



Universitatea
Transilvania
din Brașov
FACULTATEA DE MATEMATICĂ
ȘI INFORMATICĂ

Program de studiu:

Informatică

Lucrare de licență

**Platformă web full-stack pentru managementul
evenimentelor cu sistem integrat de bilete digitale și
funcționalități în timp real**

Autor: Ghiță Valentin-Gabriel

Coordonator științific: Prof. Dr. Deaconu Adrian-Marius

Brașov – 2025

Contents

1	Introducere.....	5
1.1	Motivația alegerii temei.....	5
1.2	Descrierea aplicației EventHub	5
1.3	Comparație cu platforme similare	7
1.4	Structura lucrării	7
2	Fundamente teoretice ale managementului evenimentelor	9
2.1	Evoluția industriei evenimentelor.....	9
2.2	Provocările actuale în managementul evenimentelor	10
2.3	Analiza soluțiilor existente	12
2.3.1	Eventbrite – platformă de organizare și ticketing.....	12
2.3.2	Ticketmaster – platformă globală de ticketing	14
2.4	De ce este necesară o platformă integrată	15
3	Tehnologii și arhitectura sistemului EventHub	16
3.1	Arhitectura aplicațiilor web moderne (Client-Server, REST API, SPA)	16
3.2	Tehnologii backend.....	17
3.2.1	Node.js și ecosistemul JavaScript	17
3.2.2	Framework-ul Express.js.....	18
3.2.3	Baze de date relaționale – PostgreSQL.....	18
3.2.4	Autentificare și autorizare (JWT, OAuth)	18
3.3	Tehnologii frontend	19
3.3.1	React.js și ecosistemul său	19
3.3.2	Managementul stării aplicației	20
3.3.3	Tailwind CSS pentru stilizare	23
3.4	Comunicarea în timp real (WebSocket, Socket.io)	24

3.5 Servicii externe și integrări	24
3.6 Testare și asigurarea calității (Jest, metodologii de testare)	27
3.6.1 Configurarea Jest pentru testare	27
3.6.2 Testarea modelelor	28
3.6.3 Testarea controller-elor	29
3.6.4 Testarea middleware-ului	31
3.6.5 Testarea serviciilor	32
4 Proiectarea și implementarea sistemului EventHub	34
4.1 Analiza cerințelor și specificațiile funcționale	34
4.1.1 Identificarea cerințelor funcționale	34
4.1.2 Cerințele non-funcționale	35
4.1.3 Specificațiile tehnice	36
4.1.3 Analiza riscurilor și mitigarea acestora	36
4.2 Modelarea bazei de date	37
4.2.1 Diagramele entitate-relație	37
4.2.2 Structura tabelor principale	38
4.2.3 Normalizarea și încadrarea în forma normală a treia	39
4.2.4 Indexarea și optimizarea performanței	40
4.2.5 Constrangeri și integritatea datelor	40
4.3 Arhitectura aplicației	41
4.3.1 Modelul MVC adaptat	41
4.3.2 Organizarea codului în servicii și controllere	41
4.3.3 Middleware și interceptori	42
4.4 Implementarea securității aplicației	43
4.4.1 Securitatea autentificării	43

4.4.2 Protecția împotriva vulnerabilităților comune	44
4.5 Gestionarea sesiunilor și autentificării.....	44
4.5.1 Arhitectura sistemului de autentificare	44
4.5.2 OAuth integration	45
4.5.3 Gestionarea permisiunilor și rolurilor.....	45
4.5.4 Recuperarea parolelor și securitatea conturilor	46
4.6 Integrarea serviciilor externe	46
4.6.1 Serviciul de email	46
4.6.2 Serviciile de geolocation	47
4.6.3 Serviciile de notificări	47
5 Prezentarea funcționalităților aplicației EventHub	48
5.1 Interfața utilizatorului (screenshots + UX).....	48
5.1.1 Principiile de design adoptate.....	48
5.1.2 Componenta Header și navigarea principală	48
5.1.3 Componenta Footer și structura informațională.....	49
5.1.4 Pagina principală și experiența utilizatorului	50
5.1.5 Sistemul de căutare avansată.....	51
5.2 Managementul evenimentelor.....	52
5.2.1 Arhitectura sistemului de management	52
5.2.2 Interfața de administrare evenimente	52
5.2.3 Formularul de creare și editare evenimente	53
5.2.4 Funcționalități avansate de management	54
5.2.5 Afișarea publică a evenimentelor	55
5.2.6 Integrarea cu sistemul de bilete	56
5.3 Sistemul de ticketing și rezervări.....	57

5.3.1 Arhitectura sistemului de bilete	57
5.3.2 Componenta de selecție bilete.....	57
5.3.3 Procesul de rezervare și achiziție	58
5.3.4 Gestionarea biletelor utilizatorilor.....	61
5.3.5 Sistemul de schimb și rambursare bilete.....	61
5.3.6 Validarea și check-in-ul biletelor	62
5.4 Sistemul de plăți.....	63
5.4.1 Arhitectura sistemului de plăți	63
5.4.2 Gestionarea metodelor de plată	63
5.4.3 Sistemul intern de credite	64
5.5 Dashboard-ul administrativ	65
5.6 Funcționalități în timp real	68
6 Concluzii și perspective de dezvoltare	71
6.1 Rezumatul lucrării și obiectivele atinse.....	71
6.2 Contribuții și inovații	72
6.3 Limitări ale soluției actuale	73
6.4 Perspective de dezvoltare viitoare	74
6.5 Concluzii finale.....	75
Bibliografie:.....	75

1 Introducere

1.1 Motivația alegerii temei

Industria evenimentelor s-a dezvoltat semnificativ în ultimii ani, devenind o componentă esențială în sectorul cultural, economic și social al societății contemporane. Această evoluție a fost influențată major de progresele tehnologice și de schimbările comportamentale ale publicului, care solicită metode mai eficiente și accesibile în organizarea și promovarea evenimentelor [5]. În acest context, tehnologiile digitale și platformele online au devenit indispensabile, atât pentru organizarea și promovarea evenimentelor, cât și pentru vânzarea biletelor și gestionarea participanților.

Adoptarea tehnologiilor digitale în domeniul evenimentelor aduce numeroase beneficii demonstrate prin studii și statistici. Spre exemplu, platformele digitale dedicate evenimentelor și ticketing-ului permit o creștere semnificativă a eficienței operaționale, cu reducerea timpului necesar organizării și promovării cu până la 30% [8], în paralel cu diminuarea costurilor administrative. În același timp, majoritatea achizițiilor de bilete se realizează deja exclusiv online, acest mod de tranzacționare depășind pragul de 70% în ultimii ani [8]. Platformele specializate asigură, de asemenea, o mai bună comunicare între organizatori și participanți, oferind posibilitatea transmiterii rapide a informațiilor actualizate și a modificărilor intervenite în organizarea evenimentului.

Astfel, dezvoltarea unei platforme web moderne pentru managementul evenimentelor și vânzarea online a biletelor, cum este aplicația EventHub, devine un demers justificat și necesar pentru adaptarea industriei la cerințele actuale și la standardele digitale emergente. Obiectivul acestei lucrări este prezentarea unei astfel de soluții care contribuie la eficientizarea proceselor organizatorice și la îmbunătățirea experienței generale a participanților și a organizatorilor de evenimente.

1.2 Descrierea aplicației EventHub

EventHub este o aplicație web destinată facilitării organizării și gestionării evenimentelor, permițând totodată utilizatorilor să descopere evenimente și să achiziționeze bilete online. Platforma se adresează **două categorii principale de utilizatori**: organizatorii de evenimente (ex. companii, instituții, organizații sau persoane care doresc să își promoveze și administreze propriile evenimente) și participanții/cumpărătorii de bilete (publicul interesat de evenimente). Prin intermediul EventHub, **organizatorii** pot crea și configura pagini dedicate evenimentelor, unde furnizează detalii precum descrierea evenimentului, data și

locația, numărul de locuri disponibile și prețul biletelor. Aplicația le oferă instrumente pentru a gestiona vânzarea biletelor – de la stabilirea categoriilor de bilete (de exemplu, *General Access*, VIP, etc.) și a prețurilor, până la procesarea plăților în mod securizat și urmărirea în timp real a numărului de bilete vândute. De asemenea, organizatorii pot accesa listele de participanți înregistrați și pot trimite notificări sau informații actualizate acestora (de exemplu, modificarea orei sau a datei unui eveniment, cat si anulara acestuia).

Pentru utilizatorii finali (publicul participant), EventHub funcționează ca un portal centralizat de evenimente. Aceștia pot răsfoi evenimentele disponibile într-un catalog structurat pe categorii (concerte, evenimente sportive, festivaluri etc.), având la dispoziție filtre de căutare (după dată, locație si tip de eveniment) pentru a găsi rapid evenimentele de interes. Pagina fiecărui eveniment oferă informațiile necesare (descriere, program, localizare) și opțiunea de achiziție a biletelor. Cumpărarea biletului se realizează online, utilizatorul primind confirmarea și biletul electronic (cu cod QR), atat pe email cat si in contul acestuia, eliminând nevoia imprimării biletelor fizice. La sosirea la eveniment, validarea accesului se poate face tot prin intermediul platformei de catre administratori, din panoul lor dedicat.

Problemele pe care EventHub le abordează țin de eficientizarea și îmbunătățirea experienței atât pentru organizatori, cât și pentru participanți. Pentru organizatori, aplicația reduce efortul logistic și riscul de erori: în locul gestionării manuale a listelor de participanți sau al vânzării fizice a biletelor (ce pot duce la suprasolicitare sau inconsistențe), EventHub automatizează aceste procese și oferă o evidență clară, în timp real, asupra tuturor utilizatorilor, cât și a biletelor vândute. Totodată, platforma extinde capacitatea de promovare a evenimentelor deoarece orice utilizator online poate descoperi evenimentul prin intermediul platformei, sporind vizibilitatea și adresabilitatea evenimentului dincolo de cercur restrâns al promovării tradiționale. Pentru participanți, EventHub rezolvă problema convenienței: achiziția de bilete se face rapid, de oriunde, eliminând cozile sau deplasarea la un punct de vânzare. De asemenea, participanții beneficiază de siguranță și transparență (bilete verificate, tranzacții securizate) și de acces la informații actualizate (dacă apar modificări în organizarea evenimentului, acestea pot fi comunicate instant prin platformă, cat si pe email). Nu în ultimul rând, EventHub contribuie la reducerea costurilor generale prin eliminarea tipăririi biletelor și a materialelor promoționale fizice, aliniindu-se totodată cu tendințele de digitalizare și sustenabilitate (biletele electronice fiind o alternativă ecologică la cele tipărite).

1.3 Comparație cu platforme similare

Pentru a poziționa mai clar aplicația EventHub în contextul actual al pieței, este utilă o comparație cu câteva platforme populare de ticketing: Ticketmaster, Eventbrite și iabilet.ro.

Ticketmaster este lider global în domeniul ticketingului pentru evenimente mari și medii, dominând piața datorită infrastructurii robuste, capabilă să gestioneze volume extrem de mari de tranzacții simultane, și oferind mecanisme puternice de prevenire a fraudei. Cu toate acestea, platforma este adesea criticată pentru costurile ridicate și lipsa flexibilității pentru evenimente mici sau personalizate.

Eventbrite, pe de altă parte, se adresează în mod preponderent evenimentelor mici și medii, oferind o interfață prietenoasă și flexibilă, alături de instrumente intuitive pentru personalizarea paginilor evenimentelor. Această platformă este apreciată pentru transparența costurilor și simplitatea utilizării, dar și aici costurile pot deveni o povară pentru organizatorii cu bugete reduse, din cauza comisioanelor percepute per bilet.

În România, iabilet.ro este una dintre cele mai populare platforme locale de ticketing, având avantajul unei rețele puternice de promovare regională și opțiuni adaptate pieței locale, cum ar fi plata în rate sau integrarea cu sisteme locale de plată. Totuși, aceasta tinde să fie folosită predominant pentru evenimente muzicale și culturale mai mari și medii, fiind mai puțin accesibilă sau optimizată pentru evenimente mai mici, de tip workshop sau comunitare.

În acest context, EventHub a fost creat pentru a acoperi un segment insuficient deservit de giganții industriei, acesta fiind evenimentele mici și medii, organizate local sau în comunități specifice. Obiectivul platformei este flexibilitatea și accesibilitatea maximă: organizatorii au control deplin asupra personalizării paginilor evenimentelor (branding propriu, descrieri detaliate și imagini), fără restricții impuse de un șablon rigid. Structura costurilor este, de asemenea, concepută pentru a fi cât mai favorabilă organizatorilor, minimizând comisioanele per bilet sau oferind modele de taxare reduse sau fixe.

1.4 Structura lucrării

Lucrarea este împărțită în șase capitole, fiecare tratând o etapă esențială a proiectului EventHub:

Introducere – deschide demersul prin prezentarea contextului actual al industriei de evenimente și subliniază motivele alegerii temei. Sunt evidențiate, pe scurt, funcționalitățile cheie ale platformei EventHub și avantajele sale față de alte platforme similare, precum

Ticketmaster. Capitolul se încheie cu formularea structurii lucrării, orientând cititorul către conținutul fiecărui capitol următor.

Fundamente teoretice ale managementului evenimentelor – clarifică noțiunile de bază: definiții, tipologii, etapele organizării și principiile de succes în planificare. În plus, capitolul evidențiază impactul digitalizării, abordând tendințe precum formatele hibride, realitatea augmentată și inteligența artificială, și explică modul în care acestea influențează organizatorii și participanții deopotrivă.

Tehnologii și arhitectura sistemului EventHub – detaliază arhitectura web modernă (client-server, REST API, SPA) și justifică stack-ul ales: Node.js + Express pe backend, PostgreSQL pentru persistență, React + Tailwind pe frontend și Socket.IO pentru comunicații în timp real. Include subsecțiuni despre testare (Jest & Supertest) și servicii auxiliare, oferind comparații cu alternativele disponibile.

Proiectarea și implementarea sistemului EventHub – ratează analiza cerințelor, modelarea bazei de date (ERD, tabele, 3-NF, indexare), arhitectura aplicației (MVC extins), mecanismele de securitate (JWT, OAuth) și integrarea serviciilor externe. Conține fragmente de cod relevante și descrie fluxurile importante ale aplicației.

Prezentarea funcționalităților aplicației EventHub – ilustrează prin capturi de ecran și scenarii de utilizare modul în care platforma gestionează evenimentele, biletele, plățile și dashboard-ul administrativ, punând accent pe experiența utilizatorului și pe funcționalitățile în timp real.

Concluzii și perspective de dezvoltare – rezumă realizările principale (platformă integrată, arhitectură modulară, securitate și comunicații în timp real), scoate în evidență inovațiile aduse (chat live nativ, rambursări automate, geocoding), admite limitele actuale (raportare avansată, aplicație mobilă nativă) și trasează direcții clare de extindere: AI pentru recomandări, aplicații iOS/Android, marketplace de servicii conexe și suport pentru evenimente hibride/live-stream.

2 Fundamente teoretice ale managementului evenimentelor

2.1 Evoluția industriei evenimentelor

Industria evenimentelor a cunoscut transformări majore în ultimele decenii, devenind un domeniu din ce în ce mai complex, digitalizat și profesionist. De la întruniri organizate ad-hoc în trecut, s-a ajuns la un sector global evaluat la peste \$1.1 trilioane în 2018, cu proiecții de creștere până la \$2.33 trilioane în 2026 [10]. Această creștere impresionantă reflectă importanța economică a evenimentelor și expansiunea lor la scară internațională. Totodată, evenimentele s-au profesionalizat: dacă în urmă cu câteva decenii organizarea era adesea informală, astăzi există manageri de eveniment certificați, agenții specializate și standarde profesionale. Au apărut roluri dedicate precum Customer Experience Manager, Meeting Designer sau Event Tech Expert, ceea ce ilustrează trecerea de la abordări empirice la un management bazat pe competențe specifice [9]. Succesul unui eveniment nu mai este apreciat doar intuitiv, ci se măsoară riguros prin indicatori de performanță, de la numărul de participanți și gradul lor de satisfacție, până la impactul pe rețelele sociale și ROI-ul generat [9].

Un factor cheie al evoluției industriei evenimentelor îl reprezintă digitalizarea. În anii 1990, odată cu răspândirea internetului, planificarea și promovarea evenimentelor au devenit mult mai eficiente [9]. Comunicarea cu participanții prin email și crearea de pagini web pentru evenimente au extins considerabil audiența și viteza de răspândire a informațiilor. Totuși, tranziția completă către procese digitale a fost treptată; până la mijlocul anilor 2000 încă erau folosite metode tradiționale (de exemplu, formulare de înscriere trimise prin poștă sau fax) în paralel cu noile instrumente online [9]. Ulterior, avansul laptop-urilor, al smartphone-urilor și apariția sistemelor bazate pe cloud au revoluționat planificarea: documentația tipărită (agende, liste, contracte) a fost înlocuită cu platforme colaborative, permițând echipelor să actualizeze în timp real programe și liste de sarcini și să comunice instantaneu de oriunde [9]. În plus, rețelele de socializare au schimbat radical modul de marketing al evenimentelor deoarece încă din anii 2000, platforme precum Facebook, Twitter sau Instagram au permis generarea de buzz și implicarea directă a publicului. Utilizarea hashtag-urilor personalizate (începând cu ~2007) a facilitat agregarea discuțiilor și chiar afișarea lor în timp real în cadrul evenimentelor, sporind interacțiunea participanților [9].

Un moment de cotitură în evoluția recentă a fost pandemia COVID-19, care a forțat industria să se adapteze fără precedent. În 2020, evenimentele fizice au fost suspendate sau mutate online, declanșând o transformare profundă bazată pe tehnologie [10]. S-a accelerat adoptarea modelelor hibride (combinație de componentă fizică și virtuală) și a platformelor virtuale dedicate conferințelor și meeting-urilor online. Ceea ce inițial părea o soluție provizorie s-a dovedit a fi un salt permanent în modul de planificare și execuție a evenimentelor, introducând concepte precum audiențe distribuite global, transmisii live interactive și experiențe personalizate pentru participanți în funcție de profilul și preferințele lor. Adaptabilitatea industriei în fața pandemiei a catalizat totodată inovații tehnologice (ex: evenimente cu realitate augmentată, platforme de networking virtual, aplicații de engagement în timp real) și a evidențiat necesitatea ca profesioniștii din domeniu să fie flexibili și orientați spre noile tendințe digitale. Astfel, în prezent, managementul evenimentelor se desfășoară într-un ecosistem high-tech, în care software-ul specializat joacă un rol central. De la sisteme de management al evenimentelor (Event Management Software) folosite pentru a gestiona bugete, înregistrări și logistică [10], până la instrumente de analiză a datelor despre participanți, tehnologia este integrată în aproape toate etapele ciclului de viață al unui eveniment. Acest parcurs istoric al industriei – de la adunări modeste la evenimente globale high-tech – subliniază importanța adaptării continue și a inovării în managementul evenimentelor.

2.2 Provocările actuale în managementul evenimentelor

În ciuda progreselor, organizatorii de evenimente se confruntă astăzi cu o serie de provocări complexe, generate în mare parte de creșterea așteptărilor și de ritmul evoluției tehnologice. Printre cele mai presante probleme se numără fragmentarea proceselor, lipsa de automatizare integrată, așteptările ridicate ale publicului și costurile și riscurile crescute asociate organizării.

Fragmentarea proceselor se referă la faptul că, deși există multe instrumente software specializate, ele funcționează adesea în silozuri, fără a fi complet interoperabile. Un organizator tipic ajunge să folosească în medie șase platforme diferite (pe lângă nenumărate fișiere Excel) pentru a gestiona un singur eveniment complex [12]. De exemplu, se pot utiliza aplicații separate pentru vânzarea biletelor, gestionarea agendei, trimiterea de email-uri către participanți, sondaje post-eveniment, streaming live etc. Această fragmentare duce la dublarea eforturilor, la introducerea manuală a datelor în sisteme multiple și la inconsistențe sau erori de sincronizare. Studiile arată că migrarea și reconcilerea datelor între mai multe unelte cresc probabilitatea de erori și chiar riscul de breșe de securitate a datelor personale

[12]. Practic, folosirea a numeroase instrumente disparate complică fluxul de lucru și sporește complexitatea operațională, organizatorii fiind nevoiți să aloce timp semnificativ pentru a conecta informațiile dintr-un sistem în altul. Un raport recent notează că această incompatibilitate a aplicațiilor utilizate "duce la costuri ridicate de licențiere și mentenanță, precum și la pierderi financiare adiționale și nemulțumiri ale clienților din cauza erorilor de migrare a datelor" [12]. Cu alte cuvinte, fragmentarea tehnologică introduce costuri ascunse și vulnerabilități în procesul de organizare.

Conectată strâns de fragmentare este lipsa de automatizare integrată. Multe activități recurente, de la trimiterea invitațiilor și confirmărilor de înregistrare, până la generarea de badge-uri sau centralizarea feedback-ului, încă sunt realizate manual sau semi-manual din cauza lipsei unui flux unificat. Organizatorii ajung să suplimenteze munca cu foi de calcul sau proceduri ad-hoc pentru zone pe care software-urile disponibile nu le acoperă [12]. Acest lucru consumă timp și resurse umane suplimentare și crește șansele de eroare. De exemplu, dacă platforma de ticketing nu se integrează cu cea de email marketing, atunci exportul și importul de liste de participanți devin pași manuali repetitivi. Automatizarea insuficientă se traduce și prin dificultatea de a obține o vedere de ansamblu în timp real asupra evenimentului – datele despre vânzările de bilete, preferințele participanților, logistică etc. fiind dispersate, este dificil de agregat rapid un tablou complet pentru luarea deciziilor operative.

O altă provocare majoră o reprezintă creșterea așteptărilor publicului participant. În era digitală, participanții, în special tinerii generației Millennials și Gen Z, sunt obișnuiți cu tehnologia și așteaptă experiențe personalizate, interactive și dinamice la evenimente [11]. Simpla prezență fizică la o conferință sau concert nu mai este suficientă; publicul își dorește elemente digitale complementare (aplicații mobile ale evenimentului, posibilitatea de a interacționa online în timpul sesiunilor, conținut media în timp real, opțiuni de networking virtual etc.). Un studiu din 2025 subliniază că participanții de azi trăiesc într-o lume "digital-first", unde tehnologia le modelează experiențele, așa încât evenimentele trebuie să recunoască aceste așteptări prin personalizare și interacțiuni sporită [11]. Dacă organizatorii nu reușesc să livreze o astfel de experiență modernă, riscă să dezamăgească o audiență foarte vocală și exigentă. De altfel, mulți participanți nu ezită să își manifeste nemulțumirea față de experiențele digitale neconvingătoare oferite de organizatori. Potrivit unei analize, utilizatorii se plâng adesea de experiența digitală nesatisfăcătoare la evenimente, de exemplu, aplicații sau site-uri greoaie, agende neactualizate, lipsa unei platforme unice pentru întreg conținutul, iar erorile apărute din integrarea slabă a sistemelor

(ex: discrepanțe între sistemul de înregistrare și cel de program al sesiunilor) le reduc satisfacția și loialitatea față de eveniment [12]. Publicul se așteaptă la coerență: de la un proces de check-in rapid la acces facil la informații și interacțiuni fără întreruperi, totul ideal printr-o singură interfață unificată.

Nu în ultimul rând, organizatorii se confruntă cu costuri și riscuri crescute. Costurile ridicate provin parțial din aspectele discutate – fragmentarea impune achiziția și întreținerea mai multor instrumente software (fiecare cu abonamentul și trainingul aferent) [12], precum și angajarea unui personal tehnic mai numeros pentru operarea și integrarea lor. La acestea se adaugă costurile sporite ale serviciilor și logisticii în contextul standardelor tot mai ridicate (de exemplu, evenimentele hibride necesită atât infrastructură fizică, cât și platforme digitale robuste). Pe frontul de riscuri, evenimentele actuale implică o planificare minuțioasă a siguranței (atât fizice, cât și cibernetice). Fragmentarea software crește riscul de incidente de securitate a datelor (breșe în transferul de informații între sisteme, conform constatărilor din industrie [12]) și face mai dificilă conformitatea cu cerințele de protecție a datelor (GDPR etc.). De asemenea, complexitatea mai mare a evenimentelor moderne (audiențe numeroase, componentă virtuală globală) înseamnă vulnerabilitate sporită la probleme neprevăzute: eșecul unei platforme tehnice poate compromite întreg evenimentul. Totodată, orice deficiență în execuție este rapid amplificată de rețelele sociale, amplificând riscul reputațional. În concluzie, deși evenimentele contemporane beneficiază de tehnologie și know-how avansat, ele pun organizatorilor și provocări substanțiale, de la a menține legătura operațiunilor într-un mediu tehnologic fragmentat, până la a satisface un public exigent, fără a escalada necontrolat costurile și expunerea la riscuri.

2.3 Analiza soluțiilor existente

Pentru a face față provocărilor de mai sus, industria utilizează diverse platforme software menite să simplifice organizarea de evenimente. Dintre acestea, Eventbrite și Ticketmaster se numără printre cele mai cunoscute soluții, fiecare abordând managementul evenimentelor din unghiuri diferite. Vom analiza pe scurt funcționalitățile oferite de aceste platforme, limitările lor și modul în care răspund (sau nu) provocărilor curente.

2.3.1 Eventbrite – platformă de organizare și ticketing

Eventbrite este o platformă populară de ticketing și management de eveniment, recunoscută pentru accesibilitate și ușurința în utilizare. Aceasta permite crearea și promovarea evenimentelor, vânzarea de bilete online și gestionarea participanților, totul printr-o interfață web intuitivă. Un avantaj major al Eventbrite îl constituie rețeaua sa extinsă de

utilizatori: organizatorii au acces la o bază de peste 90 de milioane de cumpărători activi de bilete, ceea ce conferă evenimentelor listate o vizibilitate considerabilă și potențial de marketing intrinsec [15]. Platforma oferă pagini dedicate pentru fiecare eveniment, personalizabile într-o anumită măsură, unde se pot afișa informații, imagini și descrieri, asigurând un proces de înscriere simplificat pentru participanți. De asemenea, Eventbrite asigură procesarea plăților și posibilitatea de check-in electronic la fața locului, inclusiv prin aplicația mobilă Eventbrite Organizer, care ajută organizatorii să monitorizeze în timp real vânzările de bilete și statutul participanților [15]. Funcții utile precum generarea de coduri promoționale (pentru discount-uri la bilete), opțiuni de rapoarte și analitice de vânzări, precum și un modul de trimitere invitații pe email sunt integrate pentru a sprijini gestionarea evenimentului în ansamblu [13].

Cu toate aceste funcționalități, Eventbrite prezintă și limitări notabile. În primul rând, structura de cost implică comisioane pe bilet și taxe care pot deveni semnificative, mai ales pentru evenimente cu taxe de participare mari sau volum mare de bilete vândute – aspect subliniat frecvent de utilizatori ca factor negativ [15]. În al doilea rând, posibilitățile de personalizare aprofundată a experienței (atât pe pagina evenimentului, cât și în procesele conexe) sunt relativ limitate: organizatorii pot adăuga branding de bază și câmpuri personalizate la înscriere, însă design-ul general și fluxurile sunt predeterminate de platformă, fără opțiuni avansate de configurare a interfeței sau logici custom (decât dacă se recurge la API-ul Eventbrite, ceea ce necesită competențe tehnice considerabile [13]). De asemenea, integrarea Eventbrite cu alte instrumente externe poate fi o provocare. Pentru funcționalități pe care Eventbrite nu le acoperă nativ – de exemplu streaming live al evenimentului, componente de engagement avansat (sondaje live, Q&A în timp real) sau website-uri dedicate evenimentului – organizatorii sunt nevoiți să apeleze la soluții terțe. Acest lucru complică fluxul de lucru, necesitând adesea proceduri manuale de transfer al datelor între Eventbrite și celelalte platforme, cu consum de timp și risc de erori [13]. Un ghid de bune practici avertizează că dependența de software terț pentru streaming, engagement sau site-uri web impune fie găsirea unor integrări compatibile, fie transferuri manuale de informații, ceea ce poate duce la greșeli și la un plus de efort [13]. În sfârșit, Eventbrite a fost conceput inițial ca un serviciu de ticketing generalist, astfel că nu acoperă toate nevoile posibile ale unui eveniment complex; de exemplu, gestionarea relației cu sponsorii, planificarea detaliată a resurselor sau funcționalități de networking între participanți nu fac parte din oferta standard. Astfel, în ciuda beneficiilor evidente (simplificare parțială, acces la piață, analiză de bază), Eventbrite nu elimină pe deplin fragmentarea ecosistemului tehnologic folosit de un organizator, mai ales pentru evenimente de amploare sau cu cerințe

speciale. Un rezumat al avantajelor și dezavantajelor platformei concluzionează că, deși este o soluție solidă și cuprinzătoare pentru multe situații, taxele, opțiunile limitate de personalizare și dificultățile de integrare cu alte sisteme pot constitui impedimente dacă evenimentul depășește ca scală și complexitate ceea ce Eventbrite poate gestiona optim [13].

2.3.2 Ticketmaster – platformă globală de ticketing

Ticketmaster este un alt nume de referință în industria evenimentelor, în special pe segmentul de concerte, spectacole de mare anvergură și evenimente sportive. Spre deosebire de Eventbrite, care se adresează și organizatorilor independenți sau evenimentelor de nișă, Ticketmaster operează mai mult ca un sistem de ticketing enterprise, colaborând strâns cu marii promotori, săli de spectacole și organizatori de turnee. Platforma oferă un set robust de funcționalități axate pe gestionarea avansată a biletelor: de la configurarea planurilor de sală și definirea categoriilor de preț (cu vizualizarea în timp real a impactului asupra veniturilor [15]), până la opțiuni de pre-vânzare și alocare a cotelor de bilete (de exemplu, rezervarea unor blocuri de bilete pentru parteneri sau promoții speciale) [15]. Ticketmaster permite organizatorilor mari să controleze parametrii evenimentelor (prețurile, numărul de bilete per cumpărător, restricții de revânzare etc.), în timp ce platforma se ocupă de procesul efectiv de vânzare către public și de accesul pe multiple canale (site-ul și aplicația Ticketmaster, parteneri afiliați, aplicații mobile de fan engagement ș.a.) [15]. Un beneficiu major al Ticketmaster este scala publicului și infrastructura sa: pentru evenimente cu cerere foarte mare (de exemplu marile concerte), sistemul poate gestiona volume mari de tranzacții și are implementate mecanisme anti-bot și anti-scalping, asigurând un acces echitabil pe cât posibil. De asemenea, Ticketmaster oferă instrumente pentru monitorizarea vânzărilor în timp real și rapoarte detaliate, ceea ce ajută promoterii să își calibreze strategiile de preț (ex. folosind dynamic pricing în unele cazuri).

Cu toate acestea, limitările Ticketmaster devin evidente mai ales din perspectiva flexibilității și a costurilor. În primul rând, structura de comisioane și taxe ale Ticketmaster este adesea criticată ca fiind oneroasă: numeroși utilizatori și organizatori au remarcat că taxele de serviciu sunt foarte ridicate și cresc prețul final al biletelor semnificativ [15]. Acest lucru poate crea nemulțumiri în rândul publicului și uneori descurajează potențialii cumpărători, afectând astfel experiența per ansamblu a evenimentului. În al doilea rând, Ticketmaster este în esență o soluție centrată pe ticketing și nu funcționează ca un sistem integrat de management pentru toate aspectele unui eveniment. De exemplu, dacă un organizator ar dori să importe în Ticketmaster bilete vândute printr-un alt canal (de pildă, invitații emise

separat sau vânzări corporative directe), platforma nu oferă suport facil pentru acest lucru, forțând practic utilizarea exclusivă a propriului sistem. Această lipsă de interoperabilitate poate fi problematică atunci când se dorește combinarea datelor din mai multe surse sau migrarea de la o altă platformă. Mai mult, pentru elemente precum managementul conținutului conferinței, interacțiunea cu participanții sau gestionarea logisticii (cazare, transport, agende personalizate etc.), Ticketmaster nu furnizează module dedicate – organizatorii vor trebui să apeleze la alte aplicații sau servicii separate, intrând din nou în sfera fragmentării de care discutăm. Practic, Ticketmaster abordează provocările legate de vânzarea biletelor la scară mare, asigurând un proces fiabil și controlat pe această componentă critică, însă nu răspunde direct altor provocări precum automatizarea planificării interne sau crearea unei experiențe digitale unificate pentru participanți dincolo de achiziția biletului. În plus, dominanța de piață a platformei vine cu condiții stricte: adesea marile locații au contracte exclusive cu Ticketmaster, ceea ce limitează opțiunile organizatorilor și le poate diminua controlul asupra canalelor de vânzare și a relației cu clienții. Conform opiniilor centralizate de analiști, deși Ticketmaster excelează ca soluție de ticketing, numeroase alternative emergente oferă funcționalități mai cuprinzătoare (precum gestionarea integrală a evenimentului) la costuri mai mici [15]. Astfel, în contextul necesităților actuale, Ticketmaster poate fi văzut mai degrabă ca o componentă (esentială pentru ticketing) dintr-un ecosistem tehnologic al evenimentului, decât ca un sistem unic care să rezolve toate neajunsurile identificate anterior.

2.4 De ce este necesară o platformă integrată

Având în vedere provocările și limitările discutate, devine evident că soluțiile existente, deși utile, nu rezolvă pe deplin problemele complexe cu care se confruntă managementul evenimentelor moderne [9][10]. Platforme populare precum Eventbrite sau Ticketmaster, deși eficiente pentru sarcini specifice cum ar fi rezervarea biletelor, suferă adesea din cauza fragmentării instrumentelor și lipsei unei experiențe integrate coerente, atât pentru organizatori cât și pentru participanți [13][15]. În acest context, necesitatea unei platforme integrate devine evidentă, iar EventHub propune exact acest tip de abordare holistică, oferind funcționalități avansate care lipsesc în mod semnificativ din alte platforme existente.

Mai mult, EventHub include o interfață avansată și coerentă de administrare ce permite organizatorilor să gestioneze evenimentele dintr-un singur panou centralizat: de la aprobarea și moderarea evenimentelor până la analiza performanței, generarea de rapoarte financiare detaliate și controlul transparent al tuturor tranzacțiilor și operațiunilor. În timp ce alte platforme necesită adesea utilizarea unor instrumente suplimentare pentru raportare și

analiză detaliată, EventHub oferă aceste capabilități încorporate direct în platformă, eliminând astfel nevoia de comutare între aplicații separate și spreadsheet-uri manuale.

Din perspectiva participanților, avantajele unice ale platformei EventHub includ autentificarea socială integrată (prin Google și Facebook), istoricul detaliat al achizițiilor și participărilor și suportul live prin chat în timp real, asigurat prin integrarea nativă a Socket.io. Spre deosebire de platformele tradiționale, care separă experiențele de căutare, achiziție, validare și suport, EventHub unifică aceste procese într-o singură interfață, oferind astfel utilizatorilor o experiență fluentă și continuă de la înscriere până la participarea efectivă.

În plus, platforma integrează elemente cheie de securitate, precum autentificarea robustă bazată pe JWT, criptarea completă a datelor sensibile și sisteme automate de limitare a ratelor și prevenție a abuzurilor. Aceste caracteristici depășesc standardele de bază ale altor platforme populare, oferind astfel organizatorilor și participanților o siguranță sporită și conformitate garantată într-un mediu unic și controlabil.

Prin urmare, EventHub nu doar că simplifică în mod considerabil gestionarea evenimentelor, dar aduce și avantaje competitive concrete prin integrarea inteligentă a funcționalităților care sunt fragmentate sau inexistente în alte soluții existente, contribuind astfel direct la reducerea costurilor operaționale, eliminarea riscurilor asociate utilizării de platforme multiple și oferind o experiență de utilizare net superioară atât pentru organizatori cât și pentru participanți.

3 Tehnologii și arhitectura sistemului EventHub

3.1 Arhitectura aplicațiilor web moderne (Client-Server, REST API, SPA)

În prezent, majoritatea aplicațiilor web se bazează pe un model arhitectural de tip client-server. În acest model, clienții (de regulă aplicații front-end care rulează în browser) comunică cu un server central care oferă date și servicii. Separarea rolurilor permite dezvoltarea independentă a interfeței (clientului) și a logicii de business (serverului), precum și scalarea independentă a fiecărei componente. Un exemplu clasic este arhitectura REST (Representational State Transfer), în care serverul expune API-uri HTTP stateless pentru resurse diferite. REST promovează constrângeri precum **statelessness** (fiecare cerere conține toate informațiile necesare) și utilizarea metodelor HTTP standard (GET, POST, PUT, DELETE etc.) [16], facilitând interoperabilitatea între componentele distribuite.

În practică, există două abordări principale de design al aplicațiilor web: aplicații web tradiționale (multi-pagină) și aplicații de tip SPA (Single Page Application). Aplicațiile tradiționale generează pe server pagini HTML complete, livrate clientului la fiecare acțiune. În schimb, o aplicație SPA încarcă o singură pagină HTML și își actualizează dinamic conținutul în browser folosind JavaScript. Logica interfeței (rute, redări de componentă) se desfășoară în browser, iar datele sunt obținute și trimise către server prin API-uri web (de obicei RESTful). Acest model SPA oferă o experiență de utilizator fluidă, fără reîncărcări complete de pagină. De exemplu, platforma EventHub este implementată ca o SPA React: browser-ul clientului solicită date de la server numai prin cereri AJAX la API-urile expuse de backend. Totuși, aplicațiile SPA necesită o atenție sporită la securitate și infrastructură (configurare build, CORS, cache, SEO), după cum subliniază literatura de specialitate. Documentațiile tehnice recomandă utilizarea SPA atunci când este necesară o interfață foarte bogată în elemente dinamice [17] și când echipa de dezvoltare este orientată JavaScript/TypeScript.

3.2 Tehnologii backend

3.2.1 Node.js și ecosistemul JavaScript

Backend-ul aplicației EventHub se bazează pe Node.js, un mediu de execuție JavaScript care permite rularea codului JS pe server. Node.js a fost creat în 2009 de Ryan Dahl și este open-source, cross-platform, construit pe motorul JavaScript V8 al Chrome. Aceasta a reprezentat o inovație semnificativă deoarece a unit limbajul JavaScript între front-end și back-end, facilitând dezvoltarea aplicațiilor web full-stack. Un aspect definitoriu al Node.js este modelul său asincron, event-driven [18]: în loc să blocheze firul principal în așteptarea operațiilor de I/O (citire fișiere, acces baza de date, rețea), Node.js folosește un loop de evenimente și callback-uri pentru a trata operațiile în mod neblokant. Acest lucru înseamnă că un server Node poate gestiona multe solicitări simultan, având întotdeauna firul de execuție principal liber, ceea ce îmbunătățește performanța aplicațiilor I/O-intensive (de exemplu servere HTTP cu trafic mare sau sisteme în timp real).

Concret, Node.js oferă module native pentru lucrul cu sistemul de fișiere, protocoalele de rețea și HTTP, permițând construirea rapidă de servere web. De asemenea, ecosistemul Node include managerul de pachete npm, care conține sute de mii de module reutilizabile pentru autentificare, baze de date, gestiune de sesiuni, criptografie și multe altele. Combinând viteza motorului V8 cu arhitectura bazată pe evenimente și I/O neblokant, Node.js atinge performanță și scalabilitate ridicate, fiind potrivit pentru aplicații real-time și de mari

dimensiuni. În ansamblu, Node.js este foarte popular datorită eficienței și faptului că dezvoltatorii pot folosi același limbaj (JavaScript/TypeScript) la nivelul întregului stack.

3.2.2 Framework-ul Express.js

Pentru a facilita dezvoltarea de API-uri și aplicații web în Node.js, EventHub utilizează framework-ul Express.js. Express este un framework web minimalist și flexibil, care oferă un set robust de funcționalități de bază [19] pentru aplicații web și mobile. În practică, Express permite definirea ușoară a rutelor HTTP (de exemplu `app.get('/evenimente', ...)` sau `app.post('/utilizatori', ...)`) și utilizarea de middleware-uri intermediare pentru procesarea cererilor (autentificare, logare, validare de date etc.). Astfel, dezvoltarea unui endpoint RESTful cu Express este rapidă, deoarece framework-ul pune la dispoziție metode HTTP și middleware-uri variate. De exemplu, Express dispune de mijloace integrate pentru manipularea JSON, setări de router și răspunsuri HTTP, iar extensibilitatea sa permite adăugarea de pachete adiționale (body-parser, cors, helmet, cookie-parser etc.) în funcție de nevoi. Pe scurt, Express oferă un schelet eficient de aplicație web care nu ascunde deloc facilitățile native ale Node.js, rămânând performant și ușor de personalizat.

3.2.3 Baze de date relaționale – PostgreSQL

EventHub folosește PostgreSQL ca sistem de gestiune a bazelor de date relaționale. PostgreSQL este un sistem ORDBMS open-source cu o reputație solidă [20] pentru fiabilitate și performanță. Documentația oficială îl descrie drept „un sistem obiect-relational puternic și open-source, cu peste 35 de ani de dezvoltare”. Spre deosebire de bazele de date NoSQL, PostgreSQL implementează pe deplin modelul relațional și limbajul SQL, oferind tranzacții ACID [21] pentru integritatea datelor, constrângeri, indexare avansată și interogări complexe (join-uri, subselecturi etc.). În EventHub, PostgreSQL stochează tabelele principale (de exemplu utilizatori, evenimente, înscrieri) și relațiile dintre ele. Accesul la bază din Node.js se face prin drivere dedicate (de exemplu modulul pg), unde serverul Node execută interogări SQL sau apeluri de proceduri stocate. PostgreSQL mai oferă extensii utile, precum PostGIS pentru date geografice (utile la geocoding) și mecanisme de replicare, care pot fi utile pentru scalabilitate și backup. În ansamblu, alegerea PostgreSQL asigură consistența și performanța necesare aplicației, beneficiind totodată de ecosistemul său matur și de comunitatea activă.

3.2.4 Autentificare și autorizare (JWT, OAuth)

Pentru gestionarea securității, EventHub implementează autentificare și autorizare pe bază de token-uri. În mod uzual, la autentificare utilizatorul furnizează credențiale, iar serverul

validează și emite un token care atestă identitatea acestuia. Un mecanism comun este **JWT [22] (JSON Web Token)**, care permite transmiterea securizată a informațiilor despre utilizator între client și server. Un JWT constă dintr-un header, un payload (cu revendicări de tip `userId`, `exp` etc.) și o semnătură criptografică. Documentația tehnică arată că JWT este utilizat pentru autentificare în aplicații web și API-uri, asigurând schimbul securizat de informații între client și server. Astfel, după ce utilizatorul se autentifică cu succes, serverul generează un JWT semnat și îl trimite clientului (de obicei în răspunsul la login). Clientul stochează tokenul și îl trimite în antetul `HTTP Authorization: Bearer <token>` la fiecare cerere ulterioară. Serverul verifică semnătura tokenului pentru a valida identitatea și permisiunile utilizatorului înainte de a accesa resursele protejate. Această abordare **fără stare (stateless)** simplifică scala serverului și elimină necesitatea stocării sesiunii pe server.

Pentru integrarea cu servicii externe sau autentificarea prin conturi de social login, se poate utiliza **OAuth 2.0**. OAuth 2.0 este un cadru de autorizare care permite aplicațiilor terțe să obțină acces limitat la resursele unui utilizator fără a-i obține direct credențialele [23]. Standardul definește fluxuri de autorizare (de exemplu `Authorization Code Grant`) care permit, de exemplu, logarea prin Google sau Facebook. Astfel, utilizatorul se autentifică la un provider (ex. Google), iar EventHub primește de la acel provider un token de acces (și eventual un refresh token) fără ca utilizatorul să-și partajeze parola cu EventHub. Conform surselor, OAuth 2.0 rezolvă problemele de securitate asociate partajării credențialelor între aplicații și oferă flow-uri bine definite pentru web și mobile. În EventHub, OAuth poate fi utilizat opțional pentru a oferi login cu conturi externe sau pentru a autoriza aplicația să acceseze API-uri protejate ale terților, extinzând astfel funcționalitatea acesteia.

3.3 Tehnologii frontend

3.3.1 React.js și ecosistemul său

Partea de interfață (front-end) a aplicației EventHub este realizată cu React.js. React este o bibliotecă JavaScript pentru construirea interfețelor de utilizator. Conform documentației, React permite dezvoltatorilor să definească vizualizări declarative [24] pentru fiecare stare a aplicației și actualizează eficient componentele afectate atunci când datele se schimbă. Un avantaj important al React este mecanismul Virtual DOM: în loc să modifice direct structura DOM real (operațiune costisitoare), React menține o copie virtuală și sincronizează doar diferențele, îmbunătățind performanța redării. De asemenea, React este orientat pe componente: interfața este împărțită în componente reutilizabile și încapsulate, fiecare gestionându-și propria stare internă. Această abordare component-based face codul mai

modular și ușor de întreținut. În EventHub, componentele React sunt folosite pentru pagini și secțiuni precum lista de evenimente, detaliile unui eveniment, formularele de înregistrare și panourile utilizatorilor.

În jurul React există un ecosistem bogat de biblioteci complementare. De exemplu, **React Router** permite definirea rutelor interne în SPA (navigație între vizualizări fără reîncărcare de pagină). Pentru configurarea build-ului se folosesc de obicei Webpack (sau utilitare similare) și Babel, asigurând suport pentru transpiler ES6/TypeScript și optimizări de pachete. Alte unelte utile sunt ESLint (linting) și Prettier (formatat cod), care contribuie la calitatea codului. În ansamblu, React și ecosistemul său oferă flexibilitate și productivitate dezvoltatorilor, iar în EventHub această tehnologie asigură un front-end reactiv, performant și ușor de extins.

3.3.2 Managementul stării aplicației

Într-o aplicație SPA (Single-Page Application) precum EventHub, conceptul de „stare” (state) desemnează totalitatea datelor volatile pe care UI-ul trebuie să le afișeze și să le modifice: utilizatorul autentificat, preferințele de filtrare, coșul de bilete, sesiunea de chat etc. React oferă mecanisme diferite pentru a manipula această stare:

Stare locală – variabile de componentă (useState, useReducer), ideale pentru date care nu părăsesc componenta.

Stare globală – date care trebuie consumate de multe componente la nivelul întregii aplicații (de exemplu identitatea utilizatorului), expuse prin Context API sau printr-un state container dedicat (Redux, Zustand, Recoil).

Stare server – date persistente venite prin API, gestionate cu biblioteci de data fetching (React Query, SWR) pentru cache, revalidare și background refresh.

Context API pentru stare globală

Context API este soluția built-in a React pentru a evita prop-drilling-ul (trimiterea de proprietăți în lanț prin mai multe niveluri de componente). Un Context înseamnă:

- un **obiect de context** creat cu createContext() – definește forma datelor;
- un **Provider** – componentă care „împarte” starea tuturor descendenților săi;
- unul sau mai mulți **Consumers** (direct sau prin useContext) care citesc/actualizează acea stare.

```

// src/context/AuthContext.jsx
import { createContext, useState, useEffect } from 'react';
import api from '../services/api';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState(null);
  const [loading, setloading] = useState(true);
  const [error, setError] = useState(null);

  useEffect(() => {
    const checkAuth = async () => {
      // Verificăm întâi în sessionStorage
      let token = sessionStorage.getItem('token');

      // Dacă nu există în sessionStorage, verificăm în localStorage
      if (!token) {
        const authType = localStorage.getItem('authType');

        // Folosim token-ul din localStorage doar dacă tipul de autentificare este persistent
        if (authType === 'persistent') {
          token = localStorage.getItem('token');
        }
      }

      if (token) {
        try {
          // Configurăm header-ul de autorizare pentru toate cererile viitoare
          api.defaults.headers.common['Authorization'] = `Bearer ${token}`;

          const response = await api.get('/auth/me');
          setUser(response.data.user);
        } catch (err) {
          console.error('Error verifying authentication:', err);

          // Curățăm datele de autentificare în caz de eroare
          sessionStorage.removeItem('token');
          localStorage.removeItem('token');
          localStorage.removeItem('authType');
          delete api.defaults.headers.common['Authorization'];
        }
      }

      setloading(false);
    };

    checkAuth();
  }, []);

  const login = async (email, password, rememberMe = false) => {
    setError(null);
    try {
      const response = await api.post('/auth/login', { email, password });

      // Save token based on "Remember me" option
      if (rememberMe) {
        localStorage.setItem('token', response.data.token);
        localStorage.setItem('authType', 'persistent');
      } else {
        sessionStorage.setItem('token', response.data.token);
        localStorage.setItem('authType', 'session');
      }

      api.defaults.headers.common['Authorization'] = `Bearer ${response.data.token}`;
      setUser(response.data.user);
    }
  };

```

```

    setUser(response.data.user);

    return response.data; // Make sure to return this for redirect Logic
  } catch (err) {
    setError(err.response?.data?.error || 'A apărut o eroare la autentificare');
    throw err;
  }
};

const register = async (userData) => {
  setError(null);
  try {
    const response = await api.post('/auth/register', userData);
    return response.data;
  } catch (err) {
    setError(err.response?.data?.error || 'A apărut o eroare la înregistrare');
    throw err;
  }
};

const logout = () => {
  localStorage.removeItem('token');
  localStorage.removeItem('authType');
  sessionStorage.removeItem('token');
  setUser(null);
};

const socialLogin = (provider) => {
  window.location.href = `${api.defaults.baseURL}/auth/${provider}`;
};

return (
  <AuthContext.Provider
    value={{
      user,
      loading,
      error,
      login,
      register,
      logout,
      socialLogin,
      setUser
    }}
  >
    {children}
  </AuthContext.Provider>
);
};

export default AuthContext;

```

Fluxul de autentificare

1. La montarea AuthProvider, hook-ul useEffect pornește validarea tokenului (dacă există).
2. Dacă tokenul este valid, API-ul /auth/me returnează profilul; contextul setează user.
3. Dacă tokenul lipsește sau este invalid, aplicația rămâne în stare guest.
4. Funcția login salvează noul token și actualizează user; logout inversează procesul.

Persistență: sessionStorage vs. localStorage

- **sessionStorage**: se golește la închiderea tab-ului sau browser-ului; potrivit sesiunilor scurte.
- **localStorage**: persistă între sesiuni; util pentru opțiunea Remember me. Alegerea este controlată de flag-ul rememberMe; avantajul este că nu expune niciodată tokenul în cookie-uri (evită CSRF), ci doar în Web Storage accesibil prin JavaScript.

Custom Hook – useAuth()

Pentru a ascunde detaliile implementării Context-ului și a oferi un API ergonomic, se definește un custom hook:

```
// src/hooks/useAuth.js
import { useContext } from 'react';
import AuthContext from '../context/AuthContext';

export function useAuth() {
  return useContext(AuthContext);
}
```

Beneficii:

- Orice componentă poate importa useAuth() și accesa direct { user, login, logout } fără a cunoaște implementarea;
- Crește lizibilitatea codului și promovează composition vs. inheritance (pattern React modern).

3.3.3 Tailwind CSS pentru stilizare

Pentru stilizarea interfețelor, EventHub folosește **Tailwind CSS**, un framework CSS de tip *utility-first* [26]. Tailwind nu impune stiluri prestabilite amplasate în componente, ci oferă clase de utilitate de nivel scăzut (margini, padding, culori, aliniere etc.) care pot fi combinate pentru a crea rapid design-uri personalizate. Documentația oficială descrie Tailwind astfel: „Tailwind CSS este un framework CSS foarte personalizabil, de nivel scăzut, care îți oferă toate elementele de bază necesare pentru a construi designuri personalizate”. Prin urmare, în loc de a scrie fișiere CSS separate sau de a rescrie stiluri existente (cum ar face un framework mai „opinioned” precum Bootstrap), un dezvoltator aplică clase direct în mark-up (de exemplu class="bg-blue-500 text-white p-4 rounded"). Avantajul este generarea unui cod CSS minim (doar clasele folosite sunt exportate) și flexibilitatea de a modifica stiluri prin

ajustarea claselor din HTML/JSX. Tailwind este configurabil prin fișiere de configurare, permițând extinderea temei (culori, fonturi) sau adăugarea de utilitare personalizate. În React, integrarea Tailwind se face ușor, și anume, doar se adaugă clasele în JSX. Astfel, stilizarea cu Tailwind menține coerența vizuală și accelerează dezvoltarea UI-ului fără costuri mari de întreținere.

3.4 Comunicarea în timp real (WebSocket, Socket.io)

Pentru facilitarea comunicațiilor bidirecționale în timp real între client și server, EventHub utilizează tehnologia **WebSocket**. Protocolul WebSocket deschide un canal de comunicație full-duplex [27], bidirecțional și persistent peste o singură conexiune TCP. Spre deosebire de cererile HTTP standard, care necesită deschiderea unei conexiuni pentru fiecare interacțiune, WebSocket-ul inițializează o singură conexiune printr-un handshake inițial (upgrade HTTP) și apoi menține conexiunea deschisă. Asta permite clientului și serverului să trimită mesaje oricând, minimizând latența. De exemplu, dacă un utilizator al EventHub participă la un chat sau la un feed de evenimente live, WebSocket-ul poate transmite instant mesajele sau actualizările de stare. În aplicația noastră, WebSocket poate fi folosit pentru a trimite notificări de tip „eveniment nou adăugat” sau pentru sincronizarea în timp real a datelor între mai mulți utilizatori.

Pe lângă WebSocket-ul nativ, EventHub poate folosi și **Socket.IO**, o bibliotecă JavaScript care oferă un API de nivel înalt pentru comunicații în timp real. Conform documentației Socket.IO, aceasta „permite comunicare în timp real [28], bidirecțională și bazată pe evenimente între browser și server”. Socket.IO combină un client JavaScript (în browser) și un server Node.js și gestionează automat fallback-uri către tehnici alternative (cum ar fi polling HTTP) dacă WebSocket-ul nu este suportat sau este blocat. Folosind Socket.IO în Node.js, se poate atașa un server Socket.IO la instanța de Express, iar clienții se conectează prin librăria de pe frontend. Aplicația poate defini evenimente personalizate (de exemplu `socket.emit('mesaj_chat', {...})`) pentru a trimite și recepționa date în timp real. În EventHub, Socket.IO ar facilita scenarii precum chat-ul live între participanți la un eveniment sau actualizarea dinamică a informațiilor de eveniment către toți clienții conectați, fără reîncărcări de pagină.

3.5 Servicii externe și integrări

Aplicația EventHub se integrează cu mai multe servicii externe pentru a extinde funcționalitatea de bază. Printre acestea se numără:

- **Servicii de email:** Pentru trimiterea de notificări și confirmări (de exemplu confirmare înregistrare la eveniment), se poate folosi Nodemailer. Nodemailer este un pachet Node.js care facilitează trimiterea de emailuri dintr-o aplicație backend [30], gestionând transportul SMTP și autentificarea cu OAuth2 dacă este necesar. El permite compunerea de mesaje text și HTML, atașamente etc., fără dependențe externe la runtime.
- **Geocoding:** Pentru asocieri geografice (căutarea locației unui eveniment), se poate apela Google Maps Geocoding API. Acest serviciu web primește o adresă sau un loc și returnează coordonatele latitudine/longitudine corespunzătoare [31]. Astfel, la crearea unui eveniment, adresa introdusă de organizator poate fi transformată în coordonate, iar ulterior acestea pot fi afișate pe o hartă.
- **Scheduled Tasks cu node-cron:** Automatizarea task-urilor recurente din EventHub se realizează cu node-cron, o implementare Node.js a sintaxei cron folosită în sistemele UNIX-like pentru programarea sarcinilor periodice [29]. Biblioteca permite declararea rapidă a unor job-uri care rulează la ore fixe, zilnic, săptămânal sau după expresii cron personalizate.

```
// backend/utils/scheduledTasks.js
import cron from 'node-cron';
import * as Ticket from '../models/Ticket.js';
import * as User from '../models/User.js';
import * as db from '../config/db.js';
import {
  sendEventCanceledEmail,
  sendEventRescheduledEmail,
  sendEventReminderEmail
} from '../emailService.js';
import refundService from '../services/refundService.js';

// Initialize all scheduled tasks
export const initializeScheduledTasks = () => {
  // Clean up past events - runs daily at 1 AM
  cron.schedule('0 1 * * *', async () => {
    try {
      await cleanupPastEvents();
    } catch (error) {
      console.error('Error in cleanup past events task:', error);
    }
  });
};
```

```
// Clean up past events
export const cleanupPastEvents = async () => {
  try {
    // Get current date and time
    const now = new Date();

    // Log with time zone information for debugging
    console.log(`Running cleanup at: ${now.toString()} (local server time)`);
    console.log(`ISO time: ${now.toISOString()}`);
    console.log(`Time zone offset: ${now.getTimezoneOffset()} minutes`);

    // Format date for SQL (use your local format to match your database)
    const currentDateStr = now.toISOString().split('T')[0];
    const currentTimeStr = now.toTimeString().split(' ')[0];

    console.log(`Comparing with date: ${currentDateStr}, time: ${currentTimeStr}`);

    // Option 1: Using PostgreSQL's timezone conversion
    const result = await db.query(`
      UPDATE events
      SET status = 'inactive'
      WHERE (date < CURRENT_DATE OR (date = CURRENT_DATE AND "time" < CURRENT_TIME))
      AND status = 'active'
      RETURNING id, name, date, time
    `);

    console.log(`Updated ${result.rowCount} past events to inactive status:`, result.rows);
    return result.rowCount;
  } catch (error) {
    console.error('Error cleaning up past events:', error);
    throw error;
  }
};
```

Explicație.

Curățarea evenimentelor expirate: rulează zilnic la 01:00 și marchează „inactive” toate evenimentele a căror dată/oră a trecut, menținând baza de date curată.

Remindere automate: rulează la 09:00; caută evenimente programate peste exact șapte zile și trimite e-mailuri participanților cu detalii și QR-code-urile biletelor.

Prin folosirea node-cron serverul rămâne stateless (task-urile nu solicită trafic suplimentar spre client) și nu sunt necesare soluții externe (RabbitMQ, Celery, etc.) pentru job-uri simple, menținând complexitatea infrastructurii la minimum [29].

3.6 Testare și asigurarea calității (Jest, metodologii de testare)

Asigurarea calității codului este un aspect esențial în proiectarea EventHub. Pentru testarea aplicațiilor JavaScript (front-end și back-end), se folosește un framework dedicat: **Jest**. Jest este descris ca „un framework de testare JavaScript plăcut [32], cu accent pe simplitate”. Acesta permite rularea testelor unitare și de integrare, oferind funcționalități precum rulează testele în procese izolate pentru performanță ridicată, suport pentru snapshot testing (util pentru componente React) și generare automată de rapoarte de acoperire a codului. De exemplu, cu Jest se pot scrie teste pentru funcțiile din backend-ul Node (verificând logica API-urilor și a modelelor de date) și pentru componente React (verificând redări condiționate sau interacțiuni simple).

3.6.1 Configurarea Jest pentru testare

Configurația Jest:

```
// backend/jest.config.cjs
module.exports = {
  roots: ['<rootDir>/tests'],
  testMatch: ['**/*.test.js'],
  testEnvironment: 'node',
  silent: true,
  transform: {}
};
```

Package.json scripts pentru testare:

```
"scripts": {
  "test": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --silent",
  "test:utils": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --testPathPattern='tests/utils' --verbose",
  "test:models": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --testPathPattern='tests/models' --verbose",
  "test:controllers": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --testPathPattern='tests/controllers' --verbose",
  "test:services": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --testPathPattern='tests/services' --verbose",
  "test:middleware": "node --experimental-vm-modules node_modules/jest/bin/jest.js\n--config=jest.config.cjs --testPathPattern='tests/middleware' --verbose",
```

3.6.2 Testarea modelelor

Exemplu de test pentru modelul Event:

```
// backend/tests/models/Event.test.js
import { jest } from '@jest/globals';

// Mock database
const mockDbQuery = jest.fn();
jest.unstable_mockModule('../../config/db.js', () => ({
  query: mockDbQuery
}));

describe('Event Model', () => {
  let Event;
  let consoleSpy, consoleErrorSpy;

  beforeAll(async () => {
    Event = await import('../../models/Event.js');
  });

  beforeEach(() => {
    jest.clearAllMocks();

    // Mock console methods
    consoleSpy = jest.spyOn(console, 'log').mockImplementation(() => {});
    consoleErrorSpy = jest.spyOn(console, 'error').mockImplementation(() => {});
  });

  afterEach(() => {
    consoleSpy.mockRestore();
    consoleErrorSpy.mockRestore();
  });

  describe('findById', () => {
    it('should find event by ID with detailed information', async () => {
      const mockEvent = {
        id: 1,
        name: 'Test Event',
        category_name: 'Concert',
        category_slug: 'concert',
        subcategory_name: 'Rock',
        subcategory_slug: 'rock',
        status: 'active'
      };

      mockDbQuery.mockResolvedValue({ rows: [mockEvent] });
```

```
      mockDbQuery.mockResolvedValue({ rows: [mockEvent] });

      const result = await Event.findById(1);

      expect(mockDbQuery).toHaveBeenCalledWith({
        text: expect.stringMatching(/SELECT e\.\.*FROM events e.*WHERE e\.id = \$1
AND e\.status IN \(\'active\', \'rescheduled\'\)/s),
        values: [1]
      });
      expect(result).toEqual(mockEvent);
    });
  });
});
```

3.6.3 Testarea controller-elor

Exemplu de test pentru EventController:

```
// tests/controllers/eventController.test.js

import { jest } from '@jest/globals';

// Test utilities
const createMockReq = (overrides = {}) => ({
  params: {},
  query: {},
  body: {},
  user: { id: 1, email: 'test@example.com', name: 'Test User', role: 'user' },
  ...overrides
});

const createMockRes = () => {
  const res = {};
  res.status = jest.fn().mockReturnValue(res);
  res.json = jest.fn().mockReturnValue(res);
  return res;
};

const createMockNext = () => jest.fn();

// Mock the EventService
const mockEventService = {
  getEventById: jest.fn(),
  incrementViewCount: jest.fn(),
  getEventTicketTypes: jest.fn()
};

jest.unstable_mockModule('../../services/EventService.js', () => ({
  EventService: jest.fn().mockImplementation(() => mockEventService)
}));

const { getEventById, incrementViewCount, getEventTicketTypes } = await import('../../controllers/eventController.js');
```

Tehnici aplicate:

- Mock Service Layer → controller testat izolat de business logic;
- req, res, next stubs pentru a verifica status & payload;
- Flux happy path + fluxuri de eroare (404, 500) pentru acoperire totală.

```

controllers/eventController.js');

describe('EventController', () => {
  let req, res, next;

  beforeEach(() => {
    req = createMockReq();
    res = createMockRes();
    next = createMockNext();

    Object.values(mockEventService).forEach(mock => mock.mockReset());
  });

  describe('getEventById', () => {
    it('should get event by ID successfully', async () => {
      const mockEvent = {
        event: {
          id: 1,
          title: 'Test Event',
          description: 'Test Description',
          date: '2024-12-31',
          time: '20:00:00',
          venue: 'Test Venue',
          view_count: 100
        },
        ticketTypes: [
          { id: 1, name: 'General', price: 50.00, available_quantity: 100 },
          { id: 2, name: 'VIP', price: 150.00, available_quantity: 50 }
        ],
        reviews: [
          { id: 1, rating: 5, comment: 'Great event!', user_name: 'John Doe' }
        ],
        averageRating: 4.5,
        totalReviews: 10
      };
      req.params = { id: '1' };
      mockEventService.getEventById.mockResolvedValue(mockEvent);

      await getEventById(req, res, next);

      expect(mockEventService.getEventById).toHaveBeenCalledWith('1');
      expect(res.status).toHaveBeenCalledWith(200);
      expect(res.json).toHaveBeenCalledWith(mockEvent);
    });
  });
});

```

Puncte-cheie:

- Se mock-uiesc interogările PG prin jest.fn, evitând dependența de baza de date;
- Se verifică SQL-ul generat prin expresii regulate – detectează modificări destructive;
- Pentru metode „create”, se simulează răspunsul DB și se asigură returnarea obiectului complet.

3.6.4 Testarea middleware-ului

Test pentru middleware-ul de autentificare:

```
// backend/tests/middleware/auth.test.js
import { jest } from '@jest/globals';

// Mock passport
const mockAuthenticate = jest.fn();
jest.unstable_mockModule('passport', () => ({
  default: {
    authenticate: mockAuthenticate
  }
})));

describe('Auth Middleware', () => {
  let authMiddleware;
  let req, res, next;

  beforeAll(async () => {
    authMiddleware = (await import('../../middleware/auth.js')).default;
  });

  beforeEach(() => {
    req = {
      headers: {},
      user: null
    };
    res = {
      status: jest.fn().mockReturnThis(),
      json: jest.fn().mockReturnThis()
    };
    next = jest.fn();
  });

  afterEach(() => {
    jest.clearAllMocks();
  });

  describe('successful authentication', () => {
    it('should authenticate valid user and call next()', () => {
      const mockUser = {
        id: 1,
        email: 'user@example.com',
        role: 'user'
      };
    });
  });
});
```

Focus pe:

- Rute protejate returnează 401 fără token;
- next() este apelat doar când tokenul este valid;
- Spionaj pe passport.authenticate simulează scenarii.


```

    email: 'user@example.com',
    role: 'user'
  });

  // Mock passport.authenticate to call callback with user
  mockAuthenticate.mockImplementation((strategy, options, callback) => {
    return (req, res, next) => {
      callback(null, mockUser, null);
    };
  });

  authMiddleware(req, res, next);

  expect(mockAuthenticate).toHaveBeenCalledWith(
    'jwt',
    { session: false },
    expect.any(Function)
  );
  expect(req.user).toEqual(mockUser);
  expect(next).toHaveBeenCalledTimes(1);
  expect(next).toHaveBeenCalledWith();
  expect(res.status).not.toHaveBeenCalled();
  expect(res.json).not.toHaveBeenCalled();
});

```

3.6.5 Testarea serviciilor

Test pentru EventService:

```

import { jest } from "@jest/globals";

// Mock models using unstable_mockModule
jest.unstable_mockModule("../models/Event.js", () => ({
  findById: jest.fn(),
  findRelated: jest.fn(),
  create: jest.fn(),
  update: jest.fn(),
  deleteEvent: jest.fn(),
  incrementViews: jest.fn(),
  findAll: jest.fn(),
  checkEventPermission: jest.fn(),
})));

jest.unstable_mockModule("../models/TicketType.js", () => ({
  findById: jest.fn(),
})));

jest.unstable_mockModule("../models/Review.js", () => ({
  findById: jest.fn(),
})));

describe("EventService", () => {
  let EventService;
  let EventModel;
  let TicketTypeModel;
  let ReviewModel;
  let eventService;

  beforeAll(async () => {
    EventModel = await import("../models/Event.js");
    TicketTypeModel = await import("../models/TicketType.js");
    ReviewModel = await import("../models/Review.js");
    const { EventService: EventServiceClass } = await import(
      "../services/EventService.js"
    );
    EventService = EventServiceClass;
  });

  beforeEach(() => {
    jest.clearAllMocks();
    eventService = new EventService();
  });
});

```

```

describe("getEventById", () => {
  it("should return event with ticket types and reviews", async () => {
    const eventId = 1;
    const mockEvent = {
      id: eventId,
      name: "Test Event",
      category_id: 1,
    };
    const mockTicketTypes = [
      { id: 1, name: "General", price: 50 },
      { id: 2, name: "VIP", price: 100 },
    ];
    const mockReviews = [
      { id: 1, rating: 5, comment: "Great!" },
      { id: 2, rating: 4, comment: "Good" },
    ];
    const mockRelatedEvents = [
      { id: 2, name: "Related Event 1" },
      { id: 3, name: "Related Event 2" },
    ];

    EventModel.findById.mockResolvedValue(mockEvent);
    TicketTypeModel.findByEventId.mockResolvedValue(mockTicketTypes);
    ReviewModel.findByEventId.mockResolvedValue(mockReviews);
    EventModel.findRelated.mockResolvedValue(mockRelatedEvents);

    const result = await eventService.getEventById(eventId);

    expect(EventModel.findById).toHaveBeenCalledWith(eventId);
    expect(TicketTypeModel.findByEventId).toHaveBeenCalledWith(eventId);
    expect(
      let EventModel: any
      EventModel.findById).toHaveBeenCalledWith(eventId);
    expect(EventModel.findRelated).toHaveBeenCalledWith(
      mockEvent.category_id,
      eventId,
      4
    );

    expect(result.event).toEqual(mockEvent);
    expect(result.ticketTypes).toEqual(mockTicketTypes);
    expect(result.reviews.averageRating).toBe("4.5");
    expect(result.relatedEvents).toEqual(mockRelatedEvents);
  });
});

```

Observații:

- Se mock-uiesc modelele – se validează doar orchestrarea serviciului;
- Se verifică excepțiile pentru date inexistente;
- Se asigură agregarea corectă a datelor conexe (tipuri de bilet, recenzii).

În plus, aplicația EventHub beneficiază și de integrări continue (CI) care execută automat suita de teste la fiecare commit și folosirea analizatoarelor statice de cod (linting). Prin această abordare, erorile sunt identificate precoce, iar comportamentul aplicației rămâne previzibil. Combinat cu refactorizare periodică și code review-uri, mecanismele de testare cu Jest asigură lansarea unei aplicații robuste și fiabile, aliniată cerințelor calitative din domeniul informaticii.

4 Proiectarea și implementarea sistemului EventHub

4.1 Analiza cerințelor și specificațiile funcționale

4.1.1 Identificarea cerințelor funcționale

Dezvoltarea sistemului EventHub a început cu o analiză detaliată a cerințelor funcționale, bazată pe studiul platformelor existente și identificarea nevoilor specifice utilizatorilor români. Prin analiza platformelor precum Eventbrite, TicketMaster și iaBilet.ro, s-au identificat următoarele categorii principale de funcționalități necesare.

Cerințele pentru utilizatorii finali includ capacitatea de a parcurge și descoperi evenimente prin diverse criterii de căutare și filtrare. Utilizatorii trebuie să poată vizualiza detalii complete ale evenimentelor, inclusiv informații despre locație, preț, disponibilitate și recenzii ale altor participanți. Sistemul trebuie să permită achiziționarea de bilete într-un mod sigur și intuitiv, cu posibilitatea de a salva metode de plată pentru tranzacții viitoare. De asemenea, utilizatorii trebuie să aibă acces la un istoric complet al achizițiilor și să poată gestiona biletele achiziționate prin intermediul unei interfețe dedicate.

Cerințele pentru organizatorii de evenimente se concentrează pe capacitatea de a crea și gestiona evenimente comprehensive, cu opțiuni pentru multiple tipuri de bilete, politici de anulare și descrieri detaliate. Sistemul trebuie să ofere instrumente pentru monitorizarea vânzărilor, gestionarea participanților și procesarea rambursărilor. O funcționalitate

esențială identificată este sistemul de check-in prin coduri QR, care să permită validarea rapidă a biletelor la intrarea în eveniment.

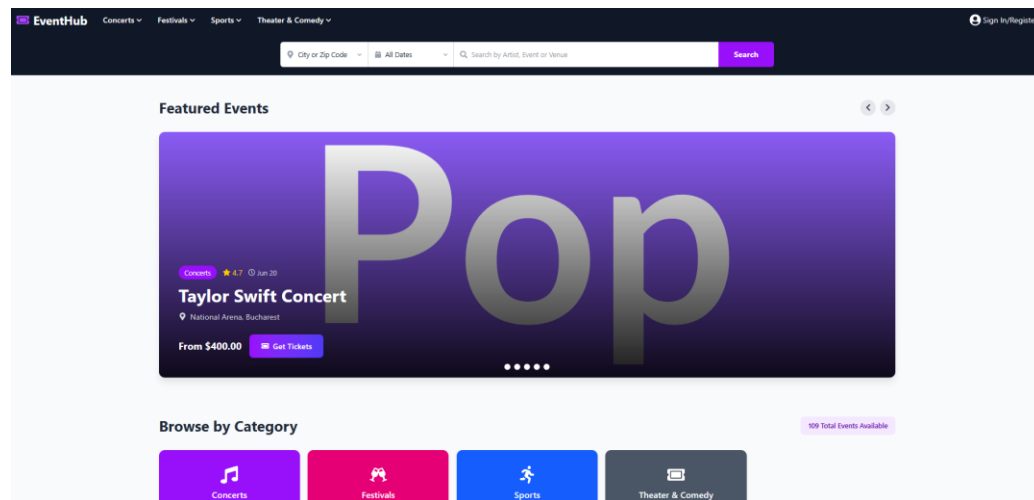
Cerințele administrative includ capacitatea de a modera conținutul platformei, de a gestiona utilizatorii și de a avea acces la rapoarte detaliate despre performanța platformei. Administratorii trebuie să poată aproba sau respinge evenimente, să gestioneze categoriile și subcategoriile de evenimente și să aibă acces la suportul pentru clienți prin intermediul unui sistem de chat integrat.

4.1.2 Cerințele non-funcționale

Pe lângă funcționalitățile explicite, s-au identificat și cerințe non-funcționale esențiale pentru succesul platformei. Performanța reprezintă un aspect critic, sistemul trebuind să răspundă la cereri în maxim 2 secunde pentru operațiile comune și să gestioneze concurrent cel puțin 1000 de utilizatori activi. Scalabilitatea a fost proiectată pentru a permite creșterea numărului de utilizatori și evenimente fără degradarea performanței, prin utilizarea unei arhitecturi modulare și a unor tehnologii care permit extinderea orizontală.

Securitatea datelor reprezintă o prioritate maximă, având în vedere că sistemul gestionează informații financiare sensibile și date personale ale utilizatorilor. S-au implementat măsuri de criptare pentru stocarea parolelor, autentificare bazată pe JWT tokens și integrare cu OAuth pentru platformele externe. Disponibilitatea sistemului trebuie să fie de cel puțin 99%, cu mecanisme de backup și recovery pentru situațiile de urgență.

Compatibilitatea cu diferite browsere și dispozitive a fost o cerință fundamentală, sistemul fiind proiectat să funcționeze optim pe desktop. Interfața trebuie să respecte principiile de accesibilitate web pentru a fi utilizabilă de către persoane cu dizabilități.



4.1.3 Specificațiile tehnice

Specificațiile tehnice au fost definite pentru a asigura îndeplinirea cerințelor funcționale și non-funcționale identificate. Arhitectura backend a fost concepută în jurul unui API RESTful dezvoltat în Node.js cu Express.js, oferind flexibilitate și performanță pentru gestionarea cererilor multiple simultane. Baza de date PostgreSQL a fost aleasă pentru capacitatea sa de a gestiona tranzacții complexe și pentru suportul robust pentru JSON, necesar pentru stocarea diverselor tipuri de date asociate evenimentelor.

Arhitectura frontend utilizează React.js pentru crearea unei interfețe de utilizator dinamice și responsive. Alegerea acestei tehnologii a fost motivată de ecosistemul bogat de componente disponibile și de capacitatea de a crea o experiență de utilizator fluidă prin Single Page Application (SPA). Integrarea cu Tailwind CSS asigură o stilizare modernă și consistentă pe toate dispozitivele.

Sistemul de comunicare în timp real a fost implementat folosind Socket.io pentru a permite funcționalități precum chat-ul pentru suportul clienților și notificările instantanee pentru schimbări importante ale evenimentelor. Această implementare asigură o experiență interactivă și modernă pentru utilizatori.

4.1.3 Analiza riscurilor și mitigarea acestora

În procesul de analiză a cerințelor s-au identificat și potențialele riscuri asociate implementării. Riscurile de securitate includ posibilitatea atacurilor asupra sistemului de plăți și accesul neautorizat la datele utilizatorilor. Pentru mitigarea acestora s-au implementat protocoale stricte de validare a datelor, criptare end-to-end pentru informațiile sensibile și monitoring continuu pentru detectarea activităților suspecte.

Riscurile de performanță pot apărea în cazul unui număr mare de utilizatori care accesează simultan platformă, în special în perioadele de vârf când se lansează eventos populare. Mitigarea acestui risc s-a realizat prin implementarea unui pool de conexiuni la baza de date, optimizarea query-urilor și pregătirea infrastructurii pentru scaling orizontal.

Riscurile de integrare cu serviciile externe (plăți, email, maps) pot afecta funcționalitatea platformei. Pentru acestea s-au implementat mecanisme de fallback și sisteme de notificare pentru administratori în cazul în care serviciile externe nu sunt disponibile.

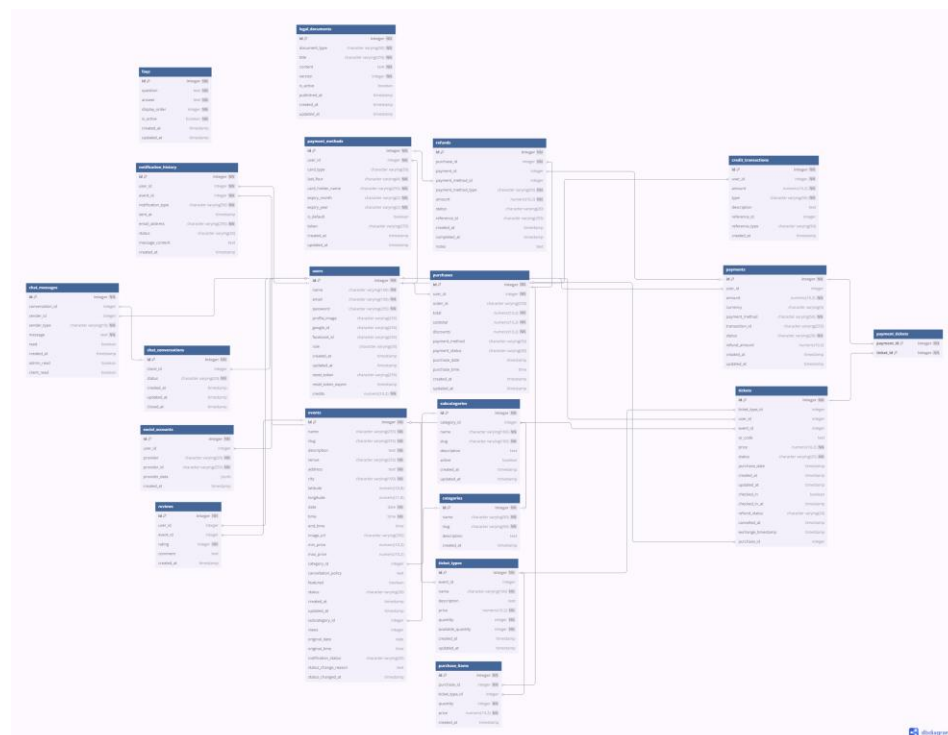
4.2 Modelarea bazei de date

4.2.1 Diagramele entitate-relație

Modelarea bazei de date pentru EventHub a început cu identificarea entităților principale și a relațiilor dintre acestea. Entitatea centrală o reprezintă Events, care stochează informațiile despre evenimente și se conectează cu majoritatea celorlalte entități din sistem. Această entitate conține câmpuri pentru numele evenimentului, descrierea, locația, data și ora, precum și informații despre status și prețuri.

Entitatea Users gestionează informațiile despre utilizatori, incluzând atât utilizatorii obișnuiți, cât și administratorii. Această entitate suportă autentificare prin multiple metode (email/parolă, Google OAuth, Facebook OAuth) și conține informații despre credite disponibile pentru fiecare utilizator. Relația dintre Users și Events se realizează prin intermediul entităților Tickets și Purchases, care gestionează procesul de rezervare și achiziție.

Sistemul de categorizare este implementat prin entitățile Categories și Subcategories, care permit organizarea evenimentelor într-o ierarhie logică. Această structură facilitează navigarea și căutarea evenimentelor de către utilizatori. Relația este de tip one-to-many între Categories și Subcategories, iar fiecare eveniment trebuie să aparțină unei categorii și unei subcategorii.



Sistemul de ticketing este modelat prin entitățile Ticket_Types și Tickets. Ticket_Types definește tipurile de bilete disponibile pentru fiecare eveniment (general, VIP, early bird), cu prețuri și cantități specifice. Entitatea Tickets reprezintă biletele individuale achiziționate de utilizatori și conține informații despre status, QR codes pentru validare și legături către achiziții.

Gestionarea financiară este realizată prin entitățile Purchases, Purchase_Items, Payments și Refunds. Această structură permite tracking-ul detaliat al tuturor tranzacțiilor și facilitează gestionarea rambursărilor și raportărilor financiare. Entitatea Credit_Transactions gestionează sistemul de credite interne al platformei.

4.2.2 Structura tabelelor principale

Tabela Events reprezintă nucleul bazei de date și conține informațiile complete despre evenimente. Câmpurile principale includ id (cheie primară), name (varchar 255), description (text pentru descrieri lungi), venue (varchar 255), address (text), city (varchar 100), date și time pentru programare. Câmpurile latitude și longitude (numeric cu 8, respectiv 11 zecimale) permit integrarea cu serviciile de hartă și localizare.

Câmpul status permite gestionarea stărilor evenimentului (active, inactive, cancelled, rescheduled), iar câmpurile original_date și original_time sunt folosite pentru tracking-ul modificărilor de programare. Câmpurile notification_status și status_changed_at facilitează sistemul automat de notificări pentru participanți.

```
CREATE TABLE public.events (
  id integer NOT NULL,
  name character varying(255) NOT NULL,
  slug character varying(255) NOT NULL,
  description text NOT NULL,
  venue character varying(255) NOT NULL,
  address text NOT NULL,
  city character varying(100) NOT NULL,
  latitude numeric(10,8),
  longitude numeric(11,8),
  date date NOT NULL,
  "time" time without time zone NOT NULL,
  end_time time without time zone,
  image_url character varying(255),
  min_price numeric(10,2),
  max_price numeric(10,2),
  category_id integer,
  cancellation_policy text,
  featured boolean DEFAULT false,
  status character varying(20) DEFAULT 'active'::character varying,
  created_at timestamp with time zone DEFAULT CURRENT_TIMESTAMP,
  updated_at timestamp with time zone DEFAULT CURRENT_TIMESTAMP,
  subcategory_id integer,
  views integer DEFAULT 0,
  original_date date,
  original_time time without time zone,
  notification_status character varying(20) DEFAULT 'pending'::character varying,
  status_change_reason text,
  status_changed_at timestamp with time zone
);
```

Tabela Users stochează informațiile despre utilizatori cu suport pentru autentificare multiplă. Câmpul role (varchar 20) permite diferențierea între utilizatori obișnuiți și administratori. Câmpul credits (numeric 10,2) gestionează sistemul de credite interne pentru rambursări și promoții. Câmpurile reset_token și reset_token_expire facilitează funcționalitatea de resetare a parolei.

Tabela Ticket_Types definește tipurile de bilete disponibile pentru fiecare eveniment. Această structură permite flexibilitate în definirea unor pachete de bilete diferite (general admission, VIP, early bird) cu prețuri și cantități specifice. Câmpul available_quantity este actualizat dinamic pe măsură ce biletele sunt vândute.

Sistemul de achiziții este structurat pe două nivele: Purchases (header-ul comenzii) și Purchase_Items (detaliile produselor). Această structură normalizată permite gestionarea comenzilor cu multiple tipuri de bilete și facilitează raportările și analizele financiare. Tabela Payment_Tickets creează legătura între plăți și biletele specifice achiziționate.

4.2.3 Normalizarea și încadrarea în forma normală a treia

Schema EventHub a fost construită astfel încât fiecare tabel să respecte consecutiv cele trei forme normale și să ajungă, în final, în 3-NF.

1. 1-NF este asigurată prin coloane atomice: nu există liste separate prin virgule sau câmpuri compuse; câmpuri precum venues.latitude și venues.longitude sunt stocate individual, nu într-un singur șir.
2. 2-NF este satisfăcută prin separarea informațiilor „header–detaliu”. De pildă, datele generale ale unei comenzi se află în purchases, iar liniile concrete ale acelei comenzi în purchase_items; astfel, niciun atribut non-cheie nu depinde doar de o parte a cheii compuse.
3. 3-NF este garantată prin eliminarea dependențelor tranzitive:
 - numele și descrierea unei categorii nu apar în subcategories sau events, ci doar cheia străină category_id;
 - prețul unui bilet este păstrat o singură dată în ticket_types, în timp ce tickets conține doar statusul individual al fiecărui bilet;
 - coloane derivate (ex. numărul total de bilete vândute) nu sunt stocate, ci calculate în view-uri.

Prin această normalizare, orice atribut non-cheie depinde exclusiv de cheia primară a propriului tabel, iar modificările (schimbarea prețului, redenumirea unei categorii, anularea unui eveniment) se fac o singură dată, fără riscul de valori divergente în alte tabele.

4.2.4 Indexarea și optimizarea performanței

Pentru asigurarea performanțelor optime, s-au creat indexuri strategici pe câmpurile cel mai frecvent folosite în interogări. Indexul pe events.date optimizează căutările de evenimente pe perioade de timp, iar indexul pe events.category_id accelerează filtrarea pe categorii.

```
CREATE INDEX idx_events_date ON events USING btree (date);  
CREATE INDEX idx_tickets_user_id ON tickets USING btree (user_id);  
CREATE INDEX idx_notification_history_user_event  
ON notification_history USING btree (user_id, event_id, notification_type);
```

Indexurile pe chei străine asigură performanțe optime pentru operațiile de join, în special pentru relațiile frecvent folosite precum users-tickets și events-ticket_types. Indexul compus pe notification_history optimizează verificările pentru dublarea notificărilor.

Optimizarea query-urilor include utilizarea de query-uri parametrizate pentru prevenirea atacurilor SQL injection și implementarea de prepared statements pentru operațiunile frecvente. Structura tabelor a fost proiectată pentru a minimiza numărul de join-uri necesare în operațiunile comune.

4.2.5 Constrângeri și integritatea datelor

Constrângerile de integritate referențială asigură consistența datelor prin chei străine cu reguli specifice de ștergere și actualizare. De exemplu, ștergerea unui utilizator va seta la NULL referințele din tickets dar va șterge complet înregistrările din credit_transactions printr-un CASCADE.

Constrângerile de validare includ verificări pentru câmpuri critice precum rating-urile (între 1 și 5 stele), statusurile evenimentelor (doar valorile permise) și sumele monetare (nu pot fi negative pentru majoritatea operațiunilor).

```
CONSTRAINT reviews_rating_check CHECK ((rating >= 1) AND (rating <= 5))  
CONSTRAINT positive_or_negative_amount CHECK (amount <> 0::numeric)
```

Triggerele

de auditare sunt implementate pentru tracking-ul modificărilor importante, în special pentru tabelele care gestionează informații financiare. Aceste triggere mențin un jurnal de audit pentru toate modificările de prețuri, statusuri și transferuri de credite.

4.3 Arhitectura aplicației

4.3.1 Modelul MVC adaptat

Arhitectura EventHub implementează o variantă adaptată a modelului Model-View-Controller (MVC), optimizată pentru aplicațiile web moderne cu separarea clară între frontend și backend. **Modelele** sunt implementate în backend și reprezintă entitățile din baza de date cu logica asociată pentru manipularea datelor. Fiecare model încapsulează operațiunile CRUD (Create, Read, Update, Delete) pentru entitatea corespunzătoare și asigură validarea datelor la nivelul bazei de date.

Controllerele funcționează ca intermediar între rutele API și serviciile de business, având responsabilitatea de a procesa cererile HTTP, de a valida parametrii de intrare și de a formata răspunsurile. Această separare permite o mai bună organizare a codului și facilitează testarea componentelor individual. Controllerele gestionează și aspectele legate de autentificare și autorizare, asigurându-se că utilizatorii au permisiunile necesare pentru operațiunile solicitate.

View-urile sunt implementate integral în frontend prin componentele React, care interacționează cu backend-ul prin API-uri RESTful. Această separație totală între frontend și backend permite dezvoltarea independentă a celor două părți și facilitează crearea de interfețe multiple pentru același backend (web app, mobile app, admin dashboard).

Nivelul de servicii adaugă un strat suplimentar de abstracție între controllere și modele, încapsulând logica de business complexă. Serviciile gestionează operațiunile care implică multiple entități sau logică de business complicată, cum ar fi procesarea plăților, gestionarea notificărilor sau calcularea statisticilor.

4.3.2 Organizarea codului în servicii și controllere

Structura serviciilor urmează principiile Single Responsibility și Dependency Injection pentru a asigura o organizare modulară și testabilă a codului. EventService gestionează toate operațiunile complexe legate de evenimente, incluzând căutarea cu criterii multiple, calcularea prețurilor și gestionarea statusurilor. PaymentService încapsulează logica de procesare a plăților, integrarea cu gateway-urile de plată externe și gestionarea rambursărilor.

```

export class EventService extends BaseService {
  async getEventById(id) {
    // Fetch event details
    const event = await EventModel.findById(id);

    if (!event) {
      throw new Error('Event not found');
    }

    // Fetch related data
    const ticketTypes = await TicketTypeModel.findById(event.id);
    const reviews = await ReviewModel.findById(event.id);
    const relatedEvents = await EventModel.findRelated(event.category_id, id, 4);

    return { event, ticketTypes, reviews, relatedEvents };
  }
}

```

Controllerele implementează pattern-ul de error handling consistent, folosind middleware-ul `asyncHandler` pentru gestionarea automată a erorilor asincrone. Fiecare controller validează datele de intrare, apelează serviciile corespunzătoare și formatează răspunsurile într-un format consistent pentru frontend.

`BaseService` oferă funcționalități comune pentru toate serviciile, incluzând gestionarea tranzacțiilor bazei de date, logging-ul operațiunilor și implementarea de pattern-uri comune precum `retry logic` pentru operațiunile cu servicii externe.

`AuthService` gestionează aspectele complexe ale autentificării, incluzând integrarea cu OAuth providers, gestionarea token-urilor JWT și implementarea logic-ii de "remember me". Acest serviciu asigură și sincronizarea datelor între conturile sociale și profilurile locale.

4.3.3 Middleware și interceptori

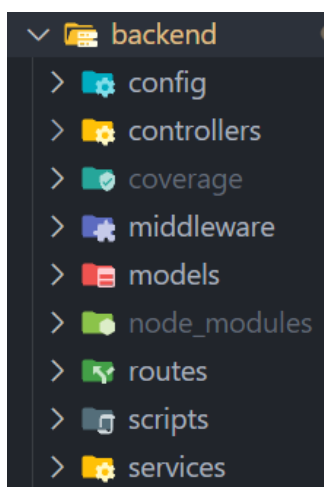
Middleware-ul de autentificare implementează verificarea JWT tokens și integrarea cu Passport.js pentru OAuth. Acest middleware decorează request-urile cu informațiile utilizatorului autentificat și permite controllerelor să acceseze datele utilizatorului fără logică specifică de autentificare.

Error handling middleware asigură un răspuns consistent pentru toate tipurile de erori, categorizând erorile în erori de validare (400), erori de autentificare (401), erori de autorizare

(403) și erori de server (500). Middleware-ul loghează erorile pentru monitoring și debug, dar nu expune informații sensibile către client.

CORS middleware gestionează politicile de Cross-Origin Resource Sharing pentru a permite accesul frontend-ului la API, configurând headers-ele necesare pentru comunicația între diferite domenii în timp ce menține securitatea.

Rate limiting middleware protejează API-ul împotriva atacurilor de tip denial-of-service prin limitarea numărului de cereri per utilizator pe unitatea de timp. Implementarea folosește algoritmi adaptivi care permit mai multe cereri pentru utilizatorii autentificați față de cei anonimi.



4.4 Implementarea securității aplicației

4.4.1 Securitatea autentificării

Sistemul de autentificare implementează multiple straturi de securitate pentru protejarea conturilor utilizatorilor. Hash-uirea parolelor se realizează cu algoritmul bcrypt folosind un cost factor de 10, care oferă un echilibru optim între securitate și performanță. Această metodă protejează împotriva atacurilor de tip rainbow table și brute force prin introducerea unui salt unic pentru fiecare parolă.

Token-urile JWT sunt configurate cu o durată de viață limitată (24 de ore) și includ informații minimale pentru a reduce riscul în cazul compromiterii. Implementarea suportă refresh token-uri pentru sesiunile persistente și invalidarea manuală a token-urilor în cazul logout-ului sau al activităților suspecte.

Sistemul OAuth pentru Google și Facebook este implementat cu verificări stricte ale state parameters pentru prevenirea atacurilor CSRF. Informațiile primite de la providerii OAuth

sunt validate și sanitizate înainte de stocarea în baza de date, iar legătura dintre conturile sociale și profilurile locale se realizează prin email-ul verificat.

Politicile de parolă enforced la nivelul frontend-ului includ cerințe pentru lungime minimă, complexitate și verificarea împotriva parolelor comune. Sistemul implementează și protecție împotriva atacurilor de tip brute force prin limitarea încercărilor de login și introducerea de întârzieri progressive.

4.4.2 Protecția împotriva vulnerabilităților comune

Prevenirea atacurilor CSRF se realizează prin verificarea header-elor Origin și Referer pentru cererile state-changing, combinată cu folosirea token-urilor CSRF pentru operațiunile sensibile. API-ul este proiectat să urmeze principiile REST, unde operațiunile GET sunt idempotente și nu modifică starea sistemului.

Protecția XSS este implementată prin sanitizarea automată a tuturor datelor afișate în interfața utilizator și folosirea principiilor de Content Security Policy (CSP). Framework-ul React oferă protecție nativă împotriva XSS prin escape-uirea automată a valorilor, dar aplicația implementează și validări suplimentare pentru conținutul generat de utilizatori.

Prevenirea SQL injection se realizează prin folosirea exclusivă a query-urilor parametrizate și prepared statements pentru toate interacțiunile cu baza de date. Input-ul utilizatorilor nu este niciodată concatenat direct în query-uri SQL, ci este pasat prin parametri validați și sanitizați.

4.5 Gestionarea sesiunilor și autentificării

4.5.1 Arhitectura sistemului de autentificare

Sistemul de sesiuni implementează o arhitectură hibridă care combină avantajele token-urilor JWT cu flexibilitatea sesiunilor tradiționale. Pentru sesiunile temporare, token-urile sunt stocate în sessionStorage și expiră automat când utilizatorul închide browser-ul. Pentru sesiunile persistente ("Remember me"), token-urile sunt stocate în localStorage cu o durată de viață extinsă.

Fluxul de autentificare începe cu validarea credențialelor utilizatorului (email/parolă sau OAuth), urmată de generarea unui JWT token care conține informații minimale despre utilizator (ID, rol, timestamp). Token-ul este semnat cu o cheie secretă configurată în

variabilele de mediu și include o dată de expirare pentru limitarea impactului în cazul compromiterii.

Strategiile Passport.js implementează trei metode de autentificare: JWT pentru API requests, Google OAuth pentru autentificare socială și Facebook OAuth ca alternativă. Fiecare strategie include logică specifică pentru gestionarea profilurilor noi versus existente și sincronizarea datelor între conturile sociale și profilurile locale.

Middleware-ul de autentificare verifică prezența și validitatea token-urilor pentru toate rutele protejate, decorând request-urile cu informațiile utilizatorului autentificat. Implementarea include logică pentru refresh-ul automat al token-urilor aproape de expirare și invalidarea în cazul detectării unei activități suspecte.

4.5.2 OAuth integration

Integrarea Google OAuth folosește biblioteca passport-google-oauth20 și implementează flow-ul de autorizare complet conform specificațiilor OAuth 2.0. Aplicația solicită permisiuni minimale (profil și email) și implementează verificări pentru email-ul verificat pentru a preveni crearea de conturi cu adrese de email false.

Procesul de linking al conturilor sociale cu profilurile existente se realizează pe baza adresei de email verificate. Când un utilizator se autentifică cu Google și email-ul corespunde unui cont existent, sistemul actualizează profilul cu informațiile sociale fără a crea un cont duplicat. Dacă nu există un cont cu email-ul respectiv, se creează automat un nou profil.

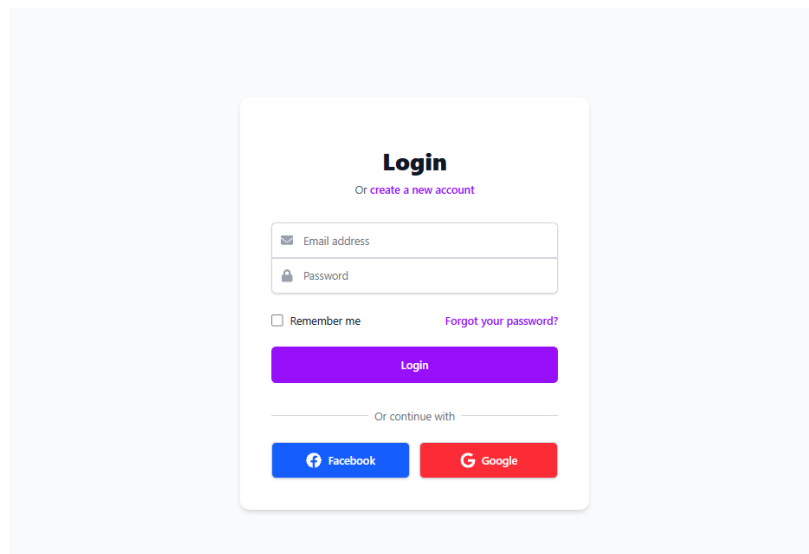
Facebook OAuth implementează o logică similară dar include măsuri suplimentare pentru cazurile în care Facebook nu furnizează o adresă de email. În aceste situații, sistemul generează un email placeholder și solicită utilizatorului să adauge o adresă validă pentru completarea înregistrării.

Securitatea OAuth include validarea parametrilor state pentru prevenirea atacurilor CSRF și verificarea signature-urilor pentru token-urile primite de la provideri. Datele primite sunt sanitizate și validate înainte de stocarea în baza de date, iar secretele aplicației sunt stocate securizat în variabile de mediu.

4.5.3 Gestionarea permisiunilor și rolurilor

Sistemul de roluri implementează o structură simplă dar extensibilă cu rolurile de "user" și "admin". Utilizatorii obișnuiți pot vizualiza evenimente, face rezervări și gestiona propriile bilete, în timp ce administratorii au acces la funcționalitățile de management al platformei și suportul pentru clienți.

Middleware-ul de autorizare verifică rolul utilizatorului pentru rutele administrative și returnează erori 403 Forbidden pentru utilizatorii fără permisiuni adecvate. Implementarea este modulară și permite adăugarea facilă de noi roluri și permisiuni pe măsură ce aplicația se extinde.



4.5.4 Recuperarea parolelor și securitatea conturilor

Sistemul de resetare a parolei implementează un flux securizat care generează token-uri temporare cu durată limitată (1 oră) pentru resetarea parolelor. Token-urile sunt hash-uite înainte de stocarea în baza de date și sunt invalidate automat după utilizare sau expirare.

Email-urile de resetare sunt generate cu template-uri profesionale și includ instrucțiuni clare pentru utilizatori. Link-urile de resetare conțin token-uri unice și redirecționează către o pagină securizată unde utilizatorii pot seta o parolă nouă cu validare în timp real.

Sistemul implementează și notificări către utilizatori pentru orice tentativă de resetare a parolei, chiar dacă cererea nu a fost inițiată de aceștia.

4.6 Integrarea serviciilor externe

4.6.1 Serviciul de email

Configurația Nodemailer implementează conectivitatea cu serverele SMTP pentru trimiterea automată a email-urilor transacționale. Configurația suportă multiple provideri de email (Gmail, SendGrid, Amazon SES) și include mecanisme de fallback în cazul în care serviciul principal nu este disponibil.

Template-urile de email sunt proiectate pentru a fi responsive și profesionale, incluzând confirmări de rezervare, notificări de anulare, reminder-uri pentru evenimente și comunicări administrative. Fiecare template include versiuni HTML și text pentru compatibilitate maximă cu clienții de email.

4.6.2 Serviciile de geolocation

Google Maps API este integrat pentru geocoding-ul automat al adreselor evenimentelor și afișarea hărților interactive. API-ul este folosit pentru conversii automate de la adrese text la coordonate GPS, facilitând căutarea evenimentelor pe baza proximității geografice.

Serviciul de geocoding implementează cache-uire pentru a reduce costurile API și ameliora performanțele. Adresele deja geocodate sunt stocate în baza de date iar cererile duplicate sunt evitate prin verificări preliminare.

Optimizarea costurilor include rate limiting pentru cererile către Google Maps API și utilizarea de batch operations pentru geocoding-ul mai multor adrese simultan. Sistemul monitorizează consumul API și implementează alerte pentru depășirea limitelor de cost.

4.6.3 Serviciile de notificări

Scheduled tasks implementează cu node-cron pentru automatizarea notificărilor și maintenance-ului sistemului. Task-urile includ trimiterea reminder-urilor pentru evenimente (cu 7 zile înainte), cleanup-ul evenimentelor expirate și procesarea automată a rambursărilor.

Sistemul de notificări categorizează mesajele pe tipuri (confirmări, reminder-uri, anulări, promoții) și implementează preferințe granulare pentru utilizatori.

Queue management pentru notificări implementează prioritizare și batch processing pentru optimizarea performanțelor. Notificările urgente (anulări de evenimente) sunt procesate imediat, în timp ce comunicările în masă sunt distribuite pe intervaluri pentru a evita supraîncărcarea serverelor de email.

5 Prezentarea funcționalităților aplicației EventHub

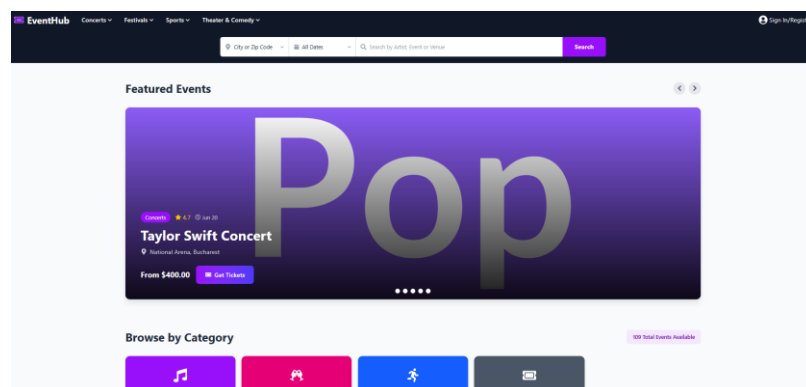
5.1 Interfața utilizatorului (screenshots + UX)

Interfața utilizatorului a aplicației EventHub a fost concepută urmând principiile moderne de User Experience (UX) și User Interface (UI) design, cu accent pe simplitate, intuitivitate și eficiență.

5.1.1 Principiile de design adoptate

Designul EventHub se bazează pe principiile Material Design și pe cele mai bune practici din industria dezvoltării web moderne. Paleta de culori principală utilizează nuanțe de violet și indigo pentru a transmite profesionalismul și creativitatea, în timp ce elementele secundare folosesc o gamă de gri pentru a asigura contrastul necesar și lizibilitatea optimă.

Arhitectura vizuală a aplicației urmează un grid sistem consistent, care permite alinierea perfectă a elementelor și creează o ierarhie vizuală clară. Spațierea între elemente este calculată matematic pentru a asigura o densitate informațională optimă, evitând atât supraîncărcarea vizuală cât și risipa de spațiu.



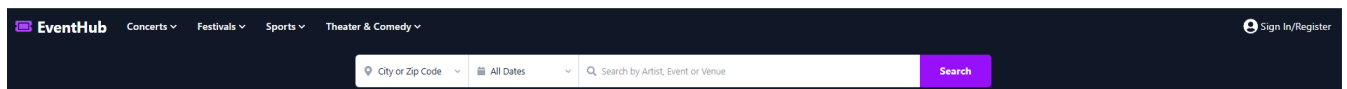
5.1.2 Componenta Header și navigarea principală

Header-ul aplicației reprezintă centrul de comandă al navigației utilizatorului, fiind implementat ca o componentă inteligentă care se adaptează contextului paginii curente. Pe pagina principală, header-ul prezintă un sistem complet de căutare cu trei câmpuri principale: locația evenimentului, intervalul de date și termenii de căutare liberă.

Sistemul de căutare geografică include funcționalitatea de detectare automată a locației utilizatorului prin intermediul API-ului de geolocalizare al browser-ului. Utilizatorii pot alege între introducerea manuală a unei locații sau utilizarea poziției curente, sistema oferind

feedback vizual clar pentru ambele opțiuni. Lista de locații predefinite include principalele orașe din România, facilitând căutarea rapidă.

Componenta de selecție a datelor integrează un calendar personalizat care permite selectarea atât a unei singure date cât și a unui interval temporal. Implementarea acestei funcționalități a necesitat o atenție deosebită la gestionarea timezone-urilor și la formatarea corectă a datelor pentru a evita discrepanțele între frontend și backend.



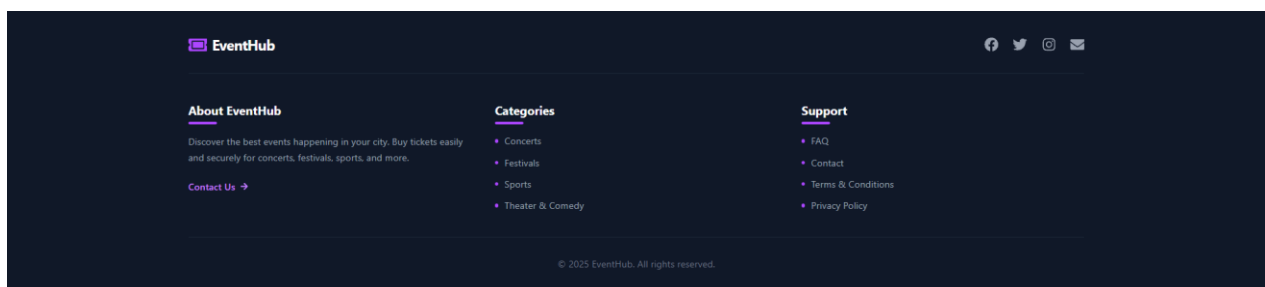
Pe paginile de categorii, header-ul se transformă într-o versiune simplificată, oferind un câmp de căutare rapid poziționat în partea dreaptă. Această adaptare contextual reduce complexitatea vizuală când utilizatorul se află deja într-o secțiune specifică a aplicației, îmbunătățind fluxul de navigare.

Meniul de utilizator implementează un sistem dropdown sofisticat care afișează diferite opțiuni în funcție de rolul utilizatorului. Utilizatorii obișnuiți au acces la profil, bilete și istoric, în timp ce administratorii beneficiază de opțiuni adiționale pentru accesarea dashboard-ului administrativ.

5.1.3 Componenta Footer și structura informațională

Footer-ul aplicației servește ca punct de ancorare pentru informațiile secundare dar importante. Designul său urmează o structură în trei coloane care organizează logic informațiile: despre EventHub, categorii de evenimente, și secțiunea de suport. Fiecare secțiune este clar delimitată vizual prin utilizarea de indicatori colorați și spațiere adecvată.

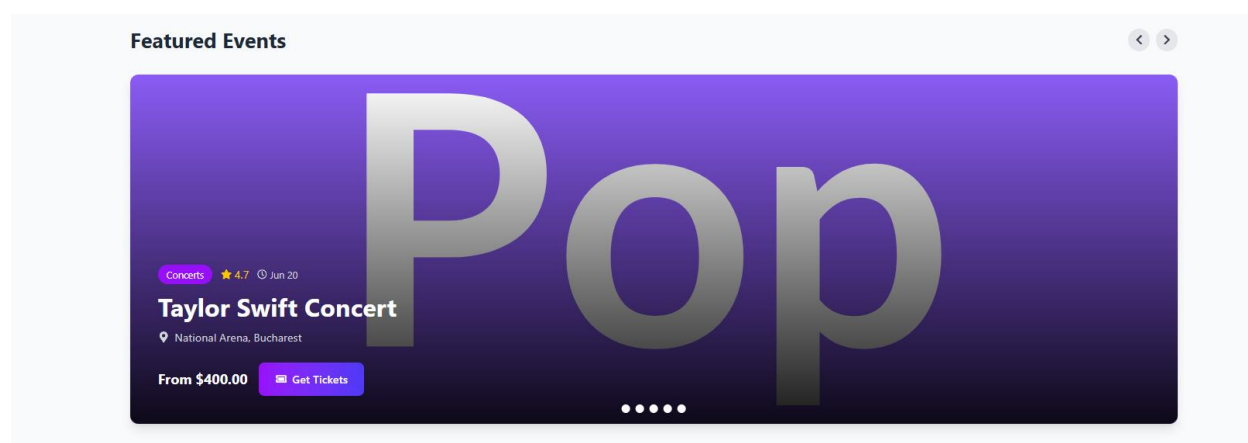
Secțiunea de rețele sociale din footer integrează linkuri către platformele principale, fiecare cu iconițe SVG optimizate pentru încărcare rapidă și scalabilitate perfectă.



5.1.4 Pagina principală și experiența utilizatorului

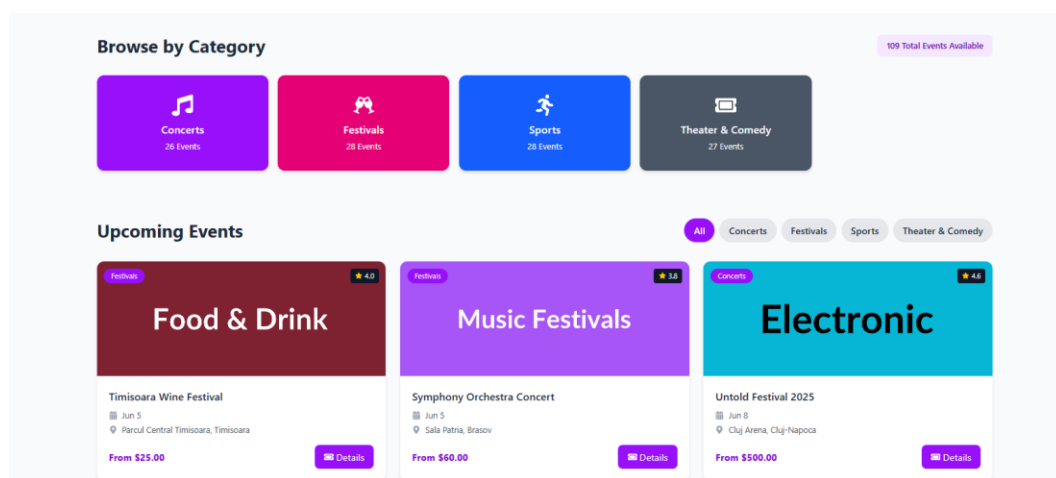
Pagina principală a EventHub funcționează ca o vitrină digitală care prezintă în mod atractiv diversitatea evenimentelor disponibile. Componenta centrală este carousel-ul evenimentelor recomandate, care combină imagini de înaltă calitate cu overlay-uri informative elegante. Sistemul de navigare al carousel-ului include atât controale manuale cât și avansare automată, oferind utilizatorilor flexibilitatea de a explora în ritmul lor.

Algoritmul de selecție a evenimentelor recomandate ia în considerare rating-ul oferit de utilizatori, și relevanța temporală. Acest sistem asigură că utilizatorii văd cele mai atractive și relevante evenimente de la prima vizită.



Secțiunea de categorii utilizează un grid responsive care se adaptează automat la dimensiunea ecranului. Fiecare categorie este reprezentată printr-o cartelă colorată cu iconițe specifice și informații despre numărul de evenimente disponibile. Această reprezentare vizuală facilitează navigarea intuitivă și oferă utilizatorilor o perspectivă rapidă asupra diversității platformei.

Grid-ul evenimentelor viitoare implementează un sistem de filtrare în timp real care permite utilizatorilor să selecteze evenimente dintr-o anumită categorie fără a părăsi pagina principală. Această funcționalitate reduce semnificativ numărul de click-uri necesare pentru a găsi evenimente relevante, îmbunătățind considerabil experiența utilizatorului.

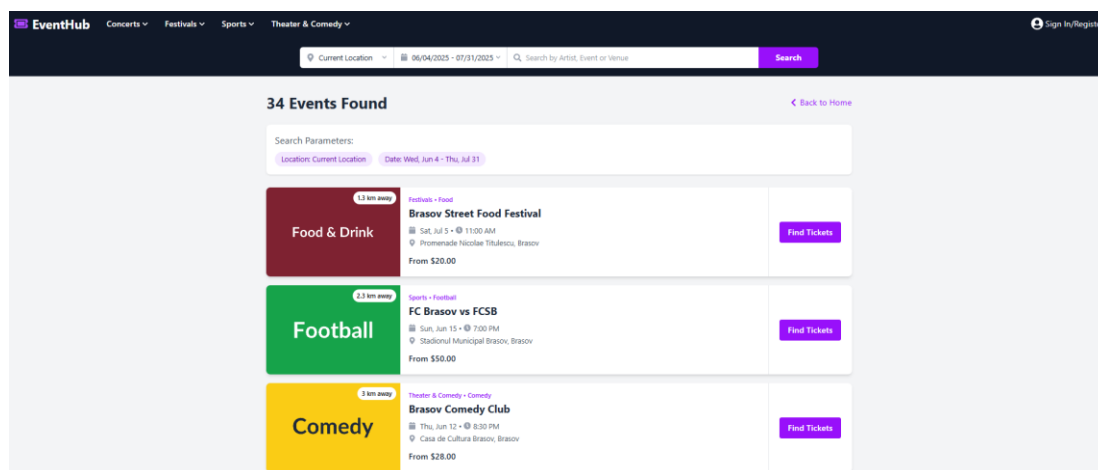


5.1.5 Sistemul de căutare avansată

Componenta SearchFilter reprezintă una dintre cele mai complexe părți ale interfeței utilizatorului, integrând multiple criterii de căutare într-o interfață unificată și intuitivă.

Sistemul de geolocalizare integrează API-ul nativ al browser-ului pentru detectarea automată a poziției utilizatorului, gestionând elegant toate scenariile posibile: permisiune acordată, permisiune refuzată, sau indisponibilitatea serviciului. Feedback-ul vizual pentru fiecare din aceste stări este clar și informativ, ghidând utilizatorul către alternative viabile.

Componenta de selecție a datelor sincronizează perfect cu parametrii URL-ului, permițând funcționalități avansate precum partajarea căutărilor prin link-uri directe și navigarea înapoi cu păstrarea stării.



5.2 Managementul evenimentelor

Sistemul de management al evenimentelor din EventHub constituie nucleul funcțional al aplicației, oferind administratorilor și organizatorilor de evenimente o platformă completă pentru crearea, editarea și monitorizarea evenimentelor.

5.2.1 Arhitectura sistemului de management

Managementul evenimentelor urmează pattern-ul Model-View-Controller adaptat pentru aplicațiile React moderne. Stratul de prezentare este reprezentat de componentele React din directorul frontend, stratul de logică de business este implementat în serviciile backend, iar persistența datelor este gestionată prin intermediul modelelor și al bazei de date PostgreSQL.

Această separare a responsabilităților permite o menținere facilă a codului și o scalabilitate excelentă. Modificările în logica de business nu afectează interfața utilizatorului, iar schimbările în structura bazei de date sunt izolate în stratul de modele.

Sistemul implementează validări comprehensive la toate nivelurile: validări de tip client-side pentru feedback instantaneu, validări server-side pentru securitate, și constrângeri la nivel de bază de date pentru integritatea informațiilor. Această abordare în profunzime asigură că datele sunt întotdeauna coerente și valide.

5.2.2 Interfața de administrare evenimente

Pagina principală de administrare evenimente, implementată în AdminEvents.jsx, oferă o perspectivă completă asupra tuturor evenimentelor din sistem. Interfața adoptă un design tabular optimizat pentru scanarea rapidă a informațiilor, cu funcționalități avansate de sortare, filtrare și căutare.

Sistemul de filtrare permite administratorilor să vizualizeze evenimente după status (active, anulate, reprogramate), să caute după nume sau locație, și să sorteze după diverse criterii relevante.

Event Management

Refresh

+ Add Event

Q

Search events...

Active

Name (A-Z)

✕ Clear (2)

Status: active

Sort: name asc

EVENT	DATE & TIME	CATEGORY	STATUS	TICKETS SOLD	ACTIONS
<div><div>Drama</div><div>A Streetcar Named Desire</div><div>Teatrul National Iasi</div></div>	7/3/2025 7:00 PM	Theater & Comedy Drama	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Rock</div><div>AC/DC Concert</div><div>Cluj Arena</div></div>	6/10/2025 8:00 PM	Concerts Rock	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Pop</div><div>Adele Live</div><div>Opera House</div></div>	6/10/2026 8:00 PM	Concerts Pop	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Electronic</div><div>Alesso DJ Set</div><div>Stadionul Municipal</div></div>	8/15/2025 11:00 PM	Concerts Electronic	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Basketball</div><div>BC Timisoara vs Alba Iulia</div><div>Sala Constantin Jude</div></div>	6/22/2025 6:00 PM	Sports Basketball	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Arts & Culture</div><div>Brasov Art Exhibition</div><div>Muzeul de Artă Brasov</div></div>	9/15/2025 10:00 AM	Festivals Arts & Culture	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Food & Drink</div><div>Brasov Beer Festival</div><div>Parc Central Brasov</div></div>	11/15/2025 2:00 PM	Festivals Food	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Comedy</div><div>Brasov Comedy Club</div><div>Casa de Cultura Brasov</div></div>	6/12/2025 8:30 PM	Theater & Comedy Comedy	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>
<div><div>Arts & Culture</div><div>Brasov Craft Fair</div><div>Centrul Vechi Brasov</div></div>	11/30/2025 10:00 AM	Festivals Arts & Culture	Active	0	<div><div>👁</div><div>✎</div><div>🗑</div><div>📅</div><div>🚩</div></div>

Fiecare eveniment din listă este afișat cu informațiile esențiale: numele, data, locația, categoria, statusul și numărul de bilete vândute. Interfața oferă acțiuni contextuale pentru fiecare eveniment: vizualizare, editare, anulare, reprogramare sau ștergere.

5.2.3 Formularul de creare și editare evenimente

Componenta AdminEventForm reprezintă inima sistemului de management, oferind o interfață comprehensivă pentru introducerea și modificarea detaliilor evenimente. Formularul este organizat în secțiuni logice care ghidează utilizatorul prin procesul de introducere a datelor.

Secțiunea de informații de bază include câmpurile fundamentale precum numele evenimentului, descrierea și slug-ul URL. Sistemul generează automat slug-ul pe baza numelui introdus, dar permite și editarea manuală pentru personalizare avansată.

Secțiunea de dată și oră implementează validări sofisticate pentru a preveni introducerea datelor din trecut sau a combinațiilor invalide de ore. Componentele de tip date picker sunt integrate natural în design, oferind o experiență de utilizare familiar.

Gestionarea locației include nu doar adresa textuală, ci și coordonatele geografice pentru integrarea cu serviciile de hartă. Acest nivel de detaliu permite funcționalități avansate precum căutarea evenimentelor în proximitatea unei locații specifice.

Sistemul de categorizare permite asocierea evenimentelor cu categorii și subcategorii, creând o taxonomie structurată care facilitează organizarea și căutarea. Subcategoriile se actualizează dinamic în funcție de categoria selectată, asigurând consistența datelor.

Edit Event

[← Back to Events](#)

Basic Information

Event Name *

A Streetcar Named Desire

Slug * (Auto-generated from name, can be edited)

a-streetcar-named-desire

Used in URLs. Example: "initiative-concept"

Description *

Classic American drama performance

Event Date *

07/02/2025

Event Time *

19:00

End Time

21:45

Location

Venue Name *

Teatrul National Iasi

Address *

Str. Agatha Barancu Nr. 18

City *

Iasi

Latitude

47.15840000

Longitude

27.60140000

Categorization

Category

Theater & Comedy

Subcategory

Drama

Additional Information

Image URL

https://placehold.co/600x400/AC1095/FFFFFF?text=Drama

Minimum Price (\$)

\$35.00

Maximum Price (\$)

\$85.00

Status

Active

☐ Featured Event

Featured events appear in highlighted sections

Cancellation Policy

Tickets can be refunded up to 7 days before the event date. Within 7 days of the event, no refunds will be issued except in exceptional circumstances. Contact support for assistance with refunds.

Cancel

Update Event

5.2.4 Funcționalități avansate de management

Sistemul include funcționalități specializate pentru situații excepționale precum anularea sau reprogramarea evenimentelor.

Procesul de anulare a unui eveniment declanșează automat o procedură de rambursare pentru toate biletele vândute, notifică participanții prin email și actualizează statusul evenimentului în baza de date. Interfața solicită administratorului să furnizeze un motiv pentru anulare, care va fi comunicat participanților.

Confirm Event Cancellation

Are you sure you want to cancel Iasi Heritage Festival?

This event has 1 tickets sold. All tickets will be automatically refunded.

Cancellation Reason *

Please provide a reason for cancellation (will be shared with attendees)

Close

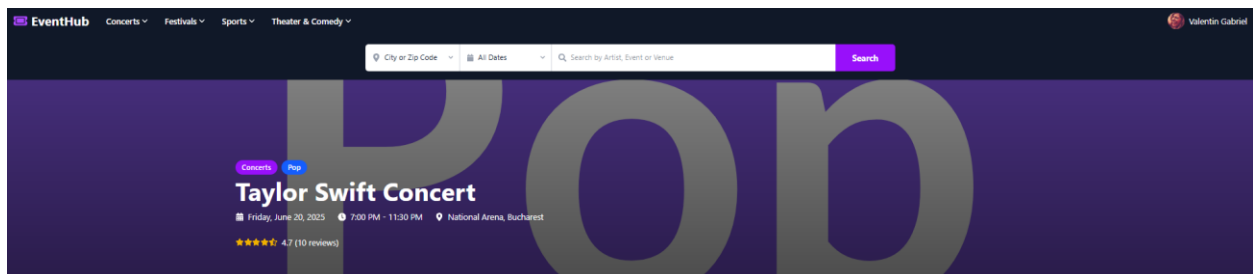
Cancel Event

Reprogramarea evenimentelor necesită introducerea unei noi date și ore, împreună cu motivul schimbării. Sistemul verifică automat disponibilitatea noii date și actualizează toate rezervările existente, păstrând legăturile cu biletele deja vândute.

5.2.5 Afișarea publică a evenimentelor

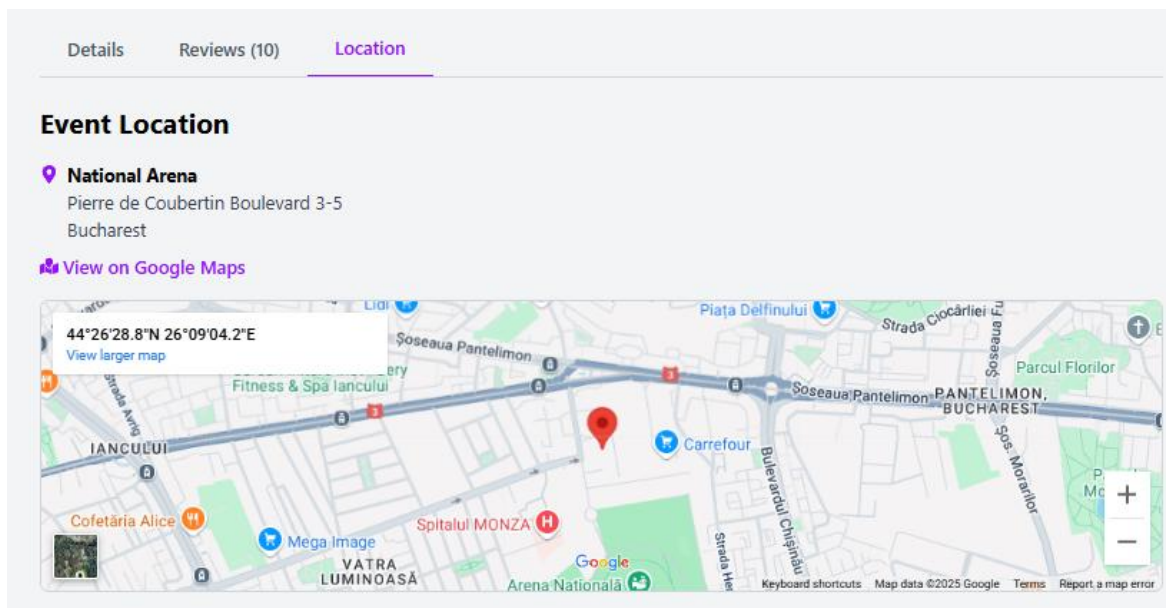
Componenta EventDetails oferă clienților o prezentare completă și atractivă a informațiilor despre eveniment. Design-ul acestei pagini combină elementele vizuale impactante cu organizarea logică a informațiilor pentru a facilita procesul de decizie al utilizatorului.

Header-ul paginii utilizează imaginea evenimentului ca fundal, cu un overlay gradient care asigură lizibilitatea textului suprapus. Informațiile principale precum data, ora și locația sunt prezentate proeminent, iar tag-urile de categorie oferă context suplimentar.



Sistemul de taburi organizează informațiile detaliate în secțiuni ușor de navigat: detalii generale, recenzii și locație. Fiecare tab oferă o perspectivă diferită asupra evenimentului, permițând utilizatorilor să acceseze informațiile relevante pentru ei fără a fi copleșiți de detalii irelevante.

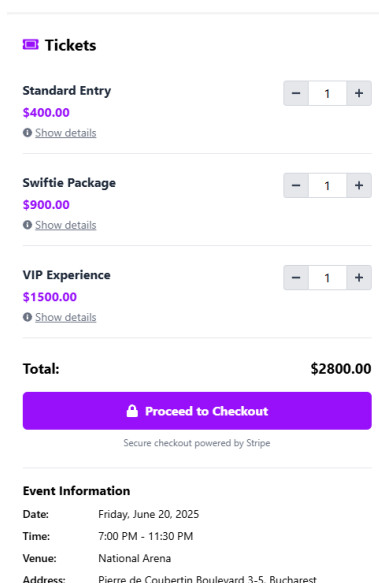
Secțiunea de locație integrează o hartă interactivă care arată poziția exactă a locului evenimentului, împreună cu linkuri către servicii externe de navigație. Această funcționalitate este deosebit de utilă pentru utilizatorii care nu cunosc zona.



5.2.6 Integrarea cu sistemul de bilete

Bara laterală de achiziție bilete reprezintă punctul de conversie al aplicației, unde utilizatorii trec de la explorare la achiziție.

Selectorul de tipuri de bilete prezintă toate opțiunile disponibile cu prețurile respective, permițând utilizatorilor să facă selecții multiple. Calculul totalului se actualizează în timp real, oferind transparență completă asupra costurilor.



5.3 Sistemul de ticketing și rezervări

Sistemul de ticketing al EventHub reprezintă una dintre cele mai complexe și importante componente ale aplicației, fiind responsabil pentru întregul proces de la selectarea biletelor până la validarea acestora la eveniment.

5.3.1 Arhitectura sistemului de bilete

Sistemul de ticketing este construit pe o arhitectură în trei straturi care separă clar responsabilitățile: stratul de prezentare gestionează interacțiunea cu utilizatorul, stratul de logică de business procesează regulile complexe de rezervare și validare, iar stratul de persistență asigură integritatea datelor și consistența tranzacțiilor.

La nivelul bazei de date, sistemul utilizează multiple tabele interconectate pentru a gestiona tipurile de bilete, stocul disponibil, rezervările active și istoricul tranzacțiilor. Modelul de date este optimizat pentru a preveni supravânzarea prin utilizarea de constrangeri la nivel de bază de date și blocări optimiste asupra înregistrărilor critice.


Fiecare eveniment poate avea multiple tipuri de bilete, fiecare cu propriile caracteristici: preț, cantitate totală, cantitate disponibilă, și metadata specifică.

5.3.2 Componenta de selecție bilete

Componenta TicketSelector reprezintă interfața principală prin care utilizatorii interacționează cu sistemul de rezervări. Această componentă implementează o logică sofisticată pentru afișarea opțiunilor de bilete, gestionarea selecțiilor utilizatorului și calcularea prețurilor în timp real.

Interfața de selecție prezintă fiecare tip de bilet într-o formă vizuală clară, cu informații despre preț, disponibilitate și orice restricții specifice. Controalele de cantitate sunt implementate cu validări instantanee care previn selectarea unei cantități mai mari decât cea disponibilă, oferind feedback vizual imediat utilizatorului.

Sistemul de alertă pentru disponibilitatea limitată atrage atenția utilizatorilor atunci când rămân mai puțin de zece bilete disponibile pentru un anumit tip.

 Tickets

Standard Ticket

-

1

+

\$25.00

[Show details](#)

Premium Seating

-

1

+

\$52.50

[Show details](#)

VIP Package

-

1


+

\$80.00

[Show details](#)

Total:

\$157.50

 Proceed to Checkout

Secure checkout powered by Stripe

Event Information

Date:

Sunday, July 20, 2025

Time:

8:30 PM - 10:30 PM

Venue:

Arena Nationala

Address:

Sos. Basarabia Nr. 37-39, Bucharest

5.3.3 Procesul de rezervare și achiziție

Procesul de rezervare este implementat ca o secvență de operațiuni atomice care asigură consistența datelor chiar și în condiții de încărcare ridicată. Când un utilizator selectează biletele dorite și procedează la checkout, sistemul inițiază o rezervare temporară care blochează biletele respective pentru o perioadă limitată.

Această rezervare temporară previne ca alte utilizatori să achiziționeze aceleași bilete în timpul procesului de plată, eliminând situațiile frustrante în care utilizatorul completează toate datele de plată doar pentru a descoperi că biletele nu mai sunt disponibile. Rezervarea expiră automat dacă plata nu este finalizată într-un interval de timp prestabilit.

Sistemul de procesare a plăților este integrat strâns cu managementul stocului pentru a asigura că actualizările de disponibilitate sunt imediate și precise. Odată ce plata este confirmată cu succes, rezervarea temporară se transformă într-o achiziție definitivă, iar biletele sunt generate automat cu coduri QR unice pentru fiecare bilet individual.

1

2

3

ReviewPaymentConfirmation

Review Your Order

Football

Steaua Bucuresti vs Rapid Bucuresti

Sunday, July 20, 2025 at 8:30 PM

Arena Nationala, Bucharest

Ticket Summary

Standard Ticket	\$25.00 each
Quantity: 1	\$25.00
Premium Seating	\$52.50 each
Quantity: 1	\$52.50
VIP Package	\$80.00 each
Quantity: 1	\$80.00
Total:	\$157.50

Cancellation Policy

Tickets can be refunded up to 7 days before the event date. Within 7 days of the event, no refunds will be issued except in exceptional circumstances. Contact support for assistance with refunds.

Back to Event

Proceed to Payment

1

2

3

ReviewPaymentConfirmation

Payment Method

Select Payment Method

Credit/Debit Card

Pay with Credits

Balance: 1203.00 credits

Your Saved Cards

Mastercard **** 4444

Default

Ghita Valentin • Expires 02/27

Discover **** 1117

Radu Rasinar • Expires 02/26

American Express **** 0005

Loren Stanoi • Expires 03/30

Visa **** 1111

Calin Barbu • Expires 03/29

+ Use a new card

Order Summary

Total Amount:

\$157.50

Back to Review

Complete Payment

59

1

Review

2

Payment

3

Confirmation

✓

Payment Successful!

Your tickets have been reserved

Order Number

ORD-5389425-E0A4

Event:

Steaua Bucuresti vs Rapid Bucuresti

Date:

Sunday, July 20, 2025

Time:

8:30 PM

Venue:

Arena Nationala

Amount Paid:

\$157.50

We've sent the tickets to your email address.

You can also view your tickets in your account.

View My Tickets

Back to Home

Football

Steaua Bucuresti vs Rapid Bucuresti

Sunday, July 20, 2025

8:30 PM

Arena Nationala

Your Tickets

VIP Package

\$80.00

Valid

Scan this QR code at the event for entry

Download

Exchange

Cancel

Premium Seating

\$52.50

Valid

Scan this QR code at the event for entry

Download

Exchange

Cancel

Standard Ticket

\$25.00

Valid

Scan this QR code at the event for entry

Download

Exchange

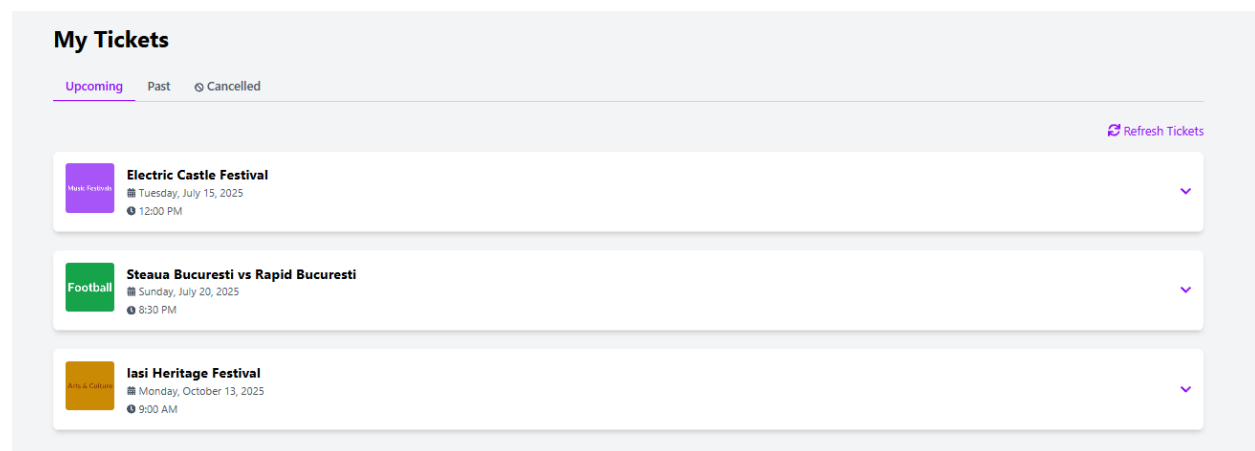
Cancel

5.3.4 Gestionarea biletelor utilizatorilor

Pagina UserTickets oferă utilizatorilor o interfață pentru gestionarea tuturor biletelor lor, organizată în trei categorii distincte: bilete viitoare, bilete din trecut și bilete anulate.

Pentru biletele viitoare, interfața afișează toate informațiile esențiale pentru participarea la eveniment: data, ora, locația, și cel mai important, codul QR necesar pentru intrare. Biletele sunt grupate per eveniment pentru a facilita navigarea atunci când utilizatorul are multiple bilete pentru același eveniment.

Sistemul permite și descărcarea biletelor în format digital .



5.3.5 Sistemul de schimb și rambursare bilete

Una dintre caracteristicile distinctive ale EventHub este sistemul sofisticat de schimb și rambursare a biletelor, care oferă flexibilitate atât utilizatorilor cât și organizatorilor de evenimente. Această funcționalitate este guvernată de politici configurabile care pot fi adaptate pentru fiecare eveniment în parte.

Modalul de schimb bilete prezintă utilizatorilor toate tipurile de bilete disponibile pentru evenimentul respectiv, calculând automat diferența de preț și oferind opțiuni de plată flexibile. Utilizatorii pot plăti diferența folosind credite acumulate în contul lor sau prin metode de plată tradiționale, în funcție de preferințele și circumstanțele lor.

Exchange Ticket

Current Ticket
Event: Electric Castle Festival
Ticket Type: General Pass
Price: \$350.00

Available Ticket Types

Premium Pass \$550.00
Enhanced festival experience with fast lane entry and premium camping
⬆️ You'll pay \$200.00 more

VIP Experience \$750.00
All-inclusive package with backstage access, exclusive viewing areas and premium accommodations

Payment Method

Pay with Credits
Balance: 1203.00 credits

Pay with Card

Cancel

Confirm Exchange

Procesul de rambursare este automatizat într-o măsură considerabilă, cu validări automate care verifică eligibilitatea în funcție de politica de anulare a evenimentului și timpul rămas până la începerea acestuia. Sistemul gestionează automat calcularea și procesarea rambursărilor, actualizând instantaneu disponibilitatea biletelor pentru a permite revânzarea lor.

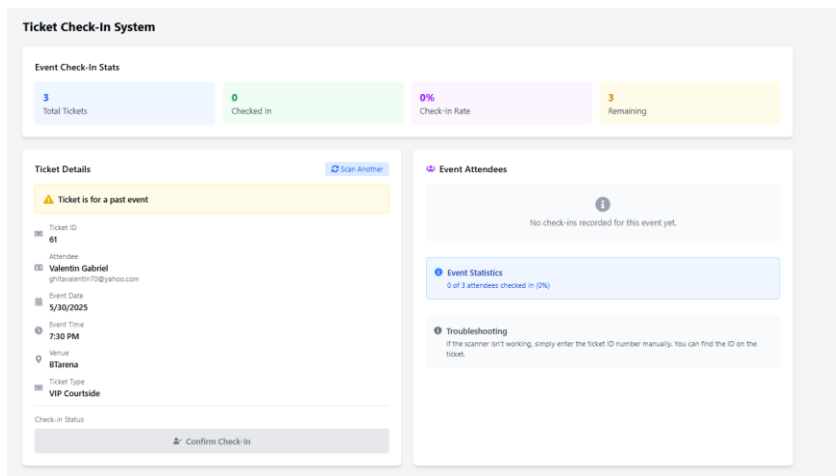
Integrarea cu sistemul de credite permite utilizatorilor să primească rambursări sub formă de credite aplicație, care pot fi folosite pentru achiziții viitoare.

5.3.6 Validarea și check-in-ul biletelor

Sistemul de validare a biletelor utilizează coduri QR criptografice pentru a preveni falsificarea și utilizarea neautorizată. Fiecare bilet generat conține un hash unic care încorporează informații despre bilet, eveniment și utilizator, făcând practic imposibilă clonarea sau falsificarea.

Procesul de check-in poate fi realizat prin interfața web administrativă. Scanarea codului QR validează instantaneu autenticitatea biletului și marchează utilizarea acestuia pentru a preveni intrările multiple cu același bilet. De asemenea, dacă camera organizatorilor nu e funcțională, utilizatorii pot arăta ID-ul biletului primit pe email.

62



Sistemul păstrează un istoric complet al tuturor operațiunilor de check-in, incluzând timestamp-uri precise și informații despre persoana care a efectuat validarea.

5.4 Sistemul de plăți

Sistemul de plăți din EventHub reprezintă o componentă critică care orchestrează întregul proces de monetizare a platformei, de la selectarea biletelor până la confirmarea finală a tranzacției și generarea biletelor digitale.

5.4.1 Arhitectura sistemului de plăți

Sistemul de plăți este construit pe o arhitectură modulară care separă clar responsabilitățile între componentele frontend și backend. La nivelul frontend, procesul de checkout este organizat în trei etape distincte: revizuirea comenzii, procesarea plății și confirmarea finală.

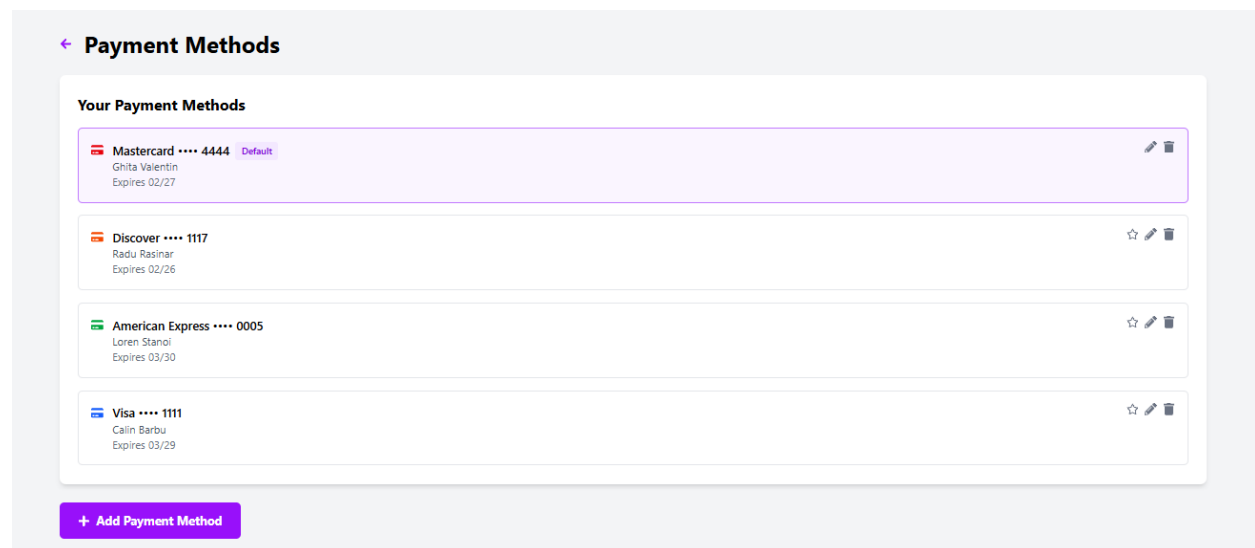
Backend-ul implementează un sistem robust de servicii care gestionează validările, procesarea plăților, crearea biletelor și integrarea cu serviciile externe. Arhitectura bazată pe tranzacții asigură că toate operațiunile sunt atomice - fie se execută complet cu succes, fie se anulează în totalitate, evitând stările inconsistente.

Sistemul suportă multiple metode de plată, inclusiv carduri de credit/debit tradiționale și un sistem intern de credite care funcționează ca o monedă virtuală.

5.4.2 Gestionarea metodelor de plată

Sistemul implementează funcționalități avansate pentru salvarea și gestionarea metodelor de plată, îmbunătățind semnificativ experiența utilizatorului pentru achizițiile viitoare. Utilizatorii pot salva multiple carduri și pot desemna unul ca metoda de plată implicită.

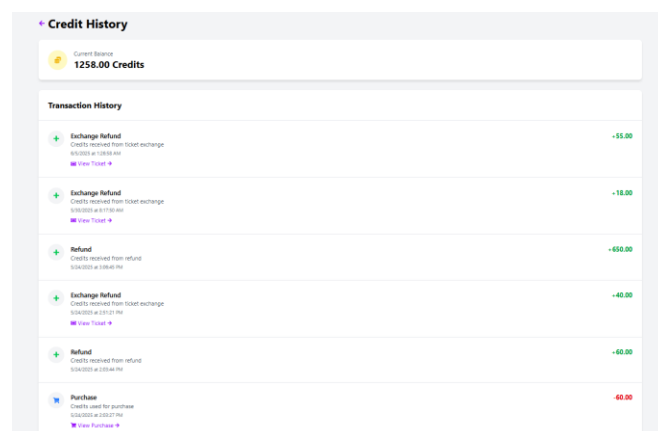
Componenta PaymentMethods oferă o interfață completă pentru adăugarea, editarea și ștergerea metodelor de plată salvate. Validările implementate asigură că doar informații corecte sunt acceptate, cu verificări în timp real pentru numerele de card, datele de expirare și codurile CVV.



5.4.3 Sistemul intern de credite

Una dintre caracteristicile distinctive ale EventHub este sistemul sofisticat de credite care funcționează ca o monedă virtuală în cadrul platformei. Utilizatorii pot acumula credite prin downgrade-uri de bilete.

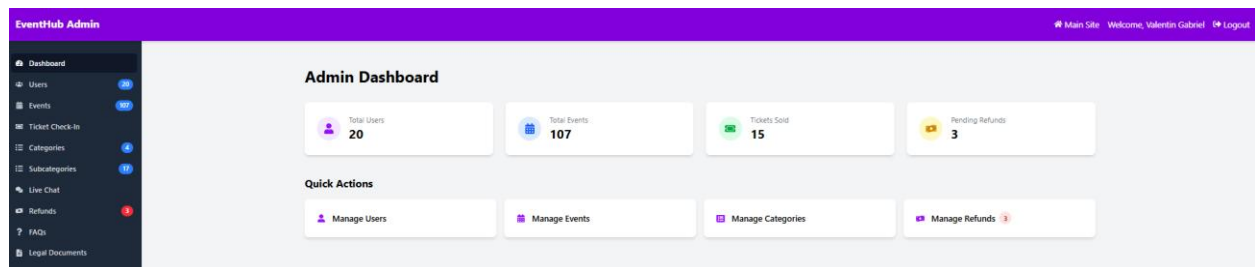
Pagina CreditHistory oferă utilizatorilor o perspectivă completă asupra activității lor de credite, cu detalii despre fiecare tranzacție, motivele pentru care creditele au fost acordate sau deduse, și linkuri către biletele sau achizițiile asociate.



Sistemul de credite facilitează și procesul de schimb al biletelor, permițând utilizatorilor să facă upgrade sau downgrade la diferite tipuri de bilete cu plata sau primirea diferenței de preț în credite.

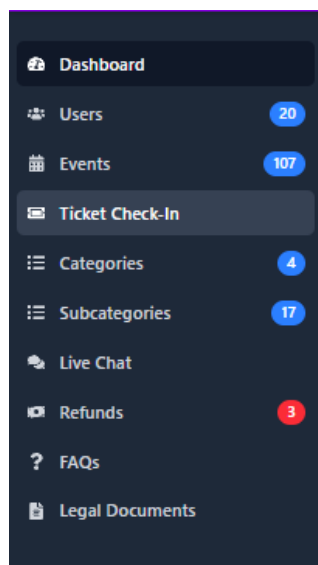
5.5 Dashboard-ul administrativ

Dashboard-ul administrativ reprezintă nucleul sistemului de management al aplicației EventHub, oferind administratorilor un control complet asupra platformei prin intermediul unei interfețe intuitive și funcționale.



























Arhitectura dashboard-ului este construită pe principiul separării responsabilităților, utilizând un sistem ierarhic de componente React care asigură modularitatea și reutilizabilitatea codului. Componenta principală AdminLayout servește drept container pentru întreaga experiență administrativă, implementând un layout responsive cu sidebar navigational și o zonă de conținut dinamică

Sistemul de autentificare și autorizare pentru zona administrativă este implementat prin intermediul componentei AdminRoute, care verifică nu doar validitatea utilizatorului autentificat, ci și privilegiile administrative necesare pentru accesarea funcționalităților sensibile.











Dashboard-ul principal afișează o panoramă comprehensivă a activității platformei prin intermediul cardurilor de statistici care prezintă indicatori cheie de performanță. Aceste metrice includ numărul total de utilizatori înregistrați, evenimentele active și programate, biletele vândute și cererile de rambursare în așteptare. Implementarea acestor statistici se bazează pe serviciul StatisticsService care agregă datele din baza de date și le prezintă într-un format accesibil și actualizat în timp real.

Gestionarea utilizatorilor constituie una dintre funcționalitățile centrale ale dashboard-ului, permițând administratorilor să monitorizeze și să controleze accesul la platformă. Interfața de management a utilizatorilor oferă o vizualizare tabelară completă a tuturor conturilor înregistrate, incluzând informații despre utilizatori și rolurile atribuite. Administratorii pot modifica privilegiile utilizatorilor, transformând conturi standard în conturi administrative sau invers, în funcție de necesitățile organizaționale.

User Management					Refresh
<input type="text" value="Search users by name or email..."/>					
USER	EMAIL	ROLE	CREATED AT	ACTIONS	
 Valy G	valighita472@yahoo.com	 Admin	5/23/2025		
 Radu Stefanescu	radu.blogger@events.ro	 User	5/22/2025		
 Sophie Laurent	sophie.laurent@france.fr	 User	5/22/2025		
 Constantin Mihai	constantin.mihai@senior.com	 User	5/22/2025		
 Ana Georgescu	ana.georgescu@student.ro	 User	5/22/2025		
 David Martinez	david.martinez@proton.me	 User	5/22/2025		

Managementul categoriilor și subcategoriilor evenimentelor reprezintă un alt pilonul esențial al dashboard-ului administrativ. Interfața de gestionare a categoriilor suportă operațiuni CRUD complete, incluzând crearea de noi categorii, editarea celor existente și ștergerea categoriilor care nu mai sunt relevante.

Category Management				Refresh	+ Add Category
NAME	SLUG	DESCRIPTION	ACTIONS		
Concerts	concerts	Live concerts from all music genres			
Festivals	festivals	Music, film, art and cultural festivals			
Sports	sports	Local and international sporting events			
Theater & Comedy	theater-comedy	Theater performances, stand-up comedy and imp...			

Sistemul de validare implementat în gestionarea categoriilor asigură consistența datelor prin verificarea unicității denumirilor și a slug-urilor URL-friendly.

Modulul de gestionare a rambursărilor integrează funcționalități avansate pentru procesarea cererilor de restituire a banilor, oferind administratorilor controlul complet asupra fluxului financiar al platformei. Interfața permite vizualizarea detaliată a tuturor cererilor de rambursare, cu informații comprehensive despre utilizatori, evenimente, sume implicate și motivele solicitărilor. Sistemul de filtrare și căutare facilitează identificarea rapidă a cererilor după diverse criterii, incluzând statusul procesării, perioada de timp și valorile financiare.

Refund Management

✖ Process Pending Refunds

🔄 Refresh

🔍 Search by user, event or ticket ID...

All Statuses

Ticket ID	User	Event	Price	Cancelled At	Status	Actions
#67	Valentin Gabriel ghitavalentin70@yahoo.com	Kendrick Lamar Performance Standard	\$330.00	5/24/2025, 1:49:01 PM	requested	🔄 ✅ ❌
#68	Valentin Gabriel ghitavalentin70@yahoo.com	Ed Sheeran Concert Premium Ticket	\$600.00	5/24/2025, 1:17:01 PM	requested	🔄 ✅ ❌
#63	Valentin Gabriel ghitavalentin70@yahoo.com	Untold Festival 2025 VIP Pass	\$1200.00	5/24/2025, 12:46:31 PM	requested	🔄 ✅ ❌
#74	Valentin Gabriel ghitavalentin70@yahoo.com	Linkin Park Reunion Premium	\$650.00	5/24/2025, 3:05:45 PM	completed	No actions available
#73	Valentin Gabriel ghitavalentin70@yahoo.com	Tosca Opera Performance	\$60.00	5/24/2025, 2:03:37 PM	completed	No actions available
#72	Valentin Gabriel ghitavalentin70@yahoo.com	Metallica Concert General Admission	\$350.00	5/24/2025, 1:49:57 PM	completed	No actions available
#70	Valentin Gabriel ghitavalentin70@yahoo.com	Cluj Food Festival VIP Experience	\$15.00	5/24/2025, 1:34:47 PM	completed	No actions available
#62	Valentin Gabriel ghitavalentin70@yahoo.com	U-BT Cluj vs Dinamo Bucuresti Lower Level	\$80.00	5/21/2025, 5:20:42 PM	completed	No actions available
#53	Valentin Gabriel ghitavalentin70@yahoo.com	David Guetta Live Standard Ticket	\$250.00	5/21/2025, 4:43:53 PM	completed	No actions available
#54	Valentin Gabriel ghitavalentin70@yahoo.com	David Guetta Live VIP Access	\$800.00	5/21/2025, 4:42:40 PM	completed	No actions available
#51	Valentin Gabriel ghitavalentin70@yahoo.com	Taylor Swift Concert VIP Experience	\$1500.00	5/21/2025, 4:02:45 PM	completed	No actions available

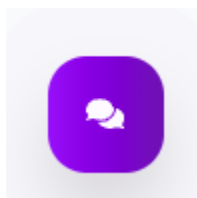
Fluxul de aprobare a rambursărilor implementează un sistem în mai multe etape care asigură atât securitatea financiară, cât și transparența procesului. Administratorii pot actualiza statusul cererilor de rambursare prin intermediul unor acțiuni clare și intuitive, marcând cererile ca fiind în procesare, completate cu succes, eșuate din motive tehnice sau respinse pe baza unor criterii stabilite.

Funcționalitatea de procesare automată a rambursărilor pendente reprezintă o inovație importantă care reduce sarcina administrativă și accelerează timpii de răspuns pentru utilizatori. Sistemul permite configurarea unor praguri temporale personalizabile, după care rambursările în status de procesare sunt finalizate automat, asigurând un flux financiar predictibil și eficient.

5.6 Funcționalități în timp real

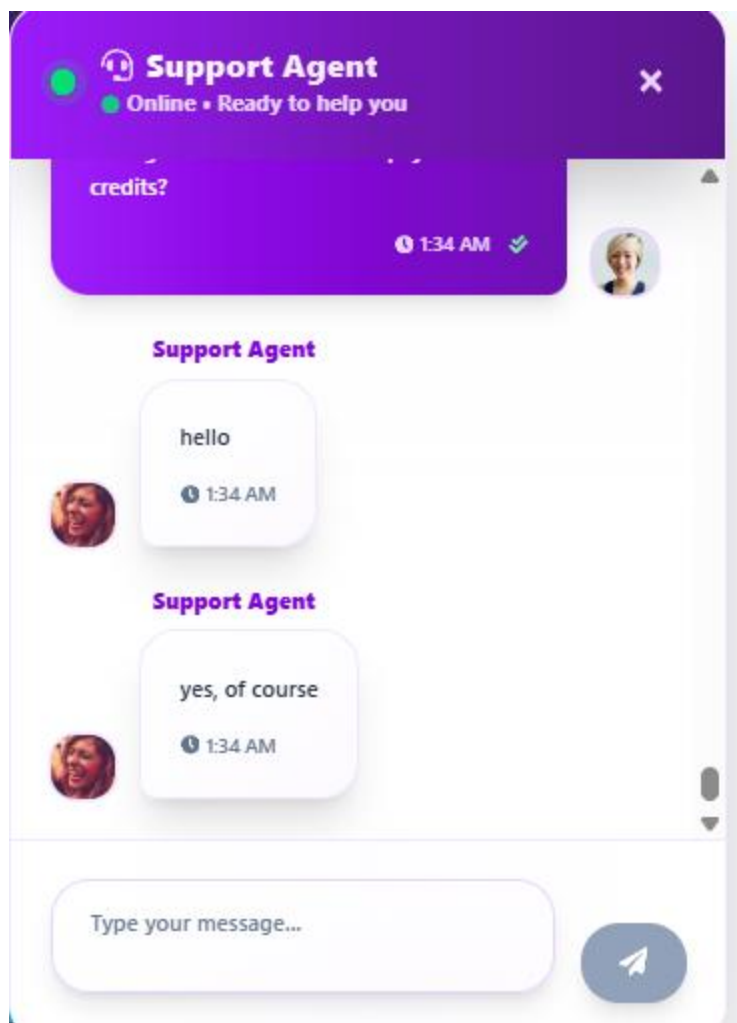
Implementarea funcționalităților în timp real în cadrul platformei EventHub marchează o evoluție semnificativă către standardele moderne de interacțiune digitală, oferind utilizatorilor o experiență dinamică și responsivă. Aceste capabilități sunt construite pe o arhitectură robustă care utilizează tehnologia WebSocket prin intermediul bibliotecii Socket.IO, asigurând comunicarea bidirecțională instantanee între clienți și server.

Fundamentul arhitectural al sistemului în timp real se bazează pe o implementare strategică care separă clar responsabilitățile între gestionarea conexiunilor, procesarea mesajelor și sincronizarea stării aplicației. Contextul SocketContext servește drept punct central de management pentru toate conexiunile WebSocket, oferind o interfață unificată pentru emisiunea și recepționarea evenimentelor în timp real. Această abordare centralizată facilitează mentenanța codului și asigură consistența comportamentului în întreaga aplicație.



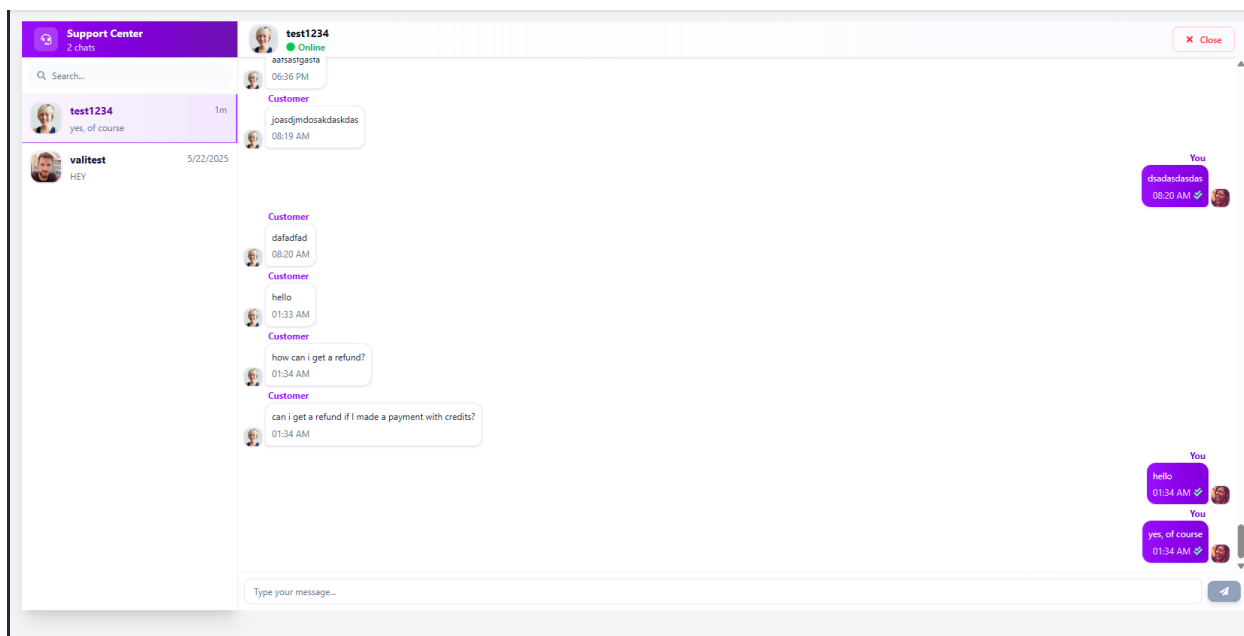
Sistemul de chat live constituie piesa de rezistență a funcționalităților în timp real, oferind o cale directă de comunicare între utilizatori și echipa de suport.

Arhitectura de messaging implementează un sistem sofisticat de gestionare a conversațiilor care permite atribuirea automată și manuală a chat-urilor către agenții de suport disponibili. Contextul ChatContext coordonează întreaga experiență de chat, gestionând starea conversațiilor active, istoricul mesajelor și indicatorii de prezență.



Funcționalitatea de typing indicators oferă utilizatorilor feedback vizual instantaneu despre activitatea interlocutorului. Implementarea acestor indicatori utilizează un sistem de evenimente temporale care gestionează automat afișarea și ascunderea indicatorilor în funcție de activitatea reală de tastare.

Sistemul de prezență online implementat în cadrul funcționalităților de chat oferă transparență completă asupra disponibilității agenților de suport și a statusului utilizatorilor.



Interfața administrativă pentru gestionarea chat-urilor reprezintă un instrument sofisticat care permite agenților de suport să gestioneze eficient multiple conversații simultane. Design-ul acestei interfețe prioritizează claritatea vizuală și eficiența operațională, oferind o prezentare organizată a tuturor conversațiilor active, cu informații relevante despre utilizatori, statusul conversațiilor și prioritatea cererilor. Sidebar-ul de conversații implementează un sistem de sortare inteligentă care prioritizează conversațiile cu mesaje necitite și cele mai recent actualizate.

Funcționalitatea de căutare și filtrare în interfața administrativă permite agenților să identifice rapid conversațiile specifice într-un volum potențial mare de interacțiuni. Sistemul suportă căutări după numele utilizatorului, adresa de email, conținutul mesajelor și alte criterii relevante.

Sistemul de notificări în timp real extinde funcționalitatea de chat către alte aspecte ale platformei, oferind actualizări instantanee despre evenimente critice precum rezervări noi, cancelări, modificări de program și alte activități relevante. Aceste notificări sunt implementate prin intermediul aceluiași sistem WebSocket, asigurând consistența tehnologică și fiabilitatea livrării mesajelor.

Gestionarea mesajelor necitite implementează un sistem sofisticat de tracking care diferențiază între mesajele citite de administratori și cele citite de utilizatori. Această funcționalitate utilizează conceptul de read receipts (confirmări de citire) pentru a oferi transparență completă asupra statusului comunicării. Indicatorii vizuali distinctivi comunică clar când mesajele au fost livrate sau au fost citite de către destinatar.

6 Concluzii și perspective de dezvoltare

6.1 Rezumatul lucrării și obiectivele atinse

Această lucrare de disertație a prezentat dezvoltarea și implementarea unei platforme web moderne pentru managementul evenimentelor, denumită EventHub, care abordează complexitatea crescândă a industriei evenimentelor contemporane. Proiectul s-a concentrat pe crearea unei soluții tehnologice comprehensive care să faciliteze organizarea, promovarea și gestionarea evenimentelor de diverse tipuri și magnitudini, răspunzând astfel nevoilor actuale ale organizatorilor și participanților.

Platforma EventHub a fost concepută ca un ecosistem digital integrat care unifică toate aspectele esențiale ale managementului evenimentelor într-o singură interfață coerentă și eficientă. De la descoperirea și rezervarea biletelor până la procesarea plăților și furnizarea suportului în timp real, sistemul oferă o experiență de utilizare fluidă care satisface așteptările utilizatorilor moderni în ceea ce privește digitalizarea serviciilor.

Implementarea tehnologică a urmărit standardele moderne de dezvoltare web, utilizând React 19 pentru frontend și Node.js cu Express pentru backend, asigurând astfel performanța și scalabilitatea necesare pentru o platformă comercială viabilă. Integrarea PostgreSQL ca sistem de management al bazelor de date a furnizat fundația solidă necesară pentru gestionarea volumelor mari de date tranzacționale și informaționale specifice domeniului evenimentelor.

Sistemul de autentificare și autorizare implementat oferă securitatea necesară pentru protejarea datelor sensibile ale utilizatorilor și a informațiilor financiare, respectând în același timp principiile moderne de securitate web și conformitatea cu regulamentele de protecție a datelor. Integrarea OAuth cu provideri majori precum Google și Facebook facilitează procesul de înregistrare și autentificare, reducând barierele de intrare pentru utilizatorii noi.

Funcționalitatea de căutare și descoperire a evenimentelor implementată răspunde nevoii utilizatorilor de a identifica rapid evenimentele relevante într-o ofertă potențial vastă. Sistemul de filtrare avansată și algoritmi de recomandare contribuie la îmbunătățirea experienței utilizatorului și la creșterea ratei de conversie din vizitatori în participanți la evenimente.

Sistemul de ticketing dezvoltat adresează complexitatea specifică rezervărilor pentru evenimente, incluzând gestionarea diferitelor tipuri de bilete, politicile de preturi dinamice și

mecanismele de validare și check-in. Implementarea codurilor QR pentru bilete oferă o soluție modernă și eficientă pentru controlul accesului la evenimente, reducând timpii de așteptare și îmbunătățind experiența participanților.

Platforma de plăți integrată asigură procesarea securizată a tranzacțiilor financiare, suportând multiple metode de plată. Sistemul de management al rambursărilor automatizează procesele administrative complexe, reducând sarcina operațională și îmbunătățind timpul de răspuns pentru solicitările utilizatorilor.

6.2 Contribuții și inovații

Lucrarea de față aduce contribuții semnificative în domeniul managementului digital al evenimentelor prin dezvoltarea unei soluții tehnologice inovatoare care integrează funcționalități avansate într-o platformă coerentă și scalabilă. Principala contribuție constă în demonstrarea practică a modului în care tehnologiile web moderne pot fi orchestrate pentru a aborda complexitatea specifică industriei evenimentelor.

Una dintre inovațiile cheie ale proiectului o reprezintă implementarea unui sistem de chat în timp real integrat nativ în platforma de evenimente, oferind suport instantaneu pentru utilizatori în toate etapele experienței lor, de la descoperirea evenimentelor până la participarea efectivă.

Arhitectura modulară dezvoltată pentru EventHub demonstrează o abordare sofisticată a separării responsabilităților în dezvoltarea aplicațiilor web complexe. Utilizarea contextelor React pentru gestionarea stării globale, combinată cu implementarea de servicii backend specializate, creează un model de arhitectură care facilitează mentenanța, testarea și extinderea viitoare a platformei.

Sistemul de management al rambursărilor implementat introduce elemente de automatizare inteligentă care reduc semnificativ intervențiile manuale necesare pentru procesarea cererilor de restituire. Algoritmii dezvoltați pentru procesarea automată a rambursărilor după criterii temporale configurabile reprezintă o inovație care poate fi adaptată și în alte domenii cu nevoi similare de automatizare a proceselor administrative.

Implementarea indicatorilor de prezență online și a funcționalităților de typing în sistemul de chat demonstrează atenția la detaliile de experiență utilizator care diferențiază platformele moderne de succes.

Integrarea serviciilor de geocoding pentru evenimentele cu locație fizică oferă o valoare adăugată substanțială pentru utilizatori, facilitând descoperirea evenimentelor în proximitatea lor geografică și îmbunătățind relevanța recomandărilor.

Sistemul de categorii și subcategorii implementat oferă flexibilitatea necesară pentru organizarea logică a unei game diverse de evenimente, de la conferințe profesionale până la evenimente culturale și de divertisment.

Dashboard-ul administrativ dezvoltat integrează funcționalități avansate de monitorizare și management care oferă operatorilor platformei controlul complet asupra tuturor aspectelor sistemului. Implementarea de statistici în timp real și instrumente de management ale utilizatorilor demonstrează o înțelegere profundă a nevoilor operaționale specifice platformelor de evenimente.

6.3 Limitări ale soluției actuale

În ciuda succesului în atingerea obiectivelor propuse, implementarea actuală a platformei EventHub prezintă anumite limitări care trebuie recunoscute și adresate în dezvoltările viitoare. Aceste limitări nu diminuează valoarea soluției dezvoltate, ci oferă direcții clare pentru îmbunătățiri ulterioare și extinderea funcționalității platformei.

Una dintre limitările principale ale implementării actuale constă în lipsa unui sistem complet de analiză și raportare pentru organizatorii de evenimente. Deși platforma colectează date valoroase despre comportamentul utilizatorilor, participarea la evenimente și tendințele de rezervare, aceste informații nu sunt încă procesate și prezentate într-o formă care să faciliteze luarea deciziilor strategice de către organizatori.

Integrarea cu sisteme externe de management al evenimentelor sau cu platforme de marketing digital rămâne limitată, reducând atractivitatea soluției pentru organizatori care utilizează deja ecosisteme tehnologice complexe. Dezvoltarea de API-uri standardizate și conectori pentru sisteme populare ar putea îmbunătăți semnificativ adoptarea platformei.

Funcționalitatea de căutare, deși eficientă pentru volumele actuale de date, nu implementează încă algoritmi avansați de căutare full-text sau tehnologii de indexare specializate care ar fi necesare pentru performanțe optime cu zeci de mii de evenimente în baza de date.

Sistemul de management al conținutului pentru organizatori nu oferă încă instrumente avansate de editare și formatare pentru descrierile evenimentelor, limitând capacitatea organizatorilor de a crea prezentări atractive și profesionale pentru evenimentele lor.

Lipsa unei aplicații mobile native reprezintă o limitare semnificativă în contextul actual în care majoritatea utilizatorilor accesează serviciile digitale în primul rând prin dispozitive mobile.

6.4 Perspective de dezvoltare viitoare

Evoluția viitoare a platformei EventHub poate fi orientată către multiple direcții strategice care să consolideze poziția acesteia ca soluție de referință în domeniul managementului digital al evenimentelor.

Un exemplu ar putea fi Implementarea de algoritmi de inteligență artificială pentru recomandarea personalizată de evenimente care ar reprezenta o oportunitate majoră de îmbunătățire a experienței utilizatorului. Acești algoritmi ar putea analiza comportamentul istoric al utilizatorilor, preferințele declarate și tendințele demographice pentru a sugera evenimente relevante, crescând astfel ratele de participare și satisfacția utilizatorilor.

Dezvoltarea unei aplicații mobile native pentru iOS și Android ar extinde semnificativ accesibilitatea platformei și ar permite implementarea de funcționalități specifice mobile precum notificări push geotargeted, integrare cu calendarul dispozitivului și capacități offline pentru vizualizarea biletelor și informațiilor despre evenimente.

Extinderea sistemului de analiză pentru a include dashboard-uri comprehensive pentru organizatori, cu metrice detaliate despre performanța evenimentelor, demografia participanților și tendințele de rezervare, ar transforma platforma într-un instrument strategic de business intelligence pentru industria evenimentelor.

Implementarea unui marketplace pentru servicii conexe evenimentelor, precum catering, decorațiuni, echipamente audio-video sau servicii de fotografie, ar putea crea un ecosistem complet pentru organizarea evenimentelor, generând fluxuri de venit suplimentare și îmbunătățind valoarea oferită utilizatorilor.

Dezvoltarea de capabilități de streaming live integrate ar permite platformei să se adapteze la tendințele actuale către evenimente hibride care combină participarea fizică cu cea virtuală.

Extinderea sistemului de plăți pentru a suporta criptomonede și soluții de plată alternative emergente ar putea atrage segmente noi de utilizatori și ar poziționa platforma ca fiind la vârful inovației tehnologice în domeniu.

Dezvoltarea de instrumente avansate de management al capacității și de optimizare a prețurilor în timp real, bazate pe algoritmi de machine learning, ar putea maximiza veniturile

organizatorilor și ar oferi utilizatorilor prețuri dinamice care reflectă cererea reală pentru evenimente.

6.5 Concluzii finale

Dezvoltarea platformei EventHub a arătat că o abordare modernă, „all-in-one”, poate schimba radical felul în care sunt organizate și gestionate evenimentele. Am reușit să îmbinăm într-un singur produs tehnologii diverse – de la microservicii și comunicații în timp real, până la plăți online și interacțiuni personalizate cu utilizatorii – într-o manieră coerentă și ușor de folosit.

Rezultatul confirmă ideea de la care am pornit: industria evenimentelor are nevoie de soluții digitale tot mai sofisticate, capabile să răspundă unor așteptări din ce în ce mai ridicate. EventHub dovedește că se poate construi o platformă bogată în funcții, dar totuși simplă pentru utilizatori și pregătită să crească odată cu numărul de clienți.

Experiența de dezvoltare a evidențiat importanța unei arhitecturi bine gândite. Separarea clară între front-end și back-end, design-ul modular și practicile de dezvoltare scalabile ne-au ajutat să ținem sub control complexitatea proiectului.

Integrarea tehnologiilor WebSocket a ridicat experiența utilizatorului la un nivel superior. Mesajele și actualizările care apar instant pe ecran nu sunt doar un „plus” tehnic, ci schimbă fundamental modul în care oamenii interacționează cu platforma.

Pe lângă aportul practic, proiectul aduce și o serie de lecții valoroase pentru alte domenii care trec prin procese similare de digitalizare. În același timp, pentru noi, ca dezvoltatori, a fost un demers educațional aplicat, care a transformat teoria în experiență concretă.

Privind înainte, viitorul EventHub va depinde de cât de repede va putea integra trenduri precum inteligența artificială, aplicații mobile dedicate sau realitatea augmentată. Flexibilitatea arhitecturii actuale ne asigură un punct de plecare solid.

În concluzie, EventHub arată că, atunci când cercetarea academică se întâlnește cu nevoile reale ale unei industrii, pot apărea soluții care aduc valoare și utilizatorilor, și comunității profesionale.

Bibliografie:

1. D. Moise, *Marketingul și managementul evenimentelor*. București: Editura ASE, 2014.

2. Oprea, Dumitru, Gabriela Meșniță, Marius Alexa. Managementul evenimentelor: personale, organizaționale, internaționale. Iași: Editura Polirom, 2016.
3. Raj, Razaq, Paul Walters, Tahir Rashid. Events Management: An Integrated and Practical Approach. London: Sage Publications, 2009.
4. Van der Wagen, Lynn și Brenda R. Carlos. Event Management for Tourism, Cultural, Business and Sporting Events. Ed. a 4-a, Pearson Education, 2010.
5. Cucu, Nicoleta Alice. "Promovarea evenimentelor în mediul online." Revista de Marketing Online, vol. 11, nr. 2, 2017, pp. 46-56
6. Hayes, Adam. "Is Ticketmaster a Monopoly?" Investopedia, 19 septembrie 2024
7. InEvent, „Top 11 Best Event Ticketing Platforms in 2025,” *InEvent Blog*, 2023.
8. Hanlon, Tom. "Are You Still Running An Event With Only Offline Sales? Not Moving To Online Ticketing Is Costing More Than You Think." *Ticketbud Blog*, 12 aprilie 2019.
9. R. Ahearn, "Then & Now: The Evolution of the Event Industry," TSV Sound & Vision Blog, 2022.
10. Allied Market Research, "Events Industry by Type, Revenue Source, Organizer and Age Group: Global Opportunity Analysis and Industry Forecast, 2019–2026," AlliedMarketResearch.com, 2019.
11. Bishop-McCann, "Corporate Events for the Modern Attendee: Capturing Attention in a Digital World," BishopMcCann Blog, Feb. 2025.
12. run.events, "Event industry stakeholders: challenges and opportunities," run.events Blog, 2023.
13. Consortium UK, "Pros and Cons of Eventbrite," Consortium Business Solutions Blog, rev. 2025.
14. MCI Group, "The benefits of an integrated event management platform," MCI Group Insights, 2023.
15. vFairs, "Top 5 Ticketmaster Competitors and Alternatives," vFairs Blog, 2023.
16. L. Richardson și S. Ruby, *RESTful Web Services*. Sebastopol, CA: O'Reilly Media, 2007.
17. M. Mikowski și J. Zaczek, Single Page Web Applications: JavaScript End-to-End. Shelter Island, NY: Manning Publications, 2014.

18. M. Casciaro și L. Mammino, *Node.js Design Patterns*, 3 ed. Birmingham, UK: Packt Publishing, 2020.
19. K. H. Thornton și T. Buhrmann, *Pro Express.js: Master Express.js—the Node.js Framework for Web Development*. New York, NY: Apress, 2014.
20. D. Y. Golash, *Learning PostgreSQL*, 3 ed. Birmingham, UK: Packt Publishing, 2020.
21. L. J. F. Helmuth și T. Schroeder, *PostgreSQL: Up & Running*, 3 ed. Sebastopol, CA: O'Reilly Media, 2017.
22. P. J. Jones, *JSON Web Tokens: A Practical Guide*. Upper Saddle River, NJ: Addison-Wesley, 2021.
23. E. Hardt, *OAuth 2.0*. Sebastopol, CA: O'Reilly Media, 2021.
24. A. Banks și E. Porcello, *Learning React: Functional Web Development with React and Redux*, 2 ed. Sebastopol, CA: O'Reilly Media, 2020.
25. M. Tim, *Redux Quickly*. Shelter Island, NY: Manning Publications, 2017.
26. A. Syu, *Tailwind CSS – From Zero to Production*. Birmingham, UK: Packt Publishing, 2022.
27. P. Lubbers și B. Albers, *HTML5 WebSocket*. New York, NY: Apress, 2011.
28. G. Ferrariolo, *Real-Time Web Apps with Socket.IO*. Shelter Island, NY: Manning Publications, 2020.
29. M. Hynes, *Node.js 14 Cookbook*. Birmingham, UK: Packt Publishing, 2021.
30. A. White, *Professional Email with Node.js*. New York, NY: Apress, 2018.
31. D. Fisher și L. Erdman, *Google Maps Platform for Developers*. Birmingham, UK: Packt Publishing, 2021.
32. V. Romano, *Testing JavaScript Applications with Jest*. Berkeley, CA: Apress, 2021.

