POLITECHNIKA POZNAŃSKA
WYDZIAŁ ELEKTRYCZNY, INFORMATYKA
Instytut Automatyki, Robotyki i Inżynierii Informatycznej

PODSTAWY TELEINFORMATYKI

Prowadzący: mgr inż. Przemysław Walkowiak

Projekt zespołowy

Tablica Informacyjna - Dokumentacja

Oliwia Bartosik
nr indeksu: 122029
grupa: TI-1 L4

Radosław Michałek
nr indeksu:
grupa: TI-1 L4

14 czerwca 2019

Na wstępie chciałabym zaznaczyć, że dokumentacja ta została napisana dla projektu spełniającego wszystkie wymagania opisane przez prowadzącego zajęcia. Podjęłam taką decyzję, gdyż gdybym miała napisać dokumentację dla tego, co udało nam się stworzyć podczas zajęć z Podstaw Teleinformatyki, to ten dokument składałby się z kilku, a nie kilkudziesięciu stron. Doszłam do wniosku, że dokumentacja to ważna część tego przedmiotu, w związku z tym postaram się opisać wszystko co planowałam zrobić, aby ten projekt działał i był kompletny.

Oliwia Bartosik

1. Opis i uzasadnienie

Tematem naszego projektu jest implementacja oprogramowania do wyświetlania informacji na elektronicznej tablicy, działającego na urządzeniu Raspberry Pi pod systemem Linux. Pozwoli ono na zdalną administrację, strumieniowanie obrazu z kamer oraz wyświetlanie dokumentów HTML, zdjęć, krótkich filmów i zdalnych stron WWW.

Przeglądając listę tematów udostępnioną przez prowadzącego zajęcia, to właśnie ten przykuł moją uwagę. Zawsze zastanawiało mnie dokładne działanie tablic informacyjnych, które w dzisiejszych czasach znajdują się na każdym kroku: w pracy, na uczelni, na przystankach, dworcach, w urzędach, przychodniach i innych instytucjach. Jedyne co potrafiłam wywnioskować z tego co widziałam, to to, że te tablice działają na systemie Windows, gdyż za każdym razem jak jakaś tablica uległa awarii, wyświetlał się piękny blue screen albo sławetne okno z czerwonym “X” i informacją o błędzie.

Temat, który wybrałam, miał pewne założenia, a mianowicie takie, że oprogramowanie miało działać pod systemem operacyjnym Linux, na urządzeniu SBC. Polecone zostało wykorzystanie do tego zadania technologii Qt. Tak jak z pracą z mikrokontrolerami miałam już doświadczenie, tak o technologii Qt nigdy nie słyszałam. Coś nowego, ciekawego, innego. W związku z tym jako kolejny powód wybrania takiego tematu było to, że mogłam nauczyć się wielu nowych rzeczy. Na co dzień nie korzystam z systemu Linux, więc wykonanie tego projektu pomogło mi jeszcze bardziej zaznajomić się z tym systemem, przez co teraz czuję się w nim pewniej. Jako urządzenie SBC wybrałam mikrokontroler Raspberry Pi, gdyż był łatwo dostępny. Ten projekt pozwolił mi na poszerzenie wiedzy i umiejętności z zakresu projektowania systemów wbudowanych i aplikacji webowych.

Popularność takich tablic dała mi do zrozumienia, że są one potrzebne, więc zaimplementowanie oprogramowania obsługującego

wyświetlanie informacji na elektronicznych tablicach jest pracą pożyteczną. Taką wiedzę można wykorzystać w przyszłości.

2. Podział prac

Oliwia Bartosik	frontend
Radosław Michałek	backend

3. Funkcjonalności oferowane przez aplikację

Zdalna administracja przez panel WWW

Aplikacja umożliwia zdalną administrację przez panel WWW. Administrator, po wejściu na odpowiednią stronę WWW, może się zalogować do systemu, po czym wyświetla mu się panel. W nim może wgrywać pliki, ustalać czas wyświetlania dla każdego pliku oraz datę ważności. Może również dodać krótką wiadomość tekstową, wyświetlaną na pasku z informacjami.

Możliwość wyświetlania materiałów różnego typu

Oprogramowanie jest zdolne do wyświetlania dokumentów HTML, obrazów JPG, PNG oraz GIF, krótkich filmów, streamowanych obrazów z kamer oraz zdalnych stron WWW. Posiada ono również pasek, na którym wyświetlone zostają wiadomości tekstowe. Wiadomości te przesuwają się z prawej strony do lewej, przez co tekst może składać się z kilku zdań. Tablica wyświetla również teraźniejszą godzinę oraz datę. Każdy obraz, odnośnik lub film ma możliwość ustawienia terminów w jakich będzie wyświetlany (np. od 10 czerwca do 23 czerwca) oraz czas wyświetlania (np. 10 sekund).

Podgląd przez panel WWW

Panel umożliwia również podgląd wyświetlanego aktualnie obrazu. Jest to możliwe dzięki zduplikowaniu widoku z mikrokontrolera do panelu WWW, który posiada obszar z podglądem tego, co jest wyświetlane w danej chwili na tablicy.

Funkcjonalności dodatkowe

Oprogramowanie umożliwia wyświetlanie tego samego widoku na wielu monitorach i jest odporne na tymczasowe błędy.

4. Wybrane technologie, środowiska programistyczne oraz narzędzia informatyczne

Raspberry Pi 3 B+

Należy on do grupy komputerów jednopłytkowych wbudowanych - SBC (ang. *single-board computer*). Jest to minikomputer składający się z pojedynczego obwodu drukowanego. Choć jego wygląd jest niepozorny, ma ogromne możliwości. Na małej płytce znajdują się:

- **procesor BCM2837B0** - Chipset Broadcom BCM2837B0 z 64-bitowym rdzeniem Quad-core ARM-8 Cortex-A53 CPU, taktowany 1,4 GHz
- **pamięć RAM** - 1GB LPDDR3 @ 900 MHz
- złącza
    - **USB 2.0** x 4 - można pod nie podłączyć myszkę, klawiaturę, kartę WiFi czy pendrive. Ilość gniazd można zwiększyć stosując zewnętrzny HUB USB
    - **Ethernet** - możliwość bezpośredniego podłączenia do sieci LAN
    - **HDMI** wersji 1.4 - przez złącze to można przesyłać zarówno obraz jak i dźwięk.
    - **gniazdo jack** 3,5 mm - można do niego podłączyć słuchawki, głośniki lub wyprowadzić wideo poprzez standard RCA Composite
    - **DSI** - wyświetlacz dotykowy
    - **CSI** - kamera
- moduł **Wi-Fi** Dual Band - wbudowany moduł WiFi Dual Band 802.11 b/g/n/ac działa w zakresie 2,4 GHz i 5 GHz
- moduł **Bluetooth** 4.2 - umożliwia przesył danych za pośrednictwem interfejsu Bluetooth
- czytnik kart **micro SD** - karta pamięci SD jest dyskiem tego minikomputera. Na nim wgrywa się system operacyjny.
- **GPIO** (40-pin) - wejścia/wyjścia ogólnego przeznaczenia. Wśród nich znajdują się piny obsługujące interfejsy: I2C, SPI oraz UART.

Urządzenie obsługuje:

- **zasilanie**
  - przez złącze **microUSB**
  - **PoE** - możliwość zasilania przez sieć Ethernet za pomocą dodatkowej nakładki typu HAT
- **kompresję H.264** (1080p30) oraz **grafikę OpenGL** ES 1.1 i 2.0

Raspbian v. 4.14

Darmowy system operacyjny oparty na Debianie, zoptymalizowany dla urządzeń Raspberry Pi. SO Raspbian to zestaw podstawowych programów i narzędzi pozwalający na uruchomienie minikomputera. Oprócz czystego systemu operacyjnego zawiera ponad 35,000 pakietów wraz z wstępnie skompilowanym oprogramowaniem w przystępnym formacie dla łatwej instalacji Raspbiana na Raspberry Pi.

Ubuntu v. 19.04

System operacyjny firmy Canonical Ltd. oparty na Debianie, który ma opinię wyjątkowo stabilnego systemu. W związku z tym, że oparty jest na sprawdzonych i stabilnych pakietach, to czas jaki upływa od powstania nowej technologii czy narzędzia, do ostatecznych testów i włączenia jej do głównych pakietów jest bardzo długi. Programiści Ubuntu "zamrażają" pakiety testowe Debiana, dostosowują je do dystrybucji, dołączają najnowsze środowisko graficzne oraz prostą, użyteczną konfigurację. W ten sposób wydawane są co pół roku kolejne wersje Ubuntu Linux.

Zdecydowałam się na wykorzystanie Ubuntu, ze względu na prostą instalację. Nie musiałam spędzać dużej ilości czasu na konfigurację systemu, ponieważ wszystko było już gotowe do użytku. Dodatkowo Ubuntu nie zajmuje dużej ilości pamięci na dysku, co było dla mnie ważne, gdyż Linux był dodatkowym systemem na moim komputerze, a nie głównym.

Można zadać sobie pytanie, dlaczego zdecydowałam się na kolejny system z Linuxem, skoro na Raspberry Pi został zainstalowany Raspbian. Otóż, dla ułatwienia sobie pracy. Programowanie na komputerze jest

wygodniejsze. A skoro Debian i Ubuntu są do siebie zbliżone, to to, co napiszę na Ubuntu, powinno działać na SBC.

Python

Python to interpretowany, obiektowy, wysokopoziomowy język programowania. Jego składnia cechuje się przejrzystością oraz zwięzłością. Posiada w pełni dynamiczny system typów i automatyczne zarządzanie pamięcią. Często używany jest jako język skryptowy. Działa na wielu platformach, takich jak: Windows, Linux czy Mac OS.

Postawiłam na język Python, ponieważ w semestrze, w którym wykonywałam ten projekt, bardzo często wykorzystywaliśmy ten język, w szczególności do uczenia maszynowego. Stwierdziłam, że skoro będę często korzystać z Pythona, łatwiej będzie mi skupić się właśnie na tym języku. Myślę, że zawsze warto rozwijać swoje umiejętności w programowaniu w Pythonie.
Do tego projektu wykorzystuję Pythona w wersji 3.7.

Qt 5.12.3

Qt to coś więcej niż tylko wieloplatformowe SDK. To strategia technologiczna, która pozwala szybko i efektywnie zaprojektować, rozwinąć, wdrożyć i utrzymać oprogramowanie zapewniając jednocześnie bezproblemowe użytkowanie przez wszystkie urządzenia. Działa na różnych oprogramowaniach i platformach sprzętowych takich jak Linux, Windows, macOS, Android lub systemach wbudowanych.
Qt to zestaw przenośnych bibliotek i narzędzi programistycznych dedykowanych dla języków C++, QML i Java.

QML

Deklaratywny język programowania oparty na JavaScript, służący do projektowania aplikacji silnie związanych z interfejsem graficznym. QML (*Qt Meta Language*, *Qt Modeling Language*) podobny jest do CSS i JSON. Jest częścią Qt Quick, komponentu Qt służącego do tworzenia interfejsów

graficznych. Dokument QML opisuje drzewo obiektów. Moduły QML dostarczone wraz z Qt obejmują graficzne elementy konstrukcyjne (np. *Rectangle*, *Image*), komponenty modelowania (np. *FolderListModel*, *XmlListModel*), komponenty behawioralne (np. *TapHandler*, *DragHandler*, *State*, *Transition*, *Animation*) oraz bardziej złożone elementy sterujące (np. *Button*, *Slider*, *Drawer*, *Menu*).

Elementy QML mogą być rozszerzone przez standardowy JavaScript zarówno w kodzie QML jak i poprzez dołączone pliki .js. Elementy można również rozszerzyć o komponenty C++ przy użyciu frameworka Qt.

PyCharm 2019.1.3 (Community)

To zintegrowane środowisko programistyczne firmy JetBrains, obsługujące język Python, CSS, HTML oraz JavaScript. Posiada funkcje podpowiadania składni, analizy i refaktoryzacji kodu źródłowego. Umożliwia łatwe nawigowanie pomiędzy plikami, klasami, metodami itp. PyCharm jest zintegrowany z systemami kontroli wersji Mercurial, Subversion, Git, CVS oraz Perforce. Wspiera programowanie i tworzenie aplikacji internetowych w Django.

Qt Creator 4.9.1 (Community)

Wieloplatformowe środowisko programistyczne dla języków C++, JavaScript oraz QML, będące częścią SDK dla biblioteki Qt. Umożliwia tworzenie aplikacji na komputery oraz urządzenia mobilne. Zawiera edytor kodu oraz zintegrowany Qt Designer dla projektowania i budowania GUI.
Qt Creator jest zintegrowany z zestawem narzędzi, takich jak:

- systemy kontroli wersji - Git, Subversion, Perforce, Bazaar, CVS Mercurial,
- Qt Simulator - narzędzie do testowania aplikacji Qt przeznaczonych dla urządzeń mobilnych.

PySide2

Oficjalny moduł Pythona dla Qt for Python, który zapewnia dostęp do

frameworka Qt 5.12+.

Flask

Mikro framework WSGI (*Web Server Gateway Interface*) dla aplikacji webowych. Został zaprojektowany, aby umożliwić szybkie i łatwe rozpoczęcie pracy. Flask nazywany jest mikroframeworkiem, ponieważ nie wymaga szczególnych narzędzi ani bibliotek. Nie posiada abstrakcyjnej warstwy bazy danych, walidacji form ani żadnych innych komponentów, gdzie istniejące biblioteki stron trzecich zapewniają wspólne funkcje. Flask sugeruje, zamiast wymuszać stosowanie konkretnych zależności czy układów projektu. To do programisty należy wybór narzędzi i bibliotek, z których chce skorzystać. Istnieje wiele rozszerzeń dostarczonych przez "społeczność", które ułatwiają dodawanie nowych funkcji.

GitHub

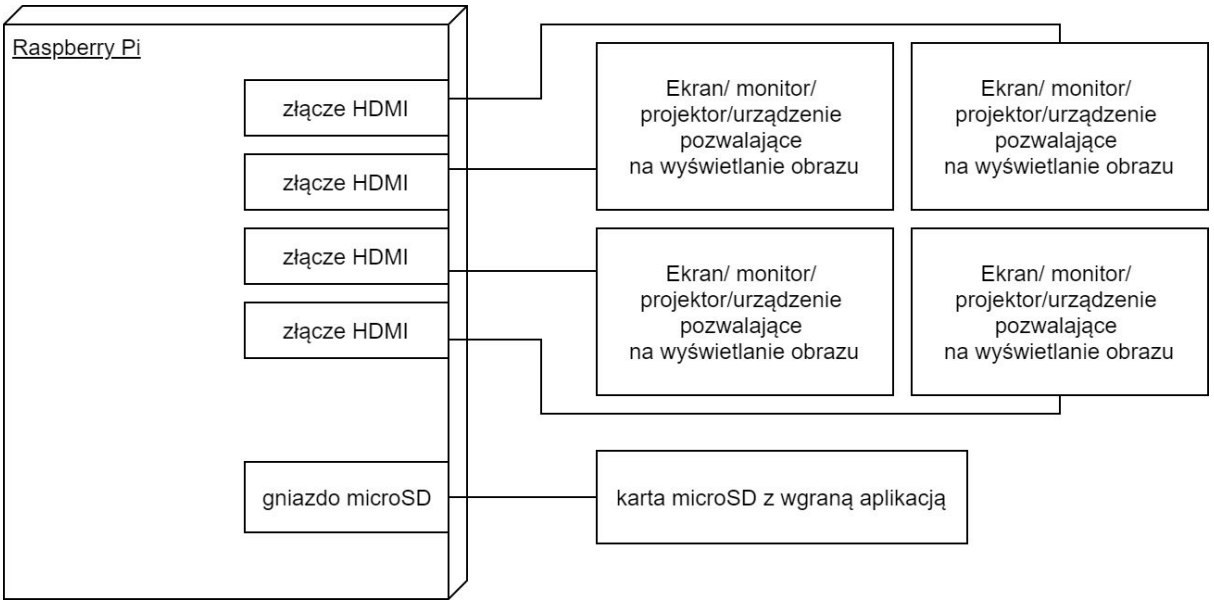
Serwis hostingowy do kontroli wersji, korzystający z Git. Pozwala na przechowywanie kodu w repozytorium, do którego mają dostęp wszystkie osoby posiadające link. Pozwala na łatwe analizowanie historii implementacji kodu. Każdy commit można opatrzyć w komentarz z informacjami na temat wprowadzonych zmian. Z jednego repozytorium może korzystać większa ilość osób, przez co cały zespół ma wgląd w postępy w pracy.

VirtualBox v. 6.0.8

Program do wieloplatformowej wirtualizacji, który pozwala uruchamiać wiele systemów operacyjnych na komputerach z systemem Windows, Linux, Mac OS, Solaris oraz OpenSolaris. W VirtualBoxie można tworzyć i zarządzać maszynami wirtualnymi systemów Windows, Linux, BSD, OS/2, Solaris, Haiku oraz OSx86.

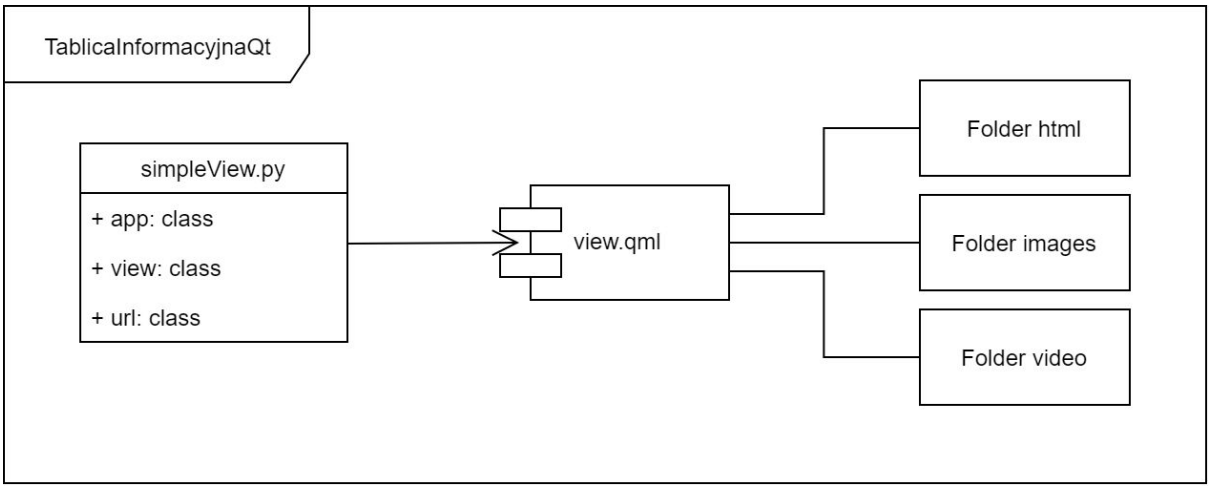
VirtualBox był mi potrzebny do tego, by móc korzystać z Ubuntu.

5. Architektura rozwiązania



Rys. 1 - Schemat przedstawiający najważniejsze połączenia między mikrokomputerem a komponentami potrzebnymi do poprawnego funkcjonowania oprogramowania

Ponieważ mikrokomputer Raspberry Pi 3 B+ posiada 4 złącza HDMI, można podłączyć do niego cztery urządzenia pozwalające na wyświetlanie obrazu. Na sprzęcie zainstalowany jest system operacyjny Raspbian, umożliwiający korzystanie z oprogramowania. Na karcie microSD zapisana jest aplikacja oraz wirtualne środowisko Pythona.



Rys. 2 - Schemat połączeń między modułami oraz folderami

Aplikacja składa się z dwóch plików oraz trzech folderów. Plik *simpleView.py* służy do uruchomienia całej aplikacji. W kodzie tego pliku znajduje się zmienna, która przyjmuje wartość klasy ze zmienną, pod którą podstawia się ścieżkę URL do pliku *view.qml*. Plik *view.qml* implementuje cały wygląd aplikacji. W nim wyświetlane są obrazy, wideo oraz pliki HTML. Ma on dostęp do plików umieszczonych w folderach *html*, *images* oraz *video*. Plik view.qml napisany jest w języku QML i składa się z modułów QML.

6. Problemy i ich rozwiązania

Głównym problem w realizacji tego projektu był brak jakiejkolwiek współpracy. Pierwszym założeniem było to, że ten projekt będę robić sama. Jednak zdecydowałam się wykonać tę pracę z kimś. Z perspektywy czasu uważam, że to była zła decyzja, ponieważ zamiast dwa razy bardziej sprawnej pracy, wszystko się posypało. Niestety (w tym przypadku) praca w grupie pozwala myśleć, że nie wszystko jest na głowie jednej osobie, że jedna osoba nie musi robić wszystkiego, że praca powinna być podzielona po równo na dwie osoby. Niestety, ponieważ ja myślałam w taki sposób i skończyłam z prawie niczym.

Podczas pracy w grupie członkowie zespołu powinni ze sobą rozmawiać, powinni być w częstym kontakcie. Powinni się starać, powinno im zależeć na oddaniu dobrze wykonanego projektu. Jak są jakieś problemy, powinno się je rozwiązywać wspólnie. Nie bez powodu mówi się, że co dwie głowy to nie jedna. W teorii praca w grupie powinna przebiegać sprawniej i bardziej efektywnie. Jednak w tym przypadku było zupełnie inaczej. W naszym "zespole" tylko jednej osobie zależało na oddaniu sprawnego programu. Tylko jedna osoba starała się przygotowywać coś na każdy sprint. Ta "współpraca" była dla mnie bardzo stresująca. Jestem osobą, która robi wszystko co w jej mocy, aby wszystko się udało, która przejmuje się, gdy widzi, że praca nie idzie tak jak powinna, że są opóźnienia. W co drugą środę byłam zestresowana tym, że osoba, z którą jestem w grupie, znowu nie ma nic do pokazania. Na początku wymuszałam na tej osobie jakiekolwiek zaangażowanie, zmuszałam ją do napisania czegoś na prezentacji, do wrzucania czegoś na GitHuba. Jednak w późniejszym czasie przestałam to robić, bo stwierdziłam, że szkoda moich nerwów. W ciągu tygodnia nie było takiej szansy, by porozmawiać o projekcie, przez co sama wybrałam sposób, w jaki wykonam swoją część zadania. Zaczęłam pracować w taki sposób, jakbym ten projekt robiła sama, bo wiedziałam, że nie mogę liczyć na drugą osobę. Niestety czas, który pozostał mi od dnia podjęcia takiej decyzji do końca semestru, był krótki i nie udało mi się wykonać wszystkiego co

zaplanowałam.

Bardzo ważne jest to z kim wykonuje się jakiś projekt. Czasami lepiej zrobić coś samemu, niż "pracować" z kimś i nic nie zrobić.

Chciałabym teraz napisać o problemach, które napotkałam podczas implementacji swojej części projektu. Pierwszym z nich był problem ze znalezieniem modułu Qt WebKit. Okazało się, że w piątej wersji Qt zostało usuniętych lub przemieszczonych wiele modułów. Ciężko było odnaleźć się w dokumentacji, bo gdy jedna rzecz działała, to druga przestała być wspierana przez Qt i trzeba było szukać nowego rozwiązania. To było dla mnie frustrujące, bo to nie był jedyny przypadek gdy napotkałam na swojej drodze tego typu problem. Na początku planowałam korzystać z PyQt 5, jednak zdecydowałam się wykorzystać PySide2, gdyż jest to teraz oficjalna biblioteka dla Pythona wspierana przez Qt. W PySide2 nie ma Qt WebKit. Został on zastąpiony przez Qt WebEngine.

Następnym problemem było zdecydowanie się na sposób, w jaki zaimplementować widok, który będzie wyświetlany na monitorze podłączonym do Raspberry Pi. Na początku chciałam wykorzystać do tego dodatek do Qt Creatora o nazwie Qt Designer. Niestety korzystając z Linuxa na maszynie wirtualnej, używanie Qt Designera stało się niemożliwością, ponieważ po każdym wstawieniu jakiegoś elementu na stół montażowy musiałam minimalizować okno, a następnie znowu je otwierać, by widok się zresetował. Stwierdziłam, że taka praca jest bez sensu, w związku z tym zaczęłam szukać innego sposobu na rozwiązanie tego problemu. W ten sposób natknęłam się na język QML. Przy użyciu QML-a udało mi się zaimplementować podstawowy widok. Lecz pojawił się tutaj następny problem. Nie potrafiłam sprawić, by wyświetlany tekst był przewijany z prawej strony do lewej. W tej sytuacji powstał pierwszy i ostatni commit na GitHubie pana Michałka. Rozwiązał ten problem korzystając z modułu NumberAnimation w module Text.

Zmianę zdjęć postanowiłam wykonać za pomocą stanów oraz timera. To była droga przez mękę. Zrozumienie działania stanów (a raczej ich nie działania) zajęło mi bardzo dużo czasu. Mimo to, nadal nie działają one tak

jak powinny. Udało mi się zaimplementować takie warunki, że te stany się zapętlają, ale nadal jest tam masa błędów. Niestety aplikacja lubi się wieszać. Nie każde przejście stanu jest płynne.

Miałam również problem z zaimportowaniem bibliotek potrzebnych do zaimplementowania modułu Video oraz WebView. Problem okazał się być bardzo trywialny. Biblioteki do Qt zainstalowały mi się w trzech różnych miejscach, a QML powiązany był z bibliotekami w takim folderze, gdzie tych bibliotek było najmniej. Umieściłam w tym folderze brakujące biblioteki i problem z importem zniknął. Po zaimplementowaniu odpowiednich modułów pojawił się problem z odtwarzaniem wideo. Było to spowodowane brakiem odpowiednich kodeków. Oczywiście zainstalowanie brakujących kodeków naprawiło problem, a po tym filmy wyświetlały się już bez problemu.

Kolejną kwestią, którą chciałabym poruszyć w dziale problemów są problemy, których nie udało mi się rozwiązać. Pierwszym z nich jest możliwość wczytywania plików z folderu. Niestety wszystkie pliki są wpisywane “z palca”. Próbowałam skorzystać z modelu `FolderListModel`, ale na niewiele się to zdało. Model ten umożliwia wyświetlanie plików w konkretnym formacie w formie listy. Pozwala również na odnoszenie się do konkretnego pliku poprzez nazwę listy oraz metodę folder. Niestety na końcu i tak należy podać nazwę pliku, więc jedyne do czego przydała mi się ta możliwość, to ustawianie widoczności pliku z konkretnego źródła poprzez odniesienie się do ID modelu Image.

7. Instrukcja użytkowania

Uruchomienie mikrokomputera oraz aplikacji

1. Użytkownik podłącza monitory do Raspberry Pi przy użyciu kabla ze złączem HDMI lub odpowiednią przejściówką, która posiada złącze HDMI męski, albo bezprzewodowo przy użyciu modułu Bluetooth.
2. Na minikomputerze znajduje się wirtualne środowisko Pythona o nazwie raspberryvenv, które ma zainstalowane biblioteki PySide2 oraz Flask. Wirtualne środowisko aktywujemy poprzez wywołanie w terminalu komendy:

```
source raspberryvenv/bin/activate.
```

3. Po wykonaniu tej czynności, w terminalu, przechodzimy do folderu, w którym znajduje się plik *simpleView.py*.
4. Uruchamiamy plik wywołując komendę:

```
python3 ./simpleView.py.
```

5. W taki oto sposób zostaje uruchomiona aplikacja do wyświetlania informacji na tablicy informacyjnej.

Obsługa aplikacji do wyświetlania

1. W celu wprowadzenia zmian do aplikacji, np. zmiana zdjęcia, filmu czy strony internetowej, potrzebny będzie program do edytowania tekstu, np. Notatnik, Notepad++, WordPad, albo bardziej zaawansowane oprogramowanie do edycji kodu programu.
2. W folderze, w którym jest plik *simpleView.py*, znajdują się trzy foldery: *html*, *images* oraz *video*. W folderze *images* umieszczamy pliki JPG, PNG oraz GIF, natomiast w folderze *video* umieszczamy pliki AVI.
3. Otwieramy plik view.qml w celu jego edycji.
4. Obrazy:
    a. format: jpg, png, gif
    b. wymiary: 1920p x 1080p, w przeciwnym razie obraz zostanie przeskalowany do wysokości 1080p
    c. rozmiar: obojętny, z zastrzeżeniem, że im większy rozmiar

pliku, tym aplikacja może się wieszać lub zacinać

d. stany, w których zmienia się ścieżkę źródłową pliku to: state 11, state 14, state 14, state, 31, state 33, state 35, state 4, state 51 oraz state 53
e. aby zmienić wyświetlany obraz, wystarczy w wybranym stanie zmienić nazwę pliku

5. Wideo:
    a. format: avi
    b. wymiary: 1920p x 1080p, w przeciwnym razie wideo zostanie przeskalowane do wysokości 1080p
    c. czas trwania: maksymalnie 10 sekund, w przeciwnym wypadku, gdy wideo będzie za długie, zostanie ono ucięte w dziesiątej sekundzie. Natomiast, gdy wideo będzie krótsze, ulegnie ono zapętleniu, aż dlugość wyświetlania wyniesie w sumie 10 sekund.
    d. rozmiar: pliki nie powinny mieć dużych rozmiarów, ponieważ aplikacja ulegnie zawieszeniu lub może się zacinać
    e. stany, w których zmienia się ścieżkę źródłową pliku to: state 12, state 2, state 32, state 52 oraz state 54
    f. aby zmienić wyświetlane wideo, wystarczy w wybranym stanie zmienić nazwę pliku
6. Pliki HTML:
    a. stany, w których zmienia się adres URL to: state 13, state 34 oraz state 55
    b. aby zmienić wyświetlany plik HTML, wystarczy w wybranym stanie wpisać odpowiedni adres URL
7. W celu zmiany wyświetlanego tekstu, wklejamy interesujący nas napis w module o id *message* w polu:

    ```
    text: qsTr(“miejsce na tekst”).
    ```

    Jeżeli tekst będzie przesuwał się w złym tempie, można to zmienić w module *NumberAnimation*, w polu *duration*.

8. Zapisujemy plik view.qml.
9. Uruchamiamy aplikację i oglądamy wprowadzone zmiany.

6. Bibliografia

[1] https://pl.wikipedia.org/wiki/Raspberry_Pi

[2] https://en.wikipedia.org/wiki/Single-board_computer

[3] https://www.raspberrypi.org/

[4] https://forbot.pl/blog/kurs-raspberry-pi-od-podstaw-wstep-spis-tresci-id23139

[5] https://botland.com.pl/pl/moduly-i-zestawy-raspberry-pi-3-b-plus/11142-raspberry-pi-3-model-b-wifi-dual-band-bluetooth-1gb-ram-14ghz-7131796402594.html

[6] https://www.raspbian.org/

[7] https://ubuntu.pl/

[8] https://ubuntu.pl/o-ubuntu/

[9] https://pl.python.org/

[10] https://pl.wikipedia.org/wiki/Python

[11] https://pl.wikipedia.org/wiki/PyCharm

[12] https://www.connectdistribution.pl/pl/dostawcy/jetbrains/pycharm.html

[13] https://pl.wikipedia.org/wiki/Qt_Creator

[14] https://www.qt.io/

[15] https://pl.wikipedia.org/wiki/Qt

[16] https://en.wikipedia.org/wiki/Qt_(software)

[17] https://en.wikipedia.org/wiki/Qt_Creator

[18] https://pl.wikipedia.org/wiki/QML

[19] https://en.wikipedia.org/wiki/QML

[20] https://pypi.org/project/PySide2/

[21] https://wiki.qt.io/Qt_for_Python

[22] https://pypi.org/project/Flask/

[23] https://www.palletsprojects.com/p/flask/

[24] https://en.wikipedia.org/wiki/Flask_(web_framework)

[25] https://en.wikipedia.org/wiki/GitHub

[26] https://en.wikipedia.org/wiki/VirtualBox

[27] https://www.oracle.com/pl/virtualization/virtualbox/