****

**Praca projektowa programowanie obiektowe**

*System rezerwacji sal/podział godzin*

Prowadzący: Autor:

mgr inż. Ewa Żesławska *Gabriel Hołda*

nr albumu: 131441

Kierunek: Informatyka. Grupa Lab 3.Click here to enter text.

Rzeszów 2024

Spis treści

[**1.** **Opis założeń projektu** 3](#_Toc166484408)

[**2.** **Opis struktury projektu** 7](#_Toc166484409)

[**3.** **Harmonogram realizacji projektu** 8](#_Toc166484410)

[**4.** **Prezentacja warstwy użytkowej projektu** 9](#_Toc166484411)

[**5.** **Podsumowanie** 11](#_Toc166484412)

[**6.** **Literatura** 12](#_Toc166484413)

1. **Opis założeń projektu**

**Cel projektu**

Celem projektu jest stworzenie systemu rezerwacji sal i podziału godzin zajęć dla różnych kierunków studiów oraz różnych form zajęć. System ma umożliwiać efektywne zarządzanie harmonogramem zajęć, eliminować konflikty rezerwacji sal i zajęć oraz oferować intuicyjny interfejs użytkownika.

**Problem do rozwiązania**

Głównym problemem jest trudność w zarządzaniu harmonogramem zajęć na uczelni, co prowadzi do częstych konfliktów rezerwacji sal i nakładania się zajęć dla tych samych grup. Problemy te wynikają z braku centralnego systemu do zarządzania harmonogramem oraz z ręcznego zarządzania podziałem godzin.

**Ważność problemu**

Problem jest istotny, ponieważ konflikty rezerwacji sal i zajęć mogą prowadzić do dezorganizacji procesu dydaktycznego, frustracji studentów i wykładowców oraz marnowania zasobów uczelni. Dowodem na istnienie problemu są liczne skargi studentów i wykładowców na brak dostępnych sal oraz nakładające się zajęcia.

**Niezbędne zasoby do rozwiązania problemu**

Aby rozwiązać problem, niezbędne są:

* Centralny system zarządzania harmonogramem zajęć.
* Intuicyjny interfejs użytkownika do zarządzania rezerwacjami sal i zajęciami.
* Mechanizmy zapobiegania konfliktom rezerwacji.

Sposób rozwiązania problemu

Problem zostanie rozwiązany poprzez stworzenie aplikacji "System rezerwacji sal/podział godzin", która będzie umożliwiać:

1. Dodawanie, edytowanie i usuwanie zajęć.
2. Wyświetlanie harmonogramu zajęć dla wybranego dnia, grupy lub sali.
3. Sprawdzanie dostępności sali i grupy przed dodaniem zajęć.

**Kroki realizacji projektu:**

1. Analiza wymagań.
2. Projektowanie struktury bazy danych i interfejsu użytkownika.
3. Implementacja systemu zarządzania harmonogramem.
4. Testowanie aplikacji.
5. Wdrożenie i dokumentacja końcowa.

Wynikiem prac będzie aplikacja do zarządzania rezerwacjami sal i podziałem godzin zajęć.

**Wymagania funkcjonalne i niefunkcjonalne:**

Wymagania funkcjonalne

1. Dodawanie zajęć do harmonogramu.

* Użytkownik może dodać nowe zajęcia, podając kierunek, nazwę przedmiotu, nazwisko prowadzącego, numer sali, dzień tygodnia i godzinę zajęć.

1. Usuwanie zajęć z harmonogramu.

* Użytkownik może usunąć istniejące zajęcia.

1. Edytowanie szczegółów zajęć.

* Użytkownik może zmienić szczegóły już dodanych zajęć.

1. Wyświetlanie harmonogramu dla wybranego dnia.

* Użytkownik może przeglądać harmonogram zajęć na wybrany dzień.

1. Wyświetlanie harmonogramu dla wybranej grupy.

* Użytkownik może przeglądać harmonogram zajęć dla wybranej grupy.

1. Wyświetlanie harmonogramu dla wybranej sali.

* Użytkownik może przeglądać harmonogram zajęć dla wybranej sali.

1. Sprawdzanie dostępności sali i grupy przed dodaniem zajęć.

* System sprawdza, czy wybrana sala i grupa są dostępne w podanym czasie przed dodaniem zajęć do harmonogramu.

Wymagania niefunkcjonalne

1. Intuicyjny interfejs użytkownika.

* Aplikacja powinna być łatwa w obsłudze i zrozumiała dla użytkowników.

1. Szybkość działania aplikacji.

* Aplikacja powinna działać płynnie i szybko reagować na działania użytkownika.

1. Niezawodność i stabilność systemu.

* System powinien być stabilny i niezawodny, minimalizując ryzyko błędów i awarii.

1. Skalowalność.

* Aplikacja powinna być zaprojektowana tak, aby łatwo można było dodawać nowe kierunki i grupy bez dużych zmian w kodzie.

1. Bezpieczeństwo danych.

* Dane w systemie powinny być zabezpieczone przed nieautoryzowanym dostępem i utratą.

Przykłady wymagań funkcjonalnych:

* Dodawanie zajęć
* Usuwanie zajęć
* Edytowanie zajęć
* Wyświetlanie harmonogramu dla dnia, grupy, sali
* Sprawdzanie dostępności sali i grupy

Przykłady wymagań niefunkcjonalnych:

* Intuicyjność interfejsu
* Szybkość działania
* Niezawodność systemu
* Skalowalność
* Bezpieczeństwo danych

1. **Opis struktury projektu**

**Struktura projektu**

Projekt "System rezerwacji sal/podział godzin" składa się z kilku kluczowych komponentów i klas, które umożliwiają zarządzanie harmonogramem zajęć. Poniżej przedstawiono zaprojektowaną strukturę oraz jej opis techniczny.

**Wykorzystywany język i narzędzia**

Język programowania: Java

Framework do GUI: JavaFX

Narzędzie do budowy projektu: Apache Maven

IDE: IntelliJ IDEA

System kontroli wersji: Git

**Minimalne wymagania sprzętowe**

Procesor: Intel Core i3 lub odpowiednik

Pamięć RAM: 4 GB

Dysk twardy: 500 MB wolnego miejsca

System operacyjny: Windows, macOS, Linux

Java Development Kit (JDK): Wersja 11 lub nowsza

**Zarządzanie danymi i baza danych**

Dane dotyczące harmonogramu zajęć są przechowywane w pliku tekstowym w formacie JSON. Struktura pliku JSON pozwala na łatwe przechowywanie i przetwarzanie informacji o zajęciach, takich jak nazwa przedmiotu, nazwisko prowadzącego, numer sali, dzień tygodnia i godzina zajęć. Aplikacja zapewnia mechanizmy odczytu i zapisu danych do tego pliku.

**Hierarchia klas**

Poniżej przedstawiono hierarchię klas wraz z krótkim opisem najważniejszych metod:

MainApp

Główna klasa uruchamiająca aplikację.

Metoda start(Stage primaryStage) inicjuje główne okno aplikacji.

Zajecia

Klasa bazowa reprezentująca ogólne zajęcia.

Atrybuty: kierunek, nazwaPrzedmiotu, prowadzacy, numerSali, dzienTygodnia, godzina

Metody: czyDlaGrupy(int nrGrupy) - sprawdza, czy zajęcia są dla określonej grupy.

Laboratorium (dziedziczy po Zajecia)

Atrybut dodatkowy: numerGrupy

Metody: Przeciążenie czyDlaGrupy(int nrGrupy).

Wykład (dziedziczy po Zajecia)

Metody: Przeciążenie czyDlaGrupy(int nrGrupy) - zawsze zwraca true.

Projekt (dziedziczy po Zajecia)

Atrybuty dodatkowe: numerGrupy1, numerGrupy2

Metody: Przeciążenie czyDlaGrupy(int nrGrupy).

Plan

Klasa zarządzająca harmonogramem zajęć.

Atrybuty: listaZajec

Metody: dodajZajecia(Zajecia zajecia), usunZajecia(Zajecia zajecia), wypiszPodzialDnia(String dzien), wypiszPodzialGrupy(int nrGrupy), wypiszPodzialSali(int nrSali)

PlikManager

Klasa odpowiedzialna za operacje na plikach.

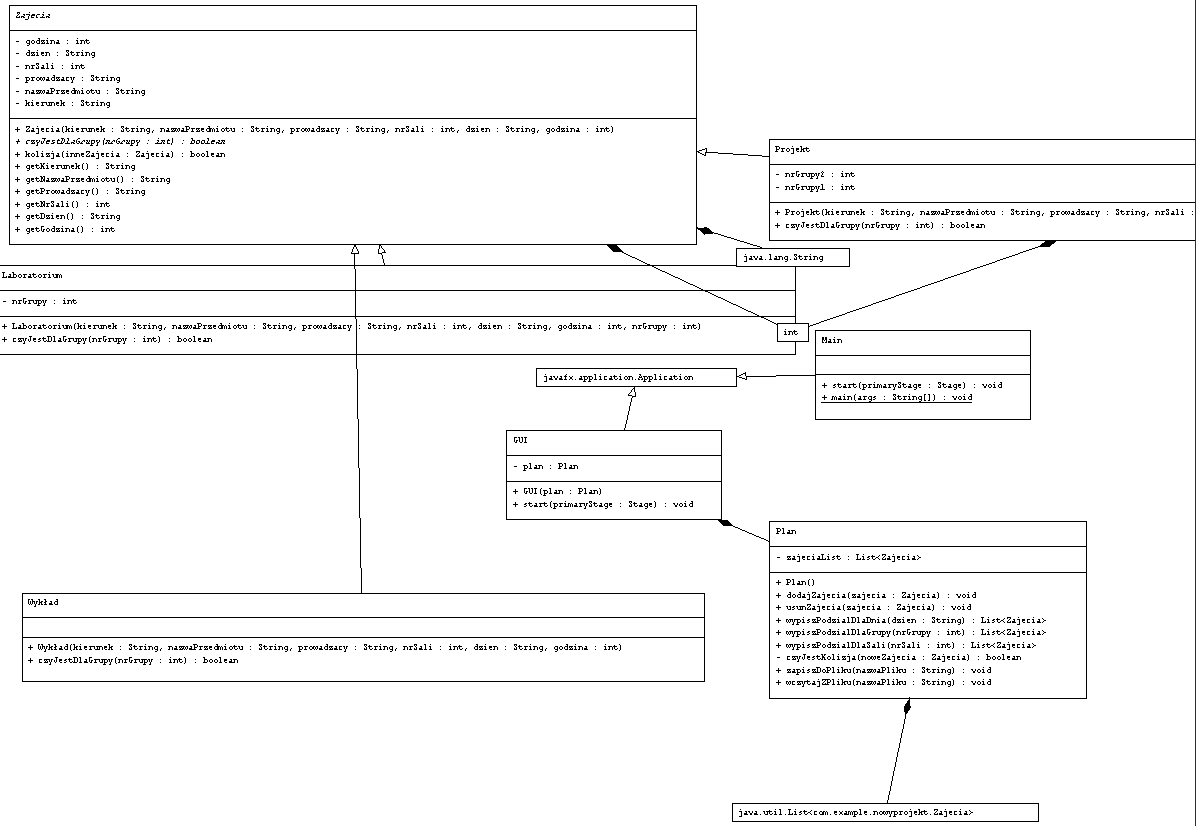
Metody: zapiszDoPliku(List<Zajecia> zajecia, String plik), wczytajZPliku(String plik)

GUIController

Klasa odpowiedzialna za interakcje z interfejsem użytkownika.

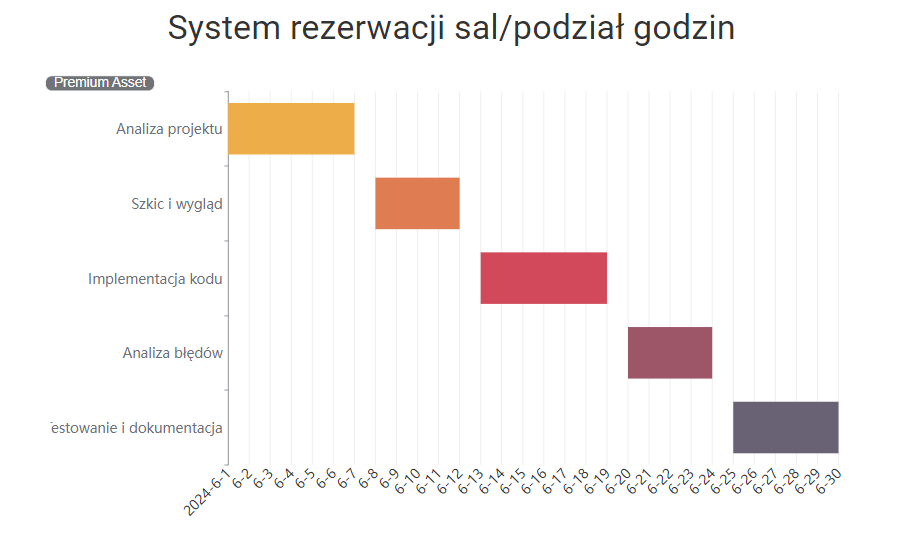
Metody: dodajZajecia, usunZajecia, pokazPodzialDnia, pokazPodzialGrupy, pokazPodzialSali

Diagram klas:



Rysunek 1. Diagram klas zaprojektowanej aplikacji

1. **Harmonogram realizacji projektu**



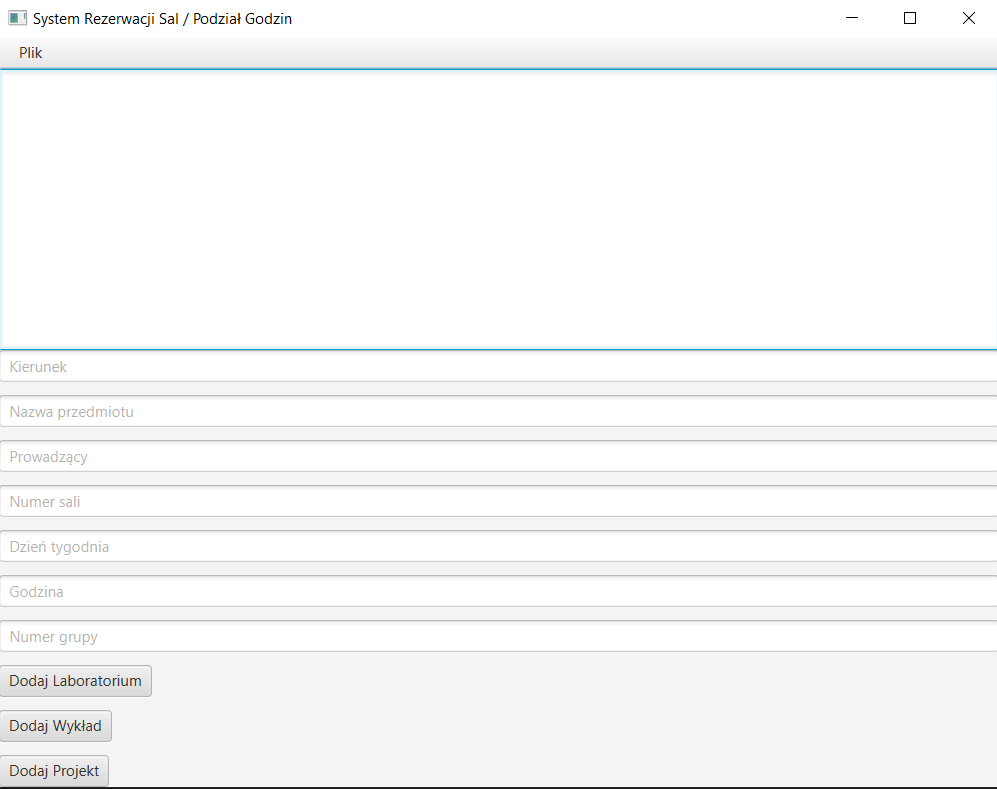
Rysunek 2. Diagram Gantta

Projekt realizowany był z wykorzystaniem systemu kontroli wersji Git, wszystkie pliki źródłowe projektu znajdują się pod adres: https://github.com/GAJAHO/zadania-java-studia i będą dostępne do 30.06.2025.

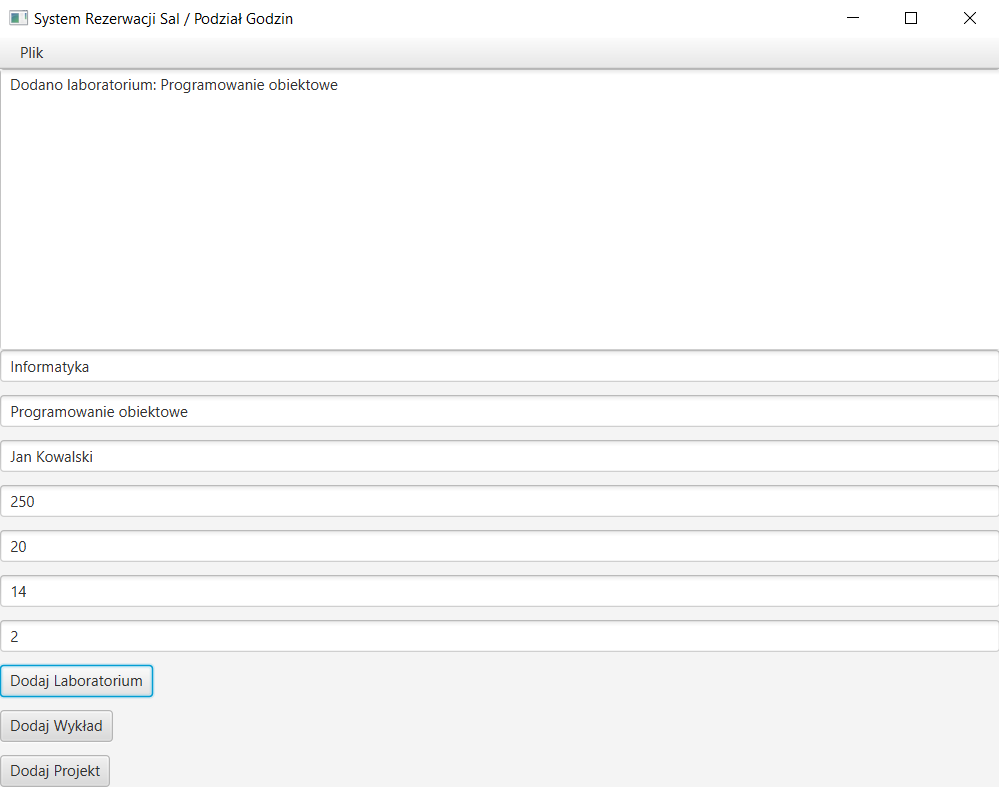
1. **Prezentacja warstwy użytkowej projektu**

Opis aplikacji:

* Główne okno aplikacji umożliwia przeglądanie harmonogramu, dodawanie, edytowanie i usuwanie zajęć.

Rysunek 4. Główne okno aplikacji

* Ekran dodawania zajęć zawiera formularz z polami dla kierunku, nazwy przedmiotu, prowadzącego, sali, dnia tygodnia i godziny zajęć.



Rysunek 5. Przykład wygenerowanego wykresu dla zadanych parametrów

1. **Podsumowanie**

Projekt "System rezerwacji sal/podział godzin" umożliwia efektywne zarządzanie harmonogramem zajęć na uczelni, zapobiega konfliktom rezerwacji oraz oferuje intuicyjny interfejs użytkownika. W przyszłości planowane jest rozszerzenie funkcjonalności o możliwość integracji z systemami zewnętrznymi oraz automatyczne generowanie harmonogramów.

1. **Literatura**
2. Strona do tworzenia diagramu: https://online.visual-paradigm.com/pl/charts/templates/gantt-charts/gantt-chart/