**MINISTERUL EDUCAŢIEI**

**UNIVERSITATEA „1 DECEMBRIE 1918” DIN ALBA IULIA**

**FACULTATEA DE INFORMATICĂ ȘI INGINERIE SPECIALIZAREA INFORMATICĂ**

**FORMA DE ÎNVĂŢĂMÂNT ZI**

**LUCRARE DE LICENȚĂ**

**COORDONATOR ȘTIINȚIFIC**

**LECT .UNIV. DR. INCZE ARPAD**

**ABSOLVENT**

**SAVU ALEXANDRU**

**ALBA IULIA**

**2024**

**MINISTERUL EDUCAŢIEI**

**UNIVERSITATEA „1 DECEMBRIE 1918” DIN ALBA IULIA**

**FACULTATEA DE INFORMATICĂ ȘI INGINERIE SPECIALIZAREA INFORMATICĂ**

**FORMA DE ÎNVĂŢĂMÂNT ZI**

**Site Web Prestării Servicii Foto Video Evenimente**

**COORDONATOR ȘTIINȚIFIC**

**LECT .UNIV. DR. INCZE ARPAD**

**ABSOLVENT**

**SAVU ALEXANDRU**

**ALBA IULIA**

**2024**

**Cuprins**

Cuprins

[Cuprins 2](#_Toc169612867)

[Introducere 3](#_Toc169612868)

[1.ANALIZA CONCEPTELOR ȘI 4](#_Toc169612869)

[LITERATURII DE SPECIALITATE 4](#_Toc169612870)

[1.1.Alternative și Motivația Alegerii Tehnologiilor 4](#_Toc169612871)

[1.2.Tehnologii WEB: HTML5 și CSS3 6](#_Toc169612872)

[1.3.Limbaje de Programare: PHP și JavaScript 6](#_Toc169612873)

[2.DEZVOLTAREA SITE-ULUI WEB 9](#_Toc169612874)

[2.1.Realizarea și Implementarea bazei de date 9](#_Toc169612875)

[2.1.1. De Ce MySQL? 9](#_Toc169612876)

[2.2. Integrarea Frontend și Interacțiunea cu Backend-ul 20](#_Toc169612877)

[3.TESTAREA FUNCȚIONALITĂȚILOR ȘI AJUSTĂRILE EFECTUATE 37](#_Toc169612878)

[3.1.Ajustările efectuate site-ului 37](#_Toc169612879)

[3.1.1. Ajustări la galeriile imagini 37](#_Toc169612880)

[3.2. Testarea Funcționalităților Site-ului 40](#_Toc169612881)

[CONCLUZII ȘI PROPUNERI 46](#_Toc169612882)

[BIBLIOGRAFIE 47](#_Toc169612883)

# **Introducere**

Într-o lume în continuare digitalizare,unde accesul rapid și ușor la servicii devine o normă, industria foto-video trebuie să se adapteze așteptărilor moderne ale clienților. Cererile frecvente pentru simplificarea proceselor,pentru a economisi timp și efort, au adus la dezvoltarea acestui proiect.

Aceste preferințe sunt reflectate în dorința lor de a utiliza un formular online pentru rezervarea serviciilor foto si video, evitând interacțiuniile directe si procedurile complicate. Tema propusă oferă o soluție elegantă și eficientă,creând un site web de prezentare a serviciilor foto-video,care facilitează rezervarea online si optimizează gestionarea evenimentelor pentru administratori.

Noutatea temei constă în integrarea completă a unui sistem de rezervare online cu functionalități avansate. Observând ca multe firme din domeniu oferă doar soluții parțiale sau deloc moderne, așadar cu ajutorul acestei teme ne propunem să acoperim acest gol printr-o abordare inovativă si accesibilă.

Scopul principal al acestei teme, este de a transforma interacțiunea clienților cu serviciile foto-video,reducând timpul de răspuns și îmbunătătind experiența generală. Automatizarea proceselor de rezervare și gestionare vizează atât satisfacerea nevoilor clienților, cât și îmbunătățirea eficienței operaționale a firmei.

Metodologia de cercetare a inclus:

* Revizuirea literaturii de specialitate: Analiza articolelor relevante pentru a înțelege tendințele actuale.
* Dezvoltarea și testarea site-ului web: Crearea unui site funcțional si testarea pentru asigurarea funcționalității.
* Testare și ajustări: Testarea propriu-zisă și ajustările necesare.

Structura Lucrării:

1.Introducere- Prezentarea temei,motivația alegerii și obiectivele lucrării.

2.Cadrul teoretic-Analiza conceptelor și a literaturii de specialitate.

3.Dezvoltarea soluției- Descrierea procesului de creatie a site-ului web.

4.Testarea și feedback-Testarea functionalităților și ajustarile necesare

5.Concluzii și propuneri de viitor- Concluziile trase și propunerile viitoare.

# **1.ANALIZA CONCEPTELOR ȘI**

# **LITERATURII DE SPECIALITATE**

Acest capitol are scopul de a explora și analiza conceptele și literatura de specialitate relevante pentru tehnologiile și instrumentele utilizate în dezvoltarea aplicațiilor web. Vom evalua diferite programe și tehnologii, motivând alegerile făcute și oferind o descriere detaliată fiecărei tehnologii.

**Structura Capitolului**

* 1. Alternative și Motivația Alegerii Tehnologiilor
* 2. Tehnologii Web: HTML5 și CSS3
* 3. Limbaje de Programare: PHP și JavaScript

## 1.1.Alternative și Motivația Alegerii Tehnologiilor

În această secțiune, vom examina alte programe și tehnologii similare celor utilizate in proiectul nostru, analizând dezavantajele acestora și motivând alegerile pe care le-am făcut.

Printre aceste suite de software,se numără XAMPP,WAMP și MAMP,fiecare oferind funcționalități specifice pentru dezvoltatori.

XAMPP(Apache,MariaDB,PHP), este o suită de software versatilă, compatibilă cu multiple sisteme de operare, inclusiv macOS, Windows și Linux. Aceasta simplifică procesul de instalare și gestionarea unui server local, fiind ideală pentru dezvoltatori care au nevoie de un mediu de testare local robust și flexibil.

WAMP SERVER(Windows,Apache,MySQL,PHP), este suită de software, disponibilă doar pentru sistemele Windows, fiind ideală pentru dezvoltatorii care folosesc Windows si doresc un mediu de dezvoltare complet.

Totuși,WAMP SERVER, este limitat intr-un mediu de dezvoltare multi-platformă. În plus acesta poate întâmpina dificultăți în actualizarea și gestionarea componentelor individuale ale pachetului, ceea ce poate afecta flexibilitatea și eficiența dezvoltatorilor care necesită configurări personalizate sau cele mai recente versiuni ale componentelor aferente.

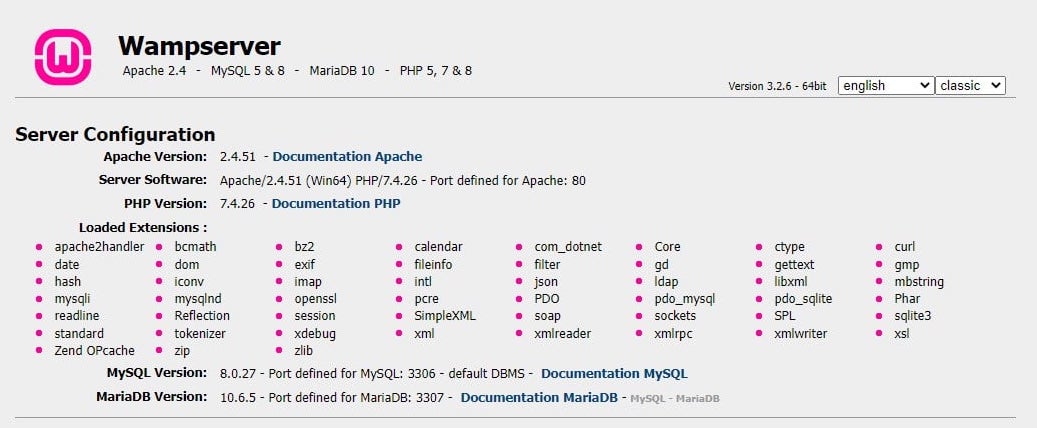


Fig.1.1.2.Interfață a suitei WAMP SERVER

MAMP(Macintosh,Apache,Nginx,MySQL), este o suită software destinată instalării unui server local,disponibilă momentan doar pentru macOS și Windows.Aceasta suită este disponibilă gratuit și se instalează rapid,fara a afecta alte instalări Apache existente.

MAMP nu include nativ multe instrumente necesare pentru depanare, ceea ce poate reduce eficiența dezvoltării unor configurări complexe și funcționalități avansate, facând astfel ca aceasta suită sa fie mai puțin atractivă pentru proiectele care necesită un mediu de dezvoltare cuprinzător și flexibil.

Motivatia Alegerii XAMPP, a fost datorită versatilității sale,ușurința instalării si configurării. Acesta oferă un set complet de funcționalitătii ceea ce il face alegerea ideală pentru un mediu de testare local robust și flexibil.

Unul dintre principalele avantaje ale XAMPP ar fii suportul său pentru o gamă largă de componente esențiale pentru dezvoltarea web, inclusiv Apache, MariaDB(MySQL), PHP și Perl. Însemnând ca avem la dispoziție toate instrumentele necesare pentru a dezvolta si testa aplicații web complexe, fără a fii nevoie configurarea și instalarea fiecărei componente in parte.

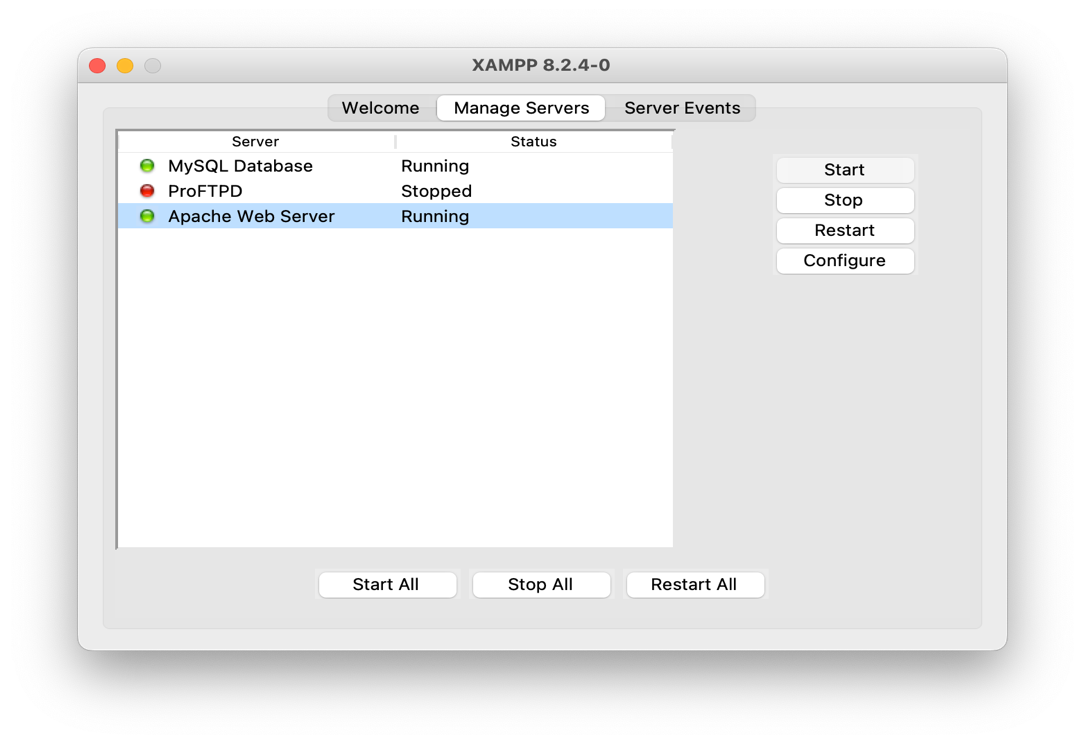
Acesta include, de asemenea un panou de control intuitiv care permite gestionarea ușoară a serviciilor de server, cum ar fii pornirea si oprirea serverul Apache sau a bazei de date MySQL,astfel face ca administrarea să fie fără bătăi de cap și accesibilă. 

Fig.1.1.4.Panou de Control XAMPP

În plus, XAMPP este actualizat în mod regulat pentru a asigura compatibilitatea cu cele mai recente versiuni ale componentelor sale și pentru a aborda eventualele vulnerabilități de securitate. Aceasta asigurand un mediu sigur si actualizat,reducând riscul de probleme de compatibilitate sau de securitate.

Prin urmare, versatilitatea, setul complet de funcționalități, ușurința de utilizare și suportul extins fac din XAMPP alegerea potrivită pentru necesitățiile temei abordate.

## 1.2.Tehnologii WEB: HTML5 și CSS3

În această parte, vom explora două dintre cele mai fundamentale tehnologii utilizate în dezvoltarea web:HTML5 și CSS3.Aceste tehnologii formează coloana vertebrală a oricărei pagini web,asigurând structura și stilizarea conținutului. Utilizarea acestor tehnologii este esențială pentru dezvoltarea unor aplicații web eficiente, scalabile și interactive,care să răspundă nevoilor utilizatorilor moderni.

HTML5(Hypertext Markup Language) este cea mai recentă versiune a limbajului standard de marcare a conținutului web. Aceasta adaugă noi elemente și atribute care oferă mai multe opțiuni pentru a structura semantic paginile web și pentru a îmbunătăți accesibilitatea.

HTML5 este utilizat pentru a defini structura și conținutul paginilor web. Acesta permite dezvoltarea unor pagini web semantice și accesibile,integrând elemente multimedia și grafice fără a necesita extensii externe.

HTML5 oferă numeroase avantaje, inclusiv:

* Suport îmbunătățit pentru elemente multimedia: Integrarea nativă a elementelor media. Acest lucru este important pentru tema abordata, deoarece permite includerea de imagini de inaltă calitate direct pe site,îmbunătățind experiența vizuală a utilizatorilor.
* Elemente semantice: Etichete noi precum ‚<header>’ ‚<footer>’,<article>’ și ‚<section>’. Acestea contribuie la îmbunătățirea structurii si clarității codului,facilitând navigarea și menținerea paginii.
* Compatibilitatea: HTML5 este suportat de toate browserele moderne, asigurând o experiență uniformă pentru utilizatori. Aceasta este esențială pentru a garanta funcționarea corectă pe o gamă largă de dispozitive.

Astfel,HTML5 reprezintă o alegere esențială pentru dezvoltarea web pentru realizarea unor site-uri eficiente si moderne.

## 1.3.Limbaje de Programare: PHP și JavaScript

În această secțiune, ne vom concentra asupra a două dintre cele mai populare limbaje de programare utilizate în dezvoltarea web: PHP și JavaScript. Aceste limbaje sunt cruciale pentru crearea aplicațiilor web dinamice și interactive,oferind funcționalități extinse atât pe partea de server, cât și pe partea de utilizator. Vom discuta avantajele si neajusunrile , precum și motivele care au stat la baza alegerii lor ïn cadrul temei abordate.

PHP(Hypertext Preprocessor), este un limbaj de programare server-side utilizat pe scară largă pentru dezvoltarea aplicațiilor web dinamice. Este recunoscut pentru flexibilitatea și simplitatea sa,permițând dezvoltatorilor să creeze rapid și eficient funcționalități complexe pentru site-uri web.

Limbajul are o sintaxă intuitivă, care faciliteaza scrierea rapidă a codului, ceea ce îl face accesibil atât pentru începători, cât și pentru cei mai experimentați. De asemenea, acesta se integrează excelent cu diverse sisteme de gestionare a bazelor de date, cum ar fi MySQL,facilitând manipularea și accesul la date.

Un avantaj semnificativ al PHP este comunitatea sa vastă și resursele abundente disponibile, precum si urmatoarele avantaje:

* Performanță Optimizată: PHP este optimizat pentru aplicații web,oferind timpi de răspuns rapizi.
* Gestionarea solicitărilor simultane: Acesta poate gestiona un număr mare de solicitări simultane,fiind ideal pentru site-uri cu trafic intens.
* Open-source: PHP e un limbaj gratuit și cu sursă deschisă,cee ace însemnă dezvoltarea continuă acestui limbaj.

Deși PHP oferă numeroase avantaje, există si cateva aspect care trebuie luate în considerare pentru a maximiza beneficiile acestui limbaj.

În primul rând, pentru a preveni vulnerabilitățile la atacuri, cum ar fi SQL injection, este essential ca programatorii să scrie codul cu atenție și să implementeze măsuri de Securitate adecvate,cum ar fi utilizarea funcțiilor de pregătire a interogărilor SQL și validarea intrărilor de la utilizatori.

De asemenea, pentru a asigura o întreținere și extindere ușoară a codului, este important ca acesta să fie bine organizat și documentat. Acest lucru poate fi realizat prin respectarea bunelor practice de programare,utilizarea comentariilor în cod și adoptarea unui stil de codare consistent.

Astfel, PHP continuă să fie un pilon fundamental în dezvoltarea web datorită flexibilității sale,comunității vaste și integrării excelente cu diverse baze de date. Cu toate acestea,pentru a maximiza beneficiile oferite de PHP, este crucială adoptarea unor măsuri de securitate stricte și menținerea unui cod bine organizat și documentat.

**JavaScript** este un limbaj de programare interpretat, predominant folosit pentru scripting pe partea de client în paginile web. Permite dezvoltatorilor să implenteze funcții complexe pe pagini web,de la încarcarea conținutului dinamic fără a reîncărca pagina(AJAX) la crearea de jocuri și animații.

Avantajele utilizării JavaScript includ interactivitatea sporită și actualizările în timp real. Totodată, beneficiază de un ecosistem si activ, cu numeroase biblioteci și cadre de lucru care contribuie la imbunătățirea calității aplicațiilor.

Alte avantaje care sunt regăsite și în tema abordată sunt:

* Interactivitatea sporită: Permite adăugarea de elemente interactive pe site-uri web, cum ar fii formulare dinamice, galerii de imagini și efecte de navigare
* Programare Asincronă:JavaScript suportă programarea asincronă prin intermediul callback-urilor, promisiunilor și funcțiilor await/asincron. Aceasta permite efectuarea de operațiuni complexe, fără a bloca firul principal de execuție, oferind astfel o experiență de utilizare fluidă și reactivă
* Interacțiuni și Gestionarea DOM: Permite manipularea Document Object Model(DOM), ceea ce înseamnă că poate adăuga, modifica sau șterge elemente HTML și CSS dintr-o pagină web în timp real.

În ciuda numeroaselor avantaje, JavaScript prezintă si cateva dezavantaje care trebuie luate în considerare pentru a asigura o utilizarea optimă și securizată a acestui limbaj. Fiind executat pe partea de client, acest lucru poate face aplicațiile vulnerabile la atacuri XSS(Cross-Site Scripting). Pentru a minimiza aceste riscuri, este esențial să se implementeze măsuri de securitate adecvante, cum ar fi igienizarea și validarea inputurilor utilizatorilor.

Execuția JavaScript pe client poate afecta performanța paginii web, în special pe dispozitive mai vechi sau cu resurse limitate. Dezvoltatorii pot optimiza codul și utiliza tehnici de încarcare asincronă pentru a îmbunătăți performanța. Funcționalitatea JavaScript poate varia între browsere,de aceea este important să se testeze aplicațiile pe mai multe platforme.

Aceste dezavantaje nu trebuie sa descurajeze utilizarea acestuia, ci sa fie o motivare pentru adoptarea unor mai bune practici și măsuri preventive pentru a crea aplicații web sigure și robuste.

# **2.DEZVOLTAREA SITE-ULUI WEB**

În acest capitol, explorăm etapele esențiale de planificare și proiectare a site-ului web, punând accent pe dezvoltarea backendului, dar oferind și o perspectivă asupra integrării cu frontend-ul. Această faza inițială este crucială pentru stabilirea funcționalităților de bază ale site-ului și pentru asigurarea unei fundații solide pe care să construim interfața utilizatorului și experiența acestuia. Scopul acestei faza este de a asigura că toate cerințele tehnice și estetice sunt îndeplinite, în concordanță cu obiectivele proiectului și așteptările utilizatorilor.

Procesul de dezvoltare al unui site web este complex și cu multe posibilități de realizare, implicând o serie de decizii tehnice care trebuie luate în concordanță cu cerințele proiectului și cu cele mai bune practici din industrie. De la alegerea tehnologiilor și până la implementarea efectivă a soluțiilor tehnice, fiecare pas este esențial pentru succesul final al proiectului.

În acest context,vom explora:

* 1. Realizarea și implementarea bazei de date
* 2. Integrarea Frontend și Interacțiunea cu Backend-ul

## 2.1.Realizarea și Implementarea bazei de date

Acest subcapitol se concentrează pe proiectarea, realizarea și optimizarea bazei de date folosind MySQL, un aspect crucial în arhitectura oricărui site web care necesită gestionarea eficientă a datelor. Discutăm despre selecția tehnologiei, designul bazei de date, implementarea securității și optimizarea performanței, toate aliniate cu cele mai noi tendințe și cele mai bune practici din industrie.

### 2.1.1. De Ce MySQL?

MySQL este un sistem de gestionare a bazelor de date relaționale, open-source, extensiv utilizat în întreaga lume. Dezvoltat inițial de MySQL AB în Suedia, acesta a fost cumpărat de Oracle Corporation în 2010.

MySQL este recunoscut pentru flexibilitatea, scalabilitatea , fiabilitatea și rapiditatea sa, fiind compatibil cu toate platformele majore de sistem de operare, inclusiv macOS , linux, UNIX și Windows.Acesta este proiectat pentru a oferi o viteză de procesare rapidă a datelor, un suport extensiv pentru baze de date mari și o serie de opțiuni de configurare care pot fi optimizate pentru diferite cerințe de performanță.

Totodata,suporta multe dintre standardele SQL, inclusiv tranzacții complexe, subinterogări, declanșatoare și proceduri stocate. De asemenea,oferă un sistem de securitate bazat pe privilegii pe mai multe niveluri, care garantează securitatea operațiunilor și datelor.

Fundamentarea deciziei pentru alegerea acestui sistem de gestionare a bazelor de date a venit datorită capabilităților sale avansate de management a tranzacțiilor și a datelor în volum mare, esențiale pentru operarea eficientă a unui site web interactiv. Alegerea este susținută de literatura de specialitate care accentuează robustețea, scalabilitatea și securitatea MySQL, elemente vitale pentru structurile web dinamice de astăzi.

**2.1.2. Implementarea Bazei de date**

În acest subcapitol, vom discuta procesul tehnic prin care baza de date Grozav\_DB a fost create și configurată folosind MySQL, gestionat prin interfața phpMyAdmin. Procesul a implicat definirea structurilor de tabel, stabilirea relatiilor dintre acestea și popularea inițială cu date de test, toate acestea pentru a suporta funcționalitățile site-ului web dezvoltat.

Baza de date Grozav\_DB include mai multe tabele esențiale pentru funcționarea site-ului, printer care ‘administrators’, ‘programate’ și ‘rezervări’. Fiecare tabel a fost creat pentru a suporta diferite aspecte ale gestionării datelor utilizatorilor și evenimentelor

Tabelul ‘administrators’ se ocupă de stocarea informațiilor despre administratorii site-ului, inclusive nume de utilizator, parolă criptată și rolul fiecărui administrator.

**CREATE** **TABLE** `administrators` (

`id` int(11) **NOT** **NULL**,

`username` varchar(255) **NOT** **NULL**,

`password` varchar(255) **NOT** **NULL**,

`**role**` enum('client','administrator') **NOT** **NULL** **DEFAULT** 'client'

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8mb4 **COLLATE**=utf8mb4\_general\_ci;

Tabelul este compus din următoarele coloane:

* id: Un camp numeric, utilizat ca cheie primară, ce identifică univoc fiecare înregistrare în tabel. Acesta este auto-incrementat, garantând unicitatea și facilitând gestionarea eficientă a înregistrărilor.
* username: Un câmp de tip varchar(255) care permite stocarea de text cu lungime variabilă. Acest câmp stochează nume de utilizatorul al administratorului. Este vital în procesul de autentificare și trebuie să fie unic în cadrul bazei de date.
* password: Un câmp de tip varchar(255), care conține parola criptată a administratorului. Criptarea asigură securitatea datelor sensibile și protejează accesul neautorizat la platformă.
* role: Un câmp de tip enumerare, care definește rolul utilizatorului în cadrul site-ului, permițând diferențierea între ‘client’ și administrator.Acest câmp este esențial pentru implentarea accesului pe bază de roluri,facilitând securizarea la funcționalitățile diferite ale site-ului.

Tabel ‚programate’ este destintat stocării informațiilor detaliate despre evenimentele programate de utilizatori pe site. Aceasta tabelă este esențială pentru gestionarea și planificarea eficientă a evenimentelor.

**CREATE** **TABLE** `programate` (

`id` int(11) **NOT** **NULL**,

`nume` varchar(255) **NOT** **NULL**,

`prenume` varchar(255) **NOT** **NULL**,

`email` varchar(255) **NOT** **NULL**,

`numar\_contact` varchar(255) **NOT** **NULL**,

`locatie` varchar(255) **NOT** **NULL**,

`nr\_persoane` int(11) **NOT** **NULL**,

`servicii` text **NOT** **NULL**,

`data\_evenimentului` date **NOT** **NULL**,

`suma\_plata\_totala` decimal(10,2) **NOT** **NULL**,

`avans\_de\_platit` decimal(10,2) **NOT** **NULL**,

`status` enum('in asteptare','confirmat','finalizat') **DEFAULT** 'in asteptare'

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8mb4 **COLLATE**=utf8mb4\_general\_ci;

Astfel tabela este compusa din următoarele coloane:

* id: Un camp numeric întreg, utilizat ca cheie primară, ce identifică fiecare rezervare în tabel,auto-incrementată asigurând unicitatea fiecărei înregistrări.
* nume:Un camp de tip varchar(255), care stochează numele clientului.
* prenume: Un câmp de tip varchar(255), ce stochează prenumele clientului
* email: Câmp de tip varchar(255), destinate stocării adresei de email a clientului.
* numar\_contact: Un camp de tip varchar(255), care conține numărul de contact al clientului.
* locatie: Un camp de tip varchar(255), care specifică locația unde va avea loc evenimentul.
* nr\_persoane:Un camp numeric întreg, care indică numărul de personae ce vor participa la eveniment.
* servicii: Un camp de tip text, care permite stocarea detaliată a serviciilor solicitate de client pentru eveniment.
* data\_evenimentului: Un câmp de tip date, care specifică data la care va avea loc evenimentul.
* suma\_plata\_totala: Un câmp de tip decimal(10,2), ce indică suma totală ce trebuie platită pentru serviciile prestate. ‘decimal(10,2)’ asigură precizia necesară pentru gestionarea sumelor financiare
* avans\_de\_platit:Similar câmpului vecin,doar că specifică avansul ce trebuie plătit de client pentru rezervarea evenimentului
* status: Un câmp de tip enumerare, care indică stadiul actual al rezervării, putând avea valorile ‘in asteptare’, ‘confirmat’, sau ‘finalizat’. Acest câmp permite urmărirea facilă a progresului rezervărilor și gestionarea lor eficientă.

Tabelul ‚Rezervari’ este conceput pentru a gestiona integral rezervările effectuate de utilizatori pe platforma web. Acest tabel este essential pentru stocarea și accesul la informațiile detaliate legate de evenimentele organizate prin intermediul site-ului,facilitând o gestionare eficientă și personalizată a fiecărei rezervări.

**CREATE** **TABLE** `rezervari` (

`id` int(11) **NOT** **NULL**,

`nume` varchar(50) **NOT** **NULL**,

`prenume` varchar(50) **NOT** **NULL**,

`data\_nasterii` date **NOT** **NULL**,

`numar\_telefon` varchar(15) **NOT** **NULL**,

`email` varchar(100) **NOT** **NULL**,

`tip\_eveniment` enum('botez','nunta','majorat') **NOT** **NULL**,

`numar\_persoane` int(11) **NOT** **NULL**,

`locatie` varchar(100) **NOT** **NULL**,

`servicii` enum('foto+video','doar foto','doar video') **NOT** **NULL**,

`album\_personalizat` tinyint(1) **DEFAULT** 0,

`data\_evenimentului` date **NOT** **NULL**,

`metoda\_plata` varchar(255) **DEFAULT** **NULL**,

`token` varchar(64) **DEFAULT** **NULL**

) ENGINE=InnoDB **DEFAULT** CHARSET=utf8mb4 **COLLATE**=utf8mb4\_general\_ci;

Tabelul ‘Rezervari’ include următoarele coloane:

* id: Câmp de tip int(11), utilizat ca cheie primară,identifică în parte fiecare rezervare înregistrată.
* nume și prenume: Câmpuri de tip varchar(50), ce stochează numele și prenumele clientului
* data\_nasterii: Un câmp tip date, care înregistrează data nașterii clientului.
* numar\_telefon: Câmp de tip varchar(15), destinate stocării numărului de telefon al clientului.
* email: Câmp de tip varchar(100), care conține adresa de email a clientului.
* tip\_eveniment: Un câmp de tip enumerare, cu opțiunile ‘botez’, ‘nunta’ si ‘majorat’,ce descrie tipul de eveniment pentru care se face rezervarea,facilitând segmentarea și organizarea specifică.
* numar\_persoane: Un câmp de tip int(11), care specifică numărul de personae așteptate la eveniment.
* locatie: Un câmp de tip varchar(100), care indică locația evenimentului.
* servicii: Un câmp de tip enumerare cu opțiunile ‘foto+video’, ‘doar foto’, ‘doar video’, ce captează preferințele de servicii ale clientului, crucial pentru alocarea resurselor.
* album\_personalizat: Un câmp de tip tinyint(1), folosit pentru stocarea de numere întregi foarte mici,precum in cazul nostru.
* data\_evenimentului: Un câmp de tip date, care stabilește când va avea loc evenimentul rezervat. Fiind marcată ca ‘unique’, nu pot exista două rezervări în aceeași zi.
* metoda\_plata: Câmp de tip varchar(255), ce descrie modalitatea de plată aleasă de client, cum ar fi cash urmând pe viitor introducerea a modalități de plată cu cardul.
* token: Un câmp de tip varchar(64), utilizat pentru stocarea unui identificatory unic asociat fiecărei tranzacții sau rezervări, îmbunătățind securitatea și urmărirea plăților.

Implementarea bazei de date a fost un proces esențial în dezvoltarea infrastructurii tehnice a site-ului nostru. Prin configurarea atentă, am asigurat o fundație solidă pentru gestionarea eficientă a datelor utilizatorilor și evenimentelor. Procesul de implementare a evidențiat importanța unei planificări meticuloase în faza inițială de proiectare a bazei de date pentru a asigura flexibilitatea și scalabilitatea acesteia.

**2.1.3. Scheme și Diagrame ale Bazei de Date**

Pentru a oferi o înțelegere completă, explorăm reprezentările vizuale care ilustrează structura și funcționalitățile bazei de date ‘Grozav\_DB’. Aceste diagrame, realizate cu ajutorul PlantUML, sunt esențiale pentru a înțelege modul în care datele sunt gestionate și interconectate.

PlantUML permite descrierea vizuală a structurilor complexe prin cod textual simplu, transformând specificațiile descriptibile în reprezentări grafice precise și eficiente. Această abordare nu numai că sprijină înțelegerea arhitecturii actuale, dar și facilitează planificarea strategică și tehnică necesară pentru evoluția ulterioară a bazei de date, aliniindu-se perfect cu obiectivele noastre de dezvoltare și optimizare.

Diagrama Proiectare a bazei de date

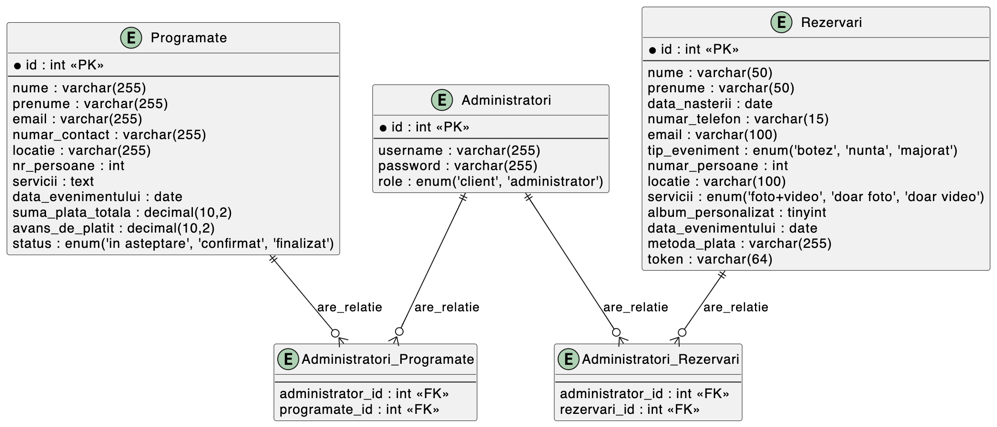


Fig.2.1.3.1. Diagrama Proiectarea Bazei de Date

Diagrama Entitate-Relație(ER)

Această diagramă oferă o reprezentare grafică a entităților din baza de date și a relațiilor dintre acestea.



Fig.2.1.3.2.Diagrama Entitate Relație a bazei de date Grozav\_DB

Această diagramă este fundamentală pentru a înțelege modul în care tabelele sunt structurate și cum interacționează, facilitând atât administrarea datelor cât și planificarea extensiilor structurii de date.

Diagrama UML de Clase

Diagrama UML de clase descrie structura claselor, incluzând atributele și metodele acestora.

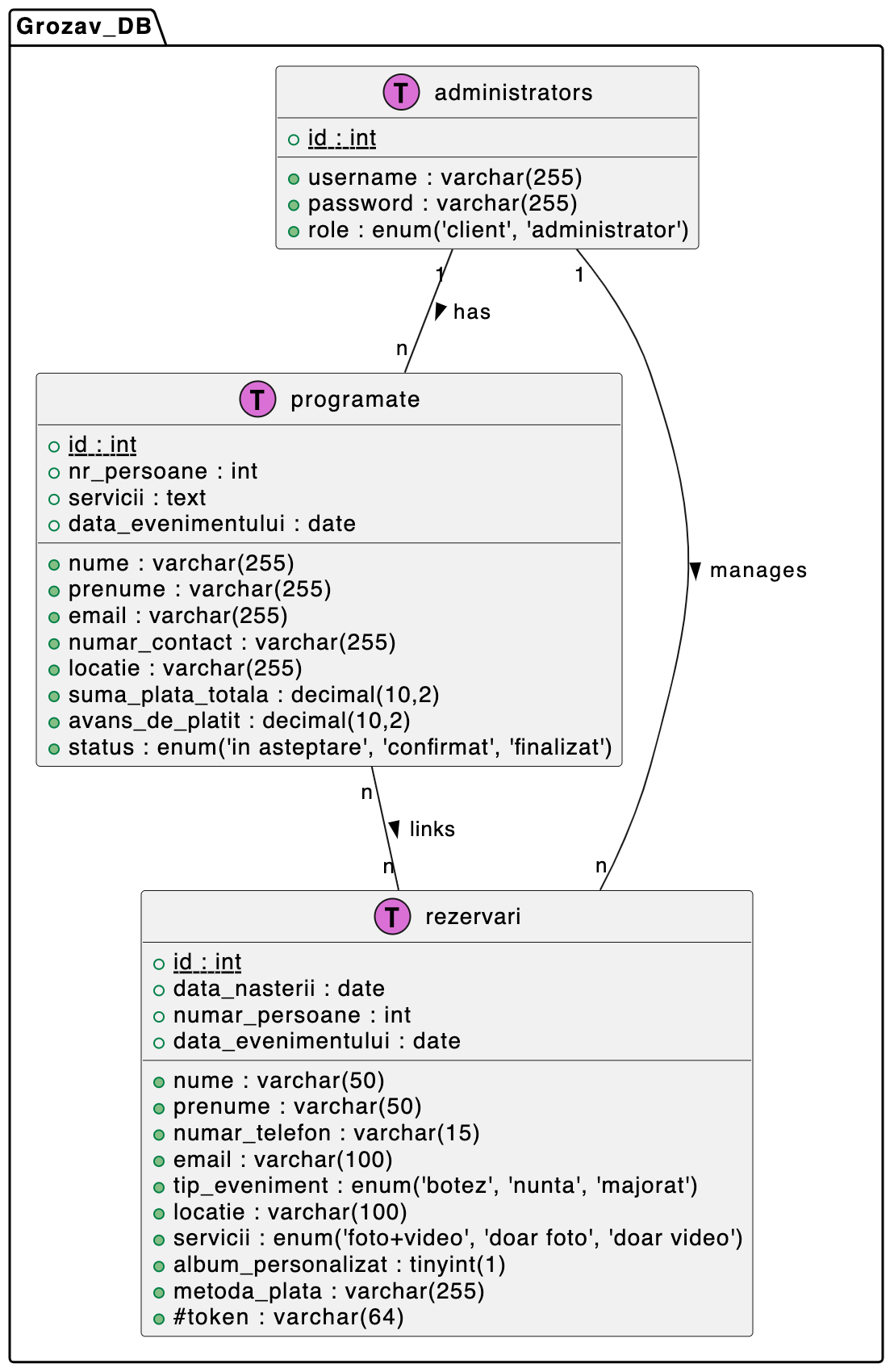


Fig.2.1.3.3. Diagramă UML Clase a bazei de date Grozav\_DB

Prin vizualizarea claselor, atributelor, metodelor și relațiilor dintre acestea, diagrama de clase uml contribuie la o mai buna înțelegere a modelului de date și a logicii aplicației

Diagrama de Secvență

Diagrama de secvență ilustrează interacțiunile dintre diferite obiecte în timpul execuției unei operațiuni specifice, cum ar fi procesul de rezervare.

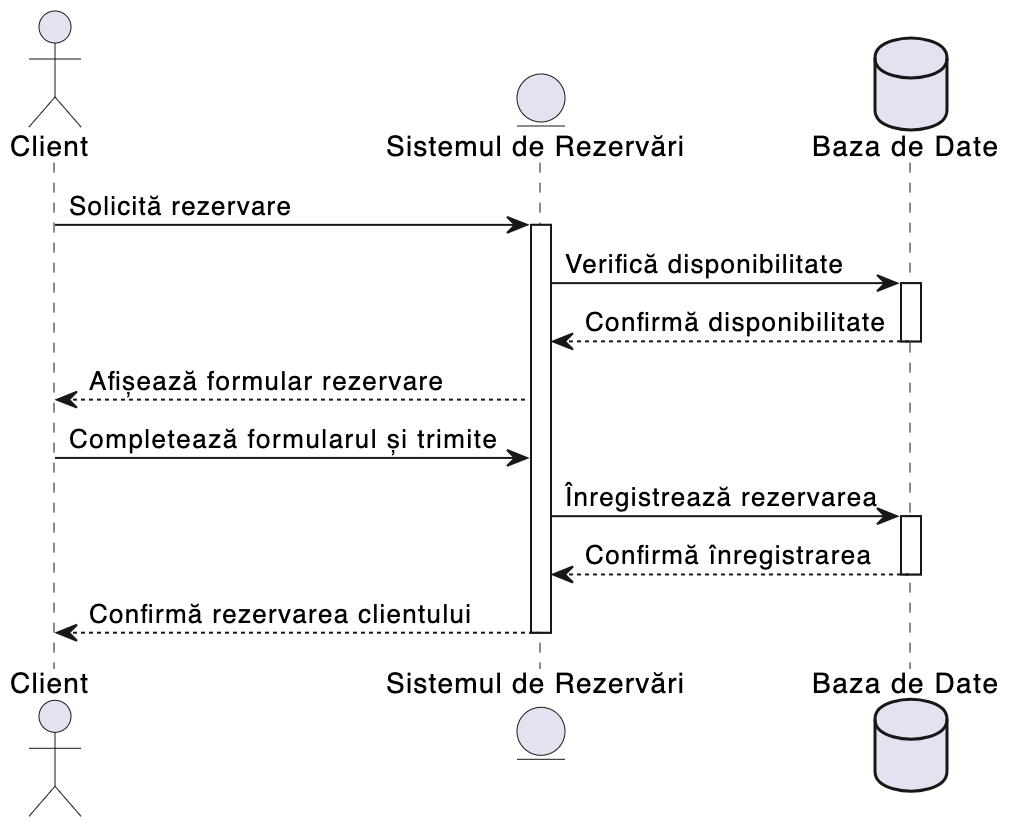


Fig.2.1.3.4. Diagrama de Secvență a bazei de date Grozav\_DB

Prin reprezentarea fluxului de mesaje între diferite componente ale sistemului, această diagramă evidențiază ordinea operațiunilor și interacțiunile necesare pentru completarea unui proces.

Prin utilizarea acestor diagrame, putem oferi o bază solidă pentru înțelegerea, dezvoltarea și optimizarea continuă a bazei de date Grozav\_DB, asigurând că toate componentele sunt corect integrate și funcționale.

2.1.4. Implementarea Securității Bazei de Date

Securitatea bazei de date este un element esențial pentru protejarea datelor sensibile ale utilizatorilor și pentru asigurarea integrității și confidențialității informațiilor gestionate de site-ul web. Astfel vom discuta măsurile implementate pentru securizarea bazei de date, începând cu stabilirea unei conexiuni securizate, continuând cu validarea și filtrarea datelor introduse de utilizatori, și finalizând cu gestionarea tranzacțiilor pentru a garanta consistența și integritatea datelor.

Primul pas pentru asigurarea securității datelor este stabilirea unei conexiuni securizate între aplicația noastră și serverul bazei de date. Utilizăm extensia ‘mysqli’ în PHP pentru a iniția conexiuni care sunt configurate să minimizeze riscurile de interceptare sau corupere a datelor transmise. Toate erorile de conectare sunt gestionate discret pentru a preveni expunerea detaliilor sistemului, asigurându-ne că mesajele de eroare prezentate utilizatorilor sunt generice și nu oferă indicii care ar putea fi folosite în atacuri.

Iată cum procedăm pentru inițierea conexiunii și gestionarea erorilor de conectare în mod securizat:

$host = "127.0.0.1";

$username = "root";

$password = "";

$dbname = "Grozav\_DB";

$conn = **new** mysqli($host, $username, $password, $dbname);

**if** ($conn->connect\_error) {

**echo** json\_encode(['success' => **false**, 'message' => "Conexiunea a eșuat: " . $conn->connect\_error]);

**exit**;

}

Prelucrarea securizată a datelor din formularele de rezervare, necesită implementarea unor măsuri riguroase de securitate pentru a preveni accesul neautorizat și a proteja integritatea datelor.

Procesul detaliat mai jos utilizează validarea și filtrarea inputurilor, tranzacții și generarea de tokeni securizați pentru a asigura o manipulare sigură a datelor.

* + ‚FILTER\_SANITIZE\_STRING’: Această funcție este folosită pentru a elimina toate caracterele care nu sunt printabile și tag-urile HTML dintr-un string, prevenind astfel diverse tipuri de atacuri de scripting.
  + ‚FITLER\_SANITIZE\_EMAIL: Aceasta curăță adresa de email, eliminând caracterele care nu sunt permise în adresele de email conform standardelor.
  + ‘FILTER\_SANITIZE\_NUMBER\_INT’ și ‘FILTER\_SANITIZE\_NUMBER\_FLOAT’: Aceste funcții se asigură că inputul este un număr întreg, respectiv un număr în virgulă mobilă valid, eliminând orice alte caractere non-numerice

Exemplu de cod pentru validarea și filtrarea datelor:

**if** (isset($\_POST['action']) && $\_POST['action'] === 'submit\_reservation') {

$firstName = filter\_input(INPUT\_POST, 'firstName', FILTER\_SANITIZE\_STRING);

$lastName = filter\_input(INPUT\_POST, 'lastName', FILTER\_SANITIZE\_STRING);

$birthDate = filter\_input(INPUT\_POST, 'birthDate', FILTER\_SANITIZE\_STRING);

$phoneNumber = filter\_input(INPUT\_POST, 'phoneNumber', FILTER\_SANITIZE\_STRING);

$email = filter\_input(INPUT\_POST, 'email', FILTER\_SANITIZE\_EMAIL);

$eventType = filter\_input(INPUT\_POST, 'eventType', FILTER\_SANITIZE\_STRING);

$guestCount = filter\_input(INPUT\_POST, 'guestCount', FILTER\_SANITIZE\_NUMBER\_INT);

$eventLocation = filter\_input(INPUT\_POST, 'eventLocation', FILTER\_SANITIZE\_STRING);

$eventDate = filter\_input(INPUT\_POST, 'eventDate', FILTER\_SANITIZE\_STRING);

$services = filter\_input(INPUT\_POST, 'services', FILTER\_SANITIZE\_STRING);

$memoryBox = isset($\_POST['memoryBox']) ? 1 : 0;

$paymentMethod = filter\_input(INPUT\_POST, 'paymentMethod', FILTER\_SANITIZE\_STRING);

$totalCost = filter\_input(INPUT\_POST, 'totalCost', FILTER\_SANITIZE\_NUMBER\_FLOAT, FILTER\_FLAG\_ALLOW\_FRACTION);

$deposit = filter\_input(INPUT\_POST, 'deposit', FILTER\_SANITIZE\_NUMBER\_FLOAT, FILTER\_FLAG\_ALLOW\_FRACTION);

Un alt aspect vital al securități este generarea de tokeni securizați, care sunt folosiți pentru a valida și urmări cererile utilizatorilor fără a expune date personale sau a permite acțiuni neautorizate.

Folosim funcția precizată mai jos, pentru a genera un string hexazecimal aleator, care servește ca identificator unic pentru fiecare rezervare.

$token = bin2hex(random\_bytes(16)); *// Generarea unui token securizat*

Implementarea tranzacțiilor și înregistrarea datelor, este esențială pentru asigurarea integrității datelor, mai ales când sunt implicate multiple operații de inserare sau actualizare.

Prin tranzacții, asigurăm că întregul set de operațiuni reușește sau eșuează ca un tot unitar, ceea ce previne stările de date parțiale sau incoerente.

În cadrul aplicației noastre, tranzacțiile sunt folosite pentru a gestiona înregistrările din formularul de rezervare, garantând că toate detaliile introduse sunt fie salvate complet, fie abandonate în caz de eroare, fără a afecta baza de date cu informații incomplete sau eronate. Acest proces incepe cu inițierea unei tranzacții în momentul în care un utilizator finalizează formularul de rezervare:

$conn->begin\_transaction();

**try** {

$stmt = $conn->prepare("INSERT INTO rezervari (nume, prenume, data\_nasterii, numar\_telefon, email, tip\_eveniment, numar\_persoane, locatie, data\_evenimentului, servicii, album\_personalizat, metoda\_plata, token) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");

$stmt->bind\_param("ssssssisssiss", $firstName, $lastName, $birthDate, $phoneNumber, $email, $eventType, $guestCount, $eventLocation, $eventDate, $services, $memoryBox, $paymentMethod, $token);

$stmt->execute();

$orderID = $stmt->insert\_id; *// Obținem ID-ul comenzii recent înregistrate din tabelul 'rezervari'*

$stmt->close();

$stmt = $conn->prepare("INSERT INTO programate (nume, prenume, email, numar\_contact, locatie, nr\_persoane, servicii, data\_evenimentului, suma\_plata\_totala, avans\_de\_platit, id) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");

$stmt->bind\_param("sssssissssi", $firstName, $lastName, $email, $phoneNumber, $eventLocation, $guestCount, $services, $eventDate, $totalCost, $deposit, $orderID);

$stmt->execute();

$stmt->close();

$conn->commit();

**echo** json\_encode(['success' => **true**, 'message' => 'Rezervarea a fost înregistrată cu succes.', 'orderNumber' => $orderID, 'token' => $token]);

} **catch** (Exception $e) {

$conn->rollback();

**echo** json\_encode(['success' => **false**, 'message' => 'Eroare la înregistrarea rezervării: ' . $e->getMessage()]);

}

$conn->close();

} **else** {

**echo** json\_encode(['success' => **false**, 'message' => 'Acțiune necunoscută sau lipsă.']);

$conn->close();

}

Implementarea detaliată a tranzacțiilor și gestionarea sigură a datelor evidențiază angajamentul nostru față de securitatea și integritatea sistemului de gestionare a datelor. Prin utilizarea metodelor de tranzacție prezentate și manevrarea atentă a erorilor, asigurăm că fiecare operațiune de înregistrare a datelor este efectuată cu maximă precauție. Aceasta nu doar ca sporește fiabilitatea sistemului nostru, dar și oferă utilizatorilor finali un grad ridicat de încredere în modul în care informațiile lor personale sunt manipulate și stocate.

Finalizarea fiecarei tranzacții cu un mesaj de succes sau de eroare clar asigură transparență și oferă feedback imediat, atat pentru administratori cât și pentru utilizatori, facilitând o mai bună gestionare a interacțiunilor cu baza de date. Astfel, fiecare componentă a codului contribuie la o infrastructură robustă și securizată, importantă pentru operațiunile noastre cotidiene și pentru protecția datelor utilizatorilor.

## 2.2. Integrarea Frontend și Interacțiunea cu Backend-ul

Acest subcapitol explorează modul în care componentele front-end și backend sunt integrate pentru a oferi o experiență de utilizarea fluentă și eficientă. Vom discuta utilizarea HTML, CSS și JavaScript pentru a construi interfața utilizatorului și cum aceasta comunică cu serverul prin intermediul scripturilor PHP.

2.2.1. Structura HTML

HTML este limbajul de bază folosit pentru crearea și structurarea paginilor web. Acesta constituie scheletul fiecărui site web, iar in cazul site-ului nostru, funcționalitatea cât și estetica sunt optimizate, facilitând o navigație intuitivă și oferind o experiență de utilizator îmbunătățită. Prin utilizarea unui layout responsiv și a tehnologiilor moderne, HTML-ul ajută la prezentarea conținutului intr-un mod clar și accesibil.

Începând cu ‘<head>’, sunt definite metadate esențiale pentru compatibilitatea cu dispozitivele mobile prin ‚<metaname= „viewport”>’, iar folosirea legăturilor către CSS extern și biblioteci, cum ar fi Swiper pentru caruseluri și FontAwesome pentru icoane, asigura funcționalități dinamice și unde design atrăgător fără a încarca inutil serverul local.

**<head>**

**<meta** charset="UTF-8"**>**

**<meta** name="viewport" content="width=device-width, initial-scale=1.0"**>**

**<title>**Grozav Photography**</title>**

**<link** rel="icon" href="imagini/favconG.svg"**>**

**<link** rel="stylesheet" href="style.css"**>**

**<link** rel="stylesheet" href="app.css"**>**

**<link** rel="stylesheet" href="https://cdn.jsdelivr.net/npm/swiper/swiper-bundle.min.css"**/>**

**<link** rel='stylesheet' href='https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css'**>**

**<script** src="https://cdn.jsdelivr.net/npm/swiper/swiper-bundle.min.js"**></script>**

**</head>**

Corpul principal ‘<body>’ începe cu un ‘<header>’ care nu doar servește ca punct focal vizual prin prezentarea logoului, dar include și un meniu navigațional comprimabil, optimizat pentru interfețele tactile prin butonul de tip hamburger. Aceasta abordare asigură că site-ul rămâne ușor de utilizat chiar și pe dispozitivele mobile.

**<header>**

**<div** class="container"**>**

**<a** href="index.php"**>**

**<img** src="imagini/logogrozav.jpg" alt="Logo Grozav Photography" style="height: 100px; width: 130px; margin-right: 10px; background-color: black; border-radius: 10px;"**>**

**</a>**

**</div>**

**<nav>**

**<button** id="hamburger-button" class="hamburger" aria-label="Toggle navigation menu" onclick="toggleMenu()"**>**☰**</button>**

**<ul** class="nav-list"**>**

**<li><a** href="#home"**>**Acasă**</a></li>**

**<li><a** href="#services"**>**Servicii**</a></li>**

**<li><a** href="booking.php"**>**Rezervare**</a></li>**

**<li><a** href="contact.php"**>**Contact**</a></li>**

**</ul>**

**</nav>**

**</div>**

Sectiunile ulterioare, cum ar fii caruselul de imagini și zonele de conținut, sunt gândite pentru a sprijini vizual narativa brandului, fiecare element fiind atent plasat pentru a atrage și menține atenția utilizatorului. Implementarea swiper pentru slide-uri aduce un element dinamic paginii, îmbunătățind interactivitatea și angajamentul vizitatorilor.

Caruselul de imagini, este o componentă centrală care aduce dinamism și atractivitate vizuală site-ului. Folosirea Swiper JS permite creaarea unui carusel interactiv, care este usor de navigat și plăcut estetic.

**<div** class="swiper" id="swiper-1"**>**

**<div** class="swiper-wrapper"**>**

**<div** class="swiper-slide"**><img** src="imagini/slide-botez.jpeg" alt="Slide 1" **/></div>**

**<div** class="swiper-slide"**><img** src="imagini/slide-sesdj3.jpeg" alt="Slide 2" **/></div>**

**<div** class="swiper-slide"**><img** src="img/slide3.jpg" alt="Slide 3" **/></div>**

**</div>**

**<div** class="swiper-pagination"**></div>**

**</div>**

Zona de conținut include titlul principal și butonul catre pagina de rezervare, oferind utilizatorilor informații clare și acțiuni directe pe care le pot întreprinde. Aceasta este o secțiune importantă pentru ghidarea utilizatorilor către acțiuni specifice.

**<div** class="header-container"**>**

**<h1>**GROZAV PHOTOGRAPHY**</h1>**

**<p** class="header-subtitle"**>**Alegerea Perfecta pentru Evenimentul tau!**</p>**

**<p** class="header-description"**>**FOTOGRAF ⚭ VIDEOGRAF**</p>**

**<a** href="booking.php" class="book-button"**>**REZERVĂ ACUM!**</a>**

**</div>**

O altă secțiune Swiper este implementată pentru a evidenția diferite sesiuni foto, continuând tema dinamică a paginii și oferind utilizatorilor mai multe exemple vizuale ale serviciilor oferite.

<section id="slider-2">

<div class="containerr wide">

<div class="swiper" id="swiper-2">

<div class="swiper-wrapper">

<div class="swiper-slide">

<figure>

<img src="imagini/slide-nunta.jpeg" />

<figcaption>

<img src="img/icon-location.svg" alt="Icon location">

Sesiune Miresica

</figcaption>

</figure>

</div>

<div class="swiper-slide">

<figure>

<img src="imagini/slide-majorat.jpg" />

<figcaption>

<img src="img/icon-location.svg" alt="Icon location">

Capturam Clipa

</figcaption>

</figure>

</div>

<div class="swiper-slide">

<figure>

<img src="imagini/slide3.jpg" />

<figcaption>

<img src="img/icon-location.svg" alt="Icon location">

Nu doar momentul

</figcaption>

</figure>

</div>

2.2.2. Designul CSS

CSS-ul este responsabil pentru aspectul vizual al site-ului, oferind stilizare și aranjamente estetice care fac site-ul atractiv și ușor de utilizat. În proiectul nostru, acesta contribuie la o interfață consistentă și responsivă, asigurând o experiență plăcută pe toate tipurile de dispozitive.

Variabilele CSS ajută la menținerea unui design consistent și usor de întreținut. Acestea definesc culorile, fonturile și alte elemente de stil care pot fi reutilizate în întregul fișier CSS, asigurând coerența designului și facilitând modificările viitoare.

:root {

--black: #151515;

--black-90: rgba(21, 21, 21, .9);

--black-75: rgba(21, 21, 21, .75);

--black-50: rgba(21, 21, 21, .5);

--bg-1: #E3CFCF;

--bg-2: #E9E9E9;

--accent: #E36262;

--primary-ff: 'EB Garamond', Georgia, serif;

--secondary-ff: 'Montserrat', Helvetica, sans-serif;

}

Stilizarea navigației și a butonului de hamburger asigură o experiență de utilizare clară și accesibilă, inclusiv pe dispozitive mobile. Linkurile din meniu au un stil consistent, cu efecte de tranziție care îmbunătățesc interacțiunea utilizatorului.

*/\* Stiluri pentru navigație \*/*

**nav** {

**display**: flex;

**justify**-**content**: **center**;

align-items: **center**;

**padding-top**: 15px; */\* Spațiu deasupra navigației \*/*

**list-style-type**: **none**;

}

**nav** **ul**{

**list-style-type**: **none**;

}

**nav** **ul** **li**{

**display**: **inline**-**block**;

**padding**: 7px 20px;

}

**nav** **a** {

**display**: **inline**-**block**;

**color**: black;

**font-size**: 1rem;

**border**-radius: 10px;

**border**: 2px **solid** black;

**padding**: 10px 20px;

**margin**: 0 10px; */\* Spațiu între linkuri \*/*

transition: **background-color** 0.3s, **color** 0.3s;

**text-decoration**: **none** !important;

}

**nav** **a**:hover {

**background-color**: black;

**color**: white;

}

**.hamburger** {

**display**: **none**; */\* Ascundem butonul inițial \*/*

**font-size**: 3.5rem; */\* Mărimea simbolului hamburger \*/*

**background**: **none**;

**border**: **none**;

**cursor**: **pointer**;

}

**.menu-overlay** {

**display**: **none**;

**position**: **fixed**;

**top**: 0;

**left**: 0;

**width**: 100%;

**height**: 100%;

**background**: rgba(0, 0, 0, 0.5); */\* Fundal semi-transparent \*/*

**z-index**: 1;

}

*/\* Ascunde hamburger-ul la rezoluții mari \*/*

**.hamburger** {

**display**: **none**;

}

Pentru a asigura că site-ul este optimizat pentru toate dispozitivele, sunt utilizate media queries. Acestea permit ajustarea stilurilor CSS în functie de dimensiunea ecranului, asigurând o experiență de utilizator consistentă și placută indiferent de dispozitiv.

\* **Media** **queries** **pentru** **rezolu**ț**ii** **mici** \*/

**@media** (**max-width**: **768px**) {

**.hamburger** {

**display**: **block**;

**position**: **absolute**;

**right**: 0;

**top**: 0;

*/\* Restul stilurilor pentru butonul hamburger \*/*

}

**nav** **.nav-list** {

**position**: **fixed**;

**top**: 0;

**left**: -110%; */\* Începe din afara ecranului \*/*

**width**: 250px; */\* Lățimea meniului \*/*

**height**: 100%;

**background-color**: rgba(0, 0, 0, 0.1);

flex-**direction**: column;

**justify**-**content**: start;

align-items: start;

**padding**: 20px;

transition: transform 0.3s ease; */\* Animația de deschidere \*/*

**z-index**: 3; */\* Meniul trebuie să fie sub buton \*/*

}

*/\* Când nav-list are clasa .active, va fi afișat \*/*

**.nav-list.active** {

**display**: flex;

transform: translateX(160%); */\* Aduce meniul în vizual \*/*

}

**.menu-overlay** {

**display**: **none**;

**position**: **fixed**;

**top**: 0;

**left**: 0;

**width**: 100%;

**height**: 100%;

**background**: rgba(0, 0, 0, 0.1); */\* Semi-transparent \*/*

backdrop-filter: blur(9px); */\* Ajustează valoarea pentru cât de intens să fie blurul \*/*

**z-index**: 2; */\* Suficient de mare pentru a acoperi conținutul paginii \*/*

}

**.menu-overlay.active** {

**display**: **block**;

}

*/\* Când nav-list are clasa .active, va fi afișat \*/*

**ul** **nav** **.nav-list.active** {

**display**: flex;

transform: translateX(165%); */\* Aduce meniul în vizual \*/*

}

Aceste stiluri sunt aplicate atunci când dimensiunea ecranului este mai mică de 768 pixeli, asigurând o experiență optimizată pentru utilizatorii de dispozitive mobile. În cazul nostru, media query efectuează următoarele modificări:

* Afișarea butonului hamburger: Pe ecranele mici este setat să fie vizibil folosind variabila ‘display: block’. Acesta e poziționat absolut in colțul dreapta sus al ecranului, permițând utilizatorilor să deschidă meniul de navigare.
* Stilizarea meniului de navigare: Meniul de navigare ‘.nav-list’ este poziționat fix și începe din afara ecranului ‘left:-110%’, acesta se transformă într-o poziție vizibilă pe ecran cu ajutorul unei tranziți de 0.3 secunde.
* Overlay-ul Meniului: Un overlay semi-transparent este afișat pentru a estompa fundalul atunci când meniul este activ. Acest lucru ajută la evidențierea meniului și oferă o experiență de utilizator mai concentrată.

2.2.3.Utilizarea JavaScript

JavaScript este un limbaj de scripting utilizat pentru a crea și controla conținut dinamic pe un site web, care spre deosebire de limbajele de marcare precum HTML, care definește structura paginii, și CSS, care se ocupă de stilizare, JavaScript este responsabil pentru comportamentul paginii web. Acesta permite adăugarea de funcții interactive, răspunderea la evenimentele utilizatorului și să manipuleze elementele paginii în timp real.

Meniul de navigare tip hamburger, este o soluție populară pentru dispozitivele mobile, permițând utilizatorilor să acceseze meniul de navigare printr-un simplu buton. JavaScript este utilizat pentru a deschide și inchide meniul, îmbunătățind astfel experiența de navigare.

**function** toggleMenu() {

**var** navList = document.querySelector('.nav-list');

**var** menuOverlay = document.getElementById('menu-overlay');

*// Toggle clase pentru a arăta sau ascunde meniul și overlay-ul*

navList.classList.toggle('active');

menuOverlay.classList.toggle('active');

}

Acest cod javascript, permite deschiderea și inchiderea meniului de navigare atunci când utilizatorul apasă pe butonul hamburger sau pe overlay-ul meniului.

Caruselul de imagini folosind Swiper JS, o bibliotecă JavaScript modernă și ușor de utilizat, aduce un element interactiv site-ului. Astfel permite prezentarea eficientă și elegantă a portofoliului foto, captând atenția vizitatorilor și menținându-i implicați.

Biblioteca Swiper JS este încarcată direct dintr-un CDN (Content Delivery Network) pentru a optimiza performanța, reduce timpii de încarcare și asigură accesibilitate rapidă si fiabilă

<script src="https://cdn.jsdelivr.net/npm/swiper@8/swiper-bundle.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/swiper@8/swiper-bundle.min.js"></script>

JavaScript nu este limitat doar la funcționalitățile discutate în acest subcapitol. În subcapitolul următor, vom explora cum acesta poate fi utilizat pentru a facilita interacțiunea cu backend-ul, inclusiv manipularea datelor, actualizarea dinamică a conținutului paginilor și multe altele.

**2.2.4.Interacțiunea cu Backend-ul**

Interacțiunea dintre front-end și backend este esențială pentru funcționarea dinamică a site-urilor web moderne. În cadrul site-ului nostru, utilizăm PHP și JavaScript pentru a comunica eficient cu serverul, gestionând solicitările și răspunsurile în timp real. Vom discuta modul în care aceste tehnologii facilitează această interacțiune, asigurându-se că datele sunt manipulate corect și că utilizatorii beneficiază de o experiență fluidă și interactivă.

Utilizarea Google Maps API pentru calcularea distanțelor, ajută la îmbunătățirea functionalității site-ului, astfel se ocupă de calcularea distanțelor și costurile de transport. Acesta permite utilizatorilor să introducă locația evenimentului și sa obțină un cost al transportului.

**function** initMap() {

*// Inițializează funcționalitatea de căutare a locațiilor dacă este necesar.*

**var** input = document.getElementById('eventLocation');

**var** autocomplete = **new** google.maps.places.Autocomplete(input);

}

Funcția ‚initMap’ este responsabilă pentru inițializarea funcționalității de căutare a locațiilor. Aceasta utilizează serviciul ‘google.maps.places.Autocomplete’ pentru a permite utilizatorilor să caute și să selecteze locații direct în formularul de rezervare.

Calcularea distanței între locații

**function** calculateDistance() {

**return** **new** Promise((resolve, reject) => {

**var** origin = 'Grozav Photography, Str. Independenței, Ocna Mureș 515700';

**var** destination = document.getElementById('eventLocation').value;

**var** service = **new** google.maps.DistanceMatrixService();

service.getDistanceMatrix({

origins: [origin],

destinations: [destination],

travelMode: 'DRIVING',

unitSystem: google.maps.UnitSystem.METRIC,

}, **function**(response, status) {

**if** (status !== 'OK') {

alert('A apărut o eroare la calculul distanței: ' + status);

reject(status);

} **else** {

**var** distance = parseFloat(response.rows[0].elements[0].distance.value) / 1000; *// Distanța în kilometri*

**var** distanceCost = distance \* 1; *// Presupunem că adăugăm 1 leu per kilometru*

resolve(distanceCost); *// Rezolvă promisiunea cu costul distanței*

}

});

});

}

* Serviciul ‘google.maps.places.Autocomplete’ permite utilizatorilor să caute locații folosind input-ul text. Acesta sugerează locații pe măsură ce utilizatorul introduce text, îmbunătățind experiența de căutare și reducând erorile de introducere.
* ‘google.maps.DistanceMatrixService’ calculează distanțele dintre două sau mai multe locații folosind diverse moduri de transport. În cazul nostru, modul de transport utilizat este ‘DRIVING’ (cu autovehiculul), iar unitățile de măsură sunt metrice (kilometri).
* Funcția ‘calculateDistance’ returnează o promisiune, care permite gestionarea asincronă a calculului distanței. În cazul în care calculul are succes, promisiunea este rezolvată cu costul distanței. În caz de eroare, promisiunea este respinsă și se afișează un mesaj de eroare.

Actualizarea stării formularului și calculul totalului

Funcțiile ‘updateFormStatus’ și ‘calculateTotal’ gestionează starea formularului și calculează costul total al rezervării, inclusiv costul distanței.

**function** updateFormStatus() {

**var** formInputs = document.querySelectorAll('#bookingForm input, #bookingForm select, #bookingForm textarea');

**var** termsCheckbox = document.getElementById('terms');

**var** gdprCheckbox = document.getElementById('gdpr');

**var** submitBtn = document.getElementById('submitBtn');

**var** summaryContainer = document.getElementById('summary');

**var** allFieldsCompleted = **Array**.from(formInputs).every(input => input.value.trim() !== '' && input.checkValidity());

**var** termsAccepted = termsCheckbox.checked && gdprCheckbox.checked;

**var** paymentMethodInputs = document.querySelectorAll('#bookingForm input[name="paymentMethod"]');

submitBtn.disabled = !(allFieldsCompleted && termsAccepted);

**if** (allFieldsCompleted && termsAccepted) {

calculateTotal(); *// Aceasta va actualiza sumarul dacă toate condițiile sunt îndeplinite*

} **else** {

summaryContainer.style.display = 'none';

}

}

*// Se așteaptă încărcarea completă a DOM-ului pentru a atașa event listeners*

document.addEventListener('DOMContentLoaded', **function**() {

*// Event listeners pentru actualizarea stării formularului*

document.querySelectorAll('#bookingForm input, #bookingForm select, #bookingForm textarea').**forEach**(input => input.addEventListener('input', updateFormStatus));

document.getElementById('terms').addEventListener('change', updateFormStatus);

document.getElementById('gdpr').addEventListener('change', updateFormStatus);

paymentMethodInputs.**forEach**(input => {

input.addEventListener('change', **function**() {

console.log('Metoda de plată selectată.');

calculateTotal();

});

});

});

Funcția ‘updateFormStatus’ verifică dacă toate câmpurile formularului sunt completate corect și dacă au fost acceptate termenii și condițiile, precum politica GDPR. Totodată, aceasta e răspunzătoare de activarea butonului de trimitere daca toate condițiile sunt îndeplinite și apelează funcția ‘calculateTotal’ pentru a calcula costul total al rezervării.

Calculul total este responsabil pentru calcularea costului total al unei rezervări.

async **function** calculateTotal() {

**return** **new** Promise((resolve, reject) => {

**var** eventType = document.getElementById('eventType').value;

**var** services = document.getElementById('services').value;

**var** memoryBoxChecked = document.getElementById('memoryBox').checked;

**var** guestCount = parseInt(document.getElementById('guestCount').value) || 0; *// Asigură-te că este un număr*

**var** paymentMethod = document.querySelector('input[name="paymentMethod"]:checked') ? document.querySelector('input[name="paymentMethod"]:checked').value : 'default\_value';

**var** totalCost = 600;

**var** discountApplied = **false**;

**switch** (eventType) {

**case** 'majorat':

totalCost += 80;

**break**;

**case** 'botez':

totalCost += 400;

**break**;

**case** 'nunta':

totalCost += 600;

**break**;

}

**switch** (services) {

**case** 'foto+video':

totalCost += 600;

**break**;

**case** 'doar foto':

totalCost += 300;

**break**;

**case** 'doar video':

totalCost += 400;

**break**;

}

**if** (memoryBoxChecked) {

totalCost += 150;

}

**var** costPerGuest = 15;

totalCost += guestCount \* costPerGuest;

**if** (guestCount > 200) {

totalCost -= totalCost \* 0.05;

discountApplied = **true**;

}

**var** deposit = totalCost \* 0.15;

calculateDistance().then(distanceCost => {

totalCost += distanceCost;

deposit = totalCost \* 0.15; *// Recalcularea avansului după adăugarea costului distanței*

**var** summary = `Tip Eveniment: ${eventType}<br>Servicii Selectate: ${services}<br>`;

summary += `Costul distanței: ${distanceCost.toFixed(2)} lei<br>`;

summary += paymentMethod === 'cash' ? `De Plătit (Avans 15%): ${deposit.toFixed(2)} lei<br>` : `De Plătit Astăzi (Avans 15%): ${deposit.toFixed(2)} lei<br>`;

summary += `Total de Plată: ${totalCost.toFixed(2)} lei<br>`;

**if** (discountApplied) {

summary += `Wow! Beneficiezi de 5% discount :)<br>`;

}

document.getElementById('summary').innerHTML = summary;

document.getElementById('summary').style.display = 'block';

updateFormStatus(); *// Actualizează starea formularului după calcul*

resolve({

totalCost: totalCost,

deposit: deposit

});

}).**catch**(error => {

console.error('A apărut o eroare la calculul distanței:', error);

reject(error);

});

Funcția ‘calculateTotal’ calculează costul total al rezervării pe baza tipului de eveniment, serviciile selectate, numarul de invitați și costul distanței. Inițial, funcția setează variabilele necesare și adaugă costuri suplimentare la costul total, in funcție de tipul de eveniment și serviciile selectate.

De asemenea, adaugă un cost de 15 lei per invitat și aplică un discount de 5% daca numărul de invitați depașește 200. Avansul este calculat ca fiind 15% din suma totala adică ‘totalCost’. După calcularea tuturor costurilor, se construiește un sumar al rezervării, care este afișat în elemental cu id-ul ‘summary’. Funcția returnează o promisiune care este rezolvată cu obiectul conținând costul total și avansul. Prin utilizarea promisiunilor și a funcțiilor asincrone, asigurăm ca toate calculele necesare sunt efectuate corect și ca utilizatori primesc informații actualizate și precise în timp real.

Verificarea disponibilități unei date selectate în calendarul interactiv din formular

$(**function**() {

$("#datepicker").datepicker({

dateFormat: 'yy-mm-dd',

beforeShowDay: **function**(date) {

**if** (**new** Date(date) < **new** Date().setHours(0, 0, 0, 0)) {

**return** [**false**, '', 'Data în trecut'];

}

**return** [**true**];

},

onSelect: **function**(dateText, inst) {

checkDateAvailability(dateText); *// verifica disponibilitatea datei*

},

minDate: 1 *// Nu permite selectarea zilelor din trecut*

});

});

Inițializam calendarul cu ajutorul bibloteci jQuery, care simplifică manipularea, gestionarea evenimentelor și realizarea cererilor AJAX. Utilizăm funcția ‘$()’ specifică jQuery, pentru a inițializa calendarul atunci cand pagina a fost complet încarcată

* beforeShowDay: O funcție care împiedică selectarea datelor din trecut. Aceasta returnează fals pentru datele din trecut, prevenind astfel selecția acestora.
* onSelect: O funcție care este apelată atunci cand utilizatorul selectează o dată, aceasta fiind ulterior trimisă către funcția de verificare a disponibilitați.

Funcția de verificare a disponibilități datei selectate

**function** checkDateAvailability(selectedDate) {

$.ajax({

url: 'check\_availability.php',

type: 'POST',

data: { data\_evenimentului: selectedDate },

success: **function**(response) {

response = response.trim();

**if**(response === 'unavailable') {

alert('Această dată este deja rezervată.');

$("#datepicker").datepicker('setDate', **null**); *// Resetează selecția*

$("#datepicker").addClass('unavailable-date');

} **else** {

$("#datepicker").removeClass('available-date');

alert('Data este disponibilă pentru rezervare.');

}

},

error: **function**(xhr, status, error) {

console.error("A apărut o eroare: " + status + ", " + error);

alert('Nu s-a putut verifica disponibilitatea datei. Vă rugăm să încercați din nou.');

}

});

}

Această funcție utlizează metoda ‘$.ajax’ pentru a trimite o cerere POST catre scriptul PHP de verificare a disponibilități, trimițând data selectată ca parametru. Funcția gestionează răspunsul de la server și oferă feedback utilizatorului.

Scriptul PHP responsabil de verificarea propriu-zisă a datei daca este rezervată sau nu

<?php

*// Conectare la baza de date*

$host = "127.0.0.1";

$username = "root";

$password = "";

$dbname = "Grozav\_DB";

$conn = **new** mysqli($host, $username, $password, $dbname);

**if** ($conn->connect\_error) {

**die**("Conexiunea a eșuat: " . $conn->connect\_error);

}

*// Preia data selectată din request*

$selectedDate = $\_POST['data\_evenimentului'];

*// Interogare pentru a verifica dacă data este disponibilă*

$query = "SELECT COUNT(\*) FROM rezervari WHERE data\_evenimentului = ?";

$stmt = $conn->prepare($query);

$stmt->bind\_param("s", $selectedDate);

$stmt->execute();

$stmt->bind\_result($count);

$stmt->fetch();

*// Închide conexiunea*

$stmt->close();

$conn->close();

*// Verifică dacă există rezervări pentru data respectivă*

**if**($count > 0) {

**echo** 'unavailable';

} **else** {

**echo** 'available';

}

?>

Acest script este esențial pentru asigurarea faptului că utilizatorii nu pot rezerva aceeași dată pentru mai multe evenimente, menținând integritatea și funcționalitatea sistemului de rezervări.

* Preluarea datei selectate: Data selectată este preluată din solicitarea POST utilizând ‘$\_POST[‘data\_evenimentului’].
* Interogarea Bazei de date: Pentru a verifica disponibilitatea datei, scriptul pregătește și execută o interogare SQL. Folosim o declarație pregătită pentru prevenirea atacurilor de tip SQL injection. Aceasta verifică numărul de înregistrări din tabelul ‘rezervari’ unde ‘data\_evenimentului’ este egală cu data selectată.
* Gestionarea Rezultatului: Rezultatul interogării este legat la variabila ‘$count’, aceasta verifică valoarea acestei variabile, daca este mai mare decât 0, înseamnă că există deja rezervări pentru acea dată și returnează ‘unavailable’. Daca nu există rezervări pentru data selectată, returnează ‘available’

Detaliile Rezervării și afișarea acestora

Acest script PHP se ocupă de prezentarea detaliilor rezervării după ce aceasta a fost înregistrată cu succes în sistem. După ce un client face o rezervare, acesta este redirecționat către o pagină care afișează un sumar al evenimentului rezervat.

<?php

session\_start();

$mysqli = **new** mysqli("localhost", "root", "", "Grozav\_DB");

**if** ($mysqli->connect\_error) {

**die**("Conexiune eșuată: " . $mysqli->connect\_error);

}

**if** (isset($\_GET['orderNumber']) && isset($\_GET['token'])) {

$\_SESSION['id'] = (int) $\_GET['orderNumber'];

$\_SESSION['token'] = $\_GET['token'];

}

**if** (!isset($\_SESSION['token'], $\_SESSION['id'])) {

**echo** json\_encode(['success' => **false**, 'message' => 'Acces neautorizat sau sesiune expirată.']);

**exit**;

}

$orderNumber = (int)$\_SESSION['id']; *// Convert to integer if not already*

$token = $\_SESSION['token'];

**if** ($stmt = $mysqli->prepare("SELECT r.\*, p.\* FROM rezervari r INNER JOIN programate p ON r.id = p.id WHERE r.id = ? AND r.token = ?")) {

$stmt->bind\_param("is", $orderNumber, $token);

**if** (!$stmt->execute()) {

error\_log("MySQL execute error: " . $stmt->error);

}

$result = $stmt->get\_result();

**if** ($result->num\_rows > 0) {

$row = $result->fetch\_assoc();

**echo** "<h1>Evenimentul a fost rezervat!</h1>";

**echo** "<h3>Pentru confirmarea evenimentului veți fi contactat de unul dintre operatorii noștri!</h3>";

**echo** "<h4>Rezumat Rezervare Eveniment</h4>";

**echo** "<p>Număr Rezervare: " . $row['id'] . "</p>";

**echo** "<p>Tip eveniment: " . $row['tip\_eveniment'] . "</p>";

**echo** "<p>Servicii dorite: " . $row['servicii'] . "</p>";

**echo** "<p>Data Evenimentului: " . $row['data\_evenimentului'] . "</p>";

**echo** "<p>Locație: " . $row['locatie'] . "</p>";

**echo** "<p>Avans de Plătit: " . $row['avans\_de\_platit'] . " RON</p>";

**echo** "<p>Total de Plătit: " . $row['suma\_plata\_totala'] . " RON</p>";

} **else** {

**echo** "<h1>Acess Neautorizat sau rezervare inexistenta!</h1>";

}

$stmt->close();

} **else** {

**echo** "<h1>Eroare la pregătirea interogării.</h1>";

}

$mysqli->close();

?>

Acest script îndeplinește mai multe roluri esențiale pentru funcționarea corectă a sistemului de rezervări

* Gestionarea Sesiunilor: Utilizează sesiuni pentru a menține datele utilizatorilor în siguranță și pentru a preveni accesul neautorizat la detaliile rezervării. ‘session\_start()’ inițiază o sesiune PHP, permițând stocarea și accesarea variabilelor de sesiune pe parcursul sesiunii utilizatorului.
* Verificarea Variabilelor din Sesiune: Acest pas este vital pentru a se asigura ca doar utilizatori ce au facut atunci rezervarea pot accesa detaliile rezervării. Scriptul verifică daca variabilele de sesiune precum ‘orderNumber’ și ‘token’ sunt setate prin metoda GET. Dacă acestea sunt prezente, atunci valorile sunt stocate în variabilele de sesiune.
* Verificarea Sesiunii: Dacă variabilele de sesiune nu sunt setate, atunci returnează un mesaj JSON care indică faptul că accesul este neautorizat sau ca sesiunea a expirat. Astfel protejând datele rezervării de accesul neautorizat.
* Afișarea Detaliilor rezervării: Interogarea returnează un mic sumar al rezervării efectuate anterior in formularul de rezervare, aceste informații sunt extrase si afișate intr-un format HTML. Informațiile afișate includ numărul rezervării, tipul evenimentului, serviciile dorite, data evenimentului, locația, avansul de plătit și totalul de plătit.

Integrarea eficientă a componentelor front-end și backend este esențială pentru funcționarea corectă a unui site web modern. Fiecare componentă contribuie la crearea unui sistem de rezervări robust și interactiv, asigurând o experintă optimă prin integrarea eficientă a frontend-ului și backend-ului.

Astfel, în acest capitol, am detaliat etapele esențiale pentru planificarea și dezvoltarea unui site web, discutând proiectarea și optimizarea bazei de date. Am abordat utilizarea MySQL, crearea tabelelor esențiale, stabilirea relațiilor dintre ele, precum și implementarea masurilor de securitate pentru protejarea datelor utilizatorilor.

Am inclus diagrame și scheme pentru a ilustra structura și funcționalitatea bazei de date Grozav\_DB, evidențiind importanța unei baze de date bine proiectate pentru realizarea temei abordate.

Am discutat despre integrarea frontend-ului, explicând modul în care HTML, CSS și JavaScript construiesc interfața utilizatorului și comunică cu serverul prin intermediul PHP. De asemenea, am detaliat implementarea unui sistem de rezervări, folosind jQuery și AJAX pentru manipularea datelor și asigurarea unei experiențe fluide pentru utilizatori. În plus, am explicat utilizarea API-ului Google Maps Matrix pentru calcularea distanțelor și costurilor de transport.

# 

# **3.TESTAREA FUNCȚIONALITĂȚILOR ȘI AJUSTĂRILE EFECTUATE**

În acest capitol, vom detalia procesul de testare a funcționalitătilor site-ului web, asigurându-ne că toate componentele funcționează corect atât pe desktop cât și pe dispozitive mobile. De asemenea, vom descrie ajustările efectuate pentru optimizarea experienței utilizatorului și remedierea eventualelor probleme identificate.Testarea și ajustările sunt etape cruciale în dezvoltarea unui site web. Procesul de testare este esențial pentru a garanta că site-ul nostru oferă o experiență fluentă și fară erori.

În acest context, vom aborda:

* 1.Ajustările efectuate site-ului
* 2.Testarea funcționalităților site-ului

## 3.1.Ajustările efectuate site-ului

În acest subcapitol, vom discuta ajustările realizate pentru îmbunătățirea experienței utilizatorului. Ajustările includ modificări la CSS pentru a asigura că galeriile de imagini se adaptează corect indiferent de dimensiunea imaginii și a ecranului, ajustării la panoul de administrare și la formularul de rezervare

### 3.1.1. Ajustări la galeriile imagini

Pentru a asigura o afișare corectă și estetică a imaginilor în galeriile de prezentare a momentelor capturate, am aduse mici modificări codului CSS care permit scalarea automată a imaginilor. Astfel, indiferent de dimensiunea originală a imaginii, aceasta va fi redimensionată și încadrată perfect in galeria specifică.

Înainte de ajustare, imaginile nu se potriveau in chenarele lor și erau distorsionate si neclare.



Fig.3.1.1.1. Imagine exemplu inaintea ajustării

După o mică dar importantă modificare adusă codului CSS, imaginile se încadrează perfect și sunt scalate corect.

.swiper-slide img {

display: block;

width: 100%;

height: 40rem;

object-fit: cover;

}



Fig.3.1.1.2. Imagine exemplu după ajustare

3.1.2. Ajustări la panoul de administrare

Panoul de administrare a fost ajustat pentru a oferi administratorilor o interfață clară și eficientă pentru gestionarea evenimentelor. Inițial, panoul era dificil de utilizat pe dispozitive mobile. Astfel, am implementat o interfață optimizată pentru mobil, facilitând administrarea evenimentelor de pe orice dispozitiv.

Înainte de ajustare, panoul era dificil de navigat și utilizat pe dispozitive mobile.

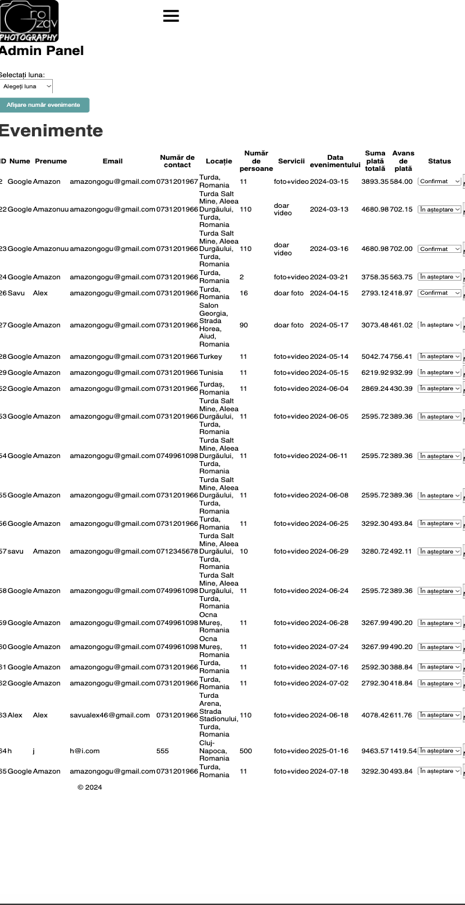


Fig.3.1.2.1. Panoul administrare varianta mobil înaintea ajustării

După ajustarea adusă, interfața a fost mult mai ușor de utlizat și clară.

**@media** **screen** **and** (**max-width**: **768px**) {

*/\* Limităm lățimea tabelului pe dispozitivele cu ecrane mici \*/*

**table** {

**max-width**: 500px;

**margin**: **auto**;

**border-collapse**: **collapse**;

}

*/\* Stilizare pentru fiecare celulă \*/*

**tr** **td** {

**display**: **block**;

**width**: 100%;

**position**: **relative**;

**white-space**: pre-wrap;

**padding-top**: 30px;

**border**: **none**;

**border-bottom**: 1px **solid** #ddd; }

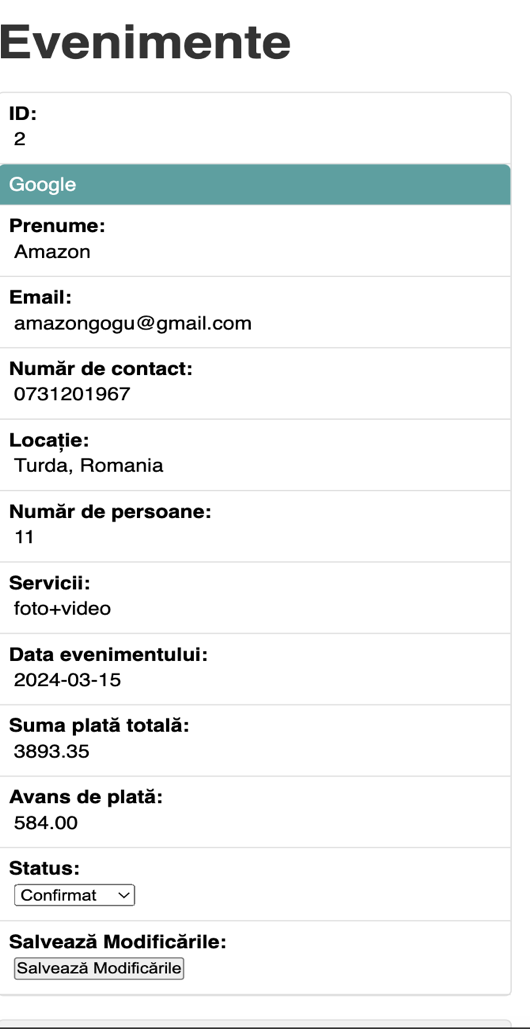


Fig.3.1.2.2. Panoul administrare varianta mobil după ajustare

## 3.2. Testarea Funcționalităților Site-ului

În acest subcapitol, vom explora testarea funcționalităților site-ului, pentru a asigura că toate componentele funcționează corect și eficient. Testele au inclus verificarea funcționării corecte a tuturor funcționalităților majore, de la formularul de rezervare la panoul de administrare.

3.2.1. Testarea Formularului de Rezervare

Am realizat nenumărate teste pentru a verifica funcționalitatea fornularului de rezervare. De exemplu, dacă nu sunt introduse toate câmpurile, sumarul rezervării nu este afișat, totodată butonul de rezervare rămâne neapăsabil.

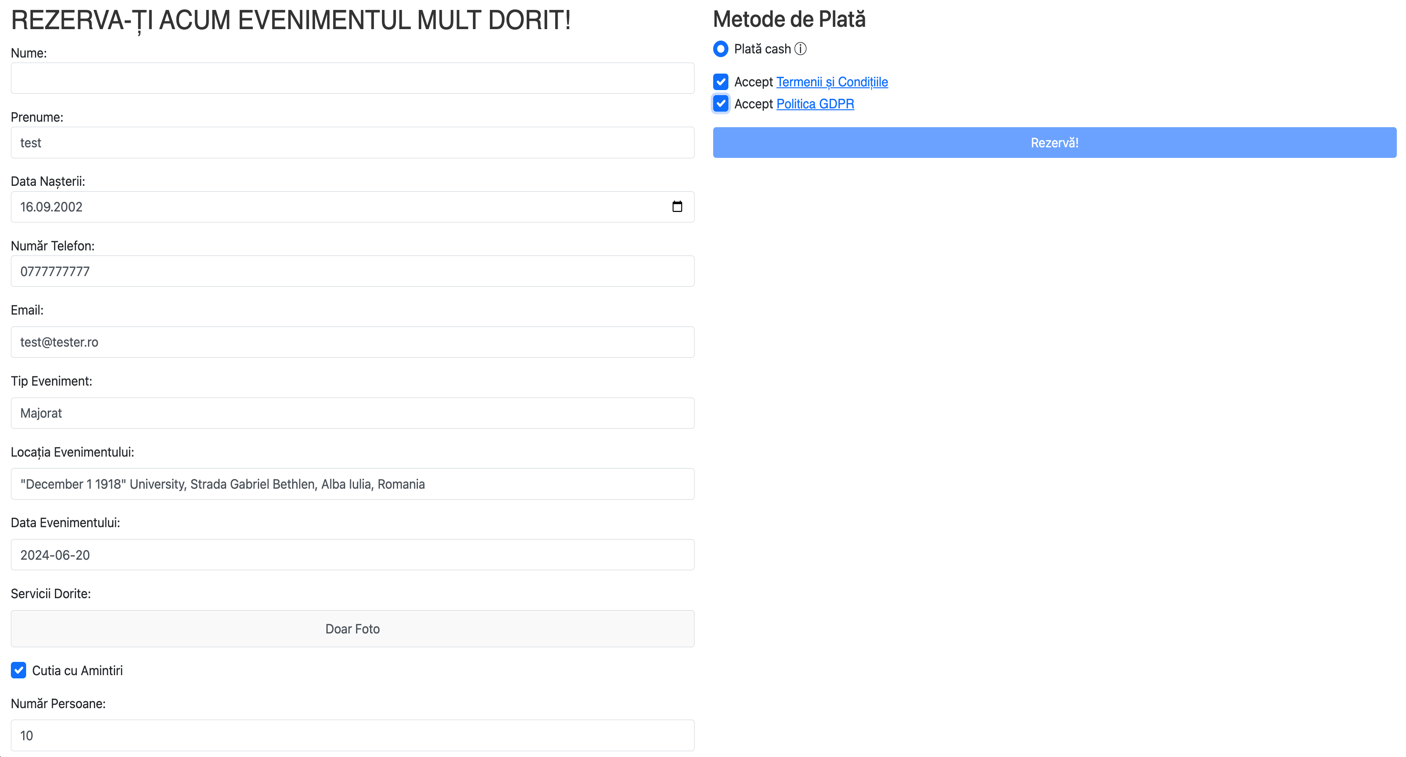


Fig.3.2.1.1.Formular Rezervare necompletat total

Dacă toate datele sunt completate și căsuțele bifate, sumarul este afișat corespunzător si butonul devine apăsabil.

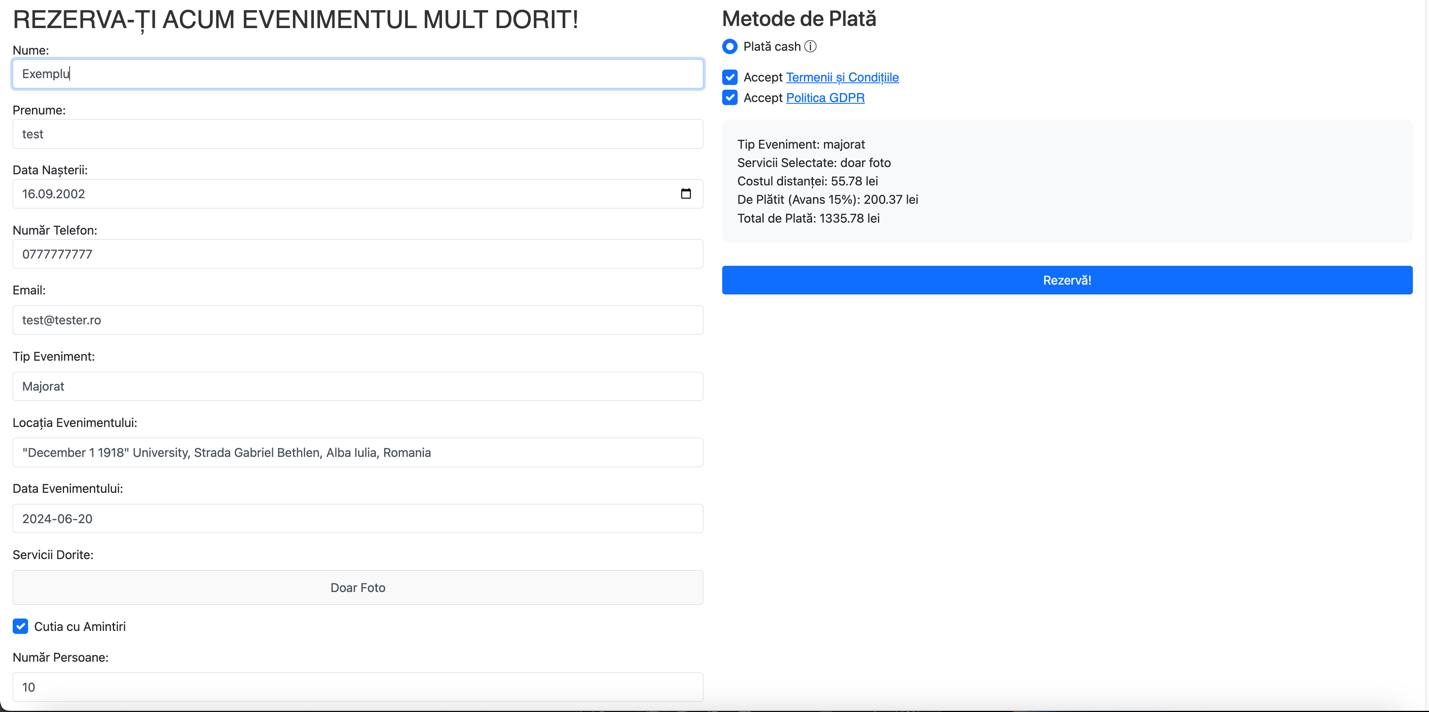


Fig.3.2.1.2.Formular Rezervare completat total

După ce s a efectuat rezervarea , utilizatorul este redirecționat către pagina de detalii a rezervării, unde poate vedea un rezumat complet al rezervării efectuate, incluzând toate informațiile importante. Astfel asigură transparență imediată pentru utilizator.



Fig.3.2.1.3.Eveniment rezervat pagina

Totodată, am testat și mecanismul de securitate al formularului, asigurându-ne că utilizatorul nu poate vizualiza alte rezervări modificând variabilele URL-ului.

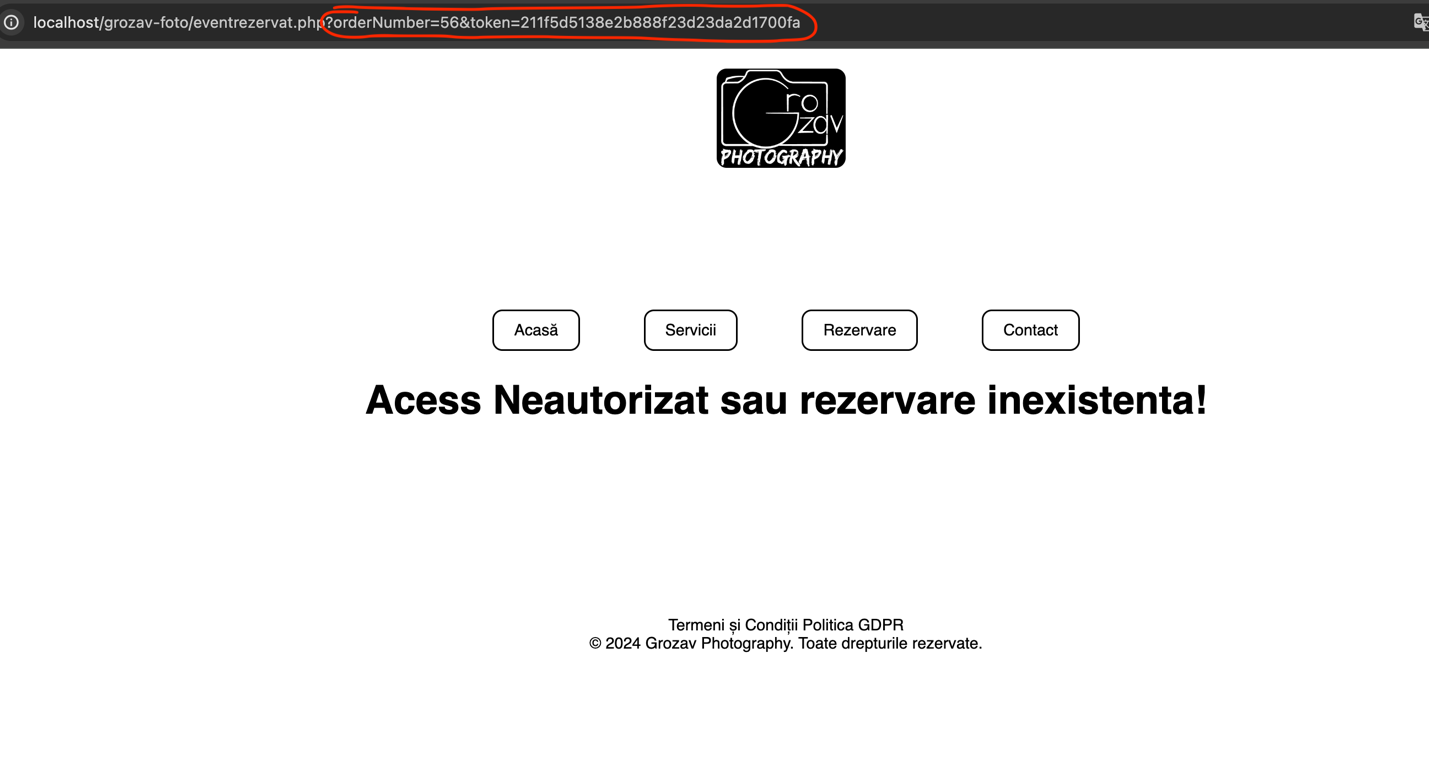


Fig.3.2.1.4. Verificarea securității acces neautorizat

Sistemul validează sesiunea și accesul este permis doar pentru rezervarea specifică a utilizatorului.

3.2.2.Testarea Panoului de Administrare

Am testat panoul de administrare pentru a ne asigura că doar administratori pot accesa aceasta secțiune. Astfel, orice încercare de a accesa pagina de administrare fără autentificare redirecționează utilizatorul către pagina de login.

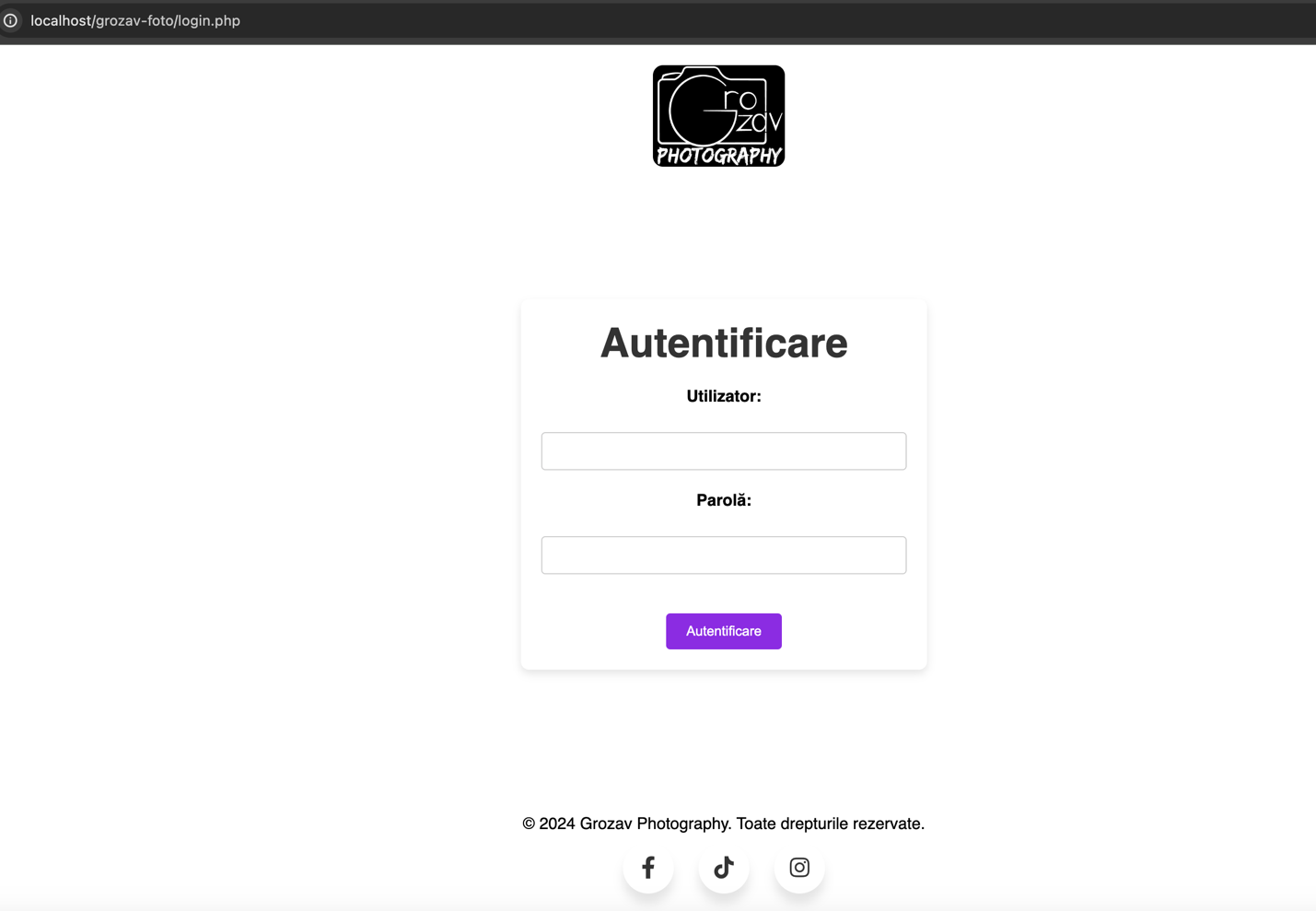


Fig.3.2.2.1.Pagina de autentificare a administratorului

Interfața panoului de administrare a fost testată pentru confirma că administratorii pot vizualiza și gestiona eficient evenimentele.

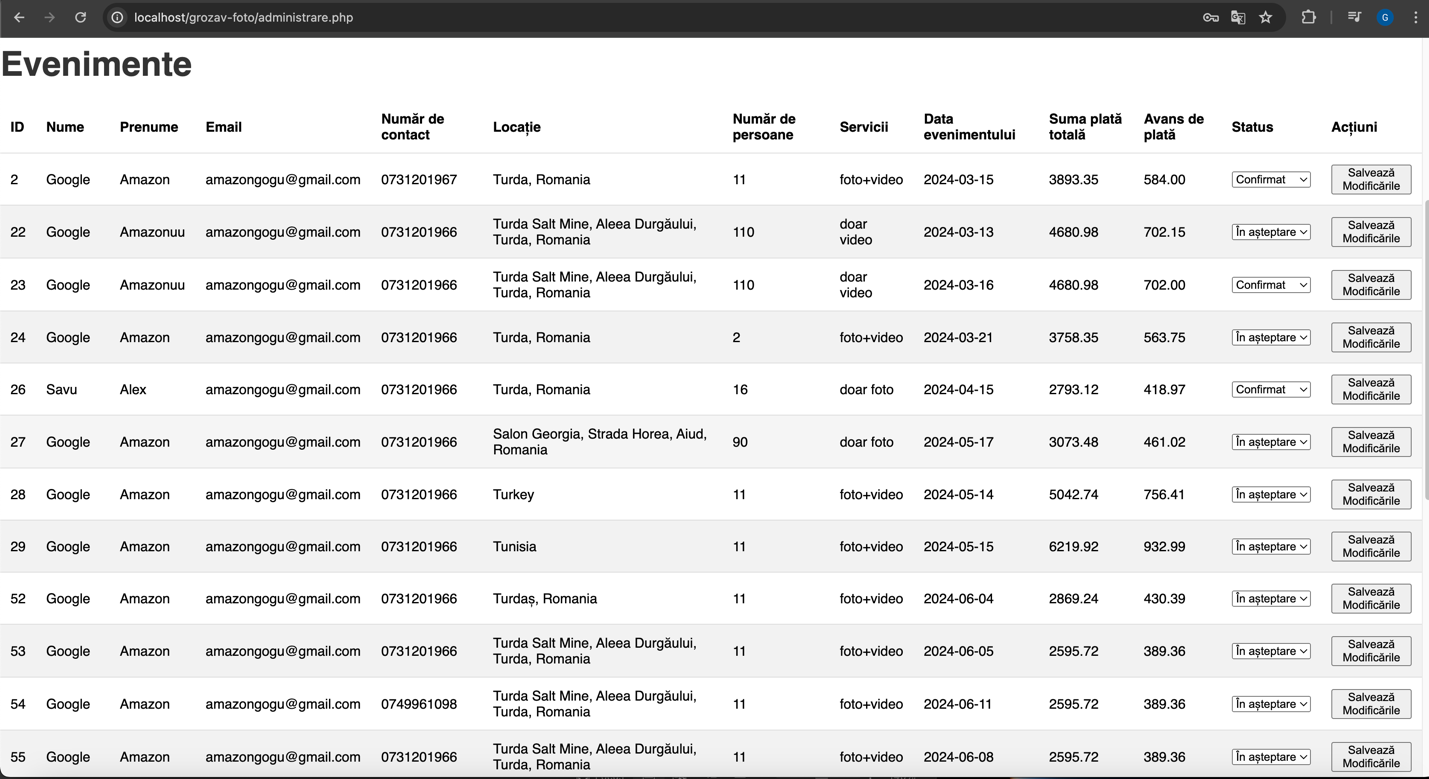


Fig.3.2.2.2. Panoul de administrare și gestionarea evenimentelor

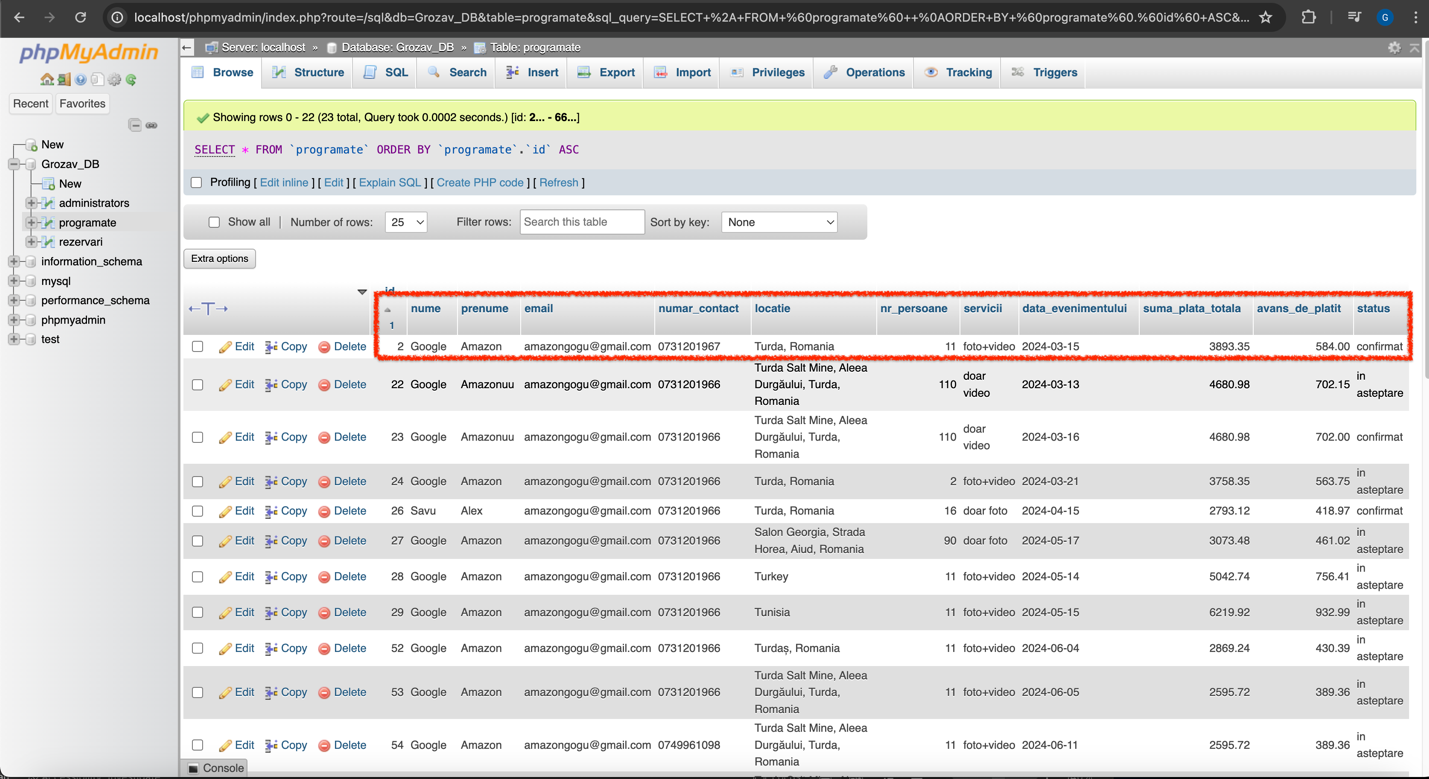


Fig.3.2.2.3.Interfața phpMyAdmin pentru gestionarea bazei de date înaintea modificării

Funcțiile de modificare și actualizare a informațiilor rezervărilor, precum și a statusului aferent, funcționează corespunzător, iar modificările sunt salvate corect in baza de date.

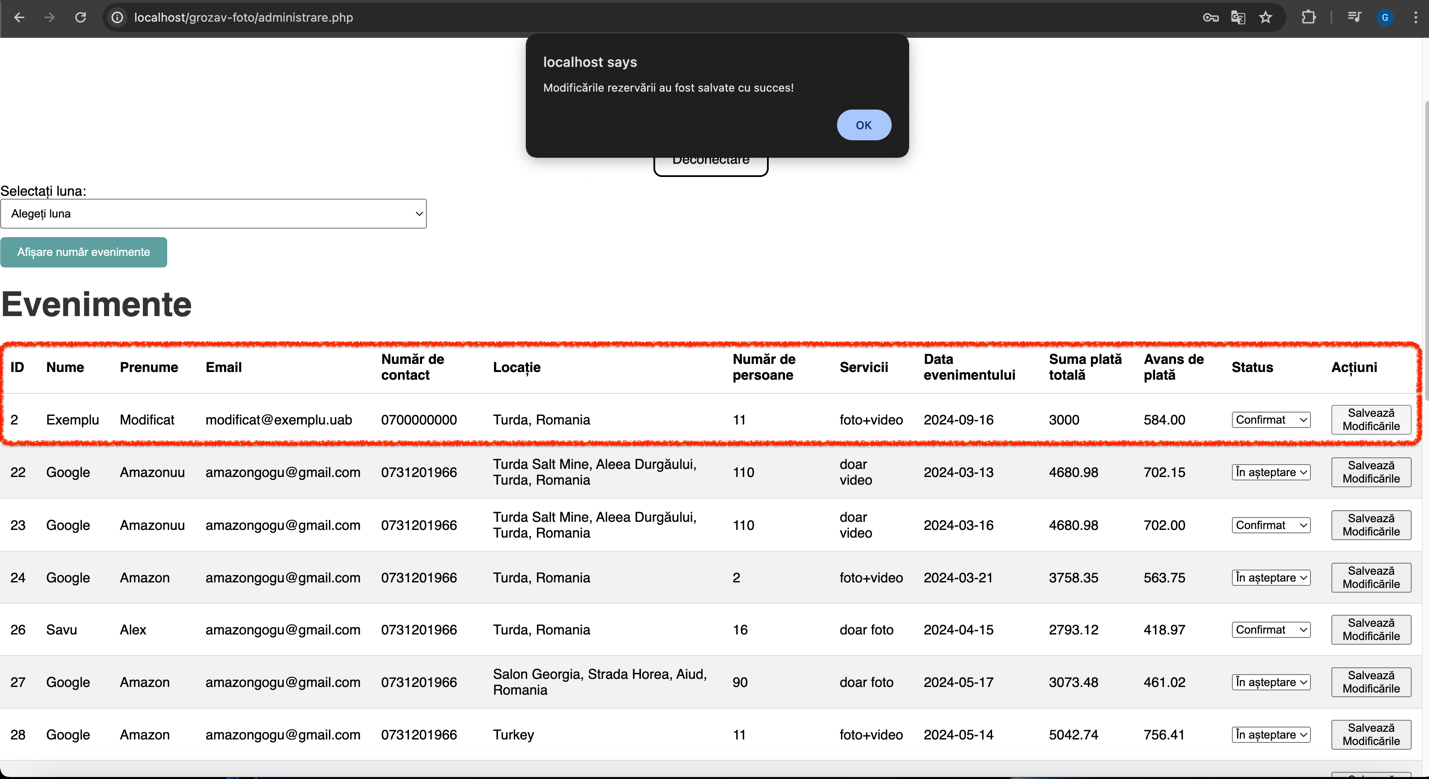


Fig.3.2.2.4.Modificarea informațiilor din panoul de administrare

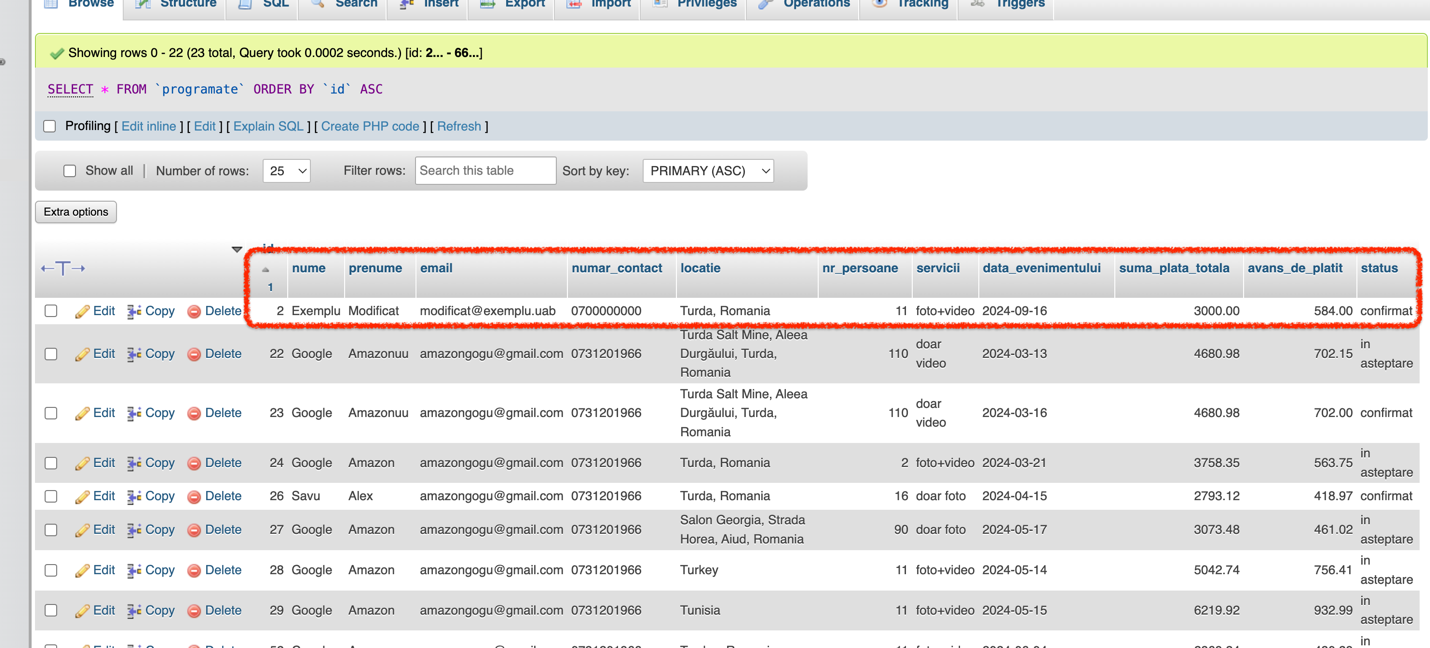


Fig.3.2.2.5. Modificare efectuată și în baza de date

Capitolul de față a detaliat testarea funcționalităților și ajustările efectuate pentru a îmbunătăți experiența utilizatorului pe site-ul nostru. Procesul de testare a inclus verificarea tututor componentelor majore, de la formularul de rezervare la panoul de administrare, asigurând astfel că toate funcționalitățiile sunt corecte și eficiente. Ajustările aduse au fost esențiale pentru a oferi o experiență de utilizare optimă, indiferent de dispozitivul folosit.

# **CONCLUZII ȘI PROPUNERI**

Scopul principal a acestei teme a fost dezvoltarea unui site web pentru prestarea serviciilor foto-video, cu accent pe rezervările online și gestionarea eficientă a evenimentelor.

Astfel pe parcursul realizării acestei teme, am parcurs următoarele etape importante:

* Evaluarea și Alegerea Tehnologiilor: Am analizat diverse alternative tehnologice și am ales soluțiile optime pentru cerințele noastre, utilizând tehnologii precum HTML5, CSS3, PHP și JavaScript, datorită beneficiilor și fiabilității acestora, precum și suportului oferit de comunitatea de dezvoltatori.
* Proiectarea și Implementarea Bazei de Date: Am proiectat și implementat baza de date folosind MySQL, creând tabele esențiale pentru gestionarea rezervărilor și a evenimentelor. Proiectarea bazei de date a inclus realizarea unor diagrame și scheme cu ajutorul PlantUML pentru a ilustra structura și relațiile dintre tabele, asigurând astfel o înțelegere clară și detaliată a modelului de date.
* Integrarea Frontend și Backend, și Implementarea Funcționalităților Avansate: Am integrat frontend-ul cu backend-ul utilizând HTML, CSS și JavaScript pentru a crea o interfață de utilizator interactivă și responsivă. Pe lângă acestea, am implementat funcționalități avansate, cum ar fi calcularea distanțelor folosind Google Maps Matrix API și validarea datelor în timp real, asigurând o experiență optimizată și fară erori.

Pentru viitor, există mai multe direcții de dezvoltare care pot îmbunătăți funcționalitatea și utilitatea site-ului.

* Implementarea plății cu cardul: Integrarea unui sistem de plată online va permite utilizatorilor să efectueze plăți securizate direct prin site, oferind astfel o experiență de rezervare completă și convenabilă
* Automatizarea Notificărilor: Implementarea unui sistem automat de notificări prin email sau sms va informa utilizatorii despre statusul rezervării și despre orice actualizări importante, asigurând astfel o comunicare proactivă și eficientă.
* Feedback și recenzii: Adăugarea unei secțiuni unde clienți pot lăsa recenzii despre serviciile folosite. Astfel va îmbunătății încrederea și interacțiunea cu alți potențiali clienți, oferind o sursă valoroasă de informații pentru viitori utilizatori posibili clienți.

Așadar, în urma celor spuse, aceasta temă a reprezentat o oportunitate de a-mi demonstra că orice idee poate fi adusă la viață daca este gandită. Realizarea acestei teme, în special partea practică, a fost o plăcere și mi-a oferit satisfacția de a vedea cum conceptele teoretice se materializează într-o soluție funcțională și eficientă.

# **BIBLIOGRAFIE**

***Cărţi şi lucrări de autor în edituri***

1. Briciu, Sorin, Contabilitate managerială, Editura Economică, București, 2006.
2. Bruce, J., Learning SQL, O’Reilly Media, California, 2007.
3. Ionescu, A., Introducere în Programarea Web, Editura Universitară, Cluj Napoca, 2021
4. Muntean, Maria, Olteanu, Emil, Baze de date Aplicații (Databases. Applications),Seria didactica Publishing House, 2014.
5. Muntean, Maria, Rîșteiu, Mircea, Sisteme de gestiune a bazelor de date.Aplicații (Database Management Systems Applications), Aeternitas Publishing House, 2013.

***Studii și articole în reviste de specialitate***

1. Brown, D., “The Evolution of JavaScript Frameworks”, Web Dev Today, vol.8, no.2, 2022, pp.10-18.
2. Green, H., Security Best Practices for PHP Applications, Secure Coding Journal, vol.11, no.4, 2023, pp. 25-30.
3. Smith, A., Introduction to HTML5, Modern Web Development Journal, vol.3, no. 1, 2021, pp. 35-47.
4. Johnson, R., CSS3 Techniques for Modern Web Design, CSS Today, vol. 12, no. 5, 2023, pp. 22-28.

***Resurse în format electronic***

1. World Wide Web Consortium (W3C), “HTML5 Specification”, 2014, <https://html.spec.whatwg.org/multipage/>, accesat la 1 aprilie 2024.
2. MySQL AB, “MySQL 8.0 Reference Manual”, 2020, <https://dev.mysql.com/doc/refman/8.0/en/> , accesat la 10 Aprilie 2024.
3. Mozilla Developer Network (MDN), „JavaScript Guide”, 2023, <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide> , accesat la 15 Aprilie 2024.
4. Stack Overflow, “How to use JavaScript Promises”, 2024, <https://stackoverflow.com/questions/14220321/how-do-i-return-the-response-from-an-asynchronous-call> , accesat la 15 Aprilie 2024
5. Smashing Magazine, “CSS”, 2024, <https://www.smashingmagazine.com/category/css/> ,accesat la 30 mai 2024.
6. GitHub, “PHP: The Right Way”, 2023, <https://github.com/ggelashvili/learnphptherightway-outline> , accesat la 15 Mai 2024.
7. freeCodeCamp.org, “Introduction to responsive web design- HTML & CSS tutorial”, Youtube, 2019, https://www.youtube.com/watch?v=srvUrASNj0s ,accesat la 20 aprilie 2024.