



**Universitatea Tehnică de Construcții din București**  
**Facultatea de Hidrotehnică**  
**Specializarea: Automatică și Informatică Aplicată**

# **LUCRARE DE LICENȚĂ**

**Coordonator științific:**  
**Dr. ing. Giorgian NECULOIU**

**Absolvent:**  
**Adrian-Gabriel BOȘNEGEANU**

**București, 2025**



**Universitatea Tehnică de Construcții din București**  
**Facultatea de Hidrotehnică**  
**Specializarea: Automatică și Informatică Aplicată**

## **Dezvoltarea unei platforme informatice de tip mobile pentru saloanele de înfrumusețare**

**Coordonator științific:**  
**Dr. ing. Giorgian NECULOIU**

**Absolvent:**  
**Adrian-Gabriel BOȘNEGEANU**

**București, 2025**

# CUPRINS

<b>INTRODUCERE .....</b>	<b>5</b>
Motivarea alegerii temei.....	5
Obiectivele propuse în cadrul lucrării .....	5
Structura .....	6
<b>1. ASPECTE TEORETICE REFERITOARE LA BOOKINGUL DIGITAL .....</b>	<b>7</b>
1.1. Arhitectura sistemului de booking digital .....	7
1.2. Utilizări ale sistemului de booking din perspectiva prestatorului de servicii .....	8
1.2.1. Gestionarea programului de funcționare .....	8
1.2.2. Stabilirea și actualizarea catalogului de servicii și prețuri .....	8
1.2.3. Digitalizarea programatorului .....	8
1.2.4. Documentarea relațiilor cu clienți .....	8
1.3. Utilizări ale sistemului de booking din perspectiva clientului .....	9
1.3.1. Vizualizarea completă a prestatorilor de servicii .....	9
1.3.2. Efectuarea de programări .....	9
1.3.3. Program de loialitate .....	9
1.3.4. Redactarea recenziilor .....	9
1.4. Caracteristicile unui sistem de booking digital .....	10
1.4.1. Accesibilitate .....	10
1.4.2. Documentarea bilaterală a interacțiunilor client - prestator de servicii .....	10
1.4.3. Sistem de feedback.....	10
1.4.4. Notificări .....	10
1.5. Beneficiile bookingului digital.....	10
1.5.1. Beneficiile individuale din perspectiva prestatorului de servicii .....	10
1.5.2. Beneficiile individuale din perspectiva clientului .....	11
1.5.3. Beneficiile colective .....	11
1.5.4. Impactul economic al bookingului digital.....	11
<b>2. TEHNOLOGII SI INSTRUMENTE HARDWARE SI SOFTWARE FOLOSITE IN IMPLEMENTAREA PROIECTULUI .....</b>	<b>12</b>
2.1. Microsoft Windows.....	12
2.2. Visual Studio Code.....	13
2.3. JavaScript .....	13
2.4. React Native .....	14
2.5. Expo .....	14
2.6. Debian Linux.....	15
2.7. API .....	15
2.8. Node.js.....	16
2.9. Google Places .....	16
2.10. Resend .....	17
2.11. SQL .....	17
2.12. MySQL.....	17

2.13. HTML.....	18
2.14. CSS.....	18
<b>3. STUDIU DE CAZ: DEZVOLTAREA UNEI PLATFORME INFORMATICE DE TIP MOBILE PENTRU SALOANELE DE ÎNFRUMUSEȚARE .....</b>	<b>19</b>
3.1. Arhitectura platformei .....	19
3.1.1. Prezentare generală .....	19
3.1.2. Componentele platformei .....	20
3.1.3. Integrarea de servicii externe .....	20
3.1.4. Fluxul de date .....	20
3.2. Rolurile utilizatorilor .....	21
3.3. Navigarea în aplicație.....	21
3.3.1. Navigatorul principal.....	22
3.3.2. Navigatorul dedicat administrării salonului de înfrumusețare .....	22
3.4. Modulele aplicației .....	23
3.4.1. Modulul de logare .....	23
3.4.2. Modulul Acasă .....	24
3.4.3. Modulul de afișare a salonului .....	25
3.4.4. Modulul de programare .....	26
3.4.5. Modulul Programări (client).....	27
3.4.6. Modulul Loialitate.....	28
3.4.7. Modulul Contul meu .....	29
3.4.8. Modulul de actualizare a datelor clientului .....	30
3.4.9. Modulul de vizualizare a istoricului clientului .....	30
3.4.10. Modulul destinat înrolării unui salon .....	31
3.4.11. Modulul Salonul meu .....	31
3.4.12. Modulul Servicii.....	33
3.4.13. Modulul destinat adăugării unui departament .....	34
3.4.14. Modulul Programări (salon).....	34
3.4.15. Modulul istoricului de programări al salonului de înfrumusețare.....	35
3.4.16. Modulul destinat adăugării unei programări din sursă externă .....	36
3.5. Elemente server .....	36
3.5.1. Comunicarea client-server.....	37
3.5.2. Pachete și dependențe.....	37
3.5.3. Baza de date .....	38
3.5.4. Endpointuri API .....	39
3.5.5. Sarcini automate și servicii externe.....	45
<b>4. METODE DE SECURIZARE A DATELOR.....</b>	<b>46</b>
4.1. Certificatul SSL.....	46
4.2. Mascarea datelor pe server .....	46
4.3. Token-ul de utilizator .....	46
4.4. Criptarea datelor în baza de date .....	47
4.4.1. Hashing.....	47
4.4.2. Criptarea flux în binar .....	47
<b>5. SCENARII DE FUNCȚIONARE ALE APLICAȚIEI.....</b>	<b>48</b>

5.1. Efectuarea unei programări – client .....	48
5.2. Recepționarea unei programări - partener .....	50
5.3. Anularea unei programări.....	51
5.4. Acordarea unei recenzii.....	52
5.5. Administrarea unui salon de înfrumusețare .....	53
5.6. Loialitate și vouchere .....	54
<b>Concluzii si contributii personale .....</b>	<b>55</b>
Concluzii .....	55
Contributii personale .....	55
Perspective de viitor .....	56
<b>Bibliografie.....</b>	<b>57</b>

# INTRODUCERE

## Motivarea alegerii temei

În contextul digitalizării accelerate pe care epoca contemporană o impune, fiecare domeniu adoptă soluții moderne pentru a se adapta progresului tehnologic și standardelor curente. Augusta Ada King, Contesă de Lovelace și reprezentantă al sexului frumos este adesea considerată primul programator de calculator din lume. În munca depusă asupra motorului analitic a lui Charles Babbage, a notat următoarea remarcă ce preconiza apariția computerelor moderne: „*Mașina analitică nu are nicio pretenție de a crea ceva original. Ea poate face orice știm noi cum să-i ordonăm să facă. Poate urma o analiză; dar nu are puterea de a anticipa vreo relație sau vreun adevăr analitic.*” (Ada Lovelace, 1843) [1].

În semn de recunoaștere a acestei contribuții remarcabile, lucrarea de față își propune dezvoltarea unei soluții informatice integrate dedicată saloanelor de înfrumusețare și clienților acestora.

## Obiectivele propuse în cadrul lucrării

„Dezvoltarea unei platforme informatice de tip mobile pentru saloanele de înfrumusețare” este o lucrare ce prevede implementarea hibridă, simultan pentru dispozitive ce rulează pe sisteme de operare Android și iOS, a unei aplicații centralizatoare pentru saloanele de înfrumusețare. Această aplicație implică beneficii atât pentru saloanele partenere, cât și pentru clienții acestora. Pentru simplitate și robustețe, atât saloanele partenere cât și clienții pot accesa funcționalitățile dedicate în cadrul aceleiași aplicații, divizată în două module principale.

Din perspectiva unui client, funcționalitatea principală de interes o reprezintă programarea online la serviciile dorite. Clientul poate vizualiza saloanele partenere înregistrate și listate pe platformă alături de serviciile și prețurile pe care le practică și poate opta pentru un loc în programatorul acestuia, pe baza intervalelor orare disponibile. Totodată, clientului i se oferă posibilitatea de a-și anula o programare viitoare, de a vizualiza și acorda feedback sub formă de recenzii asupra unui salon și ca factor de atracție, se prevede implementarea unui sistem de loialitate.

Din perspectiva unui salon de înfrumusețare, interfața aplicației necesită metode de administrare a prezentării salonului în platformă. Profilul unui salon este alcătuit din datele legale ale societății comerciale care îl reprezintă, portofoliul de materiale fotografice și lista de servicii și tarife practicate. Sistemul de „booking” trebuie să permită saloanelor control complet asupra programatorului.

Fluiditatea și promptitudinea întregii interacțiuni client-salon este consolidată de un sistem de notificări menit să țină la curent utilizatorii aplicației în privința oricărei modificări sau acțiuni necesare interlocuțiunii.

Funcționalitățile prezentate anterior implică implementarea unui server ce găzduiește o bază de date relațională ce va fi administrată prin intermediul unor comenzi predefinite accesabile prin intermediul unui API cu multiple endpointuri bine definite.

## Structura

Din punct de vedere structural, lucrarea este compusă din cinci capitole numerotate și o secțiune conclusivă ce marchează impresiile personale asupra procesului de elaborare, contribuțiile și perspectivele de dezvoltare viitoare.

Primul capitol se axează pe prezentarea teoretică a conceptului de „booking online” și prezintă beneficiile bilaterale ale digitalizării relației dintre client și prestatorii de servicii, în special în domeniul serviciilor de înfrumusețare.

Al doilea capitol prezintă tehnologiile utilizate în dezvoltarea întregii platforme și limbajele de programare aferente. Limbajul de programare utilizat preponderent este JavaScript, iar mediul de dezvoltare principal este Visual Studio Code.

Capitolul trei prezintă funcționalitățile aplicației din interfața utilizator, dar și logica din backend care îndeplinește execuția comenzilor de la utilizator.

Al patrulea capitol prezintă tehnicile prin care este menținută securitatea datelor. Informațiile sensibile sunt protejate prin stocarea lor sub formă criptată, iar comunicarea acestora se face pe canale securizate.

Ultimul capitol descrie produsul final. Modul și scenariile de utilizare sunt prezentate succint și sugestiv, astfel încât orice utilizator ce parcurge această secțiune să poată accesa întregul potențial al aplicației.

# 1. ASPECTE TEORETICE REFERITOARE LA BOOKINGUL DIGITAL

Bookingul digital reprezintă sistemele modern de rezervare online ce permit planificarea automată a întâlnirilor pentru prestări de servicii, vânzări sau evenimente fără a fi necesară interlocuțiunea umană. Platformele informatice ce deservesc acestui serviciu devin din ce în ce mai populare în toate domeniile, precum saloane de înfrumusețare, clinici medicale, centre sportive sau evenimente culturale. Ele contribuie semnificativ atât la creșterea satisfacției clienților prin reducerea timpilor de așteptare, cât și la diminuarea erorilor și automatizarea fluxului de lucru al societăților care le implementează.

Un instrument de planificare online facilitează vizualizarea disponibilității din calendar de către utilizatori. Un sistem de rezervare online poate împiedica, de asemenea, rezervarea dublă, cu disponibilitate în timp real. [2]

## 1.1. Arhitectura sistemului de booking digital

Un sistem de booking digital este compus dintr-un ansamblu de componente software și infrastructuri tehnologice ce interacționează în vederea automatizării sistemului clasic de programare. Uzual, un astfel de sistem implică o arhitectură în trei straturi sau „3-Tire Architecture”.

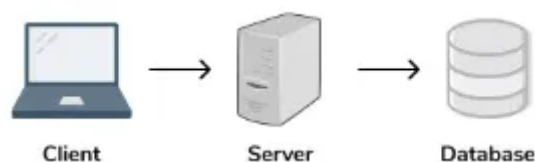


Fig 1.1. Reprezentarea schematică a 3-Tire Arhitecture

Arhitectura în trei straturi este o arhitectură software client-server ce separă o aplicație în trei niveluri diferite în scopul îmbunătățirii modularității, scalabilității, fiabilității și flexibilității sistemului software. [3]

Stratul Client constă în interfața prin care utilizatorului platformei îi sunt prezentate informațiile și modurile în care poate interacționa cu ele în mod intuitiv.

Stratul Server conține logica de funcționare a platformei și execută comenzile de manipulare a datelor conform acestei logici implementate.

Stratul Database este baza de date în care sunt stocate informațiile necesare funcționării platformei informatice.



## 1.2. Utilizări ale sistemului de booking din perspectiva prestatorului de servicii

În ceea ce privește prestatorii de servicii, o platformă informatică de booking digital aduce beneficii organizatorice prin promptitudine, eliminarea riscului erorii umane și printr-o gestiune centralizată și automatizată a programatorului.

### 1.2.1. Gestionarea programului de funcționare

Prestatorii își pot configura programul de lucru în mod personalizat, incluzând orele de funcționare ale locației, programul pe departamente, pauzele, zilele libere sau vacanțele. Această flexibilitate permite actualizări în timp real, reducând necesitatea confirmărilor telefonice sau reprogramărilor neprevăzute.

### 1.2.2. Stabilirea și actualizarea catalogului de servicii și prețuri

În procesul de configurare al profilului unui partener într-o soluție software de booking, catalogul poate fi optimizat complet prin denumirea, durata, tariful și descrierea fiecărui serviciu oferit. Astfel, clienții pot fi atrași printr-o prezentare transparentă și clară a ofertei comerciale pe care o societate o prestează.

### 1.2.3. Digitalizarea programatorului

Programatorul clasic, adesea gestionat pe hârtie și ambiguizat prin mesaje și apeluri telefonice, adnotări sau tăieturi neinteligibile, poate fi digitalizat printr-un calendar interactiv. Un astfel de calendar digital, accesibil atât prestatorilor de servicii cât și clienților, are un impact major asupra modului de organizare a activității.

Conform unei centralizării al studiilor organizate asupra impactului programărilor online în mediul sanitar din China, „Impact of digital self-scheduling on operations management and patient experience in hospital outpatient settings: a systematic review and meta-analysis” [4], se remarcă scăderi semnificative ale timpilor de așteptare. Pe un eșantion de 832 de pacienți, programarea digitală a redus timpul de așteptare cu aproximativ 22-25% față de metodele de programare tradiționale.

### 1.2.4. Documentarea relațiilor cu clienți

Unul dintre cele mai mari avantaje ale digitalizării în domeniul serviciilor este posibilitatea stocării și gestionării eficiente a informațiilor despre clienți. Avansul tehnologic a redus considerabil costurile asociate cu stocarea datelor, permițând în prezent arhivarea unui volum mare de informații într-un mod organizat și accesibil.

Într-o platformă digitală, fișele clasice de clienți adesea greu de accesat și dificil de actualizat sunt înlocuite cu profiluri electronice care se actualizează automat la fiecare programare. Sistemul poate centraliza datele clienților precum frecvența vizitelor, tipurile de servicii solicitate, valoarea tranzacțiilor sau preferințele personale, fără a mai fi necesară introducerea manuală a acestor informații.

Pe baza acestor centralizări, prestatorii pot identifica serviciile cele mai solicitate, departamentele cele mai profitabile și pot lua decizii informate pentru îmbunătățirea ofertelor. Totodată, pot implementa strategii de loializare, precum trimiterea de vouchere personalizate sau oferte speciale clienților fideli, consolidând astfel relația cu aceștia și stimulând revenirea lor în salon.

### 1.3. Utilizări ale sistemului de booking din perspectiva clientului

Din perspectiva clientului, sistemele digitale de booking contribuie la o experiență modernă, eficientă și personalizată. Principalul beneficiu este accesibilitatea permanentă, fără a depinde de programul de lucru al prestatorilor, permițând vizualizarea și efectuarea de programări în afara orelor convenționale și eliminând necesitatea interacțiunii cu personalul.

#### 1.3.1. Vizualizarea completă a prestatorilor de servicii

Toate informațiile de interes sunt stocate în server și accesibile prin intermediul soluției software. Astfel, interacțiunea umană inefficientă este eliminată, facilitând luarea unei decizii informate și lipsite de omisiuni. În permanență sunt puse la dispoziție și actualizate: serviciile oferite alături de durata lor și tarifele practicate, detalii despre locație, portofolii fotografice, recenzii ale altor clienți și chiar și detaliile de natură legală despre persoana juridică din spatele listării.

Încrederea clientului este stimulată de transparența și imparțialitatea pe care mediul digital le impune prin natura sa obiectivă, ghidată de instrucțiunile din codul sursă al platformei.

#### 1.3.2. Efectuarea de programări

Funcția centrală a unui astfel de sistem este posibilitatea de a vizualiza disponibilitatea și a selecta data și ora dorită pentru o anumită procedură. Deși se reduce semnificativ riscul de erori în program, adesea se impune necesitatea unei confirmări din partea prestatorului. Astfel, conflictele cauzate de programări din surse externe sunt evitate.

Programările pot trece prin mai multe stări: înregistrată (cererea pentru rezervare a fost transmisă către prestator), confirmată (prestatorul a verificat disponibilitatea și a confirmat rezervarea), în curs (clientul s-a prezentat și operațiunile sunt în desfășurare) și finalizată.

#### 1.3.3. Program de loialitate

În unele cazuri, funcționalitățile unei platforme de booking digital se extind printr-un program de loialitate menit să fidelizeze prin recompense bazate pe istoricul clientului. Beneficiile unui astfel de mecanism încurajează întoarcerea clienților și creează o relație de durată între aceștia și prestaotri.

#### 1.3.4. Redactarea recenziilor

Odată ce programările ajung în stadiul de finalizate, clienții sunt îndrumați spre a acorda o recenzie asupra interacțiunii și serviciilor efectuate. Mai multe studii, printre care și „Evaluating the Influence of Customer Reviews and Consumer Trust on Online Purchase Behavior” (*Imran Uddin*, 2025) [5], atestă că recenziile din mediul online influențează decizia clienților indiferent de gen. Dacă pentru persoanele de gen masculin relevanța constă în media finală a punctajului rezultat din recenzii, persoanele de gen feminin acordă o atenție semnificativ mai mare comentariilor negative luând în calcul totoodată și modul în care prestatorul gestionează și răspunde reacțiilor negative.

Serviciul de feedback este important pentru clienți în luarea unei decizii, iar înțelegerea diferențelor de gen menționate anterior poate ajuta și societățile listate într-o platformă de booking, în funcție de publicul lor țintă.

## **1.4. Caracteristicile unui sistem de booking digital**

Sisteme moderne de booking digital devin din ce în ce mai populare în toate domeniile pieței. Astfel de soluții sunt construite spre optimizarea timpului și conturării unei experiențe eficiente și lipsite de erori atât pentru societățile comerciale, cât și pentru clienții care le utilizează.

### **1.4.1. Accesibilitate**

Accesibilitatea este trăsătura fundamentală a unui sistem de programare online. Utilizatorii pot accesa platforma de pe diverse dispozitive și din orice locație cu conexiune la internet. Interfața trebuie să fie intuitivă, ușor de utilizat de persoane cu orice nivel de pregătire, iar în cazul terminalelor cu sisteme de operare moderne, o gamă largă de dizabilități este combătută prin soluții software ingenioase.

### **1.4.2. Documentarea bilaterală a interacțiunilor client - prestator de servicii**

Un sistem de rezervări eficient păstrează evidența tuturor interacțiunilor dintre clienți și prestatorii de servicii. Toate detaliile legate de rezervările efectuate sunt stocate și accesibile din conturile utilizatorilor. Acest element este esențial pentru societățile comerciale ce utilizează fișe de clienți în perfecționarea interacțiunilor viitoare.

### **1.4.3. Sistem de feedback**

Integrarea unui sistem de feedback este cât se poate de populară în rândul platformelor informatice de acest tip. Funcționalitatea conferă vizibilitate asupra calității serviciilor, stimulează prestatorii în menținerea unui standard ridicat în interacțiunea cu clienții și poate avea efect decizional în rândul utilizatorilor ce-și decid următoarea programare.

### **1.4.4. Notificări**

Nelipsite în majoritatea soluțiilor software mobile, notificările automate mențin utilizatorii informați cu privire la evenimentele programate. Notificările pot fi transmise prin email sau SMS, dar cea mai eficientă este metoda push-notification prin care nevoia de confirmare a programării sau eventualele modificări de ultim moment sunt aduse rapid și eficient în atenția destinatarului.

## **1.5. Beneficiile bookingului digital**

Trecerea la sisteme digitale de programare aduce mari beneficii în modul de interacțiune dintre clienți și prestatorii de servicii. Deși există reticente asupra tranziției din cauza lipsei de familiaritate cu tehnologia sau a neîncrederii în sistemele informatice, acestea sunt depășite pe măsură ce beneficiile devin evidente.

### **1.5.1. Beneficiile individuale din perspectiva prestatorului de servicii**

Reducerea timpului administrativ este primul beneficiu evident al utilizării unei platforme online de programare. Astfel, metodele clasice de programare prin preluarea de apeluri sau mesaje sunt depășite de eficiența bookingului digital.

Neprezentările sunt diminuate datorită notificării automate, care reamintește utilizatorilor despre programările viitoare. Totodată și prestatorii de servicii beneficiază de notificare înaintea

unei programării. În plus, în cazul unei schimbări de plan din partea clientului, anularea programării se efectuează facil din aplicație, eliminând disconfortul asocial apelurilor directe și promovând o comunicare mai eficientă și asumată.

Vizibilitatea unui prestator este sporită prin simpla prezență într-o platformă digitală, accesibilă de oriunde prin conexiunea la internet. Această vizibilitate poate fi întărită de reputația obținută prin recenzii sau prin reduceri și campanii atractive.

Analiza activității devine mai facilă prin accesul constant la istoricul de programări.

#### 1.5.2. Beneficiile individuale din perspectiva clientului

Flexibilitatea oferită de o platformă digitală, accesibilă permanent, fără a depinde de programul de lucru al prestatorului, oferă comoditate și control asupra întregului proces de programare și gestionare a programărilor.

Transparența și obiectivitatea tarifelor practicate consolidează încrederea clienților și elimină situațiile neplăcute de tarifare incorectă sau nedreaptă.

Comunicarea este mult mai clară, ghidată de acțiuni și mesaje prestabilite cu informații ce rămân întipărite și pot fi reacesate la nevoie.

#### 1.5.3. Beneficiile colective

La nivel colectiv, beneficiile sunt bilaterale și contribuie la gradul de profesionalism și la o îmbunătățire clară a experienței generale.

Din punct de vedere organizatoric, platformele de booking impun o structură clară, standardizată a procesului de interacțiune. Acest cadru bine definit elimină ambiguitatea, omisiunile de informații atât din partea clienților cât și a prestatorilor de servicii. Totodată, o astfel de soluție automatizată încurajează la definirea unui portofoliu clar și la respectarea termenelor declarate. Pentru clienți, această coerență generează predictibilitate și încredere, reducând riscul experiențelor neplăcute.

Liberul acces în aplicație oferă posibilitatea ca societățile cu posibilități mai reduse să concureze cu francize mari oferind calitate la prețuri mai reduse.

#### 1.5.4. Impactul economic al bookingului digital

Pe plan economic, digitalizarea rezervărilor stimulează creșterea întregii piețe. Automatizarea sistemului de programări prevede o structură solidă a programatorului și promovează reducerea de intervale neproductive, crescând numărul de clienți deserviți.

Însăși digitalizarea ca proces general demonstrează beneficii economice substanțiale. Banca Centrală Europeană atestă prin studiul „Digitalisation and the economy”[6] că în comerțul electronic, ajustarea prețurilor este de două ori mai frecventă decât în retailul tradițional.

Prin crearea unui mediu favorabil creșterii economice pe un întreg segment, se favorizează evoluția operatorilor economici de dimensiuni mici și încurajează apariția de noi anteprenori. Astfel, întreg segmentul poate fi balnsat și se reduce riscul de monopol. Prin accesibilitatea rapidă la mai multe portofolii și cataloage de prețuri, se sporește concurența și se menține un preț corect și echilibrat pentru fiecare serviciu.

## 2. TEHNOLOGII SI INSTRUMENTE HARDWARE SI SOFTWARE FOLOSITE IN IMPLEMENTAREA PROIECTULUI

Implementarea unei platforme digitale de tip booking presupune utilizarea unei game moderne și coerente de tehnologii software și hardware care să permită dezvoltarea eficientă, scalabilă și sigură a aplicației. Pentru proiectul de față s-a optat pentru o soluție de dezvoltare hibridă, care să permită generarea unei aplicații mobile compatibile cu principalele sisteme de operare, iOS și Android, pornind de la o bază unică de cod. Acest model a fost ales pentru a reduce considerabil timpul și costurile de dezvoltare, dar și pentru a simplifica procesul de mentenanță și actualizare ulterioară a platformei.

Setul de instrumente utilizat în cadrul proiectului include medii de dezvoltare moderne, framework-uri performante pentru interfața cu utilizatorul și pentru logica de server, precum și sisteme de gestiune a bazelor de date relaționale. Acestea sunt integrate într-o arhitectură full-stack, în care limbajul principal utilizat este JavaScript, alături de extensiile și tehnologiile asociate, cum ar fi SQL pentru manipularea datelor, precum și HTML și CSS. Această alegere unificată a permis un flux de dezvoltare coerent, în care toate componentele aplicației pot fi gestionate într-un mod integrat și eficient.

### 2.1. Microsoft Windows

Microsoft Windows este o familie de sisteme de operare dezvoltate de compania Microsoft, care oferă o interfață grafică și funcționalități pentru gestionarea resurselor hardware și software ale unui computer. Lansat inițial în noiembrie 1985 ca o extensie grafică pentru MS-DOS, Windows a evoluat într-un sistem de operare complet, devenind unul dintre cele mai utilizate la nivel mondial. Versiunile moderne, precum Windows 11, oferă suport pentru multitasking, securitate avansată și compatibilitate cu o gamă largă de aplicații și dispozitive. [7]

Microsoft Windows reprezintă sistemul de operare utilizat în cadrul proiectului pentru desfășurarea activităților de dezvoltare software, în special pentru partea de codare, testare și structurare a aplicației. Popularitatea acestui sistem de operare în mediile de dezvoltare este justificată de compatibilitatea sa extinsă cu numeroase medii de dezvoltare integrate (IDE), framework-uri și unelte de programare.

Detaliile tehnice hardware ale terminalului utilizat în implementare:

- Procesor: Intel® Core™ i7-9750H
- Memorie RAM: 16 GB
- Arhitectură x64-bit

Specificațiile sistemului de operare Microsoft Windows:

- Ediție: Microsoft Windows 11 Pro
- Versiune: 23H2
- Build: 22631.5189

## 2.2. Visual Studio Code

Visual Studio Code este un mediu de dezvoltare foarte potent, proiectat spre a facilita redactarea de aplicații web, mobile și cloud folosind limbaje de programare valabile în diferite platforme de dezvoltare. Cu Visual Studio Code se poate lucra cu fișiere individuale de cod sau cu întregi structuri de fișiere bazate pe foldere. [8]

Adesea prescurtat VS Code, este mai mult decât un simplu editor de cod sursă, ci un mediu de lucru versatil și modular dezvoltat de Microsoft. Alegerea acestui mediu de dezvoltare este strâns legată de facilitățile pe care le aduce în procesul creării unei soluții software fullstack.

Interfața intuitivă a VS Code permite navigarea facilă între fișiere și proiecte din mediul local sau stocate pe servere remote. Prin acordarea credențialelor de conectare către un server extern, putem avea acces simultan la întregul sistem de fișiere dintr-un server. Modularitatea este promovată de suportul pentru extensii dedicate diferitor limbaje de programare sau framework-uri.

Terminalul integrat permite rularea de comenzi fără a părăsi editorul. VS Code include soluții pentru executare atât pentru comenzile PowerShell sau cmd necesare pentru dezvoltarea de software în sistemul de operare Windows, cât și pentru comenzile ce se impun programării unui server Linux remote sau administrării unei baze de date la distanță.

În cadrul proiectului, VS Code a fost principalul mediu de lucru pentru dezvoltarea interfeței aplicației în React-Native, crearea și testarea rutelor API în Node.js, gestionarea scripturilor SQL pentru baza de date MySQL, execuția comenzilor în terminale și organizarea fișierelor și a structurii proiectului.

La momentul implementării, s-a utilizat versiunea Microsoft Visual Studio Code 1.100.2.

## 2.3. JavaScript

JavaScript este un limbaj de programare de nivel înalt, interpretat, cu o sintaxă flexibilă și o comunitate extinsă de dezvoltatori, fiind una dintre tehnologiile centrale ale dezvoltării web moderne. Utilizat inițial pentru a adăuga interactivitate paginilor web, JavaScript a evoluat într-un ecosistem matur care permite dezvoltarea completă a aplicațiilor atât pe partea de client (frontend), cât și pe partea de server (backend) [9].

În cadrul proiectului, JavaScript a fost limbajul principal folosit pentru logica aplicației. Pe partea de client, JavaScript este limbajul de bază în cadrul framework-ului React Native, care a fost folosit pentru dezvoltarea interfeței mobile. Utilizând componente și sintaxă JSX (JavaScript XML), dezvoltatorii pot construi interfețe declarative și reutilizabile, asigurând o experiență nativă atât pentru utilizatorii Android, cât și pentru cei iOS.

Pe partea de server, JavaScript este utilizat în cadrul mediului de execuție Node.js. Acesta permite rularea codului JavaScript în afara browserului, pe server, și oferă suport nativ pentru operații asincrone, lucru esențial într-o platformă de booking unde interacțiunea cu baza de date și gestionarea cererilor multiple de la utilizatori trebuie să fie eficiente și rapide.

Un avantaj major al utilizării JavaScript în ambele componente ale aplicației (frontend și backend) este omogenitatea limbajului, care contribuie la reducerea timpului de dezvoltare și la o mai bună mentenanță a codului. Acest lucru permite echipei de dezvoltare să utilizeze același set

de competențe pentru întreaga arhitectură a aplicației, fără a fi nevoie de schimbări de context între limbaje diferite.

JavaScript a fost, astfel, esențial în implementarea logicii aplicației, gestionarea stărilor, comunicarea cu API-ul și realizarea conexiunilor cu baza de date în mod eficient.

## 2.4. React Native

React Native este un framework open-source dezvoltat de Meta (fosta Facebook) care permite construirea de aplicații mobile native utilizând limbajul JavaScript. Prin acest framework, aplicațiile mobile sunt dezvoltate pornind de la un singur set de fișiere sursă, dar rezultatul final este compilat în cod nativ pentru platformele iOS și Android, oferind astfel o experiență de utilizare apropiată de aplicațiile scrise direct în Swift sau Kotlin [10].

Unul dintre avantajele esențiale ale React Native este utilizarea JSX (JavaScript XML), o extensie de sintaxă care permite combinarea logicii aplicației cu structura interfeței într-un mod declarativ. Această abordare facilitează construirea de componente reutilizabile și menținerea unei structuri clare a aplicației. De asemenea, React Native permite utilizarea stărilor (useState) și a efectelor secundare (useEffect), esențiale pentru gestionarea logicii reactive în cadrul aplicației mobile.

În cadrul proiectului, React Native a fost utilizat pentru a dezvolta interfața aplicației mobile, oferind utilizatorilor o experiență fluidă și consistentă pe ambele sisteme de operare. Navigarea între ecrane s-a realizat prin utilizarea bibliotecii react-navigation, iar stilizarea componentelor s-a făcut cu ajutorul Tailwind CSS adaptat pentru React Native, permițând un design modern și responsiv.

Dezvoltatorii au beneficiat de ecosistemul bogat al React Native, incluzând librării pentru afișarea calendarului (react-native-calendars), animarea interfeței (react-native-reanimated) și integrarea cu funcții native ale dispozitivului mobil. Totodată, integrarea perfectă cu platforma Expo a facilitat testarea rapidă pe dispozitive reale, fără a necesita compilare completă sau configurare suplimentară.

Versiunea utilizată la momentul implemetării este: react-native 0.79.2.

## 2.5. Expo

Expo este un framework open-source construit peste React Native, care simplifică procesul de dezvoltare, testare și distribuie a aplicațiilor mobile. Acesta oferă un set de instrumente și servicii menite să reducă complexitatea dezvoltării aplicațiilor native, în special pentru echipe mici sau proiecte cu resurse limitate. Expo permite dezvoltatorilor să creeze aplicații pentru Android și iOS folosind JavaScript și React Native, fără a necesita configurarea manuală a SDK-urilor native sau a mediilor de build complicate. [11]

Unul dintre avantajele principale ale platformei Expo este aplicația mobilă Expo Go, care permite testarea instantanee a codului scris, printr-un simplu scan al unui cod QR. Acest proces reduce drastic timpul necesar testării și oferă o experiență de tip live reload, extrem de utilă în etapele de dezvoltare și depanare. De asemenea, Expo oferă o suită de API-uri preintegrate pentru accesarea funcționalităților dispozitivului (cameră, locație, fișiere, notificări etc.), fără a fi nevoie de scrierea de cod nativ.



Pentru etapa de distribuție, serviciile Expo Application Services (EAS) permit generarea de fișiere APK, AAB sau IPA direct din cloud, fără a necesita instalarea locală a Xcode sau Android Studio. Acest sistem a fost utilizat în cadrul proiectului pentru a genera rapid build-uri de test pentru Android, dar și pentru a pregăti versiunea finală a aplicației pentru publicare.

În cadrul proiectului, Expo a facilitat dezvoltarea rapidă a aplicației mobile, asigurând testare multiplatformă eficientă, integrare facilă cu biblioteci și reducerea semnificativă a timpului de implementare. Alegerea sa a fost determinată de compatibilitatea excelentă cu React Native, documentația detaliată și comunitatea activă de suport.

La momentul implementării s-a utilizat Expo SDK 53 (versiunea 53.0.8).

## 2.6. Debian Linux

Debian este o distribuție GNU/Linux menită să deservească majoritatea utilizatorilor. Privit drept un întreg sistem de operare, Debian include soluții software și sisteme de instalare și management bazate complet pe nucleul open-source Linux. [12]

Gratuitatea, performanța ridicată construită pe simplitate și stabilitatea clădită de o comunitate extinsă implicată în dezvoltarea acestui sistem de operare sunt principalele lui avantaje. În implementarea proiectului, Debian a fost instalat pe un server accesibil remote, pe care rulează componentele esențiale ale platformei, precum: serviciul Node.js/Express care gestionează logica backend, serviciul MySQL pentru gestiunea bazei de date, precum și diverse procese auxiliare (logare, cronuri, servicii REST externe). Conectarea la server s-a realizat securizat, prin protocol SSH, iar codul a fost editat și testat direct pe acest server, folosind un mediu de dezvoltare compatibil (Visual Studio Code cu extensie de remote SSH).

Detaliile tehnice hardware ale serverului (virtual) utilizat în implementare:

- Procesor: 1 vCPU
- Memorie RAM: 2GB
- Stocare: 25GB SSD

Specificațiile sistemului de operare:

- Nucleu: Linux
- Distribuție: Debian GNU/Linux 12

## 2.7. API

Un API (Application Programming Interface – Interfață de Programare a Aplicațiilor) reprezintă un set de reguli și protocoale care permit interacțiunea între diferite aplicații software. Prin intermediul unui API, o aplicație poate accesa funcționalități sau date ale altei aplicații, fără a cunoaște detalii despre implementarea internă a acesteia. Această abordare facilitează modularitatea și reutilizarea componentelor software, fiind esențială în dezvoltarea aplicațiilor moderne. [13]

În cadrul proiectului prezent, API-ul a fost utilizat pentru a face legătura între interfața aplicației mobile și serverul backend. Toate operațiunile importante, precum înregistrarea și autentificarea utilizatorilor, trimiterea și gestionarea programărilor, obținerea informațiilor despre servicii sau actualizarea datelor salonului, au fost realizate prin apeluri către endpoint-uri



definite în cadrul unei arhitecturi REST. Acest tip de arhitectură, bazată pe metodele standard HTTP (GET, POST, PUT, DELETE), oferă simplitate în implementare și scalabilitate ridicată.

Pe lângă API-urile REST, în ecosistemul aplicațiilor moderne sunt întâlnite și alte tipuri de API-uri, precum cele bazate pe SOAP (Simple Object Access Protocol), care oferă un cadru rigid, dar sigur, pentru aplicații enterprise, sau GraphQL, care permite interogări personalizate pentru eficientizarea fluxului de date. În cazul acestui proiect s-a optat exclusiv pentru REST, datorită compatibilității excelente cu stack-ul JavaScript/Node.js utilizat.

Rolul API-ului nu este doar de a conecta două componente tehnice, ci de a crea un limbaj comun între client și server. Prin această separare clară a responsabilităților, aplicația devine mai ușor de întreținut, testat și extins. De asemenea, documentarea precisă a endpoint-urilor permite integrarea viitoare cu alte servicii externe, dacă va fi necesar.

Conceptul de API este larg utilizat în dezvoltarea aplicațiilor distribuite, cloud sau mobile, fiind considerat un pilon central al programării moderne [14].

## 2.8. Node.js

Node.js este un mediu de execuție open-source pentru JavaScript, construit pe motorul V8 al browserului Google Chrome, care permite rularea codului JavaScript în afara unui browser web. Această caracteristică a extins aplicabilitatea limbajului JavaScript dincolo de zona front-end, permițând dezvoltarea completă a aplicațiilor în același limbaj, atât pe client, cât și pe server, unificând astfel procesul de dezvoltare. [15]

Una dintre caracteristicile fundamentale ale Node.js este modelul său de execuție asincron și orientat pe evenimente. Acest model non-blocant permite gestionarea eficientă a unui număr mare de conexiuni simultane, ceea ce îl face extrem de potrivit pentru aplicații web care presupun trafic intens sau operațiuni frecvente asupra unei baze de date. Utilizarea promisiunilor și a sintaxei `async/await` face codul mai clar și mai ușor de întreținut, fără a pierde din performanță.

În cadrul proiectului, Node.js a fost folosit ca tehnologie principală pentru dezvoltarea backend-ului. S-au implementat funcționalități precum autentificarea utilizatorilor, trimiterea notificărilor, gestionarea programărilor și interacțiunea cu baza de date MySQL. De asemenea, serverul a expus un set de endpoint-uri API REST, consumate de aplicația mobilă dezvoltată în React Native.

Un alt avantaj important l-a constituit ecosistemul extrem de dezvoltat din jurul Node.js, în special platforma npm (Node Package Manager), prin care au fost integrate rapid pachete necesare cum ar fi `express`, `cors`, `multer` sau `mysql2`. Astfel, a fost posibilă dezvoltarea rapidă și modulară a unei aplicații scalabile și ușor de extins.

Versiunea Node.js la momentul implementării: v18.19.0

## 2.9. Google Places

Google Places API este un serviciu oferit de Google care permite aplicațiilor să acceseze informații detaliate despre locații, cum ar fi restaurante, hoteluri, magazine și alte puncte de interes. Prin intermediul acestui API, dezvoltatorii pot integra funcționalități precum căutarea de

locuri, completarea automată a adreselor și obținerea de detalii despre locații specifice, inclusiv recenzii, fotografii și informații de contact. [16]

În cadrul proiectului, Google Places API a fost utilizat pentru a identifica salonul oferind utilizatorilor posibilitatea de a vizualiza locația pe hartă și a acorda recenzii. Totodată, vizualizarea ultimelor cinci recenzii ale unei locații sunt extrase prin acest API și afișate în aplicație.

Integrarea Google Places API în aplicație a contribuit semnificativ la îmbunătățirea funcționalității și a experienței utilizatorilor, oferind acces rapid și precis la informații despre locații.

## **2.10. Resend**

Resend este o platformă modernă destinată dezvoltatorilor, care oferă un API REST simplu și eficient pentru trimiterea de emailuri tranzacționale și de marketing. Prin intermediul Resend, aplicațiile pot trimite emailuri cu o rată ridicată de livrare, evitând ca mesajele să ajungă în folderele de spam. Platforma oferă SDK-uri pentru limbaje de programare populare, inclusiv Node.js, Python, PHP și altele, facilitând integrarea rapidă în diverse stack-uri tehnologice. [17]

În cadrul proiectului, Resend a fost utilizat pentru a gestiona trimiterea emailurilor automate, cum ar fi confirmările de rezervare, notificările și resetările de parolă. API-ul Resend permite trimiterea de emailuri individuale sau în loturi, programarea trimitărilor și atașarea de fișiere, oferind astfel flexibilitate în funcție de nevoile aplicației.

Resend oferă, de asemenea, funcționalități avansate precum gestionarea domeniilor de trimitere, monitorizarea evenimentelor de livrare (deschideri, click-uri, bounce-uri) și integrarea cu webhooks pentru notificări în timp real. Aceste caracteristici permit dezvoltatorilor să aibă un control detaliat asupra procesului de trimitere a emailurilor și să optimizeze comunicarea cu utilizatorii.

## **2.11. SQL**

O bază de date reprezintă o colecție de informații utilizată într-o organizație, colecție care este automatizată, partajată, definită riguros (formalizată) și controlată la nivel central. Sistemul de gestiune a unei baze de date constă într-un ansamblu de programe ce permit utilizatorilor să interacționeze cu o bază de date în vederea creării, actualizării și interogării acesteia. [18]

SQL (Structured Query Language) este limbajul standard utilizat pentru interacțiunea cu bazele de date relaționale. Acesta permite definirea, manipularea și interogarea datelor stocate în tabele, oferind un set de comenzi clare și concise pentru operațiuni precum inserarea, actualizarea, ștergerea și selecția datelor. Datorită sintaxei sale intuitive și a puterii expresive, SQL a devenit un instrument indispensabil în dezvoltarea aplicațiilor care necesită gestionarea eficientă a datelor. În cadrul proiectului, SQL a fost utilizat pentru crearea și administrarea bazei de date, definirea relațiilor între tabele și implementarea logicii de acces la date.

## **2.12. MySQL**

MySQL este un sistem de gestiune a bazelor de date relaționale (RDBMS) open-source, recunoscut pentru performanța, fiabilitatea și ușurința în utilizare. Dezvoltat inițial de compania

suedeză MySQL AB și ulterior achiziționat de Oracle Corporation, MySQL a devenit una dintre cele mai populare soluții pentru stocarea și gestionarea datelor în aplicații web și software. [19]

În cadrul proiectului, MySQL a fost ales ca sistem de gestionare a bazei de date datorită compatibilității excelente cu limbajul de programare JavaScript și cu mediul de execuție Node.js. Utilizarea MySQL a permis definirea clară a structurii bazei de date, stabilirea relațiilor între tabele și implementarea eficientă a operațiunilor de manipulare a datelor.

Versiunea utilizată pentru implementare este MySQL 9.1.

### 2.13. HTML

HTML (HyperText Markup Language) este limbajul standard utilizat pentru crearea și structurarea conținutului pe web. Acesta permite definirea elementelor unei pagini web, cum ar fi titluri, paragrafe, liste, linkuri, imagini și multe altele, prin utilizarea unor etichete (tag-uri) specifice. HTML este esențial în dezvoltarea aplicațiilor web, oferind baza pe care se construiesc stilurile CSS și funcționalitățile JavaScript. [20]

Structura unui document HTML se bazează pe tag-uri și atribute. Tag-urile nu sunt altceva decât niste marcaje sau etichete pe care limbajul HTML le folosește alături de texte pentru a ajuta browser-ul de internet să afișeze corect conținutul paginii web. Atributele pot fi definite ca niste proprietăți ale tag-urilor. Atributele se pun numai în tag-ul de început. Dacă un tag nu are nici un atribut, atunci browser-ul va lua în considerare valorile implicite ale tag-ului respectiv. [21]

În cadrul proiectului, HTML a fost utilizat pentru a defini structura interfeței aplicației pentru platforma web, asigurând o organizare clară și semantică a conținutului. Prin utilizarea corectă a etichetelor HTML, s-a facilitat accesibilitatea și compatibilitatea cu diverse dispozitive și browsere.

### 2.14. CSS

CSS, prescurtarea de la Cascading Style Sheets, sunt etichete folosite pentru formatarea paginilor web (de exemplu formatare text, background sau aranjare în pagină etc.). Beneficiile sintaxei CSS sunt următoarele: formatarea este introdusă într-un singur loc pentru tot documentul; editarea rapidă a etichetelor; datorită introducerii într-un singur loc a etichetelor se obține o micșorare a codului paginii, implicit încărcarea mai rapidă a acesteia. [22]

În cadrul proiectului, CSS a fost utilizat pentru a stiliza componentele aplicației, atât în versiunea mobilă cât și în cea web, asigurând o experiență vizuală plăcută și uniformă pe diferite dispozitive și rezoluții. Prin utilizarea de clase și identificatori, s-au aplicat stiluri specifice diverselor elemente, cum ar fi butoane, formulare și meniuri de navigare. De asemenea, s-au implementat tehnici de responsive design pentru a adapta interfața la dimensiunile variate ale ecranelor utilizatorilor.

### 3. STUDIU DE CAZ: DEZVOLTAREA UNEI PLATFORME INFORMATICE DE TIP MOBILE PENTRU SALOANELE DE ÎNFRUMUSEȚARE

Societățile comerciale din domeniul prestării de servicii de înfrumusețare se confruntă cu o plinitate dezorganizată de soluții digitale. Fiecare dintre aceste soluții are în componență funcționalități avantajoase și bine implementate, dar și insuficiențe la anumite capitole. În cele ce urmează, se propune implementarea unei soluții complete care să înglobeze toate caracteristicile necesare unei experiențe optime în interacțiunea dintre saloanele de înfrumusețare și clienții acestora.

Procesul de dezvoltare a fost realizat în sistemul de operare Microsoft Windows (2.1) prin intermediul mediului de dezvoltare Visual Studio Code (2.2), ceea ce a permis totodată și conectarea la distanță la serverul de backend centralizând întreaga producție.

#### 3.1. Arhitectura platformei

Platforma informatică abordează o arhitectură în trei straturi, descrisă general în subcapitolul 1.1, dar restructurată și adaptată nivelului de complexitate propus.

##### 3.1.1. Prezentare generală

Soluția are la bază o arhitectură client-server scalabilă ce prezintă o mentenanță facilă separând clar responsabilitățile între interfața utilizatorului (frontend) și logica de procesare a datelor (backend).

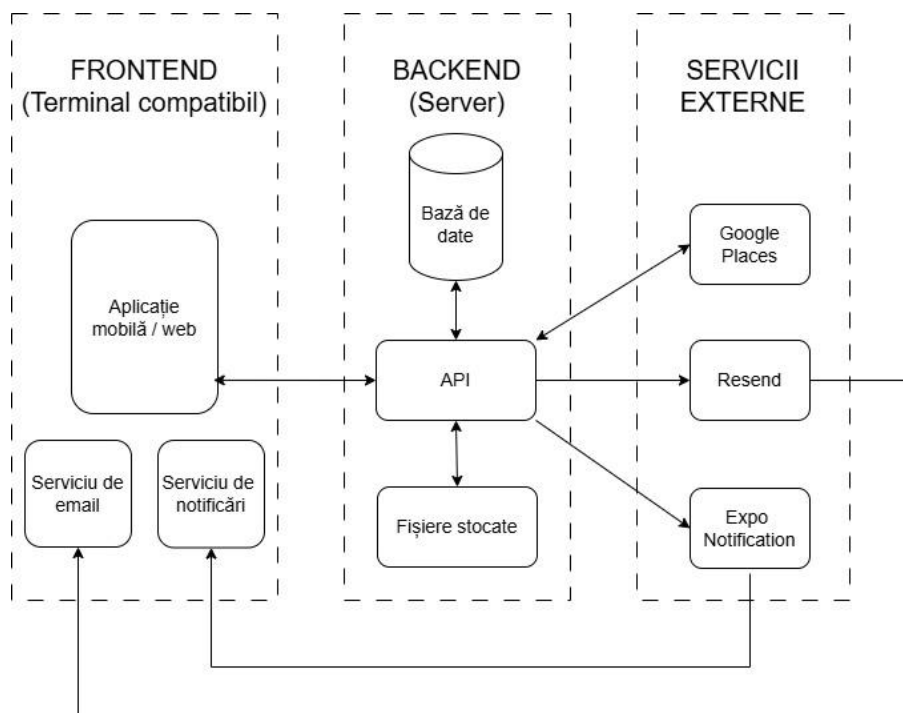


Fig. 3.1. Model schematic al arhitecturii platformei

### 3.1.2. Componentele platformei

Frontend-ul constă într-un terminal compatibil fie cu aplicația mobilă utilizabilă pe sistemele de operare Android (2.15) sau iOS (2.16), fie cu versiunea web a acesteia accesibilă dintr-un browser instalat cu acces la internet. Pentru o astfel de implementare hibridă se utilizează ca limbaj de programare predominant JavaScript (2.3) și framework-urile Expo (2.5) și React-Native (2.4).

Backend-ul constă în serverul cu sistem de operare Debian GNU/Linux (2.6). Serverul găzduiește API-ul (2.7) dezvoltat pe framework-ul NodeJS (2.8) și informațiile stocate sub formă de fișiere sau structurate în baza de date MySQL (2.12).

Totodată, prin intermediul API-ului se accesează servicii externe ale căror uzabilitate este descrisă în subcapitolul următor.

### 3.1.3. Integrarea de servicii externe

Platforma informatică propusă integrează funcționalități prin servicii externe consacrate drept vârf de gamă în domeniile lor. Prin intermediul API-ului se accesează serviciile externe Google Places (2.9), Resend (2.10) sau Expo Notification ce intervin în procesele de recenzionare și notificare.

### 3.1.4. Fluxul de date

Fluxul de date în cadrul acestui sistem poate fi descris pe baza diagramei:

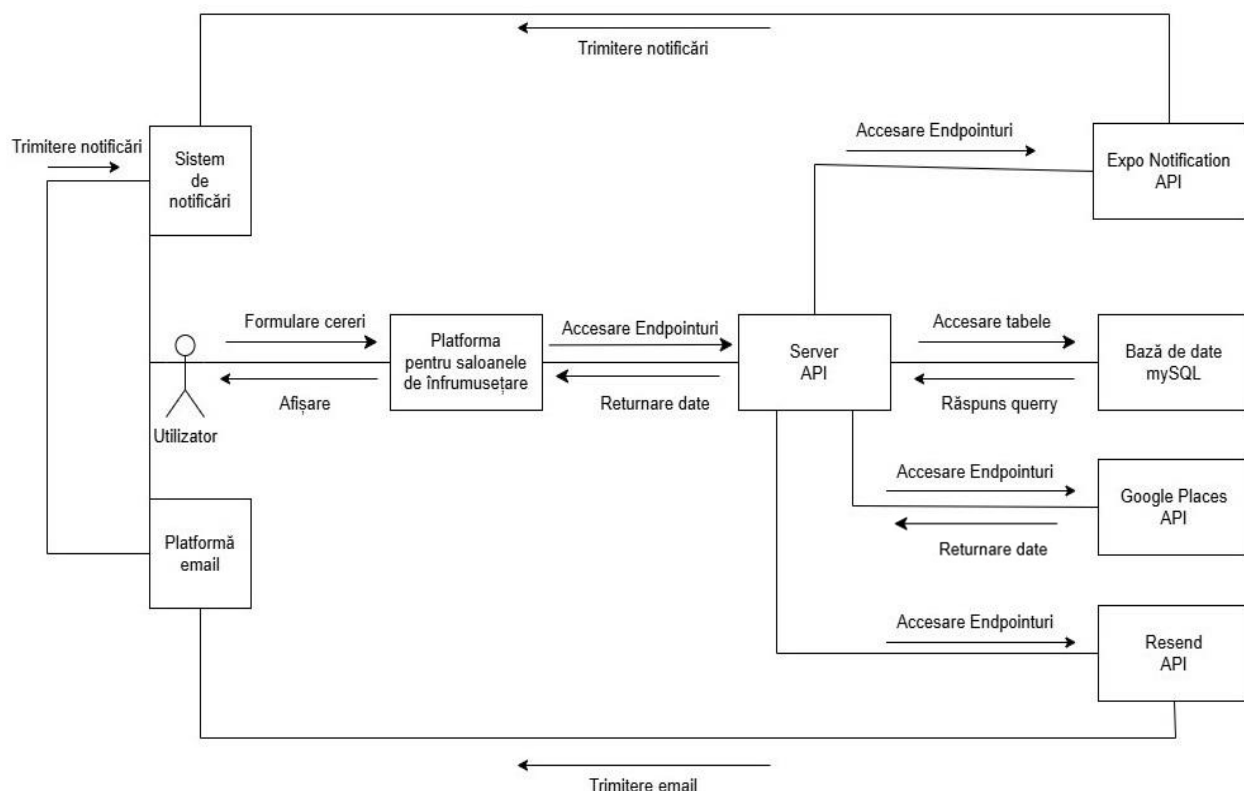


Fig. 3.2. Diagrama de colaborare a întregului sistem

Analiza completă a acestui flux are ca premisă existența unui utilizator în posesia unui terminal compatibil cu platforma informatică și cu un sistem de notificări pre-implementat în

sistemul de operare al acestui terminal. Totodată dispune de acces la internet și la un serviciu de email.

În prim plan, utilizatorul formulează cereri prin intermediul interfeței aplicației (ex. efectuarea unei programări, vizualizarea programărilor, ș.a.). Interacțiunea cu elementele din interfață este ulterior tradusă în instrucțiuni realizabile fie local, pe dispozitiv, fie prin accesarea endpointurilor din API-ul serverului.

Endpointurile sunt accesate prin cereri HTTP de tip GET, POST, PUT sau DELETE, fiecare corespunzând unei anumite operațiuni. Aceste cereri poartă în corpul lor toate informațiile necesare pentru identificarea utilizatorului și executarea logicii din backend.

Odată primită cererea, serverul o procesează în funcție de ruta apelată. În funcție de necesități, fluxul se ramifică în acest punct. Cele mai uzuale interacțiuni sunt realizate prin query-uri către baza de date. Baza de date execută query-ul și răspunde cu un array de informații.

În sens invers, array-ul de date returnat de instrucțiunea SQL este transpus în format JSON de către server și este trimis către aplicație. Aplicația prelucrează informația primită și o afișază într-o formă inteligibilă utilizatorului.

La nivel de server sunt accesate și servicii externe ce completează platforma. Prin accesarea endpointurilor API ale Google Places sunt actualizate zilnic informații despre saloanele din aplicație: scorul, recenziile și locația. În urma cererii, serverul celor de la Google returnează informații în format JSON pe baza cărora baza de date a platformei este actualizată. Procesul de notificare a utilizatorului este intermediat de soluțiile API oferite de Resend, în cazul notificării prin email și Expo Notification în cazul notificării de tip push-notification.

### **3.2. Rolurile utilizatorilor**

În scopul de a îngloba funcționalitatea de booking cu cea de management a saloanelor de înfrumusețare într-o singură aplicație, interfața se adaptează conform rolului pe care utilizatorul îl prezintă. În cadrul platformei sunt definite două roluri: client și partener.

Rolul de client este atribuit automat fiecărui utilizator al platformei la crearea unui cont nou. Astfel, fiecare utilizator al aplicației are acces la modulele dedicate clienților și la funcționalitățile acestora.

Rolul de partener se obține prin înrolarea unui salon în platforma informatică. Administratorii saloanelor de înfrumusețare pot deveni parteneri prin completarea unui formular cu informațiile de natură legală aferente societății comerciale pe care o reprezintă. Acest demers oferă acces la un panou de control dedicat managementului salonului respectiv.

### **3.3. Navigarea în aplicație**

Navigatorul aplicației este realizat cu ajutorul pachetului expo-router oferit de Expo și abordează o structură bazată pe tab-uri ce sunt prezentate în partea inferioară a interfeței. Aplicația verifică rolul utilizatorului, iar navigatorul se adaptează automat acestuia.



Fig. 3.3. Structura generică a interfeței cu navigare prin tab-uri

### 3.3.1. Navigatorul principal

Codul sursă al acestui layout de navigare se regăsește în Anexa 1 – app/\_layout.js.



Fig. 3.4. Navigatorul principal pentru rolul de client

Utilizatorii cu rol simplu de client au acces prin tab-uri la modulele: Acasă, Programări, Loialitate, Contul Meu. Stilizarea casetelor este realizată prin iconițe din dependența @expo/vector-icons/Ionicons. Aici sunt definite totodată și titlurile fiecărui modul atât în bara de taburi cât și în partea superioară a afișării ecranelor respective.

Verificarea rolului se realizează prin accesarea endpoint-ului <https://api.glowapp.ro/user> pe baza token-ului salvat în memoria locală AsyncStorage a dispozitivului. Endpointul returnează o listă de informații stocate în baza de date despre utilizator, iar în situația în care utilizatorului îi este asociat un salon, în bara navigatorului se va regăsi caseta suplimentară „Salonul meu”.



Fig. 3.5. Navigatorul principal pentru rolul de partener

### 3.3.2. Navigatorul dedicat administrării salonului de înfrumusețare

Codul sursă al acestui layout de navigare se regăsește în Anexa 1 – app/salonmanager/\_layout.js.



Fig. 3.6. Navigatorul dedicat administrării unui salon



În cazul unui utilizator cu rolul de partener, accesul este extins la navigatorul suplimentar și la modulele: Salonul meu, Servicii, Programări. Revenirea la sistemul de navigație principal se face tot prin bara de tab-uri, prin apăsarea casetei Acasa, stilizată cu o iconiță ce sugerează întoarcerea la navigatorul principal.

### 3.4. Modulele aplicației

Frontend-ul aplicației abordează un stil bazat pe simplitate în ceea ce privește interfața, dar este totodată complex prin multitudinea de funcționalități pe care le înglobează. Pentru claritate, acest subcapitol este ghidat de diagrama de flux al ecranelor aplicației.

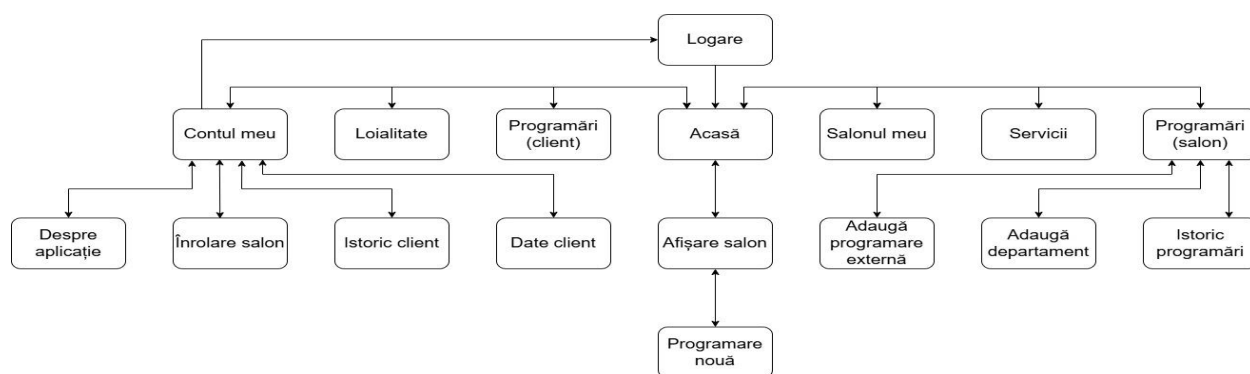


Fig. 3.7. Diagrama de flux al ecranelor

#### 3.4.1. Modulul de logare

Codul sursă pentru acest modul se regăsește în Anexa 1 – app/index.js.

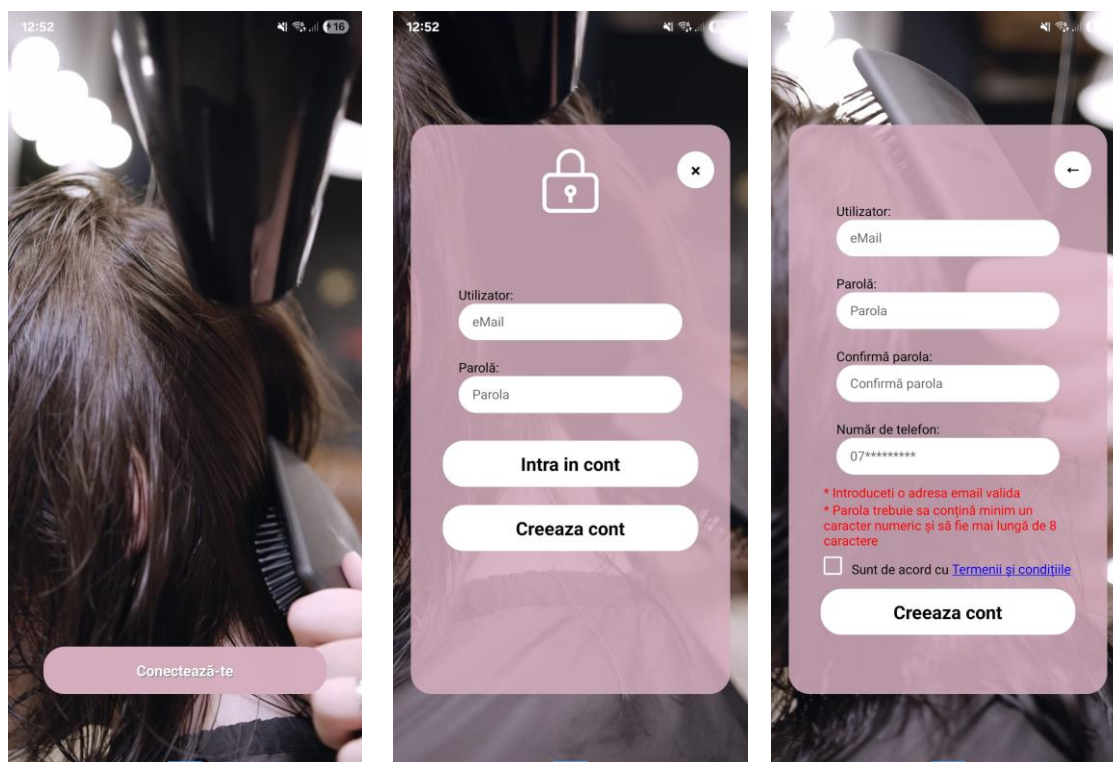


Fig. 3.8. Modulul de logare



Ecranul incipient al aplicației este cel ce identifică utilizatorul. La accesarea aplicației se verifică automat dacă utilizatorul este deja autentificat prin căutarea unui token salvat în AsyncStorage. Dacă acest token este găsit, aplicația face o solicitare către server pentru a îl valida. În funcție de răspunsul primit, utilizatorul poate fi redirecționat către modulul Acasă.

În cazul nevalidării unui token, utilizatorului îi este prezentat un ecran cu fundal dinamic ce constă într-un videoclip al procesului de înfrumusețare. În partea inferioară este prezent un buton cu textul „Conectează-te” ce declanșează o animație, aducând în prim plan formularul de conectare.

Formularul de conectare cuprinde două câmpuri de text pentru email, respectiv parolă și două butoane. Primul buton, „Intră în cont” este cel de conectare, a cărei apăsare transmite datele din formular către endpointul <https://api.glowapp.ro/login>. În cazul validării datelor introduse, serverul transmite în răspuns un token ce se salvează în AsyncStorage (în memoria internă a dispozitivului) pentru a menține utilizatorul conectat la următoarea accesare a aplicației.

Cel de-al doilea buton al formularului de conectare, „Creează cont”, este destinat înregistrării. Astfel, apăsarea lui conduce la adaptarea formularului pentru a include toate datele necesare înscrierii unui utilizator nou în platformă. Formularul de înregistrare este alcătuit din patru câmpuri de text: email, parolă, confirmare parolă și număr de telefon și butonul de înregistrare propriu-zis ce accesează endpointul <https://api.glowapp.ro/register> prin metoda POST ce creează un nou cont de utilizator. Datele din formulare sunt validate local, pentru a nu trimite solicitări inutile către server.

### 3.4.2. Modulul Acasă

Codul sursă pentru acest modul se regăsește în Anexa 1 – [app/home.js](#).

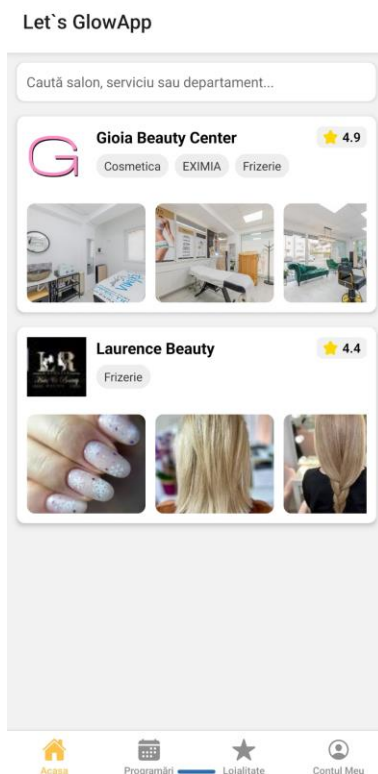


Fig. 3.9. Modulul Acasă

Modulul Acasă reprezintă ecranul principal al aplicației și oferă vizualizarea succintă asupra saloanelor de înfrumusețare disponibile în platformă. Lista de parteneri și informațiile aferente (scor, departamente, servicii) sunt extrase din baza de date a serverului prin endpointul <https://api.glowapp.ro/saloane>. Caseta de prezentarea a fiecărui salon este în fapt o componentă Touchable Opacity a cărei apăsare navighează către ecranul cu informații detaliate despre partener.

În ceea ce privește materialele media, logo-ul salonului este implicit disponibil la adresa <https://api.glowapp.ro/{partenerID}/logo>. Portofoliul fotografic este însă obținut ulterior prin apel la endpointul <https://api.glowapp.ro/{partenerID}/list/images>.

Modulul integrează o componentă de căutare (TextInput) ce filtrează în timp real lista de afișări. Aici se poate insera denumirea salonului dorit, un departament sau un serviciu căutat.

Stilizarea componentelor utilizează umbre și margini rotunjite, elemente grafice moderne, toate pentru a amplifica experiența utilizatorului și confortul vizual.

### 3.4.3. Modulul de afișare a salonului

Codul sursă pentru acest modul se regăsește în Anexa 1 – `app/salon/[id].js`. Navigatorul este adaptat dinamic conform Anexa 1 – `app/salon/_layout.js`

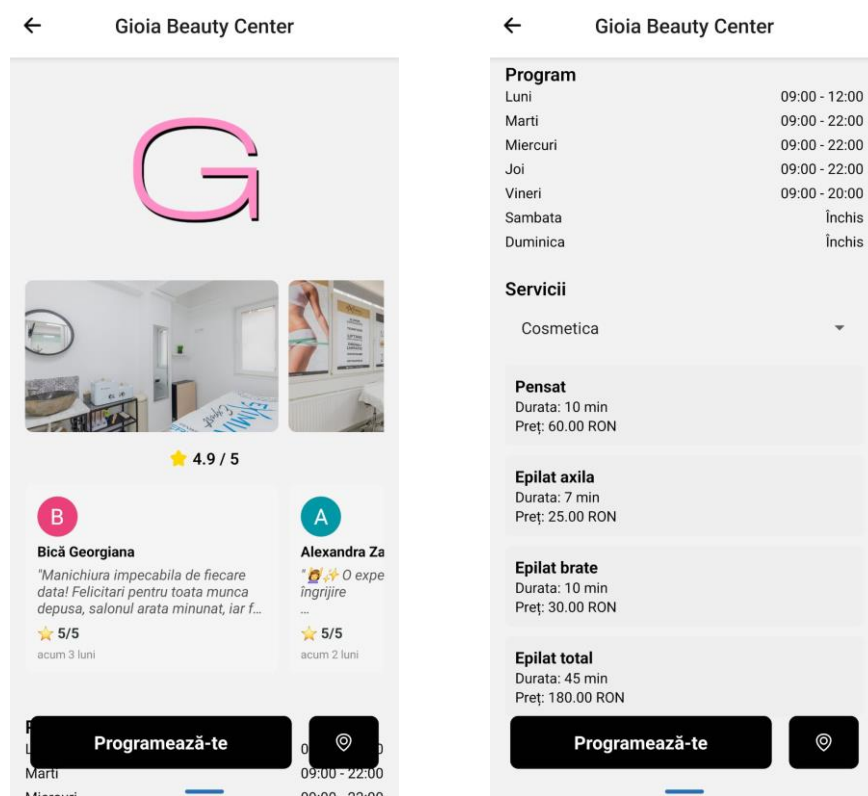


Fig. 3.10. Modulul de afișare al salonului

În cadrul modulului de afișare detaliată a salonului sunt prezentate informațiile extrase prin interogarea endpointului <https://api.glowapp.ro/salondata> cu codul unic al partenerului. Pagina este dinamică, id-ul salonului accesat fiind un parametru transmis din modulul anterior.

Logo-ul este afișat asemenea modulului anterior (3.4.2), prin link direct, iar portofoliul fotografic prin accesarea endpointului <https://api.glowapp.ro/{partenerID}/list/images>. Recenziile sunt recepționate de la server și stilizate sub formă de casete. Următoarea secțiune afișează programul orar pe zile. Serviciile conțin denumirea, durata și prețul și sunt structurate pe departamente selectabile dintr-un meniu de tip Picker din biblioteca React-Native.

În partea inferioară, cu poziționare absolută, se regăsește butonul de localizare ce redirecționează către locația salonului din Google Maps alături de butonul „Programează-te” ce navighează către modulul dedicat efectuării unei programări.

Din punct de vedere al layout-ului, titlul paginii constă în numele salonului, afișat dinamic în urma extracției acestuia din server. Pentru a oferi un spațiu extins de vizualizare, bara inferioară de navigare este ascunsă și substituită de un buton de revenire la modulul anterior.

#### 3.4.4. Modulul de programare

Codul sursă pentru acest modul se regăsește în Anexa 1 – `app/programare/[id].js`. Navigatorul este adaptat dinamic conform Anexa 1 – `app/programare/_layout.js`

Fig. 3.11. Modulul de programare

Acest modul gestionează procesul de realizare a unei noi programări de către client. Identificatorul salonului este transmis ca parametru din modulul anterior. Totodată, interfața este adaptată dinamic, iar bara de navigare rămâne invizibilă.

Serviciile se extrag din server prin apelarea endpointului <https://api.glowapp.ro/salondata> și sunt organizate pe departamente. În urma selectării unui departament din meniul de tip Picker,

sunt afișate serviciile aferente acestuia sub forma unor casete selectabile. Clientul poate selecta unul sau mai multe dintre acestea, iar pe baza selecției este calculată și afișată durata totală a eventualei programări.

După alegerea serviciilor dorite, utilizatorului i se afișează un calendar (react-native-calendars) configurat în limba română. Aplicația validează informațiile și interoghează endpointul <https://api.glowapp.ro/disponibilitate> ce returnează o listă de intervale orare posibile în funcție de ziua, departamentul selectat și durata serviciilor selectate. Aceste intervale sunt afișate într-un element Picker din care utilizatorul poate alege.

Completarea elementelor descrise anterior face vizibil butonul „Programează-te” care transmite toate datele la endpointul <https://api.glowapp.ro/programare> prin metoda POST. Înainte de apelare, toate informațiile sunt validate local, iar în cazul erorilor se afișează alerte descriptive. Dacă serverul răspunde corect la apel, utilizatorului îi este afișat un mesaj de succes și este redirecționat către ecranul principal al aplicației.

### 3.4.5. Modulul Programări (client)

Codul sursă pentru acest modul se regăsește în Anexa 1 – `app/programariclient.js`.

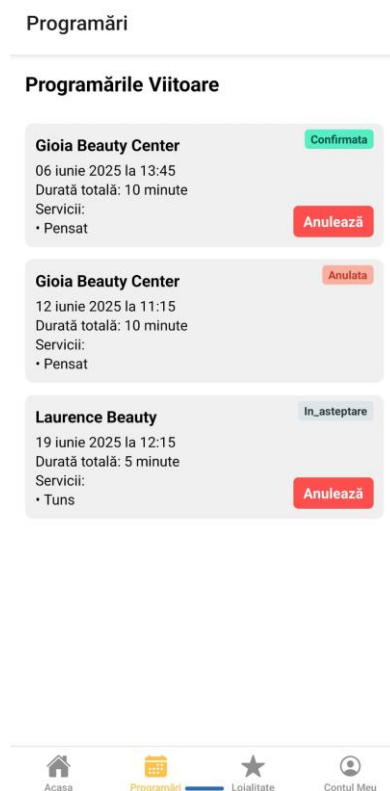


Fig. 3.12. Modulul Programări (client)

Modulul Programări oferă clientului acces la o listă cu programările sale viitoare alături de informațiile aferente. Lista este recepționată de la server prin apelarea metodei GET pe endpointul <https://api.glowapp.ro/programariclient> alături de parametrul `userToken` extras din `AsyncStorage`. Fiecare programare include numele salonului, data și ora programării, durata estimată și serviciile aferente. Apăsarea unei casete de programare redirecționează utilizatorul

către modulul de afișare a salonului (descriș în subcapitolul 3.4.3) unde se pot obține informații suplimentare.

În colțul superior din dreapta casetei cu programarea este afișat stilizat statusul programării (verde, gri sau roșu pentru confirmată, în așteptare respectiv anulată). Toate informațiile din această pagină sunt actualizate constant prin dependența `useFocusEffect` a bibliotecii `react-navigation`, astfel încât utilizatorul să poată vizualiza modificări asupra programărilor în timp real.

Anularea unei programări cu statusul de confirmată sau în așteptare se poate realiza de către client prin butonul roșu „Anulează” ce transmite o cerere `DELETE` asupra endpointului `https://api.glowapp.ro/programare` alături de codul unic de identificare al programării.

### 3.4.6. Modulul Loialitate

Codul sursă pentru acest modul se regăsește în Anexa 1 – `app/loialitate.js`.

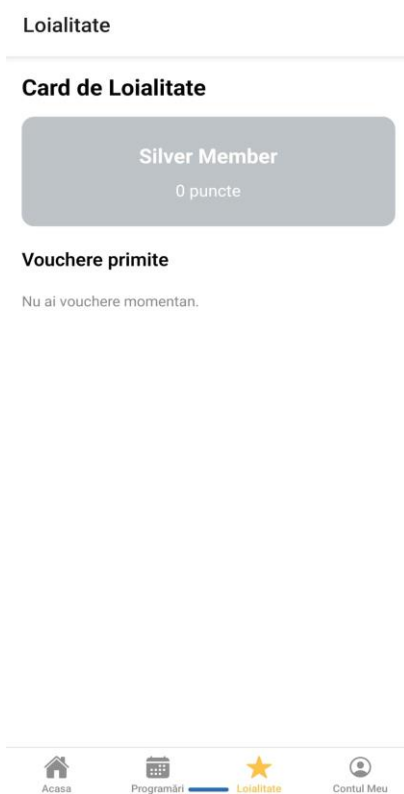


Fig. 3.13. Modulul Loialitate

Montarea acestei componente extrage token-ul utilizatorului din memoria internă a dispozitivului și apelează endpointul `https://api.glowapp.ro/loialitate` prin metoda `GET`. Răspunsul serverului va conține două elemente principale: numărul de puncte acumulate de utilizator și o listă de vouchere acordate acestuia.

Ierarhia în ceea ce privește cardul de loialitate constă în trei praguri. Nivelul „Silver” este acordat utilizatorilor cu sub 300 de puncte acumulate, urmat de nivelul „Gold” și la final de nivelul „Diamond” pentru peste 600 de puncte acumulate. Cardul virtual este stilizat în funcție de nivel prin culorile argintiu, auriu și respectiv albastru deschis.

Sub cardul de fidelitate este afișată lista voucherelor disponibile. Fiecare voucher este reprezentat vizual printr-o casetă și un simbol sugestiv al unui cadou. Voucherele sunt acordate manual de către saloanele de înfrumusețare, iar nivelul de loialitate este doar un atribut menit să influențeze saloanele de înfrumusețare în loializarea unui anumit client.

### 3.4.7. Modulul Contul meu

Codul sursă pentru acest modul se regăsește în Anexa 1 – app/accmenu/index.js. Acest modul constituie punct de legătură cu alte ecrane și logica de navigare este adaptată conform codului sursă din Anexa 1 – app/accmenu/\_layout.js.

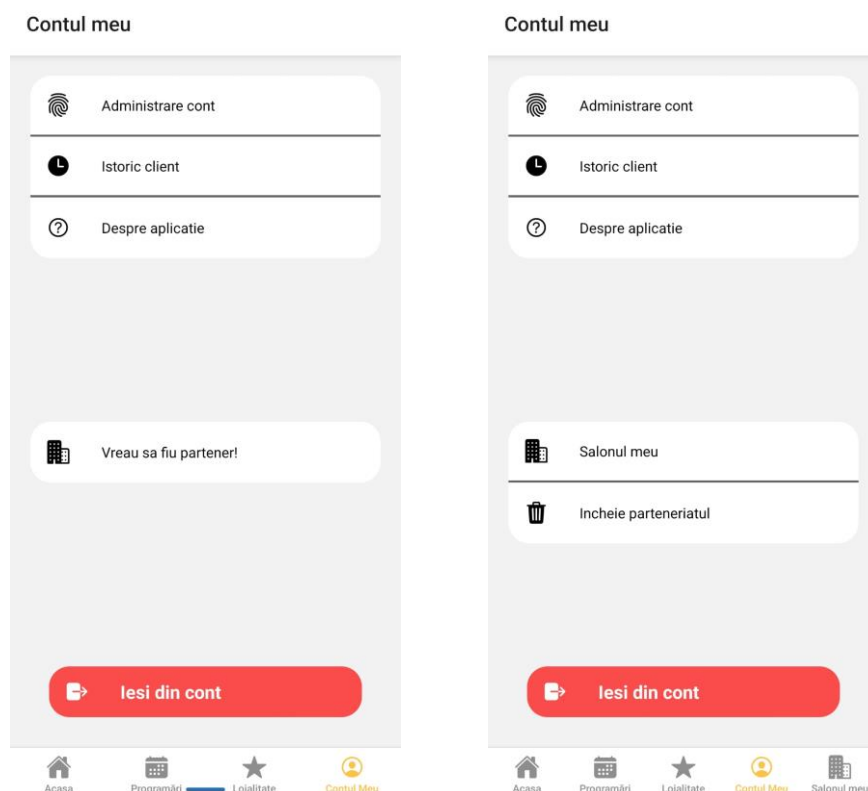


Fig. 3.14. Modulul Contul meu

Modulul contul meu servește drept centru de control și punct de acces către paginile dedicate gestionării contului. La fiecare accesare a acestui ecran, componenta useFocusEffect accesează endpointul GET <https://api.glowapp.ro/user> folosind token-ul stocat în AsyncStorage. Pe baza datelor obținute de la server, se verifică dacă utilizatorul include rolul de partener și adaptează pagina conform Fig. 3.14.

Pentru rolul de utilizator, butonul „Administrare cont” navighează către modulul de actualizare a datelor personale (3.4.8), butonul „Istoric client” redirecționează la ecranul corespunzător (3.4.9) și butonul „Despre aplicație” conduce către o pagină informativă referitoare la platforma informatică. Pentru potențialii parteneri, aceștia își pot obține rolul prin interacțiunea cu modulul destinat înrolării unui salon (3.4.10), accesibil la apăsarea butonului „Vreau să fiu partener”.

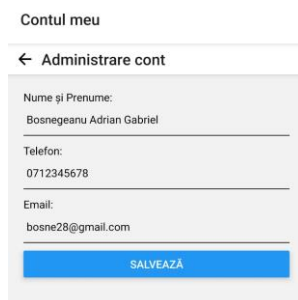
Rolul de partener modifică structura acestei pagini, butonul „Vreau să fiu partener” fiind înlocuit cu butoanele „Salonul meu” ce accesează modulul Salonul meu (3.4.11) și butonul

destinat ștergerii salonului reprezentat din baza de date, „Încheie parteneriatul”. Încheierea parteneriatului se face prin apelul DELETE la endpointul <https://api.glowapp.ro/salon>.

În partea inferioară se regăsește butonul „Ieși din cont” care șterge token-ul utilizatorului din AsyncStorage și înlocuiește ruta curentă cu cea a modului Logare (3.4.1) ceea ce conduce la părăsirea contului.

### 3.4.8. Modulul de actualizare a datelor clientului

Codul sursă al acestui modul se regăsește în Anexa 1 – `app/accmenu/accddata.js`.



Contul meu

← Administrare cont

Nume și Prenume:  
Bosnegeanu Adrian Gabriel

Telefon:  
0712345678

Email:  
bosne28@gmail.com

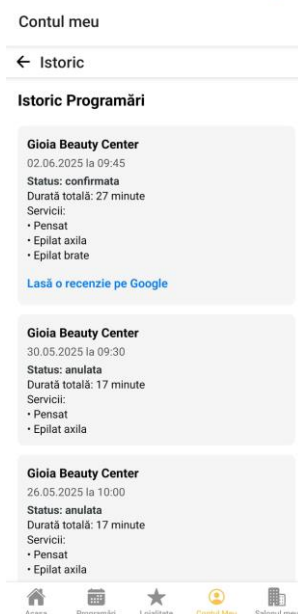
SALVEAZĂ

Fig. 3.15. Modulul de actualizare a datelor clientului

Scopul acestui modul este revizuirea datelor de identificare ale clientului. Componenta preia datele de la server din endpointul <https://api.glowapp.ro/user> prin metoda GET și le afișează într-un formular modificabil. În urma modificării, utilizatorul poate alege să actualizeze datele pe server prin butonul „Salvează” ce execută un apel PUT la endpointul <https://api.glowapp.ro/userUpdate> însoțit de informațiile din formular și de token-ul utilizatorului.

### 3.4.9. Modulul de vizualizare a istoricului clientului

Codul sursă al acestui modul este prezentat în Anexa 1 – `app/accmenu/istoricclient.js`.



Contul meu

← Istoric

Istoric Programări

**Gioia Beauty Center**  
02.06.2025 la 09:45  
Status: confirmata  
Durată totală: 27 minute  
Servicii:  
• Pensat  
• Epilat axila  
• Epilat brate  
[Lasă o recenzie pe Google](#)

**Gioia Beauty Center**  
30.05.2025 la 09:30  
Status: anulata  
Durată totală: 17 minute  
Servicii:  
• Pensat  
• Epilat axila

**Gioia Beauty Center**  
26.05.2025 la 10:00  
Status: anulata  
Durată totală: 17 minute  
Servicii:  
• Pensat  
• Epilat axila

Acasa Programări Localitate Contul Meu Salonul meu

Fig. 3.16. Modulul de vizualizare a istoricului clientului

În cadrul acestui modul se oferă utilizatorului acces la o listă completă a programărilor realizate prin intermediul platformei. Lista de programări este obținută prin solicitarea endpointului <https://api.glowapp.ro/istoricclient>, metoda GET.

În cazul programărilor confirmate ce s-au realizat (durata totală a serviciilor a fost depășită), în dreptul programării aferente este vizibil un buton cu textul „Lasă o recenzie pe Google” care conduce la pagina de recenziere a Google Maps.

#### 3.4.10. Modulul destinat înrolării unui salon

Codul sursă al acestui modul este prezentat în Anexa 1 – `app/accmenu/newpartener.js`.

Fig. 3.17. Modulul de vizualizare a istoricului clientului

În cadrul acestui modul este reprezentat un formular pentru înscrierea unui nou salon în aplicație și implicit extinderea rolului utilizatorului curent la cel de partener. Câmpurile text aferente informațiilor de natură legală și de contact sunt validate local și transmise alături de token-ul utilizatorului către endpointul <https://api.glowapp.ro/newpartener> prin metoda PUT în format JSON.

În situația favorabilă, un mesaj de tip Alert este afișat: „Mulțumim pentru interesul acordat unui parteneriat” și este redirecționat către noul modul „Salonul meu”. În cazul producerii unei erori pe server în procesul de înscriere, utilizatorul recepționează mesajul transmis de server într-o componentă Alert.

#### 3.4.11. Modulul Salonul meu

Codul sursă pentru acest modul se regăsește în Anexa 1 – `app/salonmanager/index.js`. Acest modul constituie trecerea la navigatorul secundar dedicat utilizatorilor cu rol de parteneri. Codul sursă al navigatorului secundar este reprodus în Anexa 1 – `app/salonmanager/_layout.js`.




Salonul meu


Nume salon

Gioia Beauty Center

Logo



Imagini de prezentare



Contact

Gioia SRL

221121

Str. Aurel Persu 10, Bucuresti

gioiasalon@yahoo.com

0744603099

SALVEAZĂ INFORMAȚIILE

Salonul meu

Servicii

Programări

Acasa

Salonul meu

Contact

Gioia SRL

221121

Str. Aurel Persu 10, Bucuresti

gioiasalon@yahoo.com

0744603099

SALVEAZĂ INFORMAȚIILE

Program

Luni:	09:00	-	12:00
Marti:	09:00	-	22:00
Miercuri:	09:00	-	22:00
Joi:	09:00	-	22:00
Vineri:	09:00	-	20:00
Sambata:	Deschid	-	Închider
Duminica:	Deschid	-	Închider

SALVEAZĂ PROGRAMUL

Salonul meu

Servicii

Programări

Acasa

Fig. 3.18. Modulul Salonul meu

Modulul Salonul meu este dedicat exclusiv utilizatorilor care dețin un salon înregistrat în platformă, accesul fiind condiționat de asocierea unui partenerID validat anterior în navigatorul principal. La accesarea acestui modul este reapelat endpointul <https://api.glowapp.ro/user> pentru a confirma asocierea cu un salon și pentru a obține datele societății comerciale.

În cadrul acestei secțiuni utilizatorul poate gestiona detaliile salonului pe care îl administrează: denumirea afișată în platformă, numele legal al societății comerciale, Codul Unic de Înregistrare (CUI), adresă, email, număr de telefon. Actualizarea datelor se realizează prin accesarea endpointului PUT <https://api.glowapp.ro/partenerUpdate>. Programul de funcționare este afișat pe zilele săptămânii și este actualizat în server prin endpointul PUT de la adresa <https://api.glowapp.ro/program>.

Pentru actualizarea logo-ului sau încărcarea a noi imagini de prezentare se utilizează dependența ImagePicker a pachetului expo-image-picker prin care se rezolvă și permisiunea asupra materialelor media de pe dispozitiv. Astfel, se poate selecta o imagine din galerie spre a fi încărcată pe server. Imaginile sunt transmise la endpointul <https://api.glowapp.ro/newsalonphoto> prin metoda POST.

Ștergerea unei fotografii se realizează prin apăsare lungă (dependența onLongPress) ce declanșează apelul la endpoint-ul DELETE <https://api.glowapp.ro/{partenerID}/images/{număr imagine}>.

Modificările efectuate în fiecare secțiune declanșează o logică de împrăștiere a întregului set de date la fiecare schimbare de focus sau acționare a unuia din butoanele de salvare asigurând coerența între informațiile afișate local și cele stocate pe server.

### 3.4.12. Modulul Servicii

Codul sursă pentru acest modul se regăsește în Anexa 1 – app/salonmanager/servicii.js.

**Servicii**

Cosmetica

**Servicii existente**

Pensat	60.00	10	
Epilat axila	25.00	7	
Epilat brate	30.00	10	
Epilat total	180.00	45	

**Adaugă serviciu nou**

Denumire Preț Durată

Adaugă

Salonul meu Servicii Programări Acasa

Fig. 3.19. Modulul Servicii

Modulul Servicii oferă partenerului o interfață pentru gestionarea serviciilor disponibile în cadrul fiecărui departament. Departamentele sunt preluate din server prin endpointul GET <https://api.glowapp.ro/departamente>. Selecția departamentului se realizează printr-o componentă de tip Dropdown a pachetului react-native-element-dropdown. În cadrul acestei componente există și opțiunea de adăugare a unui nou departament sau eliminarea unuia din listă prin endpointul <https://api.glowapp.ro/departamente> apelat sub metoda DELETE. Opțiunea „Adaugă un departament” din cadrul elementului Dropdown activează modulul următor.

În urma selecției unui departament, lista de servicii este preluată prin endpointul GET <https://api.glowapp.ro/servicii> alături de ID-ul unic al departamentului selectat. Fiecare serviciu prezintă trei câmpuri editabile: denumire, preț, durată (exprimată în minute). Modificările în câmpuri sunt transmise automat către server la endpointul PUT <https://api.glowapp.ro/servicii/:ID>. Același endpoint este accesat prin metoda DELETE în situația apăsării butonului aferent eliminării unui departament. Acest buton este exprimat vizual printr-un coș amplasat paralel cu serviciul respectiv.

Sub lista curentă este prezentat un formular dedicat adăugării unui serviciu nou cu aceleași trei câmpuri descrise anterior. Transmiterea acestuia către server se procesează prin apel la endpointul POST <https://api.glowapp.ro/servicii>.

Funcționalitățile cuprinse în acest modul contribuie la eficiența operațională a salonului de înfrumusețare și permite partenerului să își adapteze rapid portofoliul de servicii.

### 3.4.13. Modulul destinat adăugării unui departament

Codul sursă aferent este prezentat în Anexa 1 – app/salonmanager/(modal)/adddepart.js. Acest modul face parte dintr-un layout izolat: Anexa 1 – app/salonmanager/(modal)/\_layout.js

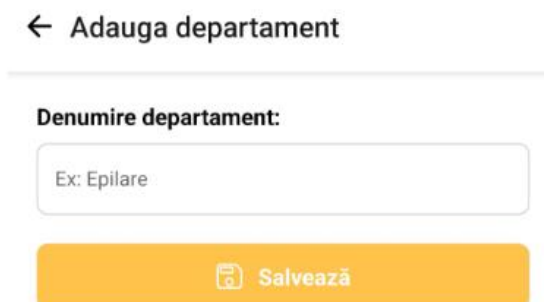


Fig. 3.20. Modulul destinat adăugării unui departament

Modulul de adăugare a unui nou departament este un formular cu un singur câmp al cărui buton de confirmare face apel cu textul introdus către endpointul POST <https://api.glowapp.ro/departamente>. Interfața este simplificată și rulează într-un navigator modal separat de fluxul aplicației. În cadrul acestui layout casetele de navigare de la baza paginii sunt ascunse și funcționalitatea le este înlocuită de o săgeată de revenire în antetul paginii.

### 3.4.14. Modulul Programări (salon)

Codul sursă al acestui modul se regăsește în Anexa 1 – app/salonmanager/programari.js.

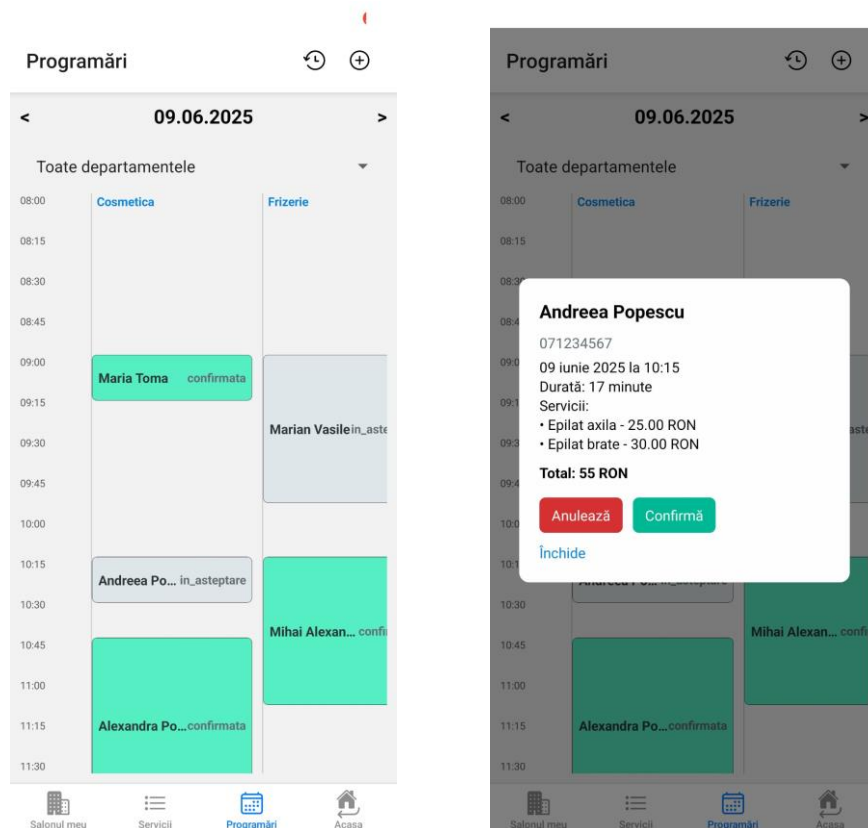


Fig. 3.21. Modulul Programări (salon)

Modulul de programări din cadrul interfeței de administrare al salonului este menit să înlocuiască complet clasicul programator fizic printr-o soluție digitală și interactivă.

Navigarea între zile se poate face fie prin săgețile încorporate în controale TouchableOpacity, fie prin glisarea spre stânga sau spre dreapta într-o zonă liberă a ecranului. Acțiunea prin gesturi este realizată prin controlul de tip PanResponder importat din pachetul react-native.

În funcție de data selectată este configurat și apelat endpointul GET <https://api.glowapp.ro/programarisalon> prin care se obține de la server lista de programări. Acest apel are loc la fiecare 10 secunde, garantând sincronizarea și afișarea în timp real a programărilor.

Programările sunt prezentate într-o interfață ce simulează o agendă fizică cu intervale notate în partea stângă. Mărimea casetei aferente programării reflectă durata totală a execuției programării. Chenarele conțin numele persoanei programate și sunt stilizate astfel încât să reflecte statusul programării (gri pentru cele în așteptare și verde pentru cele confirmate).

Apăsarea uneia din programări activează o fereastră modală din care partenerul o poate confirma sau anula. Atât butonul „Anulează” cât și „Confirmă” din fereastra modală își îndeplinesc utilitatea prin apel la endpointul PUT <https://api.glowapp.ro/programaristatus> încărcat cu ID-ul unic al programării și statusul aferent butonului acționat.

Prin navigatorul descris în `app/salonmanager/_layout.js`, această pagină are configurate în antet două butoane TouchableOpacity cu iconițe sugestive ce conduc către modulul de adăugare programare din sursă externă și respectiv modulul aferent istoricului de programări.

#### 3.4.15. Modulul istoricului de programări al salonului de înfrumusețare

Codul sursă aferent este prezentat în Anexa 1 – `app/salonmanager/(modal)/historyprog.js`. Modul este izolat prin navigatorul descris în Anexa 1 – `app/salonmanager/(modal)/layout.js`.

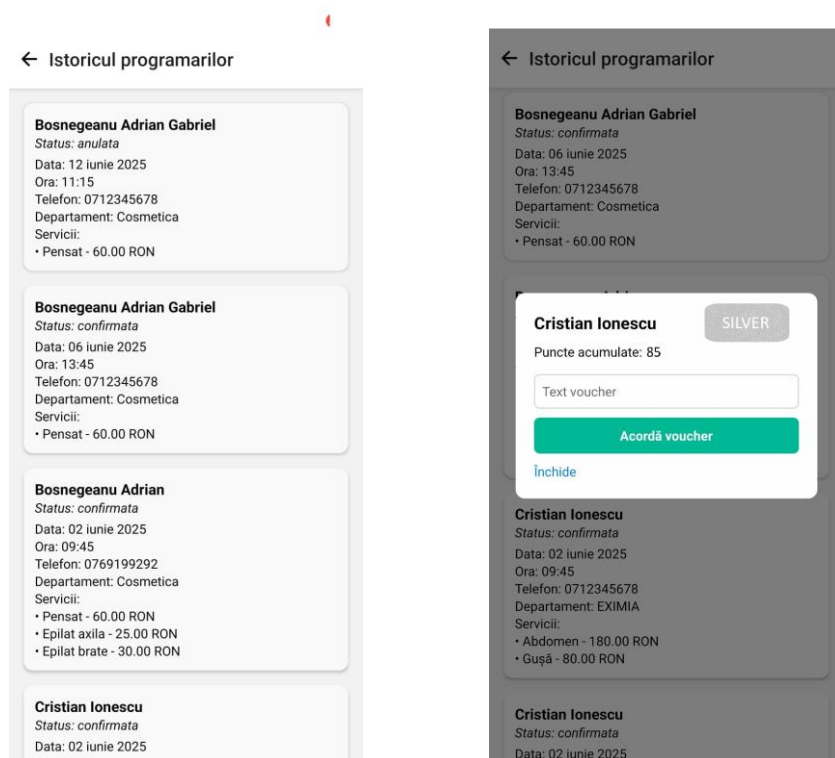


Fig. 3.22. Modulul istoricului de programări al salonului de înfrumusețare

Acest modul apelează la montare endpointul <https://api.glowapp.ro/programarisalon> prin metoda GET și obține întregul istoric ordonat cronologic. Casetele din acest ecran sunt butoane TouchableOpacity ce deschid o fereastră de tip Modal cu informații despre nivelul de loialitate aferent clientului și de numărul de puncte acumulate la salonul curent.

Punctele de loialitate și nivelul de loialitate se obțin prin apel cu parametrul ID al clientului la endpointul GET <https://api.glowapp.ro/loialitate>.

### 3.4.16. Modulul destinat adăugării unei programări din sursă externă

Codul sursă aferent este prezentat în Anexa 1 – `app/salonmanager/(modal)/progext.js`. Modul este izolat prin navigatorul descris în Anexa 1 – `app/salonmanager/(modal)/layout.js`.

Fig. 3.23. Modulul destinat adăugării unei programări din sursă externă

Acest modul permite partenerului să introducă manual o programare pentru un client fără cont oferind o structurare pe pași, similar cu 3.4.4. Modulul de programare (destinat clienților). Diferența între aceste două ecrane o constituie partea de final a formularului, unde partenerul trebuie să introducă un nume și un număr de telefon pentru clientul ce se programează.

Acționarea butonului „Programează” conduce la accesarea endpointului POST <https://api.glowapp.ro/programareext>, cerere însoțită de datele aferente introduse în formular și de informațiile de identificare ale salonului. Răspunsul favorabil din partea serverului conduce la o alertă de succes și redirecționarea către Modulul Programări (salon).

## 3.5. Elemente server

Platforma dispune de un server virtual a cărei configurație a fost realizată prin conexiune la distanță din mediul de dezvoltare Visual Studio Code. Sistemul de operare pre-instalat este

Debian GNU/Linux 12, iar conexiunea remote pentru configurarea avansată a serverului s-a realizat prin tunel SSH la adresa IP a acestuia.

### 3.5.1. Comunicarea client-server

În cadrul platformei, comunicarea client-server s-a realizat prin conexiune securizată HTTPS la domeniul personalizat „glowapp.ro”. Prin configurarea DNS în platforma cPanel, subdomeniile api.glowapp.ro și www.glowapp.ro au fost asociate cu adresa publică a serverului.

Gestionarea traficului în rețea și rutarea conform cererilor este realizată prin serviciul NGINX. Astfel, cererile primite pe api.glowapp.ro sunt redirecționate intern către logica de backend construită cu Node.js care rulează pe portul 3000. Cererile recepționate prin subdomeniul www.glowapp.ro sunt direcționate către o instanță a aplicației web.

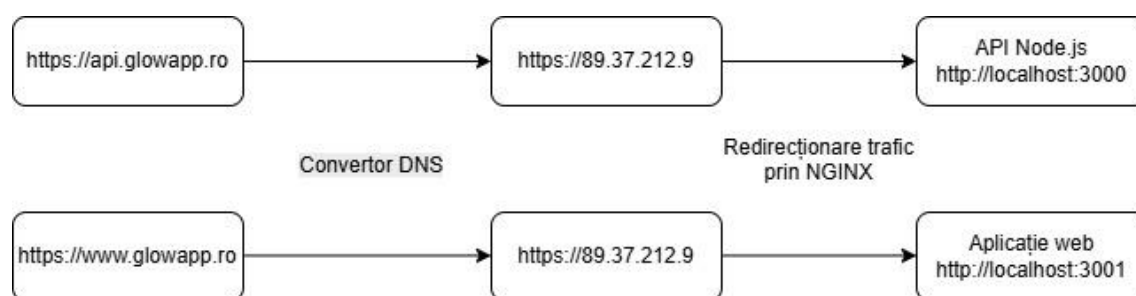


Fig. 3.24. Diagrama de rutare a traficului în rețea

Logica de backend este implementată în limbajul JavaScript și rulează prin Node.js pe portul 3000 al serverului. Fiecare endpoint este accesat prin adresa https://api.glowapp.ro.

### 3.5.2. Pachete și dependențe

Pentru configurarea unei logicii de backend pe structura API REST în limbajul JavaScript, se instalează Node.js. Baza de date se implementează prin soluția software MySQL.

Dependențele necesare implementării serverului platformei au fost instalate fie prin intermediul Advanced Package Tool (apt), preimplementat în Linux, fie prin Node Package Manager (npm). Toate comenzile efectuate în terminal sunt documentate în Anexa 2 – Configurare server.

Rularea și resetarea facilă a aplicației Node.js a implicat utilizarea PM2, un manager open-source de procese ce optimizează și procesul de gestionare a erorilor.

În vederea conectării într-un mod formal și securizat la server, s-a optat pentru achiziționarea domeniului „glowapp.ro”. Redirecționarea traficului de pe portul principal 433 pe portul 3000 pe care rulează aplicația Node.js s-a realizat prin pachetul NGINX. (Fig. 3.22)

În cadrul API-ului Node.js propriu-zis s-au folosit o serie de module necesare procesului de dezvoltare. Modulul „express” facilitează scrierea de cod propriu-zisă prin simplificarea structurii Node.js clasice. Pachetul „body-parser” permite recepționarea fișierelor JSON în corpul cererilor trimise la endpoint-uri și simplifică importarea de variabile din apel direct în cod.

Accesul din domenii sau IP-uri externe la serverul intern Node.js este asigurat de pachetul „cors” (Cross-Origin Resource Sharing). Pentru conexiunea la baza de date s-a importat biblioteca modernă de gestionare a bazelor de date MySQL: „mysql2”.

În ceea ce privește gestionarea memoriei interne a serverului, biblioteca „multer” este necesară încărcării de fișiere pe server. Modulul „fs” din Node.js permite lucrul cu directoarele și fișierele din memoria internă a serverului și este necesar pentru structurarea depozitării de materiale media pe server. O altă dependență ce simplifică manipularea de fișiere este „path”, al cărei rol s-a dovedit important în gestionarea extensiilor materialelor fotografice (.jpg, .png, .jpeg).

Pentru automatizarea anumitor procese s-a utilizat biblioteca „node-cron” ce intervine în sarcini recurente declanșate de timp. Pentru sistemul de notificare, s-a utilizat „Resend” la trimiterea de mailuri tranzacționale și „expo-server-sdk” în ceea ce privește notificările push către dispozitivele mobile. Integrarea serviciilor API externe a impus și instalarea dependenței „node-fetch”

### 3.5.3. Baza de date

Platforma informatică are în componență o bază de date relațională, implementată în MySQL. În cadrul acesteia, este stocat întreg fluxul de informații necesare funcționalităților implementate. Proiectarea a fost realizată cu respectarea principiilor normalizării. Instrucțiunile SQL prin care a fost creată baza de date sunt documentate în Anexa 3 – MySQL.

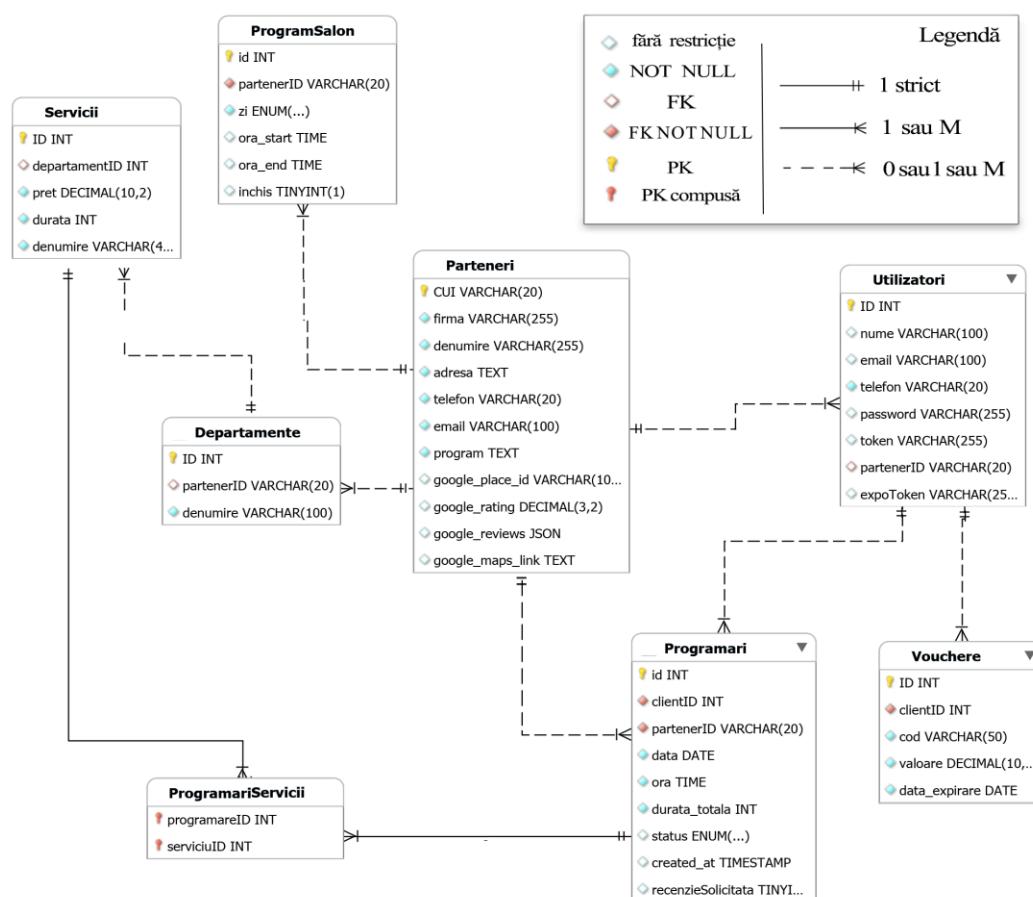


Fig. 3.25. Modelul entitate-asociere al bazei de date

Tabela Utilizatori este punctul de plecare în cadrul platformei. Datele fiecărui utilizator sunt memorate aici în câmpurile: ID, nume, email, telefon, parolă, token, expoToken și cheia externă partenerID. ID-ul este o componentă autoincrementată natural în logica tabelii. Cu



excepția numărului de telefon și a cheiei primare, coloanele au un caracter opțional pentru a putea fi gestionate și programările externe adăugate de parteneri fără nevoia unei tabelă suplimentare. În urma permisiunii acordate de utilizator, coloana expoToken reține un cod unic similar unei adrese virtuale la care se pot recepționa notificările de tip push. Cheia străină, partenerID, marchează extensia rolului de utilizator la cel de partener.

Tabela Parteneri este cea aferentă saloanelor. Cheia primară o reprezintă chiar Codul Unic de Înregistrare al societății comerciale. În aceeași tabelă sunt prezente informațiile juridice precum numele oficial al firmei, dar și date suplimentare: denumirea consacrată public, adresa, numărul de telefon și email. Ulterior, tabela conține patru coloane opționale în care sunt reținute detalii publice ale salonului, extrase automat prin API-ul oferit de Google Places. Coloana google\_place\_id reține identificatorul locației în rețeaua Google pentru rechiziționări de date ulterioare. În câmpul google\_rating este reținut și actualizat scorul obținut în platforma de recenzii Google și google\_reviews este un element JSON ce include ultimele cinci recenzii obținute de salon alături de numele și iconița de profil a celui ce a postat recenzia. Ultima coloană, google\_maps\_link, este adresa URL a locației din Google Maps. Tabela ProgramSalon este o extensie a tabelii Parteneri și reține pe zile ale săptămâni programul pentru fiecare salon de înfrumusețare.

Tabela Departamente este un intermediar către tabela Servicii și are ca scop filtrarea facilă în interfață și server. Constă în trei coloane: ID, denumire și cheia externă partenerID.

În tabela Servicii se regăsesc toate cele adăugate de salonul de înfrumusețare alături de departamentul lor, preț și durată. Relația many-to-many dintre tabela Servicii și tabela Programări a fost gestionată printr-o tabelă auxiliară, ProgramăriServicii cu doar două coloane, ambele chei secundare și totodată cheie compusă pentru această tabelă.

Tabela Programări reține toate informațiile aferente unei programări precum cheia proprie, autoincrementată, clientID-ul, partenerID-ul, data, ora și durata totală. Suplimentar, pentru eventuale analize viitoare, este reținut un timestamp al momentului în care a fost creată. Coloana status acceptă trei valori: „anulată”, „în\_asteptare” sau „confirmată”. Ultima coloană reprezintă un indicator boolean ce reține dacă au fost trimise solicitările de recenzie în privința acestei programări, astfel încât să nu fie îngreunată experiența utilizatorului prin solicitări repetate.

Tabela Vouchere include suplimentar față de cheia primară, un cod de identificare și valoarea sau mesajul aferent acestuia. Ultima coloană a acestei tabelă este data de expirare a promoției recepționate.

### 3.5.4. Endpointuri API

Pentru o sinteză clară a modului în care platforma gestionează datele prin API, acest capitol este însoțit de diagrama de clase. Codul sursă complet al serverului Node.js este disponibil în Anexa 4 – server.js.



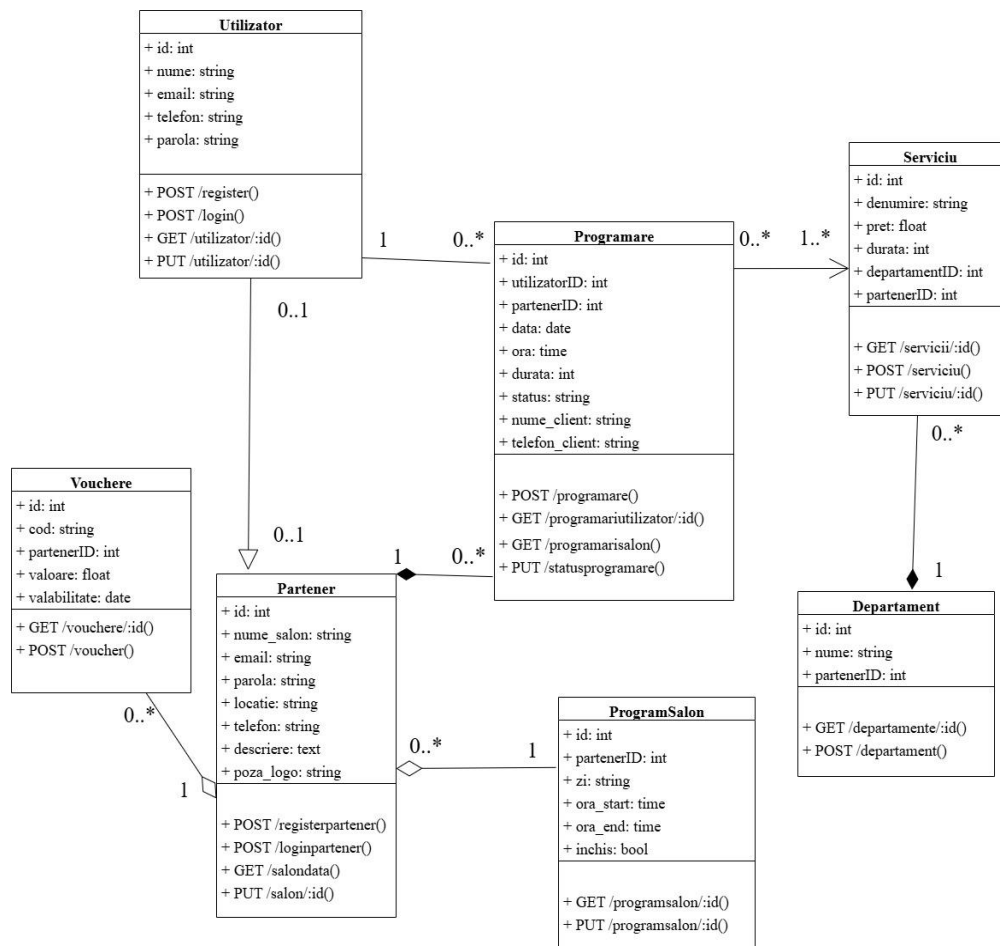


Fig. 3.26. Diagrama de clase a platformei

Pentru metoda GET, sunt configurate 16 endpointuri:

- <https://api.glowapp.ro/>

Acesta este un endpoint menit să testeze dacă API-ul funcționează și va returna întotdeauna mesajul „API-ul este funcțional”.

- <https://api.glowapp.ro/user>

În funcție de token, primit ca parametru, pe baza căruia se execută o interogare în tabela Utilizatori unită prin LEFT JOIN cu tabela Parteneri. Se returnează informațiile obținute despre utilizator și, dacă este cazul, despre salonul partener pe care îl reprezintă. Răspunsul va fi de forma: user.id, user.nume, user.emai, user.telefon, partener.firma, partener.partenerID, partener.denumire, partener.telefon, partener.email.

- <https://api.glowapp.ro/departamente>

Apelul include ca parametru token-ul utilizatorului și caută partenerID-ul corespunzător în tabela Utilizatori. Odată identificat, se folosește partenerID pentru căutarea departamentelor aferente din tabela Departamente.

- <https://api.glowapp.ro/program>

Acest endpoint returnează programul săptămân al unui salon în baza tokenului de utilizator al reprezentantului. Prima căutare se efectuează în tabela Utilizatori de unde este

extrasă cheia primară a salonului. Ulterior, se caută după această cheie în tabela ProgramSalon și se returnează numele zilei, ora deschiderii, ora închiderii pentru fiecare zi din săptămână

- <https://api.glowapp.ro/servicii>

Parametrul așteptat de acest endpoint este cheia primară din tabela Departamente pe baza căreia sunt extrase serviciile. În răspuns sunt cuprinse pentru fiecare serviciu denumirea, prețul și durata.

- <https://api.glowapp.ro/:partenerID/logo>

Prin această rută se poate accesa logo-ul salonului după identificatorul său unic. Logica de backend caută la apelare în folderul local aferent salonului un fișier de tip imagine cu denumirea logo și una din extensiile jpg, png sau jpeg. Dacă este găsit, este returnat ca răspuns vizual, iar în caz contrar se returnează un mesaj de eroare

- <https://api.glowapp.ro/:partenerID/list/images>

Endpoint conceput să returneze o lista cu denumirile imaginilor aferente unui partener. Singurul parametru este identificatorul salonului de înfrumusețare. La apelare, serverul scanează folderul local de materiale media aferent partenerului și oferă o listă de elemente prezente.

- <https://api.glowapp.ro/:partenerID/:imageNumber>

Acest endpoint este utilizat pentru a accesa în mod direct un element media din galeria unui salon. Numărul imaginii este extras din titlu alături de identificatorul partenerului, astfel încât serverul să indice exact către materialul cerut.

- <https://api.glowapp.ro/disponibilitate>

Se verifică recursiv disponibilitatea orară a unui salon pentru o anumită zi, ținând cont de programările deja existente. Sunt necesari următorii parametri: ID-ul partenerului, data aleasă, durata totală a serviciilor și denumirea departamentului. Rezultatul este o listă de intervale disponibile, din 15 în 15 minute, în care clientul poate fi programat.

- <https://api.glowapp.ro/loialitate>

Parametrul necesar acestui endpoint este token-ul utilizatorului. Pe baza acestuia, se identifică ID-ul clientului în tabela Utilizatori și este folosit pentru a îi calcula timpul petrecut în saloanele de înfrumusețare conform istoricului programărilor sale. Pentru fiecare 30 de minute, el acumulează 10 puncte de loialitate. Este returnat apoi numărul de puncte acumulate, nivelul cardului de loialitate și eventualele vouchere recepționate din tabela Vouchere a bazei de date

- <https://api.glowapp.ro/loialitateclient>

În baza parametrilor de identificare pentru client și partener, endpointul stabilește punctele de loialitate pe care un client le-a obținut exclusiv la salonul partenerului. Această rută este utilă în decizia de a acorda vouchere din partea salonului de înfrumusețare.

- <https://api.glowapp.ro/program>

Prin parametrul de identificare al partenerului, este interogată tabela ProgramSalon din baza de date și este returnat intervalul orar în care salonul este deschis. Răspunsul serverului este structurat pe zile ale săptămânii.

- <https://api.glowapp.ro/saloane>

Acest endpoint este proiectat să returneze saloanele sub formă de pagini. Interogarea se face cu o limită de 10 saloane, iar ca parametru este transmis numărul paginii cerut de frontend.

- <https://api.glowapp.ro/programariclient>

Prin parametrul de identificare al clientului, este interogată tabela Programări din baza de date. Instructiunea SQL execută JOIN-uri cu tabelele ProgramariServicii și Servicii în vederea extragerii informațiilor complete aferente programărilor viitoare exclusiv ale clientului. Sunt returnate id-ul programării, data, ora, durata totală, statusul, salonul și serviciile pentru fiecare programare identificată

- <https://api.glowapp.ro/istoricclient>

Asemănător endpointului anterior, pe baza token-ului primit din frontend se interoghează baza de date, dar fără condiții legate de data sau ora programărilor. Astfel, se returnează lista completă de programări pe care clientul le-a efectuat.

- <https://api.glowapp.ro/programarisalon>

Endpoint-ul returnează toate programările primite de un salon, prin parametrul token primit în cerere. Identificatorul salonului este extras din tabela Utilizatori și ajută la interogarea tabelor Programari, Servicii, ProgramariServicii și Departamente în vederea returnării unei liste complexe cu programările alături de numele clientului, telefonul, statusul programării, departamentele, data și ora, dar și lista de servicii programate.

Pentru metoda POST, serverul prezintă 11 endpointuri:

- <https://api.glowapp.ro/login>

Acest endpoint este responsabil pentru autentificarea utilizatorilor în aplicație. Cererea așteaptă un body care să conțină câmpurile email și parolă pe care serverul le verifică cu informațiile cuprinse în tabela Utilizatori. Dacă sunt validate, se generează și salvează un token unic printr-o funcție personalizată ce cuprinde prefixul din email, un timestamp și caractere aleatorii. Ca răspuns, serverul returnează numele utilizatorului, token-ul și un mesaj de succes.

- <https://api.glowapp.ro/register>

Responsabilitatea acestui endpoint este validarea unui formular de înregistrare și în caz favorabil, încărcarea informațiilor în baza de date.

- <https://api.glowapp.ro/token>

Endpointul primește un token de la frontend și îi verifică validitatea în baza de date. Acesta returnează numele și prenumele aferente token-ului și este util în mneținerea utilizatorului autentificat.

- <https://api.glowapp.ro/salonlogochange>

Printr-un formular de tip multipart/form-data, endpointul primește din frontend o imagine în format jpg, jpeg sau png și o salvează drept logo pentru un salon de înfrumusețare. Salonul în cauză este identificat prin asocierea parametrului token cu identificatorul de partener din baza de date. În cadrul acestor instrucțiuni sunt gestionate și situații precum lipsa directorului media conform ierarhiei gândite sau eliminarea logo-ul vechi în cazul identificării lui pe server.

- <https://api.glowapp.ro/newsalonphoto>

Asemănător endpointului anterior, o cerere tip multipart/formdata cu un material fotografic și token-ul utilizatorului este gestionată prin identificarea folderului corespunzător și stocarea imaginii primite în acesta, respectând convenția de numerotare a pozelor.

- <https://api.glowapp.ro/programare>

Acest endpoint este responsabil de înregistrarea unei programări din partea unui client identificat prin token. Identificatorul salonului trebuie să fie prezent între parametrii alături de data, ora, durata și lista de servicii dorite. Informațiile sunt restructurate și încărcate în tabelele Programari și ProgramariServicii.

- <https://api.glowapp.ro/programareext>

Endpointul este asemănător cu cel anterior, dar dedicat înregistrării unei programări direct din partea partenerului. Primește ca parametrii toate informațiile aferente unei programări (partenerID, data, ora, durata, servicii) alături de un nume și telefon al clientului și le încarcă în tabelele Programari și ProgramariServicii.

- <https://api.glowapp.ro/vouchere>

Prin acest endpoint este permisă transmiterea și înregistrarea unui voucher pentru un client pe baza codului de identificare al clientului, valoarea voucherului și data de expirare. Prin dependența expo-server-sdk este trimisă o notificare de tip push către dispozitivul utilizatorului și un email prin intermediul soluției celor de la Resend.

- <https://api.glowapp.ro/servicii>

Endpointul dedicat adăugării unui nou serviciu așteaptă ca parametrii identificatorul departamentului, denumirea, prețul și durata noului serviciu. În urma validării, noul serviciu devine vizibil în întreaga platformă.

- <https://api.glowapp.ro/departamente>

În baza parametrilor token și denumire departament, logica din backend interoghează baza de date, identifică codul unic al salonului asociat tokenului și adaugă în tabela Departamente o nouă entitate.

- <https://api.glowapp.ro/salveazatoken>

Acest endpoint este destinat salvării tokenului aferent serviciului de push-notification. La primirea cererii cu token-ul și expotoken-ul clientului se actualizează tabela Utilizatori.

Metoda PUT are atribuite 6 endpointuri::

- <https://api.glowapp.ro/userUpdate>

Endpointul este destinat actualizării informațiilor de profil a utilizatorului autentificat și primește ca parametrii token-ul pentru identificare alături de un corp populat de câmpurile nume, email și telefon. În urma validării datelor se execută o instrucțiune SQL de UPDATE pe tabela Utilizatori.

- <https://api.glowapp.ro/partenerUpdate>

În ceea ce privește rolul de partener, utilizatorul poate actualiza informațiile salonului prin acest endpoint. Solicitarea va conține token-ul în antet, iar în body se vor regăsi elementele:

numePartener, firma, telefon, email, adresa și Codul Unic de Identificare al societății comerciale. Validarea datelor conduce la executarea unui query SQL de UPDATE asupra tabelii Parteneri, dar și refacerea legăturii cheiei secundare a tabelii Utilizatori în cazul modificării de CUI. Se redenumeste și folderul aferent materialelor media pentru a menține logica de ierarhie implementată.

- <https://api.glowapp.ro/newpartner>

Acest endpoint permite înregistrarea unui nou salon în sistem. În body sunt trimise CUI-ul, denumirea firmei, numele locației, adresa, numărul de telefon și emailul. Pe baza token-ului transmis în antet se asociază salonul cu utilizatorul care l-a înscris.

- <https://api.glowapp.ro/servicii>

Prin acest apel se pot actualiza denumirea, prețul sau durata unui serviciu existent. Parametrul ID al serviciului este transmis și pe baza acestuia se actualizează rândul aferent din tabela Servicii.

- <https://api.glowapp.ro/program>

Dedicat actualizării programului orar al salonului, endpoint-ul primește un array de obiecte structurat pe zilele săptămânii, ora deschiderii și ora închiderii. Se preia token-ul din antet, se identifică utilizatorul și salonul asociat și se execută un UPDATE cu noul program asupra tabelii ProgramSalon. Automat, dacă o zi nu conține ore valide, este marcată drept „închis”.

- <https://api.glowapp.ro/programaristatus>

Acest endpoint are ca parametru identificatorul unei programări și în corpul solicitării câmpul status. Astfel, se actualizează în baza de date noul stadiu al programării. Se verifică dacă statusul curent diferă de cel nou pentru validare și se trimit notificări push și prin email către client, în funcție de noul status.

Logica backend conține patru endpoint-uri de tip DELETE:

- <https://api.glowapp.ro/salon>

Acest endpoint șterge complet un partener din sistem pe baza token-ului de utilizator din antetul cererii.

- <https://api.glowapp.ro/departamente>

Parametrul acestui endpoint este identificatorul unic al departamentului în cauză și permite ștergerea lui în cascadă cu serviciile asociate.

- <https://api.glowapp.ro/servicii>

Prin parametrul ID al serviciului se execută o instrucțiune DELETE asupra unui serviciu din tabela Servicii a bazei de date.

- <https://api.glowapp.ro/programare>

Endpointul de ștergere a unei programări intervine la solicitarea clientului și survine cazului în care o programare a fost efectuată accidental. Parametrul principal este identificatorul

programării și realizează ștergerea în cascadă a programării atât din tabela Programari, cât și din cea de legătură, ProgramariServicii.

- <https://api.glowapp.ro/images>

Imaginile sunt salvate pe server sub forma „numărimagine.ext”. Astfel, acestui endpoint îi este necesar ca parametru numărul imaginii și identificatorul unic al partenerului pentru a efectua ștergerea ei din directorul /srv/data/{ID-ul partenerului}/media. Dacă fișierul este găsit, acesta este șters prin instrucțiunea fs.unlinkSync.

### 3.5.5. Sarcini automate și servicii externe

În logica de backend sunt implementate anumite procese automate acționate fie în urma anumitor acțiuni, fie de evenimente cronologice.

Serviciul oferit de Resend a permis configurarea domeniului „glowapp.ro” pentru trimiterea de emailuri automate cu aspect profesional. Pentru clienți, emailurile sunt trimise automat în urma modificării statusului unei programări (la confirmare sau la anulare) și sunt construite în format HTML. Conținutul este adaptat dinamic cu toate informațiile aferente programării. În mod similar, saloanele de înfrumusețare partenere sunt notificate prin email atunci când recepționează o programare sau când clientul decide să o anuleze.

Un email de mulțumire cu link către serviciul recenzionarea prin Google este trimis automat către client atunci când timpul aferent unei programări confirmate s-a scurs. Verificarea se realizează pe server-ul Node.js prin pachetul cron.schedule la fiecare 15 minute pentru întreaga platformă.

O altă verificare este realizat la fiecare cinci minute prin care se verifică dacă există programări în așteptare ce și-au depășit termenul definit și le anulează automat. Codul sursă aferent acestor sarcini automate este disponibil în Anexa 4 – server.js.

Integrarea cu Google Places este realizată printr-o secvență Node.js ce rulează zilnic la ora 03:00. API-ul celor de la Google impune limite reduse de utilizare gratuită, dar informațiile pe care le rechiziționăm prin acesta nu prezintă nevoia unei recurențe mai dese. Codul sursă prin care tabela Parteneri este actualizată zilnic cu informațiile de la Google este documentat în Anexa 4 – googlePlacesTask.js.

## 4. METODE DE SECURIZARE A DATELOR

Platforma informatică destinată saloanelor de înfrumusețare gestionează informații sensibile, precum date personale ale clienților, programări și istoricul serviciilor. Pentru a asigura confidențialitatea și integritatea acestor date, aplicația implementează mai multe metode de securizare, în conformitate cu reglementările în vigoare, inclusiv GDPR. Acest capitol descrie principalele măsuri aplicate pentru a preveni accesul neautorizat și pentru a proteja informațiile stocate și transmise în cadrul sistemului.

### 4.1. Certificatul SSL

Pentru a asigura confidențialitatea și integritatea datelor transmise între client și server, platforma implementează un certificat SSL (Secure Sockets Layer), emis gratuit de serviciul Let's Encrypt. Acest certificat permite criptarea traficului prin protocolul HTTPS, astfel încât informațiile precum datele de autentificare, programările sau datele de contact să nu poată fi interceptate sau modificate de terți în timpul transferului.

Prin utilizarea certificatului SSL, aplicația respectă bunele practici moderne de securitate și oferă utilizatorilor încrederea că interacțiunile lor cu platforma se desfășoară într-un mediu protejat. În cadrul aplicației, implementarea certificatului a fost realizată pe serverul de producție folosind utilitarul Certbot. Implementarea certificatului SSL a fost realizată prin intermediul comenzilor executate în terminal, conform pașilor descriși în Anexa 5 – SSL.

### 4.2. Mascarea datelor pe server

Pentru a proteja datele sensibile și pentru a menține un nivel ridicat de securitate în cadrul aplicației, a fost implementată practica de mascare a informațiilor critice prin utilizarea variabilelor de mediu. Această abordare are rolul de a separa configurațiile confidențiale de codul sursă, prevenind totodată și expunerea accidentală a acestora.

În fișierul `.env` aferent serverului Node.js au fost redactate cheile API pentru accesul la Google Places și Resend, datele de conectare la baza de date (host, username, parola și numele bazei de date). Astfel, în loc să apară direct în cod date precum parole sau chei de acces, se folosesc apeluri către variabilele importate. Revenind la Anexa 4 – `server.js` se poate observa modul în care s-a implementat această tehnică.

### 4.3. Token-ul de utilizator

Pentru a gestiona sesiunile și pentru a identifica utilizatorii în mod securizat, aplicația folosește token-uri de autentificare. Acestea sunt generate la logare și asociate direct fiecărui cont. Token-ul este folosit ulterior pentru a valida cererile din partea clientului, fără a fi nevoie să se transmită constant date sensibile precum parola sau adresa de email.

Logica de generare a token-ului este îmbinarea prefixului adresei de email a utilizatorului cu un identificator temporal care reflectă momentul exact al logării și o secvență de caractere aleatorii formate din litere și cifre. Acest procedeu minimizează șansele de coliziune. Codul aferent descrierii de mai sus se regăsește în Anexa 5 – `generateToken`.

#### 4.4. Criptarea datelor în baza de date

Securitatea datelor salvate în baza de date este o prioritate esențială pentru orice aplicație care gestionează informații personale sau sensibile. În cadrul platformei dezvoltate, au fost implementate două mecanisme complementare de protecție: hashingul datelor critice și criptarea binară a anumitor fluxuri de date. Scopul acestor metode este de a preveni accesul neautorizat, chiar și în situația în care baza de date ar fi compromisă.

##### 4.4.1. Hashing

Pentru protejarea parolelor utilizatorilor stocate în baza de date, aplicația implementează un mecanism de hashing pe baza algoritmului SHA-256. Acest algoritm, parte a familiei SHA-2, produce un hash de 256 de biți, fiind considerat în prezent o metodă sigură și eficientă împotriva atacurilor de tip dictionary sau brute force.

În momentul în care un utilizator își creează un cont sau își modifică parola, sistemul preia parola introdusă și aplică algoritmul SHA-256 pentru a o transforma într-un șir hexadecimale unic și ireversibil. Această valoare este cea care se stochează în baza de date, nu parola în clar. La autentificare, parola introdusă este trecută prin același proces, iar hash-ul rezultat este comparat cu cel existent în baza de date.

Implementarea metodei de hashing este documentată în Anexa 5 – Hashing.

##### 4.4.2. Criptarea flux în binar

Criptarea datelor personale în baza de date se face printr-o logică proprie bazată pe cifrul Vernam și este reinterpretată pentru a deservii criptării datelor pe server. Pentru implementarea algoritmului am ales limbajul JavaScript pentru o integrare directă și eficientă în API-uri bazate pe NodeJS.

Pentru început, algoritmul generează un șir de caractere (aleatorii) de aceeași lungime cu șirul de caractere transmis către criptare. Acest șir de caractere va reprezenta cheia. Metoda de criptare aleasă are la bază aplicarea operațiunii XOR între fiecare literă din mesajul inițial și corespondența ei din cheia generată. Textul criptat și cheia vor fi intercalate pentru procesarea rapidă a decriptării. Considerăm mesajul inițial „MESAJ” și cheia generată aleator „RANDOM”.

Text clar	M	E	S	A	J
Cod binar	010110	001110	011100	001010	010011
Cheie	R	N	D	O	M
Cod binar	011011	010111	001101	011000	010110
Rezultat XOR	D	P	H	I	5
Cod binar	001101	011001	010001	010010	000101

Tabelul 4.1. Simularea algoritmului de criptare

Rezultatul operațiunii XOR va fi DPHI5, iar mesajul final criptat, cu cheia intercalată devine „DRPNHDIO5M”.

Operația XOR produce întotdeauna un rezultat care are același număr de biți ca cel mai mare dintre operanzi. Astfel, am ales un spațiu de 5 biți ce ne permite un alfabet de 63 de caractere. Codul sursă aferent acestui algoritm de criptare se regăsește în Anexa 5 – cryptFlux.js și respectiv Anexa 5 – decryptFlux.js pentru cel de decriptare.



## 5. SCENARII DE FUNCȚIONARE ALE APLICAȚIEI

Funcționalitățile principale ale aplicației se pot remarca din diagrama cazurilor de utilizare prezentată în Fig. 5.1.

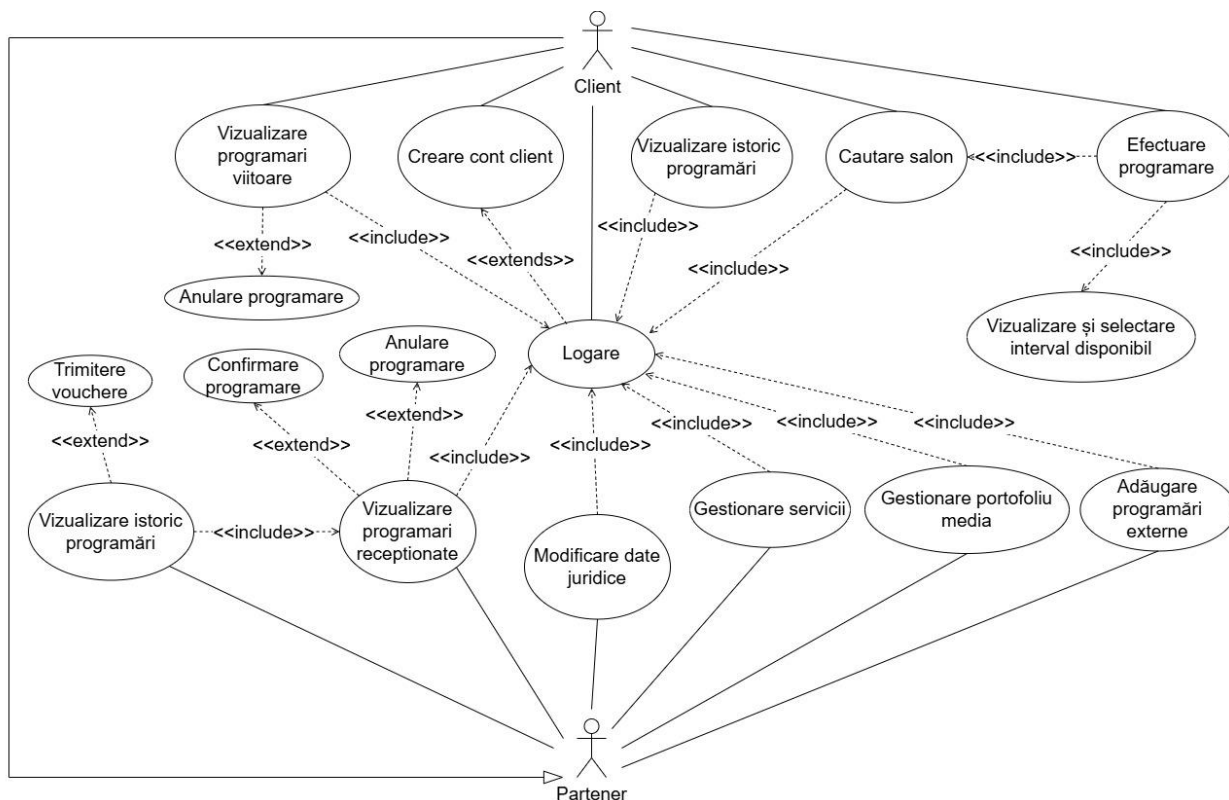


Fig. 5.1. Diagrama cazurilor de utilizare

În cadrul acestui capitol se vor explora scenariile de utilizare ale platformei, secvențial din perspectiva utilizatorului și respectiv din perspectiva prestatorului de servicii. Astfel, se poate înțelege mai precis experiența interactivă pe care platforma își propune să o contureze.

### 5.1. Efectuarea unei programări – client

Programările se pot efectua de către clienții aplicației înregistrați și autentificați în contul lor. Ulterior, în ecranul acasă îi sunt prezentate saloanele de înfrumusețare înscrise în platformă. Prin selectarea unui salon poate vizualiza informațiile extinse ale acestuia. Următorul pas este apăsarea butonului „Programează-te”, care redirecționează către o interfață dedicată selecției de servicii. Acestea sunt afișate în funcție de departamente și permit selectarea multiplă prin bifare. După alegerea serviciilor, aplicația calculează automat durata totală și trece utilizatorul în pasul următor: selecția datei și orei programării.

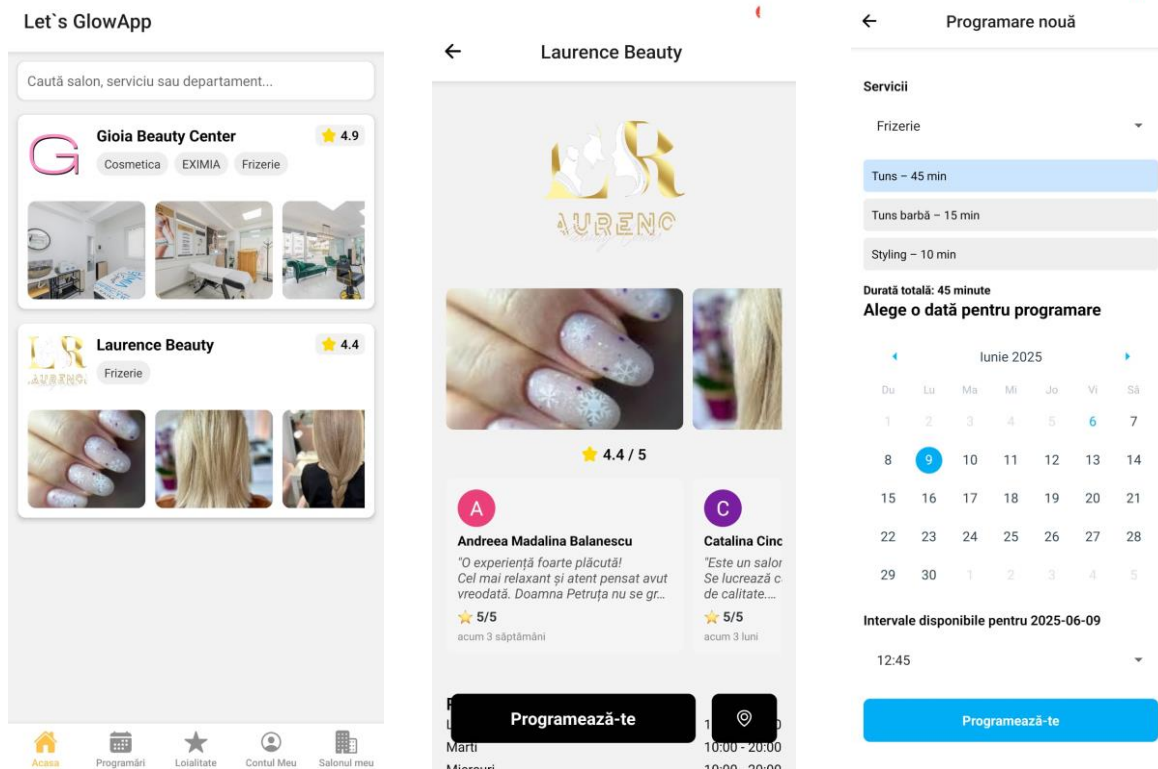


Fig. 5.2. Fluxul ecranelor în procesul de programare

Apăsarea butonului albastru „Programează-te” aferentul ultimului ecran validează cererea, adaugă programarea în baza de date cu statusul „în așteptare” și trimite un email și o notificare push către salonul de înfrumusețare.

În urma răspunsului din partea salonului, utilizatorul este notificat prin email și prin notificare push:

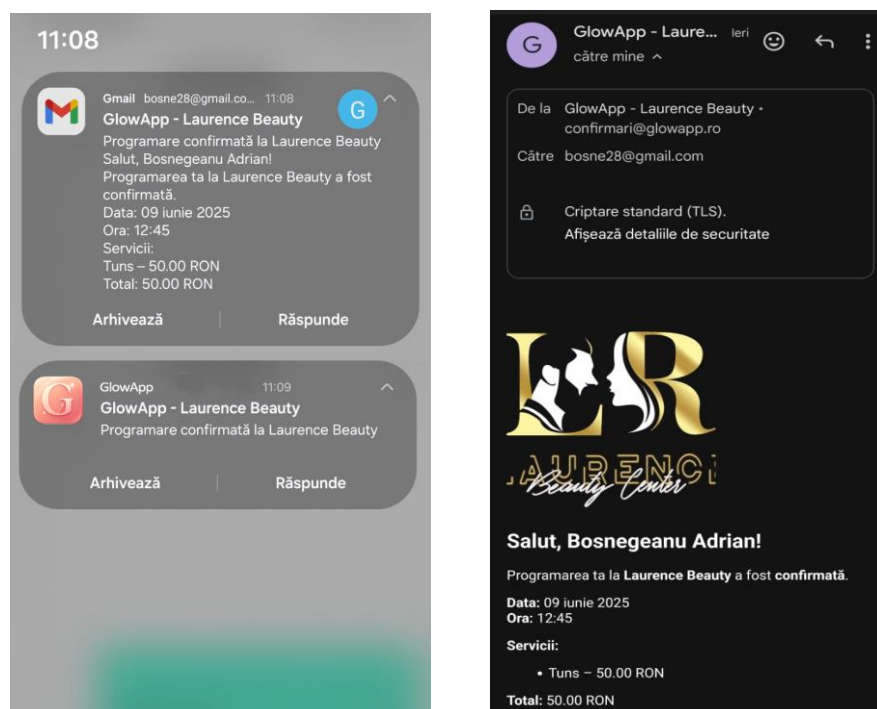


Fig. 5.3. Notificarea clientului și emailul de confirmare a programării

## 5.2. Recepționarea unei programări - partener

La înregistrarea unei noi programări în cadrul salonului, partenerul este notificat push și prin email referitor la aceasta.

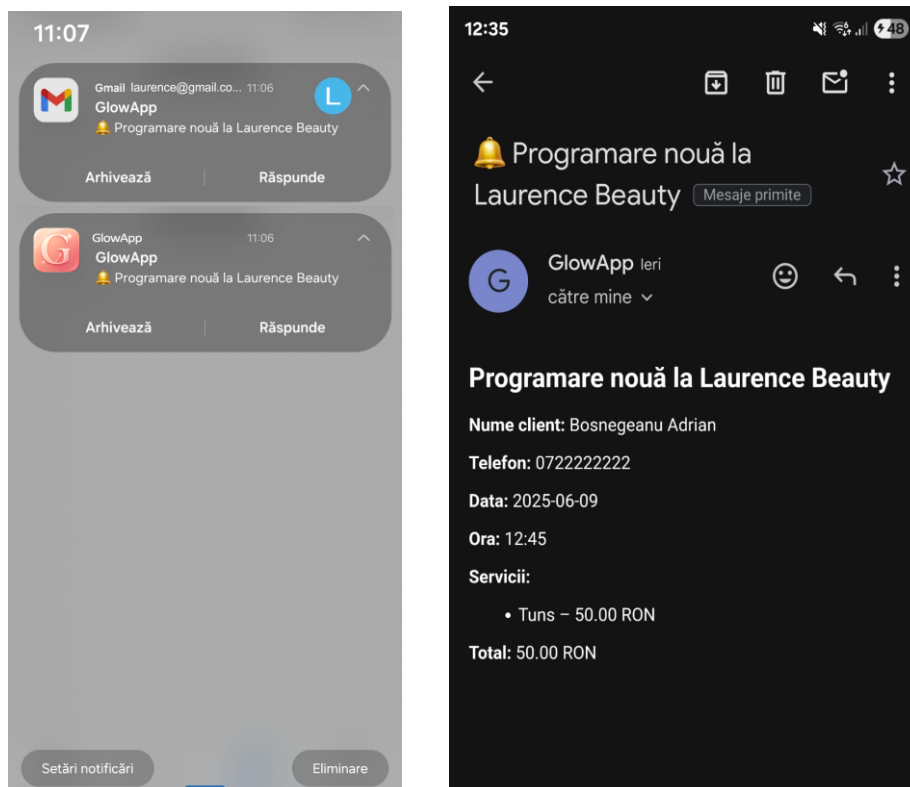


Fig. 5.4. Notificarea partenerului și emailul la recepționarea programării

În urma recepționării, partenerul poate accesa programatorul și prin apăsarea programării o poate confirma sau anula. Confirmarea programării conduce la notificarea clientului conform figuiei Fig.5.1.

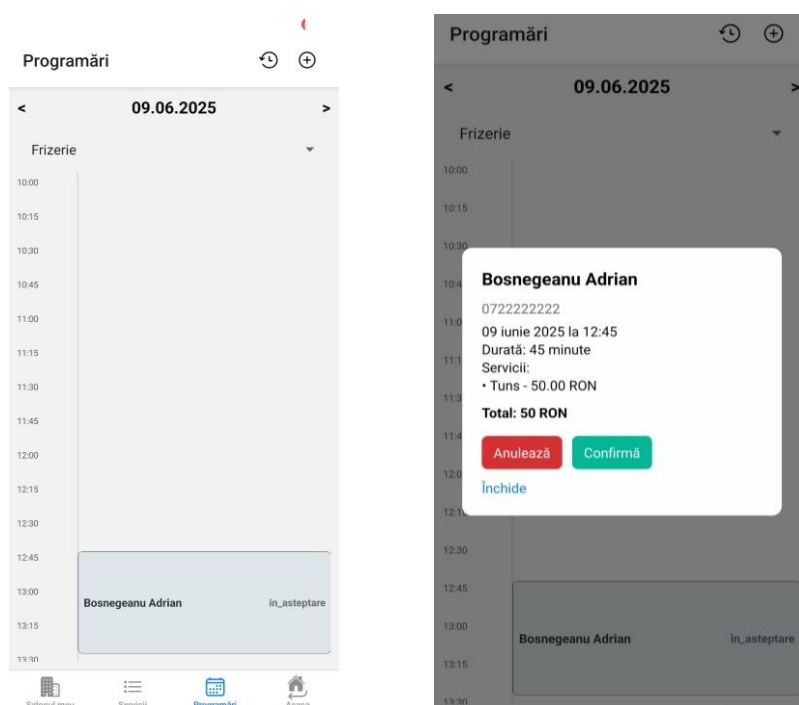


Fig. 5.5. Programatorul digital și fereastra modală a programării

### 5.3. Anularea unei programări

În cazul partenerului, detaliile unei programări sunt afișate în urma accesării modulului Programări din navigatorul de gestionare al salonului și apăsării pe caseta aferentă programării, conform figurii atașată anterior (Fig. 5.3). Apăsarea butonului roșu „Anulează” schimbă starea programării și notifică clientul prin email și notificare push în această privință.

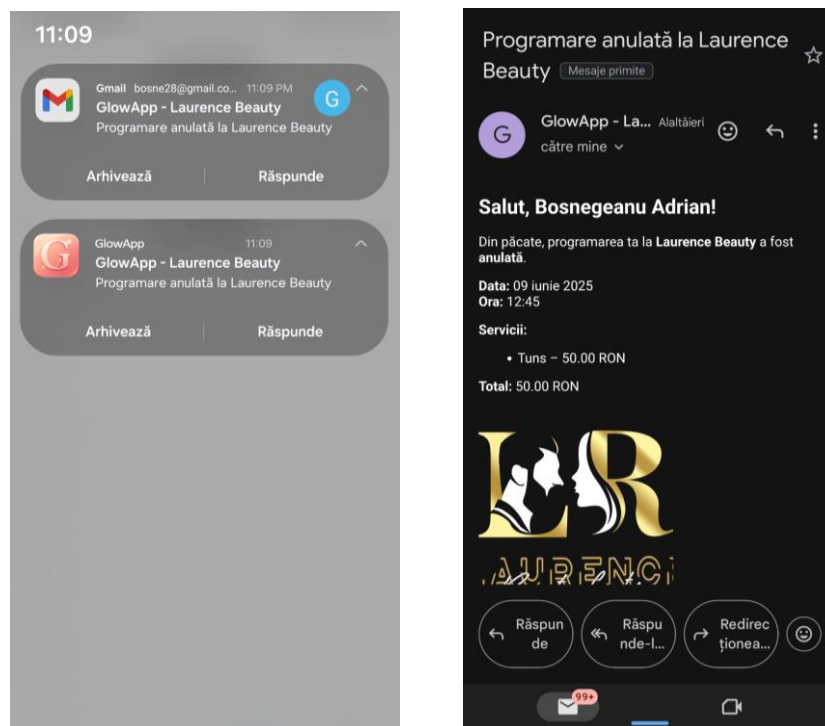


Fig. 5.6. Notificarea și mailul către client la anularea programării

În cazul clientului, o programare efectuată apare în modulul Programări aferent navigatorului principal. Aceasta poate fi anulată prin apăsarea butonului roșu „Anulează”, vizibil în cazul în care aceasta se află în stadiul „Confirmată” sau „În așteptare”.

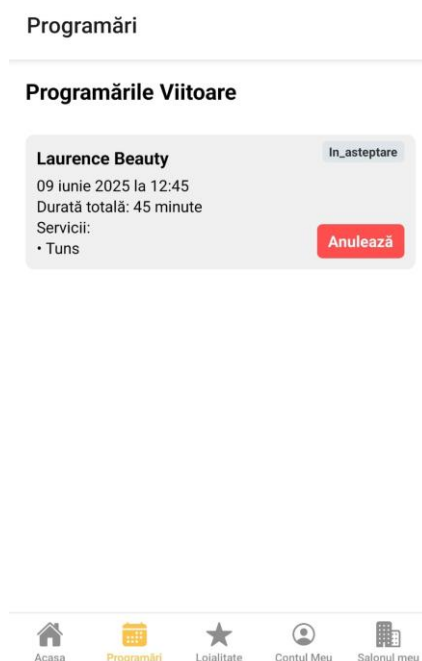


Fig. 5.7. Modulul Programări destinat clientului

Anularea unei programări de către client conduce la notificarea salonului, push și prin email, referitor la această modificare.

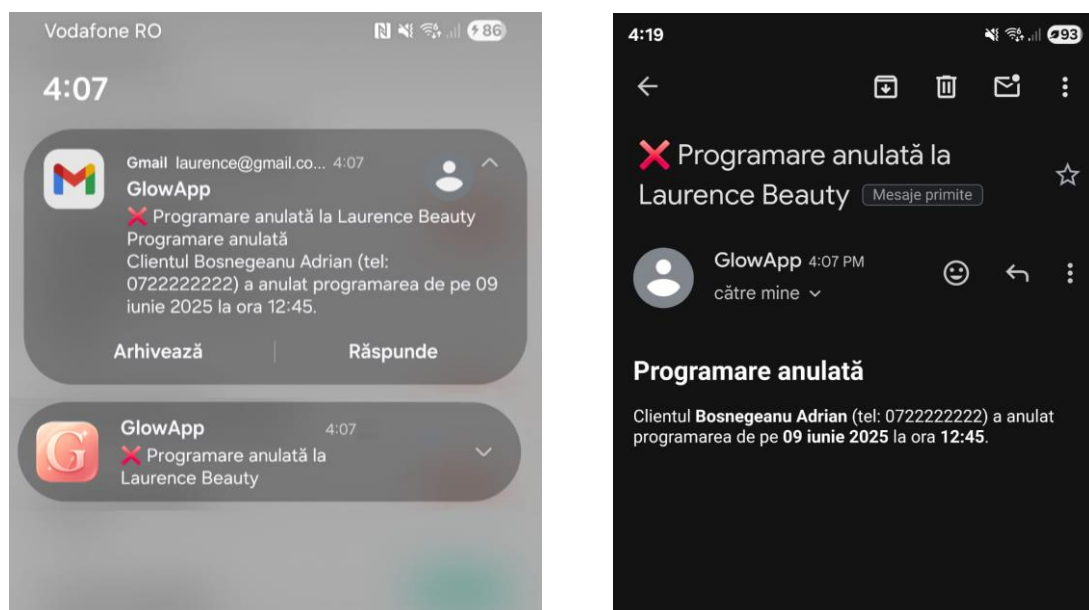


Fig. 5.8. Notificarea partenerului la anularea programării de către client

#### 5.4. Acordarea unei recenzii

După ce o programare este finalizată, dacă clientul nu a acordat în trecut o recenzie salonului de înfrumusețare, acesta este notificat prin email. În cadrul mailului este prezent un link către profilul de Google Maps al salonului, unde poate acorda o notă și își poate relata experiența.

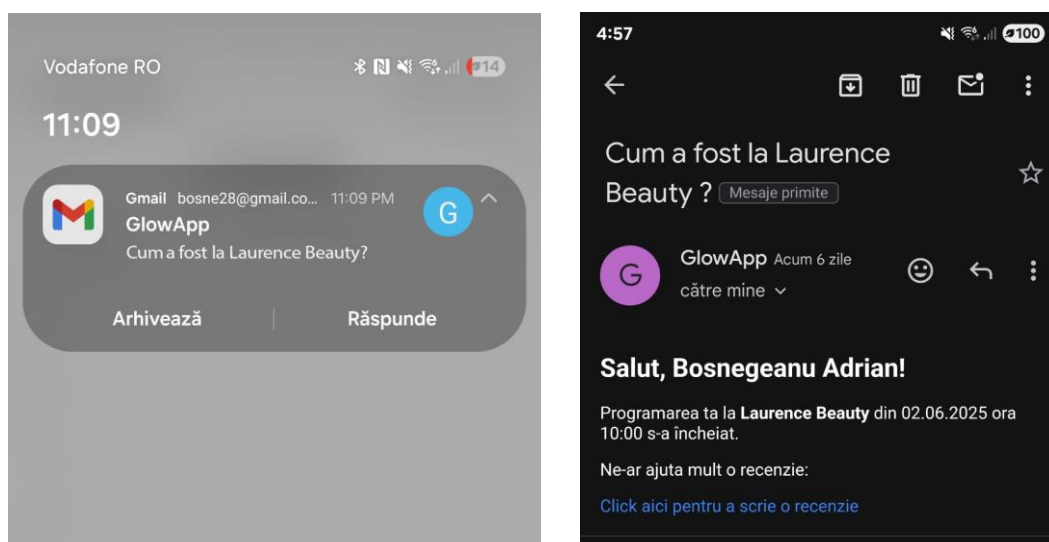


Fig. 5.9. Notificarea clientului spre a redacta o recenzie

Notificarea are loc la momentul în care se depășește durata totală pentru o programare confirmată. Un link de recenzionare este totodată disponibil în modulul „Istoric client”, accesibil în tab-ul Contul Meu al navigatorului principal.

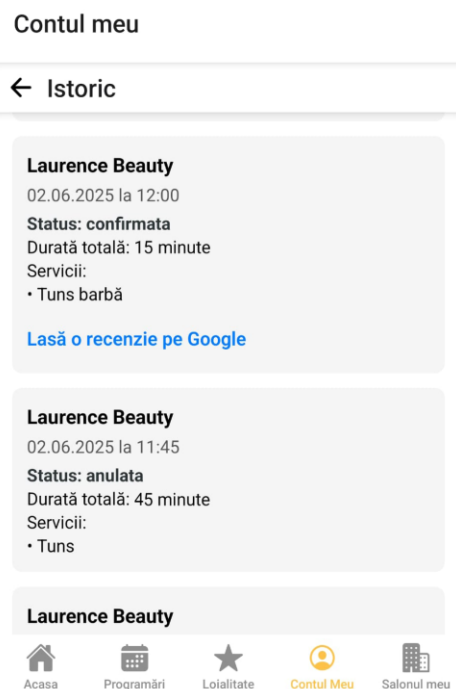


Fig. 5.10. Modulul „Istoric client”

## 5.5. Administrarea unui salon de înfrumusețare

Înscrierea unui salon de înfrumusețare și extinderea utilizatorului la rolul de partener se realizează prin butonul „Vreau să fiu partener!” din modulul Contul Meu al navigatorului principal. Astfel, se accesează un formular cu informațiile preliminare necesare înrolării.

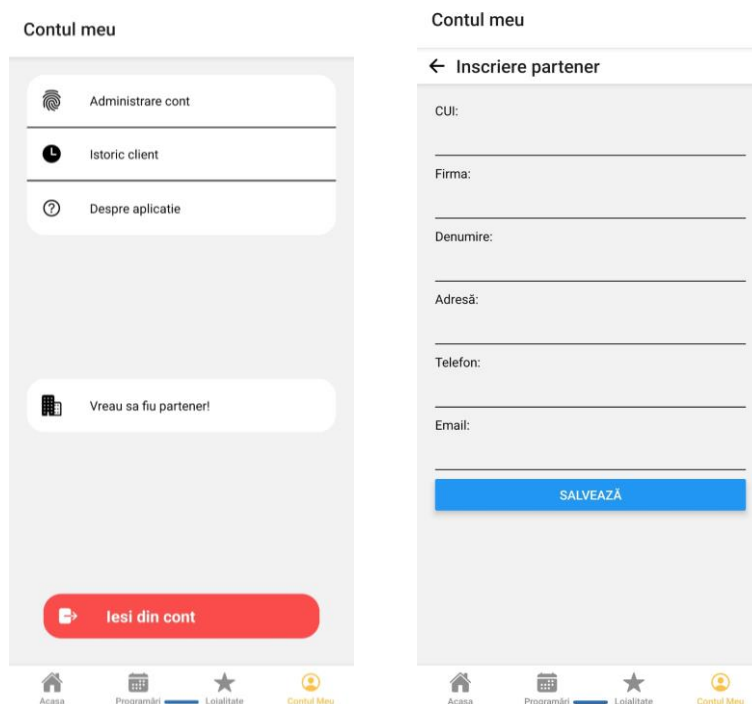


Fig. 5.11. Modulul „Contul Meu” și formularul de înrolare

După completare, este afișată o alertă de mulțumire, iar utilizatorul este redirecționat către modulul Salonul Meu pentru a completa înscrierea.

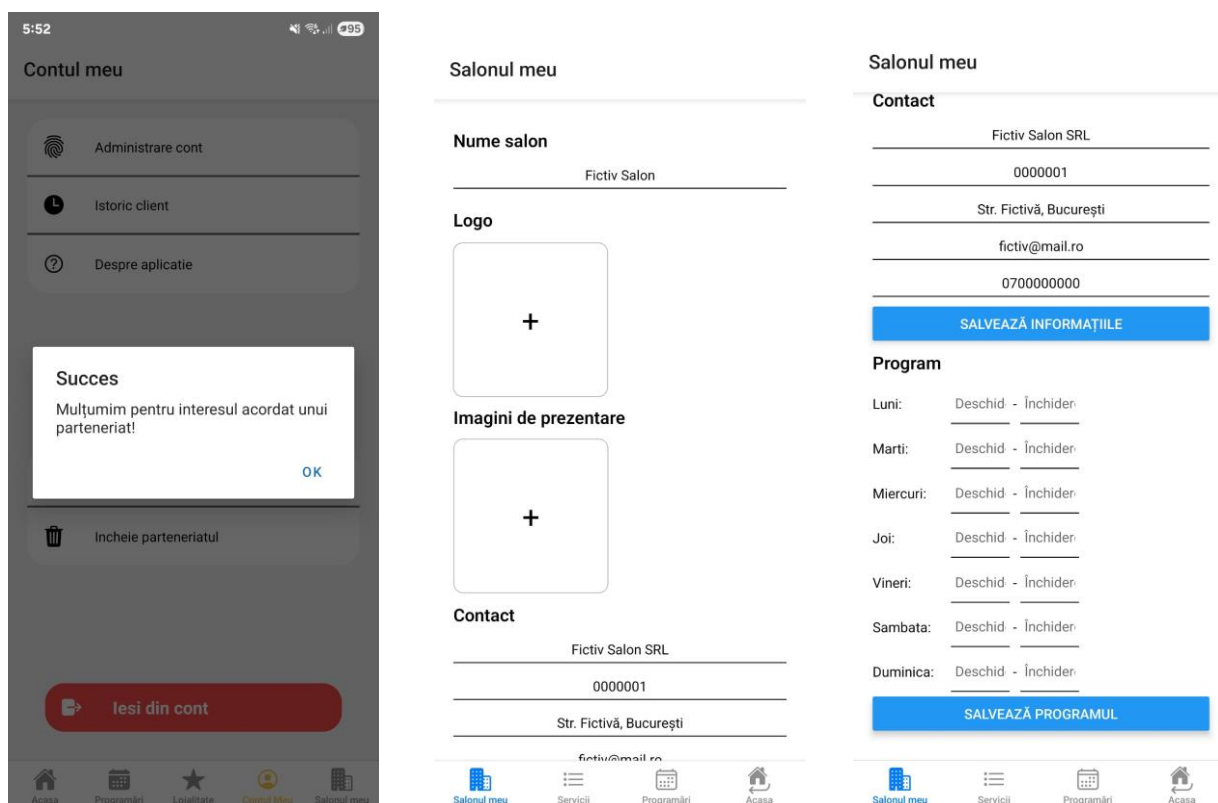


Fig. 5.12. Alerta de mulțumire și modulul „Salonul Meu”

Materialele fotografice (logo sau imagini de prezentare) se adaugă prin apăsarea pe casetele marcate cu „+”. Pozele pot fi eliminate prin apăsarea lor ulterioară. Informațiile și programul se pot modifica prin introducerea lor în câmpurile aferente și apăsarea pe butonul de salvare corespunzător.

## 5.6. Loialitate și vouchere

Saloanele de înfrumusețare pot acorda benevol vouchere către clienți prin accesarea istoricului programărilor din modulul „Programări” aferent partenerilor. Apăsarea unei programări din istoric afișează gradul de loialitate general al unui utilizator și punctele obținute în cadrul salonului în cauză.

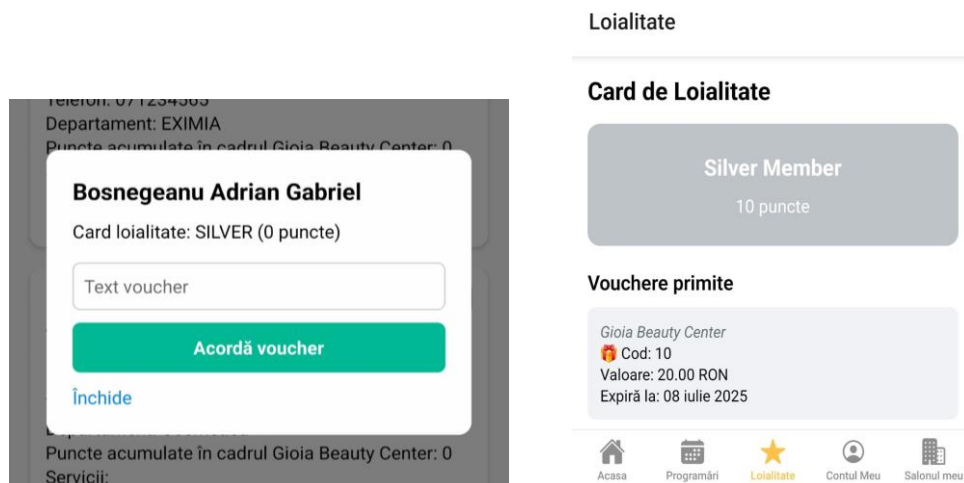


Fig. 5.13. Acordarea și recepționarea unui voucher



## CONCLUZII SI CONTRIBUTII PERSONALE

### Concluzii

Platforma informatică a fost proiectată pentru gestionarea facilă a saloanelor de înfrumusețare în paralel cu satisfacerea nevoilor consumatorilor acestui domeniu. Utilizarea aplicației prezintă beneficii pentru ambii actori.

Saloanele de înfrumusețare capătă vizibilitate sporită în mediul social și se adaptează la standardele moderne de digitalizare. Prin modul de funcționare al aplicației, programatorul clasic greu de gestionat este automatizat promovând maximizarea numărului de clienți printr-un sistem de calcul al intervalelor libere obiectiv și ferit de intervenția greșelilor umane.

Urmând tendințele curente de simplificare, consumatorii nu mai sunt nevoiți să își caute salonul potrivit pentru ei prin mijloace ineficiente, ci pot găsi toate detaliile de care au nevoie, actualizate, pe o singură aplicație. Serviciile pe care salonul le oferă și durata acestora de desfășurare sunt centralizate alături de programul de lucru și orele disponibile pentru programare, care se poate face direct din aplicație. Pozele încărcate de parteneri, alături de locația și recenziile preluate automat prin Google Places API realizează un profil complet pentru salon și întăresc încrederea clienților.

În cadrul dezvoltării, s-a beneficiat de parteneriatul celor două locații vizibile și în figurile prezente în document, Laurence Beauty Center și Gioia Beauty Studio. Alături de echipele lor, au fost analizate nevoile principale pe care un sistem de booking digital ar trebui să le atingă în domeniul înfrumusețării.

Platforma vizează în special sistemele de operare mobile, Android și iOS, iar soluțiile oferite de Expo și React-Native au permis dezvoltarea paralelă pentru ambele tipuri de dispozitive propuse. Totodată, aplicația rezultată permite și utilizarea din browsere web.

Așadar, această platformă informatică poate modifica modul în care clienții comunică cu saloanele de înfrumusețare, cât și modul în care saloanele se fac cunoscute printre potențialii clienți. Este un exemplu concret prin care tehnologia poate facilita accesul la servicii, poate eficientiza promovarea și poate crea un mediu mai organizat și mai adaptat la nevoile utilizatorilor.

### Contributii personale

Dezvoltarea propriu-zisă a implicat o perspectivă de programare full-stack. Fiecare pas a implicat acumularea de cunoștințe noi și aprofundarea celor dobândite pe perioada de școlarizare.

În backend, pas cu pas a fost configurat un server virtual Linux, inițial gol, a fost asociat unui domeniu și a fost implementată tehnologia de rutare a traficului oferită de NGINX. Ulterior prin Node.js și MySQL s-a stabilit întreaga logică de backend a platformei, proces care a implicat dependențe de simplificare a codării (pachetul Express), de gestionare a fișierelor (multer), pachetul node-cron pentru sarcini cronometrate aferente notificării push și prin email prin API-urile oferite de Expo și Resend.



În frontend, implementarea aplicației prin elemente native a implicat un design orientat mai degrabă spre simplitate și accesibilitate. Dezvoltarea paralelă limitează opțiunile de design, dar în ceea ce privește funcționalitatea, nu au fost întâmpinate dificultăți majore.

Deși există soluții pe piață orientate spre același subiect cu platforma informatică dezvoltată, fiecare prezintă lacune. În cadrul acestui proiect s-a propus înglobarea tuturor funcționalităților esențiale și includerea unor servicii externe deja consacrate, precum recenzionarea și serviciile de localizare prin Google Places. Un element de originalitate, pe care platformele de booking din domeniul înfrumusețării nu îl abordează este sistemul de loialitate și vouchere. Acest element este o oportunitate de echilibrare a pieței prin care saloanele de înfrumusețare pot fideliza clienți noi pe baza istoricului lor în platformă.

### **Perspective de viitor**

Viitorul platformei prevede separarea dezvoltării frontend pentru optimizarea și perfecționarea vizuală în fiecare sistem de operare. Deși Expo împreună cu React-Native sunt o soluție perfectă pentru a pune bazele facil unei aplicații unificate de aceeași bază de cod, pentru atragerea utilizatorilor este nevoie de o atenție la detalii sporită. Totodată, aplicația web rezultată nu are structura cu care utilizatorii sunt familiarizați și poate necesita adaptări separate de aplicația mobilă. Navigarea prin tab-uri și necesitatea conectării ca prim pas nu sunt practici populare în mediul web și pot provoca reticență utilizatorilor ca primă impresie.

Sistemul de loialitate și acordare de vouchere poate fi extins prin acordarea acestora automată, la praguri definite de parteneri. Fiind un concept relativ nou în ceea ce privește platformele de booking pentru saloanele de înfrumusețare, acest sistem este la o fază incipientă și poate fi îmbunătățit în viitor și pe bază de feedback din partea utilizatorilor aplicației.

Lansarea publică a unui astfel de utilitar poate avea și implicații legale, necesitând înființarea unei societăți pentru a proteja legal toți actorii din această platformă și oficializarea parteneriatelor.

## BIBLIOGRAFIE

- [1] Richard Taylor, „Scientific Memoirs: Selected From The Transactions Of Foreign Academies Of Science And Learned Societies, And From Foreign Journals”, ISBN-10 : 1016131712, Legare Street Press, 2022
- [2] <https://www.microsoft.com/ro-ro/microsoft-365/business/scheduling-and-booking-app>
- [3] <https://medium.com/@shrestha.matina.20/what-is-the-3-tier-architecture-4520522e0720>
- [4] Miao Zheng, Teng Li, Hongqian Wang & Hongling Zhong, „Impact of digital self-scheduling on operations management and patient experience in hospital outpatient settings: a systematic review and meta-analysis”, DOI: 10.21203/rs.3.rs-4243854/v1, 2024.
- [5] Shuvo Kumar Mallik, Irman Uddin, Farzana Akter, Shafin Rahman, „Evaluating the influence of customer reviews and consumer trust on online purchase behavior”, World Journal of Advanced Research and Reviews, DOI: 10.30574/wjarr.2025.25.1.0015, 2025
- [6] Luca Dedola, Michael Ehrmann, Peter Hoffmann, Ana Lamo, Gonzalo Paz Pardo, Jiri Slacalek, Georg Strasser, „Digitalisation and the economy”, European Central Bank, DOI: 10.2866/93858, 2023
- [7] [https://ro.wikipedia.org/wiki/Microsoft\\_Windows](https://ro.wikipedia.org/wiki/Microsoft_Windows)
- [8] Alessandro Del Sole, „Visual Studio Code Distilled”, ISBN-13: 9781484242230, 2019
- [9] David Flanagan, „JavaScript: The Definitive Guide”, O'Reilly Media, ISBN-13: 9780596101992, 2006
- [10] Bonnie Eisenman, „Learning React Native: Building Native Mobile Apps with JavaScript”, ISBN-13: 9781491989142, 2017
- [11] Dan Ward, „Understanding Expo and React Native”, React Native Cookbook, ISBN-13: 9781788991926, 2019
- [12] Raphaël Hertzog, Roland Mas, „The Debian Administrator's Handbook”, Freexian SARL, ISBN: 979-10-91414-19-7, 2020
- [13] <https://support.apple.com/ro-ro/guide/shortcuts-mac/apd2e30c9d45/>
- [14] George-Alex Stelea, Florin Sandu, Livia Sângeorzan, „Programarea în Internet, Mediu Distribuit și Cloud”, Editura Universității Transilvania din Brașov, ISBN: 978-606-19-0949-0, 2017
- [15] [https://nodejs.org/en/learn/getting-started/introduction-to-nodejs?utm\\_source=chatgpt.com](https://nodejs.org/en/learn/getting-started/introduction-to-nodejs?utm_source=chatgpt.com)
- [16] <https://developers.google.com/maps/documentation/places/web-service/overview>
- [17] <https://resend.com/docs/api-reference/introduction>
- [18] Note de curs „Baze de date”, Universitatea Tehnică de Construcții București, 2024
- [19] Vikram Vaswani, „MySQL: Utilizarea și administrarea bazelor de date MySQL”, ISBN: 978-973-7881-62-5, Editura Rosetti, 2010
- [20] <https://tutorialehtml.com/ro/>
- [21] Universitatea Tehnică de Construcții din București, „HTML pe înțelesul tuturor”, <https://www.studocu.com/ro/document/universitatea-tehnica-de-construcții-din-bucurești/law-contracts/html-manual-in-limba-romana/33381584>, 2020.
- [22] Laborator 4 – CSS „Tehnologii Web”, Universitatea Tehnică de Construcții București, 2024