Implementacja

# Kod po stronie serwera (procedury wbudowane i triggery)

## **Domyślne contratinty:**

ALTER TABLE order\_t MODIFY order\_date DEFAULT SYSDATE;

ALTER TABLE service MODIFY begin\_date DEFAULT SYSDATE;

ALTER TABLE service MODIFY cost DEFAULT 0;

## **Sekwencje i triggery automatycznie inkrementujące ID:**

CREATE SEQUENCE vehicles\_sequence

START WITH 1

INCREMENT BY 1

CACHE 100;

CREATE OR REPLACE TRIGGER vehicles\_sequence\_trigger

BEFORE INSERT ON vehicle

FOR EACH ROW

BEGIN

:new.vehicle\_id := vehicles\_sequence.nextval;

END;

CREATE SEQUENCE clients\_sequence

START WITH 1

INCREMENT BY 1

CACHE 100;

CREATE OR REPLACE TRIGGER clients\_sequence\_trigger

BEFORE INSERT ON client

FOR EACH ROW

BEGIN

:new.client\_id := clients\_sequence.nextval;

END;

CREATE SEQUENCE orders\_sequence

START WITH 1

INCREMENT BY 1

CACHE 100;

CREATE OR REPLACE TRIGGER orders\_sequence\_trigger

BEFORE INSERT ON order\_t

FOR EACH ROW

BEGIN

:new.order\_id := orders\_sequence.nextval;

END;

CREATE SEQUENCE vversion\_sequence

START WITH 1

INCREMENT BY 1

CACHE 100;

CREATE OR REPLACE TRIGGER vversion\_sequence\_trigger

BEFORE INSERT ON vehicle\_version

FOR EACH ROW

BEGIN

:new.vversion\_id := vversion\_sequence.nextval;

END;

## **Zmiana kosztu realizacji usługi, zwiększa/zmniejsza koszt zamówienia:**

CREATE OR REPLACE TRIGGER update\_order\_cost\_by\_service

before UPDATE OF cost ON service

FOR EACH ROW

BEGIN

if (:new.cost <> :old.cost) then

update order\_t set value = value + :new.cost - :old.cost where order\_id = :old.order\_id;

end if;

END;

/

## **Zliczanie sumy kosztów zamówień dla klienta o podanym id:**

create or replace function get\_sum\_of\_client\_orders (i\_client\_id in client.client\_id%type) return order\_t.value%type is

val order\_t.value%type;

begin

select sum(value) into val from order\_t where client\_id = i\_client\_id;

if val is null then

val := 0;

end if;

return val;

end;

/

/\*\* Example \*/

set serveroutput on;

begin

dbms\_output.put\_line('Sum of orders: ' || get\_sum\_of\_client\_orders(0));

end;

/

## **Procedura licząca i wypisująca klientów którzy nie złożyli zamówienia:**

create or replace procedure get\_clients\_without\_orders is

amount NUMBER(5,0);

cursor cur is select \* from client where client\_id not in (select distinct client\_id from order\_t);

os client%rowtype;

begin

amount := 0;

open cur;

loop

fetch cur into os;

exit when cur%notfound;

dbms\_output.put\_line(os.client\_id || ': ' || os.first\_name || ' ' || os.last\_name || ' ' || os.name);

amount := amount + 1;

end loop;

close cur;

dbms\_output.put\_line(' Total amount of clients without orders: ' || amount);

end;

/

/\*\* Example \*/

set serveroutput on;

begin

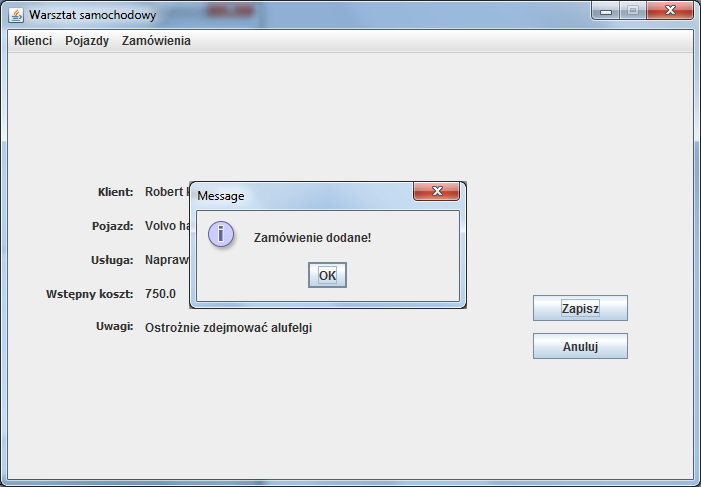
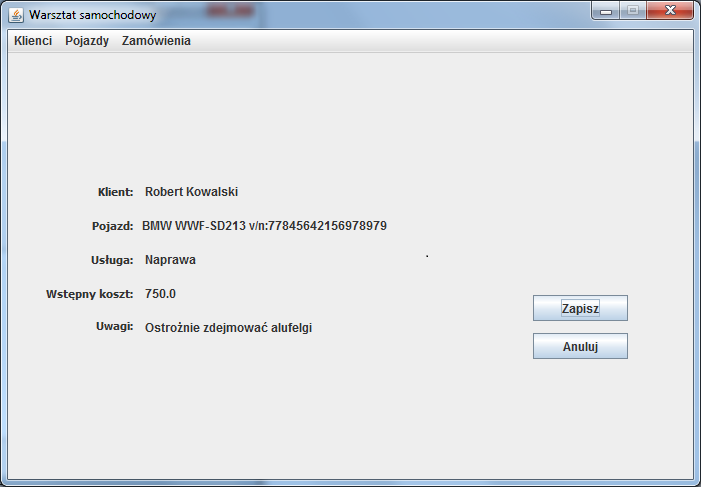
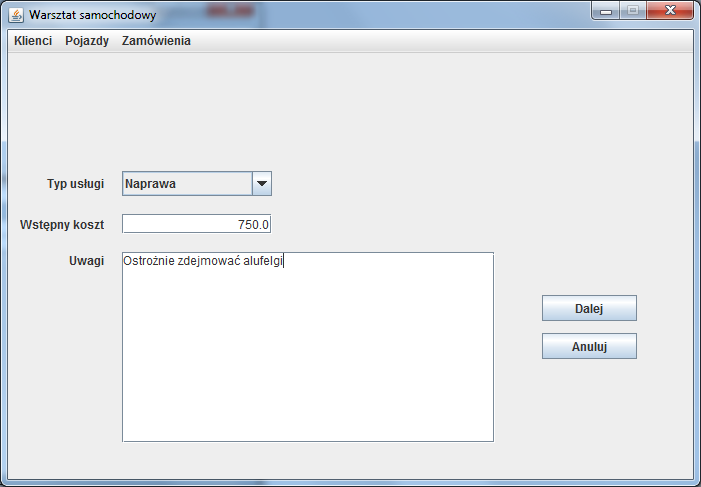
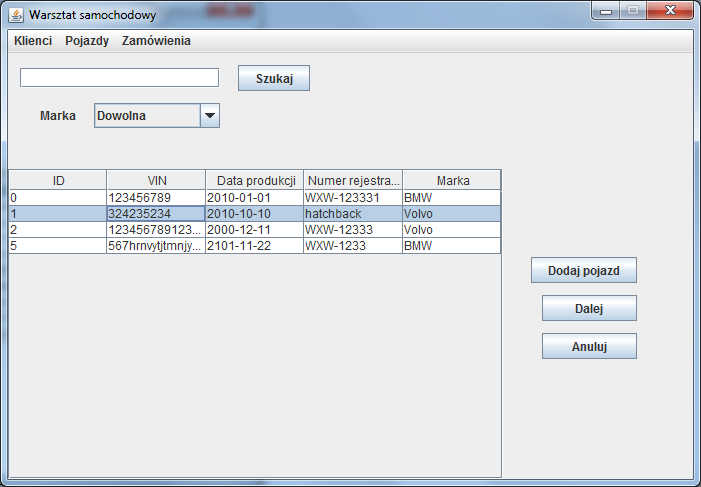
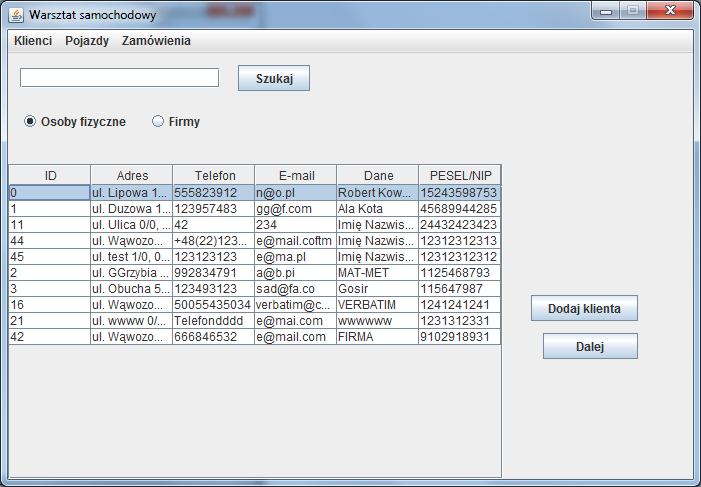
get\_clients\_without\_orders();

end;

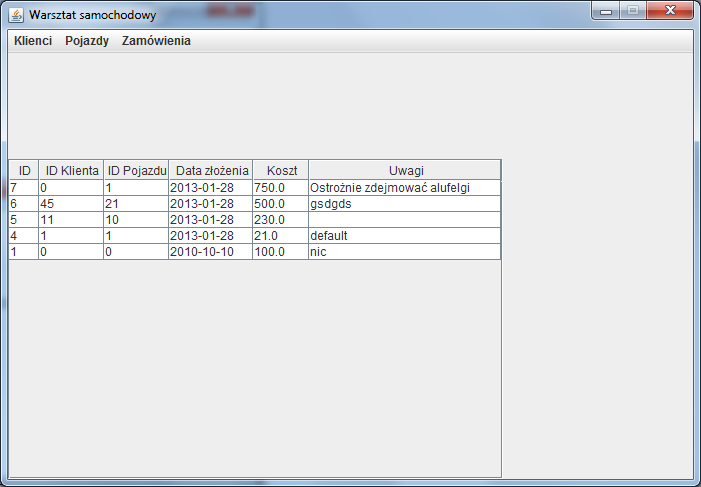
/

# Screenshot'y z aplikacji

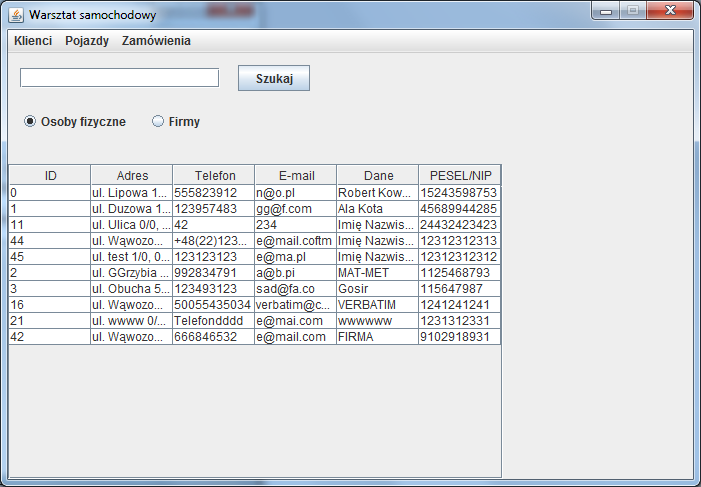
Dodawanie zamówienia (kolejne ekrany transakcji):



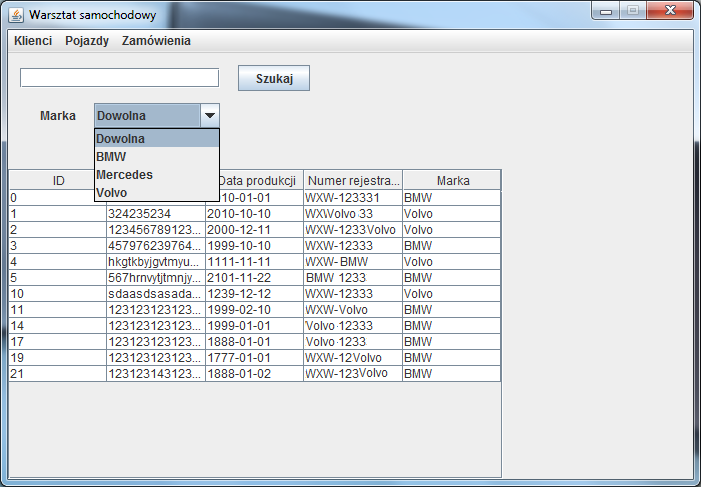
Wyświetlenie zamówień na zakończenie:



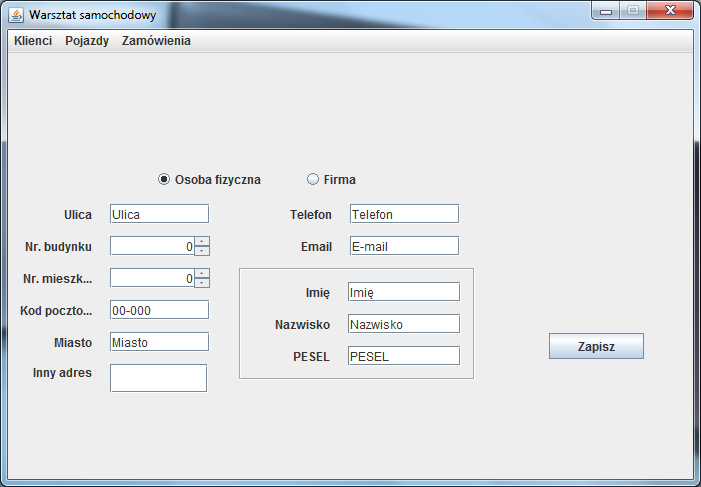
Wyświetlenie klientów:



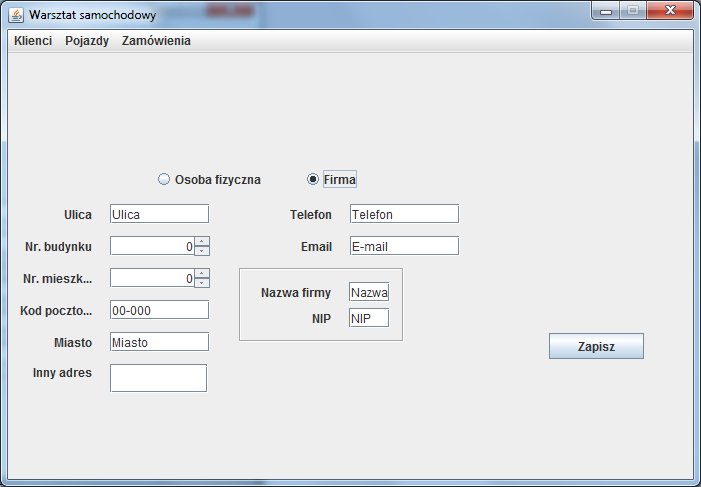
Wyświetlenie pojazdów:



Dodawanie klienta (osoby):



Dodawanie klienta (firmy):



Dodawanie pojazdu:

