EventPlanner

Aplicație web pentru gestionarea bugetului în organizarea de evenimente

Arhitectura sistemului

| **Nume proiect** | EventPlanner |
| --- | --- |
| **Nume document** | Arhitectura sistemului / Architectural Design Document |
| **Autori** | Apetroaei Cezar-Ștefan  Barila Sabina Nadejda  Goriuc Antonia  Marcu Alexia |
| **Reviewed by** | Alexandrescu Adrian Rusu Iuliana-Elena |
| **Status** | Finalizat |

***Versiuni***

| Dată | Versiune | Modificări |
| --- | --- | --- |
| 10.10.2024 | 0.1 | Structura documentului |
| 19.10.2024 | 0.2 | Descriere generală și arhitectura top-level |
| 20.10.2024 | 0.3 | Descrierea componentelor și diagrama ER a bazei de date |
| 21.10.2024 | 0.4 | Descrierea bazei de date |
| 22.10.2024 | 0.5 | Descrierea serviciilor |

# Introducere

## Scopul documentului

Arhitectura sistemului (ADD) conține designul arhitectural pentru platforma **EventPlanner**. Documentul detaliază structura proiectului software, împărțită în multiple componente distincte, fiecare descrisă în detaliu, incluzând relațiile și interacțiunile dintre ele. De asemenea, sunt prezentate toate dependențele software necesare pentru implementarea și funcționarea fiecărei componente în raport cu celelalte, asigurând o integrare eficientă și coerentă a întregului sistem.

## Scurtă descriere a proiectului

**EventPlanner** este o platformă software complexă, concepută pentru a facilita organizarea și gestionarea eficientă a evenimentelor, incluzând atât o aplicație desktop, cât și o interfață web. Aplicația desktop este optimizată pentru a funcționa pe laptopuri / calculatoare care utilizează sistemul de operare Windows / Linux / MacOS.

Întregul sistem, format din aplicația desktop și interfața web, este proiectat și dezvoltat de echipa **Eventures**, având ca obiectiv principal simplificarea procesului de planificare a evenimentelor. EventPlanner facilitează utilizatorilor organizarea evenimentelor prin furnizarea celor mai bune opțiuni adaptate bugetului disponibil, asigurând o experiență optimizată și personalizată.

Platforma oferă o gamă variată de servicii, inclusiv gestionarea invitaților, organizarea agendei, urmărirea bugetului și coordonarea logistică, toate într-o manieră intuitivă și accesibilă. Utilizatorii pot beneficia de instrumente de analiză și raportare pentru a evalua eficiența evenimentelor planificate și a optimiza resursele alocate.

## Definiții și abrevieri

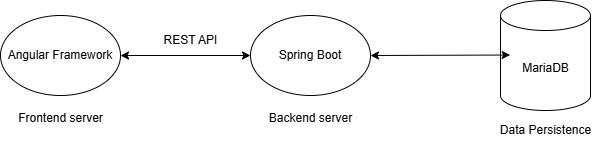
## Eveniment – o ocazie specială sau activitate planificată (ex: nuntă, botez, congres).

## Furnizor – entitate sau persoană fizică ce oferă servicii necesare organizării unui eveniment (ex: catering, foto-video, locație).

## ADD – document ce descrie arhitectura aplicației / Architectural Design Document

# Descriere generală

EventPlanner este o platforma software. Platforma conține o aplicație desktop și o aplicație web care vor fi folosite de utilizatori.



# 2.1 Background

Platforma Event Planner este concepută de echipa Eventures. EventPlanner dorește să îmbunătățească calitatea unui eveniment prin gama largă de oferte cu servicii pentru orice tip de eveniment, personalizate după buget și preferințe. Aceste servicii vor fi disponibile clienților pe aplicația noastră web, servicii adăugate de cei care le prestează, prin aplicația desktop.

# 2.2 Design și context

EventPlanner constă într-o aplicație web și o aplicație desktop. În cadrul aplicației web, utilizatorul poate fi de două tipuri: cel ce caută să furnizorii de servicii pentru organizarea unui eveniment și furnizorii propriu-ziși, care se vor înregistra și își vor expune serviciile pe care le furnizează. De asemenea, este de menționat și existența unui admin care va avea rolul sa accepte sau nu ca serviciile expuse de furnizori să fie disponibile către clienți.

# 2.3 Design

Această secțiune prezintă deciziile luate din punct de vedere al design-ului și al tehnologiilor folosite.

## 2.3.1 Angular

**Angular** este un framework open-source dezvoltat de Google, destinat creării de aplicații web dinamice și scalabile. Este construit pe TypeScript, ceea ce adaugă beneficii precum tipizarea statică și o dezvoltare mai ușoară a aplicațiilor complexe.

Angular promovează o arhitectură bazată pe componente, facilitând organizarea codului și reutilizarea acestuia. Aceasta permite echipelor să lucreze eficient și să mențină codul ușor de gestionat.

Un avantaj al folosirii framework-ului Angular este de a ne împărți task-urile pe componente și putem lucra independent unii fata de altii.

## 2.3.2 MariaDB

**MariaDB** este un sistem de gestionare a bazelor de date relaționale open-source, derivat din MySQL, care oferă o performanță ridicată, scalabilitate și securitate. Acesta păstrează compatibilitatea cu SQL standard și adaugă funcționalități avansate, precum suport pentru stocarea de date în format JSON, replicare avansată și motor de stocare flexibil.

În aplicația noastră de organizare a evenimentelor, vom folosi MariaDB datorită rapidității sale în execuția interogărilor, a securității sporite și a suportului extins pentru funcționalități avansate. MariaDB ne permite să centralizăm informațiile despre evenimente, clienți și furnizori, asigurând o gestionare eficientă și o accesibilitate rapidă la datele necesare pentru luarea deciziilor.

## 2.3.3 Spring Boot

**SpringBoot** este un framework Java open-source, bazat pe Spring Framework, destinat dezvoltării rapide a aplicațiilor enterprise și web. Creat pentru a simplifica configurarea și implementarea aplicațiilor Spring, Spring Boot permite dezvoltatorilor să construiască aplicații standalone, fără a necesita configurații complexe.   
Pentru aplicația EventPlanner vom folosi Spring Boot pentru că Spring Boot simplifică configurarea aplicațiilor, permițând dezvoltatorilor să se concentreze pe cod, și este ideal pentru construirea de microservicii, oferind scalabilitate și flexibilitate în dezvoltare.

## 2.3.4 Electron

**Electron** este un framework open-source dezvoltat de GitHub, care permite crearea de aplicații desktop multi-platformă folosind tehnologii web familiare, precum HTML, CSS și JavaScript. Electron folosește un motor de randare bazat pe Chromium și Node.js, oferind astfel acces complet la API-urile sistemului de operare. Vom folosi Electron pentru aplicatia noastra Desktop deoarece permite utilizarea componentelor Angular, astfel putem reutiliza codul scris pentru cealalta aplicatie client si cu un singur cod putem crea mai multe aplicații, pentru toate sistemele de operare.

## 2.3.5Bootstrap

**Bootstrap** este un framework open-source destinat dezvoltării rapide și eficiente de interfețe responsive pentru aplicații și site-uri web. Creat inițial de Twitter, Bootstrap oferă un set de componente predefinite și stiluri CSS, JavaScript și HTML, facilitând designul uniform și modern. Bootstrap este utilizat pe scară largă datorită simplității, flexibilității și a numeroaselor componente disponibile, fiind ideal pentru dezvoltarea rapidă a site-urilor și aplicațiilor web. Vom folosi Bootstrap deoarece ne va ușura stilizarea platformelor web si desktop.

## 2.3.6 Stilul client-server

Arhitectura client-server este un stil de design utilizat în dezvoltarea aplicațiilor web, în care aplicația este împărțită în două părți principale: **clientul** și **serverul**. În acest model, clientul și serverul comunică între ele printr-o rețea (de obicei, internetul) pentru a furniza funcționalitatea completă a aplicației. Arhitectura client-server este baza majorității aplicațiilor web moderne, asigurând o interacțiune fluentă și eficientă între utilizatori și sistemele din spatele aplicației.

Avantajele acestei arhitecturi includ scalabilitatea, deoarece serverul poate gestiona mai multe cereri simultan, flexibilitatea de a actualiza sau modifica părțile serverului fără a afecta clientul, și centralizarea datelor, ceea ce facilitează gestionarea și securizarea acestora. De asemenea, permite distribuirea eficientă a sarcinilor între client și server, optimizând performanța aplicației și asigurând o experiență de utilizare îmbunătățită.

# Arhitectura top-level

## Identificarea componentelor

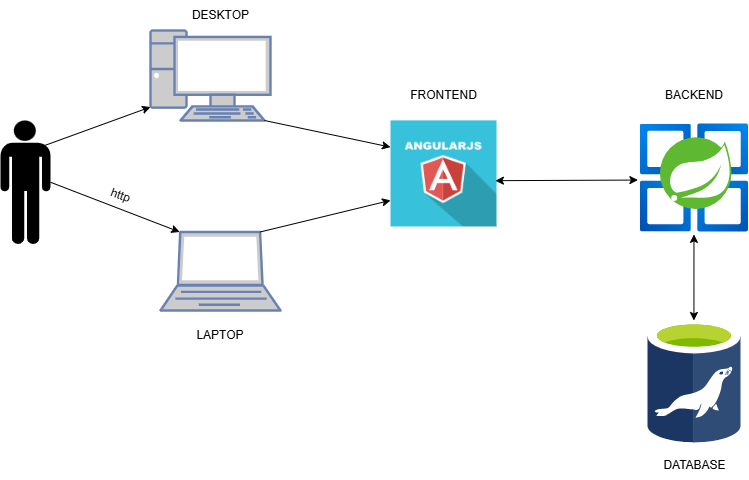
**Front-end:**

* **Aplicația Web (Angular)**: Interfața principală pentru utilizatori, unde aceștia pot naviga prin diferitele tipuri de evenimente, pot căuta furnizori și pot gestiona bugetele.
* **Aplicația Desktop -** aplicația desktop va fi dezvoltată în framework-ul Electron. Ea va permite prestatorilor de servicii sa adauge, sa modifice sau să șteargă serviciile puse la dispoziție pentru eventualii clienți.

**Back-end:**

* **Baza de date -** baza de date va stoca informații despre utilizatori, clienți și furnizori, inclusiv detalii de contact și tipuri de servicii oferite. De asemenea, va înregistra evenimentele planificate, cum ar fi detalii despre ce fel de eveniment este, costurile implicate și serviciile alese. Această structură permite o gestionare mai bună a resurselor și facilitează organizarea eficientă a evenimentelor, asigurându-se că toate informațiile sunt centralizate și ușor accesibile.
* **Aplicația web -** dezvoltată în Java, Spring Boot framework**.**
* **Controlul de acces -** este structurat pe trei roluri: admin, client și furnizor. Adminul are privilegii complete pentru a gestiona serviciile și utilizatorii, inclusiv acceptarea sau respingerea serviciilor furnizorilor. Furnizorii pot adăuga servicii, dar nu au acces la datele altor utilizatori. Clienții pot vizualiza și utiliza doar serviciile acceptate de admin. Această structură asigură un management eficient și protecția datelor.

## Relațiile și comunicația între componente



# Descrierea componentelor

## 4.1 Frontend

4.1.1 **Home page** - va fi pagina principală, care va întâmpina utilizatorii/vizitatorii aplicației noastre web. Aici vor fi prezentate serviciile furnizorilor de pe aplicația noastră (într-o ordine aleatoare), în cazul în care vizitatorul va dori să acceseze un serviciu pentru a vedea mai multe detalii despre acesta sau chiar să înceapă organizarea unui eveniment din template-urile noastre puse la dispoziție pe pagină, acesta va fi informat prin intermediul unui mesaj apărut pe pagină, în care i se va comunica faptul că trebuie să se înregistreze (dacă nu are cont) sau să se logheze (în cazul în care este deja înregistrat pe platformă).

4.1.2 **Pagina de înregistrare** - aceasta conține un formular de înregistrare, care, în dependență de tipul user-ului (utilizator care dorește să organizeze un eveniment, furnizor de servicii sau administrator) va completa formularul respectiv, care constă din câmpuri precum : nume, prenume, email, parolă, număr de telefon, tipul serviciului furnizat (în cazul furnizorilor). Pentru utilizatorii simpli înregistrarea este mai ușoară, întrucât aceștia vor fi redirecționați pe home page-ul aplicației și pot naviga, în timp ce ca și furnizor, dacă adaugi un serviciu, trebuie să aștepți aprobarea administratorului pentru a posta acel serviciu.

4.1.3 **Pagina de logare** - utilizatorii care deja sunt înregistrați în aplicația noastră se pot loga, introducând loginul și parola stabilite în formularul de înregistrare.

4.1.4 **Dashboard** - în cadrul dashboard-ului, utilizatorul are acces la tipul de eveniment pe care l-a selectat, de asemenea serviciile pentru fiecare categorie din evenimentul respectiv. În cadrul dashboard-ului vor fi afișate bugetele, furnizorii și detaliile legate de organizare stabilite de utilizator. În cadrul acestei pagini, utilizatorul va putea să facă rezervare de servicii puse la dispoziție de către furnizorii înregistrați pe platforma noastră.

4.1.5 **Dashboard pentru administrator** - această pagină va fi disponibilă doar administratorului, care poate crea, șterge sau modifica un serviciu prestat de un furnizor de pe platforma EventPlanner. De asemenea, în cadrul acestei pagini, administratorul va putea vizualiza serviciile care sunt în așteptare de a fi aprobate pentru a fi postate, ulterior, pe pagină. Acestea din urmă pot fi acceptate sau refuzate, dacă nu respectă anumite criterii. Administratorul poate vedea toți utilizatorii și furnizorii din cadrul aplicației.

4.1.6 **Dashboard pentru furnizor de servicii** - în cadrul acestei pagini, un furnizor logat în aplicație, va putea să adauge un serviciu, precum și detaliile acestuia (preț, disponibilitatea, etc.).

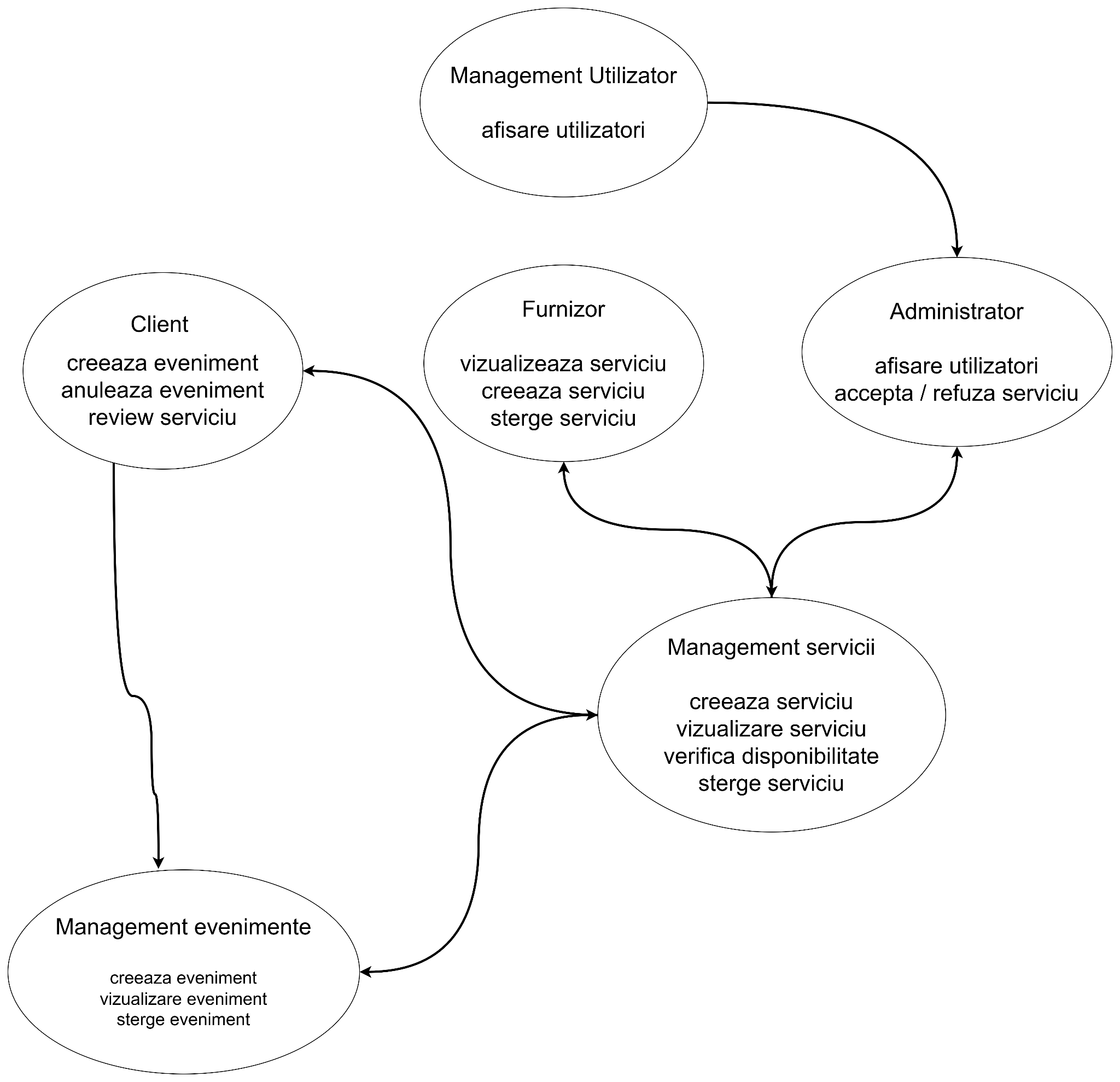
4.1.7 **Pagina de feedback** - în cadrul acestei pagini, utilizatorii care au beneficiat de serviciile prestate de furnizorii de pe aplicația EventPlanner, vor putea lăsa recenzii furnizorilor, sub forma unui feedback care constă în 5 stele - 0 fiind cea mai mică notă, respectiv 5 - cea mai mare.

4.1.8 **Pagină de modificare a datelor de utilizator** - în cadrul acestei pagini, utilizatorul va putea să își modifice informațiile de profil, precum nume, parolă, etc.

4.1.9 **Wishlist** - în cadrul acestei pagini, utilizatorul care își organizează un eveniment poate să adauge un serviciu în wishlist, ca ulterior să poată să compare alegerile și să îl aleagă pe cel mai potrivit pentru eveniment. În cazul în care se răzgândește, poate să șteargă serviciile din wishlist.

4.1.10 **Pagină de efectuare a plății** - după ce utilizatorul și-a ales tipul evenimentului și serviciile din cadrul acestuia, acesta va putea să achite evenimentul respectiv, fiind redirecționat pe pagină de plăți din cadrul aplicației EventPlanner.

## 4.2 Backend



## 4.2.1 Serviciu de management al utilizatorilor



1. Serviciul de management al utilizatorilor
   * Interacțiune cu frontend:
     + Creare cont (secțiunea 4.1.2)
     + Logare (secțiunea 4.1.3)

2. Interacționează cu serviciul de administrator unde acesta poate vedea toți utilizatorii inregistrati în aplicație și detaliile lor publice

3. Interacțiune cu baza de date:

* + Tabela utilizatori
  + Tabela clienți
  + Tabela furnizori

4.2.2 Serviciu de management al serviciilor

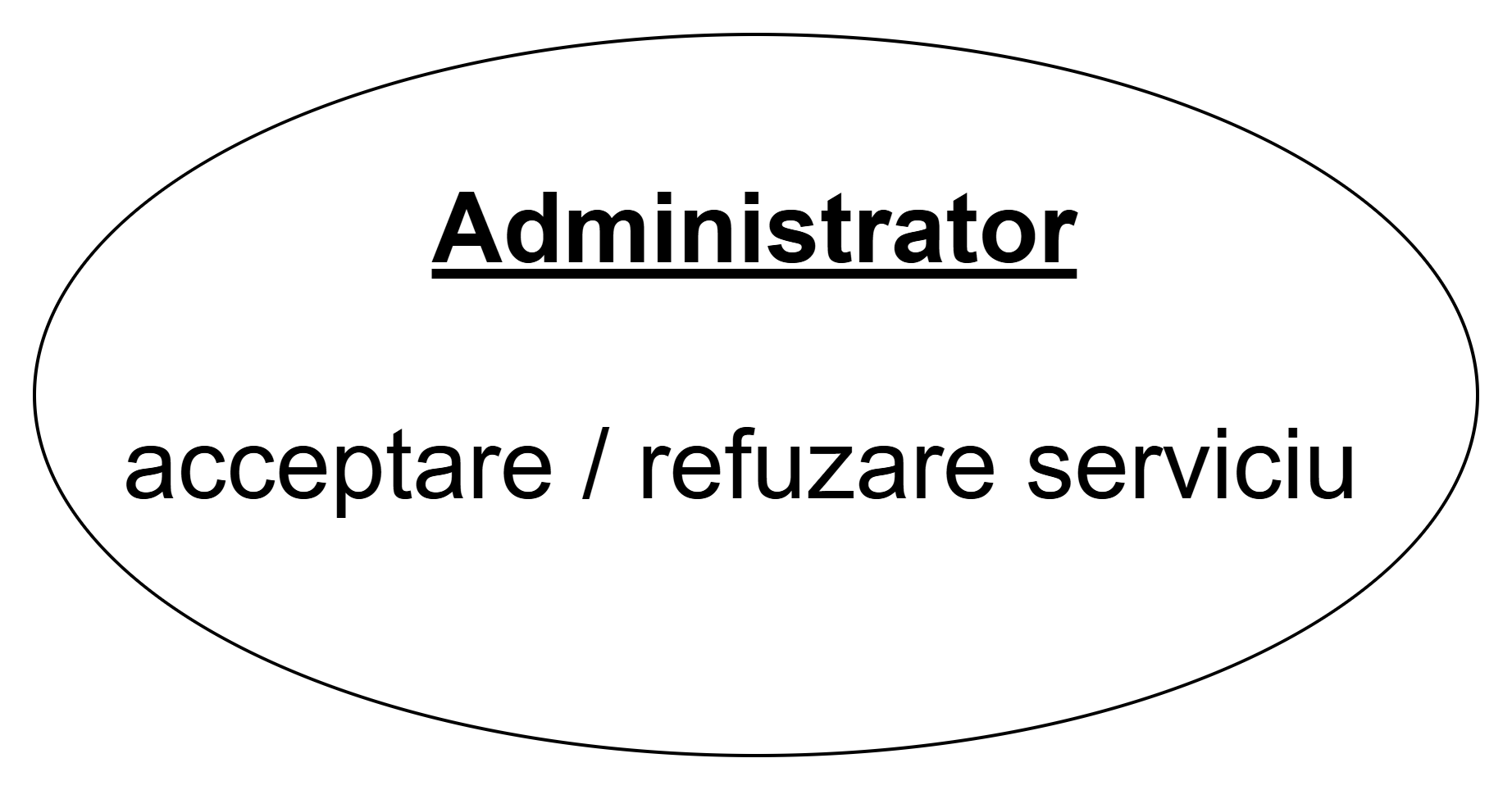


1. Serviciu de management al serviciilor
   * Interacțiune cu frontend:
     + Dashboard admin (secțiunea 4.1.5): Vizibilitate servicii în așteptare
     + Creare eveniment (secțiunea 4.1.10): Vizualizare servicii aprobate
     + Pagina furnizor (secțiunea 4.1.6): Vizualizare servicii furnizor
2. Interacțiune cu serviciu de administrator:
   * Vizualizare servicii în așteptare
   * Acceptare/Refuzare servicii
3. Interacțiune cu serviciu management de evenimente:
   * Pagina de creare eveniment: Modificare disponibilitate serviciu
   * Pagina de vizualizare eveniment creat
4. Interacțiune cu serviciu de clienți:
   * Oferire feedback după eveniment

5. Interacțiune cu baza de date:

* + Inserare serviciu nou (furnizor)
  + Update serviciu (acceptare/refuzare)
  + Update disponibilitate serviciu (clienți - creare/anulare eveniment)
  + Update serviciu ( clientul -> review )

## 4.2.3 Serviciu administrator

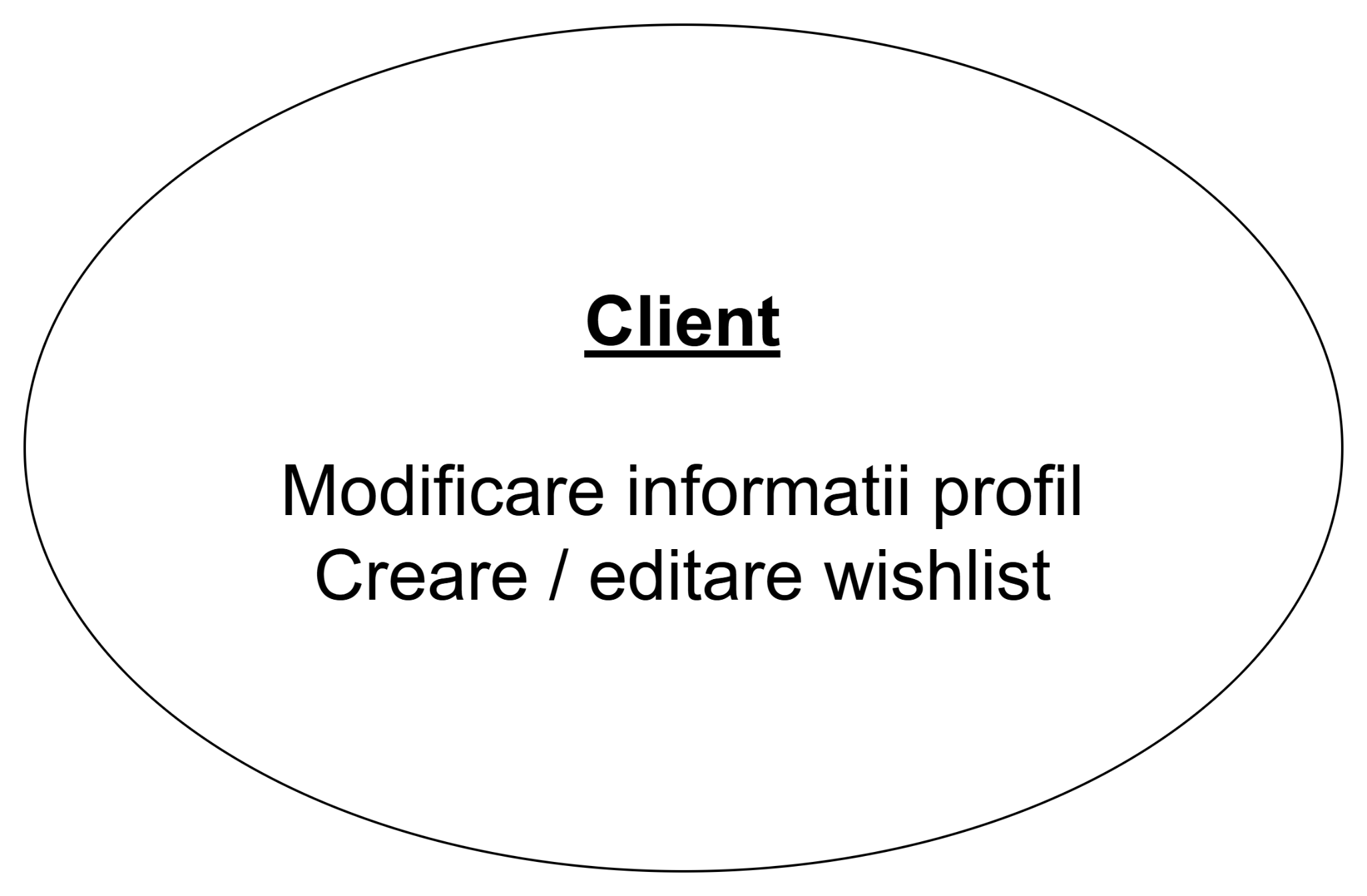


1. Interacțiune cu frontend:
   * + Dashboard admin (secțiunea 4.1.5):
       - Acceptare/Refuzare servicii adăugate de furnizor
2. Interacțiune cu Serviciul de Management Utilizatori:
   * Vizualizare utilizatori înregistrați
   * Vizualizare tip utilizator și informații publice
3. Interacțiune cu Serviciul de Management al Serviciilor:
   * Vizualizare servicii în așteptare
   * Acceptare/Refuzare servicii

Nu interacționează direct cu baza de date:

* + Interacțiunile cu baza de date sunt gestionate de serviciile de mai sus

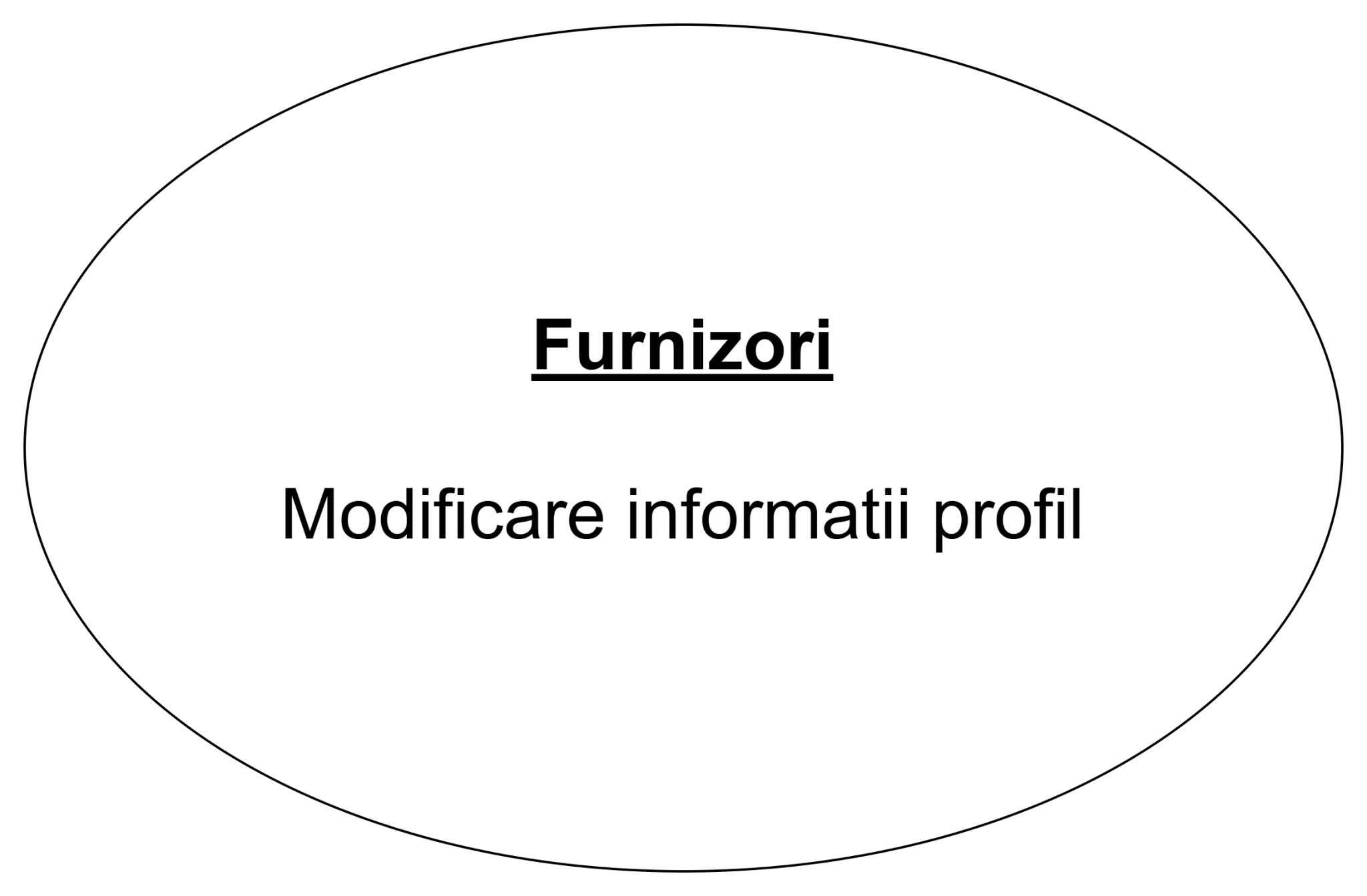
## 4.2.4 Serviciu clienți



* Interacțiune cu frontend:
  + Modificare informații profil (actualizare date utilizator)
  + Creare/Editare Wishlist (gestionare liste de dorințe)
  + Vizualizare servicii rezervate / plătite / anulat de client
  + Vizualizare recenzii oferite de client
* Interacționează cu serviciul management de servicii unde poate vizualiza ce evenimente a rezervat / plătit / anulat ( EVENIMENT = grup de servicii? ) și ce recenzii a oferit evenimentelor
* Interacțiune cu baza de date:
  + Tabela clienți:
    - Update date profil

## 

## 4.2.5 Serviciu furnizori



1. Interacțiune cu frontend:
   * + Modificare informații profil (secțiunea 4.1.X)
     + Vizualizare servicii disponibile/respinse/în așteptare ale furnizorului
2. Interacțiune cu serviciul de management al serviciilor:
   * Vizualizare servicii oferite de furnizor
   * Adăugarea / Editarea / Ștergerea unui serviciu nou în lista proprie
3. Interacțiune cu baza de date:
   * Modifica detalii despre furnizor

## 

## 4.2.6 Serviciu management de evenimente



1. Interacțiune cu frontend:
   1. Pagina de creare a unui eveniment printr-un formular în care se aleg toate serviciile pentru evenimentul dorit.
2. Interacțiune cu alte servicii:
   1. Se actualizează disponibilitatile unui serviciu cand se creaza / șterge / modifica un eveniment
3. Interacțiune cu baza de date:

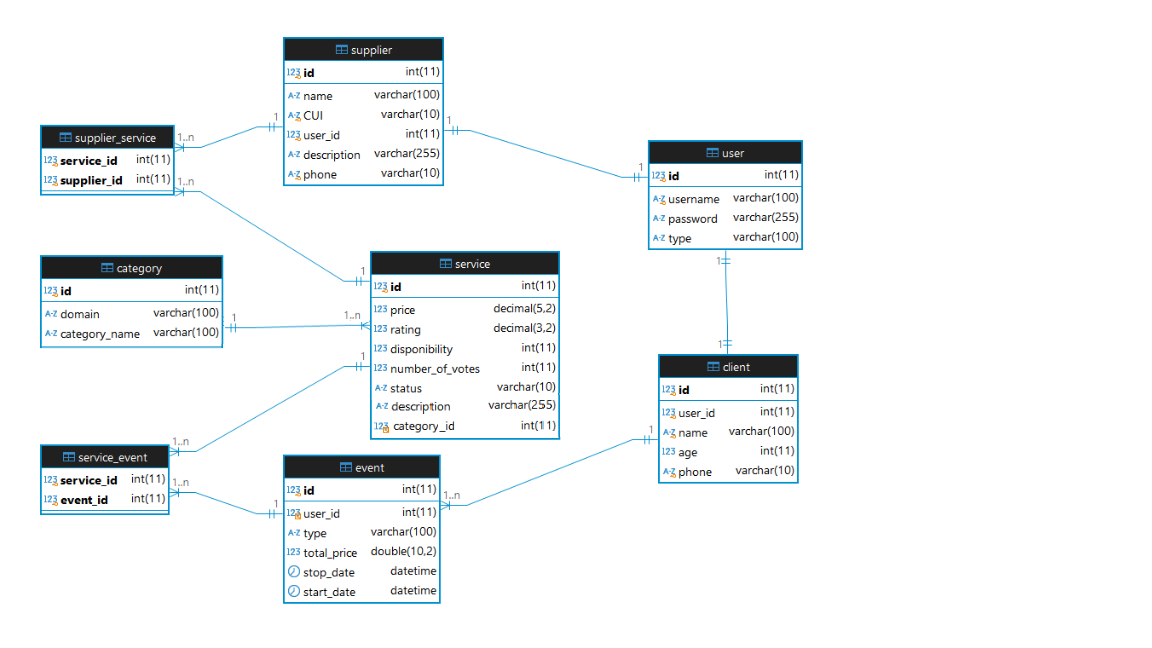
* Tabela utilizatori
* Tabela clienți
* Tabela furnizori

## 4.3 Baza De Date

4.3.1 Descriere și scop

Baza de date susține o platformă de rezervare de servicii pentru evenimente, permițând utilizatorilor să creeze conturi, să rezerve servicii, să gestioneze evenimente și să lase evaluări. Utilizatorii pot fi clienți sau furnizori de servicii, iar datele despre servicii, evenimente și rezervări sunt gestionate centralizat în baza de date.

4.3.2 Diagrama ER a bazei de date



4.3.3 Descrierea tabelelor

### **1. Tabela USER**

* **Descriere**: Această tabelă stochează informațiile utilizatorilor din sistem, indiferent dacă aceștia sunt clienți sau furnizori.
* **Atribute**:
  + id (int) - Cheia primară pentru identificarea unică a unui utilizator.
  + username (varchar) - Numele de utilizator pentru autentificare.
  + password (varchar) - Parola utilizatorului.
  + type (varchar) - Tipul utilizatorului (“CLIENT”, “ADMIN”, “FURNIZOR”).
* **Relații**:
  + Este legat de tabelul client și de tabelul supplier prin câmpul user\_id, indicând că utilizatorul poate fi fie un client, fie un furnizor.

### **2. Tabela CLIENT**

* **Descriere**: Această tabelă stochează detalii despre clienții sistemului, care pot face rezervări la evenimente.
* **Atribute**:
  + id (int) - Cheia primară pentru identificarea unui client.
  + user\_id (int) - Cheie externă care face referință la tabela user (indică care utilizator este client).
  + name (varchar) - Numele clientului.
  + age (int) - Vârsta clientului.
  + phone (varchar) - Numărul de telefon al clientului.
* **Relații**:
  + Relație de tip 1 la 1 cu user prin user\_id.

### **3. Tabela SUPPLIER**

* **Descriere**: Această tabelă stochează informațiile furnizorilor din sistem, care oferă servicii pentru evenimente.
* **Atribute**:
  + id (int) - Cheia primară pentru identificarea furnizorului.
  + name (varchar) - Numele furnizorului.
  + CUI (varchar) - Codul unic de identificare al furnizorului
  + user\_id (int) - Cheie externă care face referință la user (indică care utilizator este furnizor).
  + description (varchar) - Descrierea furnizorului.
  + phone (varchar) - Numărul de telefon al furnizorului.
* **Relații**:
  + Relație de tip 1 la 1 cu user prin user\_id.

### **4. Tabela SERVICE**

* **Descriere**: Această tabelă stochează detalii despre serviciile oferite de furnizori. Fiecare serviciu poate fi rezervat pentru evenimente.
* **Atribute**:
  + id (int) - Cheia primară pentru identificarea unui serviciu.
  + price (decimal) - Prețul serviciului.
  + domain (varchar) - Domeniul serviciului (ex: cofetărie, locație etc.).
  + categories (longtext) - Categoriile serviciului.
  + rating (decimal) - Rating-ul (evaluarea) serviciului.
  + disponibility (int) - Numărul maxim de rezervări care pot fi făcute într-o zi.
  + number\_of\_votes (int) - Numărul de voturi (evaluări) primite de serviciu.
  + status (varchar) - Statusul serviciului (activ/inactiv etc.).
  + description(varchar): Acest câmp poate fi utilizat pentru a detalia sau clarifica ce implică fiecare serviciu.
  + category\_id(int): Cheie externă care face referință la id-ul din tabela category. Pune un serviciu în legătură cu o anumită categorie, pentru ca un serviciu să fie specific unei anumite categorii.
* **Relații**:
  + Relație de tip 1 la N cu supplier prin tabela intermediară supplier\_service.
  + Relație de tip 1 la N cu event prin tabela intermediară service\_event.

### **5. Tabela SUPPLIER\_SERVICE**

* **Descriere**: Aceasta este o tabelă de legătură între furnizori (supplier) și servicii (service).
* **Atribute**:
  + service\_id (int) - Cheie externă care face referință la service.
  + supplier\_id (int) - Cheie externă care face referință la supplier.
* **Relații**:
  + Relație de tip N la N între supplier și service.

### **6. Tabela EVENT**

* **Descriere**: Această tabelă stochează informațiile despre evenimentele la care clienții fac rezervări.
* **Atribute**:
  + id (int) - Cheia primară pentru identificarea unui eveniment.
  + user\_id (int) - Cheie externă care face referință la user (indică utilizatorul/clientul care a făcut rezervarea).
  + type (varchar) - Tipul evenimentului (ex: nuntă, petrecere etc.).
  + total\_price (double) - Prețul total al evenimentului, calculat în funcție de serviciile rezervate.
  + date (datetime) - Data evenimentului.
* **Relații**:
  + Relație de tip 1 la N cu user (prin câmpul user\_id).
  + Relație de tip 1 la N cu service prin tabela intermediară service\_event.

### **7. Tabela SERVICE\_EVENT**

* **Descriere**: Aceasta este o tabelă de legătură între evenimente (event) și servicii (service), reprezentând rezervările de servicii pentru un anumit eveniment.
* **Atribute**:
  + service\_id (int) - Cheie externă care face referință la service.
  + event\_id (int) - Cheie externă care face referință la event.
* **Relații**:
  + Relație de tip N la N între service și event.

**8. Tabela CATEGORY**

* **Atributele relevante**:
  + id (int) - Cheia primară pentru identificarea unei categorii de servicii.
  + domain: Acesta poate reprezenta domeniul general (de exemplu, MANCARE, FOTO-VIDEO, LOCATIE, etc.).
  + category\_name: Acesta definește categoriile specifice pentru fiecare domeniu (de exemplu, CATERING, FOTOGRAF, FLORI, etc.).
* **Relația cu alte tabele**:
  + Tabela category este legată de tabela service prin *category\_id*, deoarece fiecare serviciu oferit de un furnizor poate aparține unei anumite categorii dintr-un anumit domeniu.

4.3.3 Relațiile dintre tabele

* Relațiile 1:1
  + Între **user** și **client** există o relație de one-to-one fiindcă unui cont îi este asociat un singur profil de client.
  + Analog între **user** și **furnizor**, deoarece unui cont îi poate fi asociat un profil de furnizor. Tipul profilului este dat de campul “type” din tabela user.
* Relațiile N:N (many-to-many):
  + Între **supplier** și **service**: Furnizorii pot oferi mai multe servicii, iar serviciile pot fi oferite de mai mulți furnizori. Pentru gestionarea acestei relații, a fost creată o tabelă intermediară denumită supplier\_service. Această tabelă conține două coloane: supplier\_id și service\_id, fiecare fiind o cheie străină care referă la tabela supplier și, respectiv, la tabela service. Astfel, se poate stoca asociația dintre furnizori și serviciile pe care le oferă.
  + Între **event** și **service**: Un eveniment poate include mai multe servicii, iar un serviciu poate fi asociat cu mai multe evenimente. Această relație este gestionată printr-o tabelă intermediară denumită service\_event, avand același principiu ca situația prezentată anterior.
* Relațiile 1:N:
  + Între **category** și **service**: o categorie poate avea mai multe servicii, dar fiecare serviciu aparține unei singure categorii,

### 4.3.4 Conceptele cheie și business logic

* **Gestionarea disponibilității**: Câmpul *disponibility* din tabela service este folosit pentru a limita numărul de rezervări pe care un serviciu le poate primi într-o zi. Dacă acest număr este atins, serviciul nu mai poate fi rezervat pentru acea zi.
* **Rating-ul serviciilor**: Fiecare serviciu are un rating calculat pe baza evaluărilor primite de la clienți, stocat în câmpurile *rating* și *number\_of\_votes*.
* **Finalizarea unui eveniment**: După ce toate serviciile au fost introduse în coșul clientului, acestea vor fi plătite și prin urmare vor fi modificate tabelele event și service\_event și câmpul *disponibility* a fiecărui serviciu cumpărat din tabela service. Va fi necesară reverificarea disponibilității pentru fiecare serviciu introdus înainte de finalizarea plății. Dacă totul decurge corect și evenimentul este setat, clientul primește prin email confirmarea.

### 4.3.5 Interacțiunea cu utilizatorii

* **Autentificare și autorizare**: Utilizatorii se autentifică folosind tabela user. În funcție de câmpul *type*, utilizatorii pot fi fie client, fie supplier. Aceste roluri determină ce acțiuni pot realiza în cadrul platformei (de ex. clienții pot rezerva servicii, iar furnizorii își pot gestiona serviciile).
* **Rezervarea serviciilor**: Clienții pot rezerva servicii pentru evenimente prin intermediul tabelei service\_event.

4.3.6 Constrângeri importante

1. Fiecare utilizator din platformă poate avea doar unul dintre următoarele **tipuri**:

* CLIENT: Un utilizator de tip client poate face rezervări pentru evenimente.
* ADMIN: Utilizatorul de tip admin are permisiuni de gestionare a platformei, inclusiv aprobarea sau respingerea serviciilor.
* FURNIZOR: Un furnizor oferă servicii pe platformă pentru clienți. Furnizorul poate crea și gestiona propriile servicii.

1. Fiecare serviciu oferit de un furnizor trebuie să se încadreze într-unul din **domeniile prestabilite**: MANCARE, FOTO-VIDEO, LOCATIE, ARANJAMENTE, SANATATE, SECURITATE, ENTERTAINMENT
2. Fiecare domeniu poate avea mai multe **categorii specifice**, care definesc tipul exact de servicii pe care le oferă furnizorul. De exemplu:

* Domeniul MANCARE include categoriile: CATERING, PRAJITURI, BAUTURI.
* Domeniul FOTO-VIDEO include categoriile: FOTOGRAF, CAMERAMAN, DRONA, EDITARE VIDEO, FOTOCABINA.
* Domeniul LOCATIE include categoriile: CORT, SALA DE BAL, GRADINA, HOTEL, RESTAURANT, LOCATIE NON-TRADITIONALA(cabane, vapoare, etc.)
* Domeniul ARANJAMENTE include categoriile: FLORI, DECORATIUNI, BALOANE, LUMINI.
* Domeniul SANATATE include categoriile: PERSONAL MEDICAL, AMBULANTA, FITNESS (ex. yoga, masaj, etc.)
* Domeniul SECURITATE include categoriile: PAZA, SUPRAVEGHERE VIDEO, CONTROL ACCES (ex. scanare bilete, verificare identitate, etc.)
* Domeniul ENTERTAINMENT include categoriile: DJ, BAND LIVE, ANIMATOR, MAGICIAN

1. Fiecare serviciu creat de un furnizor are un **status asociat**:

* APROBAT: Serviciul este aprobat de un administrator și este vizibil pentru rezervări.
* REFUZAT: Serviciul a fost respins de administrator și nu este vizibil pe platformă.
* ASTEPTARE: Serviciul este în așteptarea revizuirii de către un administrator.

1. Ratingul unui serviciu este un scor mediu care reflectă evaluările clienților.

* **Valoare permisă**: Ratingul trebuie să fie între 0 și 5, inclusiv. Va fi stocat cu două zecimale pentru a permite mai multă granularitate (ex. 4.25).

1. Serviciile au un câmp de disponibilitate (*disponibility*) care definește câte rezervări simultane (pe aceeași zi) pot fi acceptate. Acesta trebuie să fie mai mare ca 0.

# Concluzii

Arhitectura acestui sistem oferă o soluție scalabilă și flexibilă pentru gestionarea serviciilor și evenimentelor în cadrul unei platforme destinate atât clienților, cât și furnizorilor. Așadar, cu ajutorul componentelor descrise anterior și relațiilor definite în această documentație, s-a realizat și prezentat arhitectura sistemului aplicației EventPlanner.