

**Informatyka, Bezpieczeństwo systemów informatycznych
Wydział Elektryczny
Politechnika Poznańska**

Projekt Podstawy teleinformatyki
**Tablica informacyjna oraz aplikacja do jej zdalnej
obsługi**

Krzysztof Nitschke - 122094
krzysztof.nitschke@student.put.poznan.pl
Marek Wojciechowski - 122088
marek.wojciechowski@student.put.poznan.pl

Prowadzący zajęcia
mgr inż. Przemysław
Walkowiak

Spis treści

1. Charakterystyka ogólna projektu	3
2. Architektura systemu	3
3. Wymagania systemu	4
Wymagania funkcjonalne	4
Wymagania нефunkcjonalne	5
4. Opis najważniejszych protokołów	5
5. Diagramy UML	5
8. Wykorzystywane narzędzia programistyczne i zespołowe oraz biblioteki	10
9. Aplikacja kliencka - oraz jej działanie	11
13. Podział prac	25
14. Problemy	26
15. Uzasadnienie projektu	26
16. Rozwój aplikacji	26
17. Doświadczenie wyniesione z projektu	26
18. Instrukcja użytkownika	27

1. Charakterystyka ogólna projektu

Projekt zakłada utworzenie tablicy informacyjnej - postawionej na mikrokomputerze Raspberry Pi 3, która będzie miała możliwość wyświetlania filmów oraz obrazów znajdujących się na jej dysku.

Aby włączyć tryb wyświetlania na tablicy, musimy zalogować się na konto administratora, lub jedno ze stworzonych przez administratora w aplikacji zarządzania tablicą na komputerze znajdującym się w tej samej sieci co Raspberry.

Na aplikacji zarządzającej na wejściu mamy możliwość podania adresu IP tablicy, zalogowania się. Jeżeli logowanie się powiedzie, zostajemy przeniesieni do okna z opcjami aplikacji.

Pierwszą opcją jest wyświetlenie listy katalogu ze zdjęciami znajdującymi się na dysku raspberry, wybranie konkretnych z listy do wyświetlania oraz wpisanie czasu przez który ma wyświetlać się pojedyncze zdjęcie.

Następną opcją aplikacji jest wyświetlanie listy katalogu z filmami znajdującymi się na dysku Raspberry, wybranie konkretnych z listy do wyświetlania oraz wpisanie czasu przez który ma wyświetlać się pojedynczy film.

Kolejną opcją jest ustawienie czasu wyłączenie wyświetlania na tablicy, oraz ostatnią opcją dotyczącą tablicy jest zatwierdzenie wszystkich ustawionych opcji oraz przesłanie ich na tablicę.

Dodatkowo każdy użytkownik posiadający konto, może zmienić hasło, a główny administrator ma możliwość dodania kont, lub ich usunięcia.

2. Architektura systemu

System ma składać się z aplikacji klienckiej z pełną funkcjonalnością, dostępnej wraz z bazą danych do zarządzania kontami użytkowników oraz z aplikacji wyświetlającej informacje przeznaczonej na Raspberry Pi 3 z systemem operacyjnym Raspbian.

Aplikacja do zarządzania, posiada prostą bazę danych - SQLite, gdzie przechowuje loginy oraz hasła użytkowników których zarejestrował główny administrator, a także wysyła i odbiera w odpowiednim momencie komendy, które w odpowiedni sposób interpretuje aplikacja tablicy lub ona sama. Komendy te przesyłane i odbierane są poprzez ustanowione połączenie TCP w obu aplikacjach.

Przykładowe komendy:

- **vid** - żądanie przesłania listy filmów znajdujących się w folderze video tablicy
- **pic** - żądanie przesłania listy zdjęć znajdujących się w folderze pictures tablicy
- **svid** - lista wybranych filmów do wyświetlania na tablicy, wraz z czasem ich wyświetlania
- **spic** - lista wybranych zdjęć do wyświetlania na tablicy, wraz z czasem ich wyświetlania
- **start** - rozpoczęcie wyświetlania tablicy
- **time** - żądanie przesłania aktualnego czasu urządzenia
- **time2** - przesłanie czasu w którym tablica ma przestać wyświetlać

3. Wymagania systemu

Wymagania funkcjonalne określają ogólne funkcje, czynności i usługi realizowane przez aplikację dotyczące zachowania systemu oraz wymagania dotyczące danych.

Wymagania pozafunkcjonalne specyfikują pożądane cechy tworzonego systemu, są związane bezpośrednio z ograniczeniami i właściwościami aplikacji dot. wydajności, bezpieczeństwa, przenośności i niezawodności

- Wymagania funkcjonalne

Dla niezalogowanego użytkownika:

- a) Logowanie użytkownika - okna tekstowe na login i hasło oraz sąsiadujący przycisk do logowania.

Dla zalogowanego użytkownika:

- a) Zmiana hasła.
- b) Wybór filmów znajdujących się w folderze tablicy do wyświetlenia.
- c) Ustawienie czasu wyświetlania filmów.
- d) Wybór zdjęć znajdujących się w folderze tablicy do wyświetlenia.
- e) Ustawienie czasu wyświetlania zdjęć.
- f) Wyświetlenie aktualnego czasu maszyny z tablicą.
- g) Ustawienie czasu wyłączenie wyświetlania tablicy.
- h) Rozpoczęcie wyświetlania informacji na tablicy.
- i) Wylogowanie użytkownika.

Dla zalogowanego administratora:

- a) Zmiana hasła.
- b) Dodanie nowego użytkownika.
- c) Usunięcie użytkownika.
- d) Wybór filmów znajdujących się w folderze tablicy do wyświetlenia.
- e) Ustawienie czasu wyświetlania filmów.
- f) Wybór zdjęć znajdujących się w folderze tablicy do wyświetlenia.
- g) Ustawienie czasu wyświetlania zdjęć.
- h) Wyświetlenie aktualnego czasu maszyny z tablicą.
- i) Ustawienie czasu wyłączenia wyświetlania tablicy.
- j) Rozpoczęcie wyświetlania informacji na tablicy.
- k) Wylogowanie użytkownika.

- Wymagania niefunkcjonalne

- a) Hasła użytkowników przechowywane w bazie danych.
- b) Klient aplikacji działający w systemie Windows.
- c) Aplikacja klienta napisana w C#.
- d) Aplikacja tablicy działająca na Raspbianie.
- e) Aplikacja tablicy napisana w Pythonie 3.
- f) Interfejs aplikacji w języku polskim.
- g) Krótkie czasy odpowiedzi z bazy danych.
- h) Krótkie czasy komunikacji między aplikacjami.
- i) Połączenie aplikacji poprzez TCP.

4. Opis najważniejszych protokołów

TCP - Protokół sterowania transmisją (*Transmission Control Protocol*) – połączeniowy, niezawodny, strumieniowy protokół komunikacyjny stosowany do przesyłania danych między procesami uruchomionymi na różnych maszynach, będący częścią szeroko wykorzystywanego obecnie stosu TCP/IP (korzysta z usług protokołu IP do wysyłania i odbierania danych oraz ich fragmentacji wtedy, gdy jest to konieczne). Protokół sterowania transmisją operuje w warstwie transportowej modelu OSI. Opracowano go na podstawie badań Vintona Cerfa oraz Roberta Kahna. Został opisany w dokumencie RFC 793.

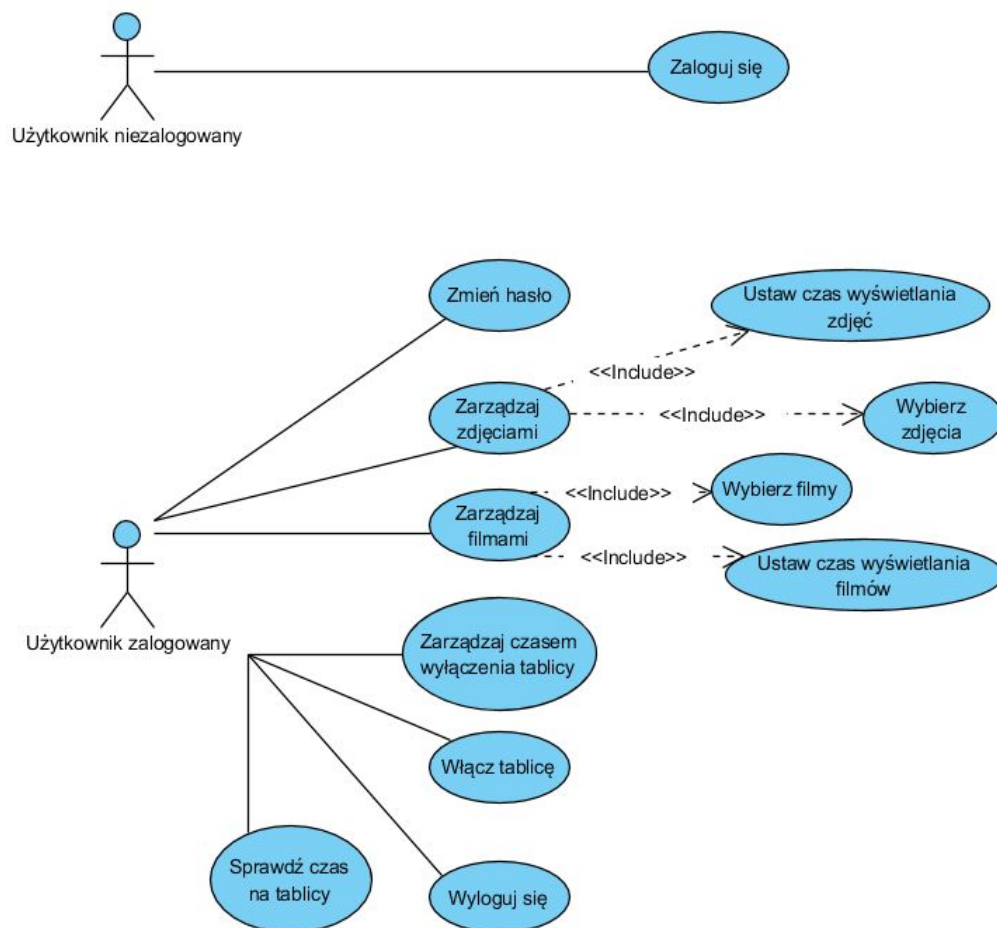
5. Diagramy UML

UML to język pół - formalny wykorzystywany do modelowania różnego rodzaju systemów, szczególnie często systemów informatycznych bez względu na

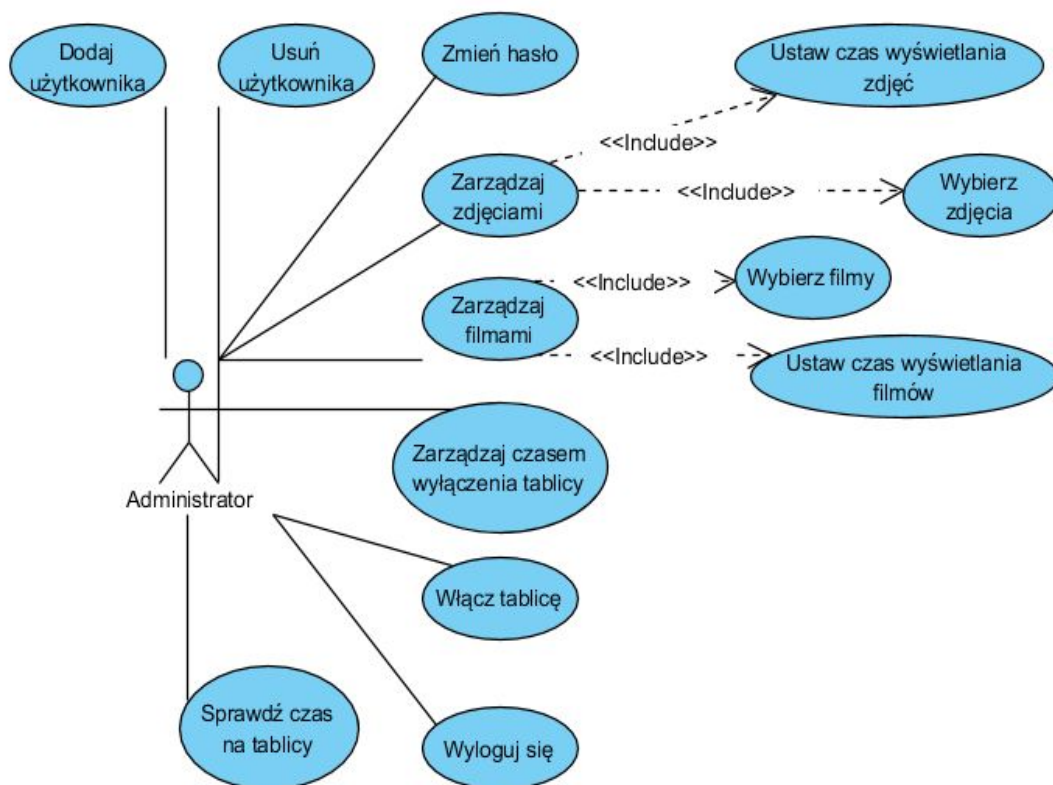
przeznaczenie. Diagramy UML są wykorzystywane ze względu na jednakową interpretację i łatwość zrozumienia dla wszystkich członków zespołu. Zdefiniowano 13 różnych rodzajów diagramów, które dzielą się na opisujące strukturę systemu i opisujące zachowanie.

Poniższe diagramy wygenerowane zostały w programie **Visual Paradigm**.

- Diagram przypadku użycia(ang. use case) z podziałem na aktorów(użytkownik niezalogowany/zalogowany) przedstawia podstawowe funkcjonalności systemu wraz z otoczeniem, które są widoczne z zewnątrz systemu.

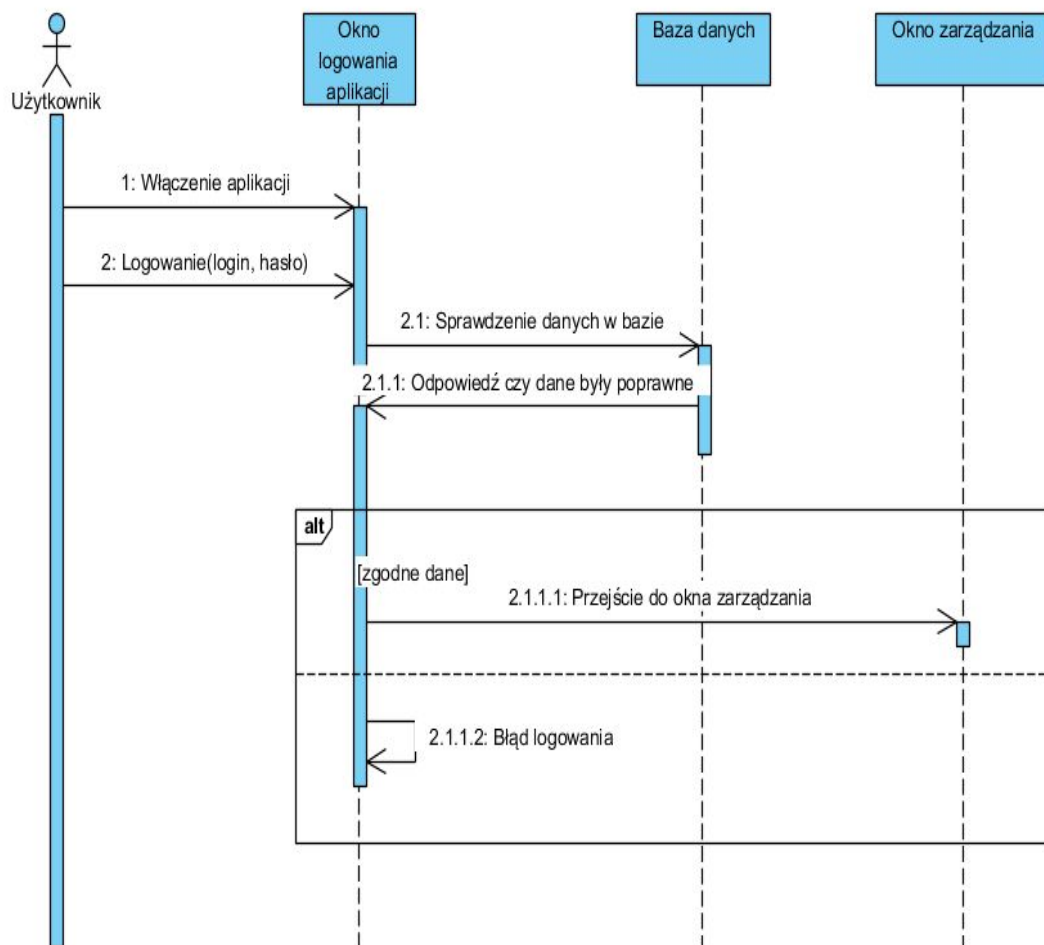


Rys. 1 cz.1 Diagram UML przypadku użycia z podziałem na aktorów

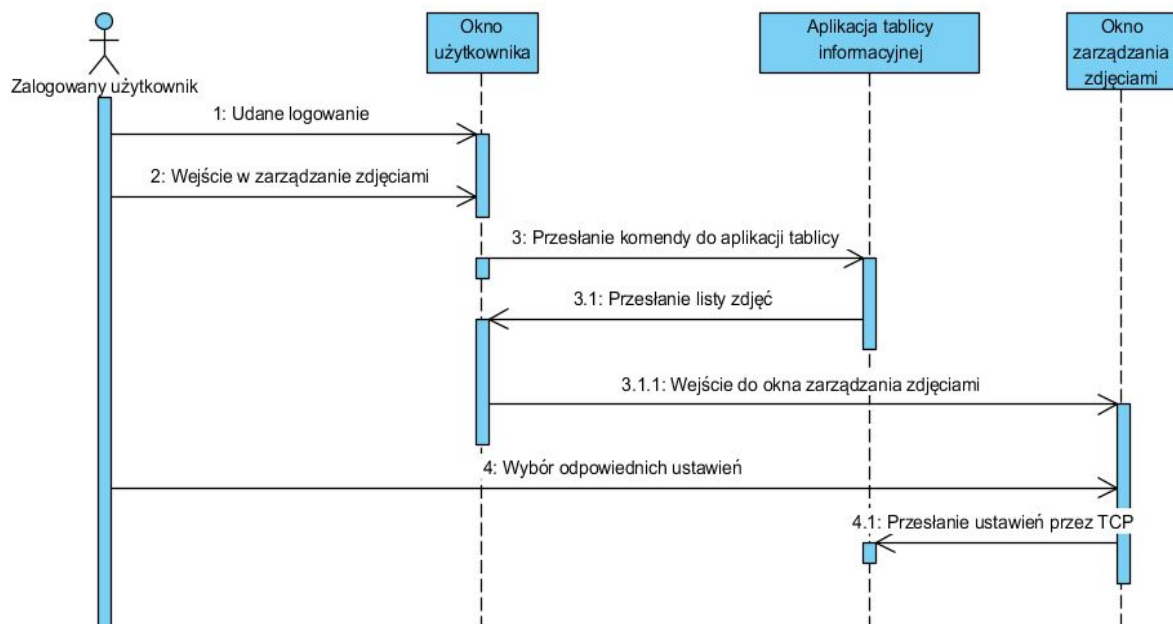


Rys. 1 cz.2 Diagram UML przypadku użycia z podziałem na aktorów

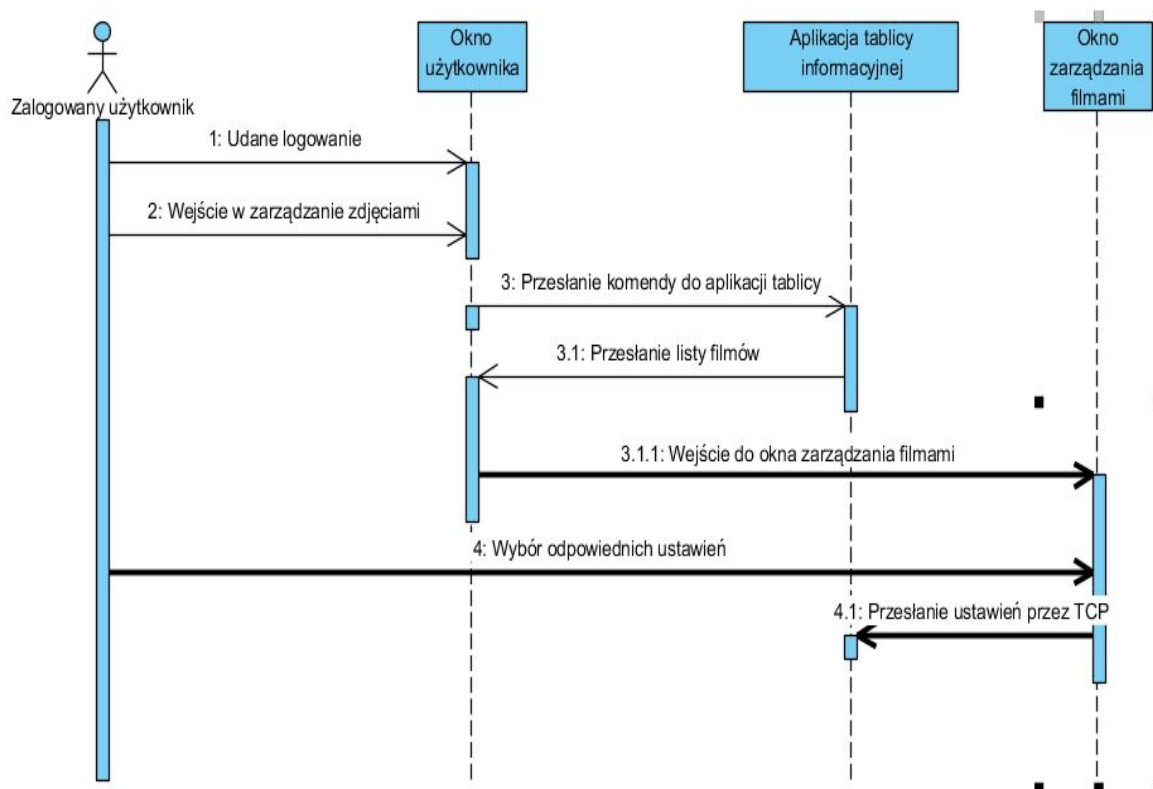
- Diagram sekwencji służy do prezentowania interakcji pomiędzy obiektami wraz z uwzględnieniem w czasie komunikatów, jakie są przesyłane pomiędzy nimi. Obiekty ułożone są wzdłuż osi X, a komunikaty i odpowiedzi systemu wzdłuż osi Y. Stosowany jest do przedstawienia zachowania systemu w kontekście różnych scenariuszy.



Rys. 2 Diagram sekwencji - proces logowania użytkownika

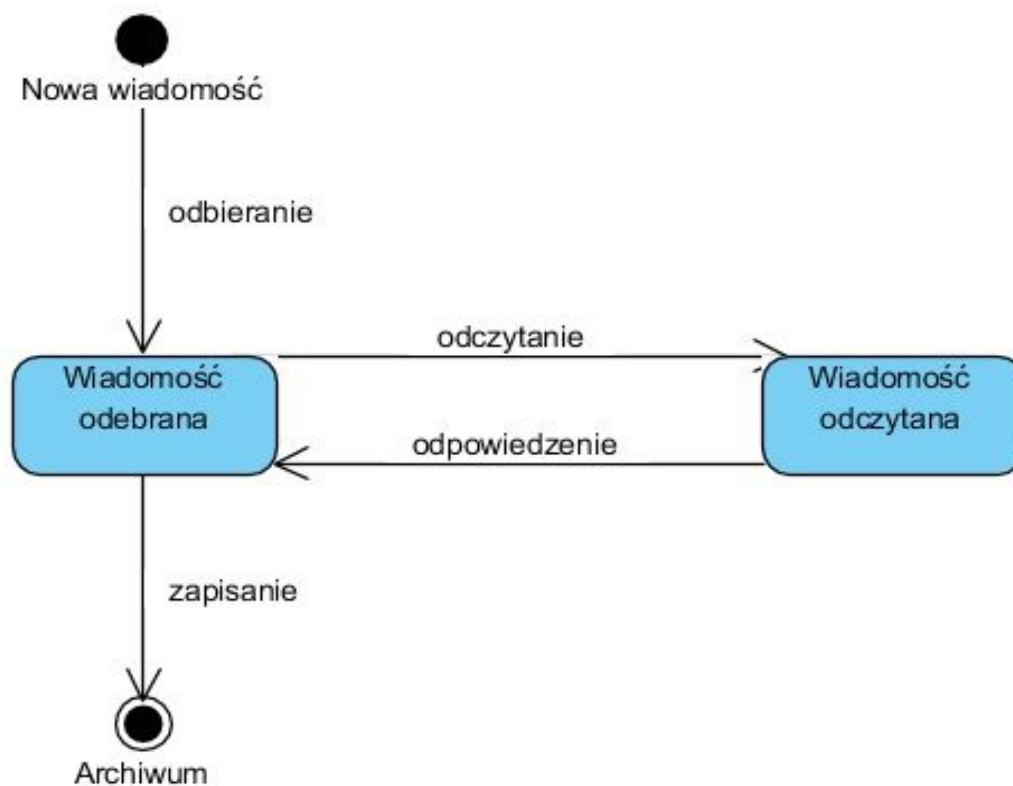


Rys. 3 Diagram sekwencji ustawień zdjęć



Rys. 4 Diagram sekwencji ustawień filmów

- Diagram stanów w notacji UML pozwala na kompletne i jednoznaczne określenie stanów obiektu i zdarzeń powodujących zmiany.



Rys. 5 Diagram stanu wiadomości odebranej przez tablicę

8. Wykorzystywane narzędzia programistyczne i zespołowe oraz biblioteki

1. **Środowisko programistyczne** - Visual Studio 2017, Spyder.
2. **Baza danych SQLite** - do zadania potrzebna jest niewielka lokalna baza danych.
3. **System kontroli wersji GIT** - serwis hostujący GitHub
4. **Diagramy UML** - Visual Paradigm 2017 - darmowy program do tworzenia diagramów
5. **Biblioteka graficzna Tkinter** - wykorzystana do ustawienia wyświetlania zdjęć na raspberry w trybie pełnoekranowym

- 6. Putty** -Bezpłatny program będący klientem usług TELNET, SSH i rlogin, działający pod systemami operacyjnymi Microsoft Windows oraz Unix/Linux. Aplikacja została stworzona przez Simona Tathama i rozpowszechniana jest na licencji MIT.

PuTTY emuluje terminal tekstowy, co pozwala w łatwy sposób łączyć się z serwerem za pomocą jednego z następujących protokołów: TELNET, rlogin, SSH-1.

W naszym przypadku wykorzystujemy protokół SSH-1.

9. Aplikacja kliencka - oraz jej działanie

Interfejs stworzony za pomocą Windows Form w Visual Studio 2017.

- Ekran startowy zawiera okno logowania użytkownika oraz okno na adres IP tablicy informacyjnej.

The image shows a Windows-style window titled "log" with a blue background. It contains three text input fields and a button. The first field is labeled "Podaj IP tablicy:" and contains the text "192.168.0.16". The second field is labeled "Podaj login:" and contains the text "admin". The third field is labeled "Podaj hasło:" and contains six asterisks "*****". Below the fields is a button labeled "Zaloguj".

Rys. 6 Okno logowania aplikacji

Po naciśnięciu zaloguj, pole IP wpisywane jest w pamięci aplikacji w celu nawiązywania połączeń TCP, a pola loginu oraz hasła przesyłane są w zapytaniu do bazy danych w celu ich weryfikacji.

- Podczas logowania wysyłane jest zapytanie do bazy danych SQLite, mające sprawdzić czy istnieje w bazie danych rekord o podanych danych (login, hasło). Fragment kodu odpowiedzialnego za sprawdzenie w bazie admina:

```

string sql = "select * from Administrator where login='" + Functions.login +
    "'" and password='" + Functions.pwd + "'";";
string sql2= "select * from BoardController where login='" + Functions.login +
    "'" and password='" + Functions.pwd + "'";";
SQLiteCommand command = new SQLiteCommand(sql, Functions.m_dbConnection);
SQLiteCommand command2 = new SQLiteCommand(sql2, Functions.m_dbConnection);
SQLiteDataReader reader = command.ExecuteReader();

while (reader.Read())
{
    if (reader["login"].ToString() != null)
    {
        Functions.admin = 1;
        Form1 f = new Form1();
        f.Show();
        this.Hide();
    }
}

```

Rys. 7 Kod odpowiedzialny za weryfikację logowania administratora

- Po udanym logowaniu użytkownikowi wyświetla się okno z opcjami aplikacji, Użytkownik ma możliwość wyboru zdjęć i filmów do wyświetlania, dodatkowe opcje to ustawienie czasu wyłączenia aplikacji. Zarządzanie kontami dotyczy tylko konta administratora - pozwala mu tworzyć nowe konta oraz usuwać istniejące. Użytkownik ma także możliwość zmiany hasła.



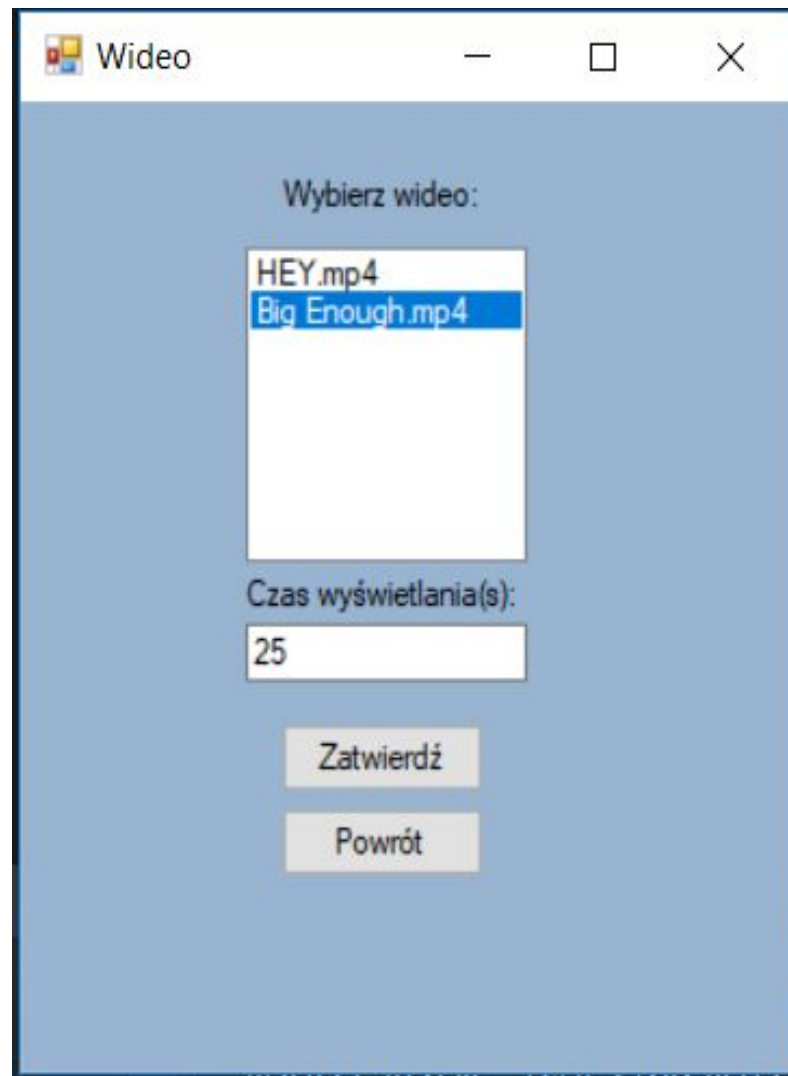
Rys. 8 Okno opcji aplikacji

Opcje dotyczące zarządzania tablicą posiadają wbudowane połączenie TCP, przez co przesyłają komendy oraz odbierają niektóre dane. Aplikacja tablicy posiada wbudowane rozróżnianie wszystkich komend otrzymywanych od aplikacji zarządzającej.

```
if req=='pic':
    msg=showfiles(req)
    print("pic")
if "spic" in req:
    i=0
    temp=""
    piclist=req.split(":")
    print(piclist)
    display_time_img=int(piclist[1])
```

Rys. 9 Przykładowy kod tablicy odpowiedzialny za weryfikowanie komend przesyłanych przez opcję aplikacji

- Wybór opcji video wysyła żądanie do wylistowania wszystkich plików znajdujących się w folderze videos tablicy. Aplikacja umożliwia wybór konkretnych filmików oraz ustawienie ich czasu wyświetlania. Zatwierdzenie wysyła komunikat z wybranymi opcjami.



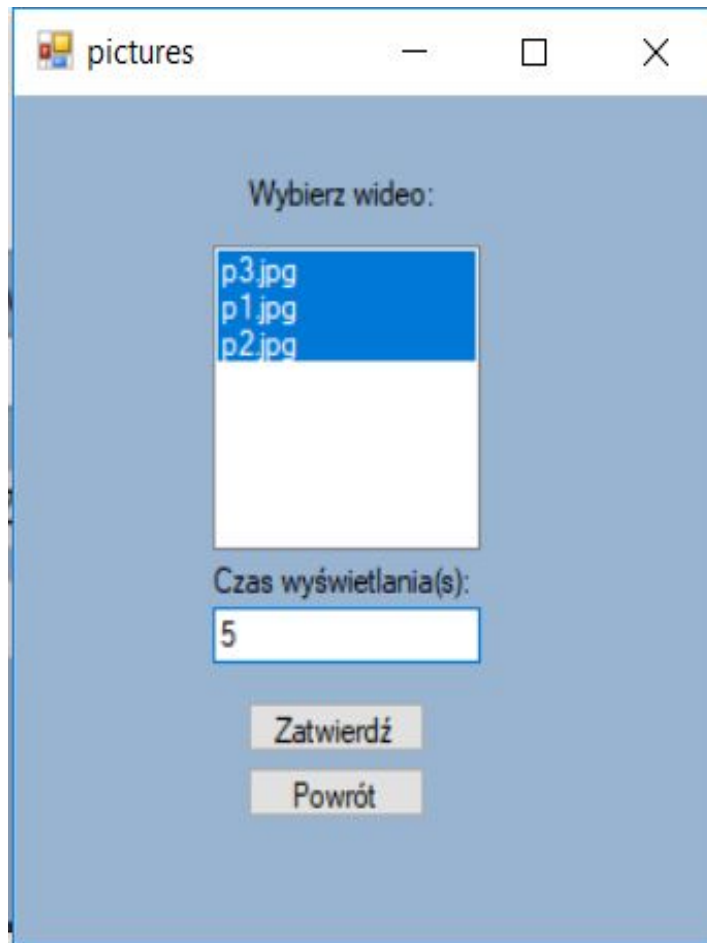
Rys. 10 Okno opcji zarządzania wyświetlaniem wideo

Wyświetlanie wideo na tablicy, odbywa się poprzez wykorzystanie z omxplayera, playera wideo stworzonego specjalnie na system Raspbian. Wykorzystany jest poprzez komendę stworzenia potoku z odpowiednim filmem znajdującym się na liście wybranych filmów. Po ustalonym czasie proces jest wyłączany.

```
if len(vidlist)>2:
    while y!=len(vidlist):
        timex=str(datetime.datetime.time(datetime.datetime.now()))
        movie1="/home/pi/Videos/"+vidlist[y]
        omxc = Popen(['omxplayer', '-b', movie1])
        time.sleep(display_time_movie)
        os.system('killall omxplayer.bin')
        y=y+1
```

Rys. 11 Fragment kodu odpowiedzialny za wyświetlanie filmów na tablicy

- Wybór opcji zdjęcia wysyła żądanie do wylistowania wszystkich plików znajdujących się w folderze pictures tablicy. Aplikacja umożliwia wybór konkretnych zdjęć oraz ustawienie ich czasu wyświetlania. Zatwierdzenie wysyła komunikat z wybranymi opcjami.



Rys. 11 Okno opcji zarządzania wyświetlaniem zdjęć

Przy użyciu biblioteki PIL (python imaging library) odpowiednio ustawiane są parametry wyświetlania zdjęć, tak aby były one wyświetlane na pełen ekran.

```
imgWidth, imgHeight = pilImage.size
if imgWidth > w or imgHeight > h:
    ratio = min(w/imgWidth, h/imgHeight)
    imgWidth = int(imgWidth*ratio)
    imgHeight = int(imgHeight*ratio)
    pilImage = pilImage.resize((imgWidth, imgHeight), Image.ANTIALIAS)
image = ImageTk.PhotoImage(pilImage)
```

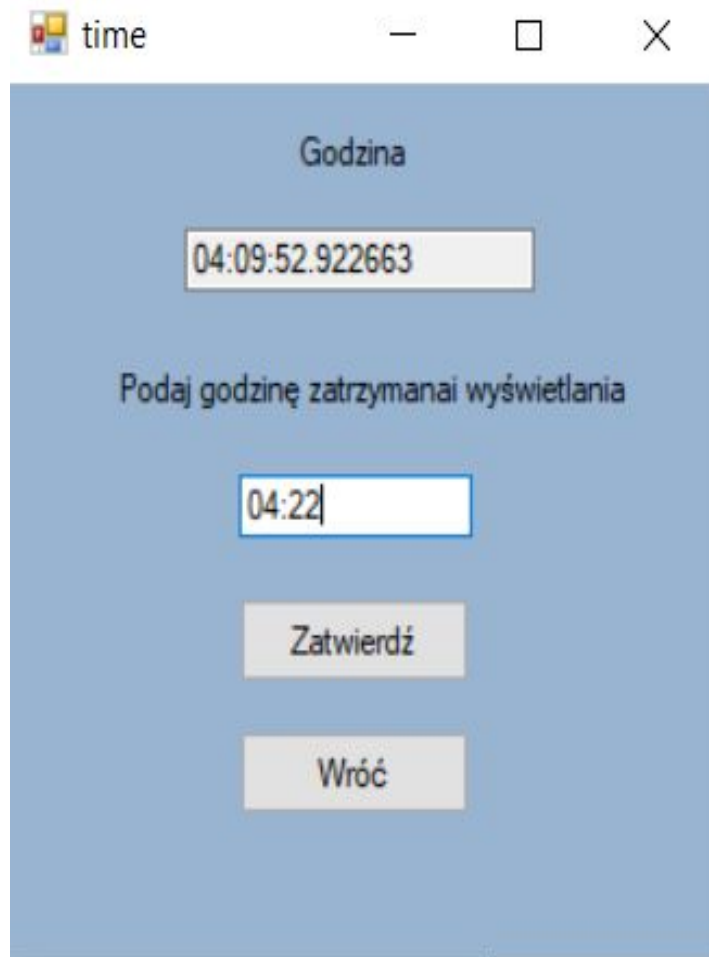
Rys. 12 Fragment kodu tablicy odpowiedzialny za wyświetlanie zdjęć w odpowiednim formacie

Lista plików znajdujących się w folderach tablicy wyszukiwana jest poprzez funkcję `os.walk()` - listującą wszystkie pliki w danym folderze

```
def showfiles(code):
    if code=='pic':
        for root, dirs, files in os.walk("/home/pi/Pictures/."):
            files1=""
            for filename in files:
                files1+=":"+filename
            print(files1)
            return files1
    if code=='vid':
        for root, dirs, files in os.walk("/home/pi/Videos/."):
            files1 = ""
            for filename in files:
                files1 += ":"+filename
            print(files1)
            return files1
```

Rys. 13 Fragment kodu tablicy odpowiedzialny za listowanie plików w folderach

- Wybranie opcji dodatkowe opcje wysyła do tablicy zapytanie o godzinę systemową płytki. Jest ona odsyłana poprzez połączenie TCP. Użytkownik ma możliwość nastawienie dokładnej godziny zaprzestania wyświetlania tablicy. Poprzez zatwierdzenie do tablicy przesyłany jest komunikat z dokładnymi ustawieniami.



Rys. 14 Okno zarządzania z możliwością ustawienia czasu wyłączenia tablicy

Przetwarzanie odebranych komunikatów dotyczących czasu poprzez tablicę. W podanym niżej kodzie time zwraca aktualny czas do aplikacji zarządzającej, a time2 przypisuje czas wyłączenia aplikacji do zmiennych, aby je później porównać.

```
if "time" in req:
    msg=str(datetime.datetime.time(datetime.datetime.now()))
    timex=msg
    print("X")
if "time2" in req:
    abc=req.split(":")
    checkhr=abc[1]
    checkmin=abc[2]
    print(abc[1])
    print(abc[2])
```

Rys. 15 Okno zarządzania z możliwością ustawienia czasu wyłączenia tablicy

Fragment kodu odpowiedzialny za sprawdzenie/porównanie aktualnego czasu z czasem wyłączenia:

```
yy=timex.split(':')
print(yy[0])
print(yy[1])
if int(yy[0])>=int(checkhr) and int(yy[1])>=int(checkmin):
```

- dostępna tylko dla administratora głównego opcja zarządzania kontami użytkowników. Posiada możliwość dodania i usunięcia konta. Wszystkie działania te wysyłają do bazy danych odpowiednie zapytania



The image shows a screenshot of a Windows application window titled "add". The window has a standard Windows title bar with a minimize button, a maximize button, and a close button. The main content area has a light blue background. It contains two text input fields: the first is labeled "Login:" and the second is labeled "Hasło:". Below these fields are three buttons stacked vertically: "Dodaj konto", "Usuń konto", and "Powrót".

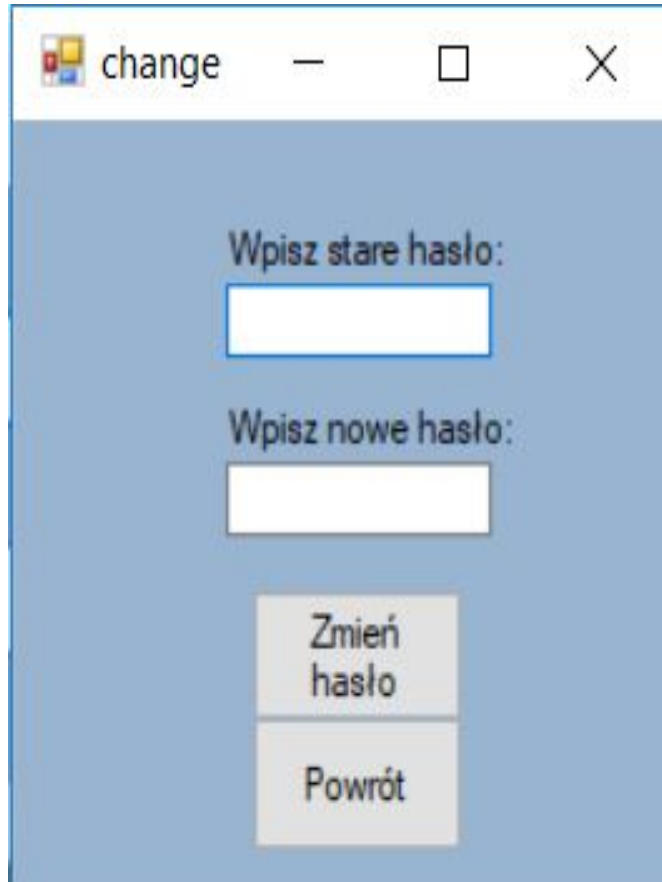
Rys. 16 Okno zarządzania kontami aplikacji

Kod aplikacji uruchamiający się po naciśnięciu odpowiedniego przycisku wysyła odpowiednie komendy do bazy danych. Funkcja wewnętrzna, nie łączy się z aplikacją tablicy.

```
private void button1_Click(object sender, EventArgs e)
{
    string sql= "insert into BoardController(login, password) values('" +
    textBox1.Text+"', '" + textBox2.Text + "')";
    SQLiteCommand command = new SQLiteCommand(sql, Functions.m_dbConnection);
    command.ExecuteNonQuery();
    Form1 f = new Form1();
    f.Show();
    this.Hide();
}
//del
1 reference | knitschke, 6 hours ago | 1 author, 1 change
private void button2_Click(object sender, EventArgs e)
{
    string sql = "delete from BoardController where login='" + textBox1.Text + "'";
    SQLiteCommand command = new SQLiteCommand(sql, Functions.m_dbConnection);
    command.ExecuteNonQuery();
    Form1 f = new Form1();
    f.Show();
    this.Hide();
}
```

Rys. 17 Fragment kodu aplikacji odpowiedzialnego za dodawanie i usuwanie użytkownika

- Okno zmiany hasła użytkownika. Zalogowany użytkownik wpisując hasło którym się zalogował w pierwszy wiersz jest w stanie zmienić hasło na to wpisane w drugim wierszu, poprzez naciśnięcie odpowiedniego przycisku.



The image shows a graphical user interface window titled "change". The window has a light blue background and a white title bar with standard Windows window controls (minimize, maximize, close). Inside the window, there are two text input fields. The first field is preceded by the label "Wpisz stare hasło:" and the second by "Wpisz nowe hasło:". Below the input fields, there are two buttons stacked vertically. The top button is labeled "Zmień hasło" and the bottom button is labeled "Powrót".

Rys. 18 Okno zmiany hasła

Fragment kodu porównujący pierwsze wpisane hasło z tym wpisanym do okienka. Jeżeli oba hasła nie będą zgodne - komenda nie zostanie wysłana do bazy danych.

```
private void button1_Click(object sender, EventArgs e)
{
    string match = textBox1.Text;
    if (match == Functions.pwd) {
        string sql = "update BoardController set password='" + textBox2.Text +
            "' where login='" + Functions.login + "'";
        SQLiteCommand command = new SQLiteCommand(sql, Functions.m_dbConnection);
        command.ExecuteNonQuery();

        Form1 f = new Form1();
        f.Show();
        this.Hide();
    }
}
```

Rys. 19 Fragment kodu aplikacji odpowiedzialnego za zmianę hasła

13. Podział prac

Krzysztof Nitschke:

- implementacja aplikacji tablicy
 - system komunikacji
 - system wyświetlania
 - system interpretowania komend
- przygotowanie mikrokomputera Raspberry PI
- pomoc w implementacji interfejsu użytkownika
- przygotowanie dokumentacji

Marek Wojciechowski:

- zaprojektowanie interfejsu użytkownika
- zaimplementowanie interfejsu użytkownika
- połączenie aplikacji za pomocą TCP
- przygotowanie dokumentacji

14. Problemy

Problem z wyświetlaniem plików na raspberry Pi. Do rozwiązania problemu musieliśmy wykorzystać odtwarzacz przeznaczony na system operacyjny raspbian.

Były to dla nas w większości nowe technologie i początkowo mieliśmy problem z wykorzystaniem danej technologii. Lecz po pewnym czasie dochodziliśmy do rozwiązań wykorzystując różne fora internetowe i dokumentację.

15. Uzasadnienie projektu

Temat projektu wybraliśmy, ponieważ chcieliśmy poznać możliwości płytki SBC. Od dawna interesowały nas możliwości tej płytki i chcieliśmy spróbować, poznać jak ją wykorzystywać. Szczególnie raspberry pi. Płytki ma ogromne możliwości i jak nadarzyła się okazja żeby przekonać się o jej potencjale, postanowiliśmy skorzystać z tej okazji. Bardzo nam się spodobało programowanie tej płytki zwłaszcza, że nie zamyka nas od wyboru języka programowania. Można programować w javie, c++, pythonie itd. My zdecydowaliśmy się na pythona. Wybór swój uzasadniamy chęcią poznania nowego języka. Wcześniej mało korzystaliśmy z niego. Bardzo sporadycznie proste algorytmy dodawania usuwania elementów z tablicy. Natomiast chcieliśmy się przekonać, o możliwościach tego języka. W połączeniu z płytką SBC. Projekt nam się spodobał, jego pomysł i możliwości. Zawsze można rozwinąć w różne kierunki. Nie zamykamy się w ramach jednej wizji.

16. Rozwój aplikacji

Kierunków rozwoju aplikacji jest kilka. Można wykorzystać ten projekt w różnych dziedzinach np. w szkole jako tablica informacyjna. Można np. w sporcie do wyświetlania czy to na basenie czy wakeparku reklam czy sprawdzania, ile komu zostało czasu do wykorzystania. Możliwość wyświetlania zdjęć, filmów daje nam możliwość prezentacji swoich możliwości w różnych dziedzinach, aby się zaprezentować. Płytki raspberry pi ma bardzo dużo możliwości. Możemy dołączyć różne moduły np. aby sprawdzać obecność na lekcji poprzez wykorzystanie opasek z modułem.

17. Doświadczenie wyniesione z projektu

Projekt dał nam dużo odpowiedzi co do naszej wiedzy i współpracy w grupie. Byliśmy odpowiedzialni nie tylko za siebie ale i za całą grupę. Poznaliśmy jak wykorzystać urządzenie SBC i jego możliwości. Dowiedzieliśmy się z czym wiąże się wykorzystanie tej

technologii i na co należy zwracać uwagę. Po utworzeniu harmonogramu, wzięliśmy się za wybór technologii potrzebnych do stworzenia projektu. Trzeba było zwracać na szczegóły, które przy małych projektach nie występują. Musieliśmy nauczyć się współpracy i dzielenia obowiązków przy tworzeniu naszego zadania. Dysponowanie czasem, który musieliśmy dzielić z innymi obowiązkami jak praca, inne projekty na uczelni. Poznaliśmy swoje lepsze i gorsze strony z używanych technologii. Kolejnym doświadczeniem, które nabyliśmy na było przygotowywanie i prezentacja przed grupą. Mieliśmy okazję na sprzedanie swojego projektu i zainteresowanie nim innych uczestników zajęć. Jak się okazało nie jest to wcale takie proste zadanie jakby mogło się wydawać. Nie wszystkie nasze wybory okazywały się trafne, więc trzeba było korygować błędy na bieżąco.

18. Instrukcja użytkownika

1. Na urządzeniu Raspberry Pi włączyć aplikację tablicy, która rozpocznie nasłuchiwanie TCP na porcie 5007.
2. Włączyć aplikację zarządzającą na maszynie podłączonej do sieci lokalnej - tej samej co Raspberry.
3. Podać IP raspberry oraz zalogować się (przy pierwszym logowaniu login/hasło administratora to admin/admin, hasło to należy zmienić od razu po zalogowaniu).
4. Jeżeli zajdzie taka potrzeba, dodać nowych użytkowników.
5. Ustawić odpowiednio tablice (wszystkie opcje są opcjonalne):
 - wybrać zdjęcia, ustawić ich czas wyświetlania
 - wybrać filmy, ustawić ich czas wyświetlania
 - ustawić czas wyłączenia tablicy
6. Włączyć tablice.

Przykładowy wygląd wyjścia tablicy:

Fragment filmu:



Wyświetlane zdjęcie:

