

Politechnika Świętokrzyska
Wydział: Elektrotechniki, Automatyki i Informatyki

Przedmiot: Programowanie obiektowe w języku Java

Temat projektu: Aplikacja kliencka do bazy danych

Skład zespołu projektowego: Rudzki Marcin, Sadza Jakub, Rozpara Bartosz

Grupa: 2ID13A , rok studiów: 2rok

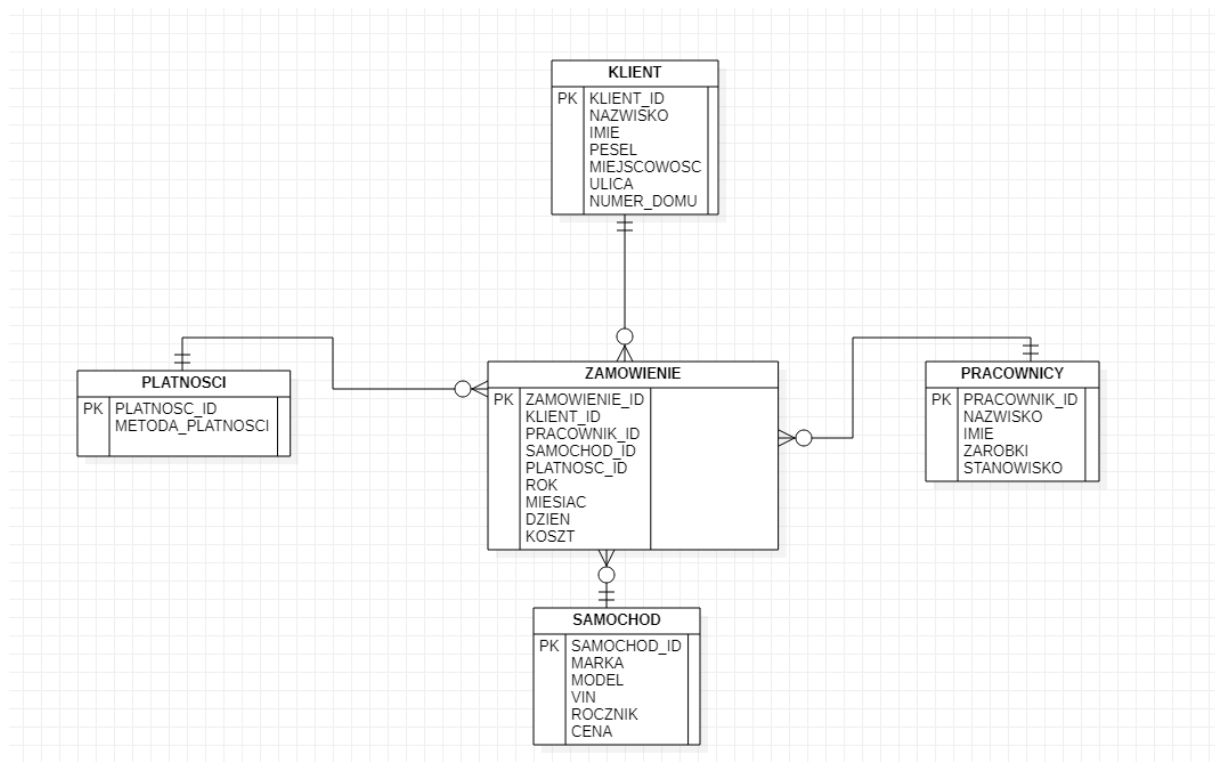
Ogólny opis:

Klient bazy danych stworzony w Javie, przy pomocy programu IntelliJ IDEA. Użyty framework to javafx, oraz biblioteki służące do wyświetlania obiektów, scen, aplikacji. Projekt został stworzony jako projekt Maven. Nasz klient to aplikacja służąca do zarządzania bazą danych. Do tego celu stworzyliśmy własnoręcznie zaprojektowaną bazę aby to właśnie o nią oprzeć nasze poszczególne zapytania pisane w naszym projekcie. Aplikacja ta posiada najbardziej potrzebne funkcje do obsługi bazy danych. A są nimi: dodawanie rekordów, usuwanie oraz ich wyświetlanie.

Wykorzystane wersje frameworków/komponentów:

- IntelliJ IDEA 2021.1.2 Ultimate Edition
- javafx 17-ea + 11
- ojdbc 8 w wersji 21.1.00.0.0
- JUnit5 w wersji 5.8
- Maven 4.0.0
- JDK w wersji 15.1
- Oracle 19

Schemat ERD bazy danych:



Definicje kolumn oraz relacje bazy:

```
CREATE TABLE platnosci (
  platnosc_id NUMBER(5) NOT NULL PRIMARY KEY,
  metoda_platnosci VARCHAR2(20)--np karta/gotowka/przelew/
);

CREATE TABLE samochod (
  samochod_id NUMBER(5) NOT NULL PRIMARY KEY,--numer samochodu
  marka VARCHAR2(20),--marka samochodu
  model VARCHAR2(20),--model samochodu
  VIN VARCHAR2(17),--numer vin
  rocznik NUMBER(4),--rocznik auta np 2013
  cena NUMBER(9,2)--cena za samochod w przypadku kupna podana w euro np. 30203,33
);

CREATE TABLE pracownicy (
  pracownik_id NUMBER(5) NOT NULL PRIMARY KEY,--numer klienta
  nazwisko VARCHAR2(40),--nazwisko np. szchachmed
  imie VARCHAR2(35),--imie np abdul
  zarobki NUMBER(10),--zarobki w euro
  stanowisko VARCHAR2(40)--stanowisko jakie obejmuje pracownik, np. doradca klienta
);

create sequence klientdane start with 21 increment by 1;
create sequence pracownicydane start with 13 increment by 1;
create sequence samochoddane start with 26 increment by 1;
create sequence zamowieniadane start with 21 increment by 1;

CREATE TABLE klient (
  klient_id NUMBER(5) NOT NULL PRIMARY KEY,--numer unikalny
  nazwisko VARCHAR2(20),--nazwisko np brzeczyszczkiewicz
  imie VARCHAR2(15),--imie np zdzichu
  PESEL NUMBER(11),--numer pesel
  miejscowosc VARCHAR2(30),--np. dorohusk, kielce, warszawa,monaco
  ulica VARCHAR2(30),--np. dziury duze
  numer_domu VARCHAR2(7)--np. 95499
);

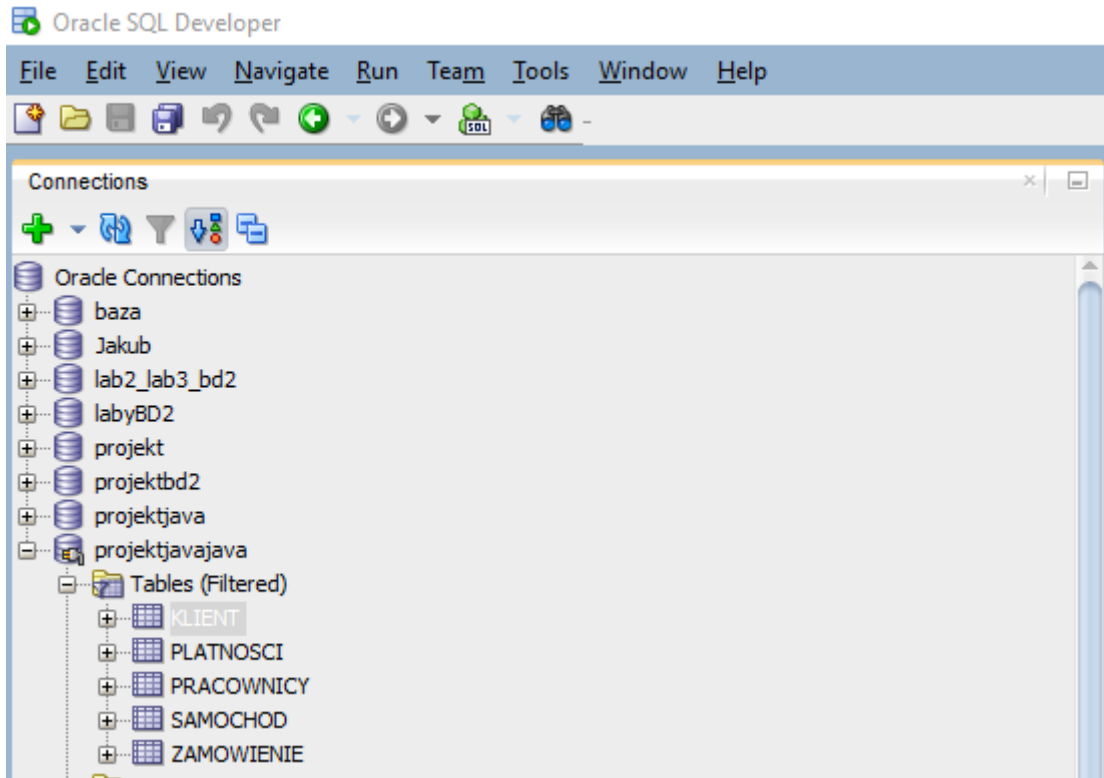
CREATE TABLE zamowienie (
  zamowienie_id NUMBER(5) NOT NULL PRIMARY KEY, --numer zamowienia
  klient_id NUMBER(5) NOT NULL, --numer klienta
  pracownik_id NUMBER(5) NOT NULL, --numer pracownika
  samochod_id NUMBER(5) NOT NULL, --numer samochodu
  platnosc_id NUMBER(5) NOT NULL, -- numer platnosci
  rok number(5), -- rok zlozenia zamowienia
  miesiac number(5), -- miesiac zlozenia zamowienia
  dzien number(5), -- dzien zlozenia zamowienia
  koszt number(10),-- koszt zamowienia na samochod
  CONSTRAINT ZAMOWIENIE_fk FOREIGN KEY (klient_id) REFERENCES klient(klient_id),
  CONSTRAINT ZAMOWIENIE_fk1 FOREIGN KEY (platnosc_id) REFERENCES platnosci(platnosc_id),
  CONSTRAINT ZAMOWIENIE_fk2 FOREIGN KEY (samochod_id) REFERENCES samochod(samochod_id),
  CONSTRAINT ZAMOWIENIE_fk3 FOREIGN KEY (pracownik_id) REFERENCES pracownicy(pracownik_id)
);
```

Sposób importowania plików csv do bazy danych (przykład):

Po wpisaniu kodu tworzącego tabele i relacje należy:

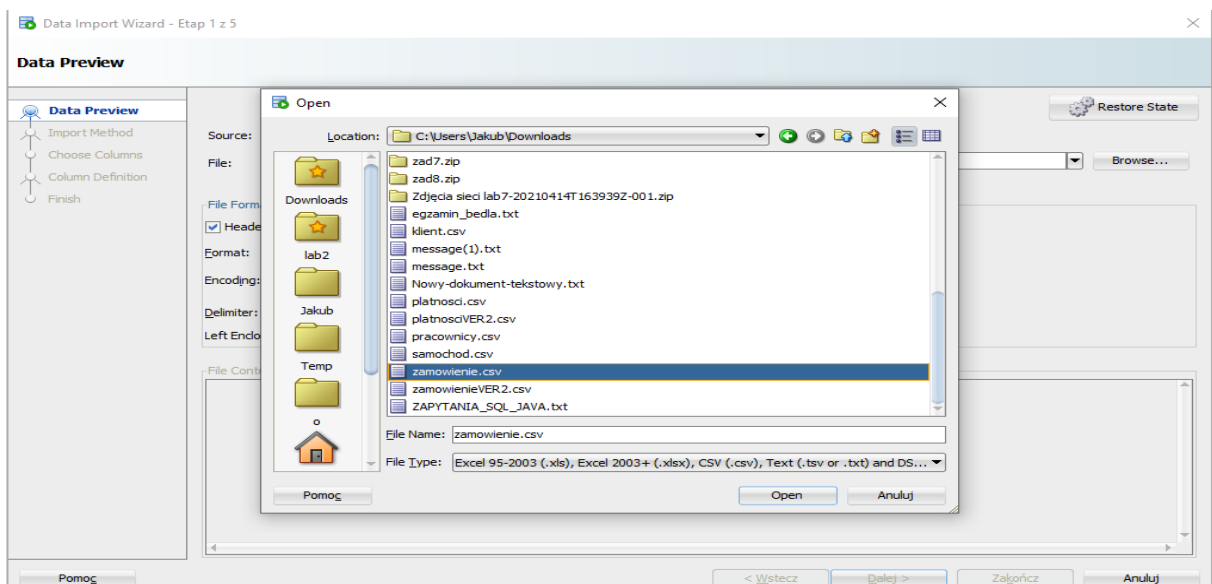
Krok.1

Wybieramy tabelę, do której będziemy ładować dane (klikamy PPM na tabelę, rozwija nam się menu poleceń, i wybieramy opcję „import data”).



KROK.2

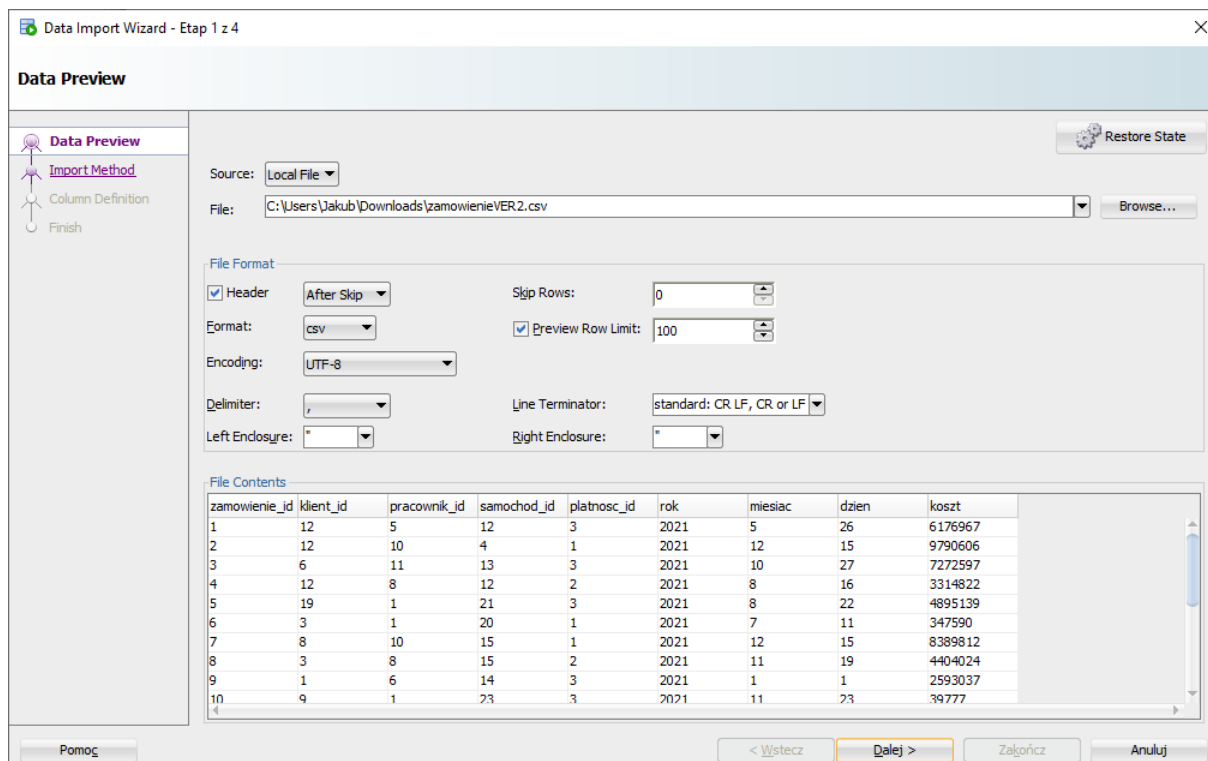
Wybieramy interesujący nas plik z wygenerowanymi danymi. Klikamy **Open**



KROK.3

W tym kroku wybieramy odpowiednie kodowanie naszych danych, oraz rodzaj oddzielenie jednej kolumny od drugiej, w moim przypadku jest to znak ',' .

Klikamy **Dalej**

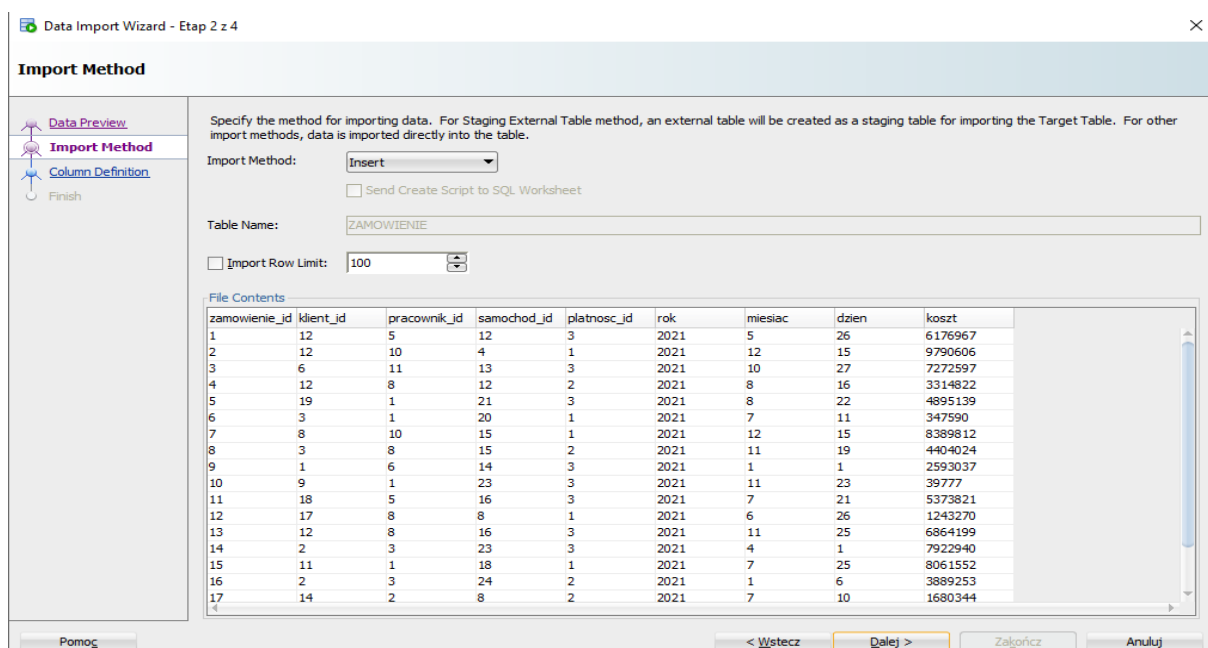


The screenshot shows the 'Data Preview' step of the Data Import Wizard. The 'Source' is set to 'Local File' and the 'File' is 'C:\Users\Jakub\Downloads\zamowienieVER2.csv'. The 'File Format' section shows 'Header' checked, 'Format' as 'csv', 'Encoding' as 'UTF-8', 'Delimiter' as ',', and 'Line Terminator' as 'standard: CR LF, CR or LF'. The 'File Contents' table displays 10 rows of data with columns: zamowienie_id, klient_id, pracownik_id, samochod_id, platnosc_id, rok, miesiac, dzien, and koszt.

zamowienie_id	klient_id	pracownik_id	samochod_id	platnosc_id	rok	miesiac	dzien	koszt
1	12	5	12	3	2021	5	26	6176967
2	12	10	4	1	2021	12	15	9790606
3	6	11	13	3	2021	10	27	7272597
4	12	8	12	2	2021	8	16	3314822
5	19	1	21	3	2021	8	22	4895139
6	3	1	20	1	2021	7	11	347590
7	8	10	15	1	2021	12	15	8389812
8	3	8	15	2	2021	11	19	4404024
9	1	6	14	3	2021	1	1	2593037
10	9	1	23	3	2021	11	23	39777

KROK.4

Następnie wybieramy metodę importowania danych, zazwyczaj nic tutaj nie zmieniamy. (metoda insert) Klikamy **Dalej**



The screenshot shows the 'Import Method' step of the Data Import Wizard. The 'Import Method' is set to 'Insert'. The 'Table Name' is 'ZAMOWIENIE'. The 'Import Row Limit' is set to 100. The 'File Contents' table displays 17 rows of data with columns: zamowienie_id, klient_id, pracownik_id, samochod_id, platnosc_id, rok, miesiac, dzien, and koszt.

zamowienie_id	klient_id	pracownik_id	samochod_id	platnosc_id	rok	miesiac	dzien	koszt
1	12	5	12	3	2021	5	26	6176967
2	12	10	4	1	2021	12	15	9790606
3	6	11	13	3	2021	10	27	7272597
4	12	8	12	2	2021	8	16	3314822
5	19	1	21	3	2021	8	22	4895139
6	3	1	20	1	2021	7	11	347590
7	8	10	15	1	2021	12	15	8389812
8	3	8	15	2	2021	11	19	4404024
9	1	6	14	3	2021	1	1	2593037
10	9	1	23	3	2021	11	23	39777
11	18	5	16	3	2021	7	21	5373821
12	17	8	8	1	2021	6	26	1243270
13	12	8	16	3	2021	11	25	6864199
14	2	3	23	3	2021	4	1	7922940
15	11	1	18	1	2021	7	25	8061552
16	2	3	24	2	2021	1	6	3889253
17	14	2	8	2	2021	7	10	1680344

KROK.5

W tym etapie nic nie zmieniamy, lecz możemy zdecydować które nazwy kolumn mają się znajdować w naszej hurtowni. Klikamy **Dalej**

Data Import Wizard - Etap 3 z 5

Choose Columns

Select the columns to import from the data set and arrange them in the order you want.

Available Columns

Selected Columns

zamowienie_id
klient_id
pracownik_id
samochod_id
platnosc_id
rok
miesiac
dzien
koszt

File Contents

	zamowienie_id	klient_id	pracownik_id	samochod_id	platnosc_id	rok	miesiac	dzien	koszt
1	12	5	12	3	2021	5	26	6176967	
2	12	10	4	1	2021	12	15	9790606	
3	6	11	13	3	2021	10	27	7272597	

Pomoc < Wstecz **Dalej** > Zakończ Anuluj

KROK.6

W kolejnym etapie klikamy **Dalej**. Jest to etap w którym możemy podejrzeć co trafi z naszego pliku do tabeli bazy danych.

Data Import Wizard - Etap 4 z 5

Column Definition

For each column in the Source Data Columns list on the left, select a Target Table column on the right.

Match By: Name

Source Data Columns

zamowienie_id
klient_id
pracownik_id
samochod_id
platnosc_id
rok
miesiac
dzien
koszt

Status

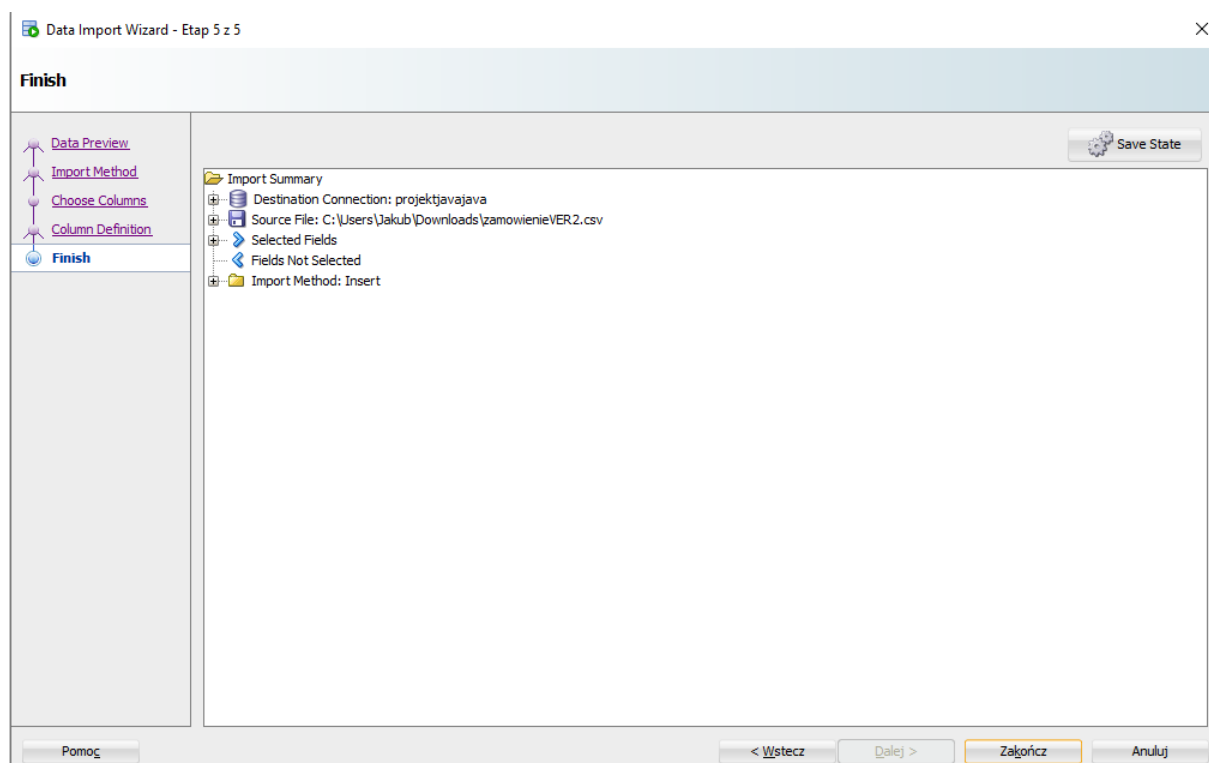
Target Table Columns

Name: ZAMOWIENIE_ID
Data Type: NUMBER
Size/Precision: 5
Scale: 0
Nullable?: ☐
Default:
Comment:
Data:
1
2
3
4
5
6
7
8
9

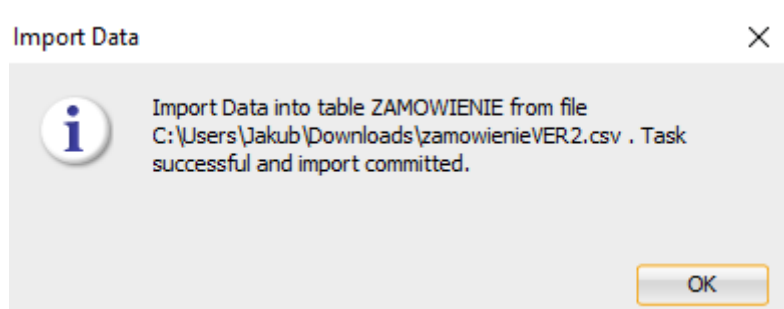
Pomoc < Wstecz **Dalej** > Zakończ Anuluj

KROK.7

Jest to etap kończący cały proces importowania danych z pliku .csv do tabeli.
Klikamy **Zakończ**

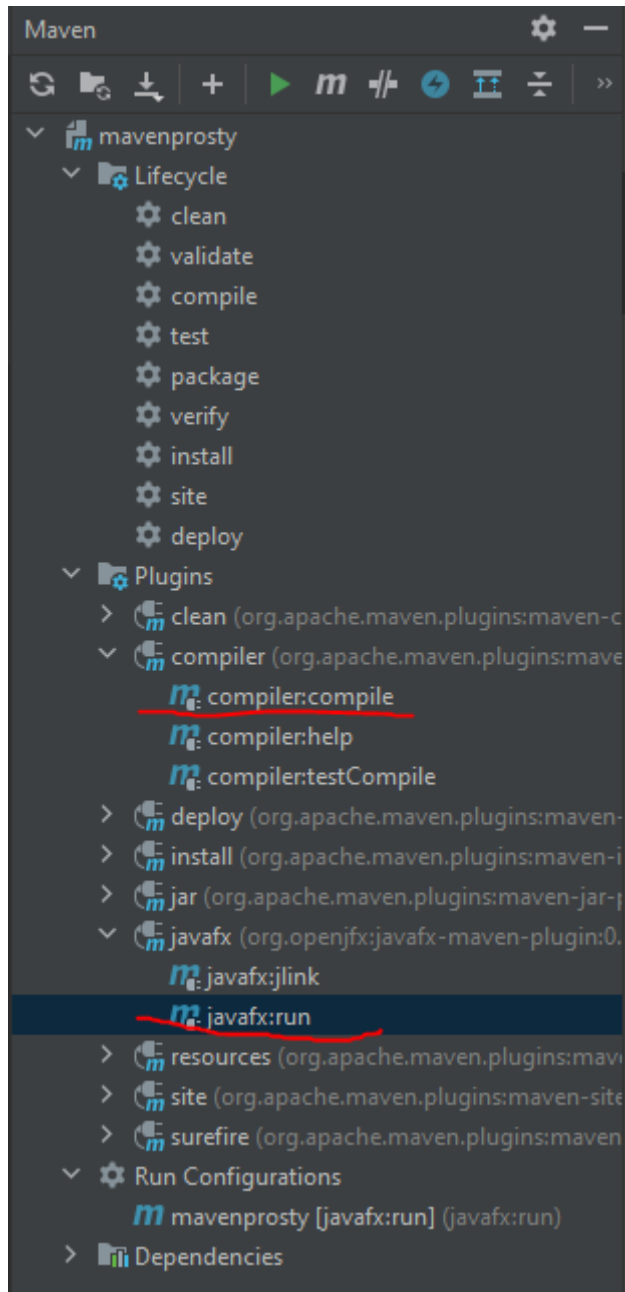


Na koniec możemy ujrzeć informację po tym procesie, która informuje nas, że wszystko przebiegło prawidłowo.



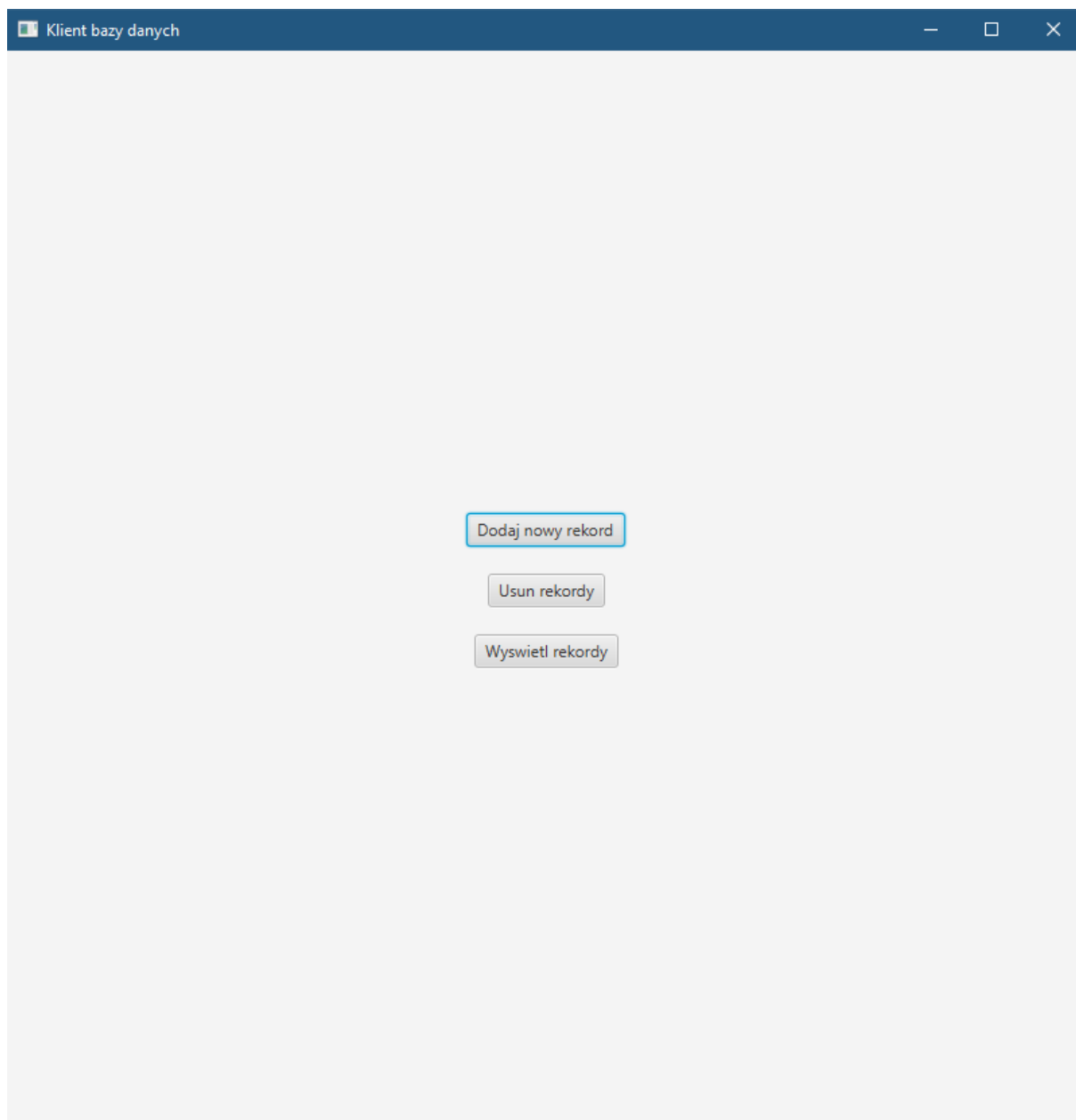
Sposób uruchomienia oraz obsługa projektu:

Projekt przygotowany został przy użyciu środowiska JetBrains IntelliJ'a więc zalecany jest ten program do użytku, aby uruchomić naszą aplikację. Aby uruchomić program należy otworzyć go za pomocą opcji „plik” -> „Otwórz projekt”, a następnie wybrać katalog projektu. Później należy chwilę poczekać na załadowanie się wszystkich komponentów projektu oraz załadowanie użytych w projekcie zależności Maven. Aby uruchomić projekt należy wybrać z menu kontekstowego Maven wybrać pierw opcję :compilowania projektu a następnie jego uruchomienia poprzez opcję :javaFX:run



Uruchamiając projekt nawiązywana jest łączność z bazą danych za pomocą sterownika „ojdbc” w wersji 8.

Obsługa projektu jest równie prosta co jego uruchomienie. Po uruchomieniu wyświetla się nam tak zwane „okienko startowe” aplikacji, gdzie użytkownik ma doczynienia z różnymi przyciskami.



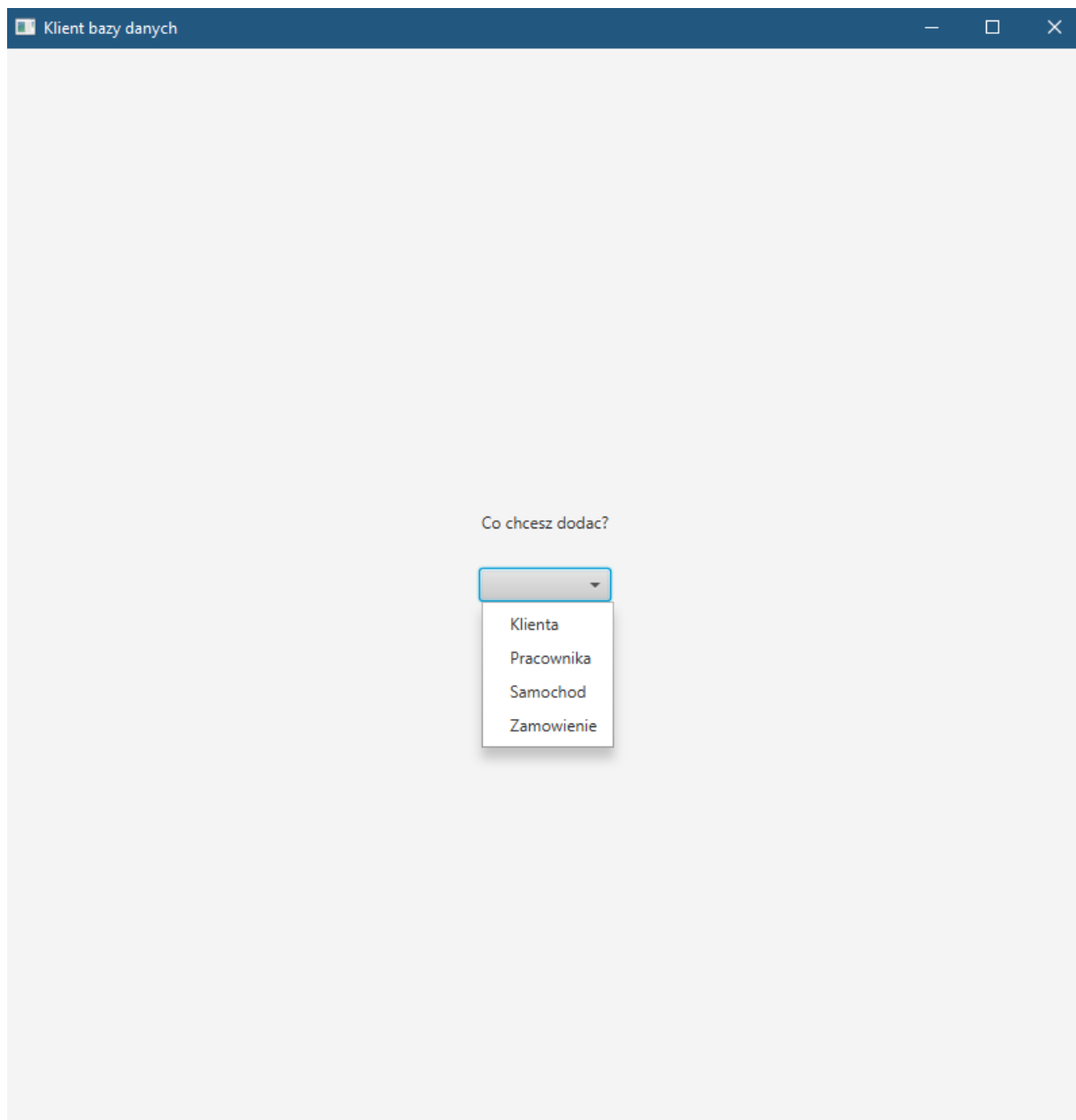
Tymi przyciskami są właśnie polecenia które użytkownik/ osoba obsługująca bazę danych chce wykonać w bazie. Poprzez wybór odpowiedniej frazy/przycisku, przechodzimy odpowiednio do kolejnego etapu, do kolejnej sceny, w którym użytkownik decyduje na jakiej tabeli mają zajść zmiany, i swój wybór potwierdza klikając odpowiedni przycisk.

Informacje na temat funkcjonalności:

Dostępne są trzy funkcjonalności:

- Dodawanie rekordów
- Usuwanie rekordów
- Wyświetlanie rekordów

Po wybraniu opcji „Dodawania rekordów” wyświetla nam się nowa scena, w której użytkownik zostaje zapytany o rodzaj informacji jaką chce dodać do bazy.



Klient bazy danych

Imie:

Nazwisko:

Pesel:

Miejscowosc:

Ulica:

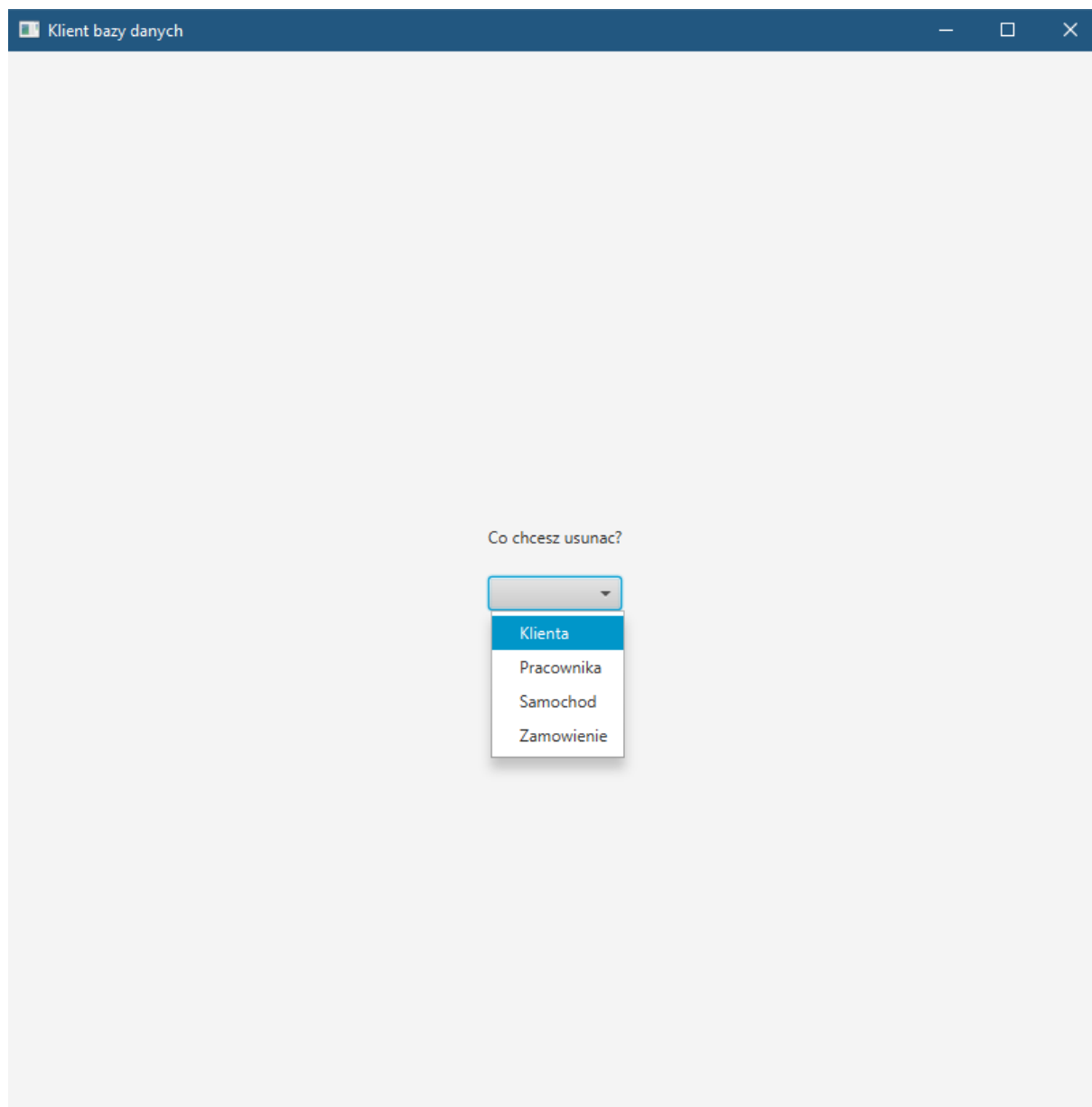
Numer domu:

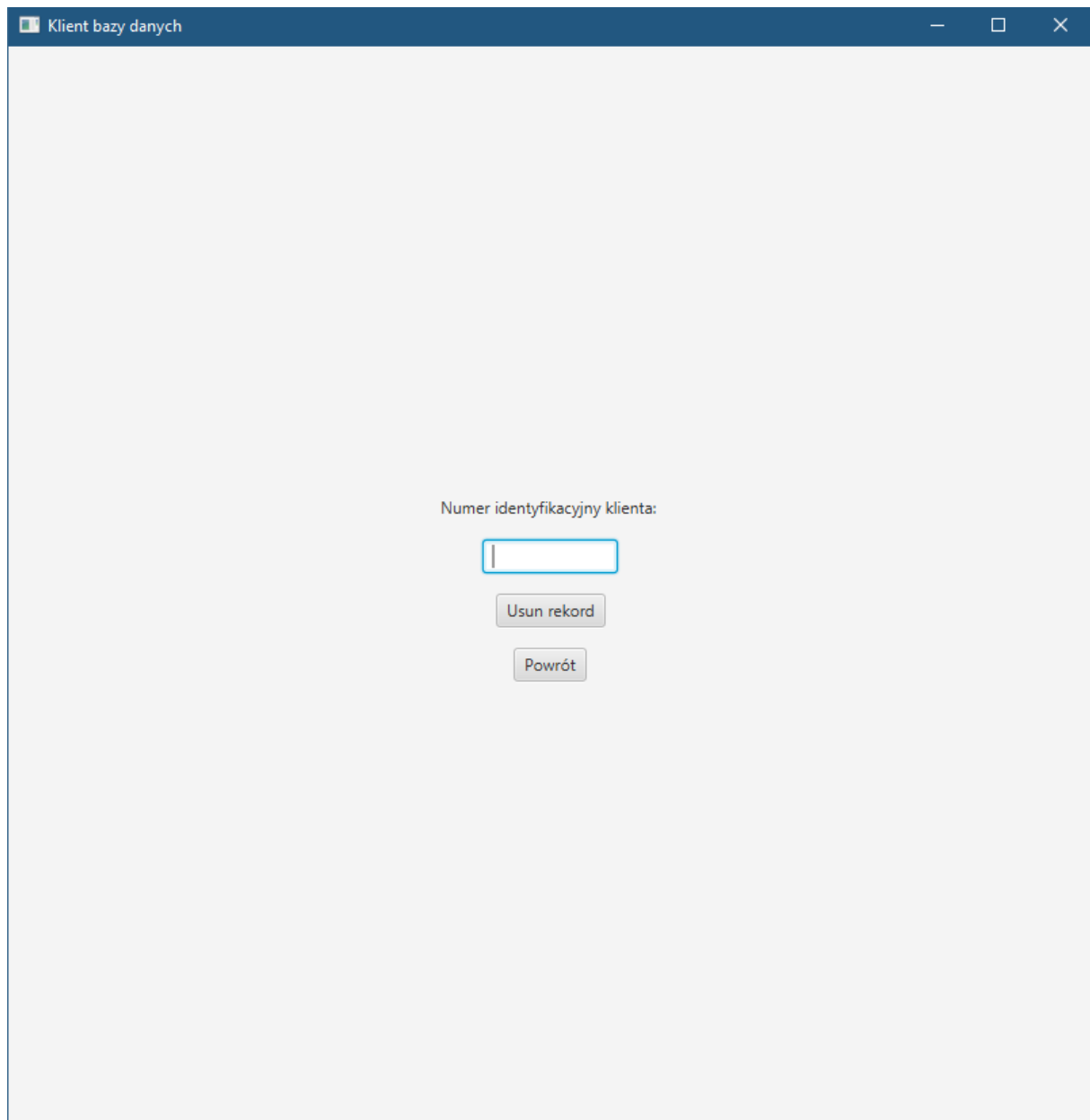
Dodaj rekord

Powrot

Po wypełnieniu formularza, jeżeli wszystkie dane są poprawne*, to znaczy nie występują żadne powtórzenia w polach identyfikacyjnych (np. Tabela zamówienia) zostaje dodany nowy rekord z podanymi przez użytkownika danymi. Po dodaniu rekordu użytkownik ma możliwość powrotu do poprzedniej sceny czyli wyboru tabeli lub wypełnienia obecnego formularza nowymi danymi.

Po wybraniu opcji „Usuwanie rekordów” wyświetla nam się nowa scena, w której użytkownik zostaje zapytany o rodzaj informacji jaką chce usunąć z bazy.





Klient bazy danych

Numer identyfikacyjny klienta:

Usun rekord

Powrót

Po podaniu numeru identyfikacyjnego klienta, jeżeli został znaleziony dany rekord w bazie*, następuje jego usunięcie. Po usunięciu rekordu użytkownik ma możliwość powrotu do poprzedniej sceny czyli wyboru tabeli lub wypełnienia obecnego formularza nowymi danymi. Zdecydowaliśmy się na taką funkcjonalność ponieważ dane w tabelach np. Klient lub pracownik zawierają dane osobiste na które dana osoba może nie wyrażać zgody na dalsze przetwarzanie.

Po wybraniu opcji „Wyświetlanie rekordów” wyświetla nam się nowa scena, w której użytkownik zostaje zapytany o rodzaj informacji jaką chce wyświetlić z bazy .

Informacje na temat stworzonych klas, metod i funkcji:

--Connect.java

Klasa w która opisuje jak aplikacja nawiązuje połączenie z bazą danych.

Rezultat tej klasy można zauważyć chociażby w konsoli, poprzez zobaczenie komunikatu: „Połączono z bazą danych” – w przypadku powodzenia, oraz „Połączenie z bazą danych zostało zakończone” – ten komunikat następuje po wyjściu z aplikacji.

Connect() – funkcja w której następuje połączenie z bazą danych.

getConnection() - Pobiera bazowe ADO.NET DbConnection dla tego elementu DbContext .

public void insertClientsData, public void insertWorkersData, public void insertCarData, public void insertOrderData – są to funkcję służące jako przekaźnik kodu/zapytania SQL.

public void disconnect() – funkcja odpowiedzialna za zamknięcie połączenia z bazą danych.

--DBConnector.java

Klasa która przechowuje dane dotyczące logowania się z bazą danych.

public class DBConnector()- funkcja w której przekazanymi parametrami są dane logowania do bazy danych.

--App.java

Klasa odpowiadająca za okienko główne aplikacji.

W niej zawarty jest kod, w którym poprzez formularze, przesyłamy dane do naszej bazy danych.

Zawiera podstawowy kod dotyczący funkcjonalności bazy – dodawanie rekordów, wyświetlanie, i usuwanie.

public class App extends Application implements EventHandler<ActionEvent> - funkcja główna aplikacji odpowiedzialna za zdefiniowanie nowego okna i nowych scen.

public void start() – funkcja określająca scenę startową.

VBox view = new VBox(20); - funkcja VBox układa wszystkie podrzędne składowe w jednej kolumnie.

Label label = new Label("Co chcesz wyświetlić?"); - wyświetla napis .

ChoiceBox<String> choiceBox = new ChoiceBox<>(); - metoda ta wyświetla menu rozwijane.

Button backButton = new Button("Powrot"); - metoda ta tworzy nowy przycisk o podanej w parametrze nazwie.

AlertBox.display – wyświetla komunikat błędu

mainView = new Scene(view, 800,800); - metoda opisująca parametry okienka, sceny.

mainDelettee.setAlignment(Pos.CENTER); - metoda która ustawia treść na scenie w pozycji centralnej.

backOptio.setOnAction(e->changeScene(window,scene1)); - metoda odpowiedzialna za cofnięcie się do sceny początkowej

window.show();

- metoda odpowiedzialna za wyświetlenie się okienka

window.setTitle –metoda odpowiedzialna za ustawienie tytułu okna

public void sceneChoice()- funkcja odpowiedzialna za wybór sceny.

public void changeScene()- funkcja odpowiedzialna za zmianę sceny,

public void getChoice()- funkcja odpowiedzialna za wybór pola w polu ChoiceBox, zwraca wartość.

public static void main()- funkcja główna aplikacji

Informacje na temat ilości pracy włożonej przez poszczególnych członków zespołu w tworzenie projektu:

-- Praca przy kodzie aplikacji: Rudzki Marcin

- Funkcje usuwania i dodawania rekordów : Rudzki Marcin
- Funkcja wyświetlania tabel : Rozpara Bartosz

-- Wykonanie testów jednostkowych : Rudzki Marcin

-- Testowanie aplikacji : Rudzki Marcin, Sadza Jakub

-- Wykonanie bazy danych i zapytań : Sadza Jakub, Rudzki Marcin, Bartosz Rozpara

-- Sprawozdanie : Sadza Jakub

-- Dokumentacja javadoc : Sadza Jakub