Dokumentacja projektu wykonywanego w ramach zajęć
BAZY DANYCH I
System rezerwacji lotów samolotowych

I. Projekt koncepcji, założenia

1.Zdefiniowanie tematu projektu

System rezerwacji lotów samolotowych umożliwia rezerwację istniejącego połączenia pomiędzy lotniskami. Gdy użytkownik jest zalogowany może zarezerwować lot dla siebie, oraz innych współpodróżnych. System pokazuje tylko aktualne loty. Rezerwacji można dokonać tylko, gdy są dostępne miejsca, system wyświetla ilość wolnych miejsc. Użytkownik wprowadza dane osobowe podróżujących. Odprawić się można 2 dni przed odlotem; bezpłatnie, oraz miejsce jest przydzielane losowo. By, odprawić się z możliwością miejsca siedzącego, trzeba zapłacić 20 zł (można to zrobić w dowolnym czasie). Użytkownik może wyświetlić historię swoich rezerwacji.
Z poziomu administratora można dodać linie, samolot, lot, oraz wyświetlić wszystkich pasażerów wybranego lotu.

2.Analiza wymagań użytkownika

Użytkownik rejestruje/loguje się do systemu. Wyszukuje interesujące go połączenie, wprowadza dane osobowe, rezerwuje, a potem odprawia. Admin wprowadza nową linie, lot, samolot, cenę.

3.Zaprojektowanie funkcji:

- szukaj - wyświetla wszystkie loty razem ze szczegółami
- mojerezerwacje - wyświetla wszystkie zarezerwowane loty danego użytkownika
- mojerezerwacjev2 - wyświetla wszystkie zarezerwowane loty danego użytkownika wraz z danymi osobowymi pasażera, oraz miejsce jeśli się odprawił
- dostepne miejsca - wyświetla wszystkie niezajęta miejsca danego lotu
- ilosc_wolnych - wyświetla ilość wolnych miejsc w samolocie
- dodajLinieSamolot - do tabeli linia wstawia nową linie,do tabeli samolot nowy samolot z ilością miejsc, tabele miejsce uzupełnia odpowiednimi lokalizacjami miejsc w nowym samolocie.
- dodajLot - dodaje nowy lot
- usunLot - usuwa lot
- usunRezerwacje – usuwa rezerwacje

II. Projekt diagramów (konceptualny)

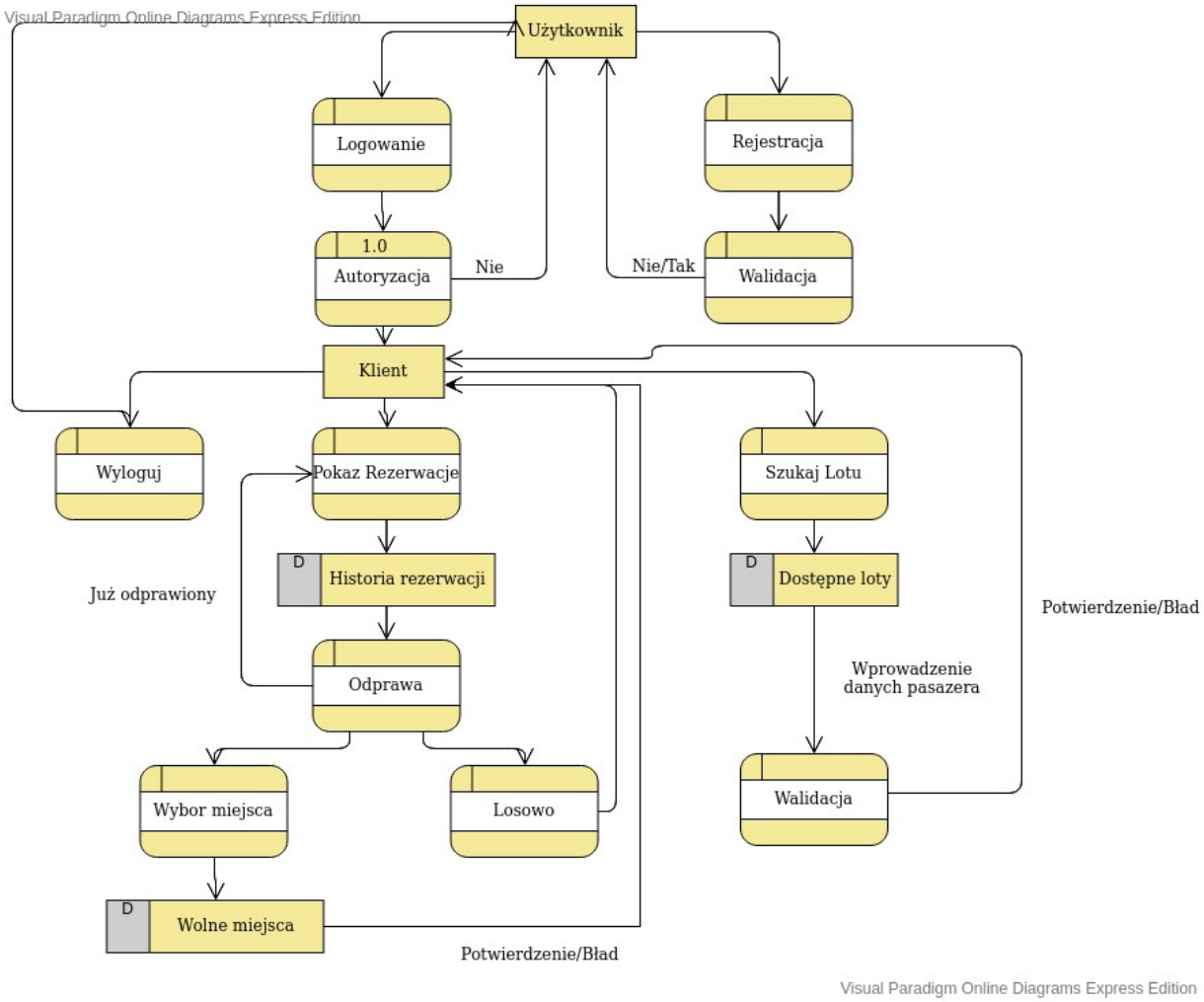
4.Budowa i analiza diagramu przepływu danych (DFD):

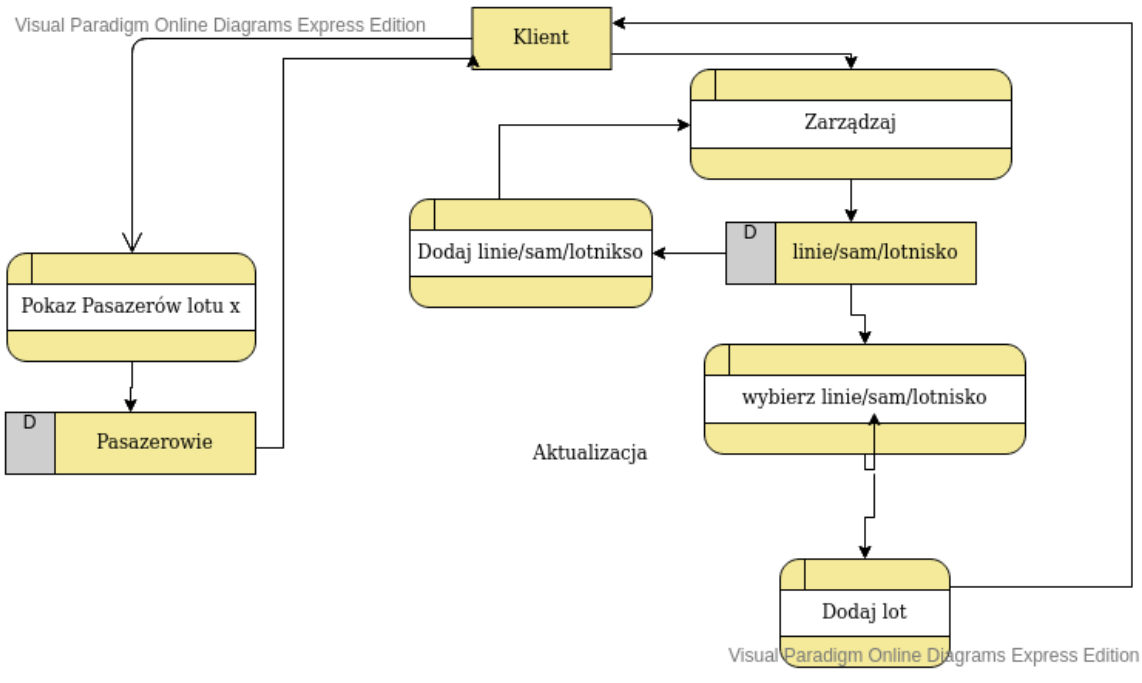
Użytkownik wprowadza dane, jeśli istnieje w tabeli “uzytkownik”, ma dostęp do panelu głównego. Jeśli użytkownika nie ma w bazie może się zarejestrować.
W panelu głównym:
Szukanie lotu: Użytkownik wybiera dostępne parametry lotu i proces “szukaj” wprowadza dostępne loty do składnic. Użytkownik podaje interesujący go lot. Zostaje pobrany id_lot. Po wprowadzeniu danych trafiają one do Procesu walidacji. Rezerwacja jest ukończona. Informacja rezerwacji trafia do składnicy Historia rezerwacji.

Sposoby odprawy:
- Wybór miejsca: powoduje wstawienie dostępnych wolnych miejsc do składnicy “Wolne miejsca”. Po wyborze miejsca, parametr identyfikujący miejsce zostaje wstawiony do składnicy Historia rezerwacji.
- Losowo: Miejsce zostaje przydzielone losowo i parametr idetyfikujący miejsce zostaje wstawiony do składnicy Historia rezerwacji. ‘

Panel admina:
Proces Zażądaj wprowadza dane do składnicy dostępnych “linie/sam/lotnisko”. Admin Wybiera ze składnicy parametry, w procesie “Dodaj lot” dodaje inne parametry takie jak data/godz/cena i aktualizowana jest baza danych.
Ze składnicy linie/sam/lotnisko dane mogą trafić do procesy “Dodaj, linie/sam/lot”. Gdzie są porównywane z nowo wprowadzonymi parametrami(np. czy nie występuje taka Linia).

Diagram Przepływu danych





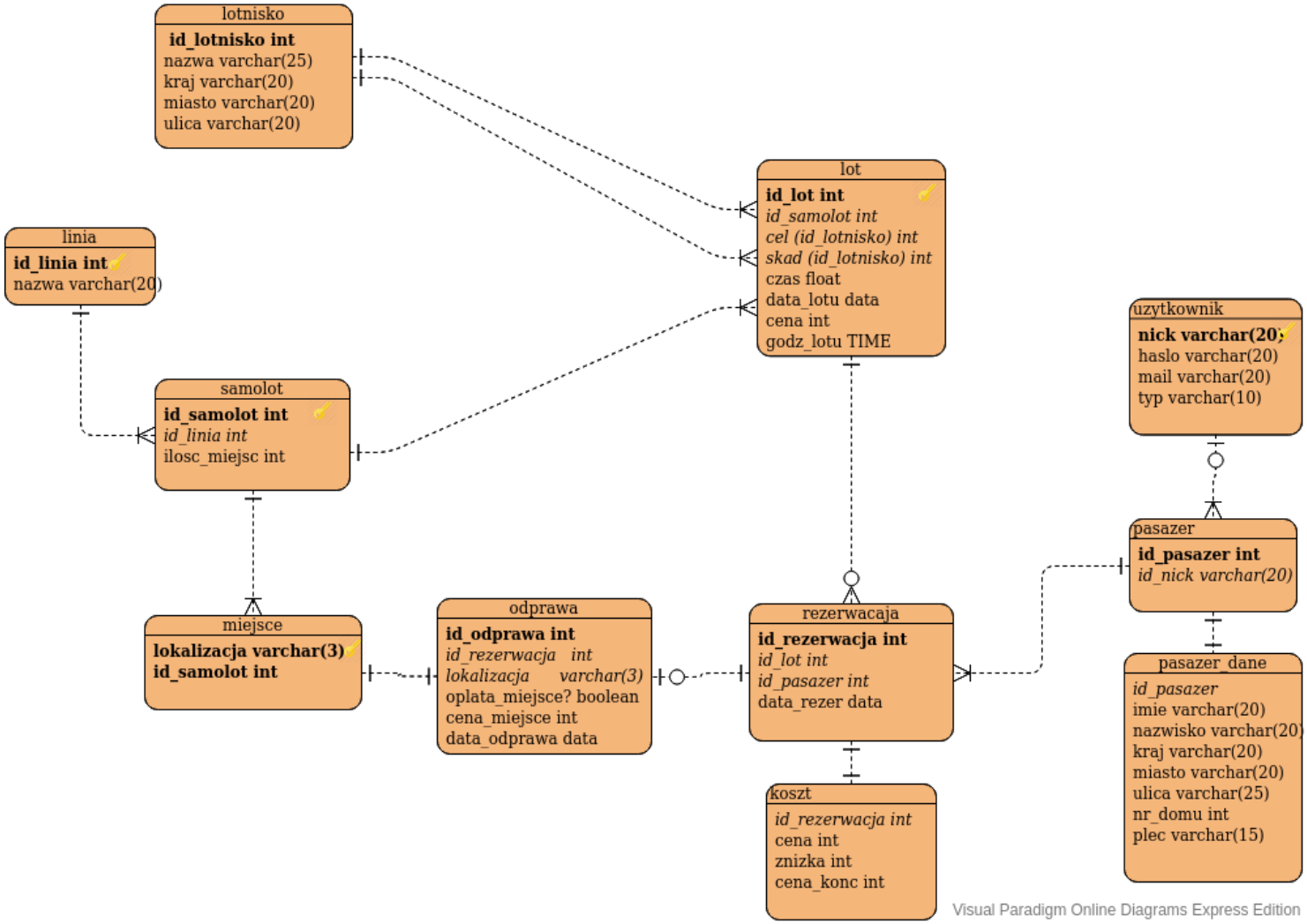
5.Zdefiniowanie encji (obiektów) oraz ich atrybutów.

lotnisko: id_lotnisko, nazwa, kraj, miasto, ulica
linia: id_linia, nazwa
samolot: id_samolot, id_linia, ilosc_miejsc
miejsce: lokalizacja, id_samolot
lot: id_lot, id_samolot, cel, skąd, czas, data_lotu,godz_lotu,cena
odprawa: id_odprawa, id_rezerwacja, lokalizacja, cena_miejsce, data_odprawa
rezerwacja: id_rezerwacji, id_lot, id_pasazer, data_rezerwacji, znizka
uzytkownik: nick, haslo, mail, typ
pasazer: id_pasazer, id_nick
pasazer_dane: id_pasazer, imie, nazwisko, kraj, miasto, ulica, nr_domu, plec

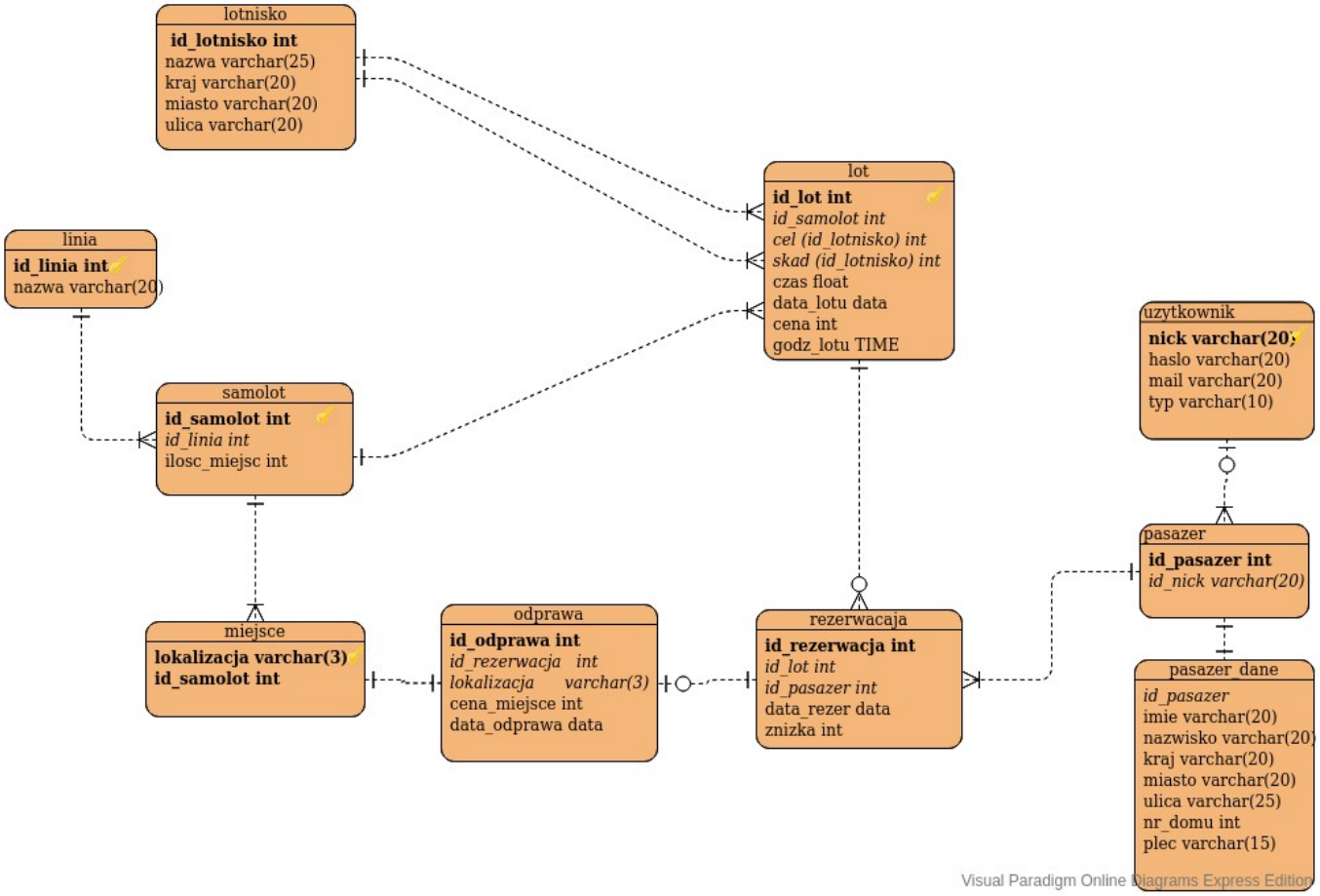
id_lotnisko INTEGER > 0
nazwa(lotniska) VARCHAR,
kraj VARCHAR,
miasto VARCHAR,,
ulica VARCHAR,
id_linia INTEGER > 0
nazwa VARCHAR,
id_samolot INTEGER > 0
ilosc_miejsc INTEGER > 0
lokalizacja VARCHAR (3)
id_lot INTEGER > 0
cel(id_lotniska) INTEGER > 0
skad(id_lotnisko) INTEGER > 0
czas TIME godzina, minuta, sekunda
data_lotu DATA rok,miesiąc,dzień
id_odprawa INTEGER > 0
id_rezerwacja INTEGER > 0
cena_miejsce INTEGER = 20, lub 0
data_odprawa DATA rok,miesiąc,dzień
data_rezerwacji DATA rok,miesiąc,dzień
cena INTEGER > 0
znizka INTEGER E(0,100)
nick VARCHAR,
haslo VARCHAR,
mail VARCHAR,
typ user/admin
id_pasazer INTEGER > 0
imię VARCHAR,
nazwisko VARCHAR,
nr_domu INTEGER > 0
plec Kobieta/Męszczyzna

6.Zaprojektowanie relacji pomiędzy encjami:

ERD przed normalizacją:



ERD po normalizacji:



III. Projekt logiczny

7.Projektowanie tabel, kluczy, indeksów: (załącznik create.sql)

-Tworzenie tabel:

```
create table linia(id_linia int,nazwa varchar(20));
create table lotnisko(id_lotnisko int,nazwa varchar(20),kraj varchar(20),miasto
varchar(20),ulica varchar(20));
create table samolot(id_samolot int,id_linia int,ilosc_miejsc int);
create table miejsce(lokalizacja varchar(3),id_samolot int);
create table odprawa(id_odprawa int,cena_miejsce int, data_odprawa
date,id_rezerwacja int,lokalizacja varchar(3) );
create table rezerwacja(id_rezerwacja int,id_lot int, id_pasazer int, data_rezerw
```

```
date,znizka int);
create table pasazer(id_pasazer int,nick varchar(20));
create table pasazer_dane(id_pasazer int,imie varchar(20),nazwisko varchar(20),kraj
varchar(20),miasto varchar(20),ulica varchar(20),nr_domu int,plec varchar(15));
create table uzytkownik(nick varchar(20),haslo varchar(20),mail varchar(20),typ
varchar(10));
create table lot(id_lot int,id_samolot int,cel int,skad int,czas float,data_lotu
date,godz_odlotu time,cena int);
```

-Dodanie kluczy głównych

```
alter table linia ADD PRIMARY KEY (id_linia);
alter table lotnisko ADD PRIMARY KEY(id_lotnisko);
alter table samolot ADD PRIMARY KEY(id_samolot);
alter table miejsce ADD PRIMARY KEY(lokalizacja,id_samolot);
alter table odprawa ADD PRIMARY KEY(id_odprawa);
alter table rezerwacja ADD PRIMARY KEY(id_rezerwacja);;
alter table pasazer add PRIMARY KEY(id_pasazer);
alter  table uzytkownik add PRIMARY KEY(nick);
alter table lot add PRIMARY KEY(id_lot);
```

-Dodanie kluczy obcych

```
alter table samolot add FOREIGN KEY(id_linia) references linia(id_linia);
alter table miejsce ADD FOREIGN KEY(id_samolot) references
samolot(id_samolot);--
alter table rezerwacja ADD FOREIGN KEY(id_lot) references lot(id_lot);
alter table rezerwacja add FOREIGN KEY(id_pasazer) references
pasazer(id_pasazer);
alter table pasazer add FOREIGN KEY(nick) references uzytkownik(nick);
alter table pasazer_dane add FOREIGN KEY(id_pasazer) references
pasazer(id_pasazer);
alter table lot add FOREIGN KEY(cel) references lotnisko(id_lotnisko);
alter table lot add FOREIGN KEY(skad) references lotnisko(id_lotnisko);
alter table lot add FOREIGN KEY(id_samolot) references samolot(id_samolot);
alter table odprawa add FOREIGN KEY(id_rezerwacja) references
rezerwacja(id_rezerwacja);
```

-Uzupełnianie tabel

```
insert into linia(id_linia,nazwa) values
(1,'Lot'),
(2,'Ryanair');
```

```
insert into samolot(id_samolot,id_linia,ilosc_miejsc) values
(1,1,5),
(2,1,5),
(3,2,2),
(4,2,2);

insert into miejsce(lokalizacja,id_samolot) values
('1A',1),
('1B',1),
('1C',1),
('1D',1),
('1E',1),

('1A',2),
('1B',2),
('1C',2),
('1D',2),
('1E',2),

('1A',3),
('1B',3),

('1A',4),
('1B',4);

insert into lotnisko(id_lotnisko,nazwa,kraj,miasto,ulica) values
(1,'Krakow-Balice','Polska','Krakow','Medweckiego'),
(2,'Chopina','Polska','Warszawa','Zwirki i Wigory'),
(3,'Barcelona-Girona','Hiszpania','Girona','Vilobi dOnyar');

insert into lot(id_lot,id_samolot ,cel ,skad ,czas ,data_lotu ,godz_odlotu ,cena )
VALUES
(1,1,1,3,1.5,'07-02-2020','11:00:00',200);

insert into uzytkownik(nick,haslo,mail,typ) VALUES ('admin','admin','admin','admin');
```

Tabela lot: id_lot identyfikuje jednoznacznie każdy lot, posiada 2 klucze obce które wskazują na 2 lotniska (miejsca wylotu i docelowe). Są połączone ze sobą N:1 ponieważ dany lot ma tylko jedno miejsce wylotu i przylotu. Taka sama relacja jest z encja samolot, gdyż lot może mieć tylko 1 samolot.

Lotnisko - posiada atrybuty varchar opisujące encje. Jest połączona z lotem 1:N - z danego lotniska może wylatywać jak i przylatywać wiele samolotów.

Linia: relacja z samolot 1:N - linia może posiadać wiele samolotów

Samolot: Może należeć tylko do jednej linii(N:1),każdy samolot ma określoną ilość_miejsc i każde ma inna lokalizacje(1:N). Samolot, może należeć do kilku lotów(1:N) np. różniących się datą.

Miejsce: Klucz główny składa się z lokalizacji np. (1A), oraz samolotu: ponieważ każdy samolot oznacza lokalizacje miejsca tak samo(1A,1B,1C,2A itp) by zidentyfikować jednoznacznie miejsce musimy mieć tez samolot w którym jest to miejsce.

rezerwacja - identyfikuje ja id_rezerwacja, musi mieć tylko jednego pasażera i dotyczyć tylko 1 lotu

koszt - w zależności od okresu pomiędzy datą lotu a, datą rezerwacji ustalana jest zniżka od ceny bazowej zawartej w encji lot

odprawa - odprawa musi mieć tylko jedną rezerwacje, ale rezerwacja nie musi mieć odprawy. Encja posiada informacje,czy ktoś kupił możliwość wyboru miejsca, czy też system przydzielił losowo “oplata_miejsce”.

uzytkownik - kluczem głównym jest nick, posiada atrybut “typ”, który może być wartością typu user, lub admin. Użytkownik może być pasażerem, lub rezerwować lot dla inny.

pasazer - pasazer nie musi być użytkownikiem, dane osobowe pasażera są w innej encji “pasazer_dane”, żeby nie trzeba było ich za każdym razem wyszukiwać.

8.Analiza zależności funkcyjnych i normalizacja tabel:

Pierwsza postać normalna tabel:
W każdej tabeli jest jednoznacznie definiujący encję klucz główny. tabela miejsce posiada klucz główny złożony z dwóch atrybutów. Wszystkie dane są atomowe. Można się jedynie zastanowić, czy lokalizacji (1A) nie rozdzielić na nr. rzędu (1), oraz miejsce w rzędzie (A).

Druga postać normalna:
każda tabela przechowuje dane dotyczące tylko konkretnej klasy obiektów.->(Z rezerwacji została wyciągnięta klasa dotycząca kosztów.) ,oraz z pasażera wyciągnięto dane osobowe do osobnej tabeli.

Trzecia postać normalna 3NF
Z tabeli odprawa usunąłem atrybut boolean oplata_miejsce, który mówił, czy

pasażer odprawił się wybierając miejsce, czy skorzystał z losowości. W encji odprawa istnieje atrybut “cena” za miejsce Jeśli jest większa od zera to znaczy, że pasażer zapłacił.
Z tabeli koszt usunąłem atrybut cena_końcowa, ponieważ można go wyliczyć z “zniżka”, oraz “cena”. Po normalizacji zauważyłem, że tabela koszt nie ma większego sensu i przeniosłem atrybut zniżka do encji rezerwacja.
Teraz każda kolumna informacyjna nie należąca do klucza nie zależy od innej kolumny informacyjne, oraz dane nie są redundantne.

9.Zaprojektowanie operacji na danych (załącznik funview.sql)

*/*Funckcja zwraca wszystkie dostępne miejsca (lokalizacje)*
w samolocie, który należy do zarezewowanego przez pasażera lotu/*

```
CREATE OR REPLACE FUNCTION dostepneMiejsca(id_rezerw int)
RETURNS TABLE(lok VARCHAR) AS
$$
BEGIN
    RETURN QUERY
   SELECT lokalizacja FROM miejsce WHERE id_samolot =
   (SELECT lot.id_samolot FROM lot JOIN rezerwacja r
    USING(id_lot) WHERE r.id_rezerwacja =id_rezerw) EXCEPT
    SELECT lokalizacja FROM odprawa JOIN rezerwacja USING(id_rezerwacja)
JOIN lot USING(id_lot) WHERE id_samolot=(SELECT lot.id_samolot FROM lot JOIN
rezerwacja r USING(id_lot) WHERE r.id_rezerwacja =id_rezerw);
END;
$$

LANGUAGE 'plpgsql';
```

Funckcja zwraca ilość wolnych miejscw samolocie, który należy do wybranego lotu

```
CREATE OR REPLACE FUNCTION ilosc_wolnych(lot_id int)
RETURNS INTEGER AS
$$
DECLARE
  _ilosc_rezerw INTEGER := 0;
  _ilosc INTEGER := 0;
BEGIN
       _ilosc_rezerw := (SELECT count(*) FROM rezerwacja WHERE id_lot=lot_id);
        _ilosc:= (SELECT ilosc_miejsc FROM samolot s JOIN lot USING(id_samolot)
          WHERE id_lot=lot_id);

       RETURN _ilosc - _ilosc_rezerw;
```

```
END;
$$
LANGUAGE 'plpgsql';
```

*/*Funckcja dodaje nową Linie i dodaje do niej samolotem o określonej ilości miejsc. Funkcja usupełnia również lokalizacje miejsc w samolocie (1A,1B itp)*/*

```
CREATE OR REPLACE FUNCTION dodajLinieSamolot(nowaNazwa
VARCHAR(20),nowaIlosc int)
RETURNS INTEGER AS
$$
DECLARE
  _idlinia INTEGER := 0;
  _idsamo INTEGER := 0;
  j INTEGER := 1;
  _temp VARCHAR(1) :='A';
  liniaGood INTEGER := 0;

BEGIN
    _idlinia := (SELECT COUNT(*) FROM linia);
    INSERT INTO linia(id_linia,nazwa) VALUES (_idlinia+1,nowaNazwa)
RETURNING id_linia INTO liniaGood;
    IF liniaGood > 0 THEN
        liniaGood:=1;
    END IF;

    _idsamo := (SELECT COUNT(*) FROM samolot);
    INSERT INTO samolot(id_samolot,id_linia,ilosc_miejsc) VALUES
(_idsamo+1,_idlinia+1,nowaIlosc);

    FOR i IN 1..nowaIlosc LOOP

        IF MOD(i,4) = 1 THEN
            _temp :='A';
        ELSIF MOD(i,4) = 2 THEN
            _temp :='B';
        ELSIF MOD(i,4) = 3 THEN
            _temp :='C';
        ELSIF MOD(i,4) = 0 THEN
            _temp :='D';
        END IF;
```

```
INSERT INTO miejsce(lokalizacja,id_samolot) VALUES (CONCAT(CAST(j AS
VARCHAR(1)),_temp),_idsamo+1);

        IF MOD(i,4) = 0 THEN
            j:=j+1;
        END IF;

    END LOOP;

RETURN liniaGOOD;

END;
$$
LANGUAGE 'plpgsql';
```

*/*Funckcja dodaje nowy lot o zadanych parametrach*/*

```
CREATE OR REPLACE FUNCTION dodajLot(NazwaLini VARCHAR(20),id_sam
int,newSkad VARCHAR(20),newDokad VARCHAR(20),newData DATE,godz
time,newCzas FLOAT,newCena int)
RETURNS void AS
$$
DECLARE
  _idlot INTEGER := 0;
  _idskad INTEGER := 0;
  _iddokad INTEGER := 0;

BEGIN
     _iddokad := (SELECT id_lotnisko FROM lotnisko WHERE  nazwa=newDokad);
     _idskad := (SELECT id_lotnisko FROM lotnisko WHERE  nazwa=newSkad);
    _idlot := (SELECT COUNT(*) FROM lot);


    INSERT INTO lot(id_lot , id_samolot ,cel , skad , czas , data_lotu  ,
godz_odlotu,cena)VALUES
                (_idlot+1,id_sam,_iddokad,_idskad,newCzas,newData,godz,newCena);

END;
$$
LANGUAGE 'plpgsql';
```

*/*Widok szukaj pozwala wyświetlić wszystkie loty jakie są w bazie*/*

```
CREATE VIEW szukaj AS
SELECT DISTINCT lo1.nazwa AS Skad, lo1.miasto as miasto_z,lo2.nazwa as Do,
lo2.miasto as miasto_do,lot.data_lotu,lot.godz_odlotu, linia.nazwa as Linia, lot.id_lot,
lot.cena
   FROM lotnisko lo1
   JOIN lot ON lo1.id_lotnisko=lot.skad
   JOIN lotnisko lo2 ON lo2.id_lotnisko=lot.cel
   JOIN samolot USING(id_samolot)
   JOIN linia USING(id_linia)
   WHERE data_lotu>=now();
```

*/*Widok mojeRezerw pozwala wyświetlić wszystkie rezerwacje jakie są w bazie*/*

```
CREATE VIEW mojeRezerw AS
   SELECT DISTINCT r.id_rezerwacja,p.nick, pd.imie, pd.nazwisko,l1.nazwa AS
z,l2.nazwa AS do,lot.data_lotu,lot.godz_odlotu, CASE WHEN true THEN
lot.cena*(100-r.znizka)*0.01 END as cena_konc ,linia.nazwa FROM lot JOIN lotnisko
l1 ON lot.skad=l1.id_lotnisko JOIN lotnisko l2 ON lot.cel=l2.id_lotnisko JOIN
rezerwacja r USING(id_lot) JOIN pasazer p USING(id_pasazer) JOIN pasazer_dane
pd USING(id_pasazer)  JOIN samolot s USING(id_samolot) JOIN linia
USING(id_linia)  ;
```

*/*Widok mojeRezerwv2 pozwala wyświetlić wszystkie rezerwacje jakie są w bazie, ale z większą ilością informacji, zawiera również info czy pasażer się odprawił, czy nie*/*

```
CREATE VIEW mojeRezerwv2 AS
   SELECT DISTINCT r.id_rezerwacja,p.nick, pd.imie,
pd.nazwisko,pd.kraj,pd.miasto,lot.id_lot,l1.nazwa AS z,l2.nazwa AS
do,lot.data_lotu,lot.godz_odlotu, CASE WHEN true THEN lot.cena*(100-
r.znizka)*0.01 END as cena_konc,linia.nazwa,odprawa.lokalizacja FROM lot JOIN
lotnisko l1 ON lot.skad=l1.id_lotnisko LEFT JOIN lotnisko l2 ON lot.cel=l2.id_lotnisko
LEFT JOIN rezerwacja r USING(id_lot) JOIN pasazer p USING(id_pasazer) JOIN
pasazer_dane pd USING(id_pasazer)  JOIN samolot s USING(id_samolot) JOIN
linia USING(id_linia) LEFT JOIN odprawa USING(id_rezerwacja);
```

```
/*Funkcja usuwa kolejno odprawy dla danego lotu, rezewacje, oraz lot*/

CREATE OR REPLACE FUNCTION usun_lot(id_l int)
RETURNS INTEGER AS
$$
DECLARE

kursor1 CURSOR FOR SELECT * FROM odprawa JOIN rezerwacja r
USING(id_rezerwacja) WHERE id_lot=id_l;
kursor2 CURSOR FOR SELECT * FROM rezerwacja  WHERE id_lot=id_l;

odpr RECORD;
rezerw RECORD;

flag INTEGER:=0;

BEGIN
   OPEN kursor1;
   LOOP
      FETCH kursor1 INTO odpr;
      EXIT WHEN NOT FOUND;
      DELETE FROM odprawa WHERE odprawa.id_odprawa=odpr.id_odprawa;
   END LOOP;
   CLOSE kursor1;

   OPEN kursor2;
   LOOP
      FETCH kursor2 INTO rezerw;
      EXIT WHEN NOT FOUND;
      DELETE FROM rezerwacja WHERE
rezerwacja.id_rezerwacja=rezerw.id_rezerwacja;
   END LOOP;

   CLOSE kursor2;

   DELETE FROM lot WHERE id_lot=id_l RETURNING id_lot INTO flag;


   RETURN flag;
END;
$$

LANGUAGE 'plpgsql';
```

*/*Funkcja usuwa kolejno odprawy dla danej rezerwacji, oraz rezerwacje*/*

```
CREATE OR REPLACE FUNCTION usun_rezerwacje(id_r int)
RETURNS INTEGER AS
$$
DECLARE

flag INTEGER:=0;

BEGIN

   DELETE FROM odprawa WHERE odprawa.id_rezerwacja=id_r;
   DELETE FROM rezerwacja WHERE id_rezerwacja=id_r RETURNING
id_rezerwacja INTO flag;

   RETURN flag;
END;
$$

LANGUAGE 'plpgsql';
```

IV. Projekt funkcjonalny

10.Interfejsy do prezentacji, edycji i obsługi danych: zdefiniowanie struktury poszczególnych formularzy do wprowadzania danych oraz powiązań między formularzami.

- formularz rejestracji - nick, hasło, powtórz hasło, mail
  Rejestracja jest powiązana z logowaniem: jeśli użytkownika nie ma, może się zarejestrować,

- formularz logowania- nick, hasło
  Jeśli użytkownik jest w bazie może się zalogować podając nick i hasło

- formularz szukaj lotu - skąd, dokąd, linia
  Użytkownik musi wybrać obowiązkowo skąd chce lecieć z listy dostępnych lotnisk

- formularz pokaż pasażerów – id_lot
  Admin wybiera z listy dostępnych lotów lot, by wyświetlić pasażerów

- formularz miejsce – lokalizacja
  wybór lokalizacji z listy dostępnych lokalizacji

- formularz odprawa – id_rezerwacji
  wybór rezerwacji z listy dostępnych rezerwacji, którą chcemy odprawić

- formularz dane pasażera - imie, nazwisko, kraj, miasto, ulica, nr_domu, płeć
  wprowadzenie danych osobowych

- formularz dodaj lot - nazwa linii, id samolot, skad,dokad, data, godzina, czas trwania, cena
  wprowadzenie danych nowego lotu;
  nazwa linii, id samolot, skad,dokad,data – wybieramy z listy;
  godzina, czas trwania, cena – wprowadzamy ręcznie

- formularz dodaj lotnisko - nazwa, kraj,miasto, ulica
  wprowadzamy dane nowego lostniska

- formularz dodaj linie - nazwa, ilosc_miejsc(w nowym samolocie)
  wprowadzamy dane nowej linii

- formularz rezerwuj – id_lot
  wybieramy z listy lot który chcemy zarezerwować

11.Wizualizacja danych: określenie formy i struktury raportów które będą generowane przez bazę danych.

Dane wizualizowane w tabeli:
- dostępne loty
- lista rezerwacji
- lista pasażerów, wraz z danymi

Dane wizualizowane w liście:
- dostępne lotniska z których istnieje wylot
- dostępne lotniska do których można dolecieć
- dostępne linie
- dostępne rezerwacje do odprawy
- dostępne samoloty danej linii
- dostępne miejsca

Dane wizualizowane jako pop message
- rezerwacja została już odprawiona
- brak wolnych miejsc

- odprawa niemożliwa lot już się odbył
- użytkownik już istnieje
- Błąd logowania

12.Zdefiniowanie panelu sterowania aplikacji.

Główny Panel user -
- Przyciski: logowania, rejestracji, wylogowania, szukania lotu, moje rezerwacje
- Listy z możliwością wyboru - skąd, dokąd, linia

Główny Panel admin -
- Przyciski: Pokaż pasażerów, dodaj lot, usuń lot, usuń rezerwacje
- Listy z możliwością wyboru - nr lotu

V. Dokumentacja

13.Wprowadzanie danych: zdefiniowanie sposobu wprowadzania danych (ręczne, automatyczne, import)

Dane Wprowadzane ręcznie:
- Dane osobowe
- Dane użytkownika
- nr rezerwacji do odprawy
- nr lotu do rezerwacji
- parametry lotu: skąd, dokąd, linia
- miejsce w samolocie
- Dane nowego lotu

Dane wprowadzane automatycznie:
- wszystkie identyfikatory id tabel
- lokalizacje w samolocie
- daty rezerwacji i odprawy
- typ użytkownika

Wszystkie tabele można uzupełnić z poziomu aplikacji z wyjątkiem rejestracji jako admin

14.Dokumentacja użytkownika: krótka instrukcja obsługi aplikacji.

By zacząć rezerwować loty z aplikacji użytkownik musi najpierw się zarejestrować, a kolejno zalogować. Bez zalogowania, gość może jedynie wyszukiwać dostępne loty.

Loty wyszukujemy wybierając interesujące nas parametry, takie jak: miejsce docelowe, wylotu, lub ulubioną linie. Po wyborze należy wybrać lot który nam pasuje(data, cena, ilość wolnych miejsc). Kolejnym krokiem jest podanie danych osobowych pasażera, a następnie zapłaty. Gdy, wszystko przebiegło pomyślnie rezerwacje można zobaczyć w panelu “Moje rezerwacje”. Tam też dokonujemy odprawy wybierając id_rezerwacji. Odprawy można dokonać w dowolnym czasie i z możliwością wyboru miejsca w samolocie za dodatkową opłatą. Odprawa bez opłat możliwa 2 dni przed odlotem.

15.Opracowanie dokumentacji technicznej

Dokumentacja została wygenerowana w javadoc. Znajduje się w folderze docs.

16.Wykaz literatury

- www.stackoverflow.com
- www.docs.oracle.com/en/java/javase/11/