

# AKADEMIA NAUK STOSOWANYCH W NOWYM SĄCZU

Wydział Nauk Inżynieryjnych  
Katedra Informatyki

## DOKUMENTACJA PROJEKTOWA PROGRAMOWANIE URZĄDZEŃ MOBILNYCH

### **Raptor**

Autor:  
Mateusz Stanek  
Dawid Szołdra  
Filip Wachała

Prowadzący:  
mgr inż. Dawid Kotlarski

Nowy Sącz 2024

## Spis treści

<b>1. Ogólne określenie wymagań projektu</b>	<b>3</b>
1.1. Ogólny zarys wymagań . . . . .	3
1.2. Wykorzystane czujniki . . . . .	3
<b>2. Określenie wymagań szczegółowych</b>	<b>4</b>
2.1. Ogólny zarys narzędzi użytych w projekcie . . . . .	4
2.2. Wykorzystanie czujników . . . . .	4
2.3. Zarys interfejsu . . . . .	5
2.4. Zachowanie w niepożądanych sytuacjach . . . . .	7
2.5. Dalszy rozwój . . . . .	7
<b>3. Projektowanie</b>	<b>8</b>
<b>4. Implementacja</b>	<b>9</b>
<b>5. Testowanie</b>	<b>10</b>
<b>6. Podręcznik użytkownika</b>	<b>11</b>
<b>Literatura</b>	<b>12</b>
<b>Spis rysunków</b>	<b>12</b>
<b>Spis tabel</b>	<b>13</b>
<b>Spis listingów</b>	<b>14</b>

# 1. Ogólne określenie wymagań projektu

## 1.1. Ogólny zarys wymagań

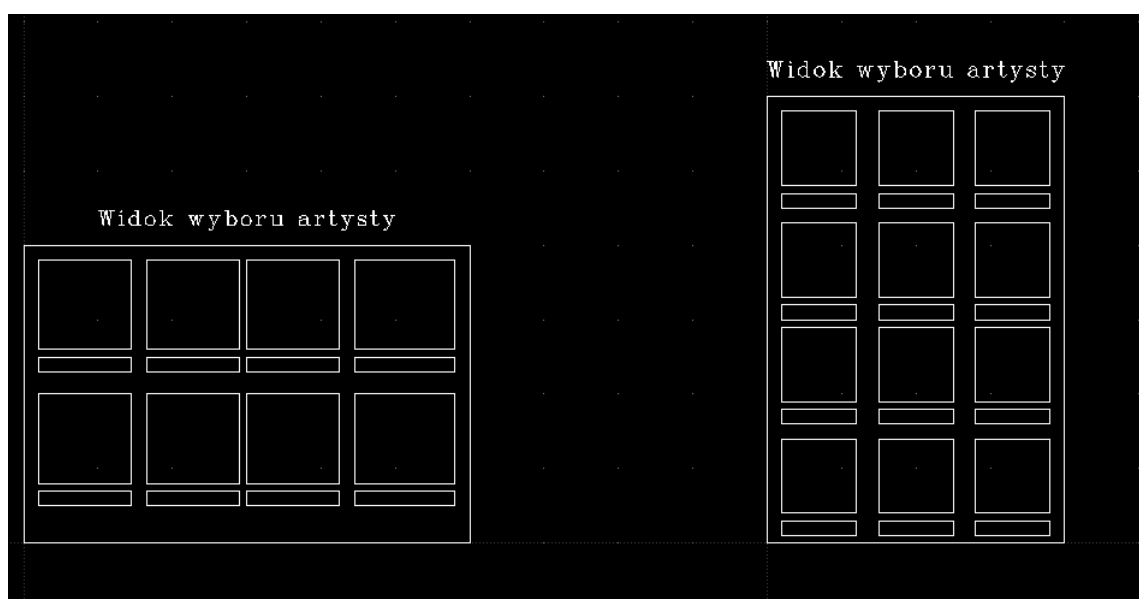
Celem programu jest pełnienie funkcji odtwarzacza muzyki oraz dodatkowo ma on pełnić rolę dyktafonu. Program będzie mógł skanować dany folder, a w nim tagi zawartych plików muzycznych i tworzyć na jego podstawie graficzną reprezentację biblioteki.

## 1.2. Wykorzystane czujniki

Program ma na celu wykorzystanie trzech czujników, z którymi użytkownik będzie wchodził w interakcję. Zostaną użyte następujące:

- Żyroskop - Interfejs programu będzie się zmieniał w zależności od orientacji urządzenia.
- Mikrofon - Program będzie posiadał funkcję nagrywania dźwięku. Nagrane pliki będzie można odtwarzać w odtwarzaczu
- Czujnik światła - Interfejs programu będzie mógł zmieniać swoje kolory w zależności od wykrytego poziomu światła na czujniku

## 1.3. Zarys interfejsu



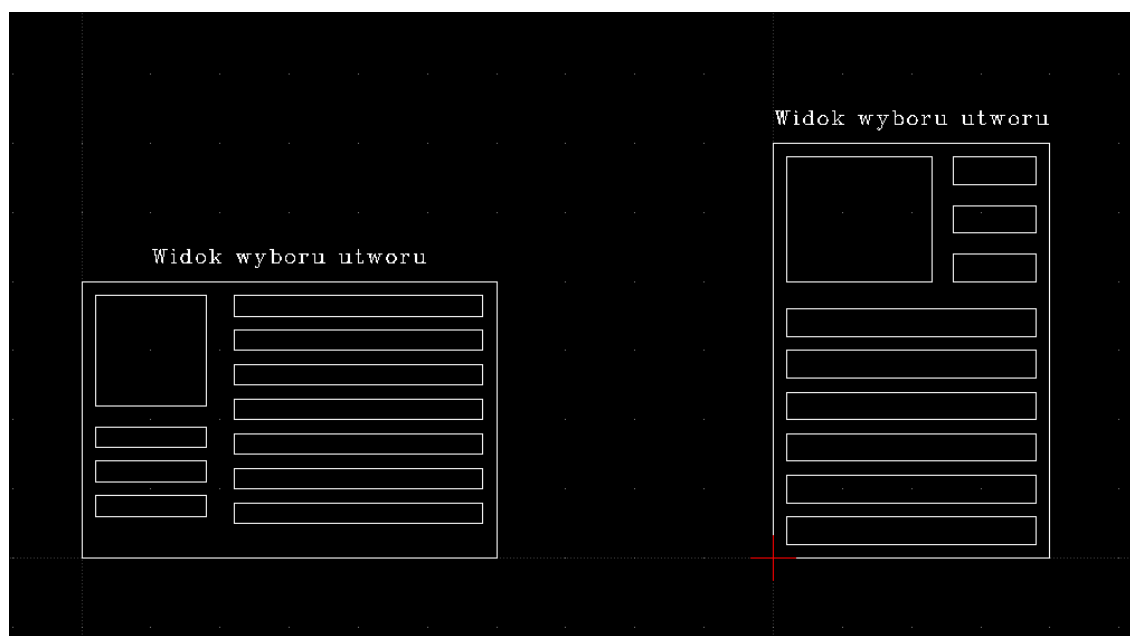
**Rys. 1.1.** Mockup widoku biblioteki - listing wykonawców

Widok wykonawców będzie ekranem startowym aplikacji. "Kafelki" będą zdjęciami wykonawców. Klikanie na jeden z nich przejdzie do widoku albumów danego wykonawcy



**Rys. 1.2.** Mockup widoku albumów danego wykonawcy

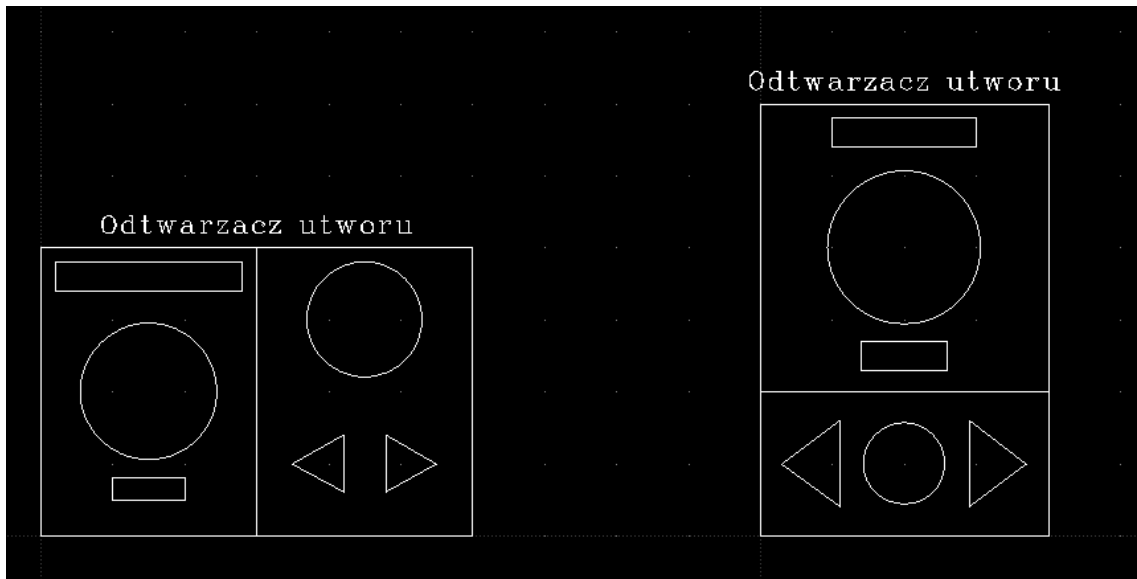
Widok albumów jest identyczny jak widok wykonawców. Jedyną różnicą polega tym, że zdjęcia na kafelkach będą zdjęciami albumów.



**Rys. 1.3.** Mockup widoku wyboru utworu

Po wejściu na jakiś album zaprezentowane zostaną zawarte w nim utwory. W lewym górnym jest zdjęcie danego albumu, a obok niego jest kilka informacji o

albumie jak wykonawca, data, tytuł. Dłuższe paski to lista tytułów piosenek, które można kliknąć, aby daną piosenkę włączyć.



**Rys. 1.4.** Mockup odtwarzacza

Odtwarzacz będzie działał następująco: duże koło będzie stylizowane na płytę, gdzie wypełniona ona będzie obrazem albumu. Płyta ta będzie się kręcić w czasie gdy gra piosenka. Kąt płyty (od  $0^\circ$ , do  $360^\circ$ ) będzie określał jak duża część piosenki została odtworzona. Kąt ten będzie określony jeszcze niezdefiniowanym efektem graficznym. Prostokąty wokół płyty to tytuł piosenki, a na dole czas grania. Kółko i wokół niego trójkąty to przyciski odtwarzania - graj/pauza, następny, poprzedni.



**Rys. 1.5.** Mockup dyktafonu

Do dyktafonu będzie można się dostać przesuając palcem w prawo na ekranie

wykonawców. Dyktafon jest aktywowany wielkim okrągłym przyciskiem. Te kreski obok niego to wizualizacja dźwięku z mikrofonu. Menu w rogu będzie pozwalało na m.in. skonfigurowanie folderu zapisu nagrań.

## 2. Określenie wymagań szczegółowych

### 2.1. Ogólny zarys narzędzi użytych w projekcie

Aplikacja jest zaprojektowana w Android Studio w języku Kotlin. Całe UI aplikacji będzie zbudowane na podstawie Frameworka Jetpack Compose<sup>1</sup>. Używając wbudowanych bibliotek w SDK Androida, będzie mogła odczytywać pliki ze wskazanego folderu. Odczytywanie tagów z plików odbędzie się za pomocą biblioteki Taglib<sup>2</sup>, która posiada nieoficjalne bindingi do Kotlin. Wszelki processing audio np. na potrzeby wizualizacji może zostać wykonany za pomocą SDK i wbudowanego modułu AudioProcessor<sup>3</sup>. Odtwarzaniem pliku będzie się zajmował moduł MediaPlayer<sup>4</sup>.

### 2.2. Wykorzystanie czujników

- Żyroskop - Z racji, że każdy element interfejsu w Jetpack jest generowany kodem, można, przynajmniej na początku, ustawić każdą wersję interfejsu jako osobną funkcję. Następnie, w zależności od wykrytej orientacji, przy użyciu API sensorów<sup>5</sup>, można wywoływać odpowiednią funkcję.
- Mikrofon - Funkcja dyktafonu najprawdopodobniej będzie całkiem oddzielnym Activity. Funkcjonalność ta, z natury, jest dosyć oddzielna od reszty aplikacji. Nagrania dyktafonem powinny być zapisywane do osobnego folderu. Można by zintegrować nagrania z resztą aplikacji jako osobnego wykonawcę w widoku biblioteki. Mikrofon będzie nagrywany poprzez moduł MediaRecorder<sup>6</sup>
- Czujnik światła - Android Studio oferuje możliwość definiowania własnych klas zajmujących się kolorystyką. Oznacza to że można używać różnych obiektów w zależności od warunków. Wykrywanie światła będzie się odbywało używając API sensorów<sup>7</sup>

### 2.3. Zachowanie w niepożądanych sytuacjach

Głównym wyjątkiem, na który może napotkać się aplikacja jest błąd odczytu albo plików, albo tagów z pliku. Kotlin, na szczęście, pozwala na łatwe sprawdzanie war-

---

<sup>1</sup><https://developer.android.com/compose>

<sup>2</sup><https://github.com/timusus/KTagLib>

<sup>3</sup><https://developer.android.com/reference/androidx/media3/common/audio/AudioProcessor>

<sup>4</sup><https://developer.android.com/media/platform/mediaplayer>

<sup>5</sup>[https://developer.android.com/develop/sensors-and-location/sensors/sensors\\_overview](https://developer.android.com/develop/sensors-and-location/sensors/sensors_overview)

<sup>6</sup><https://developer.android.com/media/platform/mediarecorder>

<sup>7</sup>Patrz, przypis 5

tości null danych zmiennych operatorem '?. W odpowiednich fragmentach kodu dotyczących ładowania plików, będzie sprawdzana poprawność danych i najprawdopodobniej pojawi się pop-up po stronie użytkownika, że wystąpił błąd, a po stronie dewelopera błąd zostanie logowany.

## 2.4. Dalszy rozwój

Jeżeli praca nad aplikacją będzie się odbywała w przyszłości, należy skupić uwagę na lepszym zarządzaniu biblioteką (auto tagowanie, pobieranie miniatur z internetu, itp.). Ponadto, należy szukać błędów, które nadal zostały w aplikacji.



### **3. Projektowanie**

## 4. Implementacja

## **5. Testowanie**

## 6. Podręcznik użytkownika

## Spis rysunków

2.1. Mockup widoku biblioteki - listing wykonawców . . . . .	5
2.2. Mockup widoku albumów danego wykonawcy . . . . .	5
2.3. Mockup widoku wyboru utworu . . . . .	6
2.4. Mockup odtwarzacza . . . . .	6
2.5. Mockup dyktafonu . . . . .	7

## **Spis tabel**

## **Spis listingów**