A screenshot of a computer program

Description automatically generated

**Arhitectura Aplicației de Ticketing**

Aplicația mea de ticketing este realizată folosind microservicii și este împărțită în 3 componente principale:

1. **Frontend (Client UI)**:
   * Aplicația va fi accesibilă printr-o interfață web (sau mobilă) prin care utilizatorii interacționează cu sistemul.
   * Aici utilizatorii pot crea, urmări, și anula ticketuri, iar managerii pot vizualiza și actualiza ticketurile.
   * Frontend-ul trimite cererile (cum ar fi crearea unui ticket sau obținerea unui ticket existent) către API-ul REST.
2. **API REST (Microserviciul Backend)**:
   * API-ul va expune toate rutele necesare pentru gestionarea ticketurilor (creare, vizualizare, actualizare, anulare).
   * Acesta va utiliza C# cu .NET Core pentru a implementa logica de business.
   * Interacțiunile cu baza de date SQLite se vor face prin acest API.
   * API-ul va include rutele pentru gestionarea resurselor Tickets, Users, și Categories.
   * Acesta va valida cerințele și permisiunile, oferind funcționalități bazate pe roluri (Client User sau Manager).
3. **Baza de Date SQLite**:
   * **Users**: Tabelul va conține datele despre utilizatori (Client User și Manager), cum ar fi nume, email, rol etc.
   * **Category**: Tabelul va defini categoriile de ticketuri (de exemplu, probleme tehnice, solicitări de suport etc.).
   * **Ticket**: Tabelul va conține informațiile detaliate ale ticketurilor, inclusiv titlu, descriere, status (în lucru, rezolvat etc.), utilizatorul asociat și categoria.

**Fluxul aplicației:**

1. Un utilizator (Client User sau Manager) interacționează cu aplicația frontend.
2. Frontend-ul trimite cereri API (cum ar fi POST /tickets, GET /tickets/{id}, PUT /tickets/{id}) pentru a crea, vizualiza sau actualiza ticketuri.
3. API-ul REST procesează cererile și interacționează cu baza de date SQLite pentru a manipula informațiile respective.
4. Baza de date SQLite stochează toate informațiile persistente.

**Containere și Docker-Compose:**

Aplicatia va fi containerizată folosind Docker, cu ajutorul unui fișier **docker-compose.yml** care va lega serviciile dintre ele:

1. **Ticketing API Container**: Conține microserviciul REST.
2. **SQLite Database Container**: Conține baza de date.
3. Utilizarea unui volum Docker pentru a persista datele SQLite între reporniri ale containerelor