MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
CATEDRA CALCULATOARE

Proiect de Semestru

la disciplina
Introducere in Baze de Date

Proiect realizat de catre :

Balcanu Vlad-Andrei si Albu Cristian-Gabriel

Grupa 30221 , AN II , CTI romana

Cuprins

MINISTERUL EDUCAȚIEI ȘI CERCETĂRII ȘTIINȚIFICE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

1.Introducere

Proiectul presupune dezvoltarea unei aplicatii care lucreaza cu baze de date pentru gestiunea unei companii de zboruri aeriene “Black Magic”. Scopul aplicatiei este simplificarea operatiilor cu baza de date prin oferirea unei interfete grafice pe care angajatii si clientii companiei sa o poata utiliza. Aplicatia ofera sprinjin atat pentru interogarea bazei de date cat si pentru manipularea acesteia.Pentru a putea accesa aplicatia , clientii trebuie sa isi creeze prima data un cont .

Cu ajutorul aplicatiei clientii pot sa vada toate calatoriile care sunt disponibile , impreuna cu pretul , numarul de locuri disponibile , numele avionului , firma si modelul acestuia . Clientii pot sa caute anumite calatorii in functie de orasul de plecare si orasul de aterizare al calatoriei . Ei pot sa faca rezervari , isi pot vedea calatoriile pe care au facut rezervari (incluzand si calatoriile care s-au intamplat deja) si pot sa anuleze o rezervare daca doresc.

Administratorii pot sa adauge curse , avioane, piloti si aeroporturi noi dar pot si sa anuleze curse si sa stearga avioane , aeroporturi ,pilotii si chiar si useri daca este nevoie . Ei pot sa mai vada toate datele legate de orice cursa , avion , aeroport si piloti , rezervarile oricarui user si pot sa vada toti userii care s-au inscris la o anumita cursa .

Aplicatia vine ca raspuns la stocarea de informatii pentru gestionarea unei companii aeriene corespunzator anului in care ne aflam deoarece informatiile se pot stoca si modifica mult mai usor virtual decat in registre, pe hartie.

Pentru dezvoltarea proiectului au fost folosite:

MySQL Workbench 8.0 - pentru crearea bazei de date, popularea initiala, dezvltarea de viewuri, proceduri si triggere si pentru crearea diagramei UML a tabelelor

Intellij – mediu de dezvoltare Java si aplicatie prin care am realizat conexiunea cu baza de date

Astah – pentru realizarea diagramei de clase din Java

2.Analiza cerintelor utilizatorilor

2.1. Ipotezele companiei aeriene (cerinte si constrangeri)

Aplicatia gestioneaza actiunile clientiilor si administratorilor dintr-o companie aeriana . Aceasta utilizeaza o baza de date care este supusa urmatoarelor cerinte :

- Exista doua tipuri de utilizatori : useri (clienti) si administratori ;
- Un administrator este unic identificat prin Id-ul sau . Acesta mai are nume , prenume , adresa , cnp , email-ul , username-ul si parola cu care se logheaza in aplicatie . Un administrator nu poate fi adaugat decat din baza de date de catre alt administrator ;
- Un user/client este unic identificat prin Id-ul sau . Acesta mai are nume , prenume , adresa , cnp , email-ul , username-ul si parola cu care se logheaza in aplicatie . Acestia trebuie sa isi creeze un cont prima data pentru a putea accesa aplicatia ;
- In aplicatie sunt mai multe curse disponibile , fiecare cursa este identificata prin Id-ul ei unic si dispune de mai multe caracteristici : Orase de plecare respectiv sosire , Data in care va avea loc cursa , ora de plecare respectiv sosire , pretul , id-ul avionului cu care se va efectua calatoria , locuri disponibile , locuri ocupate , id-ul aeroportului de plecare ;
- Avem mai multe avioane disponibile , fiecare fiind identificate de id-ul sau unic si avand Nume , Model , Numele companiei si capacitatea ;
- Avem mai multe aeropoarte de plecare disponibile , fiecare fiind identificat de id-ul sau unic si avand Nume , Oras , Adresa ;
- O cursa poate avea doar un singur avion , un singur aeroport de start , un singur pilot si mai multi clienti;
- Un user poate sa faca rezervare pe mai multe curse si pe mai multe locuri ;
- Toate rezervarile sunt stocate in tabela de rezervare . Fiecare rezervare este identificata prin id-ul sau unic si aceasta contine id-ul cursei pe care a fost facuta rezervarea , id-ul userului care a facut rezervarea precum si datele personale ale userului , numarul de locuri pe care userul le-a achizitionat pentru cursa respective si daca acesta are bagaj ;
- Pilotii sunt alesi de catre administrator cand se adauga o cursa si pot fi vazuti doar de catre acesta ;

2.2. Organizarea structurata tabelar a cerintelor utilizator

Baza de date trebuie sa stocheze urmatoarele informatii :

- Toti utilizatorii companiei aeriene (client si administratori);
- Date personale despre toti utilizatorii companiei ;
- Rezervarile realizate de catre clienti;
- Cursele impreuna cu caracteristicile acestora ;
- Orarul curselor ;
- Date despre avioanele folosite in timpul curselor ;
- Date despre aeroporturile din care se realizeaza plecare ;
- Date personale despre pilotii companiei ;

Aplicatia trebuie sa mai permita si urmatoarele operatii:

- Adaugarea si stergerea de clienti ;
- Adaugarea si anularea curselor de catre administratori;

- Adaugarea si stergerea avioanele de catre administratori;
- Adaugarea si stergerea aeropoartelor de catre administratori;
- Realizarea si anularea unei rezervari de catre clienti;
- Adaugarea si stergerea de piloti;

Aplicatia Java trebuie sa permita prelucrarea informatiilor din baza de date, cautare si afisarea informatiilor si conectarea utilizatorilor inregistrati.

2.3 Determinarea si caracterizarea profilurilor de utilizator

Avem doua tipuri de utilizatori :

1)Clienti , care au urmatoarele posibilitati in aplicatie si asupra bazei de date :

- Autentificare in aplicatie ;
- Vizualizarea curselor ;
- Cautarea anumitor curse dupa orasul de plecare is orasul de aterizare;
- Vizualizarea tuturor rezervarilor efectuate de catre el insasi ;
- Realizarea unei rezervari cat si anularea acesteia ;
- Vizualizarea informatiilor personale ;

2)Administratori , care au urmatoarele posibilitati in aplicatie si asupra bazei de date :

- Autentificare in aplicatie;
- Vizualizarea tuturor curselor;
- Stergerea conturilor de client daca este nevoie;
- Adaugarea si anularea unor curse;
- Adaugarea si stergerea unor avioane;
- Adaugarea si stergerea unor aeropoarte;
- Adaugarea si stergerea unor piloti;
- Vizualizarea de informatii despre curse ;
- Vizualizarea de informatii despre clienti;

3. Modelul de date si descrierea acestuia

3.1. Entitati si atributele lor

Tabelele :

User – Informatii despre toti utilizatorii aplicatiei companiei aeriene . Atribute : iduser, nume, prenume , adresa , CNP , nrTelefon , username , password si Admin (este un atribut de tipul Tinyint ce ne spune daca userul este administrator ; aceste este implicit 0 deoarece adminii pot fi adaugati doar direct din baza de date si nu din aplicatie) . Se leaga direct de Rezervare .

Aeroport – Informatii despre aeroporturile din care se fac decolarile avioanelor . Atribute : idaeroport , Nume , Adresa , Oras .Se leaga direct de Orar .

Date_Avion – Informatii despre avioanele folosite de catre aceasta companie aeriana . Atribute : id_avion , Nume ,NumeCompanie ,Model , NrLocuriDisponibile . Se leaga direct de Cursa.

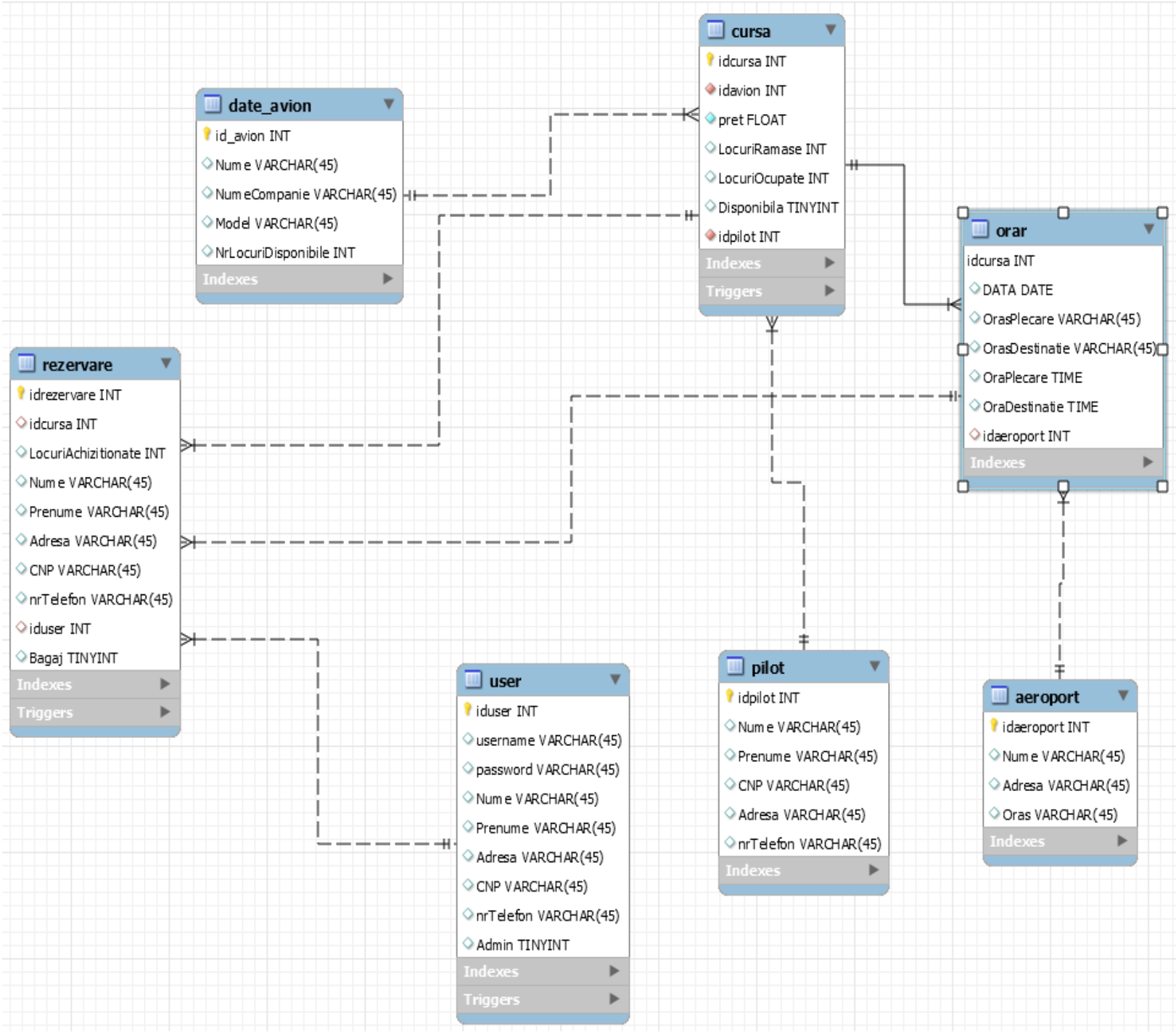
Cursa – Informatii despre curse in legatura cu pretul , locurile dispobibile , avionul . Atribute : idcursa , idavion , pret , LocuriRamase , LocuriOcupate , Disponibila (atribut de tipul tinyint , ne spune daca este disponibila cursa , este afectata de anumite informatii) . Se leaga direct de Orar ,Date_avion si Rezervare .

Orar – Informatii despre curse in legatura cu data la care are loc , locatia de decolare si de aterizare , ora de decolare si respective aterizare si aeroportul din care se face plecarea . Atribute : idcursa , DATA , OrasPlecare , OrasDestinatie , OraPlecare , Ora destinatie , idaeroport . Se leaga direct de Cursa , Aeroport si Rezervare .

Rezervare – Informatii despre rezervari . Atribute : idrezervare , idcursa , LocuriAchizitionate , Nume , Prenume , Adresa ,CNP , nrTelefon , iduser , Bagaj (atribut de tipul tinyint) . Se leaga direct de User , cursa si orar .

Pilot – Informatii despre pilotii companiei . Atribute : idpilot , Nume , Prenume , CNP , Adresa , nrTeefon . Se leaga direct de cursa .

3.2. Diagrama EER/UML pentru modelul de date complet



3.3. Proceduri , Triggere , view-uri

Proceduri:

1.AdaugareAeroport(in Nume1 varchar(45),in Oras1 varchar(45) , in Adresa1 varchar(45)) – Adauga un aeroport in baza de date . Nume1 – Numele noului aeroport , Oras1- Orasul in care se afla aeroportul , Adresa1 – Adresa aeroportului .

2.AdaugareAvion(in Nume1 varchar(45),in NumeCompanie1 varchar(45) ,in Model1 varchar(45) , in NrLocuriDisponibile1 int) – Adauga un avion in baza de date . Nume1 – Numele avionului , NumeCompanie1 – Numele companiei care a fabricat avionul , Model1 – Modelul avionului , NrLocuriDisponibile1 – Numarul de pasageri pe care il poate avea acest avion .

3.AdaugareCursa(in avion int(11) ,in suma float , in data1 date ,in destinatie varchar(45) ,

in ora1 time ,in ora2 time , in aeroport1 int , in pilot1 int) – Adauga o cursa noua in baza de date (in tabele Cursa si Orar) . Avion – id-ul avionului care va fi folosit , suma – pretul de achizitionare al unui bilet , data1 – data in care va avea loc zborul , destinatie – orasul de aterizare , ora1 – ora de decolare , ora2 – ora de aterizare , aeroport1 – id-ul aeroportului de plecare , pilot1 – id-ul pilotului ce va realiza cursa .

4.AdaugareRezervare(IN cursa1 INT,IN Locuri1 int,IN user1 int,IN bagaj1 tinyint) – Adauga o rezervare (functie folosita de clienti). Cursa1 – id-ul cursei la care se va face rezervarea , Locuri1 – numarul de locuri pe care clientul vrea sa le achizitioneze , user1 – id-ul clientului care doreste sa faca rezervarea , bagaj1 – un atribut de tip tinyint ce ne spune daca are sau nu bagaj .

5.AdaugareUser(IN user1 varchar(45),IN parola VARCHAR(45),IN nume1 VARCHAR(45),IN prenume1 VARCHAR(45) ,in adresa1 varchar(45) ,in CNP1 varchar(45) ,in nrTel1 varchar(45),in Admintype tinyint) – Adauga un user nou . User1 – username-ul contului nou, parola – parola contului , nume1 – Numele user-ului , prenume1- Prenumele user-ului , adresa1 – adresa userului , CNP1 – cnp-ul user-ului , nrTel1 – numarul de telefon al userului , Admintype – atribut de tip tinyint care ne spune daca userul este administrator (este pus automat 0 la sign-up , administratorii se adauga doar din baza de date direct) .

6.CautareDupaOrase(in Nume1 varchar(45), in Nume2 varchar(45)) – Cauta o cursa dipa orasul de plecare si cel de aterizare . Nume1 – Orasul de plecare , Nume2 – orasul de aterizare .

7.CautareDupaUser(in id1 int) – Ne afiseaza toate rezervarile facute de un user . Id1 – id-ul userului caruia dorim sa ii vedem rezervarile .

8.CautareRezervare(in id1 int) – Ne afiseaza toate datele despre o anumita rezervare . Id1 – id-ul rezervarii pe care dorim sa o vizualizam .

9.StergereAeroport(in id1 int) – Sterge aeroportul cu id-ul dat . Id1 – id-ul aeroportului pe care dorim sa il stergem .

10.StergereAvion(in id1 int) – Sterge avionul cu id-ul dat . Id1 – id-ul avionului pe care dorim sa il stergem .

11.StergereDinTabelCursa(in id1 int) – Sterge cursa cu id-ul dat din tabelele cursa si orar si toate rezervarile ce erau facute pe aceasta cursa . Id1 – id-ul cursei pe care dorim sa o stergem .

12.StergereRezervare(in id1 int) – Sterge rezervarea cu id-ul dat . Id1 – id-ul rezervarii pe care dorim sa o stergem .

13.StergereUser(int id1 int) – Sterge user-ul cu id-ul dat . Id1 – id-ul user-ului pe care dorim sa il stergem .

14.AdaugarePilot(in Nume1 varchar(45),in Prenume1 varchar(45) , in Adresa1 varchar(45) , in nrTel varchar(45) , in CNP1 varchar(45)) – Adauga un pilot nou in tabela piloti . Nume1 – Numele noului pilot , Prenume1 – prenumele noului pilot , Adresa1 – Adresa de resedinta a pilotului , nrTel – numarul de telefon al pilotului , CNP1 – cnp-ul pilotului .

15.StergerePilot(in id1 int) – Sterge pilotul cu id-ul dat din tabela pilot daca acesta nu este angajat intr-o cursa in acel moment . Id1 – id-ul pilotului pe care dorim sa il stergem.

Triggere :

1.RezervareCompletare – Cand se realizeaza o rezervare , completeaza datele personale ale user-ului din rezervare automat dupa id-ul userului (Nume,Prenume , Adresa, CNP , nrTelefon);

2.CursaCompletare – Cand se adauga o cursa noua seteaza numarul de locuri ramase ale cursei dupa avionul ales pentru cursa si seteaza numarul de locuri ocupate la 0 ;

3.CursaCompletare1 – Dupa ce se face o rezervare se actualizeaza numarul de locuri ocupate si numarul de locuri ramase ale cursei pentru care s-a facut rezervarea ;

4.CursaVerificareNrLocuri – Inainte de a se face o rezervare , se verifica daca numarul de locuri ramase ale cursei dupa rezervare este mai mic sau egal cu 0 . Daca da , cursa devine indisponibila .

5.VerificareData – Dupa ce se face un update in tabela cursa , se verifica daca datele curselor sunt in viitor fata de data curenta . Daca nu sunt , cursa va fi stersa automat din orar (nu si din tabela cursa).

6.VerificareData1 – Dupa ce se adauga o cursa noua , se verifica daca datele curselor sunt in viitor fata de data curenta . Daca nu sunt , cursa va fi stersa automat din orar (nu si din tabela cursa).

7.StergereRezervare – Dupa ce a fost stearsa o rezervara (anulata) , se actualizeaza numarul de locuri ocupate si numarul de locuri ramase ale cursei la care s-a anulat rezervarea . Daca , inainte de anulare cursa era indisponibila , aceasta va deveni din nou disponibila .

8.StergereRezervariDupaStergereUser – Daca un user este sters din baza de date , toate rezervarile acestuia sunt sterse automat din baza de date .

View-uri :

1.Dateaeroportcursa – View cu informatii despre aeroporturile din care se fac plecarile .

2.Dateavionincursa – View cu informatii despre avioanele folosite in curse .

3.Trasee – View cu clientii (Numele si prenumele) si informatii despre cursele viitoare (Orasele de plecare si de destinatie si orele de plecare si aterizare).

4.Cursacompleta – View cu informatii complete despre toate cursele .

5.Viewuser – View cu informatii despre fiecare user .

3.4. Normalizarea Datelor

Definitia formei normale Boyce-Codd (FNBC) este: Fie R o schemă de relație si F multimea de dependențe funcționale asociată. Se spune ca R este în forma normală Boyce-Codd dacă și numai dacă oricare ar fi o dependență netrivială X →Y din F , atunci X este supercheie pentru R.

Baza de date respectă. Atributele fiecărui tabel nu depind de alte atribute. Fiecare tabel are o singură cheie primară după care sunt identificate înregistrările și este suficientă pentru a identifica în mod unic orice înregistrare din baza de date.

În fiecare tabel avem doar o cheie și toate dependențele au în partea stângă o supercheie (cheia primară a tabelei). De exemplu, pentru tabela Cursa avem cheia primară idcursa și toate dependențele au în stânga această supercheie, în tabelul User această supercheie este iduser etc.

3.5. Interogari MySQL

select * from aeroport where oras="Bucuresti";

$\pi_{oras}\sigma_{oras=”Bucuresti”;}$

Imi afiseaza toate informatiile despre aeroporturile ce se afla in Bucuresti ;

select * from user where username='"+user1+"';

$\pi_{username}\sigma_{username=user1”;}$

Selecteaza user-ul ce are username-ul egal cu cel introdus de la tastatura ;

select idcursa, pret, OrasPlecare, OrasDestinatie, LocuriOcupate from orar inner join cursa using(idcursa) where OrasPlecare="Constanta";

Selecteaza informatii din orar si cursa despre cursele care au un anumit oras de plecare (In acest exemplu am ales Constanta) .

Select * from Dateaeroportcursa ;

$\pi_{*}\sigma($Dateaeroportcursa$)$

Selecteaza toate datele din view-ul Dateaeroportcursa;

Select * from Trasee;

$\pi_{*}\sigma($Trasee$)$

Selecteaza toate datele din view-ul Trasee;

Select * from cursacompleta ;

$\pi_{*}\sigma($cursacompleta$)$

Selecteaza toate datele din view-ul cursacompleta;

Select NumeComplet,NumeDeUtilizator from viewuser where ID=" + id+ ";

$\pi_{*}\sigma_{\text{ID=id}}($viewuser$)$

Selecteaza informatii despre un user din view-ul viewuser unde id-ul userului este egal cu id-ul dat;

Select Nume,NumeCompanie,Model,LocuriRamase from dateavionincursa where idcursa="+id+";

$\pi_{*}\sigma_{\text{idcursa=id}}($dateavionincursa$)$

Selecteaza informatii despre avion din view-ul dateavionincursa unde id-ul cursei este egal cu id-ul dat;

Select iduser from user where username='"+user1+"' and password ='"+ password1 + "';

$\pi_{\text{username,password}}\sigma_{\text{username=user1,password=password1}}$";

Selecteaza id-ul userul-ui care are username-ul egal cu user1 si password-ul egal cu password1 . Poate fi folosit la verificarea de crededentiale .

select LocuriAchizitionate iduser,Nume,Prenume,idcursa,OrasPlecare,OrasDestinatie from rezervare join orar using (idcursa) ;

Selecteaza informatii despre user si cursa din tabele cursa si rezervare .

3.6. Cod MySQL

3.6.1 Cod pentru crearea bazei de date si a tabelelor

```
CREATE SCHEMA `aeroport` ;
use aeroport;

CREATE TABLE `aeroport`.`aeroport` (
  `idaeroport` INT NOT NULL AUTO_INCREMENT,
  `Nume` VARCHAR(45) NULL,
  `Adresa` VARCHAR(45) NULL,
  `Oras` VARCHAR(45) NULL,
  PRIMARY KEY (`idaeroport`));

CREATE TABLE `aeroport`.`cursa` (
  `idcursa` INT NOT NULL AUTO_INCREMENT,
  `idavion` INT NOT NULL,
  `pret` FLOAT NOT NULL,
  `LocuriRamase` INT NULL DEFAULT NULL,
  `LocuriOcupate` INT NULL DEFAULT NULL,
  `Disponibila` TINYINT NULL DEFAULT 1,
  PRIMARY KEY (`idcursa`));

CREATE TABLE `aeroport`.`date_avion` (
  `id_avion` INT NOT NULL AUTO_INCREMENT,
```

```
  `Nume` VARCHAR(45) NULL DEFAULT NULL,
  `NumeCompanie` VARCHAR(45) NULL DEFAULT NULL,
  `Model` VARCHAR(45) NULL DEFAULT NULL,
  `NrLocuriDisponibile` INT NULL,
  PRIMARY KEY (`idavion`));


CREATE TABLE `aeroport`.`orar` (
  `idcursa` INT NOT NULL AUTO_INCREMENT,
  `DATA` DATE NULL DEFAULT NULL,
  `OrasPlecare` VARCHAR(45) NULL DEFAULT NULL,
  `OrasDestinatie` VARCHAR(45) NULL DEFAULT NULL,
  `OraPlecare` TIME NULL DEFAULT NULL,
  `OraDestinatie` TIME NULL DEFAULT NULL,
  `idaeroport` INT NULL,
  PRIMARY KEY (`idcursa`));


CREATE TABLE `aeroport`.`rezervare` (
  `idrezervare` INT NOT NULL AUTO_INCREMENT,
  `idcursa` INT NULL DEFAULT NULL,
  `LocuriAchizitionate` INT NULL DEFAULT NULL,
  `Nume` VARCHAR(45) NULL DEFAULT NULL,
  `Prenume` VARCHAR(45) NULL DEFAULT NULL,
  `Adresa` VARCHAR(45) NULL DEFAULT NULL,
  `CNP` VARCHAR(45) NULL DEFAULT NULL,
  `NrTelefon` VARCHAR(45) NULL DEFAULT NULL,
  `iduser` INT NULL DEFAULT NULL,
  `Bagaj` TINYINT NULL DEFAULT NULL,
  PRIMARY KEY (`idrezervare`));
```

```
CREATE TABLE `aeroport`.`user` (
  `iduser` INT NOT NULL AUTO_INCREMENT,
  `username` VARCHAR(45) NULL DEFAULT NULL,
  `password` VARCHAR(45) NULL DEFAULT NULL,
  `Nume` VARCHAR(45) NULL DEFAULT NULL,
  `Prenume` VARCHAR(45) NULL DEFAULT NULL,
  `Adresa` VARCHAR(45) NULL DEFAULT NULL,
  `CNP` VARCHAR(45) NULL DEFAULT NULL,
  `nrTelefon` VARCHAR(45) NULL DEFAULT NULL,
  `Admin` TINYINT NULL DEFAULT NULL,
  PRIMARY KEY (`iduser`));

ALTER TABLE `aeroport`.`rezervare`
CHANGE COLUMN `NrTelefon` `nrTelefon` VARCHAR(45) NULL DEFAULT NULL ;

ALTER TABLE `aeroport`.`cursa`
DROP FOREIGN KEY `avion1`;
ALTER TABLE `aeroport`.`cursa`
ADD INDEX `avion_idx` (`idavion` ASC) VISIBLE,
DROP INDEX `avion1_idx` ;
;
ALTER TABLE `aeroport`.`cursa`
ADD CONSTRAINT `avion`
  FOREIGN KEY (`idavion`)
  REFERENCES `aeroport`.`date_avion` (`id_avion`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;

ALTER TABLE `aeroport`.`orar`
```

```
ADD INDEX `aeroport_idx` (`idaeroport` ASC) VISIBLE;
;
ALTER TABLE `aeroport`.`orar`
ADD CONSTRAINT `aeroport`
  FOREIGN KEY (`idaeroport`)
  REFERENCES `aeroport`.`aeroport` (`idaeroport`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;

ALTER TABLE `aeroport`.`orar`
ADD CONSTRAINT `cursa`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`cursa` (`idcursa`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;

ALTER TABLE `aeroport`.`rezervare`
ADD INDEX `cursa1_idx` (`idcursa` ASC) VISIBLE;
;
ALTER TABLE `aeroport`.`rezervare`
ADD CONSTRAINT `cursa1`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`cursa` (`idcursa`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;


ALTER TABLE `aeroport`.`rezervare`
ADD INDEX `user1_idx` (`iduser` ASC) VISIBLE;
;
```

```
ALTER TABLE `aeroport`.`rezervare`
ADD CONSTRAINT `user1`
  FOREIGN KEY (`iduser`)
  REFERENCES `aeroport`.`user` (`iduser`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;


ALTER TABLE `aeroport`.`rezervare`
ADD CONSTRAINT `cursa2`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`orar` (`idcursa`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;

ALTER TABLE `aeroport`.`cursa`
DROP FOREIGN KEY `avion`,
DROP FOREIGN KEY `pilot1`;
ALTER TABLE `aeroport`.`cursa`
ADD CONSTRAINT `avion`
  FOREIGN KEY (`idavion`)
  REFERENCES `aeroport`.`date_avion` (`id_avion`)
  ON DELETE CASCADE,
ADD CONSTRAINT `pilot1`
  FOREIGN KEY (`idpilot`)
  REFERENCES `aeroport`.`pilot` (`idpilot`)
  ON DELETE CASCADE;

ALTER TABLE `aeroport`.`orar`
DROP FOREIGN KEY `aeroport`;
```

```
ALTER TABLE `aeroport`.`orar`
ADD CONSTRAINT `aeroport`
  FOREIGN KEY (`idaeroport`)
  REFERENCES `aeroport`.`aeroport` (`idaeroport`)
  ON DELETE CASCADE;

ALTER TABLE `aeroport`.`rezervare`
DROP FOREIGN KEY `cursa1`;
ALTER TABLE `aeroport`.`rezervare`
ADD CONSTRAINT `cursa1`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`cursa` (`idcursa`)
  ON DELETE CASCADE;

ALTER TABLE `aeroport`.`orar`
DROP FOREIGN KEY `cursa`;
ALTER TABLE `aeroport`.`orar`
ADD CONSTRAINT `cursa`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`cursa` (`idcursa`)
  ON DELETE CASCADE;

ALTER TABLE `aeroport`.`rezervare`
DROP FOREIGN KEY `cursa2`,
DROP FOREIGN KEY `user1`;
ALTER TABLE `aeroport`.`rezervare`
ADD CONSTRAINT `cursa2`
  FOREIGN KEY (`idcursa`)
  REFERENCES `aeroport`.`orar` (`idcursa`)
  ON DELETE CASCADE,
```

```
ADD CONSTRAINT `user1`
  FOREIGN KEY (`iduser`)
  REFERENCES `aeroport`.`user` (`iduser`)
  ON DELETE CASCADE;
```

3.6.2 Inserari Date Existente

```
DELETE FROM `aeroport`.`aeroport` WHERE (`idaeroport` = '5');

INSERT INTO `aeroport`.`aeroport` (`idaeroport`, `Nume`, `Adresa`, `Oras`) VALUES ('1', 'Avram Iancu', 'str.Capitan Busila', 'Bucuresti');

INSERT INTO `aeroport`.`aeroport` (`idaeroport`, `Nume`, `Adresa`, `Oras`) VALUES ('2', 'George Enescu', 'str. E85', 'Bacau');

INSERT INTO `aeroport`.`aeroport` (`idaeroport`, `Nume`, `Adresa`, `Oras`) VALUES ('3', 'Aer. Int. Maramures', 'str E58', 'Baia Mare');

INSERT INTO `aeroport`.`aeroport` (`idaeroport`, `Nume`, `Adresa`, `Oras`) VALUES ('4', 'Henri Coanda', 'str. DN1', 'Bucuresti');

INSERT INTO `aeroport`.`aeroport` (`idaeroport`, `Nume`, `Adresa`, `Oras`) VALUES ('5', 'Mihail Kogalniceanu', 'str. Aeroportului', 'Constanta');

INSERT INTO `aeroport`.`aeroport` (`Nume`, `Adresa`, `Oras`) VALUES ('Avram Iancu', 'str. Traian Vuia nr. 149', 'Cluj');



INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('1', 'Dreamliner', 'Boeing', '787', '260');

INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('2', 'Airbus A', 'Airbus', 'A380', '500');

INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('3', 'Airbus A', 'Airbus', 'A320', '400');

INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('4', 'Douglas', 'McDonnel', 'MD-80', '480');

INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('5', 'Douglas', 'McDonnel', 'MD-90', '520');

INSERT INTO `aeroport`.`date_avion` (`id_avion`, `Nume`, `NumeCompanie`, `Model`, `NrLocuriDisponibile`) VALUES ('6', 'TU', 'Tupolev', '154', '350');
```

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('1', 'Vlad', '12345', 'Balcanu', 'Vlad', 'Tg.ocna Bacau', '12451234', '21412581', '1');

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('2', 'Cristi', '12345', 'Albu', 'Cristian', 'Alba-Iulia', '125131', '23615', '1');

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('3', 'Mada', '00000', 'Kasler', 'Madalina', 'Deva', '215152342135', '12315321', '0');

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('4', 'Octav', '12345', 'Stanciu', 'Octavian', 'Piatra-Neamt', '21513214312', '2142415', '0');

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('5', 'Ghera', '00000', 'Gherasim', 'Andrei', 'Baia-Mare', '12341534', '48325714', '0');

INSERT INTO `aeroport`.`user` (`iduser`, `username`, `password`, `Nume`, `Prenume`, `Adresa`, `CNP`, `nrTelefon`, `Admin`) VALUES ('6', 'Adi', '12345', 'Birle', 'Adrian', 'Baia-Mare', '2145351432', '21445141', '0');

DELETE FROM `aeroport`.`pilot` WHERE (`idpilot` = '1');

INSERT INTO `aeroport`.`pilot` (`idpilot`, `Nume`, `Prenume`, `CNP`, `Adresa`, `nrTelefon`) VALUES ('1', 'Balasa', 'Iustin', '2155123', '124541', '12431422');

INSERT INTO `aeroport`.`pilot` (`idpilot`, `Nume`, `Prenume`, `CNP`, `Adresa`, `nrTelefon`) VALUES ('2', 'Cornelius', 'Alex', '125124', '4321512', '321125');

INSERT INTO `aeroport`.`pilot` (`idpilot`, `Nume`, `Prenume`, `CNP`, `Adresa`, `nrTelefon`) VALUES ('3', 'Soner', 'Iulia', '12431514632', '94535703824', '3209483');

INSERT INTO `aeroport`.`pilot` (`idpilot`, `Nume`, `Prenume`, `CNP`, `Adresa`, `nrTelefon`) VALUES ('4', 'Caraman', 'Victor', '1299569959`', '3495828', '4193453');

INSERT INTO `aeroport`.`pilot` (`idpilot`, `Nume`, `Prenume`, `CNP`, `Adresa`, `nrTelefon`) VALUES ('5', 'Lubej', 'Ioana', '2194182481', '491581248', '214912831');

INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('1', '1', '500', '0', '0', '1', '1');

```
INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('2', '4', '350', '0', '0', '1', '5');

INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('3', '6', '420', '0', '0', '1', '3');

INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('4', '2', '450', '0', '0', '1', '2');

INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('5', '3', '300', '0', '0', '1', '4');

INSERT INTO `aeroport`.`cursa` (`idcursa`, `idavion`, `pret`, `LocuriRamase`, `LocuriOcupate`, `Disponibila`, `idpilot`) VALUES ('6', '5', '370', '0', '0', '1', '3');
```

```
INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('1', '2021-10-10', 'Londra', '15:00', '19:00', '1');

INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasPlecare`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('2', '2021-01-12', '', 'BeiJing', '12:00', '21:00', '4');

INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasPlecare`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('3', '2022-01-01', '', 'Boston', '01:30', '11:20', '2');

INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('4', '2021-02-06', 'Paris', '08:30', '01:30', '3');

INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('5', '2022-05-05', 'Berlin', '02:30', '10:00', '5');

INSERT INTO `aeroport`.`orar` (`idcursa`, `DATA`, `OrasDestinatie`, `OraPlecare`, `OraDestinatie`, `idaeroport`) VALUES ('6', '2022-02-02', 'Madrid', '00:00', '02:00', '6');
```

```
INSERT INTO `aeroport`.`rezervare` (`idrezervare`, `idcursa`, `LocuriAchizitionate`, `iduser`, `Bagaj`) VALUES ('1', '3', '5', '2', '1');

INSERT INTO `aeroport`.`rezervare` (`idrezervare`, `idcursa`, `LocuriAchizitionate`, `iduser`, `Bagaj`) VALUES ('2', '1', '2', '4', '1');

INSERT INTO `aeroport`.`rezervare` (`idrezervare`, `idcursa`, `LocuriAchizitionate`, `iduser`, `Bagaj`) VALUES ('3', '3', '2', '3', '0');

INSERT INTO `aeroport`.`rezervare` (`idrezervare`, `idcursa`, `LocuriAchizitionate`, `iduser`, `Bagaj`) VALUES ('4', '4', '4', '2', '0');

INSERT INTO `aeroport`.`rezervare` (`idrezervare`, `idcursa`, `LocuriAchizitionate`, `iduser`, `Bagaj`) VALUES ('5', '6', '10', '6', '1');
```

3.6.3 Cod Proceduri

1.AdaugareAeroport

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugareAeroport`(in Nume1 varchar(45),in Oras1 varchar(45)  , in Adresa1 varchar(45))

BEGIN

INSERT INTO aeroport(Nume,Oras,Adresa) VALUES (Nume1,Oras1,Adresa1);

END
```

2.AdaugareAvion

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugareAvion`(in Nume1 varchar(45),in NumeCompanie1 varchar(45) ,in Model1 varchar(45) , in NrLocuriDisponibile1 int)

BEGIN

INSERT INTO date_avion(Nume,NumeCompanie,Model,NrLocuriDisponibile) VALUES (Nume1,NumeCompanie1,Model1,NrLocuriDisponibile1);

END
```

3.**AdaugareCursa**

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugareCursa`(in avion int(11) ,in suma float , in data1 date ,in destinatie varchar(45)  ,

in ora1 time ,in ora2 time , in aeroport1 int , in pilot1 int)

BEGIN

INSERT INTO cursa (idavion,pret,Disponibila,idpilot) VALUES (avion,suma,1,pilot1);

INSERT INTO orar (DATA,OrasDestinatie,OraPlecare,OraDestinatie,idaeroport)

VALUES(data1,destinatie,ora1,ora2,aeroport1);

update orar

set OrasPlecare = (select Oras from aeroport where idaeroport=orar.idaeroport);

END
```

4.AdaugareRezervare

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugareRezervare`(IN cursa1 INT,IN Locuri1 int,IN user1 int,IN bagaj1 tinyint)

BEGIN

INSERT INTO rezervare (idcursa,LocuriAchizitionate,iduser,Bagaj) VALUES (cursa1,locuri1,user1,bagaj1);

END
```

5.AdaugareUser

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugareUser`(IN user1 varchar(45),IN parola VARCHAR(45),IN nume1 VARCHAR(45),IN prenume1 VARCHAR(45) ,in adresa1 varchar(45) ,in CNP1 varchar(45) ,in nrTel1 varchar(45),in Admintype tinyint)

BEGIN

INSERT INTO user (username,password,Nume,Prenume,Adresa,CNP,nrTelefon,Admin) VALUES (user1,parola,nume1,prenume1,adresa1,CNP1,nrTel1,Admintype);

END
```

6.CautareDupaOrase

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `CautareDupaOrase`(in Nume1 varchar(45), in Nume2 varchar(45) )

BEGIN

SELECT * FROM cursa, orar, aeroport where cursa.idcursa=orar.idcursa and orar.idaeroport=aeroport.idaeroport and OrasPlecare=Nume1 and OrasDestinatie=Nume2 order by DATA;

END
```

7.CautareDupaUser

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `CautareDupaUser`(in id1 int )

BEGIN

select * from rezervare where iduser=id1;

END
```

8.CautareRezervare

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `CautareRezervare`(in id1 int )
BEGIN
select * from rezervare where idrezervare = id1;
END
```

9.StergereAeroport

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergereAeroport`(in id1 int)
BEGIN
DELETE FROM aeroport where idaeroport=id1;
END
```

10.StergereAvion

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergereAvion`(in id1 int)
BEGIN
DELETE FROM date_avion where id_avion=id1;
END
```

11.StergereDubTabelCursa

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergereDinTabelCursa`(in id1 int)
BEGIN
DELETE FROM rezervare where idcursa=id1;
DELETE FROM orar where idcursa=id1;
DELETE FROM cursa where idcursa=id1;
END
```

12.StergereRezervare

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergereRezervare`(in id1 int)
BEGIN
DELETE FROM rezervare where idrezervare=id1;
END
```

13.StergereUser

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergereUser`(in id1 int)
BEGIN
DELETE FROM user where iduser=id1;
END
```

14.AdaugarePilot

```
USE `aeroport`;
DROP procedure IF EXISTS `AdaugarePilot`;

DELIMITER $$
USE `aeroport`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `AdaugarePilot`(in Nume1 varchar(45),in
Prenume1 varchar(45)  , in Adresa1 varchar(45) , in nrTel varchar(45) ,
in CNP1 varchar(45))
BEGIN
INSERT INTO pilot(Nume,Prenume,CNP,Adresa,nrTelefon) values
(Nume1,Prenume1,CNP1,Adresa1,nrTel);
END$$

DELIMITER ;
```

15.StergerePilot

```
USE `aeroport`;
DROP procedure IF EXISTS `aeroport`.`StergerePilot`;
;

DELIMITER $$
USE `aeroport`$$
CREATE DEFINER=`root`@`localhost` PROCEDURE `StergerePilot`(in id1 int)
```

```
BEGIN
DELETE FROM pilot where idpilot=id1;
END$$

DELIMITER ;
;
```

3.6.4 Cod Triggere

#triggere pe tabela cursa

1.CursaCompletare

```
DROP TRIGGER CursaCompletare;
DELIMITER //
CREATE TRIGGER CursaCompletare before insert on cursa
FOR EACH ROW
BEGIN
set new.LocuriRamase=(select NrLocuriDisponibile from date_avion where date_avion.id_avion
=new.idavion);
set new.LocuriOcupate=0;

END //
DELIMITER ;
```

2.VerificareData

```
DROP TRIGGER VerificareData;
DELIMITER //
CREATE TRIGGER VerificareData after update on cursa
FOR EACH ROW
BEGIN
```

```
DELETE FROM ORAR where DATA < CURRENT_DATE ;
END //
DELIMITER ;
```

3.VerificareData1

```
DROP TRIGGER VerificareData1;
DELIMITER //
CREATE TRIGGER VerificareData1 after insert on cursa
FOR EACH ROW
BEGIN
DELETE FROM ORAR where DATA < CURRENT_DATE ;
END //
DELIMITER ;
```

4.CursaVerificareNrLocuri

```
DROP TRIGGER CursaVerificareNrlocuri;
DELIMITER //
CREATE TRIGGER CursaVerificareNrlocuri before update on cursa
FOR EACH ROW
BEGIN
if(new.LocuriRamase<=0) then
set new.Disponibila=0;
end if;
END //
DELIMITER ;
```

#Triggere pe tabela rezervare

5.RezervareCompletare

```
DROP TRIGGER RezervareCompletare;
```

```
DELIMITER //
CREATE TRIGGER RezervareCompletare before insert on rezervare
FOR EACH ROW
BEGIN
set new.Nume=(select Nume from user where user.iduser = new.iduser);
set new.Prenume=(select Prenume from user where user.iduser=new.iduser);
set new.Adresa=(select Adresa from user where user.iduser=new.iduser);
set new.CNP=(select CNP from user where user.iduser=new.iduser);
set new.nrTelefon=(select nrTelefon from user where user.iduser=new.iduser);

END //
DELIMITER ;
```

6.CursaCompletare1

```
DROP TRIGGER CursaCompletare1;
DELIMITER //
CREATE TRIGGER CursaCompletare1 before insert on rezervare
FOR EACH ROW
BEGIN
update cursa
set LocuriOcupate=LocuriOcupate+new.LocuriAchizitionate where cursa.idcursa=new.idcursa;
update cursa
set LocuriRamase=LocuriRamase-new.LocuriAchizitionate where cursa.idcursa=new.idcursa;
END //
DELIMITER ;
```

7.StergereRezervare

```
DELIMITER ;
DROP TRIGGER StergereRezervare;
DELIMITER //
```

```
CREATE TRIGGER StergereRezervare Before delete on rezervare
FOR EACH ROW
BEGIN
update cursa
set LocuriRamase=LocuriRamase+old.LocuriAchizitionate where idcursa=old.idcursa;
update cursa
set LocuriOcupate=LocuriOcupate-old.LocuriAchizitionate where idcursa=old.idcursa;
update cursa
set Disponibila=1 where idcursa=old.idcursa;
END //
DELIMITER ;
```

#Triggere pe tabela user

8.StergereRezervariDupaStergereUser

```
DELIMITER ;
DROP TRIGGER StergereRezervariDupaStergereUser;
DELIMITER //
CREATE TRIGGER StergereRezervariDupaStergereUser Before delete on user
FOR EACH ROW
BEGIN
DELETE FROM rezervare where iduser=old.iduser ;
END //
DELIMITER ;
```

3.6.5 Cod View-uri

1.View cu informatii despre aeroporturile din care se fac plecarile la fiecare cursa

```
CREATE
   ALGORITHM = UNDEFINED
```

```
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `dateaeroportcursa` AS
    SELECT
        `orar`.`idcursa` AS `idcursa`,
        `aeroport`.`idaeroport` AS `idaeroport`,
        `aeroport`.`Nume` AS `Nume`,
        `aeroport`.`Adresa` AS `Adresa`,
        `aeroport`.`Oras` AS `Oras`,
        `orar`.`DATA` AS `DATA`,
        `orar`.`OraPlecare` AS `OraPlecare`
    FROM
        (`aeroport`
        JOIN `orar` ON ((`aeroport`.`idaeroport` = `orar`.`idaeroport`)))
```

2. View cu informatii despre avioanele folosite in fiecare cursa

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `dateavionincursa` AS
    SELECT
        `date_avion`.`id_avion` AS `id_avion`,
        `date_avion`.`Nume` AS `Nume`,
        `date_avion`.`NumeCompanie` AS `NumeCompanie`,
        `date_avion`.`Model` AS `Model`,
        `cursa`.`idcursa` AS `idcursa`,
        `cursa`.`pret` AS `pret`,
        `cursa`.`LocuriRamase` AS `LocuriRamase`,
        `cursa`.`LocuriOcupate` AS `LocuriOcupate`
```

```
  FROM
    (`date_avion`
    JOIN `cursa`)
  WHERE
    (`date_avion`.`id_avion` = `cursa`.`idavion`)
```

3. View cu informatii despre clienti si despre cursele lor viitoare

```
CREATE
  ALGORITHM = UNDEFINED
  DEFINER = `root`@`localhost`
  SQL SECURITY DEFINER
VIEW `trasee` AS
  SELECT
    `rezervare`.`Nume` AS `Nume`,
    `rezervare`.`Prenume` AS `Prenume`,
    `rezervare`.`iduser` AS `iduser`,
    `rezervare`.`idrezervare` AS `idrezervare`,
    `rezervare`.`idcursa` AS `idcursa`,
    `orar`.`OrasPlecare` AS `OrasPlecare`,
    `orar`.`OrasDestinatie` AS `OrasDestinatie`,
    `orar`.`OraPlecare` AS `OraPlecare`,
    `orar`.`OraDestinatie` AS `Oradestinatie`
  FROM
    (`rezervare`
    JOIN `orar` ON ((`rezervare`.`idcursa` = `orar`.`idcursa`)))
```

4.View cu informatii complete despre toate cursele

```
CREATE
  ALGORITHM = UNDEFINED
```

```
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `trasee` AS
    SELECT
        `rezervare`.`Nume` AS `Nume`,
        `rezervare`.`Prenume` AS `Prenume`,
        `rezervare`.`iduser` AS `iduser`,
        `rezervare`.`idrezervare` AS `idrezervare`,
        `rezervare`.`idcursa` AS `idcursa`,
        `orar`.`OrasPlecare` AS `OrasPlecare`,
        `orar`.`OrasDestinatie` AS `OrasDestinatie`,
        `orar`.`OraPlecare` AS `OraPlecare`,
        `orar`.`OraDestinatie` AS `Oradestinatie`
    FROM
        (`rezervare`
        JOIN `orar` ON ((`rezervare`.`idcursa` = `orar`.`idcursa`)))
```

5.View cu informatii despre useri

```
CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
VIEW `viewuser` AS
    SELECT
        `user`.`iduser` AS `ID`,
        CONCAT(`user`.`Nume`, ' ', `user`.`Prenume`) AS `NumeComplet`,
        `user`.`username` AS `NumeDeUtilizator`,
        `user`.`password` AS `parola`
    FROM
        `user`
```

3.6.6 Cod Creare Useri

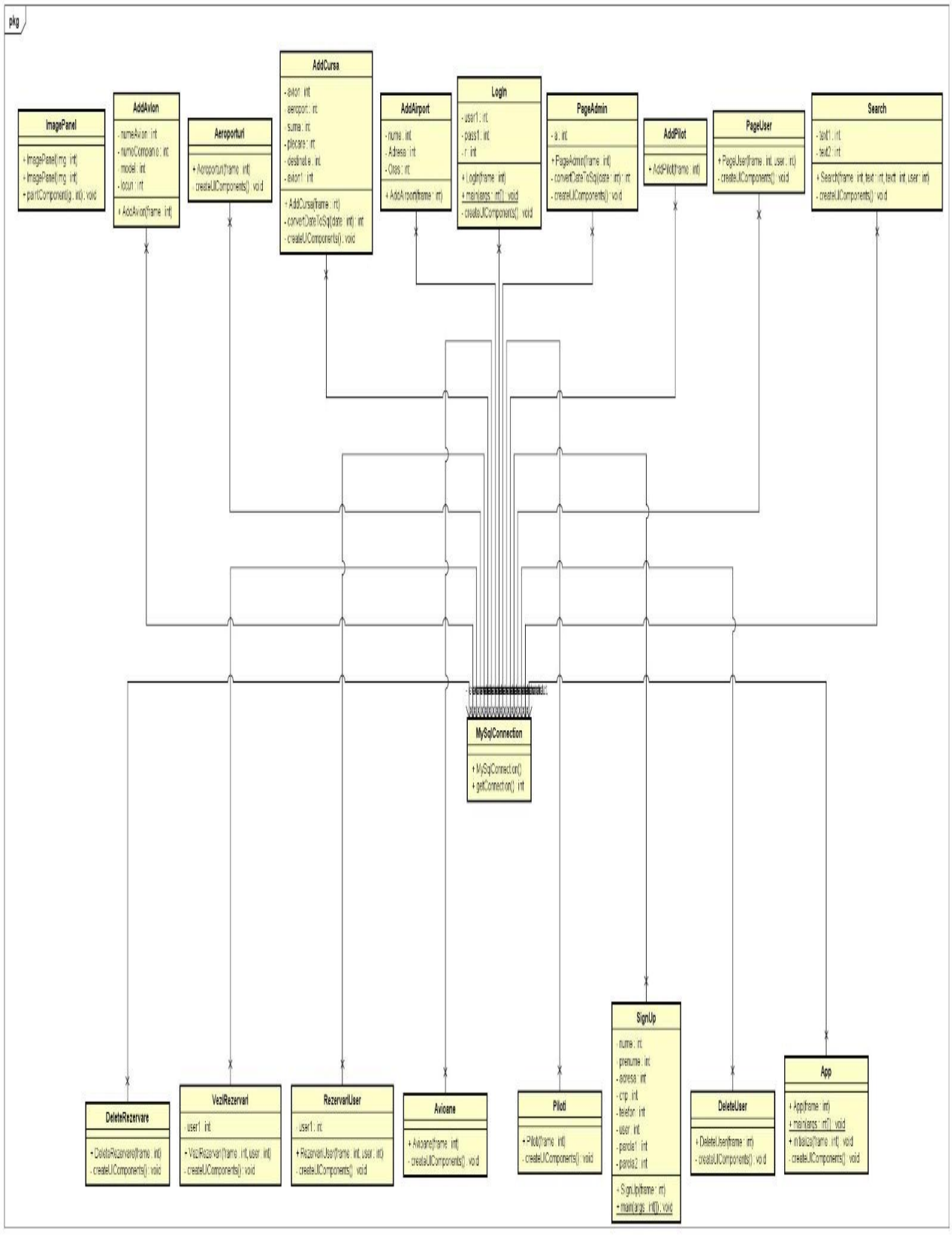
```
DROP USER 'admin'@'localhost';
CREATE USER 'admin'@'localhost';
GRANT SELECT ON aeroport.*TO'admin'@'localhost';
GRANT INSERT ON aeroport.*TO'admin'@'localhost';
GRANT UPDATE ON aeroport.*TO'admin'@'localhost';
GRANT DELETE ON aeroport.*TO'admin'@'localhost';
GRANT SHOW VIEW ON aeroport.*TO'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugareAeroport TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugareAvion TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugareCursa TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugarePilot TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugareUser TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.CautareDupaUser TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.CautareRezervare TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergereAeroport TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergereAvion TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergereDinTabelCursa TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergerePilot TO 'admin'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergereUser TO 'admin'@'localhost';

DROP USER 'client'@'localhost';
CREATE USER 'client'@'localhost';
GRANT SELECT ON aeroport.*TO'client'@'localhost';
GRANT SHOW VIEW ON aeroport.*TO'client'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.AdaugareRezervare TO 'client'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.CautareDupaOrase TO 'client'@'localhost';
GRANT EXECUTE ON PROCEDURE aeroport.StergereRezervare TO 'client'@'localhost';
```

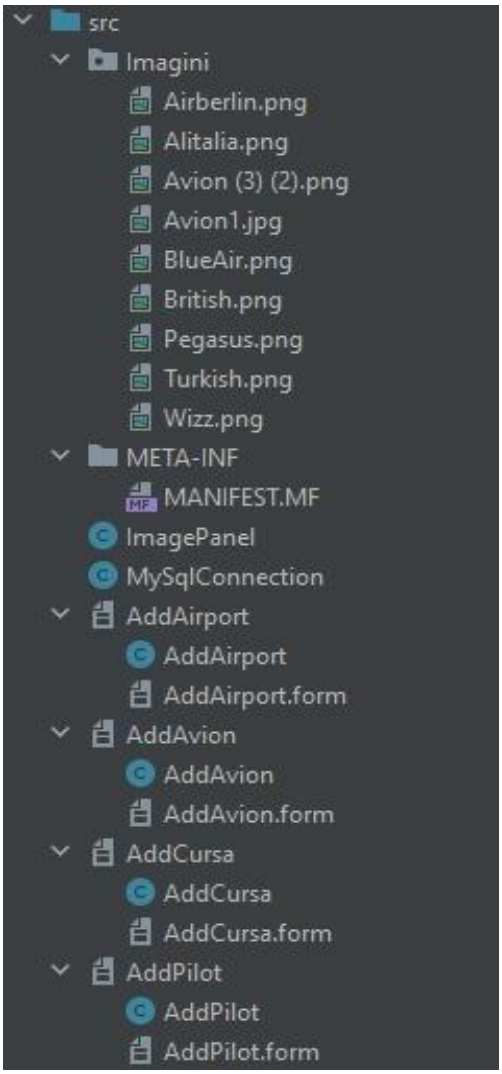
4.Detalii de implementare

4.1 Structura claselor in Java

Diagrama de clase :


pkg
ImagePanel
+ ImagePanel(img: int)
+ ImagePanel(img: int)
+ paintComponent(g: int): void
AddAvion
- numeAvion: int
- numeCompanie: int
model: int
- locuri: int
+ AddAvion(frame: int)
Aeroporturi
+ Aeroporturi(frame: int)
- createUIComponents(): void
AddCursa
- avion: int
- aeroport: int
- suma: int
- plecare: int
- destinatie: int
- avion1: int
+ AddCursa(frame: int)
- convertDateToSql(date: int): int
- createUIComponents(): void
AddAirport
- nume: int
- Adresa: int
- Oras: int
+ AddAirport(frame: int)
Login
- user1: int
- pass1: int
- r: int
+ Login(frame: int)
+ main(args: int[]): void
- createUIComponents(): void
PageAdmin
- a: int
+ PageAdmin(frame: int)
- convertDateToSql(date: int): int
- createUIComponents(): void
AddPilot
+ AddPilot(frame: int)
PageUser
+ PageUser(frame: int, user: int)
- createUIComponents(): void
Search
- text1: int
- text2: int
+ Search(frame: int, text: int, text1: int, user: int)
- createUIComponents(): void
MySqlConnection
+ MySqlConnection()
+ getConnection(): int
DeleteRezervare
- DeleteRezervare(frame: int)
- createUIComponents(): void
VeziRezervari
user1: int
- VeziRezervari(frame: int, user: int)
- createUIComponents(): void
RezervariUser
user1: int
+ RezervariUser(frame: int, user: int)
- createUIComponents(): void
Avioane
+ Avioane(frame: int)
- createUIComponents(): void
Piloti
+ Piloti(frame: int)
- createUIComponents(): void
SignUp
nume: int
prenume: int
- adresa: int
- cnp: int
- telefon: int
- user: int
- parola1: int
- parola2: int
+ SignUp(frame: int)
+ main(args: int[]): void
DeleteUser
+ DeleteUser(frame: int)
- createUIComponents(): void
App
+ App(frame: int)
+ main(args: int[]): void
+ initializ(frame: int): void
- createUIComponents(): void

Dupa cum se poate observa avem un folder in care am retinut imaginile si logo-urile folosite , un folder in care avem clasa pentru conectare la baza de date si clasa pentru imagini iar restul claselor au foloderul lor separat pentru a putea fi mai usor de gasit fiecare .



-folder cu imagini folosite la interfete

-clasa pentru imagini

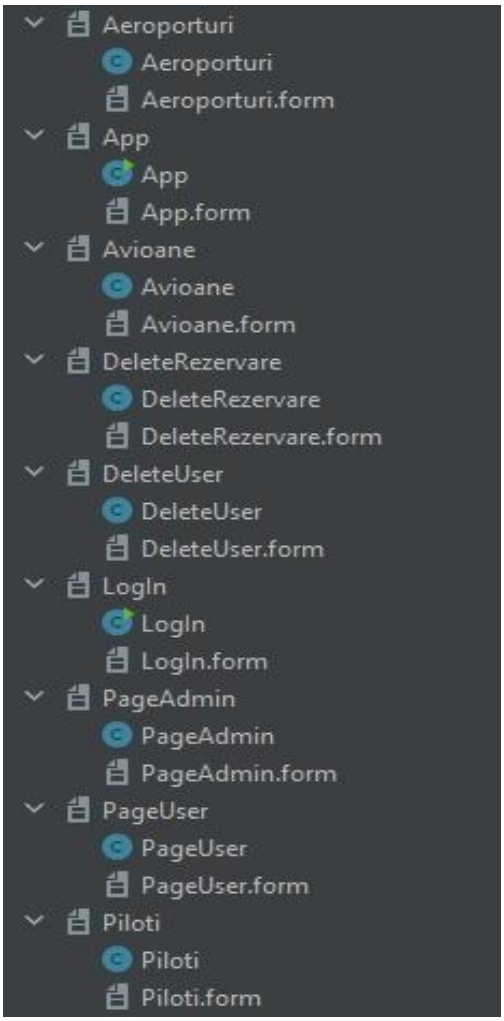
-clasa de conectare la baza de date

-contine adaugarea unui aeroport

-contine adaugarea unui avion

-contine adaugarea unei curse

-contine adaugarea unui pilot



-contine tabel cu aeroporturile

-aplicatia de start(main)

-contine tabel cu avioanele

-contine rezervarile si permite stergerea lor

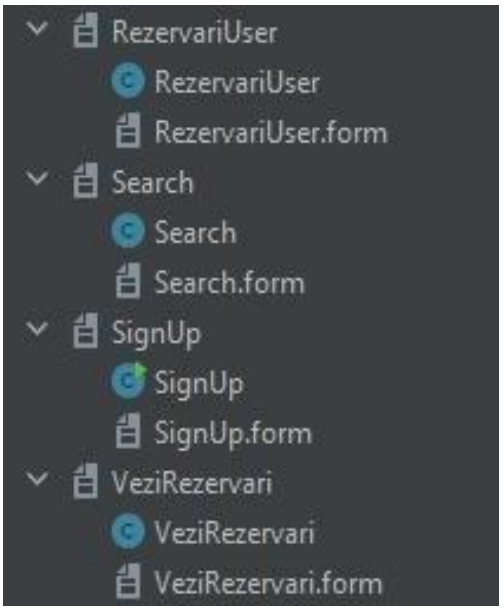
-permite stergerea unui user

-interfata de login

-contine pagina de admin

-contine pagina de user

-contine tabel cu pilotii si permite stergere/adaugare



-contin rezervarile unui user vazute de admin

-afiseaza inregistrarile din tabela rezervari cand cautam dupa orase

-pagina de signup

-pagina de rezervari pentru client

4.2 Manual de utilizare/instalare

Pentru utilizarea aplicatie este nevoie de un mediu de dezvoltare preponderant in java cum ar fi Eclipse sau Intellij .In acesta trebuie sa realizam conexiunea cu baza de date printr-un anumit cod .

La lansarea in executie a aplicatie se deschide o fereastra de bun-venit si de unde putem sa mergem spre pagina de log-in sau pagina de sign-up . Daca mergem pe pagina de sign-up putem sa ne facem un cont de client in aplicatie (conturile de admin se pot face doar de catre admini direct din baza de date) . Daca mergem pe pagina de log-in putem sa ne conectam la aplicatie printr-un cont de client sau printr-un cont de administrator . Daca ne conectam cu contul de Administrator intram in pagina de Admin de unde putem sa manipulam un multe moduri informatiile din baza de date cum ar fi sa adaugam date noi , sa modificam, sa vizualizam si sa stergem . Daca ne conectam la aplicatie cu un cont de client putem sa realizam rezervari la curse si sa anulam rezervarile pe care le avem .

Pentru conectarea ca administrator se pot folosi :"Vlad" si parola "12345" sau "Cristi" si parola "12345".

Pentru conectarea ca client se pot folosi : "Mada" si parola "00000" sau orice cont pe care il cream prin intermediul aplicatiei .

4.3 Elemmente de Securitate a aplicatiei

Aplicatia dispune de un nivel de securitate atat la conectarea la nivelul aplicatiei cat si la nivelul conectarii la baza de date .

In functie de modul de conectare ales (client sau administrator) sunt disponibile urmatoarele operatii :

- Administrator : poate accesa baza de date pentru a adauga curse , avioane , aeropoarte si piloti noi . Poate sa vada orice informatie care se afla in baza de date . Poate sa stearga absolut orice din baza de date (curse , aeropoarte , avioane , piloti , useri) mai putin rezervari care se sterg doar in momentul in care stergem un user si i se sterg rezervarile automat si alti administratori.
- Client : Poate doar sa vada cursele care sunt disponibile , sa faca rezervari pe cursele disponibile si sa anuleze rezervarile sale .

5.Concluzii . Limitari si dezvoltari ulterioare

Toate cerintele au fost respectate si indeplinite , acestea ducand la realizarea unui mediu placut prin care clientii pot interactiona cu o companie aeriana si la un mod convenabil , simplu si rapid de gestiune a datelor de catre administratori .

Aplicatia ofera suport deplin pentru utilizatori pentru a realiza toate interactiunile cu o companie aeriana .

Aplicatia ofera control si acces deplin administratorilor asupra datelor pentru a putea manipula informatiile in cel mai bun mod posibil .

Ca dezvoltare ulterioara mentionez adaugarea unui utilizator de tip pilot astfel incat si acesta sa se poata loga in aplicatie iar in momentul in care un administrator pune o cursa noua , un pilot sa poate sa se angajeze la acea cursa daca nu a facut-o nimeni si daca nu are in orar alta cursa la care s-a angajat in acea data . Acest tip de user ar putea doar sa vada informatii legate de curse si sa se inscrie pentru acestea .

6. Utilizare -Explain pentru detalierea interogarilor

Instructiunea -explain ofera informatii despre modul în care MySQL executa interogarile. La o interogare cu explain vom avea ca raspuns o tabela cu urmatoarele coloane:

Column	JSON Name	Meaning
id	select_id	The SELECT identifier
select_type	None	The SELECT type
table	table_name	The table for the output row
partitions	partitions	The matching partitions
type	access_type	The join type
possible_keys	possible_keys	The possible indexes to choose
key	key	The index actually chosen
key_len	key_length	The length of the chosen key
ref	ref	The columns compared to the index
rows	rows	Estimate of rows to be examined
filtered	filtered	Percentage of rows filtered by table condition
Extra	None	Additional information

1.Pentru interogarea:

```
explain select idcursa, pret, OrasPlecare, OrasDestinatie, LocuriOcupate from orar inner join cursa using(idcursa)
where OrasPlecare="Constanta";
```

Obtinem:

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	orar	NULL	ALL	PRIMARY	NULL	NULL	NULL	5	20.00	Using where
	1	SIMPLE	cursa	NULL	eq_ref	PRIMARY	PRIMARY	4	aeroport.orar.idcursa	1	100.00	NULL

Este descompus in 2 select-uri simple. Primul este slab optimizat deoarece tipul de join este ALL ast insemnand ca nu se foloseste niciun index si este cautata totata tabela.. Al doilea select se face dupa o cheie primara si se foloseste de indexul de cheie primara pentru cautare deci e mai eficient ca celelalalt, eq_ref simbolizeaza ca se face cautare dupa un index folosind operatorul „=”.

2.Pentru interogarea :

```
explain SELECT * FROM rezervare, user where rezervare.iduser=user.iduser;
```

Obtinem :

	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	rezervare	NULL	ALL	user1_idx	NULL	NULL	NULL	2	100.00	Using where
	1	SIMPLE	user	NULL	eq_ref	PRIMARY	PRIMARY	4	aeroport.rezervare.iduser	1	100.00	NULL

Catalogul de sistem

Numele tabelelor si al vederilor
Folosim interogarea:

```
select table_name, table_type, engine from information_schema.tables
where table_schema="aeroport" order by table_type;
```

Obtinem:

	TABLE_NAME	TABLE_TYPE	ENGINE
	cursa	BASE TABLE	InnoDB
	date_avion	BASE TABLE	InnoDB
	orar	BASE TABLE	InnoDB
	pilot	BASE TABLE	InnoDB
	rezervare	BASE TABLE	InnoDB
	user	BASE TABLE	InnoDB
	dateaeroportcursa	VIEW	NULL
	dateavionincursa	VIEW	NULL
	trasee	VIEW	NULL
	viewuser	VIEW	NULL

Numele , domeniile si lungimea atributelor fiecarei relatii
Folosim interogarea:

```
select table_name, column_name, ordinal_position, is_nullable,character_maximum_length, column_type, column_key
from information_schema.columns where table_schema="aeroport" and table_name not like '%ii%'
and table_name not like '%nr_%';
```

Obtinem:

TABLE_NAME	COLUMN_NAME	ORDINAL_POSITION	IS_NULLABLE	CHARACTER_MAXIMUM_LENGTH	COLUMN_TYPE	COLUMN_KEY
aeroport	Adresa	3	YES	45	varchar(45)	
aeroport	idaeroport	1	NO	NULL	int	PRI
aeroport	Nume	2	YES	45	varchar(45)	
aeroport	Oras	4	YES	45	varchar(45)	
cursa	Disponibila	6	YES	NULL	tinyint	
cursa	idavion	2	NO	NULL	int	MUL
cursa	idcursa	1	NO	NULL	int	PRI
cursa	idpilot	7	NO	NULL	int	MUL
cursa	LocuriOcupate	5	YES	NULL	int	
cursa	LocuriRamase	4	YES	NULL	int	
cursa	pret	3	NO	NULL	float	
date_avion	id_avion	1	NO	NULL	int	PRI
date_avion	Model	4	YES	45	varchar(45)	
date_avion	NrLocuriDisponi...	5	YES	NULL	int	
date_avion	Nume	2	YES	45	varchar(45)	
date_avion	NumeCompanie	3	YES	45	varchar(45)	
dateaeropor...	Adresa	4	YES	45	varchar(45)	
dateaeropor...	DATA	6	YES	NULL	date	
dateaeropor...	idaeroport	2	NO	NULL	int	
dateaeropor...	idcursa	1	NO	NULL	int	
dateaeropor...	Nume	3	YES	45	varchar(45)	
dateaeropor...	OraPlecare	7	YES	NULL	time	
dateaeropor...	Oras	5	YES	45	varchar(45)	
dateavioninc...	id_avion	1	NO	NULL	int	
dateavioninc...	idcursa	5	NO	NULL	int	
dateavioninc...	LocuriOcupate	8	YES	NULL	int	
dateavioninc...	LocuriRamase	7	YES	NULL	int	
dateavioninc...	Model	4	YES	45	varchar(45)	
dateavioninc...	Nume	2	YES	45	varchar(45)	
dateavioninc...	NumeCompanie	3	YES	45	varchar(45)	
dateavioninc...	pret	6	NO	NULL	float	
orar	DATA	2	YES	NULL	date	
orar	idaeroport	7	YES	NULL	int	MUL
orar	idcursa	1	NO	NULL	int	PRI
orar	OraDestinatie	6	YES	NULL	time	
orar	OraPlecare	5	YES	NULL	time	
orar	OrasDestinatie	4	YES	45	varchar(45)	
orar	OrasPlecare	3	YES	45	varchar(45)	
pilot	Adresa	5	YES	45	varchar(45)	
pilot	CNP	4	YES	45	varchar(45)	
pilot	idpilot	1	NO	NULL	int	PRI
pilot	nrTelefon	6	YES	45	varchar(45)	
pilot	Nume	2	YES	45	varchar(45)	
pilot	Prenume	3	YES	45	varchar(45)	
rezervare	Adresa	6	YES	45	varchar(45)	
rezervare	Bagaj	10	YES	NULL	tinyint	
rezervare	CNP	7	YES	45	varchar(45)	
rezervare	idcursa	2	YES	NULL	int	MUL
rezervare	idrezervare	1	NO	NULL	int	PRI
rezervare	iduser	9	YES	NULL	int	MUL
rezervare	LocuriAchizition...	3	YES	NULL	int	
rezervare	nrTelefon	8	YES	45	varchar(45)	
rezervare	Nume	4	YES	45	varchar(45)	
rezervare	Prenume	5	YES	45	varchar(45)	
trasee	idcursa	5	YES	NULL	int	
trasee	idrezervare	4	NO	NULL	int	
trasee	iduser	3	YES	NULL	int	
trasee	Nume	1	YES	45	varchar(45)	

trasee	Nume	1	YES	45		varchar(45)	
trasee	Oradestinatie	9	YES	NULL		time	
trasee	OraPlecare	8	YES	NULL		time	
trasee	OrasDestinatie	7	YES	45		varchar(45)	
trasee	OrasPlecare	6	YES	45		varchar(45)	
trasee	Prenume	2	YES	45		varchar(45)	
user	Admin	9	YES	NULL		tinyint	
user	Adresa	6	YES	45		varchar(45)	
user	CNP	7	YES	45		varchar(45)	
user	iduser	1	NO	NULL		int	PRI
user	nrTelefon	8	YES	45		varchar(45)	
user	Nume	4	YES	45		varchar(45)	
user	password	3	YES	45		varchar(45)	
user	Prenume	5	YES	45		varchar(45)	
user	username	2	YES	45		varchar(45)	
viewuser	ID	1	NO	NULL		int	
viewuser	NumeComplet	2	YES	91		varchar(91)	
viewuser	NumeDeUtilizator	3	YES	45		varchar(45)	
viewuser	parola	4	YES	45		varchar(45)	

Constrangerile de integritate

Folosim interogarea:

```
table_name, column_name, constraint_name, referenced_table_name, referenced_column_name
formation_schema.key_column_usage where constraint_schema="Aeroport";
```

Obtinem:

	TABLE_NAME	COLUMN_NAME	CONSTRAINT_NAME	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAME
▶	aeroport	idaeroport	PRIMARY	NULL	NULL
	cursa	idcursa	PRIMARY	NULL	NULL
	cursa	idavion	avion	date_avion	id_avion
	cursa	idpilot	pilot1	pilot	idpilot
	date_avion	id_avion	PRIMARY	NULL	NULL
	orar	idcursa	PRIMARY	NULL	NULL
	orar	idaeroport	aeroport	aeroport	idaeroport
	orar	idcursa	cursa	cursa	idcursa
	pilot	idpilot	PRIMARY	NULL	NULL
	rezervare	idrezervare	PRIMARY	NULL	NULL
	rezervare	idcursa	cursa1	cursa	idcursa
	rezervare	idcursa	cursa2	orar	idcursa
	rezervare	iduser	user1	user	iduser
	user	iduser	PRIMARY	NULL	NULL

Utilizatorii si autorizarile lor

Folosim interogarea:

```
select * from information_schema.schema_privileges where table_schema="Aeroport";
```

Obtinem:

	GRANTEE	TABLE_CATALOG	TABLE_SCHEMA	PRIVILEGE_TYPE	IS_GRANTABLE
▶	'admin'@'localhost'	def	aeroport	SELECT	NO
	'admin'@'localhost'	def	aeroport	INSERT	NO
	'admin'@'localhost'	def	aeroport	UPDATE	NO
	'admin'@'localhost'	def	aeroport	DELETE	NO
	'admin'@'localhost'	def	aeroport	SHOW VIEW	NO
	'client'@'localhost'	def	aeroport	SELECT	NO
	'client'@'localhost'	def	aeroport	SHOW VIEW	NO