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

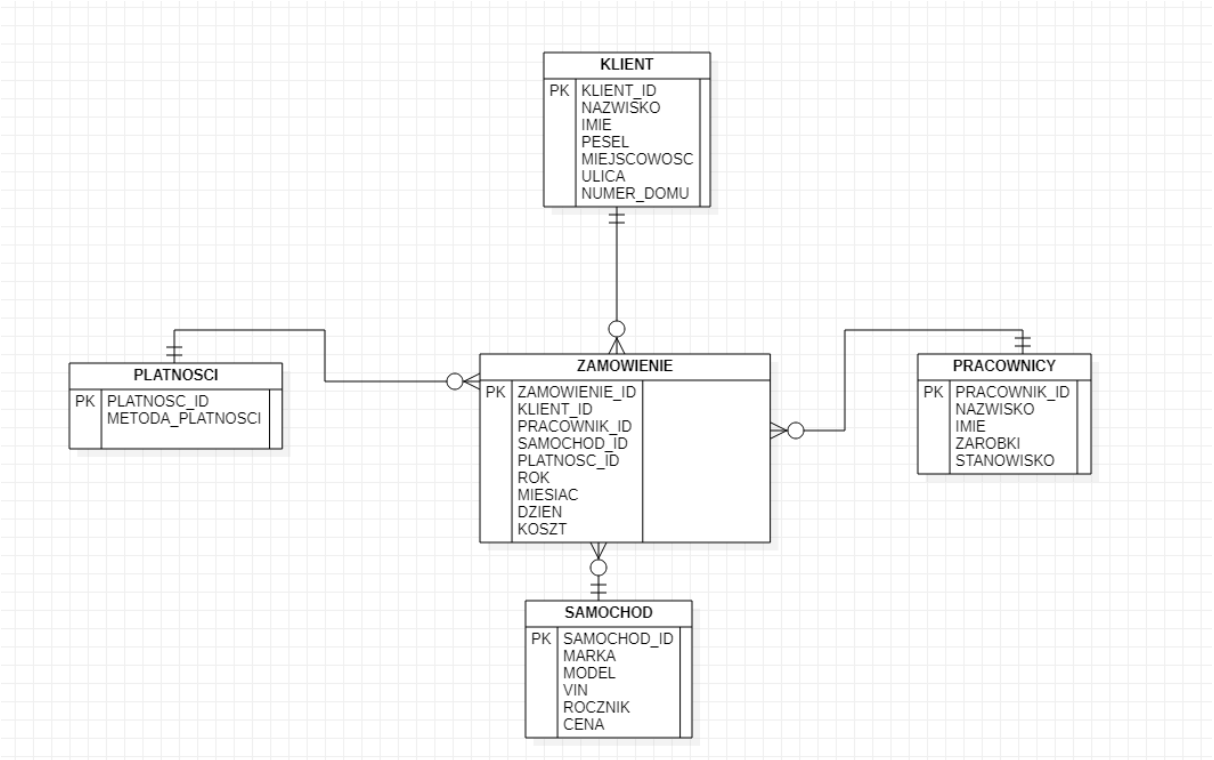
Ogólny opis:

Klient bazy danych stworzony w Javie, przy pomocy programu IneliJ 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 zaprojektowana bazę aby to własnie 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:

- InteliJ 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

Schemat ERD bazy danych:

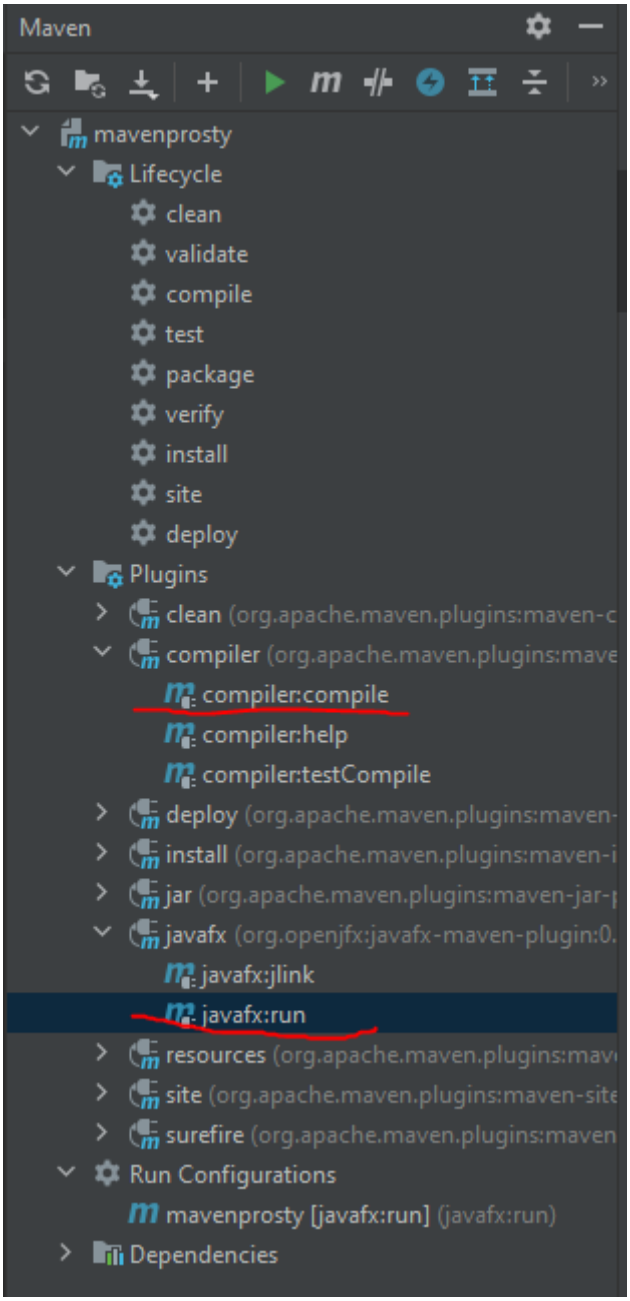


Definicje kolumn oraz relacje bazy:

```
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 brzeczyszczykiewicz
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)
);
```

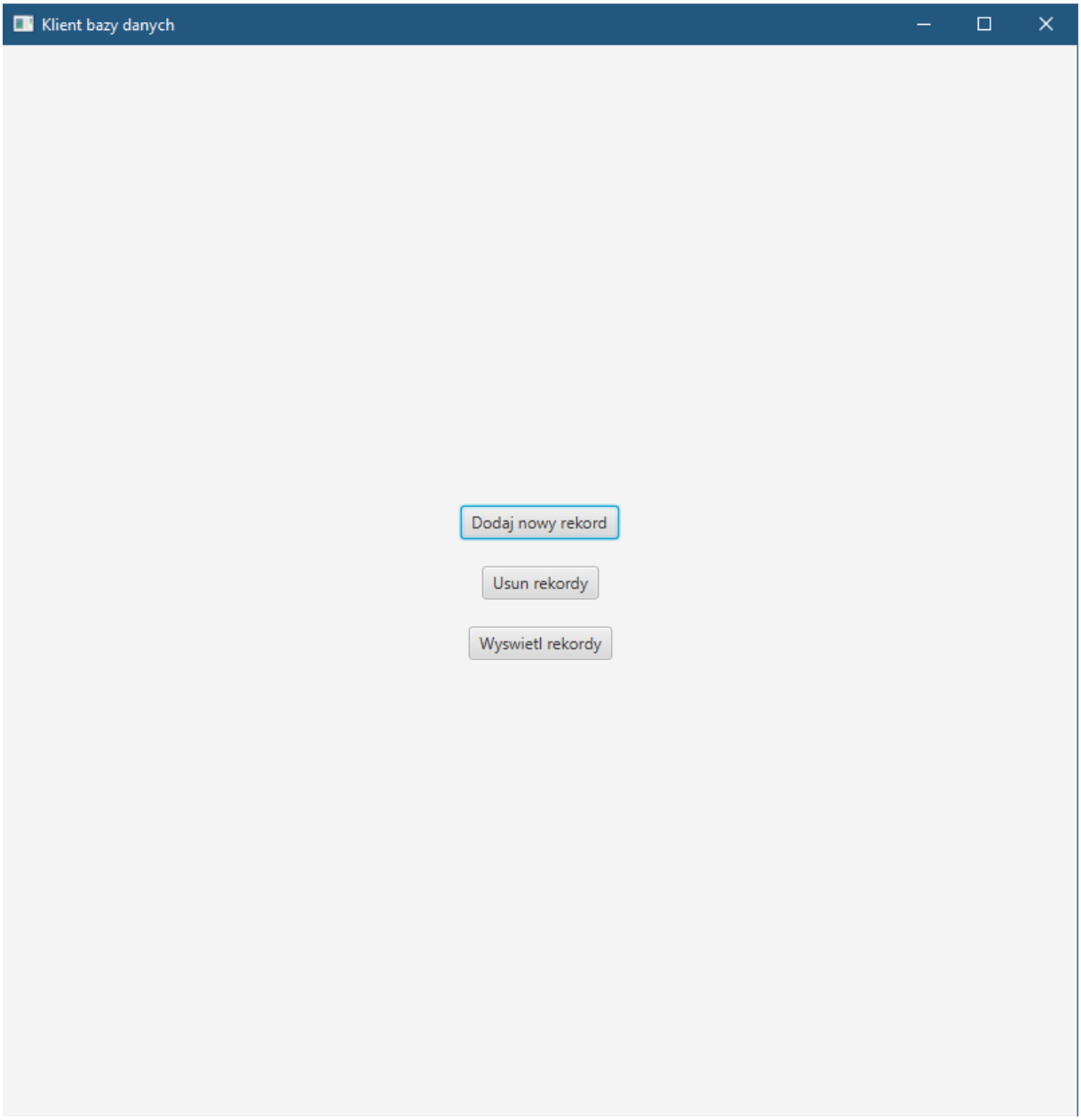
Sposób uruchomienia oraz obsługa projektu:

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



Uruchamiając projekt nawiązywana jest łączność z bazą danych za pomocą steronika „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 róznymi przyciskami.



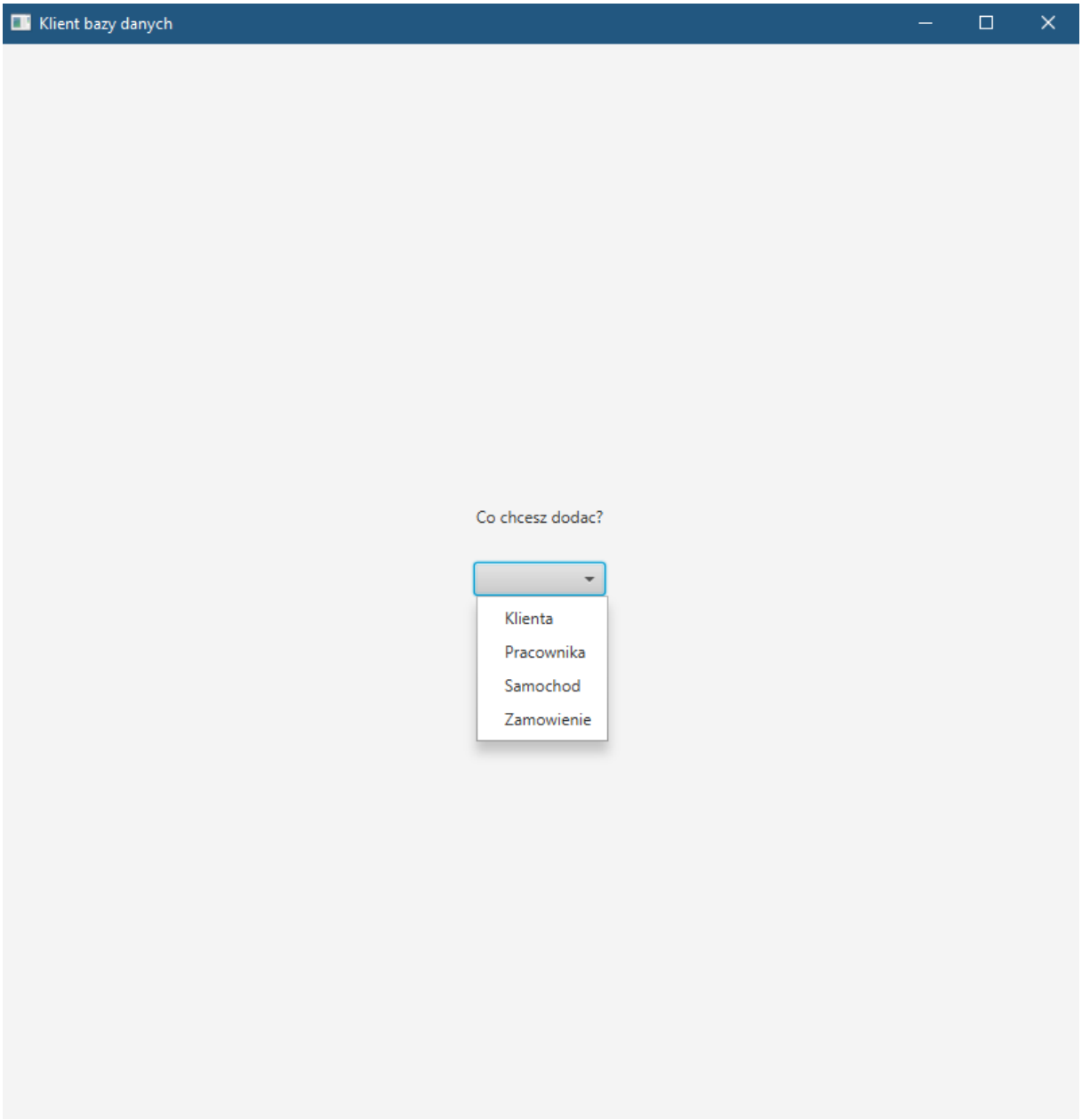
Tymi przyciskami są właśnie polecenia które użytkownik/ osoba obsługująca bazę danych chce wykonać w bazie. Poprzez wybór odpowieniej frazy/przycisku, przechodzimy odpowiednio do kolejnego etapu, do kolejnej sceny, w którym użytkownik decyduje na jakiej tabeli mają zajść zmiany, i swój wybór potwierdza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

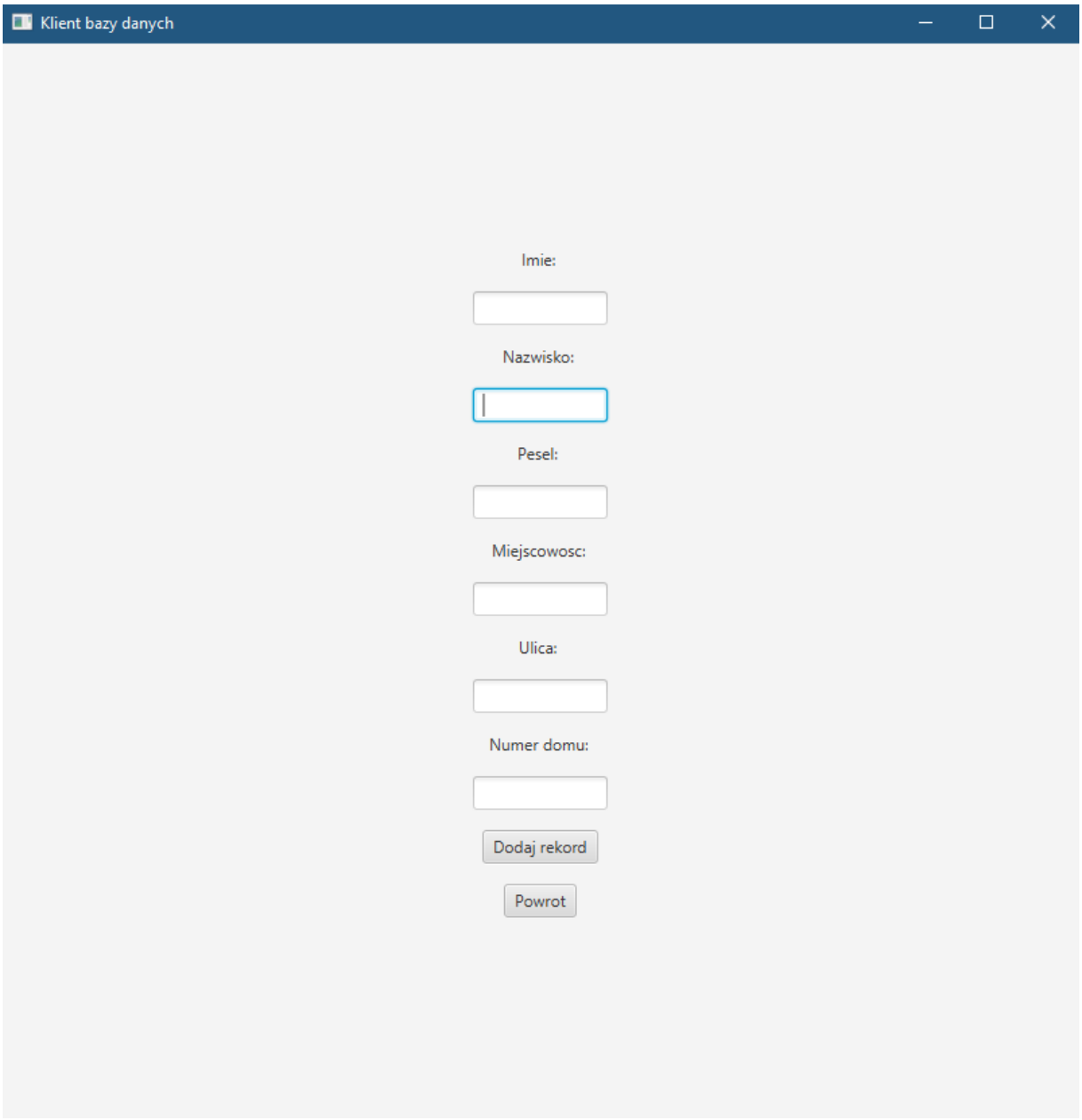
Informacje na temat funkcjonalności:

Dostępne są trzy funkcjonalności:

- Dodawanie rekordów
- Usuwanie rekordów
- Wyświetlanie rekordów

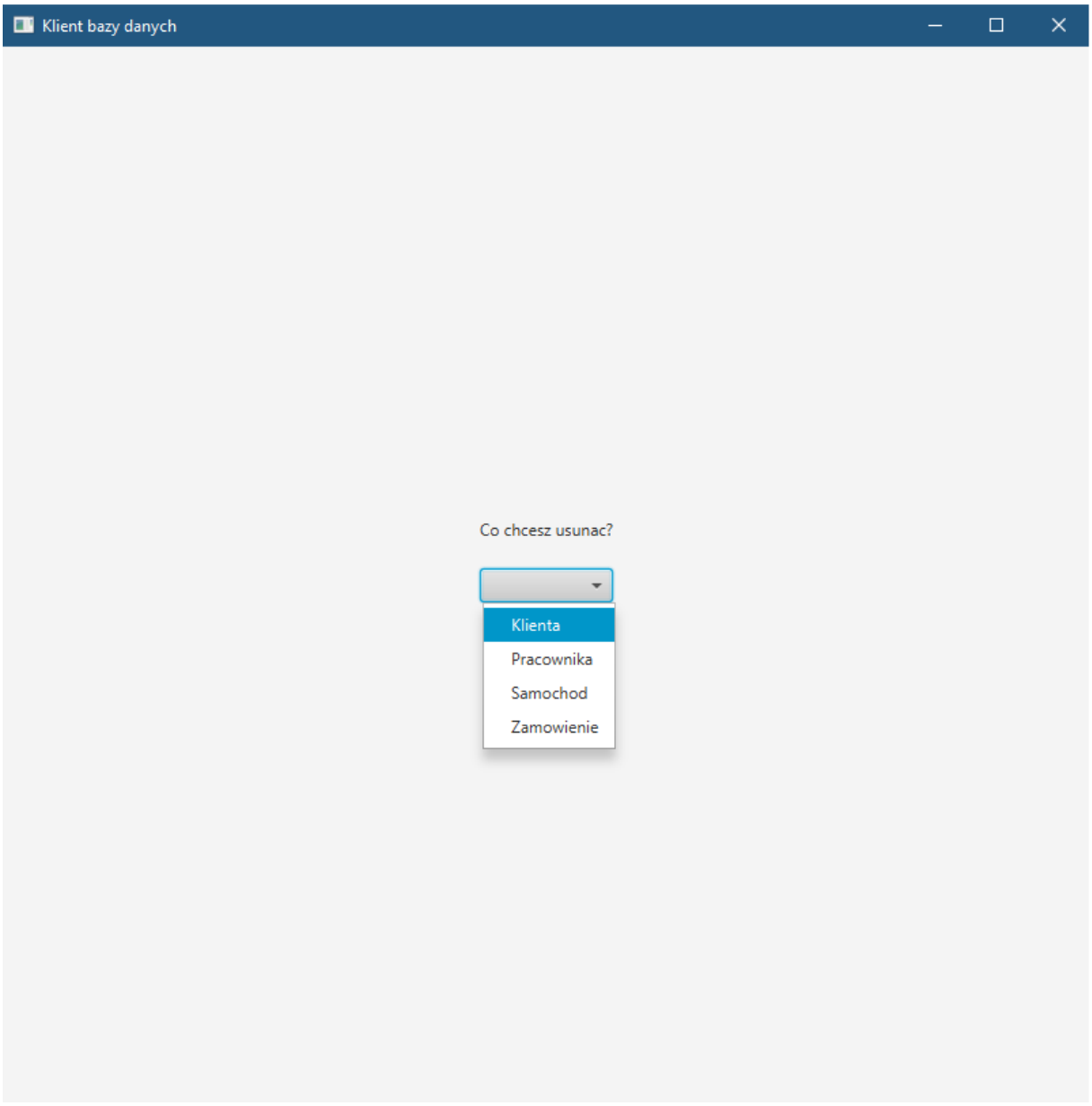
Po wybraniu opcji „Dodawania rekordów” wyświetla nam się nowa scena, w której użytkownik zostaje zapytany o rodzaj informacji jaką chce dodać do bazy.

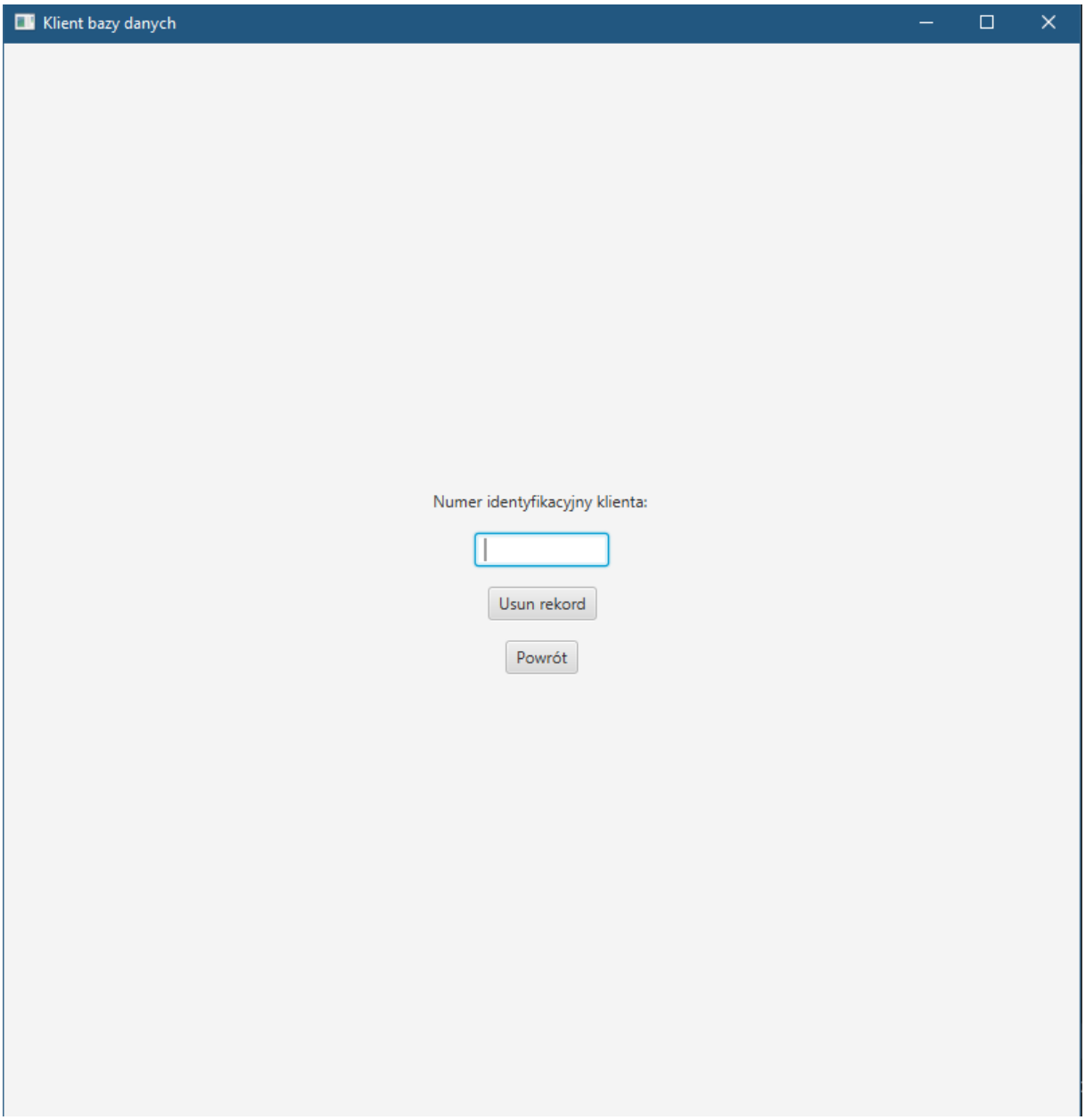




Po wypełnieniu formularza, jeżeli wszystkie dane są poprawne*, to znaczy nie występują żadne powtórzenia w polach identyfikacyjnych (np. Tabela zamówienia) zostaje dodany nowy rekord z podanymi przez użytkownika danymi. Po dodaniu rekordu użytkownik ma możliwość powrotu do poprzedniej sceny czyli wyboru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





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.

getDBConnection() - 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 sie z bazą danych.

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

--App.java

Klasa odpowiadająca za okienko głowne 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 głowna aplikacji odpowiedzialna za zdefiniowanie nowego okna I nowych scen.

public void start() – funkcja określająca scene startową.

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

Label labelek = new Label("Co chcesz wyswietlic?"); - wyświetla napis .

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

Button backButttt = new Button("Powrot"); - metoda ta towrzy nowy przycisko o podanej w parametrze nazwie.

AlertBox.display – wyświetla komunikat błędu

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

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

backOptio.setOnAction(e->changeScene(window,scene1)); - metoda odpowiedzialna za cofniecie sie do sceny pocztątkowej

window.show();
- metoda odpowiedzialna za wyswietlenia sie okienka
window.setTitle –metoda opowiedzialna za ustawienie tytułu okna

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

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

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

public static void main()- funkcja głowna aplikacji

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

-- Praca przy kodzie aplikacji: Rudzki Marcin

-- 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