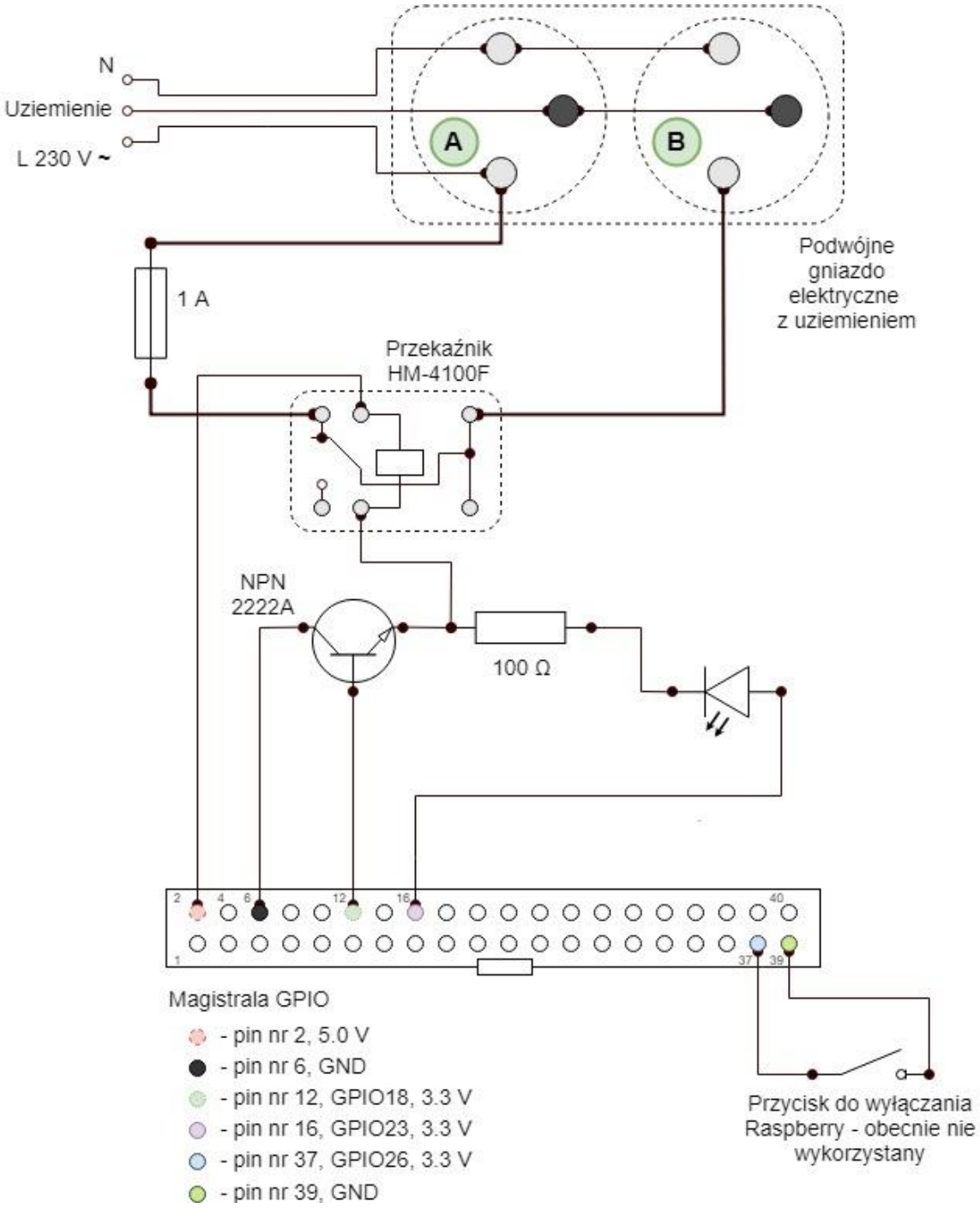
5 – Dedykowany zasilacz Raspberry Pi

6 - Podwójne gniazdo elektryczne ~ 230 V z uziemieniem

*A – gniazdko **niesterowane przekaźnikiem** – gdy przewód główny jest pod napięciem, gniazdo A również. Do tego gniazda może być podłączony zasilacz Raspberry Pi.*

*B – gniazdo **sterowane przekaźnikiem** – gdy dioda sygnalizacyjna jest zapalona, gniazdo jest gotowe do użycia. Do niego należy podłączyć kabel zasilacza drukarki.*

Poniżej pokazano diagram przedstawiający sposób połączenia i zasadę działania poszczególnych elementów przystawki:



Rysunek 24: Diagram przedstawiający sposób połączeń poszczególnych elementów przystawki

Poniżej opisano elementy wraz z uzasadnieniem, których użyto do budowy omawianej przystawki:

- podwójne gniazdko sieciowe 230 V (gniazdo A jest niesterowane, jest pod napięciem od momentu podłączenia przystawki do sieci energetycznej. Gniazdo B jest sterowane za pomocą Raspberry Pi, jest pod napięciem gdy styki przekaźnika są zwarte. Stan ten, sygnalizowany jest za pomocą żółtej kontrolki LED).

- przekaźnik HM-4100F (zamiennik HFD-41)

  napięcia sterowania stykami: $V_{DC} = 5.0\ V$

  maksymalny prąd przewodzenia na stykach:

$$I_{AC}(V_{AC=120V}) = 2\ A \tag{1}$$

  co przy napięciu prądu zmiennego równego $V_{AC} = 230\ V$ pozwala sterować prądem o natężeniu: $I_{AC\ max}(V_{AC=230\ V}) \approx \mathbf{3,83\ A}$

  Obliczenia:

$$\begin{matrix} 120\ V_{AC} & -2\ A \\ 230\ V_{AC} & -I_{AC\ max}\ A \end{matrix} \tag{2}$$

  z proporcji (2) wynika:

$$I_{AC\ max}(V_{AC=230\ V}) = \frac{230\ V_{AC} \cdot 2\ A}{120\ V_{AC}} \approx \mathbf{3,83\ A} \tag{3}$$

  Przeprowadzono również pomiary natężenia prądu pobieranego przez zasilacz drukarki, w celu oszacowania wielkości poboru prądu przy symulowanym maksymalnym obciążeniu. W trakcie pomiaru włączono

podgrzewanie stołu i głowicy drukarki oraz poruszano trzema osiami na raz. Amperomierz wskazywał wartość rzędu 0,5 A , a maksymalne chwilowe wskazanie wyniosło ~0,6 A. Przy normalnej pracy drukarki wartość natężenia pobieranego prądu oscyluje pomiędzy 0,4 – 0.6 A. Wyniki pomiaru dowiodły, że wybrany przekaźnik będzie wystarczający, aby bezpiecznie obsłużyć wymagania projektowe urządzenia.

- bezpiecznik zabezpieczający przekaźnik: walcowy, szklany, o maksymalnym prądzie przewodzenia 1 A (dobrano na podstawie wyżej opisanych pomiarów. Maksymalny prąd przewodzenia bezpiecznika jest większy niż wykazały przeprowadzone pomiary, a mniejszy niż maksymalny prąd przewodzenia przekaźnika. Zadaniem bezpiecznika jest ochrona przed uszkodzeniem przekaźnika HFD-41.

- tranzystor NPN 2222A – przekaźnik sterowany jest za pomocą tego tranzystora. Maksymalne napięcie na pinach wejścia/wyjścia magistrali GPIO to 3.3 V, natomiast cewka przekaźnika musi być sterowana napięciem minimum 5.0 V. Wyżej wymienionego tranzystora użyto w charakterze włącznika przekaźnika. Polaryzacja półprzewodnika sterowana jest pinem logicznym o napięciu 3.3 V, kontrolując przepływ prądu pomiędzy pinem o stałym napięciu 5.0 V i cewką przekaźnika – połączenie przedstawiono na rysunku 24. Gdy na bramę tranzystora zostanie podane napięcie z pinu 12, następuje polaryzacja złącza w kierunku przewodzenia.

- kontrolka LED z rezystor zabezpieczający 100 Ω:

  Opornik zabezpieczający wyznaczono w następujący sposób:

  Przyjęte dane:

  $U_z = 3{,}3\ V$ – napięcie zasilania diody

  $U_D = 2{,}8\ V$ – napięcie przewodzenia żółtej diody LED (2,2 – 3,0 V)

  $I_D = 5\ mA = 0{,}005\ A$ – prąd przewodzenia (nie więcej niż 0,020 A)

U_R – napięcie na rezystorze

Napięcie zasilania to suma napięcia opornika i diody:

$$U_z = U_R + U_D \tag{4}$$

Przekształcając powyższy wzór możemy wyliczyć szukaną wartość rezystancji opornika zabezpieczającego:

$$U_R = U_Z + U_D \quad \Rightarrow \quad R = \frac{U_R}{I_D} \tag{5}$$

Podstawiając odpowiednio wyznaczone wielkości wyznaczono ogólny wzór na wartość szukanego oporu:

$$R = \frac{U_Z - U_D}{I_D} \tag{6}$$

$$R = \frac{3{,}3 - 2{,}8}{0{,}005} = 100 \left[\frac{V}{A}\right] = \mathbf{100}\ \Omega \tag{7}$$

[58, 59]

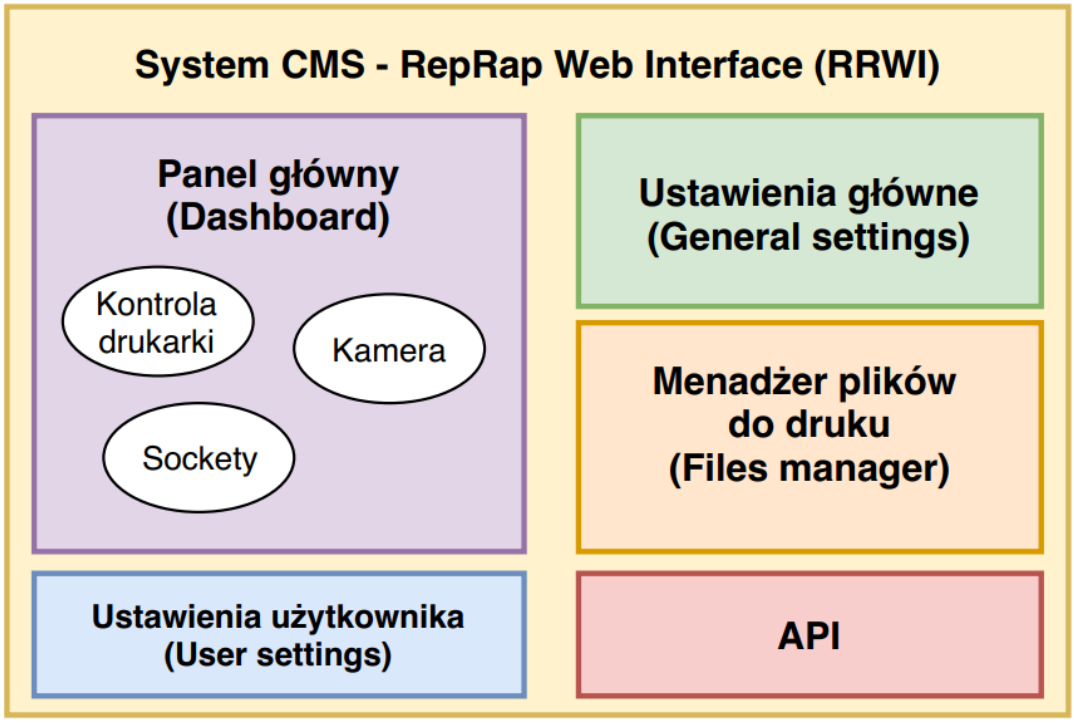
- taśma 40-to pinowa złącza GPIO,
- uniwersalne złącze do 40-to pinowej taśmy,
- uniwersalna płytka PCB – płytka została zabezpieczona kilkukrotnie warstwą lakieru izolacyjnego do płytek PCB.

Podsumowując:

- gniazdo **A** nie jest sterowane, służy tylko do podłączenia zasilacza Raspberry Pi,
- gniazdo **B** (sterowane) jest pod napięciem w momencie, gdy odpowiednie styki przekaźnika HM-4100F są zwarte. Zamykają ona obwód wysokiego napięcia,
- przekaźnik sterowany jest tranzystorem NPN 2222A;
- użyte piny to:
    - pin nr 2 o stałym napięciu 5,0 V,
    - pin nr 6 – GND, tak zwana „masa”,
    - pin nr 12 (GPIO 18) o napięciu 3,3 V ,
        - stan wysoki – złącze NPN spolaryzowane w kierunku przewodzenia, gniazdo B jest aktywne,
        - stan niski – złącze NPN spolaryzowane w kierunku zaporowy – gniazdo B jest wyłączone,
    - pin nr 16 (GPIO 23) zasilanie kontrolki LED.

6.4. Struktura systemu CMS

Stworzony system to współpraca pięciu głównych modułów. Przedstawiono je na poniższym diagramie:



Rysunek 25: Diagram przedstawiający zestawienie głównych modułów całego systemu

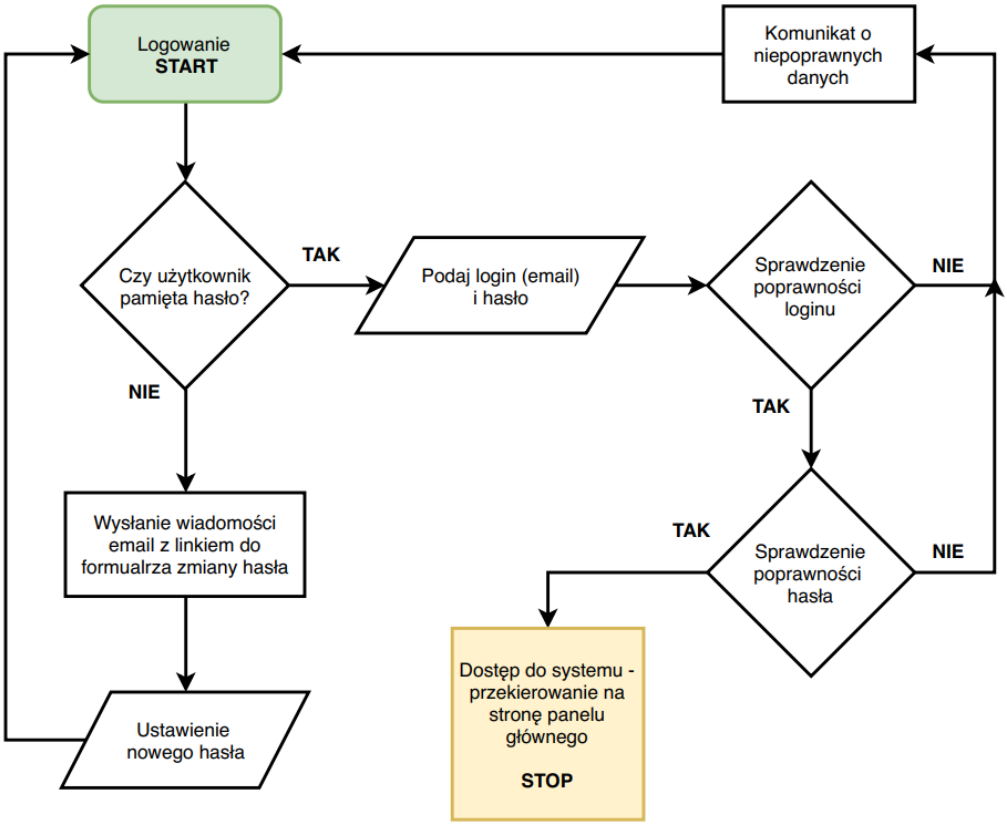
Każdy z nich pełni tak samo ważną rolę w całym projekcie począwszy od podstawowych ustawień, skończywszy na wysyłaniu i odbieraniu stosownych informacji pomiędzy komponentami aplikacji. W kolejnych podpunktach opisano działanie poszczególnych modułów wraz z przebiegiem ich procedur.

6.5. Proces logowania do systemu z uwzględnieniem zmiany hasła

Do stworzonego systemu, aktualnie może zalogować się tylko jeden użytkownik – zdefiniowane jest tylko jedno konto. Formularz logowania umieszczony jest na stronie głównej systemu CMS. Domyślne dane logowania to:

Login:	admin@admin.com
Hasło:	asd123

Na rysunku 26 przedstawiono proces logowania użytkownika od interfejsu:

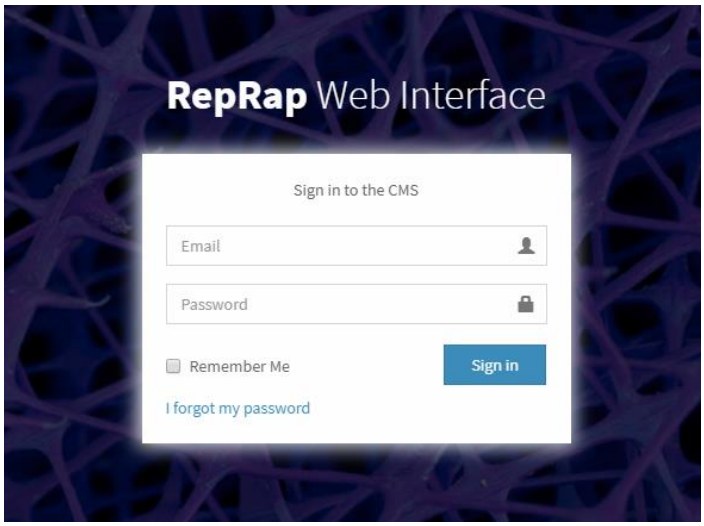


Rysunek 26: Proces logowania z uwzględnieniem zmiany zapomnianego hasła

Po wprowadzeniu danych do formularza, następuje ich walidacja. W przypadku, gdy podane dane są poprawne, następuje uwierzytelnianie użytkownika i przekierowanie do panelu sterowania drukarką. Jednakże,

w przypadku gdy użytkownik nie pamięta hasła, może zresetować stare hasło, poprzez kliknięcie w odpowiedni link, widoczny pod formularzem logowania.

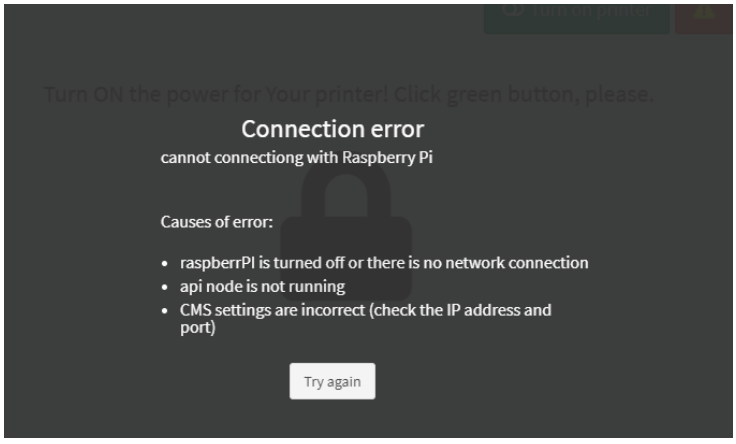
Na adres email użytkownika przesyłany zostaje unikalny link do formularza zmiany hasła. Użytkownik wprowadza nowe hasło i znów ma możliwość zalogowania się do systemu.



Rysunek 27: Formularz logowania do systemu

Formularz logowania zawiera dwa pola tekstowe – email oraz hasło. Zaznaczenie pola wyboru, spowoduje zapamiętanie sesji użytkownika. Na samym dole widoczny jest link do formularza zmiany hasła.

Po poprawnym zalogowaniu, użytkownikowi ukaże się widok panelu sterowania drukarką – więcej informacji o panelu sterowania w punkcie 6.4.3 lub komunikat o błędzie komunikacji przedstawiony na rysunku 28. W tej sytuacji użytkownik powinien przejść do edycji ustawień – opis w kolejnym punkcie tego rozdziału.

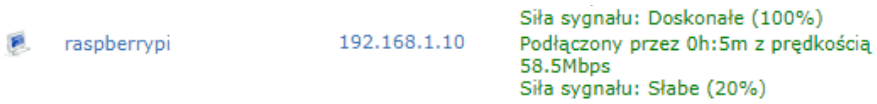


Rysunek 28: Komunikat o błędzie połączenia z API NodeJS

6.6. Ustawienia systemu

Po lewej stronie ekranu widoczne jest menu główne systemu. Przechodząc do zakładki „General Settings" użytkownik ma możliwość zmiany głównych parametrów systemu takich jak:

- „Base Url" - Adres Raspberry Pi – jest to adres Raspberry Pi w sieci lokalnej. Można nim zarządzać z poziomu własnego routera lub sprawdzić bezpośrednio w Raspberry przy użyciu polecenia *ifconfig* [44]. Jest to adres pod którym serwowane jest API uruchomione na Raspberry Pi.



Rysunek 29: Fragment panelu domowego routera z widocznymi parametrami połączenia Raspberry Pi w sieci lokalnej

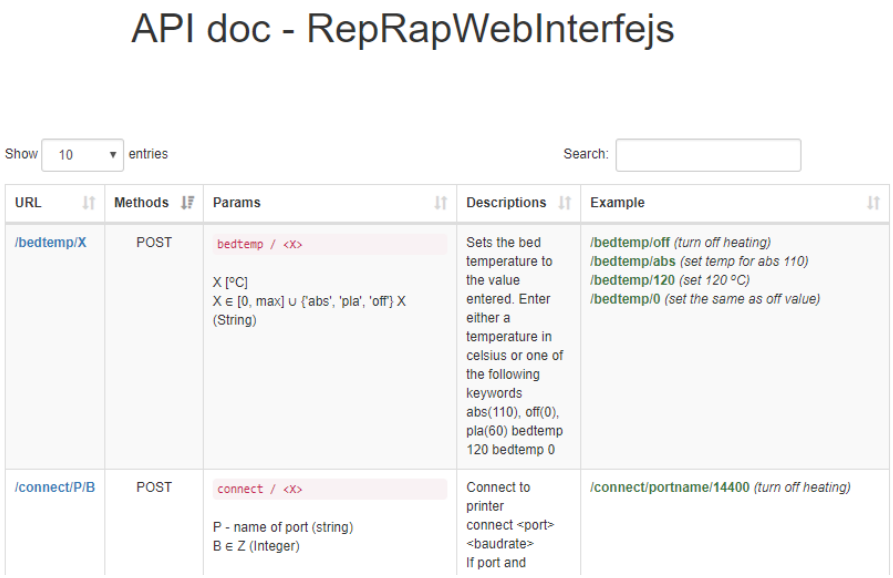
- „Port RRWI-API" – port na którym pracuje aplikacja API NodeJS. Domyślnie w kodzie aplikacji zdefiniowano wartość portu na 3000. Jeśli jednak z jakichś powodów zaszłaby konieczność zmiany tej wartość (np. port 3000 byłby zajęty przez inną aplikację) należy edytować plik */home/rrwi/src/index.js* ustawiając dowolny niewykorzystany port. Ustawiony port należy również zdefiniować w ustawieniach CMS[45].

- „Api doc" – jest to adres do dokumentacji API NodeJS stworzonej w trakcie projektowania komponentu. Powstałe endpointy służą do komunikacji systemu CMS z aplikacją NodeJS. Na serwerze

[44] Użycie polecenia ifconfig jest możliwe w przypadku gdy mamy bezpośredni dostęp do Raspberry poprzez standardowe urządzenia wejścia – wyjścia, takie jak klawiatura i ekran.
[45] Do połączenia z API NodeJS potrzebny jest adres IP serwera (Raspberry PI) oraz port na którym pracuje API (domyślnie 3000). System CMS generuje z tych dwóch parametrów pełny adres do API.

Apache2 znajduje się plik HTML zawierający dokumentację sporządzoną w trakcie tworzenia endpointów RestAPI. Adres do dokumentacji:

http://<<adres Raspberry PI>>/api



API doc - RepRapWebInterfejs

Show 10 entries Search:

URL	Methods	Params	Descriptions	Example
/bedtemp/X	POST	bedtemp / <X> X [°C] X ∈ [0, max] ∪ {'abs', 'pla', 'off'} X (String)	Sets the bed temperature to the value entered. Enter either a temperature in celsius or one of the following keywords abs(110), off(0), pla(60) bedtemp 120 bedtemp 0	/bedtemp/off *(turn off heating)* /bedtemp/abs *(set temp for abs 110)* /bedtemp/120 *(set 120 °C)* /bedtemp/0 *(set the same as off value)*
/connect/P/B	POST	connect / <X> P - name of port (string) B ∈ Z (Integer)	Connect to printer connect <port> <baudrate> If port and	/connect/portname/14400 *(turn off heating)*

Rysunek 30: Fragment dokumentacji API NodeJS

- „Port RRWI-CAM” – to kolejny parametr dotyczący portu aplikacji NodeJS. Kamerka Raspberry sterowana jest przez aplikację NodeJS i działa jako podproces. Domyślnie zdefiniowano port 4000 dla serwera kamery.

- „Home path” – pełna ścieżka do lokalizacji plików źródłowych aplikacji NodeJS na mikrokomputerze Raspberry.

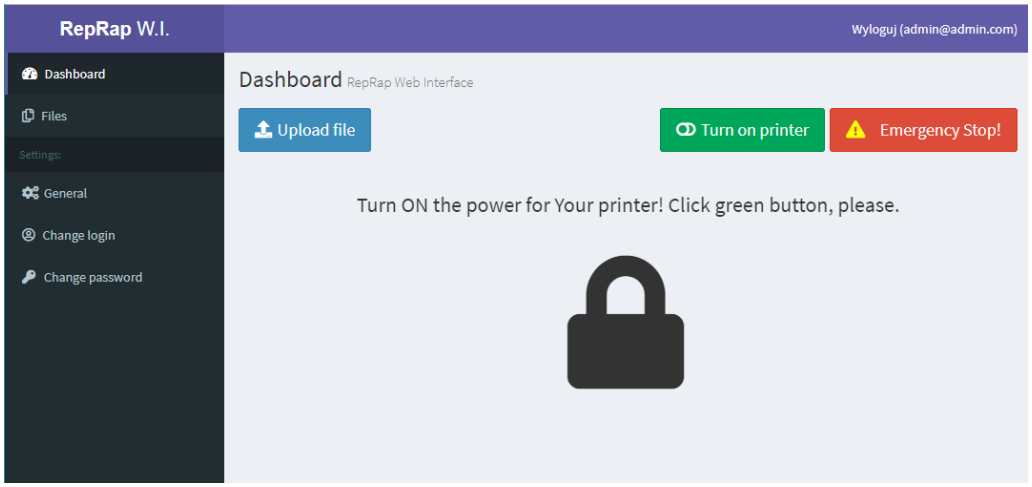
- „Max hotend temp” oraz „Max bed temp” – maksymalne temperatury ustawione dla danej drukarki [46].

[46] Ustawienia te nie mają wpływu na zmianę parametrów granicznych drukarki. Służą jedynie do ustawienia skali temperatury na potrzeby interfejsu.

- „Port of printer” – domyślny port, na którym realizowana jest komunikacja pomiędzy Raspberry a drukarką 3D.

- „Baudrate” – szybkość transmisji danych połączenia. Wartość ta definiowana była przy kompilacji kodu oprogramowania drukarki. W przypadku użytej drukarki *Prusa i3* ustawiono baudrate na wartość 14400.[47]

Gdy wszystkie parametry są poprawne, po przejściu do strony Panelu sterowania powinny pokazać się aktywny przycisk „Turn ON” umożliwiający uruchomienie drukarki. W przeciwnym wypadku należy zweryfikować ponownie wprowadzone parametry.



Rysunek 31: Widok po poprawnym zalogowaniu i skonfigurowaniu parametrów konta.

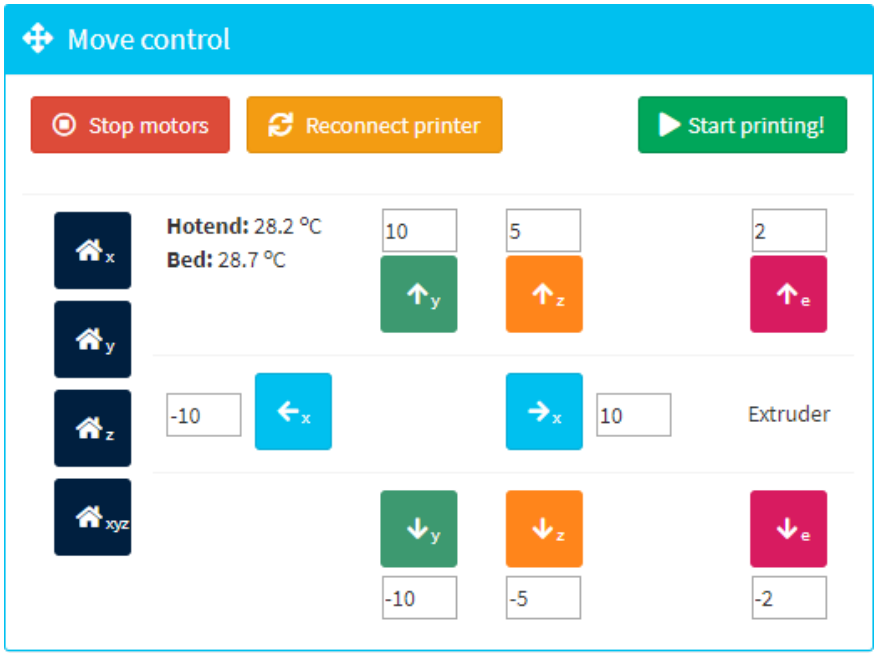
[47] Szybkość transmisji danych (ang. Baudrate) definiowana w CMS musi być zgodna z wartością ustawioną w oprogramowaniu drukarki 3D.

6.7. Główny panel sterowania

Po poprawnym nawiązaniu połączenia oraz włączeniu zasilania (zielony przycisk) drukarki, ukaże się panel sterowania podzielony na cztery główne sekcje.

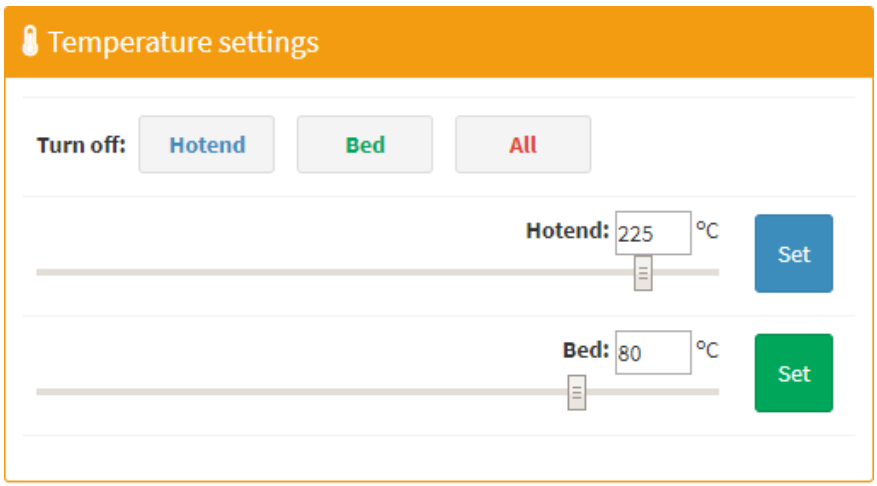
Są to:

a) „Move control” – sekcja sterowania ruchem stołu oraz głowicy. Umożliwia manualne przemieszczanie wyżej wymienionych elementów o ustawiony krok. Możliwe jest również zresetowanie pracy drukarki oraz ustawienie łoża i głowicy w pozycji startowej (0, 0, 0).



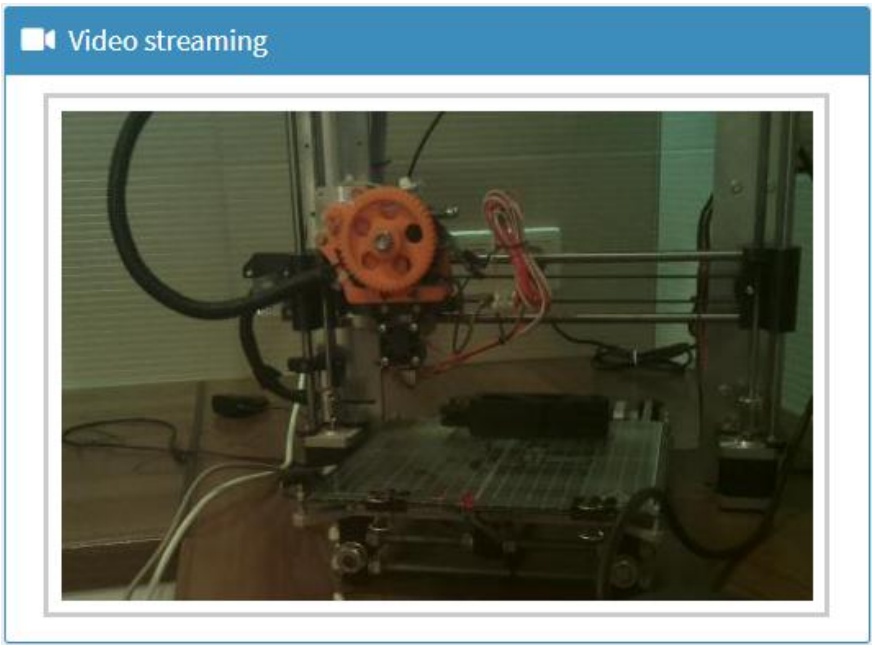
Rysunek 32: Sekcja "Move control" - kontrola ruchomych elementów drukarki

b) „Temperature settings” – sekcja kontroli ustawień temperatury stołu i głowicy. W tej części można ustawić wybraną temperaturę za pomocą suwaków. Zmiany temperatury widoczne będą w sekcji „Move control”, obok ikony „Home X”.



Rysunek 33: Sekcja "Temperature settings" do kontroli nad temperaturą głowicy i stołu

c) „Video streaming” – sekcja podglądu pracy drukarki – transmisja „na żywo”



Rysunek 34: Sekcja "Video streaming" - umożliwia podgląd pracy drukarki "na żywo"

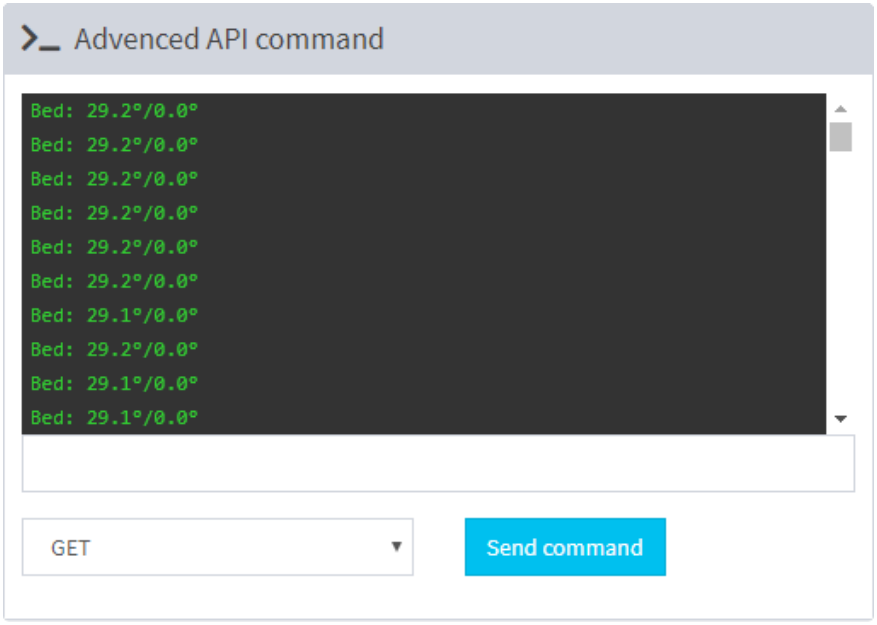
Parametry wideo zdefiniowane są „na sztywno” w głównym pliku index.js:

```
// DEFAULT PARAMS
const port = process.env.PORT || 3000;
const CAM = {
    PORT: 4000,
    WIDTH: 480,
    HEIGHT: 320,
    TIMEOUT: 50,
    QUALITY: 10,
}
```

Rysunek 35: Domyślna konfiguracja parametrów wideo streamingu

d) „Advenced API command” – sekcja podglądu stanu procesu biblioteki „Pronsole”. Jak wspomniano wcześniej, aplikacja NodeJS uruchamia w wewnętrznym procesie bibliotekę „Pronsole”, służącą do komunikacji i sterowania drukarką 3D. Jako proces, zwraca ona pewne parametry w zależności od użytej komendy. Komendy przekazywane są do procesu poprzez API, natomiast ich wynik są zwracane na bieżąco poprzez gniazda (sockets) i wyświetlane w omawianej sekcji. Parametry można wprowadzać w polu tekstowym pod czarnym prostokątem. Po lewej stronie przycisku można wybrać metodę przesyłania informacji.

Dostępne komendy API opisane są w przygotowanej dokumentacji, umieszczonej na serwerze Apache Raspberry PI *http://<ip Raspberry>/api*.



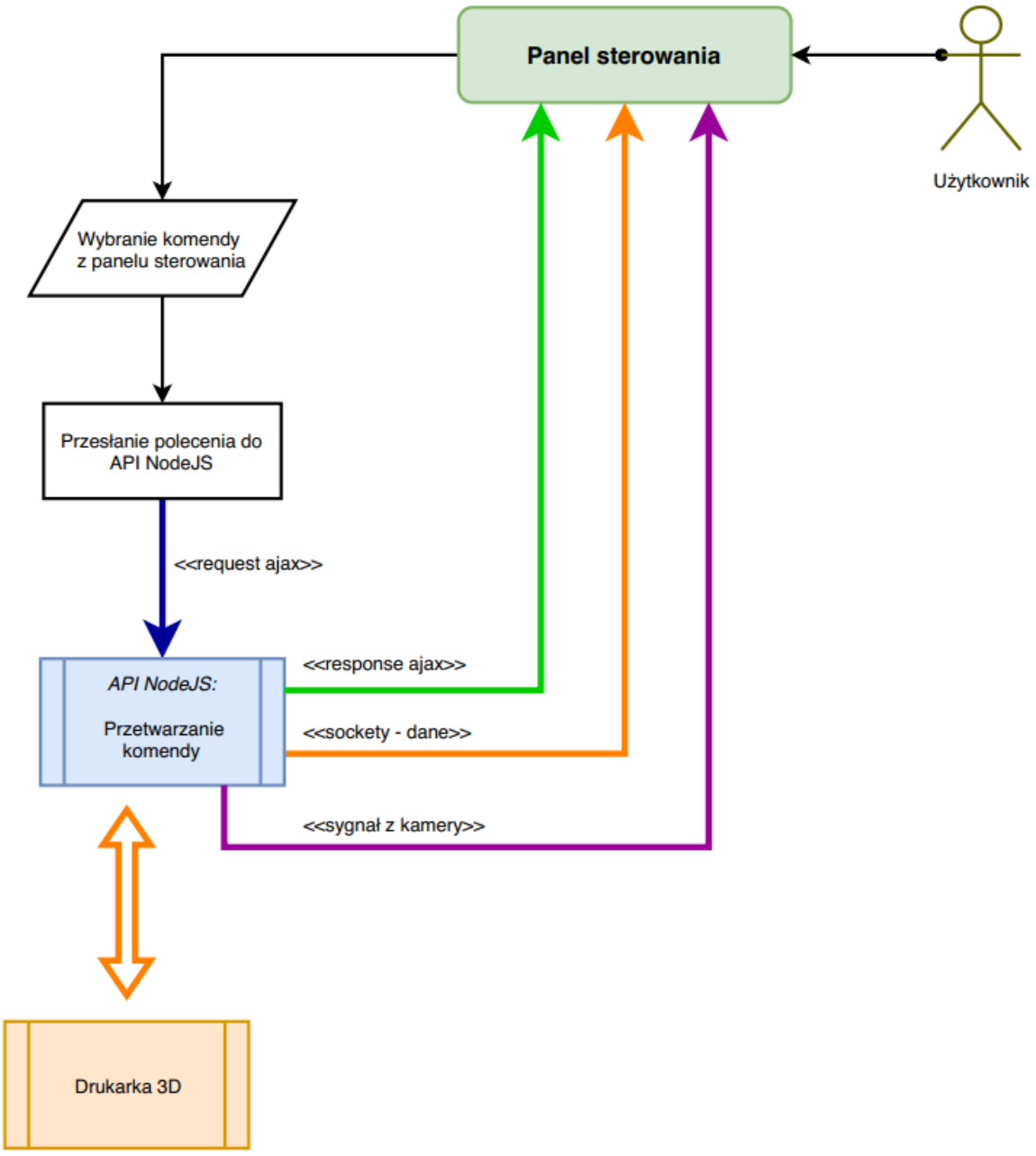
Rysunek 36: Sekcja "Advenced API command" z przykładowymi odczytami – aktualna temperatura łoża

Oprócz powyższy sekcji, na samej górze interfejsu umieszczone są trzy przyciski:

a) „Upload file” umożliwiający przesłanie pliku do systemu (nie do druku),
b) „Turn On/Off” do sterowania zasilaniem drukarki poprzez przystawkę,
c) „Emergency Stop!” do awaryjnego odcięcia zasilania.

Zasadę pracy i różnice pomiędzy przyciskiem „Turn Off” i „Emergency Stop!” opisano w dalszej części tego rozdziału.

6.7.1. Proces przesłania dowolnej komendy, czyli współpraca modułu Panelu głównego z API



Rysunek 37: Diagram obrazujący przesłanie komendy sterującej do drukarki

W panelu sterowania, po kliknięciu w wybrany przycisk wysyłana jest konkretna komenda do API NodeJS, przy użyciu technologii Ajax.

Dla przykładu: użytkownik drukarki chce ustawić głowicę w pozycji początkowej, wzdłuż osi X.

1. Klika więc ikonkę z miniaturą domku o indeksie „x”.
2. Po kliknięciu w przycisk wyzwalana jest akcja, przesłania komendy do API NodeJS: „home/x”.
3. Po stronie API komenda jest odpowiednio interpretowana i przekazywana do wewnętrznego procesu „Pronsole”.
4. Następnie biblioteka ta, komunikuje się z drukarką i przekazuje dalej polecenie w postaci kodu maszynowego Gcode.
5. Kod interpretowany jest przez oprogramowanie drukarki, na skutek czego głowica przesunie się do pozycji początkowej osi X.

Warto zwrócić uwagę na to, że w trakcie przesyłania komend pomiędzy systemem CMS, API i drukarką, w tym samym czasie zostają pobierane aktualne parametry drukarki i przesyłane do panelu sterowania. Kamerka również działa jako osobny proces. Jak widać, zadanie nie zakłócają się wzajemnie, działają niezależnie, dzięki czemu można sprawnie posługiwać się interfejsem.

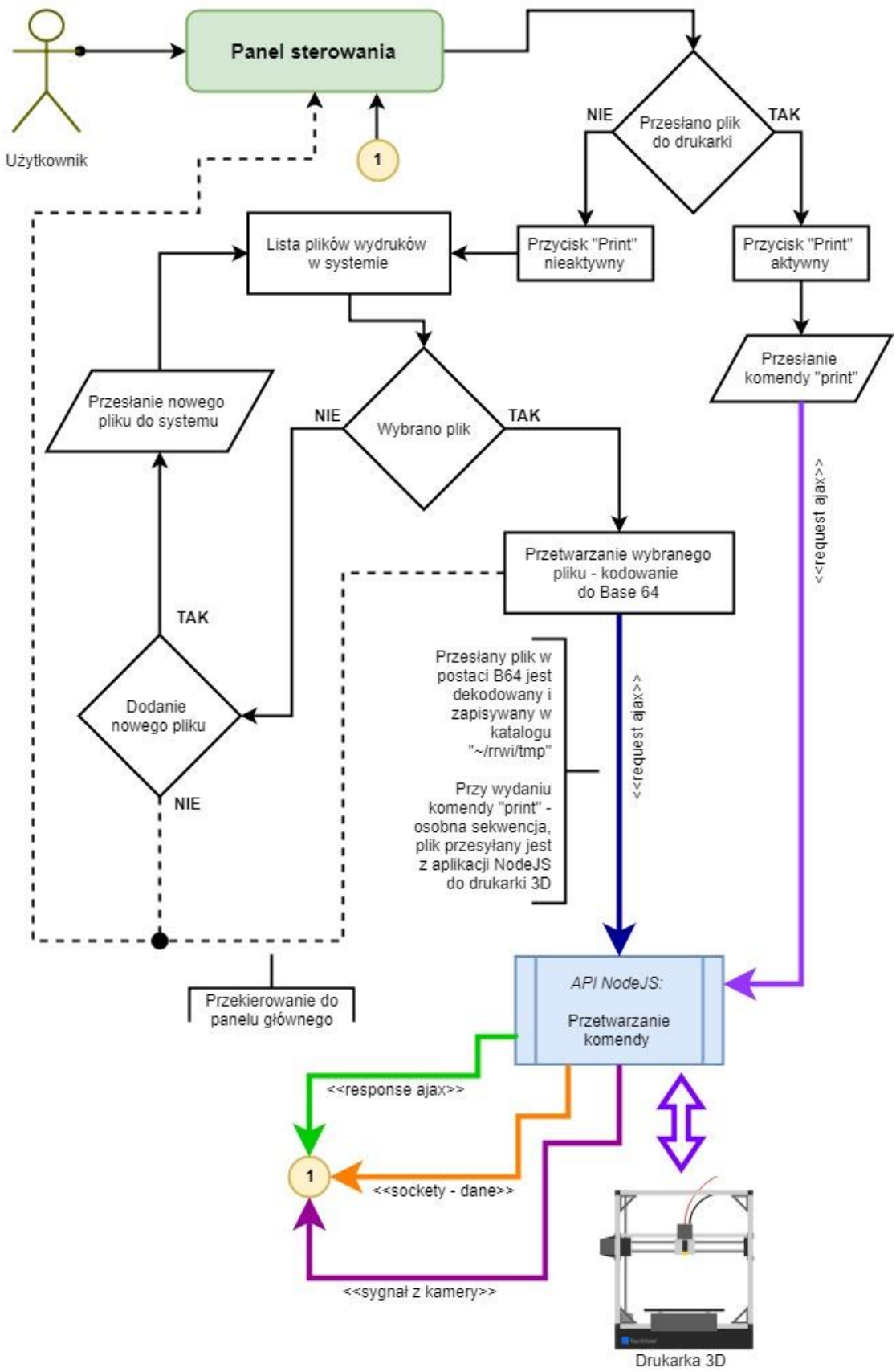
rrwi-node / src / commands / funcs / home.js

```
1   /**
2    * Command to send homeing command for select axis
3    * @param {*} pronsole
4    * @param {*} params
5    */
6   module.exports = (pronsole) => (params) => {
7
8           pronsole.stdin.write('home ' + params + '\n');
9
10  };
11
```

Rysunek 38: Fragment kodu realizujący wywołanie komendy zerowania pozycji dla poszczególnych osi

6.7.2. Przesłanie pliku do druku – Menadżer plików

Przesłanie pliku do druku jest nieco bardziej skomplikowane niż obsługa pojedynczych komend sterowania. Na poniższym diagramie (rysunek 39) przedstawiono algorytm zarządzania i przesyłania plików do druku:



Rysunek 39: Schemat przesyłania pliku do druku

Użytkownik, za pomocą panelu sterowania może przesłać plik do wydruku. W przypadku, gdy nie przesłano jeszcze pliku do wydruku, przycisk „Print" jest nieaktywny. Aby wysłać plik, użytkownik musi wybrać go z listy przesłanych wcześniej plików do systemu – pliki przetrzymywane są po stronie systemu CMS, dopiero stamtąd można przekazać je pojedynczo do wydruku.

Po przejściu do listy plików, użytkownik może wybrać plik lub dosłać kolejny. Gdy wybrano plik do wydruku, system konwertuje go do postaci kodu Base64, a następnie zostaje wysyłany do API NodeJS. Przesłany plik jest dekodowany z powrotem do postaci pliku gcode i zapisywany w katalogu tymczasowym *~/rrwi/tmp*. Po przesłaniu pliku do aplikacji NodeJS, przycisk „Print" staje się aktywny. Po jego kliknięciu, w osobnej komendzie wysyłane jest żądanie wydruku wcześniej przesłanego pliku. Plik jest przesyłany za pośrednictwem procesu „Pornsole" do drukarki – wydruk się rozpoczyna, a użytkownik może monitorować postępy poprzez podgląd z kamery i aktualnych parametrów drukarki.

6.7.3. Przyciski sterowania zasilaniem drukarki

Sterowanie zasilaniem drukarki odbywa się poprzez przyciski „Turn On", Turn Off" oraz „Emergancy Stop!". Wyzwalają one akcje sterowania przekaźnikiem przystawki.

- Akcja „Turn On"

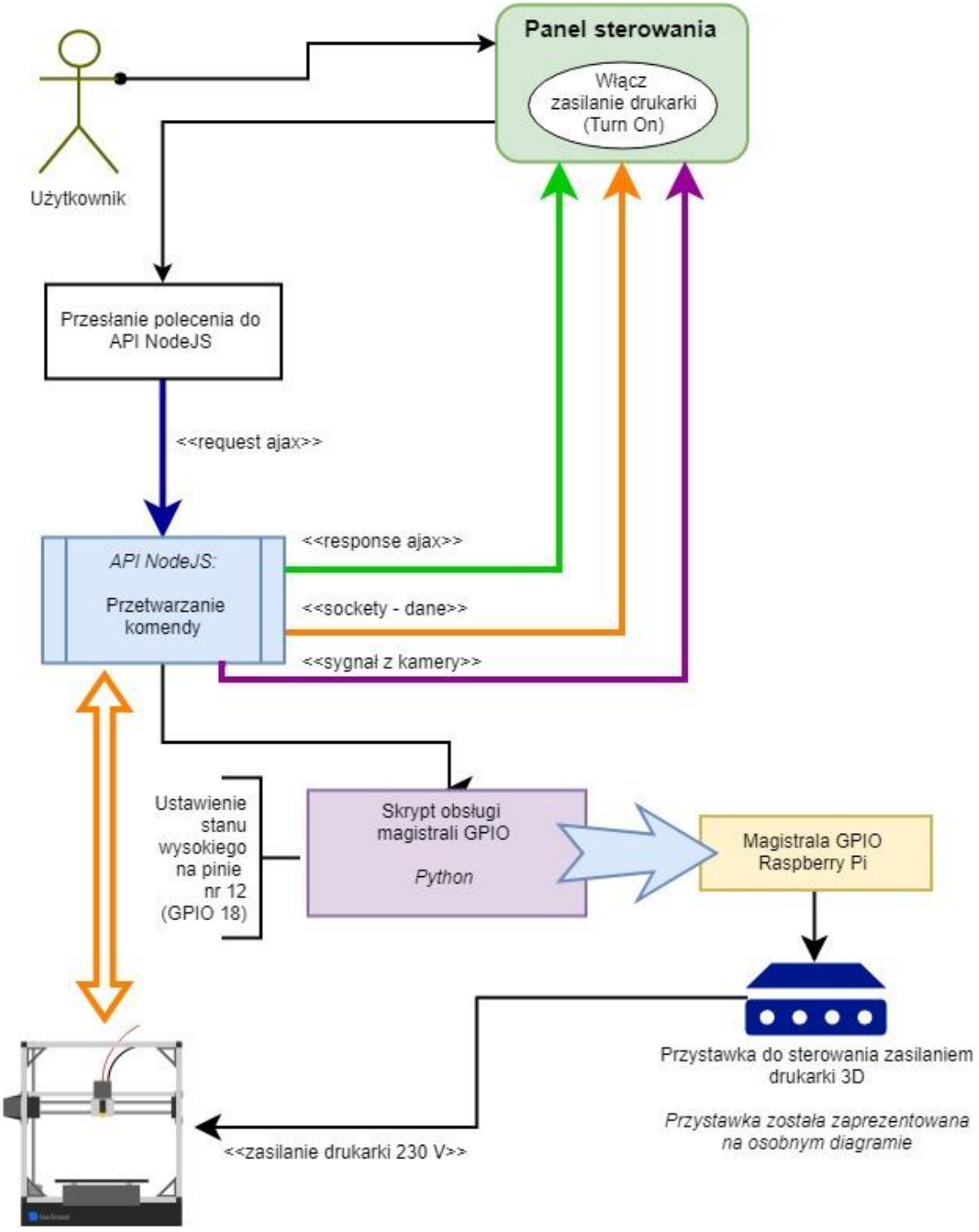
  Akcja ta włącza zasilanie drukarki 3D. Warto tutaj zwrócić uwagę na to, że zasilanie to jest potrzebne do fizycznej realizacji komend, jak np. ruch silników krokowych, czy podgrzewanie stołu. Sama komunikacja aplikacji NodeJS i oprogramowania drukarki następuje po uruchomieniu aplikacji, gdyż elektronika drukarki zasilana jest z portu USB – interfejs sam w sobie działa, ale nie ma z nim żadnej interakcji, zostało zrealizowane połączenie, a przesłane komendy nie są wykonywane.

rrwi-node / py / on.py

```
1   import RPi.GPIO as gpio
2   import time
3
4
5   gpio.setmode(gpio.BCM)
6   gpio.setup(18, gpio.OUT)
7   gpio.setup(23, gpio.OUT)
8
9   #pin for led control
10  gpio.output(23, gpio.HIGH)
11
12  #pin for switch transistor
13  gpio.output(18, gpio.HIGH)
14
15  print 'Power ON'
16  text_file = open("/home/pi/rrwi/py/power_status.txt", "w")
17  text_file.write("1")
18  text_file.close()
19
```

Rysunek 40: Fragment kodu odpowiedzialny za zmianę stanu pinów na magistrali GPIOdla akcji "Turn On"

Na poniższym schemacie przedstawiono algorytm obsługi przystawki, w celu zasilenia drukarki:



Rysunek 41: Akcja "Turn On"

Użytkownik wyzwala akcję „Turn On”. API uruchamia skrypt obsługi magistrali GPIO. Rezultatem jego działania jest zwarcie styków przekaźnika – drukarka jest uruchomiona.

Więcej informacji na temat działania przystawki można znaleźć w punkcie 6.3.

- Akcja „Turn Off”

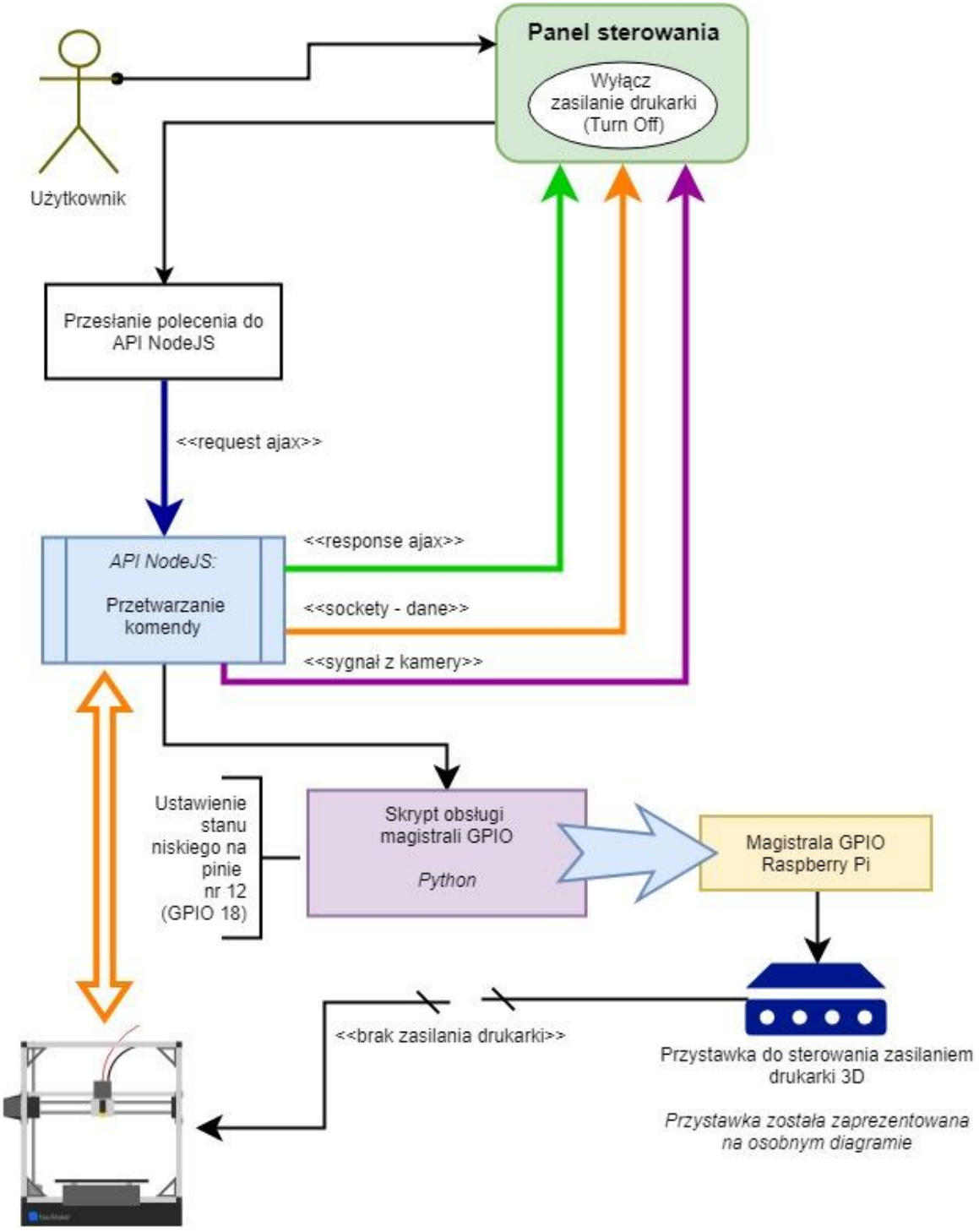
  W interfejsie przewidziano dwa niezależne od siebie sposoby odłączenia zasilania drukarki. Pierwszy z nich, o którym wspomniano już wcześniej to przycisk „Turn Off”. Akcja ta, wykonywana jest z poziomu API – uruchamiany jest odpowiedni program w języku Python, odpowiedzialny za wysterowanie stanu niskiego na bramie tranzystora. Poniżej kod skryptu Python dla tej akcji:

rrwi-node / py / **off.py**

```
1   import RPi.GPIO as gpio
2   import time
3
4
5   gpio.setmode(gpio.BCM)
6   gpio.setup(18, gpio.OUT)
7   gpio.setup(23, gpio.OUT)
8
9   #pin for led control
10  #gpio.output(23, gpio.HIGH)
11
12  #pin for switch transistor
13  gpio.output(18, gpio.LOW)
14
15  print 'Power OFF'
16  text_file = open("/home/pi/rrwi/py/power_status.txt", "w")
17  text_file.write("0")
18  text_file.close()
19
```

Rysunek 42: Fragment kodu python odpowiedzialnego za zmianę stanów pinów GPIO dla akcji „Turn Off”

Algorytm akcji widoczny na kolejnym diagramie (rysunek 43) :



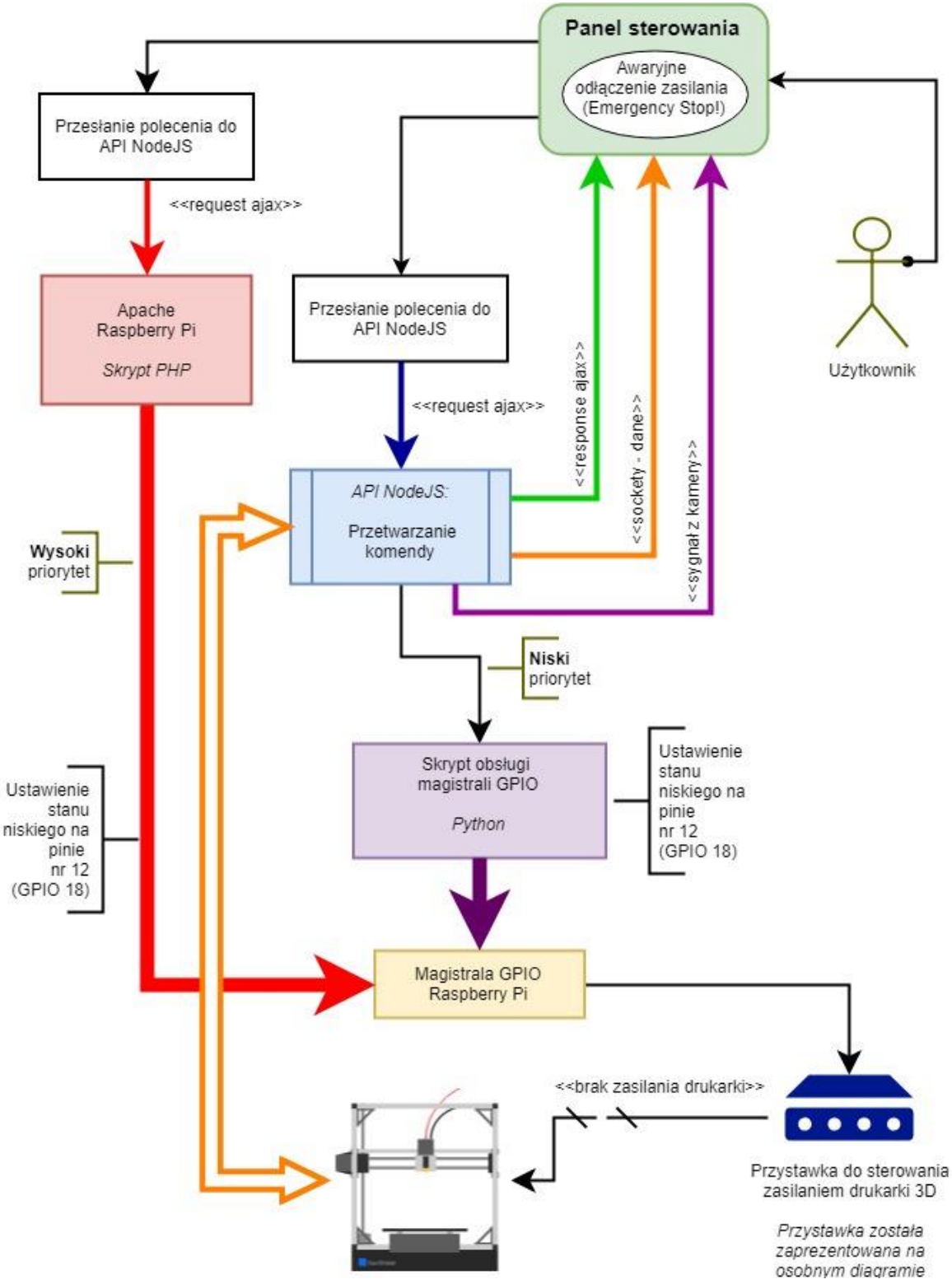
Rysunek 43: Akcja "Turn Off"

Podobnie jak w przypadku akcji „Turn On” przesyłana jest informacja do API. Uruchamiany jest kolejny skrypt obsługi pinów GPIO w celu rozwarcia styków przekaźnika i tym samym odcięcia zasilania drukarki.

- Akcja „Emergency Stop!”

  Druga metoda to przycisk „Emergency Stop!”. Akcja ta nie pomija w swym działaniu aplikacji NodeJS, a jedynie zmniejsza jej priorytet. Głównym zadaniem tego przycisku jest wyzwolenie akcji z poziomu PHP (obsługa przez serwer apache2 na Raspberry Pi). Skutkiem wykonania się skryptu PHP jest rozwarcie styków przekaźnika (więcej informacji o sposobie działania przystawki w punkcie 6.3). Akcja ta jest nieco szybsza niż za pośrednictwem API NodeJS. Dodatkowo po skrypcie PHP wykonywany jest skrypt z akcji „Turn Off” poprzez API NodeJS (w przypadku gdyby serwer apache2 przestał dziać lub wystąpił błąd wykonania skryptu PHP).

  Poniżej przedstawiono algorytm akcji *Emergency Stop:*



Rysunek 44: Akcja Emergency

rrwi-node / apache / rrwi / **stop.php**

```php
<?php

$gpio_off = shell_exec('gpio -g mode 18 out & gpio -g write 18 0');

echo json_encode(['msg' => "Turn off POWER SUPPLY!"]);

$status = "0";
$file_path = "/home/pi/rrwi/power_status.txt";
$file_handle = fopen($file_path, 'w');
fwrite($file_handle, $status);
fclose($file_handle);

?>
```

Rysunek 45: Fragment kodu PHP realizujący fragment akcji „Emergency Stop!”

7. Uruchomienie aplikacji

1. Aby uruchomić interfejs należy posiadać serwer stron www lokalny lub zdalny.
2. Sklonować projekt ze zdalnego repozytorium, a następnie uruchomić instalację zgodnie z instrukcją w projekcie i na stronie freamework’a Yii2.
3. Pobrać obraz skonfigurowanego Raspbiana.
4. Nagrać poprany obraz na kartę SD i uruchomić go w Raspberry.
5. Podłączyć drukarkę do portu USB, wtyk zasilacza umieścić w gnieździe B przystawki.
6. Skonfigurować połączenie sieciowe.
7. Uruchomić ponownie Raspberry.
8. Zmodyfikować ustawienia w systemie CMS (IP itp.) – więcej w punkcie 6.6.

Jeśli wszystko jest poprawnie wykonane i ustawione, połączenie z API NodeJS powinno zostać poprawnie nawiązane.

8. Podsumowanie i wnioski

a) Wady i zalety powstałego interfejsu

Wady interfejsu:

- Brak wersji mobilnej – szablon strony nie jest w pełni responsywny,
- Jedna wersja językowa – tylko język angielski,
- Problemy ze stabilnością – darmowa biblioteka Pronsole jest rozwiązaniem tworzonym przez społeczność, wraz z rozwojem tejże aplikacji można natknąć się na problemy w jej działaniu. Ponadto poprzez wykorzystanie wielu technologii webowych zdarza się chwilowy przestój w przepływie informacji – głównie problem synchronizacją stanu aplikacji NodeJS w systemie CMS,
- Konfiguracja w Raspberry Pi pierwszego połączenia z Internetem – konieczność użycia zewnętrznych urządzeń w celu uruchomienia i konfiguracji Raspberry Pi,
- Publicznego udostępnienia Raspberry – może mieć to negatywny wpływ na bezpieczeństwo w sieci,
- Jeden użytkownik systemu – w obecnej wersji możliwe jest korzystanie z systemu tylko przez jednego użytkownika.

Zalety systemu:

- Zdalna kontrola nad urządzeniem drukującym wraz z podglądem video,
- Skalowalność i elastyczność – system można rozbudowywać w dowolnym kierunku, na dowolnej warstwie (CMS, API, Raspberry Pi),
- Mobilność – rozumiana na dwa sposoby:
    - Dostęp do interfejsu z dowolnego miejsca na Ziemi (wszędzie tam, gdzie użytkownik ma do dyspozycji komputer oraz połączenie z Internetem),

- Przystawka jest kompaktowa i małych rozmiarów - dzięki temu nie ma problemu z transportem. Można ją w szybki i łatwy sposób podłączyć do drukarki,

- Łatwość instalacji – aby uruchomić aplikację na Raspberry wystarczy skorzystać z gotowego, w pełni skonfigurowanego obrazu Raspbian'a wraz z plikami i odpowiednią konfiguracją (jedyne co trzeba zrobić samodzielnie to, ustanowić połączenie z Internetem).

b) Propozycje dalszego rozwoju systemu

Jak wspomniano w poprzednim punkcie, cały system jest skalowalny i bardzo elastyczny w kwestii rozbudowy na wszystkich poziomach. Biblioteka realizująca komunikację z drukarką jest rozwijana nieustannie przez społeczność, więc tą kwestię pominięto w tym punkcie.

- Warstwa dotycząca NodeJS to główny element całego systemu. Możliwości technologii NodeJS pozwalają na stworzenie całego interfejsu z pominięciem PHP i MySQL.,
- Użyte RestAPI otwiera pole do popisu na wielu platformach, głównie dla urządzeń mobilnych, dając możliwość stworzenia aplikacji dedykowanej na system Android czy iPhone,
- Powstały system CMS można rozbudować o większą liczbę użytkowników. Wiązałoby się to oczywiście z gruntowną modyfikacją warstwy NodeJS. Obecny system jest dobrym fundamentem pod stworzeniem systemu „ksero 3D".

c) Wnioski

Podsumowując, w ramach magisterskiej pracy dyplomowej:

- Uruchomiono i odpowiednio skonfigurowano Raspberry Pi 3 B +,
- Zrealizowano komunikację mikrokomputera z drukarką 3D poprzez zastosowanie darmowej biblioteki Pronsole,

- Zaprojektowano i wykonano przystawkę do sterowania zasilaniem drukarki,
- Stworzono aplikację w technologii NodeJS, działającą na Raspberry, kontrolującą pracę biblioteki Pronsole oraz umożliwiającą sterowanie przystawką i transmisję wideo „na żywo”,
- Wykonano system CMS umożliwiający: kontrolowania parametrów drukarki, zarządzanie wymaganymi ustawieniami i plikami druku – aplikacja na zdalnym serwerze, komunikująca się z sekcją NodeJS poprzez RestAPI.

Obecna wersja systemu łączy w sobie działanie wielu technologii webowych. Spośród nich, zdecydowanym liderem jest NodeJS, gdyż ma bardzo szerokie spektrum użyteczności. Z powodzeniem, w projektowanym interfejsie można było pominąć warstwę PHP i zaimplementować CMS z poziomu NodeJS. Pozwoliłoby to uniknąć wielu problemów związanych z synchronizacją stanów obu modułów, co z kolei poprawiłoby stabilność i ogólne działanie całego systemu.

Powstały projekt stanowi dobrą podstawę do tworzenia i rozwoju zdalnych interfejsów sterowania urządzeniami. Raspberry Pi w połączeniu z NodeJS i RestAPI, tworząc potężne narzędzie, które może znaleźć zastosowanie w wielu dziedzinach życia. W szczególności może posłużyć do kontrolowania urządzeń i systemów pomiarowy oraz umożliwić komunikację między nimi. Jest to zadanie, które będzie coraz bardziej powszechne w dobie rozwoju systemów Internetu Rzeczy.

Kod interfejsu oraz obraz skonfigurowanego Rasbian'a dostępny jest pod linkami:

Linki do projektu:

https://github.com/mb92/RRWI.git

https://github.com/mb92/RRWI---App-NodeJS-.git

Obraz skonfigurowanego Raspbian'a (wraz z plikami aplikacji i szkicami układu elektronicznego):

https://mega.nz/#F!11wB0K4a

Wersja elektroniczna pracy dyplomowej:

https://www.thesis.maciejborowiec.pl/rrwi-maciejborowiec.pdf

9. Bibliografia

[1] *https://github.com/w-A-L-L-e/printerface*
(data dostępu: 04.2017)

[2] *http://walter.schreppers.com/index.php?page=blogpost&pos=98*
(data dostępu: 04.2017)

[3] Maciej Borowiec, *Przygotowanie drukarki 3d reprap prusa mednel i3 do pracy z nowymi materiałami o podwyższonej temperaturze topnienia. Analiza parametrów cieplnych drukarki,* Praca dyplomowa inżynierska, Politechnika Krakowska, 2016

[4] *http://reprap.org/wiki/FDM* (data dostępu: 04.2017)

[5] E. Canessa C. Fonda, M. Zennaro*, L O W- C O S T 3D P R I N T I N G FOR SCIENCE, EDUCATION & SUSTAINABLE DEVELOPMEN*

[6] *https://en.wikipedia.org/wiki/Rapid_prototyping* (data dostępu: : 04.2017)

[7] Anna Kaziunas France*, tłum.* Zbigniew Waśko*. Świat druku 3D. Przewodnik. Kompendium wiedzy o druku 3D!* Gliwice: Helion, 2014

[8] *http://reprap.org/wiki/Prusa_Mendel/pl* (data dostępu: 11.2015)

[9] Arkadiusz Merta, *Raspberry Pi jak zacząć?*, Młody Technik 8/2014

[10] Eben Upton, Gareth Halfacree, *Raspberry Pi - Przewodnik użytkownika,* Gliwice: Helion, 2013

[11] *https://www.debian.org*
(data dostępu: 08.2017)

[12] *https://www.raspberrypi.org/products/raspberry-pi-3-model-b*
(data dostępu: 08.2017)

[13] *https://botland.com.pl/moduly-i-zestawy-raspberry-pi-3/5576-raspberry-pi-3-model-b-wifi-bluetooth-1gb-ram-12ghz.html#Specyfikacja* (data dostępu: 08.2017)

[14] Arkadiusz Merta, *Raspberry Pi: UART,* Młody Technik, 2014

[15] Dr Jan Kurzyk, mgr Jarosław Dąbrowski, *Internet,* Materiały dydaktyczne na studia podyplomowe "Informatyka dla nauczycieli", Kraków, Politechnika Krakowska Wydział Fizyki, Matematyki i informatyki, 2012

[16] Mark Sportack, *Sieci komputerowe. Księga eksperta. Wydanie II poprawione i uzupełnione,* Gliwice: Helion, 2004

[17] *https://home.cern/topics/birth-webxcvxcvxc* (data dostępu: 08.2017)

[18] *http://info.cern.ch/hypertext/WWW/TheProject.html* (data dostępu: 08.2017)

[19] *https://popul.ifj.edu.pl/historia/* (data dostępu: 08.2017)

[20] *https://pl.wikipedia.org/wiki/Apache_HTTP_Servers/* (data dostępu: 08.2017)

[21] *https://wiki.apache.org/httpd/ApacheVirtualHostMysql/httpd.conf* (data dostępu: 08.2017)

[22] *https://blog.sloniupl.eu/2016/01/09/instalacja-lamp-linux-apache-mysql-i-php-na-systemie-debian-8/* (data dostępu: 08.2017)

[23] *https://pl.wikipedia.org/wiki/Webmaster* (data dostępu: 08.2017)

[24] Radosław Sokół, *Kurs - Tworzenie stron internetowych,* Gliwice: Helion, 2007

[25] *http://www.kurshtml.edu.pl/* (data dostępu: 08.2017)

[26] Tom Negrino, Dori Smith, *Po prostu JavaScript i Ajax, wyd. IV,* Gliwice: Helion, 2007

[27] *https://pl.wikipedia.org/wiki/JScript* (data dostępu: 08.2017)

[28] Steven Holzner, *PHP 5 - Radocha z programowania,* Gliwice: Helion, 2006

[29] *https://pl.wikipedia.org/wiki/PHP* (data dostępu: 08.2017)

[30] Marcin Lis, *PHP - 101 praktycznych skryptów,* Gliwice: Helion, 2003

[31] Luke Welling, Laura Thomson, *PHP i MySQL - Tworzenie stron WWW. Vademecum profesjonalsity,* wyd. IV, Gliwice: Helion, 2009

[32] *http://php.net/docs.php*
(data dostępu: 08.2017)

[33] *https://pl.wikibooks.org/wiki/Zanurkuj_w_Pythonie*
(data dostępu: 08.2017)

[34] *https://pl.wikipedia.org/wiki/Git_(oprogramowanie)*
(data dostępu: 08.2017)

[35] *http://git-scm.com/book/pl/v1/*
(data dostępu: 08.2017)

[36] *https://getcomposer.org/doc/*
(data dostępu: 08.2017)

[37] *http://itcraftsman.pl/composer-czyli-jak-zarzadzac-zaleznosciami-w-php/* (data dostępu: 08.2017)

[38] Mike Cantelon, Marc Harter, TJ Holowaychuk, Nathan rajlich, *NodeJS w akcij ,* wyd. IV, Gliwice: Helion, 2009

[39] *https://pl.wikipedia.org/wiki/Npm_(manager_pakiet%C3%B3w)*
(data dostępu: 08.2017)

[40] *https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo* (data dostępu: 08.2017)

[41] *https://pl.wikipedia.org/wiki/Framework* (data dostępu: 08.2017)

[42] *https://poznajprogramowanie.pl/czym-jest-framework-i-po-co-go-uzywac/* (data dostępu: 08.2017)

[43] *https://jquery.com/* (data dostępu: 08.2017)

[44] *httos://pl.wikipedia.org/wiki/JQuery* (data dostępu: 08.2017)

[45] Bogdan Brinzarea-Iamandi, Cristian Darie, Audra Hendrix*, AJAX i PHP - Tworzenie interaktywnych aplikacji internetowych,* wyd. II, Gliwice: Helion, 2011

[46] *http://getbootstrap.com/* (data dostępu: 08.2017)

[47] *https://pl.wikipedia.org/wiki/Bootstrap_(framework)* (data dostępu: 08.2017)

[48] Jeff Dickey*, Nowoczesne aplikacje internetowe- Mongo DB, Express, AngularJS, Node.js* , Gliwice: Helion, 2016

[49] Brad Dayley, *Node.js, MongoDB, AngularJS - Kompendium wiedzy,* Gliwice: Helion, 2015

[50] Łukasz Sosna*, yii framework,* Gliwice: Helion, 2014

[51] Alexander Makarov, *Tworzenie aplikacji z Yii - Receptury,* Gliwice: Helion, 2014

[52] Lorna Jane Mitchell*, API nowoczesnej strony WWW. Usługi sieciowe w PHP,* Gliwice: Helion, 2015

[53] *http://yarpo.pl/2012/07/29/rest-ciekawszy-sposob-na-komunikacje-client-server/* (data dostępu: 08.2017)

[54] *http://commint.pl/baza/tworzenie-aplikacji/244-wymagania-niefunkcjonalne-aplikacji-internetowej* (data dostępu: 05.2018)

[55] *http://www.is.umk.pl/~grochu/wiki/doku.php?id=zajecia:ppz:analiza_wymagan* (data dostępu: 05.2018)

[56] *http://www.laro.com.pl/pdf/hm4100f-5v.pdf* (data dostępu: 07.2018)

[57] *http://www.onsemi.com/pub/Collateral/PN2222-D.PDF*
(data dostępu: 07.2018)

[58] *http://www.if.pwr.wroc.pl/~popko/w4/Lab/1*
(data dostępu: 08.2018)

[59] *http://www.serwis-elektroniki.com.pl/wp-content/uploads/2017/02/170221_sr_2.pdf*
(data dostępu: 08.2018)

[60] *http://mega.nz/#F!11wB0K4a*
(data dostępu: 08.2018)