UNIVERSITATEA TEHNICĂ “GHEORGHE ASACHI” IAŞI
FACULTATEA DE AUTOMATICĂ ŞI CALCULATOARE
SPECIALIZAREA CALCULATOARE ŞI TEHNOLOGIA INFORMAŢIEI

DISCIPLINA BAZA DE DATE

Gestiunea rezervărilor de vacanță pe croazieră

Coordonator,
Prof. Avram Sorin

Studenți,
Constantin Alexandru
Minea Eduard-Constantin

Iași, 2024

Introducere

Ne propunem ca în acest proiect să simulăm o aplicație de gestionare a unei baze de date a unei agenții care se ocupă cu vacanțe de tip croazieră. Ne dorim să oferim posibilitatea clientului de a își rezerva vacanța dorită într-un mod cât mai ușor și într-o interfață cat mai intuitivă.

Descrierea proiectului

Proiectul este realizat in IDE-ul PyCharm, scris in limbajul Python, folosind plugin-ul PyQT6 si biblioteci din PySide6, precum: QApplication, QMainWindow, QMessageBox, QListWidgetItem și diverse biblioteci de sistem, precum: datetime, hashlib, sqlite3, sys, random.

Conectarea la baza de date se face prin intermediul comenzilor:

```
conn = sqlite3.connect("cruise.db") – deschide database-ul pentru a-i putea fi folosite datele
cursor = conn.cursor() – creaza un obiect ce va fi utilizat pentru rularea a altor comenzi din cadrul sqlite

conn.commit() – aplica schimbarile realizate asupra database-ului
conn.close()  – inchide accesul la database
```

Aspectul generalizat al user interface-ului este alcătuit din 4 ferestre, care prin intermediul unor “push buttons” putem naviga între ele. Astfel la deschiderea aplcației apare fereastra de autentificare unde trebuie introdus un username și o parolă (care a fost criptată la introducerea în database) care se află în baza de date în tabela “users” sau putem înregistra un nou utilizator în fereastra de înregistrare, unde va trebui să își introducă datele personale, care ulterior vor trebuie să treacă de anumite constrângeri pentru a fi validate.

Dupa logare, clientul va selecta un interval de timp și un număr de camere dorite pentru vacanța sa, iar programul îi va afișa opțiunile de camere disponibile pe o anumită navă, împreună cu ruta croazierei și prețul fiecărei camere.

Clientul are și opțiunea de a își vedea rezervările create anterior, cât și de a le anula (șterge) din database prin intermediul ultimei ferestre a programului.

Descrierea funcțională a aplicației

Principalele funcții ale aplicației sunt:

- Evidența clienților
- Evidența navelor și a camerelor
- Evidența tranzacțiilor create

Structura și relațiile dintre entități

1. Entitatea ***users***:
   - id: INTEGER (primary key , autoincrement)
   - username: TEXT (unique , not null)
   - password: TEXT (not null)

2. Entitatea ***user_detail***:
   - id: INTEGER (foreign key to users(id))
   - nume: VARCHAR(50) (not null)
   - prenume: VARCHAR(50) (not null)
   - mail: VARCHAR(50) - (unique , not null)
     - (check like '%_@%_.__%')

3. Entitatea ***ships***:
   - nume: VARCHAR(50) (primary key , not null)
   - data_plecare: DATE
   - destinatie: TEXT (not null)
   - nr_2: INTEGER
   - nr_3: INTEGER
   - nr_4: INTEGER

4. Entitatea ***prices***:
   - id_camera: INTEGER (primary key, autoincrement)
   - nume_nava: VARCHAR(50) (foreign key to ships(nume))
   - tip: INTEGER
   - pret: INTEGER

5. Entitatea ***transactions***:
   - id_client: INTEGER (foreign key users(id))
   - nume_nava: VARCHAR(50) (foreign key to ships(nume), not null)
   - camera_rezervata: TEXT (unique , not null)
   - data_rezervarii: DATETIME
   - pret: INTEGER

Entitățile din această aplicație sunt:

- Clientul (user-ul)
- Date Client
- Navele
- Camerele de pe nave
- Tranzactii

În proiectarea acestei baze de date s-au identificat următoarele tipuri de relații: 1:1 (one-to-one), 1:n (one-to-many).

Relații one-to-one

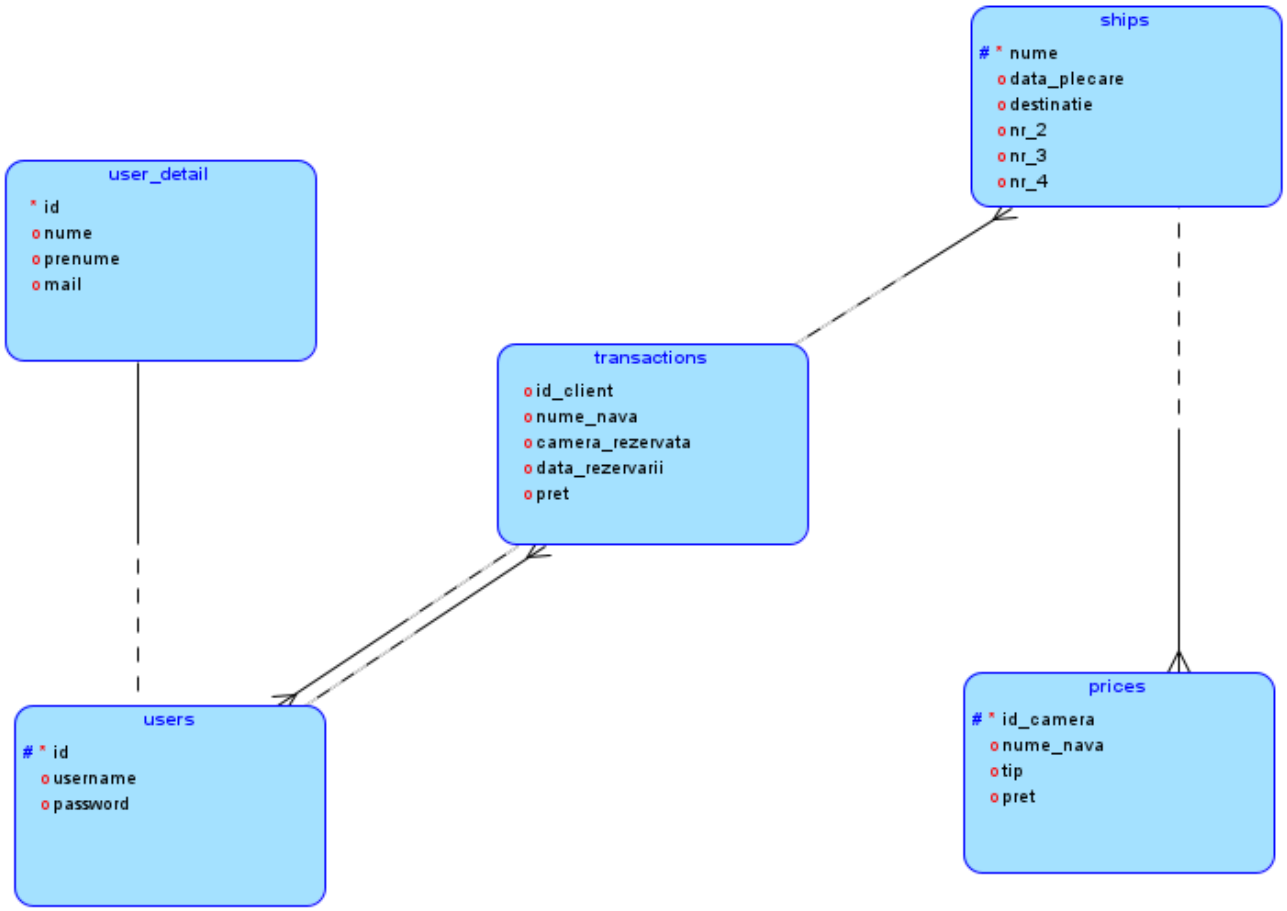
Fiecare client va avea salvate datele personale, nefiind posibilă salvarea a mai multor date personale ale aceluiași client. Legătura între entități se realizează prin atributului “id”.

Relații one-to-many

Între entitatea “nave” și entitatea “camere” se stabileste o relație de 1:n.
O nava poate avea mai multe camere in cadrul sau. Legătura între entități se realizează prin atributul “NumeNava”.

Între entitatea “tranzactii” și entitatea ”nave”, cat si intre entitatea “client” și entitatea ”tranzactii” se stabileste o relație de 1:n.
In cadrul unei tranzactii, un client poate avea mai multe camere rezervate in acelasi timp.
Legătura între entități se realizează prin atributul “camera_rezervata” si “id_client”.

Diagrama ER



Descrierea constrângerilor

În cadrul proiectului am folosit 5 tipuri de constrângeri: PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL și CHECK.

PRIMARY KEY:ele sunt generate de bază de date printr-un mecanism de tip autoincrement (user_id si room_id).

FOREIGN KEY: a fost folosit pentru a indica moștenirea unor parametri din alte tabele, cum ar fi “nume_nave” si “clinet_id-uri”.

UNIQUE: a fost folosit pentru a indica imposibilitatea de existență a doi parametri egali.

NOT NULL:a fost folosit pentru a indica imposibilitatea de existență a unui parametru vid.

CHECK:a fost folosit pentru a indica respectarea unui anumit format de introducere în tabele, cum ar fi atributul mail care trebuie să respecte formatul '%_@%_.__%'

Screenshot-uri cu interfață