****

**Programowanie obiektowe**

*Aplikacja Magazynowa*

Prowadzący: Autor:

mgr inż. Ewa Żesławska Jakub Wojdyła

w66065

Kierunek: 3 IIZ/2021, grupa GP02

Rzeszów, r.a. 2022/2023

Spis treści

[**1.** **Cele projektu** 3](#_Toc126788847)

[**2.** **Opis techniczny projektu** 3](#_Toc126788848)

[**3.** **Harmonogram realizacji projektu** 4](#_Toc126788849)

[**4.** **Prezentacja warstwy użytkowej projektu** 5](#_Toc126788850)

[**5.** **Testy jednostkowe** 11](#_Toc126788851)

[**6.** **Repozytorium, system kontroli wersji** 11](#_Toc126788852)

[**7.** **Podsumowanie** 12](#_Toc126788853)

[**8.** **Literatura** 12](#_Toc126788854)

1. **Cele projektu**

**Wymagania funkcjonalne**

* Połączenie z bazą danych
* Zalogowanie się ( należy podać prawidłowy login i hasło z bazy danych ).
* Wyświetlanie tabeli ( dane pobiera z bazy danych ).
* Dodawanie, usuwanie oraz edytowanie rekordów w bazie danych.
* Wyświetlanie danych z tabeli na podstawie parametrów takich jak ID, ilość, miejsce.
* Możliwość dodania własnych zapytań SQL do programu.
* Eksportowanie do pliku csv danych z bazy danych.
* Importowanie z pliku csv danych do bazy danych.
* Walidowanie danych wejściowych, tak aby uniemożliwić wprowadzenie niepoprawnych danych.
* Obsługa wyjątków i wyświetlanie odpowiednich komunikatów w przypadku wystąpienia błędów.

**Wymagania niefunkcjonalne**

* Program jest przyjazny, przejrzysty oraz prosty w użyciu przez pracowników.
* Program posiada GUI
* Program tworzony jest w języku Java JDK 17.
* Program nawiązuje połączenie z bazą danych i używa rekordów w niej zapisanych.
* Program jest stabilny i nie powoduje awarii bazy danych ani nieprzewidzianych zachowań.

1. **Opis techniczny projektu**

* Środowisko programistyczne Javy: Java Oracle OpenJDK 17, IntelliJ IDEA
* Komputer z systemem operacyjnym obsługujący język Java wraz z zainstalowanym i skonfigurowanym środowiskiem programistycznym.
* Dodatkowe biblioteki: mssql-jdbc 11.2.3.jre17, JavaFX
* Baza danych MsSQL Server

1. **Harmonogram realizacji projektu**

Poniżej zamieszono harmonogram realizacji projektu.

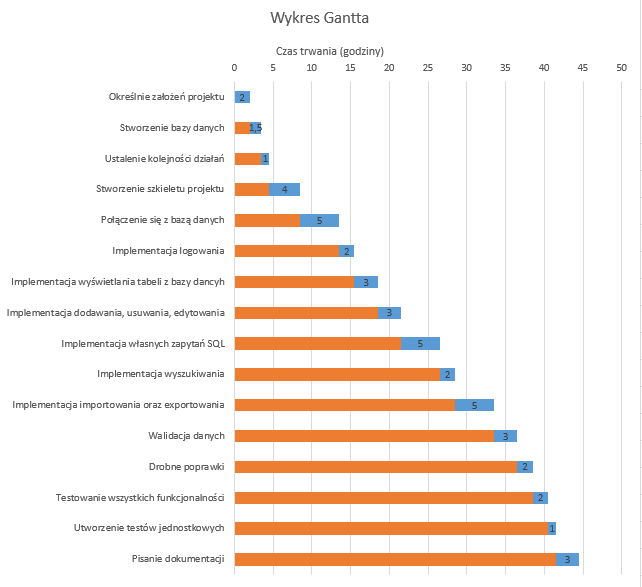
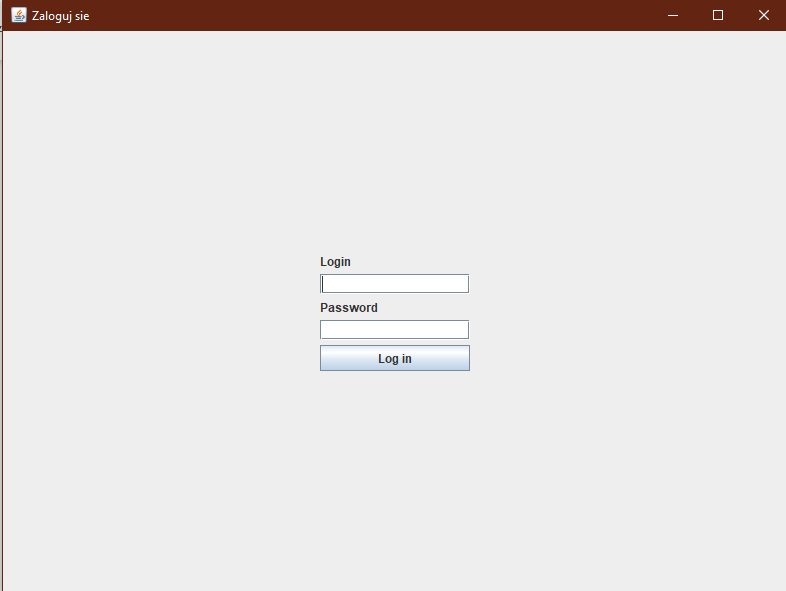


Diagram Gantta

1. **Prezentacja warstwy użytkowej projektu**

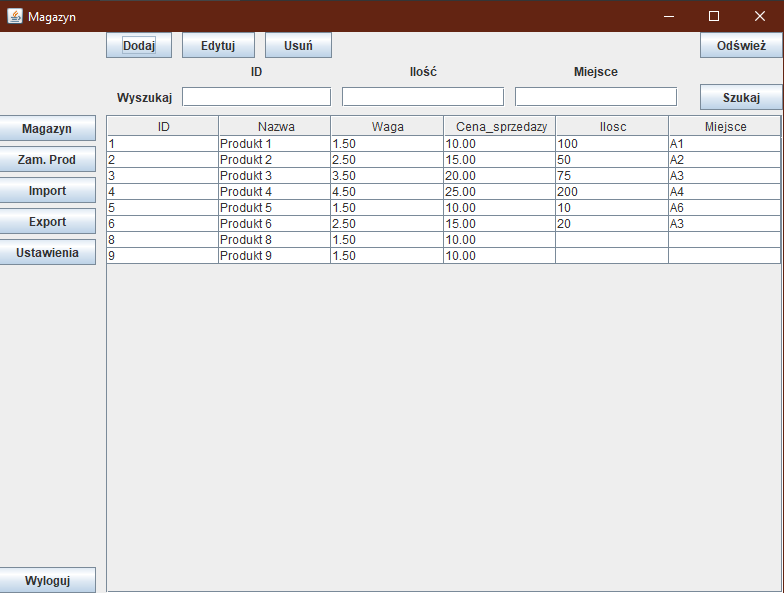
Na rysunku 1 przedstawiono okno logowania. Użytkownik po uruchomieniu aplikacji otrzymuje ekran logowania na którym może:

* Wprowadzić login i hasło po czym spróbować się zalogować.



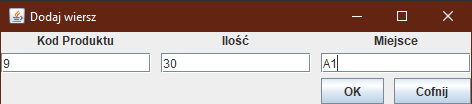
Rysunek 1. Panel logowania

Na rysunku 2 przedstawiono okno główne z domyślną tabelą oraz kilkoma funkcjonalnymi przyciskami. Użytkownik po prawidłowym zalogowaniu się otrzymuje ekran główny na którym może:



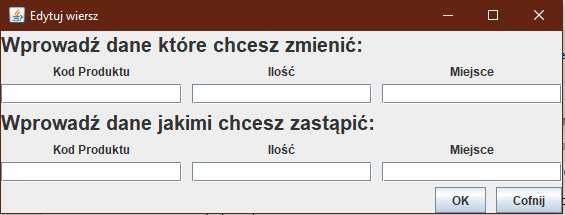
Rysunek 2. Panel główny

* .Dodać, edytować albo usunąć wiersz.
  + Okno dodawania z 3 polami do wprowadzenia danych. Użytkownik po wprowadzeniu odpowiednich danych może dodać kolejny wiersz do tabeli:



Rysunek 3. Panel dodawania

* + Kolejne okno pozwala edytować dane. Użytkownik wprowadza jakie dane chce zamienić na inne



Rysunek 4. Panel edytowania

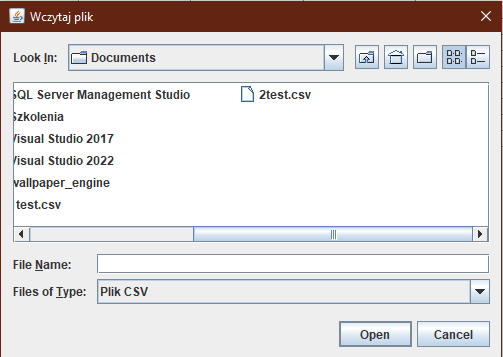
* + Ostatnie Okno pozwala usuwać dane. Użytkownik wprowadza jakie dane chce usunąć.



Rysunek 5. Panel usuwania

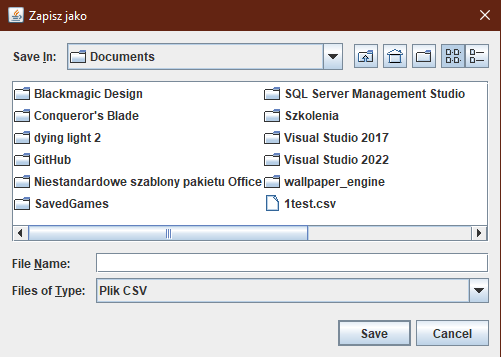
* Odświeżyć
* Wyszukać w tabeli konkretne wartości z ID, ilości oraz miejsca.
* Przełączyć się pomiędzy tabelami

* Import oraz export do pliku csv.
  + Otwiera nam się explorer wtórym wypieramy jaki plik csv chcemy zaimportować.



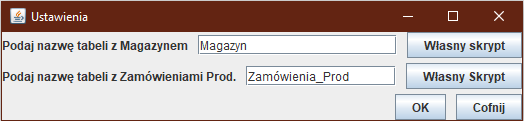
Rysunek 6. Importowanie

* + Otwiera nam się explorer w którym wybieramy nazwę oraz gdzie chcemy zapisać plik csv.



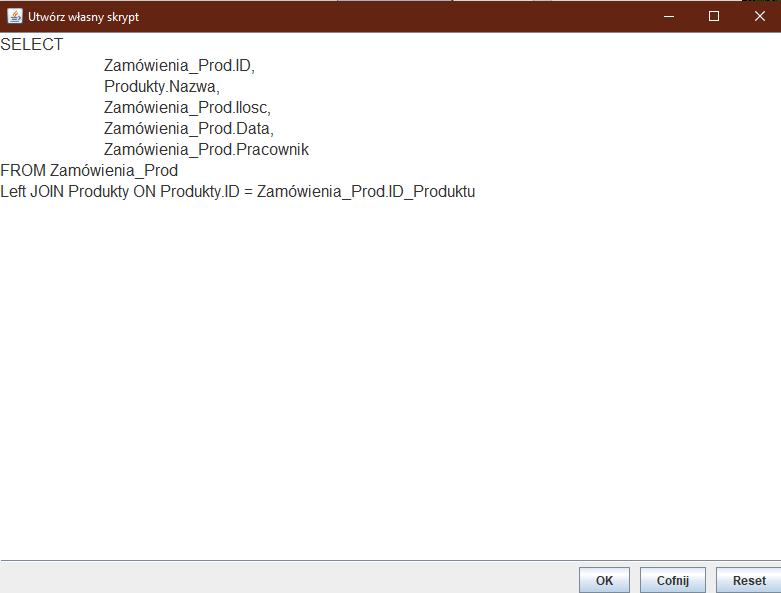
Rysunek 7. Exportowanie

* Ustawienia czyli dodać własne zapytanie SQL
  + Wchodzimy w „ustawienia” możemy skorzystać z domyślnego zapytania podając tylko nazwę albo wprowadzić własny.



Rysunek 8. Ustawienia

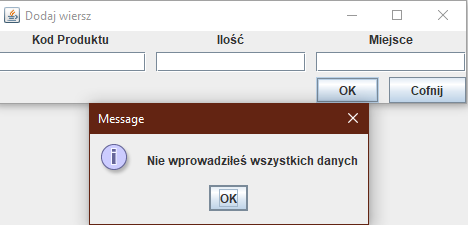
* + „Własny Skrypt” wprowadzamy własne zapytanie, które zmieni nam tabele w panelu głównym. W tym oknie mamy 3 przyciski OK czyli zatwierdzenie, Cofnij aby wyjść bez zmian, Reset napisanie podstawowego zapytania.



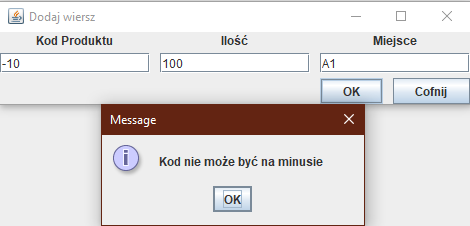
Rysunek 9. Własny skrypt

* Wylogować się

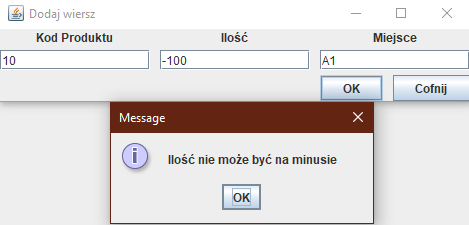
Wszystkie wymagające pola posiadają walidacje danych, w celu uniknięc błędów. Posiadają komunikaty które informują o danym będzędzie.



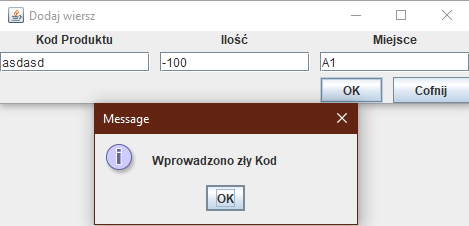
Rysunek 10. Walidacja danych – brak danych



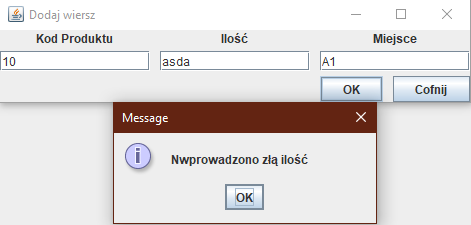
Rysunek 11. Walidacja danych – niepoprawny kod



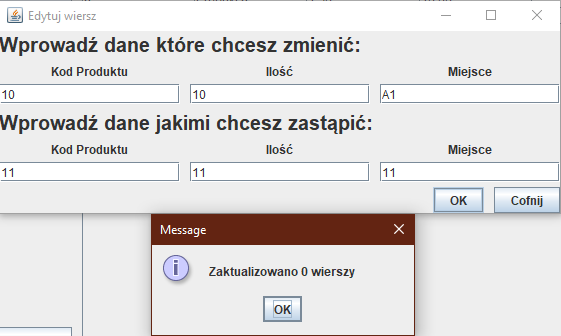
Rysunek 12. Walidacja danych – ujemna ilość



Rysunek 13. Walidacja danych – niepoprawny kod



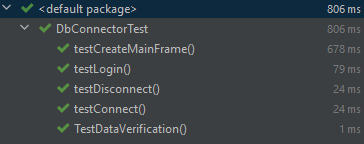
Rysunek 14. Walidacja danych – niepoprawna ilość



Rysunek 10. Informacja o zaktualizowanych/usuniętych wierszach

1. **Testy jednostkowe**

Wykonałem testy jednostkowe dla kilku funkcji takich jak:  
testowałem czy metoda odpowiedzialną za logowanie, czy program prawidłowo łączy się z baza danych oraz rozłącza, czy weryfikacja danych poprawnie sprawdza podając przykładowe dane i czy występuje błąd podczas tworzenia okna głównego.



1. **Repozytorium, system kontroli wersji**

Projekt został zrealizowany z wykorzystaniem systemu kontroli wersji Git. Dokumentacja oraz projekt został umieszczony w repozytorium dostępnym pod adresem:

<https://github.com/CZARNY77/Object_Programming>

1. **Podsumowanie**

Dokumentacja powstała w celu przybliżenia odbiorcy działania programu oraz jego aspektów technicznych. Program spełnia wszystkie założenia oraz jest łatwy do obsługi. Nie było by większego problemu z rozbudową, chociaż potrzeba by była lekka reorganizacja kodu, co spowodowane było brakiem doświadczenia pisaniu w java.

1. **Literatura**
2. <https://chat.openai.com/> (data dostępu 05.02.2023)
3. https://www.youtube.com/ (data dostępu 04.02.2023)

1. <https://stackoverflow.com> (data dostępu: 04.02.2023)