****

**Programowanie obiektowe**

*Tytuł projektu*

Prowadzący: Autor:

pracownik UR *Imię Nazwisko*

nr albumu:

Kierunek: Wybierz element., grupa Wybierz element.

Rzeszów 20XX

Spis treści

[**1.** **Opis założeń projektu** 3](#_Toc128654617)

[**2.** **Specyfikacja wymagań** 3](#_Toc128654618)

[**3.** **Diagram przypadków użycia** 4](#_Toc128654619)

[**4.** **Harmonogram realizacji projektu (diagram Gantta)** 4](#_Toc128654620)

[**5.** **Opis techniczny projektu** 5](#_Toc128654621)

[**6.** **Prezentacja warstwy użytkowej projektu** 6](#_Toc128654622)

[**7.** **System kontroli wersji** 7](#_Toc128654623)

[**8.** **Literatura** 8](#_Toc128654624)

1. **Opis założeń projektu**

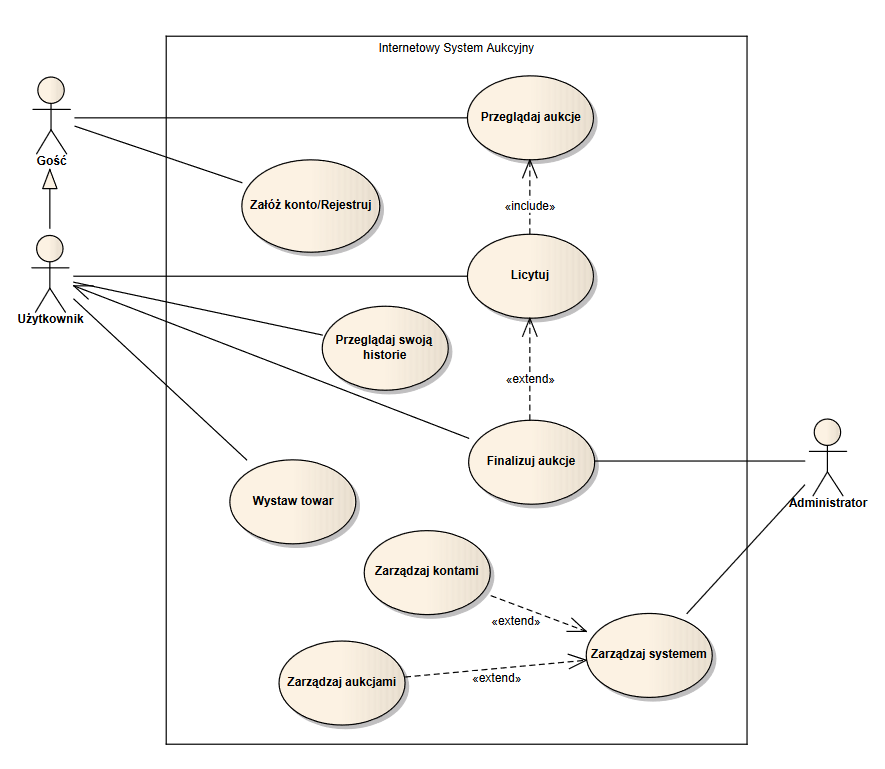
Należy podać założenia do projektu opis. UWAGA!!! Tekst piszemy bezosobowo np. zaprojektowano, stworzono, umieszczono itp.…

Pracownik linii lotniczych może dodawać do harmonogramu nowe loty. Każdy lot jest reprezentowany przez następujące dane: numer lotu, numer samolotu, lotnisko początkowe, lotnisko docelowe, data i przewidywany czas lotu. Klient może zarejestrować się na stronie podając swoje dane osobowe a następnie zakupić bilet lotniczy, wyszukując żądany lot w wyszukiwarce i wybierając dostępną taryfę. Klient przed zakupem będzie widział, jakie są koszty biletów i będzie mógł zakupić ich kilka. Pracownik linii lotniczych może dodatkowo modyfikować i usuwać dane o lotach i zakupionych biletach, oraz dane zarejestrowanych kont. Klient może przeglądać dostępne loty za pośrednictwem strony internetowej lub bezpośrednio za pomocą specjalnego programu. Zakłada się, że klientów jednocześnie przeglądających loty, oraz dokonujących zakupu biletów lotniczych może być około 100. Linie lotnicze mogą oferować kilkanaście różnych tras. Linie lotnicze oferują swoje usługi lotnicze w kilku lotniskach na świecie. Zaleca się język Python + framework Django.

1. **Specyfikacja wymagań**
   1. **Wymagania funkcjonalne**

* Linie lotnicze oferują loty klientom, którzy są zarejestrowani.
* Linie lotnicze dodają nowe oferty lotów, oraz usuwają loty, które już się odbyły.
* Pracownik linii korzysta z systemu i ma dostęp do harmonogramu lotów, oraz danych klienta.
* Klient może zakupić bilet, który jest dostępny na stronie po wyszukaniu. Zakupionego biletu nie można zwrócić za pomocą strony.
* Pracownik może dodawać nowe loty, edytować je i usuwać. Może także zarządzać danymi użytkownika.
  1. **Wymagania niefunkcjonalne**
* Możliwość dodawania, usuwania, oraz edycji rekordów w bazie z poziomu pracownika.
* Aplikacja jest przyjazna dla klienta i jego rodziny, oraz jest prosta w użyciu.
* Aplikacja działa na systemach, które posiadają przeglądarkę internetową.
* Aplikacja tworzona jest w języku Python z frameworkiem Django.
* Aplikacja nawiązuje połączenie z bazą danych i używa rekordów w niej zapisanych.

1. **Diagram przypadków użycia**

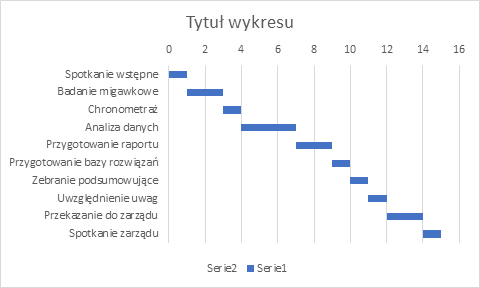


Rysunek 1. Diagram przypadków użycia

Szczegółowe informacje odnośnie diagramu przypadków można znaleźć pod [adresem](https://brasil.cel.agh.edu.pl/~09sbfraczek/diagram-przypadkow-uzycia%2C1%2C9.html).

1. **Harmonogram realizacji projektu (diagram Gantta)**

Poniżej zamieszono harmonogram realizacji projektu.



Rysunek 2. Diagram Gantta

1. **Opis techniczny projektu**

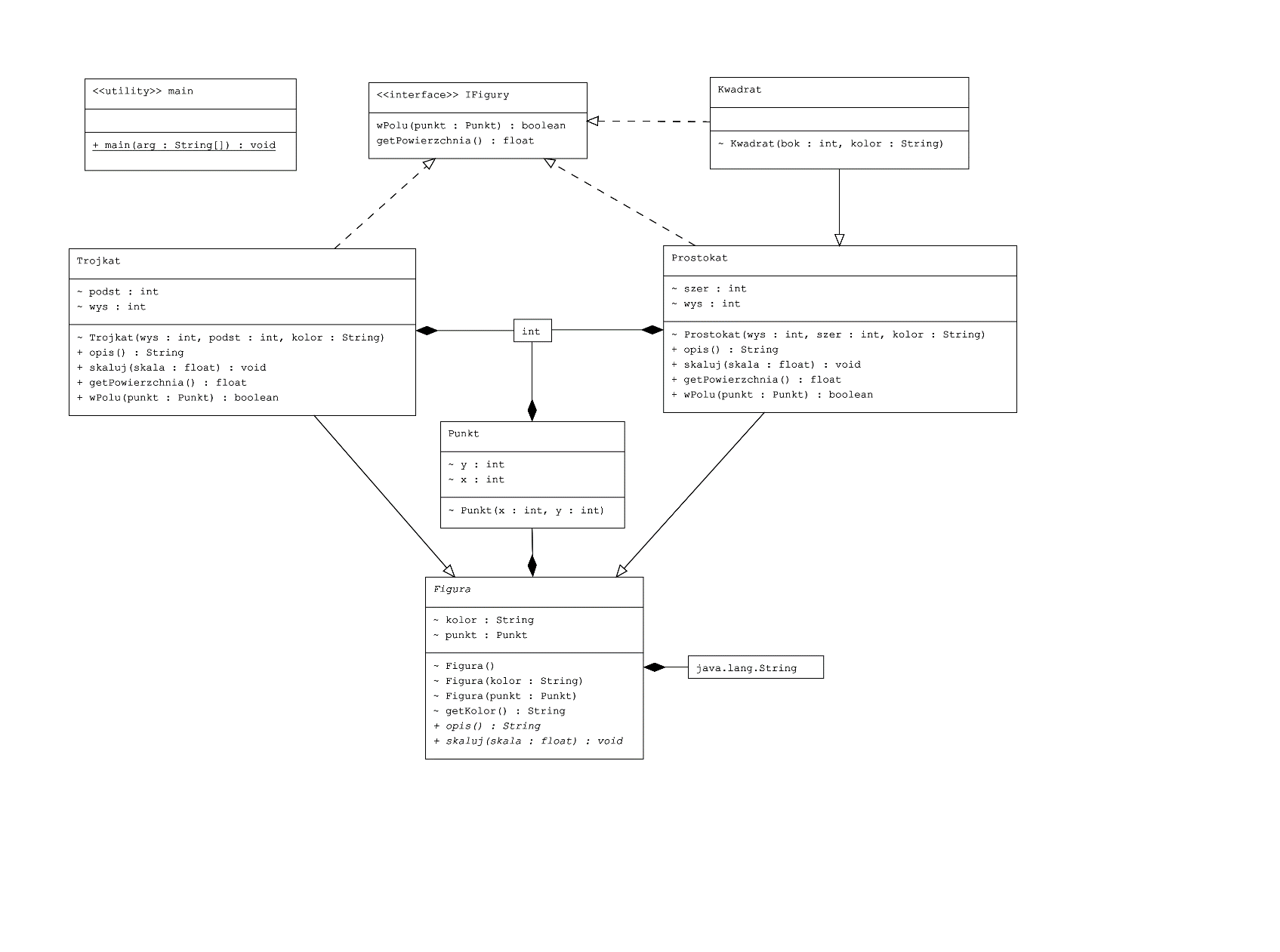
Należy umieścić informacje odnośnie wykorzystywanego języka, narzędzi oraz minimalnych wymagań sprzętowych.

Np.:

* Środowisko programistyczne Javy: Java JDK Kit 7u79.
* Środowisko programistyczne: Android Studio 2.1.2.
* Urządzenie z system operacyjnym Android 4.4 KitKat – może być emulator.
* Ze względów na liczbę szczegółów aplikacja była projektowana na urządzenia typu tablet – w Samsung Tab Pro 10,1’.

Należy przedstawić diagram klas projektowanej aplikacji (diagram można wygenerować po zrealizowaniu projektu) oraz opisać strukturę przechowywanych danych w pliku/bazie.

Przykładowy diagram klas:



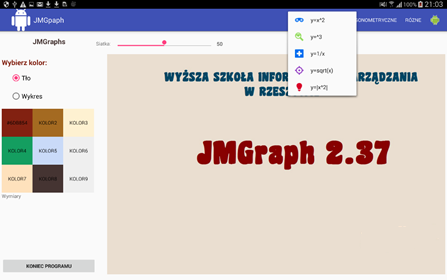
Rysunek 3. Diagram klas projektowanej aplikacji

1. **Prezentacja warstwy użytkowej projektu**

Należy przedstawić opis warstwy użytkowej projektu w tym celu należy umieścić opis aplikacji oraz PrtSc o których jest mowa. Np.:

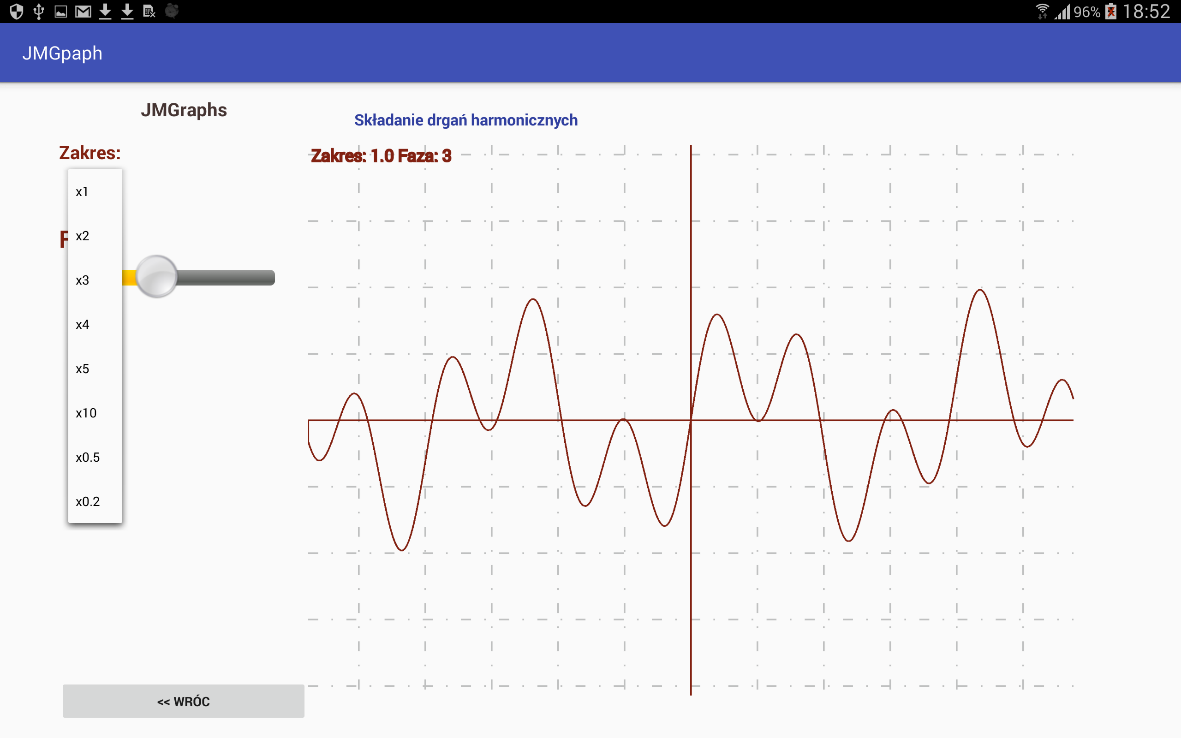
Na rysunku 1 przedstawiono główne okno aplikacji. Użytkownik po uruchomieniu aplikacji otrzymuje ekran startowy na którym może:

* Ustawić kolor tła i linii wykresu – domyślnie kolor tła jest czarny, a wykresu biały.
* Zobaczyć wymiary grafiki i obszaru rysowania.
* Z menu górnego wybiera grupy wykresów:
  + Proste wykresy: y=x2, y=x3, y=1/x, y=√(x,) y=|x2|.
  + Funkcje trygonometryczne: y=sin(x), y=cos(x), y=th(x), y=ctg(x).
  + Różne: 2 fraktale: Smok Heighwaya, Krzywe Kocha oraz superpozycja funkcji harmonicznych.
* Ostatnia pozycja z ikoną Androida wyświetla informację o autorze.
* Dodatkowo poniżej menu można ustawić skalę siatki wykresu.



Rysunek 4. Główne okno aplikacji

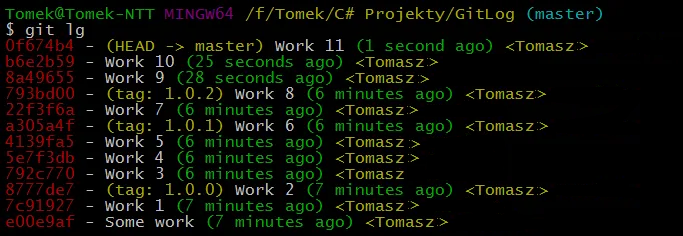
Krzywe Kocha oraz krzywe harmoniczne wyświetlane są w nowych aktyvity z własnym layoutem (patrz Rysunek 2). Generując krzywe Kocha możemy wybrać ich stopień: od 0 – 6. Ponieważ są to funkcje rekurencyjne generowanie wyższych stopni trwa zbyt długo. Tworząc funkcje harmoniczne możemy ustalić zakres wykresu – wielokrotność 3600 oraz różnicę faz między funkcjami – w płynnej skali względnej 0 – 10.

****

Rysunek 5. Przykład wygenerowanego wykresu dla zadanych parametrów

1. **System kontroli wersji**

Projekt realizowany był z wykorzystaniem systemu kontroli wersji Git, wszystkie pliki źródłowe projektu znajdują się pod adres: <https://github.com/> i będą dostępne do 31.01.2024. Na rysunku 6 przedstawiono zrzut ekranu pokazujący historię kosmitów.



Rysunek 6. Historia komitów

1. **Literatura**

Należy podać literaturę książki/podręczniki/artykuły lub link do stron www.

1. Jacek Matulewski, Visual Studio 2017. Tworzenie aplikacji Windows w języku C#, 2018, Wyd. Helion.
2. Kurs WPF [www.przykaldowastrona.pl](http://www.przykaldowastrona.pl) (data dostępu: 05.06.2021).