**Spis Treści:**

[**Opis:**](#_8wa5tcpuypbp) **2**

[**Instalacja**](#_qqc6nbmhyjxc) **2**

[**Pakiety**](#_ngzmqv6lsgbt) **2**

[Wykorzystane bezpośrednio](#_hh1gylnyijp4) 2

[Wykorzystane pośrednio (potrzebne dla innych)](#_n3vjopsj6svw) 2

[**Testowanie**](#_xspcgpduu691) **3**

[**Zrealizowane funkcjonalności**](#_h0c9mu473a9u) **3**

[**Opis Funkcjonalności**](#_azmrk7z56ig6) **4**

[**Opis klas:**](#_58tou0rfkdob) **6**

[Worker(QObject)](#_v974zodj9n6x) 6

[Opis](#_9atblbynn0o2) 6

[Atrybuty](#_1xxkw3yh47h) 6

[Metody](#_39cgrcizww0b) 6

[Song](#_693u9f3k0nxy) 7

[Opis](#_xnv2tkn9p7cd) 7

[Atrybuty](#_842wjn29qv90) 7

[Metody](#_phlu41ecvijy) 7

[Ui\_MainWindow(QMainWindow)](#_2nyfuylofc7r) 8

[Opis](#_scoop4d5rn0g) 8

[Atrybuty](#_pulumlkzfein) 8

[Metody](#_gf9w8nvai4to) 11

[Metody globalne](#_9recmmwnxg56) 15

[**Problemy**](#_7c0zcomtaan4) **15**

[**Rozwiązania**](#_2k5hy2jtpdvj) **15**

[**Zgodność z projektem**](#_rc29v0wparr5) **15**

**Odtwarzacz Mp3**

# Opis:

Mała aplikacja okienkowa napisana w Pythonie przeznaczona do odtwarzania plików w formacie “.mp3”. Umożliwia pobieranie informacji o słuchanej piosence (tytuł, autor, album, rok wydania, tekst) oraz ich zapisywanie.

# Instalacja

* Pobranie pliku wykonywalnego
* Pobranie i zainstalowanie Lav Filters (wymagane do poprawnego odtwarzania plików mp3)
  + wypakować pliki z paczki.
  + uruchomienie skryptu install\_audio jako administrator.
* Pobranie i instalacja ffmpeg (wymagane aby pobieranie informacji o piosence działało poprawnie)
  + wypakować pliki z paczki.
  + zmienić nazwę wypakowanego folderu na FFmpeg
  + dodać ścieżkę .\FFmpeg\bin do zmiennej środowiskowej path

do pobrania z:

<https://drive.google.com/drive/folders/1w2_vepZFko9UjR1a7BTyY1UfYFacH5sF?usp=sharing>

# Pakiety

## Wykorzystane bezpośrednio

* PyQt5
* ShazamAPI
* os
* sys

## Wykorzystane pośrednio (potrzebne dla innych)

* PyQt-Qt5
* PyQt-sip
* PyQt-stubs
* PyQt-WebEngine
* PyQt-WebEngine-Qt5
* algorithm
* altgraph
* certifi
* chardet
* dependency-injection
* dill
* ffmpeg
* future
* idna
* numpy
* pefile
* pip
* pydub
* pyinstaller
* pywin32-ctypes
* pyinstaller-hooks-contrib
* request
* setuptools
* simpleaudio
* urllib3

# Testowanie

Wysłałem plik .exe do kilku znajomych z prośbą aby przetestowali działanie aplikacji, samemu natomiast próbowałem wywołać każdą z dostępnych funkcjonalności na każdy możliwy sposób np. wybranie folderu gdy mam piosenki w playliście i gdy nie nie mam, anulowanie wybierania folderu gdy mam piosenki w playliście i gdy nie mam itd.

# Zrealizowane funkcjonalności

* Możliwość wybrania jednej piosenki do odtworzenia (3 - 5) - zrealizowane w terminie
* Możliwość zatrzymania i wznowienia słuchania (3 - 5) - zrealizowane w terminie
* Możliwość odtworzenia piosenki od nowa (3 - 5) - zrealizowane w terminie
* Możliwość wybrania folderu z piosenkami (3 - 5) - zrealizowane w terminie
* Gdy został wybrany folder dać możliwość przemieszczania się między piosenkami (3 - 5) - zrealizowane w terminie
* Wyświetlenie listy piosenek wybranych (3 - 5) - zrealizowane w terminie
* Możliwość tworzenia playlist z piosenek z różnych folderów (8 - 9)

- zrealizowane w terminie

* Możliwość zmiany nazw playlist (8 - 9) - zrealizowane w terminie
* Możliwość zapisywania playlist do pliku konfiguracyjnego (8 - 9)

- zrealizowane w terminie

* możliwość pobrania informacji o piosence i aktualizacja ich danych (8 - 9) - zrealizowane w terminie
* możliwość wyświetlania tekstu odsłuchiwania piosenki (8 - 9)

- zrealizowane w terminie

* zapamiętywanie do 5 poprzednio wybranych folderów z plikami mp3 (11-12)

- zrealizowane przed terminem

* oznaczanie piosenek i playlist jako ulubione (11-12) - zrealizowane przed terminem
* wyszukiwanie piosenek i playlist po nazwie (11-12) - zrealizowane przed terminem
* Suwak głośności (11-12) - zrealizowane przed terminem

**Dodatkowo z funkcjonalności perspektywicznych**

* Suwak długości piosenki i możliwość przewijania do wybranego momentu

- zrealizowane

* Opcja losowego odtwarzania piosenek z playlisty - zrealizowane

# Opis Funkcjonalności

* Wybierz piosenkę - usuwa piosenki z listy obecnie odtwarzanych i dodaje do tej listy piosenkę wybraną.
* Wybierz folder - usuwa piosenki z listy obecnie odtwarzanych i dodaje do tej listy piosenki w wybranym folderze.
* Dodaj piosenkę - do listy obecnie odtwarzanych dodawana jest wybrana piosenka.
* Dodaj folder - do listy obecnie odtwarzanych dodawane są piosenki w wybranym folderze.
* lista piosenek - wyświetla listę obecnie odtwarzanych piosenek i pozwala na zmianę kolejności ich odtwarzania (przeciągnij i upuść na właściwe miejsce).
* wyszukaj piosenkę - pozwala wyszukać po nazwie piosenkę zapisaną w jakiejkolwiek playliście
* historia folderów - pozwala sprawdzić listę do 5 poprzednio wybranych folderów oraz wybranie i wczytanie zawartości jednego z nich.
* stwórz playlistę - zapisuje listę piosenek obecnie odtwarzanych jako playlistę o nazwie podaną przez użytkownika. W przypadku podania powtarzającej się nazwy lub nie podania jej wcale nazwa zmieniona jest na akceptowalną.
* zmień nazwę playlisty - daje możliwość zmiany nazwy obecnie słuchanej playlisty.
* dodaj piosenkę do playlisty - pozwala dodać obecnie słuchaną piosenkę do wybranej playlisty.
* dodaj piosenkę do ulubionych - dodaje obecnie słuchaną piosenkę do playlisty “Ulubione”.
* przeglądaj zapisane playlisty - umożliwia przeglądanie playlist i wczytanie jednej z nich.
* pobierz informację piosenki - pobiera informacje dotyczące obecnie słuchanej piosenki.
* wyświetl informację piosenki - wyświetla informacje dotyczące obecnie słuchanej piosenki.
* wyświetl tekst piosenki - wyświetla tekst obecnie słuchanej piosenki (jeżeli taki został wcześniej pobrany).
* zmień kolory - zmienia styl graficzny GUI z ciemnego na jasny i z jasnego na ciemny.
* poprzednia - Przechodzi do poprzedniej piosenki na liście obecnie odtwarzanych.
* następna - Przechodzi do następnej piosenki na liście obecnie odtwarzanych.
* start/stop - zatrzymuje/startuje obecnie odtwarzaną piosenkę.
* zmiana głośności - zmienia głośność obecnie odtwarzanej piosenki.
* pasek odtwarzania - pozwala śledzić i zmienić czas obecnie odtwarzanej piosenki.
* wyświetlanie nazwy piosenki/playlisty - wyświetlanie nazwy piosenki i playlisty która obecnie jest odtwarzana.

# Opis klas:

## Worker(QObject)

### Opis

* + - Klasa opisująca zachowanie wątku tymczasowego przeznaczonego do pobierania informacji piosenek.

### Atrybuty

* + - path typu string - ścieżka piosenki której informacje chce pobrać
    - info typu pyqtSignal - atrybut globalny definiujący nadawany przez klasę sygnał.

### Metody

* + - \_\_init\_\_(self, path) - konstruktor klasy przyjmujący jeden argument path.
    - run(self) - metoda przeciążona bezargumentowa zawierająca instrukcje które wątek ma do wykonania po jego rozpoczęciu.

## 

## Song

### Opis

* + - Klasa przechowuje informacje na temat wczytanych i zapisanych plików.

### Atrybuty

* + - path - przechowuje ścieżkę pliku jako string
    - name - przechowuje nazwę piosenki/pliku jako string
    - author - przechowuje imię i nazwisko autora piosenki jako string (jeżeli znana)
    - album - przechowuje nazwę albumu jako string (jeżeli znana)
    - year - przechowuje datę wydania piosenki (jeżeli znana)

### Metody

* + - \_\_init\_\_(self, path, name=’’, author=’def’, album=’def’, year=’def’) - konstruktor klasy przyjmujący 5 argumentów z czego obligatoryjnym jest tylko path, pozostałe mają przypisane wartości domyślne
    - setName(self, name) - wywoływana na obiekcie klasy, przyjmuje string i przypisuje do zmiennej klasy name.
    - setAuthor(self, author) - wywoływana na obiekcie klasy, przyjmuje string i przypisuje do zmiennej klasy author.
    - setAlbum(self, album) - wywoływana na obiekcie klasy, przyjmuje string i przypisuje do zmiennej klasy album.
    - setYear(self, year) - wywoływana na obiekcie klasy, przyjmuje string i przypisuje do zmiennej klasy year.
    - \_\_str\_\_(self) - metoda bezargumentowa wywoływana na obiekcie klasy zwracająca format w jakim drukowany ma być obiekt klasy.

## 

## Ui\_MainWindow(QMainWindow)

### Opis

* + - Klasa dziedzicząca z QMainWindow z pakietu PyQt5.QtWidgets. Odpowiada ona za wygląd aplikacji i obsługę sygnałów nadawanych przez widgety w niej umieszczonych.

### Atrybuty

* + - centralwidget typu QtWidgets.QWidget(self) - tworzy główne okno aplikacji.
    - toolBar typu QtWidgets.QToolBar(self) - tworzy pasek narzędziowy.
    - statusbar typu QtWidgets.QStatusBar(self) - tworzy pasek statusu.
    - menubar typu QtWidgets.QMenuBar(self) - tworzy pasek opcji.
    - menuSong typu QtWidgets.QMenu(self.menubar) - tworzy rozwijany pojemnik na opcje związanymi z piosenkami.
    - menuInfo typu QtWidgets.QMenu(self.menubar) - tworzy rozwijany pojemnik na opcje związanymi z informacjami.
    - menuPlaylist typu QtWidgets.QMenu(self.menubar) - tworzy rozwijany pojemnik na opcje związanymi z playlistami.
    - menuOgolne typu QtWidgets.QMenu(self.menubar) - tworzy rozwijany pojemnik na opcje ogólne.
    - max\_vol typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zaznaczająca gdzie suwak głośności ma maksymalną wartość.
    - prev\_button typu QtWidgets.QPushButton(self.centralwidget) - tworzy przycisk do wybrania poprzedniej piosenki.
    - min\_vol typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zaznaczająca gdzie suwak głośności ma minimalną wartość.
    - next\_button typu QtWidgets.QPushButton(self.centralwidget) - tworzy przycisk do wybrania następnej piosenki.
    - song\_name typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zawierającą nazwę piosenki/pliku odtwarzanej/ego.
    - start\_button typu QtWidgets.QPushButton(self.centralwidget) - tworzy przycisk do startowania/zatrzymywania odtwarzania.
    - reset\_button typu QtWidgets.QPushButton(self.centralwidget) - tworzy przycisk do resetowania piosenki.
    - volume\_slider typu QtWidgets.QSlider(self.centralwidget) - tworzy suwak do regulowania głośności piosenki.
    - timeSlider typu QtWidgets.QSlider(self.centralwidget) - tworzy suwak do śledzenia i zmieniania obecnego momentu odtwarzanej piosenki.
    - playlist\_name typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zawierającą nazwę obecnie słuchanej playlisty.
    - current\_time typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zawierającą obecny czas odtwarzanej piosenki.
    - max\_time typu QtWidgets.QLabel(self.centralwidget) - tworzy etykietę zawierającą maksymalny czas trwania piosenki.
    - gridLayout typu QtWidgets.QGridLayout(self.centralwidget) - tworzy układ według którego rozmieszczone będe inne widgety w głównym oknie.
    - configFile typu string - zawiera nazwę pliku konfiguracyjnego.
    - curr\_playlist\_paths typu list of Song - zawiera listę obecnie wczytanych piosenek.
    - prev\_folders typu list of string - zawiera listę poprzednio wczytanych folderów.
    - saved\_playlists typu dict of Song - zawiera słownik zapisanych playlist gdzie kluczem jest nazwa playlisty a zawartość to obiekty typu Song z wczytanych w momencie zapisu piosenek.
    - color\_mode typu int - flaga określająca styl graficzny głównego okna.
    - playlist typu QMediaPlaylist() - tworzy pomocjniczą listę odtwarzania z samych ścieżek wczytanych plików.
    - player typu QMediaPlayer() - tworzy odtwarzacz muzyki.
    - self.actionwybierz\_piosenke typu QAction - tworzy akcje wybierającą piosenkę.
    - actiondodaj\_piosenke typu QAction - tworzy akcje dodającą piosenkę
    - actionwybierz\_folder typu QAction - tworzy akcje wybierającą folder piosenek.
    - actiondodaj\_folder typu QAction - tworzy akcje dodającą folder piosenek.
    - actionlista\_piosenek typu QAction - tworzy akcje wyświetlającą listę piosenek.
    - actionstworz\_playliste typu QAction - tworzy akcje tworzącą playlistę.
    - actionzmien\_nazwe\_playlisty typu QAction - tworzy akcje zmieniającą nazwę playlisty.
    - actiondodaj\_piosenke\_do\_play typu QAction - tworzy akcje dodającą obecnie słychaną piosenkę do playlisty.
    - actiondodaj\_piosenk\_do\_ulubionych typu QAction - tworzy akcje dodającą obecnie słychaną piosenkę do ulubionych.
    - actionpobierz\_informacje\_piosenek typu QAction - tworzy akcje pobierającą informacje obecnie słuchanej piosenki.
    - selectSavedS typu QAction - tworzy akcję pozwalającą wybrać jedną z zapisanych playlist i wczytać ją.
    - actionwy\_wietl\_tekst\_piosenk typu QAction - tworzy akcje pozawalającą wyświetlić tekst piosenki jeżeli został on pobrany.
    - actionhistoria\_folderow typu QAction - tworzy akcję pozwalającą na przeszukiwanie do 5 poprzednio wybranych folderów i wczytać jeden z nich.
    - actionWyszukaj\_piosenk typu QAction - tworzy akcję pozwalającą wyszukać i wczytać jedną z zapisanych piosenek.
    - randomizePlaylist typu QAction - tworzy akcje pozwalającą przesłuchanie piosenek wczytanych losowo.
    - show\_information typu QAction - tworzy akcje pokazującą informacje piosenki słuchanej.
    - color\_change typu QAction - tworzy akcje pozwalającą na zmianę kolorystyki GUI.

### Metody

* + - \_\_init\_\_(self) - konstruktor bezargumentowy deklarujący zmienne.
    - setupUi(self) - metoda bezargumentowa wywoływana na obiekcie klasy rozmieszczająca widgety w głównym oknie, nadająca im nazwy i określająca ich stan początkowy. Wywołuje inne metody tworzące GUI.
    - retranslateUi(self) - metoda bezargumentowa wywoływana na obiekcie klasy nadająca nazwy widgetom w oknie głównym.
    - connections(self) - metoda bezargumentowa wywoływana na obiekcie klasy tworząca połączenia między sygnałami nadawanymi przez stałe fragmenty interfejsu a metodami. Tworzy także przyciski w pasku opcji i narzędzi, nazywa je i łączy z odpowiednimi metodami.
    - create\_menus(self) - metoda bezargumentowa wywoływana na obiekcie klasy która przypisuje stworzone akcje do odpowiednich kontenerów.
    - dark\_mode(self) - metoda bezargumentowa wywoływana na obiekcie klasy która ustawia styl GUI na jeden z 2 możliwych zależnie od zmiennej color\_mode.
    - dynamicConnections(self) - metoda bezargumentowa wywoływana na obiekcie klasy która tworzy połączenia między sygnałami nadawanymi przez zmienne części aplikacji a metodami.
    - closeEvent(self, event) - metoda przeciążona która opisuje zachowanie aplikacji w przypadku nadania przez główne okno sygnału zamykającego.
    - volumeChange(self) - metoda bezargumentowa opisująca zachowanie aplikacji przy przesuwaniu suwaka głośności.
    - endOfPlaylist(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy playlista się skończy.
    - next(self) - metoda bezargumentowa opisująca zachowanie aplikacji przy naciśnięciu przycisku “następna”.
    - prev(self) - metoda bezargumentowa opisująca zachowanie aplikacji przy naciśnięciu przycisku “poprzednia”.
    - start(self) - metoda bezargumentowa opisująca zachowanie aplikacji przy naciśnięciu przycisku “start”/”stop”.
    - reset(self) - metoda bezargumentowa opisująca zachowanie aplikacji przy naciśnięciu przycisku “restart”.
    - is\_saved(self, path) - metoda z argumentem path typu string wywoływana na obiekcie klasy która sprawdza czy podana ścieżka jest taka sama jak ścieżka zapisanej już piosenki i zwracająca krotke typów (bool, Song/None) zależnie od wyniku wyszukiwania.
    - select\_song(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja wybrania piosenki.
    - select\_folder(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja wybrania folderu.
    - add\_song(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja dodania piosenki.
    - add\_folder(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja dodania folderu.
    - media\_setter(self) - metoda bezargumentowa wywoływana na obiekcie klasy odpowiadająca za deklaracje zmiennych player i playlist.
    - random\_mode(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "graj piosenki losowo"
    - setting\_player(self) - metoda bezargumentowa wywoływana na obiekcie klasy odpowiadająca za przygotowywanie playera do odtwarzania wybranych piosenek.
    - select\_song\_common(self, song) - metoda z argumentum song typu string zawierającym ścieżkę pliku mp3 wywoływana na obiekcie klasy, jest to metoda pomocnicza wydzielająca wspólne części metod add\_song i select\_song.

* + - select\_folder\_popup(self) - metoda bezargumentowa wywoływana na obiekcie klasy, jest to metoda pomocnicza zawierająca proces wybierania folderu i zwracająca krotkę (string, list of string) zawierającą ścieżkę wybranego folderu oraz listę plików w nim zawartych.
    - select\_folder\_common(self, onlyfiles, response) - metoda z argumentami onlyfiles typu list of string zawierającym ścieżki plików w wybranym folderze oraz response typu string zawierającym ścieżkę wybranego folderu wywoływana na obiekcie klasy, jest to metoda pomocnicza wydzielająca wspólne części metod add\_folder i select\_folder.
    - add\_to\_fav(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "dodaj piosenkę do ulubionych".
    - create\_playlist(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "Stwórz playlistę".
    - change\_playlist\_name(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "dodaj piosenkę do playlisty".
    - add\_song\_to\_playlist(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "dodaj piosenkę do ulubionych".
    - select\_playlist(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "Przeglądaj zapisane playlisty"
    - song\_list(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "lista piosenek".
    - zamknij(self, t\_window, s\_list=None) - metoda pomocnicza z argumentami t\_window typu QDialog oraz s\_list typu QListWidget opisująca zachowanie okna z listą piosenek gdy zostanie wybrany przycisk “zapisz”, “anuluj” lub “X”.
    - serch\_song(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "Wyszukaj piosenkę".
    - history(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "historia folderów".
    - add\_folder\_histoty(self, folder\_path) - metoda pomocnicza z argumentem folder\_path typu string która zapisuje podaną ścieżkę folderu do zmiennej prev\_folders
    - download\_info(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "pobierz informacje piosenki".
    - save\_info(self, metadata) - metoda pomocnicza przyjmująca argument metadata typu tuple of string zawierający pobrane informacje piosenki. Metoda ta przypisuje pobrane informacje odpowiednim zmiennym
    - save\_song\_text(self, text) - metoda pomocnicza z argumentem text typu string która zapisuje pobrany tekst piosenki do pliku typu .txt.
    - show\_song\_text(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "wyświetl tekst piosenki".
    - show\_song\_info(self) - metoda bezargumentowa opisująca zachowanie aplikacji gdy zostanie wybrana opcja "Wyswietl informacje".
    - update\_position(self, position) - metoda z argumentem position typu int która przesuwa suwak długości piosenki podczas jej trwania oraz przesuwa moment w którym odtwarzana jest piosenka gdy użytkownik przesunie ten sam suwak oraz stale utrzymuje zmienna current\_time aktualna.
    - update\_duration(self, duration) - metoda z argumentem duration typu int która przy zmianie piosenki dostosowywuje wartości timeSlider i etykiety max\_time z całkowitym czasem trwania piosenki.

## Metody globalne

* load\_saved\_songs(conFile) - metoda z argumentem conFile typu string zawierającym ścieżkę pliku konfiguracyjnego, która wczytuje zapisane w podanym pliku dane tworzy słownik typu Song na podstawie tych danych i zwraca ten słownik.
* save\_playlists(saveFile, playlists) - metoda z argumentami saveFile typu string zawierający ścieżke pliku do którego metoda będzie zapisywać dane oraz playlists typu dict of Song zawierającą piosenki zapisane w playlistach. Metoda zapisuje przekazane dane w podanym pliku.
* hhmmss(ms) - metoda z argumentem ms typu int zawierającym długość piosenki w milisekundach. Metoda zwraca czas w formacie ) "%d:%02d:%02d" lub "%d:%02d" zależnie od długości.
* get\_metadata(song\_path) - metoda z argumentem song\_path typu string zawierającą ścieżkę pliku .mp3 i zwracającą krotkę zawierającą domyślne lub, jeżeli udała się operacja, pobrane tytuł, autor, album, rok i tekst piosenki.

# Problemy

1. ShazamAPI zwraca informacje w cyrylicy
2. ShazamAPI uruchamia się z linii komend ale nie z IDE
3. ShazamAPI brakuje ffmpeg
4. mimo że plik mp3 jest wczytywany i odtwarzany, muzyka nie gra.
5. pyinstaller tworzy niedziałający plik .exe
6. plik exe na innych komputerach nie odtwarza wszystkich plików mp3
7. plik exe na innych komputerach nie pobiera informacji o piosence

# Rozwiązania

1. Zmiana adresu url na który wysyłane są zapytania
2. Instalacja pythonowskiego opakowania do ffmpeg do wirtualnego środowiska
3. instalacja ffmpeg w systemie windows
4. instalacja pluginów lav filters do systemu windows
5. pobieranie brakujących pakietów i odinstalowanie tych powodujących kolizje typów
6. instalacja lav filters na innym komputerze
7. instalacja ffmpeg na innym komputerze.

# Zgodność z projektem

Realizacja będzie w dużym stopniu zgodna z projektem.