

Universitatea Națională de Știință și Tehnologie POLITEHNICA București
Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei

Tehnologii de Programare in Internet

Ghid Turistic

Studenti:

Serbanescu Daniela-Cristina
Budur Maria

Grupa: 434D

Profesor coordonator:

Ş.l. dr. ing Boicescu Laurentiu

1. Introducere

1.1 Obiectivul lucrării

Alegerea realizării unei aplicații web pentru un ghid turistic este fundamentată pe nevoia reală de digitalizare a serviciilor din domeniul turismului și pe potențialul semnificativ al acestui sector de a contribui la dezvoltarea economică locală și regională. Într-o societate în continuă digitalizare, utilizatorii caută soluții rapide, eficiente și accesibile pentru planificarea călătoriilor. În acest context, o aplicație web dedicată informării turistice reprezintă o inițiativă practică, actuală și necesară.

Turismul constituie o componentă esențială a economiei pentru numeroase localități și stațiuni din România, oferind oportunități de promovare culturală, naturală și economică. O aplicație web care centralizează informații despre obiective turistice, posibilități de cazare, restaurante, trasee sau evenimente locale poate deveni un instrument util nu doar pentru turiști, ci și pentru autoritățile locale și afacerile din domeniul ospitalității.

1.2 Caracteristicile cheie ale aplicatiei

Un avantaj major al aplicației web constă în faptul că este accesibilă de pe orice dispozitiv conectat la internet, indiferent de sistemul de operare sau locația utilizatorului. Fiind o aplicație bazată pe web, aceasta nu necesită instalare și poate fi utilizată direct din browser, ceea ce o face ușor de folosit, rapidă și eficientă. În plus, datorită tehnologiilor responsive utilizate în dezvoltarea interfeței (HTML, CSS, JavaScript, PHP), aplicația se adaptează automat la dimensiunea ecranului, oferind o experiență plăcută pe desktop, tabletă sau telefon mobil.

Un alt aspect important al aplicației este funcționalitatea de autentificare a utilizatorilor. Aceștia pot crea un cont personal, completând un formular cu informații precum nume de utilizator (username) și parolă. Aceste date sunt preluate de aplicație și stocate dinamic într-o bază de date relațională (MySQL), prin intermediul scripturilor PHP. Sistemul de autentificare permite personalizarea experienței utilizatorului și accesul la funcționalități suplimentare, precum adăugarea de recenzii.

1.3 Utilizatori tinta

Tendințele actuale arată că tot mai mulți turiști utilizează internetul pentru a-și planifica vacanțele. Aplicația web se adresează în primul rând turiștilor, atât celor din România, cât și vizitatorilor internaționali, care doresc să planifice eficient vacanțele și să descopere informații actualizate despre destinațiile turistice. Acești utilizatori caută o soluție practică și rapidă pentru a identifica obiective turistice, posibilități de cazare, restaurante și evenimente locale. De asemenea, aplicația se adresează agenților economici din domeniul ospitalității, precum hoteluri, pensiuni, restaurante sau organizatori de evenimente, care pot beneficia de promovare prin includerea în platformă.

Pe lângă funcția de informare și organizare a vacanței, aplicația are și un rol important în promovarea turismului local, în special în cazul stațiunilor mai puțin cunoscute sau a atracțiilor turistice locale care nu beneficiază de suficiente resurse pentru promovare tradițională. Prin includerea acestora în platformă, aplicația contribuie la creșterea vizibilității lor, atragerea de turiști și, implicit, la dezvoltarea economică regională.

2. Proiectare

2.1 Schema bloc

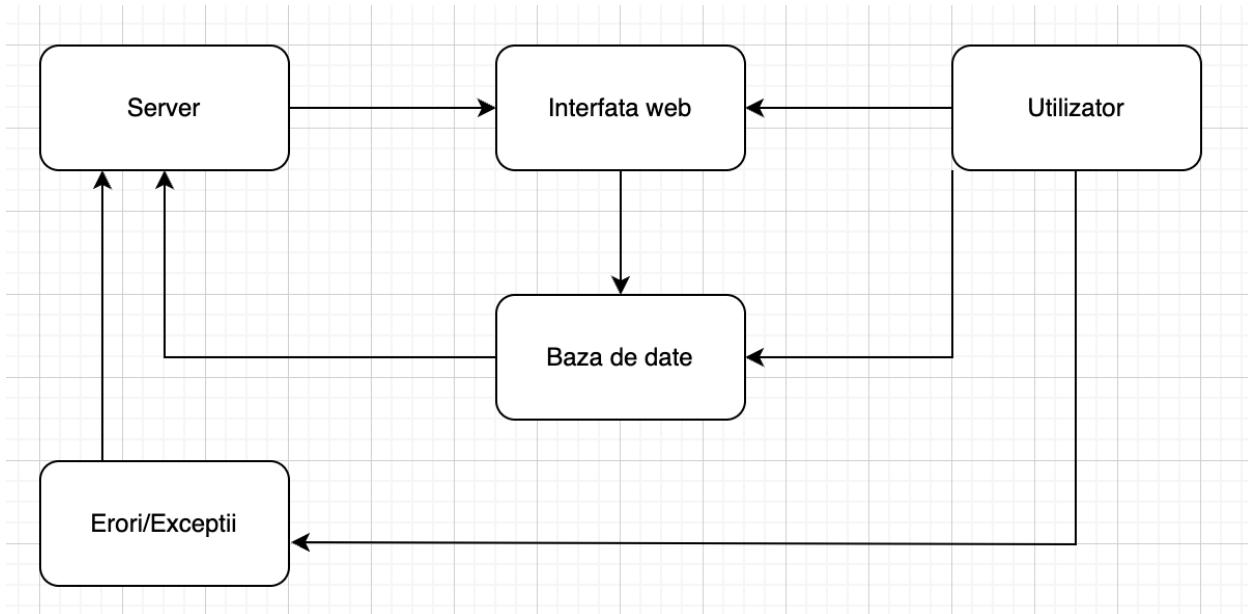


Figura 1. Schema bloc

Etapa de proiectare are un rol esențial în dezvoltarea unei aplicații web eficiente, încrucișând stabilește fundamentele arhitecturale, structura funcțională, logica de comunicare între componente și modul de interacțiune al utilizatorului cu sistemul. Proiectarea corectă a acestor elemente asigură o implementare coerentă, modulară și scalabilă.

În etapa de proiectare a aplicației web de tip ghid turistic, se definesc componentele arhitecturale ale sistemului, modul de interacțiune al utilizatorilor cu aplicația, precum și structura funcțională a acesteia. Aplicația este concepută după modelul clasic client-server, având la bază o arhitectură modulară și scalabilă.

Serverul reprezintă componenta centrală a aplicației și este responsabil de gestionarea logicii de procesare, a interacțiunilor cu baza de date și a comunicării dintre utilizator și conținutul stocat. Cererile trimise de utilizatori prin intermediul interfeței web sunt procesate pe server cu ajutorul scripturilor PHP. Acestea preiau datele, le verifică, le trimit către baza de date (sau le extrag din aceasta) și returnează rezultatul sub formă de pagini PHP.

Utilizatorul care trebuie să beneficieze de un dispozitiv care are conectivitate la internet, dar și de conexiune la internet. Aplicatia noastră dispune de două tipuri de conectare. Fie ca vizitator, fie ca utilizator. Ca vizitator poate să vadă informația pe care utilizatorul o salvează într-o baza de date, iar ca utilizator poate să își editeze, stearge, insereze datele din baza de date, dar să le și vizualizeze.

Interfața web reprezintă componenta vizuală a aplicației, cu care utilizatorii interacționează direct. Aceasta oferă utilizatorilor o interfață intuitivă pentru vizualizarea destinațiilor, orașelor, accesarea locurilor de vizitat, hotelurilor și restaurantelor ce pot fi găsite în acele orașe.

Baza de date se ocupă cu stocarea informațiilor și detaliile utilizatorilor ce se pot autentifica, dar și cu vizualizarea lor prin intermediul serverului de phpmyadmin. Una dintre responsabilitățile bazei de date este de a asigura persistența și integritatea datelor într-un mod vizual și intuitiv.

Blocul pentru gestionarea erorilor și exceptiilor din aplicația web are rolul de a asigura funcționarea stabilă și sigură a platformei. Acesta monitorizează constant execuția aplicației și identifică eventualele erori apărute în timpul utilizării. Erorile detectate sunt înregistrate în log-uri, facilitând procesul de depanare. În cazul apariției unor probleme, sistemul afișează mesaje de eroare clare. Prin tratarea controlată a exceptiilor, aplicația evită blocajele sau intreruperile bruste. Astfel, se menține o interfață coerentă și funcțională. Acest sistem contribuie la creșterea fiabilității și profesionalismului aplicației.

Aplicația este dezvoltată folosind o arhitectură modulară, în care fiecare componentă are un rol bine definit:

- Frontend-ul este realizat utilizând HTML, CSS și JavaScript, asigurând prezentarea informațiilor într-un mod accesibil și responsive.
- Backend-ul este implementat în PHP și preia responsabilitatea prelucrării cererilor utilizatorilor, efectuării de operații asupra bazei de date și returnării rezultatelor către interfața web.

Această structură permite scalarea ușoară a aplicației, adăugarea de noi funcționalități și mențenanță eficientă.

2.2 Propunere interfață grafică



Figura 2. Pagina de start

Interfața grafică a aplicatiei este simplă, intuitivă și organizată, cu un meniu lateral pentru navigare ușoară între secțiuni. Imaginea de fundal ilustrativă și titlul „România” evidențiază tematica turistică. Designul este receptiv și adaptat pentru toate tipurile de dispozitive.

Structura paginii include un meniu de navigare lateral, care permite acces rapid la secțiuni precum „Orașe”, „Cazare”, „Restaurante” și „Recenzii”. Butoanele sunt evidențiate printr-un efect de hover pentru o experiență vizuală plăcută.

2.3 Schema de navigabilitate

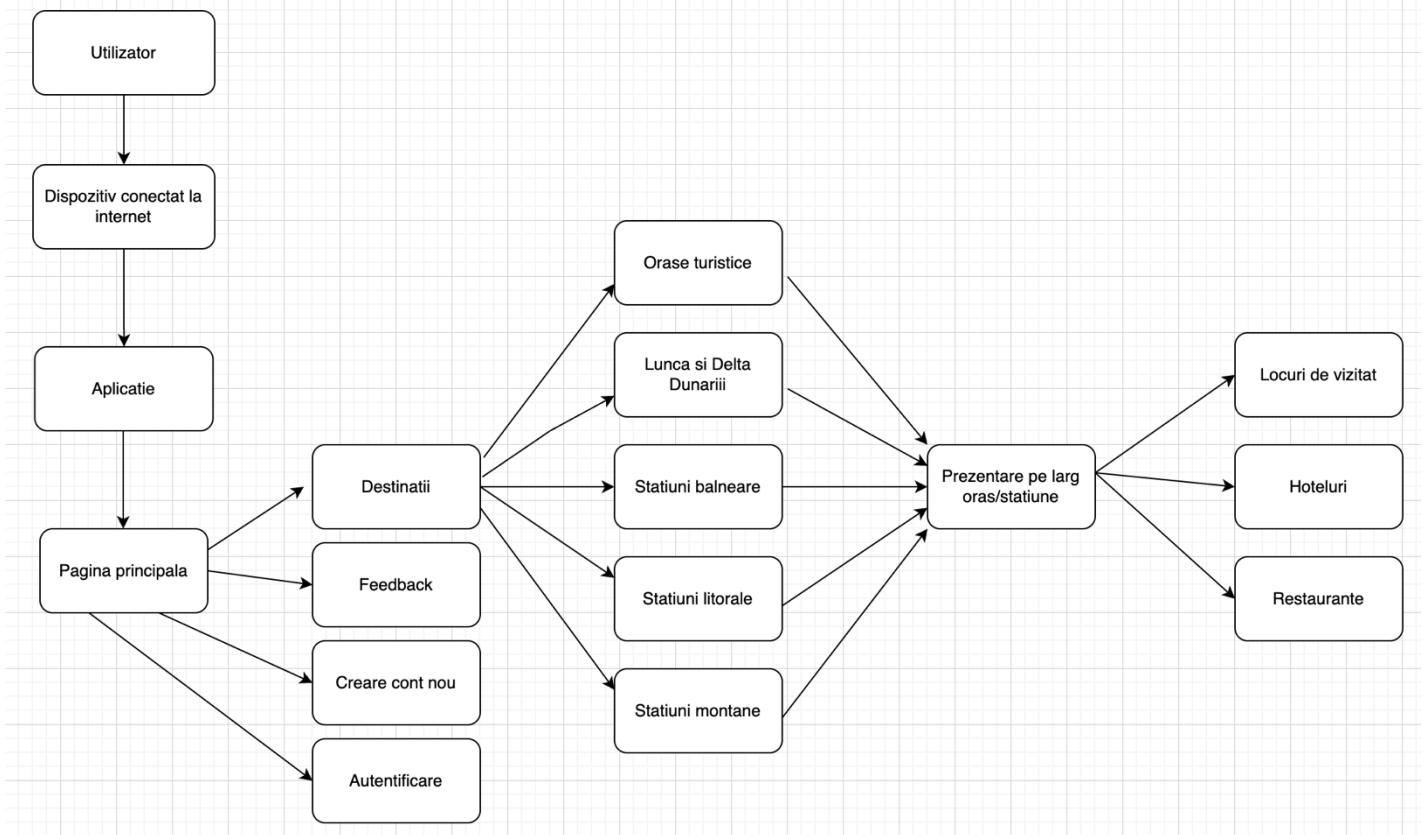


Figura 3. Schema de navigabilitate

Schema de navigabilitate reflectă structura logică a aplicației și traseul utilizatorului de la accesarea platformei până la obținerea informațiilor detaliate despre destinațiile turistice. Mai jos sunt explicate funcțiile fiecărui bloc:

Utilizator: Acest bloc reprezintă persoana care interacționează cu aplicația. Utilizatorul poate fi orice persoană interesată să obțină informații turistice despre diverse destinații din România, să își creeze un cont, să lase feedback sau să exploreze locuri de vizitat, hoteluri și restaurante.

Dispozitiv conectat la internet: Acest bloc simbolizează mijlocul tehnic prin care utilizatorul accesează aplicația. Poate fi vorba despre un smartphone, o tabletă sau un calculator cu conexiune la internet, condiție necesară pentru funcționarea aplicației.

Aplicație: Blocul denumit „Aplicație” desemnează platforma digitală propriu-zisă, care centralizează toate funcționalitățile. Aplicația este cea care oferă acces la paginile și informațiile prezentate, într-o interfață prietenoasă și interactivă.

Pagina principală: Punctul de plecare în aplicație, reprezintă ecranul de start al aplicației, punctul central de unde utilizatorul poate accesa toate celelalte secțiuni importante, cum ar fi destinațiile turistice, feedback-ul, autentificarea sau crearea unui cont nou.

Destinații: Redirecționează utilizatorul către o listă de categorii turistice. Aceste categorii sunt grupate în funcție de tipul locației: orașe turistice, stațiuni montane, stațiuni balneare, stațiuni litorale sau zona Deltei Dunării.

Feedback: Acest bloc permite utilizatorilor să transmită opinii și sugestii despre aplicație sau despre locurile vizitate. Este o funcționalitate utilă pentru îmbunătățirea aplicației și pentru colectarea de impresii reale din partea turiștilor.

Creare cont nou: Interfață permite utilizatorilor să transmită opinii și sugestii despre aplicație sau despre locurile vizitate. Este o funcționalitate utilă pentru îmbunătățirea aplicației și pentru colectarea de impresii reale din partea turiștilor. Utilizatorii existenți introduc datele pentru a se conecta și a avea acces la funcții suplimentare (ex. adăugare comentarii).

Autentificare: Acest bloc oferă posibilitatea utilizatorilor deja înregistrati să se conecteze în aplicație prin introducerea datelor de autentificare (email și parolă). După conectare, utilizatorii pot accesa secțiuni personalizate ale aplicației.

Orașe turistice: Subsecțiune din „Destinații” conține informații despre localitățile din România cu valoare culturală, istorică și turistică ridicată.

Lunca și Delta Dunării: Blocul include informații despre o regiune unică din România, cunoscută pentru biodiversitate, peisaje naturale spectaculoase și turism ecologic. Utilizatorii pot descoperi localități din delta, activități recreative și trasee pe apă.

Stațiuni balneare: Grupează locații renumite pentru apele minerale și tratamentele terapeutice. Utilizatorii pot accesa informații despre servicii de spa, centre de sănătate și activități relaxante.

Stațiuni litorale: Se referă la promovarea stațiunilor aflate pe litoralul Mării Negre. Utilizatorii pot găsi detalii despre plaje, hoteluri, restaurante și activități recreative specifice sezonului estival.

Stațiuni montane: Include informații despre zone montane ideale pentru drumeții, sporturi de iarnă sau relaxare în natură. Se regăsesc aici stațiuni precum Sinaia, Bușteni, Predeal sau Râncă.

Prezentare pe larg oraș/stațiune: Pagina care oferă o descriere detaliată a unei destinații turistice, fie că este vorba despre un oraș, fie despre o stațiune. Aici sunt incluse informații generale, istorie, obiective turistice, activități specifice și fotografii reprezentative.

Locuri de vizitat: Acest bloc oferă o descriere detaliată a unei destinații turistice, fie că este vorba despre un oraș, fie despre o stațiune. Aici sunt incluse informații generale, istorie, obiective turistice, activități specifice și fotografii reprezentative.

Hoteluri: Blocul oferă informații despre opțiunile de cazare din destinația respectivă. Sunt prezentate detalii precum clasificarea hotelurilor, prețuri, facilități disponibile și eventuale recenzii.

Restaurante: Acest bloc pune la dispoziția utilizatorilor o listă cu restaurante din orașul sau stațiunea aleasă. Informațiile pot include tipul de bucătărie, programul de funcționare, locația și sugestii de preparate.

Principalele caracteristici ale aplicatiei:

- Interfață intuitivă și prietenoasă:** Aplicația este concepută pentru a fi ușor de utilizat, indiferent de vârstă sau experiență tehnologică a utilizatorului. Navigarea se face logic, prin meniuri clare și elemente vizuale bine structurate.
- Categorii turistice bine definite:** Destinațiile sunt grupate în funcție de specificul lor: orașe turistice, stațiuni balneare, montane, litorale și Delta Dunării, pentru a facilita alegerea în funcție de interesul utilizatorului.
- Design responsive:** Platforma este optimizată pentru toate tipurile de dispozitive – telefoane, tablete și desktop – asigurând o experiență fluidă și coerentă indiferent de ecran.
- Acces la servicii conexe:** Aplicația oferă linkuri utile sau informații de contact pentru agenții de turism, transport, centre SPA sau puncte de informare turistică.
- Fotografii și hărți interactive:** Fiecare destinație este prezentată cu imagini reprezentative.
- Feedback și recenzii reale:** Aplicația încurajează comunitatea să își exprime părerea despre locurile vizitate, oferind astfel o sursă autentică de informații pentru alți turiști.

3. Implementare

3.1 Implementarea bazei de date

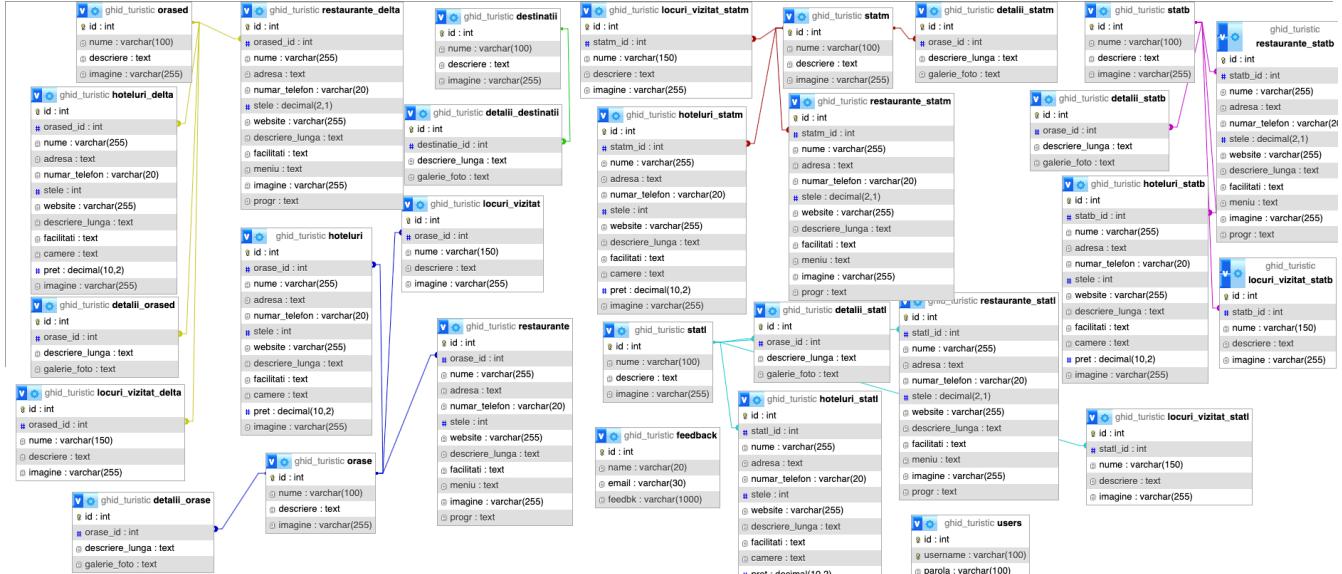


Figura 4. Diagrama EER (Extended Entity-Relationship)

Baza de date pentru aplicația noastră este construită folosind MySQL, un sistem de gestionare a bazelor de date relaționale open-source, recunoscut pentru performanța, scalabilitatea și fiabilitatea sa. MySQL utilizează limbajul standard de interogare SQL (Structured Query Language) pentru a permite manipularea și interogarea datelor în cadrul bazei de date.

Am implementat chei strâine pentru a stabili relațiile între tabele și pentru a asigura integritatea referențială a datelor. Modelarea relațiilor între tabele a fost realizată cu atenție, identificând și definind corect relațiile de tip unu-la-unu, unu-la-mulți între entități. De asemenea, am aplicat principiile normalizării bazelor de date pentru a minimiza redundanța și a asigura coerența și integritatea datelor.

Pentru optimizarea performanței, au fost definite indice (indexuri) pe câmpurile cel mai frecvent utilizate în interogări, ceea ce a contribuit la reducerea timpului de răspuns al aplicației. De asemenea, am folosit constrângeri de unicitate (UNIQUE) pentru a preveni introducerea de date duplicate în anumite câmpuri critice, precum adresele de email ale utilizatorilor. În vederea unei administrații eficiente, am implementat politici de backup periodic al bazei de date, asigurând astfel protecția informațiilor împotriva pierderii accidentale. Întregul model de date a fost proiectat ținând cont de extinderea ulterioară a aplicației, astfel încât adăugarea de noi funcționalități sau entități să poată fi realizată fără impact major asupra structurii existente.

```
CREATE TABLE restante (
    id INT AUTO_INCREMENT PRIMARY KEY,
    orase_id INT NOT NULL,
    nume VARCHAR(255) NOT NULL,
    adresa TEXT NOT NULL,
    numar_telefon VARCHAR(20) NOT NULL,
    stele INT CHECK (stele BETWEEN 1 AND 5),
    website VARCHAR(255),
```

```

descriere_lunga TEXT NOT NULL,
facilitati TEXT NOT NULL,
menui TEXT,
imagine VARCHAR(255),
progr TEXT NOT NULL,
FOREIGN KEY (orase_id) REFERENCES orase(id) ON DELETE CASCADE
);

```

Figura 5. Scriptul pentru crearea tăbelei “locuri de vizitat”

Structura bazei de date a fost proiectată cu accent pe optimizarea interogărilor și a relațiilor între tăbele. S-a realizat o denormalizare controlată în anumite cazuri, acolo unde a fost necesară îmbunătățirea performanței pentru interogările frecvente. Pentru menținerea consistenței datelor la nivel logic, s-au implementat restricții suplimentare precum verificări de validare (CHECK constraints) și proceduri de trigger pentru actualizări automate între entități dependente. Această abordare asigură nu doar eficiența funcționării aplicației, ci și menținerea integrității datelor în scenarii de utilizare intensă.

3.2 Implementarea in Visual Studio Code

Visual Studio Code (VS Code) este un editor de cod sursă gratuit, ușor și puternic, dezvoltat de Microsoft. Deși nu este un IDE complet (Mediu de dezvoltare integrat) precum Visual Studio, funcționează ca unul atunci când este extins cu extensiile și configurațiile potrivite.

Pentru partea de back-end am ales să folosim limbajul de programare **php** deoarece este un limbaj server-side, folosit în principal pentru dezvoltarea de aplicații web dinamice. Oferă o sintaxă clară și ușor de înțeles, precum și un sistem puternic de gestionare a memoriei și de tratare a exceptiilor. Limbajul **php** este utilizat pentru manipularea datelor, interacțiunea cu baze de date, gestionarea evenimentelor și navigarea între ecrane.

Unul dintre principalele avantaje ale PHP este integrarea sa excelentă cu baze de date, în special MySQL. PHP permite crearea, citirea, actualizarea și ștergerea datelor dintr-o bază de date MySQL printr-o serie de funcții specifice, oferind astfel un control complet asupra gestionării datelor într-o aplicație web.

Pentru administrarea și gestionarea bazei de date MySQL, am ales să folosim Visual Studio Code (VS Code), un editor de cod sursă puternic și flexibil, care, prin extensii și integrări, ne permite să realizăm o gamă largă de operațiuni legate de manipularea și administrarea bazelor de date. VS Code este recunoscut pentru ușurința în utilizare și pentru faptul că oferă un mediu de dezvoltare personalizabil, care poate fi adaptat nevoilor specifice ale proiectului.

Una dintre cele mai utile funcționalități ale VS Code este integrarea cu extensii precum **SQLTools** sau **MySQL**, care ne permit să ne conectăm direct la baza de date MySQL din cadrul editorului. Aceste extensii facilitează efectuarea operațiunilor de administrare a bazei de date, fără a fi necesar să folosim un instrument dedicat separat, cum ar fi phpMyAdmin sau MySQL Workbench.

Pentru îmbunătățirea eficienței în dezvoltare, am utilizat extensii suplimentare în Visual Studio Code, cum ar fi PHP Intelephense pentru sugestii și completări inteligente de cod PHP și Live Server pentru reîncărcarea automată a paginilor frontend la salvarea modificărilor. Aceste unelte au accelerat procesul de implementare și testare a aplicației.

3.3 Implementarea interfetei web

Partea de front-end joacă un rol esențial în realizarea acestui proiect, deoarece reprezintă zona cu care utilizatorul interacționează direct. Prin intermediul front-end-ului, utilizatorul poate naviga, accesa funcționalitățile disponibile și beneficia de o experiență plăcută și intuitivă. Pentru implementarea acestei componente vizibile, am utilizat limbajele fundamentale ale dezvoltării web: HTML pentru structura paginilor, CSS pentru stilizarea și aspectul vizual, respectiv JavaScript pentru adăugarea de interactivitate și dinamism.

HTML (HyperText Markup Language) este limbajul standard folosit pentru a crea pagini web. Nu este un limbaj de programare, ci un limbaj de marcăre care organizează și structurează conținutul unei pagini prin etichete (tags). HTML definește structura paginii (titluri, paragrafe, liste, tabele, imagini, linkuri, formulare) și determină cum va arăta conținutul în browser. Fiecare element HTML are o etichetă de deschidere, un conținut și o etichetă de închidere. Structura de bază a unei pagini HTML include etichetele <!DOCTYPE html>, <html>, <head>, <body>, unde <head> conține informații despre pagină (titlu, charset, legături externe), iar <body> conține tot ce este vizibil pentru utilizator. Există multe etichete importante, precum <h1> pentru titluri, <p> pentru paragrafe, <a> pentru linkuri, pentru imagini și / pentru liste. HTML poate fi combinat cu CSS (pentru stilizare) și JavaScript (pentru funcționalități dinamice). Este esențial deoarece orice site web, de la cele mai simple la cele mai complexe, este construit pe o fundație de cod HTML, fiind baza întregului internet vizibil.

CSS (Cascading Style Sheets) este limbajul folosit pentru **stilizarea** paginilor web create cu HTML. Practic, CSS controlează **aspectul vizual** al elementelor de pe o pagină: culori, fonturi, poziționare, dimensiuni, spațieri, borduri, animații și multe altele. Cu ajutorul CSS, poți transforma o pagină simplă de HTML într-un design modern și atrăgător. CSS funcționează pe baza unor **reguli**: fiecare regulă are un **selector** (care alege elementele HTML asupra cărora acționează) și un **set de proprietăți și valori** (care definesc cum arată elementele). CSS poate fi adăugat direct în HTML (inline), în interiorul paginii (într-un tag <style>), sau, cel mai bine, într-un fișier separat (style.css) care este apoi legat de HTML. De asemenea, CSS ajută site-urile să fie **responsive**, adică să se adapteze automat la telefoane, tablete sau ecrane mari, făcând experiența utilizatorului mai plăcută.

Pe partea de styling, am pus accent pe crearea unui sidebar modern și funcțional, care să ofere acces rapid la principalele secțiuni ale aplicației. Acest element a fost realizat folosind tehnici CSS, combinând proprietăți de layout precum flexbox și grid, alături de efecte vizuale atractive precum hover effects și tranziții animate. Sidebar-ul contribuie la o navigare intuitivă, grupând butoanele de acces într-un mod logic și vizibil, astfel încât utilizatorii să poată explora aplicația eficient și confortabil.

```
.sidebar {
width: 200px;
height: 100vh;
background-color: #2c3e50;
color: white;
display: flex;
flex-direction: column;
padding-top: 20px;
position: fixed;
left: 0;
top: 0;
}
```

```

.sidebar a {
text-decoration: none;
color: white;
padding: 15px;
display: block;
font-size: 18px;
transition: 0.3s;
}

.sidebar a:hover {
background-color: #34495e;
}

```

Figura 6. Partea de sidebar a interfetei creata cu CSS

JavaScript (prescurtat JS) este un limbaj de programare folosit pentru a adăuga interactivitate și funcționalitate pe paginile web. Dacă HTML creează structura paginii și CSS o face să arate frumos, JavaScript o face vie, adică permite elementelor să răspundă la acțiunile utilizatorului (clickuri, tastări, mișcarea mouse-ului, trimiterea de formulare etc.).

Pentru a oferi utilizatorilor o experiență vizuală îmbunătățită și interactivă, am implementat un script JavaScript care permite afișarea imaginilor în format full-screen atunci când sunt accesate cu un click. Această funcționalitate este utilă în special în contextul aplicației turistice, unde imaginile joacă un rol esențial în prezentarea locațiilor, hotelurilor sau atracțiilor.

Avantajele funcționalității:

- **Vizualizare detaliată:** Utilizatorii pot vizualiza imaginile la rezoluție completă, fără a fi constrânsi de dimensiunile reduse din pagină.
- **Interacțiune naturală:** Deschiderea imaginii printr-un simplu click este un comportament intuitiv și familiar pentru utilizatori.
- **Estetică modernă:** Integrarea efectelor vizuale (ex. fundal întunecat, tranzitii) contribuie la o experiență plăcută și profesională.

```

<div id="zoomModal" class="modal" onclick="closeModal()">
<span class="close" onclick="closeModal()">&times;</span>
<img id="modalImage" class="modal-content" onclick="closeModal()">
</div>

<script>
function openModal(src) {
document.getElementById("zoomModal").style.display = "block";
document.getElementById("modalImage").src = src;
}
function closeModal() {
document.getElementById("zoomModal").style.display = "none";
}
</script>

```

Figura 7. JavaScript ajuta la modelarea zoom-ului de pe imagini

Pentru a îmbunătăți experiența utilizatorilor în procesul de identificare și selecție a destinațiilor turistice din România, aplicația web a fost extinsă prin integrarea unei hărți interactive dezvoltate cu ajutorul Leaflet – o bibliotecă JavaScript open-source, cunoscută pentru performanța ridicată, flexibilitate și ușurința în utilizare.

Leaflet oferă o interfață intuitivă pentru afișarea hărților dinamice și permite adăugarea rapidă de marcatori, straturi (layers), ferestre de tip popup și alte elemente interactive. Astfel, utilizatorii pot vizualiza într-un mod clar și atractiv localizarea obiectivelor direct pe hartă.

Motivări și beneficii:

- **Ușurință în utilizare:** Utilizatorii pot explora vizual harta României, ceea ce face căutarea unei destinații mult mai intuitivă decât printr-o listă textuală.
 - **Interactivitate:** Prin utilizarea Leaflet, harta oferă funcționalități precum zoom, pan și marcaje (markers) pentru diferite locații (hoteluri, restaurante, obiective turistice).
 - **Performanță optimă:** Leaflet este optimizat pentru performanță chiar și pe dispozitive mobile sau conexiuni mai slabe, ceea ce îl face potrivit pentru o aplicație web accesibilă tuturor utilizatorilor.

Implementare:

- Harta a fost centrată pe teritoriul României și redată într-un container HTML folosind tile-uri de la OpenStreetMap.
 - Pentru fiecare destinație importantă, au fost adăugați markeri personalizați care afișează informații utile atunci când utilizatorul interacționează cu aceștia (ex: numele locației, o imagine, link către pagina dedicată).
 - Sistemul permite căutarea sau filtrarea destinațiilor direct pe hartă, pentru o navigare rapidă și eficientă.

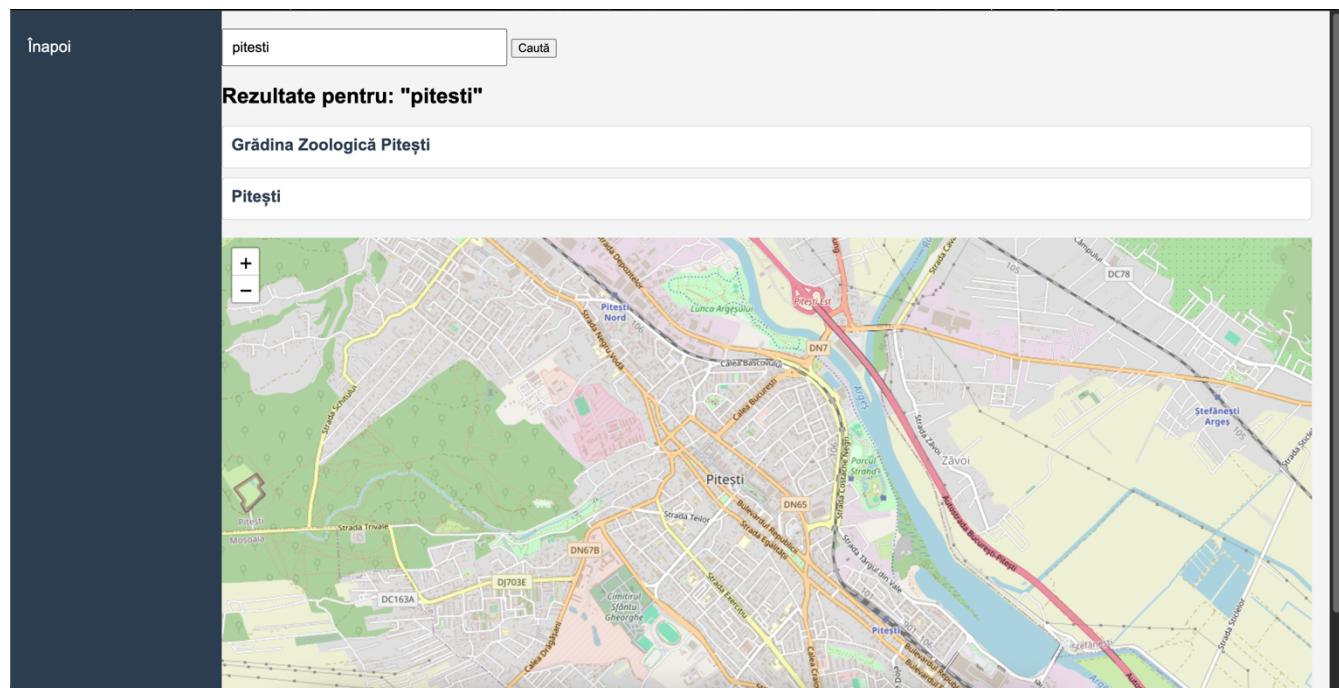


Figura 8. Cautarea pe harta a destinațiilor

```

<?php
$con = mysqli_connect('localhost:8889', 'root', 'root', 'ghid_turistic');

if (!$con) {
die("Conexiunea a eşuat: " . mysqli_connect_error());
}
mysqli_close($con);
?>

```

Figura 9. Conexiunea intre web si baza de date

3.4 MAMP/WAMP

Pentru a rula întregul program, este necesară utilizarea unei aplicații specializate care poate crea un server local pe calculatorul utilizatorului. În acest proiect, am folosit pachete precum MAMP sau WAMP, care facilitează instalarea rapidă și configurarea unui mediu complet de dezvoltare web. Aceste aplicații permit stabilirea unei conexiuni între baza de date și interfața web, folosind phpMyAdmin ca instrument de administrare vizuală a bazelor de date.

Serverul local este activat prin intermediul unor pachete software precum MAMP (pentru macOS) sau WAMP (pentru Windows), care asigură toate componentele necesare rulării aplicației web într-un mediu de dezvoltare controlat. Aceste pachete includ serverul web Apache, sistemul de gestionare a bazelor de date MySQL/MariaDB, precum și utilitarul phpMyAdmin, o aplicație web puternică pentru administrarea bazelor de date relaționale.

După pornirea serverului local prin MAMP sau WAMP, accesul la baza de date se realizează cu ușurință deschizând în browser adresa <http://localhost/phpmyadmin>.

În cadrul procesului de configurare, folderul care conține toate fișierele proiectului (fișiere HTML, CSS, JavaScript și PHP) trebuie salvat în directorul de la locația „Document Root”. Acesta este folderul principal de unde serverul local încarcă paginile web atunci când utilizatorul accesează site-ul prin browser, de exemplu prin adresa <http://localhost/numele-proiectului>.

MAMP și WAMP sunt amândouă pachete de software complete care instalează și configurează automat componente esențiale pentru dezvoltarea web locală:

- **MySQL** pentru crearea și gestionarea bazelor de date
- **Apache** care funcționează ca server web pentru a livra paginile către browser
- **PHP** limbaj de programare server-side folosit pentru a genera conținut dinamic și pentru a interacționa cu baza de date

Prin această structură tehnologică bazată pe MAMP/WAMP și phpMyAdmin, mediul local reușește să emuleze cu acuratețe comportamentul unei aplicații web într-un mediu real de producție. Această reproducere fidelă a condițiilor de funcționare oferă dezvoltatorului un cadru de testare robust și controlat, esențial pentru identificarea timpurie a eventualelor erori, pentru optimizarea performanțelor și pentru rafinarea funcționalităților implementate.

Această abordare permite dezvoltatorului să perfecționeze aplicația într-un mod iterativ, asigurându-se că tranziția către mediul de producție se face fără probleme majore. Prin testarea extensivă și în condiții similare cu cele reale, se reduce semnificativ riscul de erori neașteptate după lansare, ceea ce duce la un produs final mai stabil, mai sigur și mai performant. În acest fel, mediul local nu doar că accelerează dezvoltarea, ci contribuie direct la calitatea aplicației livrate utilizatorilor.

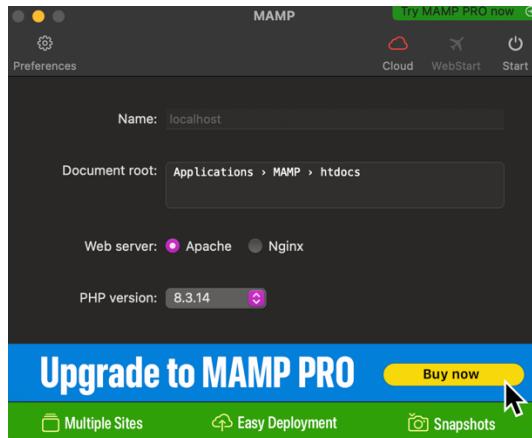


Figura 10. Interfata serverului

În cadrul procesului de dezvoltare și testare a aplicației, serverul local Apache a fost configurat să ruleze pe portul 8888. Această alegere reflectă o practică frecvent întâlnită în dezvoltarea web, având ca scop principal evitarea conflictelor cu alte aplicații sau servicii care utilizează porturi standard, precum portul 80 — rezervat de obicei pentru serverele web de producție sau alte servicii active.

Prin accesarea aplicației la adresa `http://localhost:8888`, dezvoltatorii beneficiază de un mediu controlat, izolat de rețeaua publică, ideal pentru testarea și validarea funcționalităților înainte de lansarea în producție. Această abordare permite identificarea timpurie a erorilor, ajustarea interfeței utilizator și optimizarea performanței fără riscul de a perturba sistemele reale în uz.

Utilizarea unui port alternativ, cum ar fi 8888, aduce și avantajul flexibilității în gestionarea mai multor aplicații web care rulează simultan pe aceeași mașină. Fiecare aplicație poate fi asignată unui port diferit, facilitând testarea modulară și dezvoltarea paralelă în echipe. De exemplu, o aplicație backend poate rula pe portul 8888, în timp ce o interfață frontend poate fi testată pe portul 3000, fără conflicte sau necesitatea unor configurații complexe.

În concluzie, rularea serverului Apache pe portul 8888 reprezintă o soluție eficientă și profesională în contextul dezvoltării locale, contribuind la organizarea logică a mediului de lucru și asigurând un ciclu de testare robust, predictibil și scalabil.

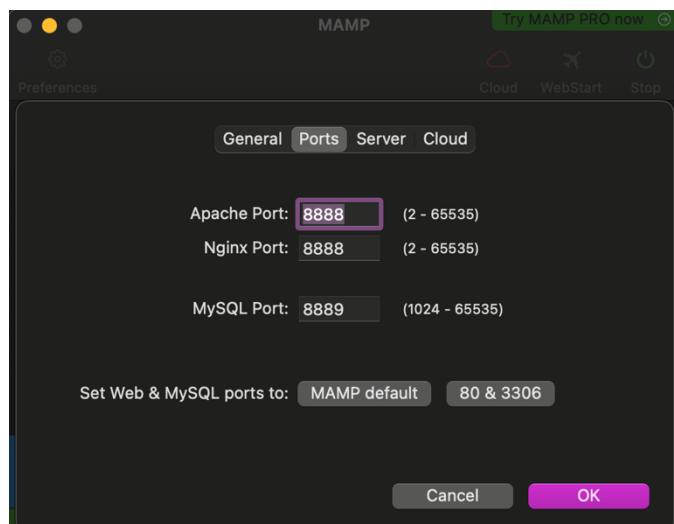


Figura 11. Setările pentru porturi

```

<?php
$con = mysqli_connect('localhost:8889', 'root', 'root', 'ghid_turistic');

// Verificare conexiune
if (!$con) {
die("Conexiunea a eșuat: " . mysqli_connect_error());
}

$mesaj = "";
$stil = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
$username = $_POST['username'];
$parola = $_POST['parola'];

// Caută utilizatorul în baza de date
$query = "SELECT * FROM users WHERE username = ?";
$stmt = $con->prepare($query);
$stmt->bind_param("s", $username);
$stmt->execute();
$result = $stmt->get_result();

if ($result->num_rows == 1) {
$user = $result->fetch_assoc();

// Verifică parola
if (password_verify($parola, $user['parola'])) {
$mesaj = "Autentificare reușită. Bine ai venit, <strong>$username</strong>!";
$stil = "background-color: #d4edda; color: #155724; border: 1px solid #c3e6cb;";
} else {
$mesaj = "Parolă incorectă!";
$stil = "background-color: #f8d7da; color: #721c24; border: 1px solid #f5c6cb;";
}
} else {
$mesaj = "Utilizatorul nu există!";
$stil = "background-color: #f8d7da; color: #721c24; border: 1px solid #f5c6cb;";
}

$stmt->close();
}

mysqli_close($con);
?>

```

Figura 12. Metoda autentificare utilizator

Pentru implementarea funcționalității de înregistrare a utilizatorilor, am dezvoltat un script PHP care gestionează procesul de creare a conturilor într-un mod sigur și eficient. La primirea datelor din formularul de înregistrare, am aplicat funcția htmlspecialchars() pentru a preveni atacurile de tip XSS (Cross-Site Scripting) și am criptat parolele utilizatorilor folosind password_hash() cu algoritmul PASSWORD_DEFAULT, asigurând astfel protecția datelor sensibile.

Pentru a evita introducerea de date duplicate în baza de date, am implementat un mecanism de verificare a existenței utilizatorului înainte de inserarea unui nou cont. Toate operațiile de interogare și inserare a datelor au fost realizate folosind instrucțiuni pregătite (prepare, bind_param) pentru a preveni atacurile de tip SQL Injection și pentru a menține integritatea bazei de date.

Utilizând metoda POST, se conectează la baza de date, execută instrucțiunea SQL și verifică dacă utilizatorul există. Se închide conexiunea la baza de date și se returnează mesajul "Autentificare reușită. Bine ai venit, \$username!" dacă autentificarea a avut succes, altfel se returnează mesajul "Parola incorectă!" sau "Utilizatorul nu există!".

Transmiterea datelor către server se realizează prin metoda HTTP POST, una dintre cele mai utilizate metode pentru trimiterea în siguranță a informațiilor confidențiale. După conectarea la baza de date, scriptul execută instrucțiunea SQL corespunzătoare și verifică dacă utilizatorul există deja.

În plus, am implementat un sistem de feedback vizual pentru utilizator, folosind mesaje de succes sau eroare stilizate, care oferă informații clare privind starea înregistrării. Astfel, aplicația oferă o experiență sigură, intuitivă și plăcută pentru utilizatorii care doresc să își creeze un cont nou.

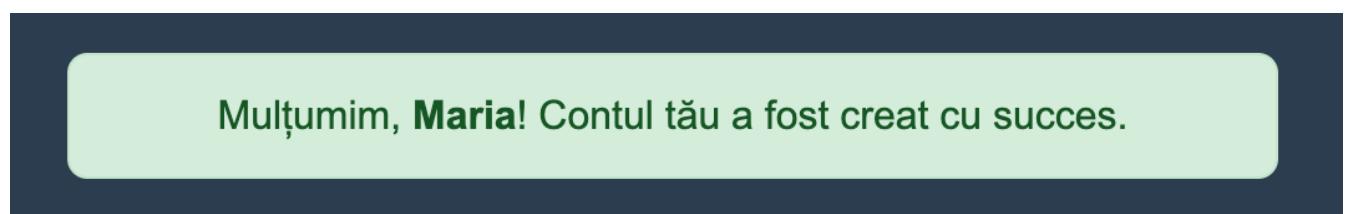


Figura 13. Confirmarea adăugării unui nou utilizator

| | ← T → | id | username | parola |
|--------------------------|--------------------|----|----------|----------------------------------------------------------|
| <input type="checkbox"/> | Edit Copy Delete | 1 | Maria | \$2y\$10\$RdF2sGNqhDINmEXycToFP.S9.QAFzQ362JMxZkEjlbw... |

Figura 14. Verificarea ca utilizatorul a fost adăugat cu succes în baza de date

3.5 phpMyAdmin

Pentru ca utilizatorul (în acest caz, administratorul aplicației) să poată efectua operații de modificare, ștergere sau inserare a datelor din baza de date, aplicația web oferă o interfață grafică intuitivă, integrată în serverul local. Această interfață simplifică semnificativ procesul de gestionare a datelor, eliminând necesitatea scrierii directe de comenzi SQL și permitând interacțiuni rapide și eficiente prin elemente vizuale, cum ar fi formulare, butoane și liste dinamice.

În timpul dezvoltării, phpMyAdmin permite și efectuarea de operații suplimentare, cum ar fi:

- Importul sau exportul bazelor de date (în format .sql)
- Executarea manuală a interogărilor SQL pentru teste specifice
- Vizualizarea datelor în timp real pentru verificarea corectitudinii funcționalităților implementate pe partea de back-end

Prin folosirea MAMP/WAMP și phpMyAdmin, procesul de dezvoltare a aplicației web devine mult mai accesibil, oferind un cadru de lucru complet și prietenos pentru gestionarea atât a codului, cât și a datelor.

The screenshot shows the phpMyAdmin interface with the following details:

- Server:** localhost:8889
- Database:** ghid_turistic
- Table:** locuri_vizitat
- Rows:** 0 - 24 (125 total)
- Query took:** 0.0036 seconds

The table structure is as follows:

| | id | orase_id | nume | descriere | imagine |
|----|-----------|-----------------|--------------------------------------|--------------------------------------------------------|----------------------------------|
| 1 | 1 | 1 | Muzeul National al Unirii Alba Iulia | Muzeul National al Unirii Alba Iulia este unul din... | images/muzeul_alba.jpg |
| 2 | 2 | 1 | Râpa Roșie | Râpa Roșie este o rezervație naturală impresionant... | images/rapa_rosie.jpg |
| 3 | 3 | 1 | Cetatea Câlnic | Cetatea Câlnic, o fortificație medievală impresion... | images/cetatea_calnic.jpg |
| 4 | 4 | 1 | Satul Rimetea | Satul Rimetea este un loc deosebit, cunoscut pentr... | images/satul_rimetea.jpg |
| 5 | 5 | 1 | Castelul Bethlen-Haller | Castelul Bethlen-Haller din Cetatea de Baltă, jude... | images/castelul_bethlen.jpg |
| 6 | 6 | 2 | Castelul Regal de la Săvârșin | Castelul Regal de la Săvârșin, construit în secolu... | images/castelul_regal.jpg |
| 7 | 7 | 2 | Parcul Natural Lunca Mureșului | Parcul Natural Lunca Mureșului, situat în vestul R... | images/parcul_natural.jpg |
| 8 | 8 | 2 | Palatul Culturii din Arad | Palatul Culturii din Arad, construit între 1911 și... | images/palatul_culturii_arad.jpg |
| 9 | 9 | 2 | Stațiunea Moneasa | Stațiunea Moneasa, situată în Munții Codru Moma, e... | images/statiunea_moneasa.jpg |
| 10 | 10 | 2 | Mănăstirea Hodoș-Bodrog | Mănăstirea Hodoș-Bodrog, un lăcaș monahal de tradi... | images/manastirea_hodos.jpg |
| 11 | 11 | 3 | Cheile Tazlăului | Cheile Tazlăului sunt o atracție naturală spectacu... | images/cheile_tazlaului.jpg |
| 12 | 12 | 3 | Mănăstirea Măgura Ocnei | Mănăstirea Măgura Ocnei, situată în județul Bacău... | images/manastirea_magura.jpg |
| 13 | 13 | 3 | Casa Memorială George Bacovia | Casa Memorială George Bacovia este locul de nașter... | images/casa_bacovia.jpg |
| 14 | 14 | 3 | Lacul Bacău | Lacul Bacău, cunoscut și sub denumirea de Lacul Ba... | images/lacul_bacau.jpg |
| 15 | 15 | 3 | Cheile Bistriței | Cheile Bistriței sunt o formație geologică impre... | images/cheile_bistritel.jpg |
| 16 | 16 | 4 | Turnul lui Ștefan | Turnul lui Ștefan din Bârsana este un monument i... | images/turnul_stefan.jpg |
| 17 | 4 | 4 | Cascada Calitor | Cascada Calitor, situată în Munții Maramureșului, e... | images/cascada_calitor.jpg |

Figura 15. Interfata phpMyAdmin

3.6 OpenAI – ChatGPT

ChatGPT este un model de inteligență artificială dezvoltat de OpenAI, bazat pe o arhitectură numită transformer (mai exact, o variantă îmbunătățită din familia GPT – Generative Pre-trained Transformer). Pe scurt, ChatGPT este un asistent virtual capabil să înțeleagă întrebări și să genereze răspunsuri într-un mod asemănător comunicării umane.

În contextul actual al dezvoltării tehnologice accelerate, tranziția de la elaborarea unui cod de bază, către crearea unor aplicații web complexe cu interfețe grafice avansate a devenit un proces mult mai facil. Această transformare este posibilă datorită integrării **inteligenței artificiale** în procesul de generare a codului sursă. Prin transmiterea unor comenzi și specificații clare, utilizatorii pot solicita sistemelor bazate pe inteligență artificială să producă cod funcțional, optimizat și lipsit de erori de execuție.

În vederea obținerii unor rezultate relevante și adaptate cerințelor proiectului, este esențial ca instrucțiunile inițiale transmise modelului de inteligență artificială să fie formulate explicit și să contureze obiectivul final al aplicației. Astfel, motorul de procesare din spatele conversației este antrenat corespunzător și calibrat pentru a susține o dezvoltare coerentă și eficientă a proiectului.

Mai mult, utilizarea inteligenței artificiale în procesul de dezvoltare aduce beneficii semnificative din perspectiva optimizării timpului de lucru și a eficienței utilizării resurselor de sistem. În cadrul prezentului proiect, inteligența artificială a fost utilizată pentru automatizarea procesului de proiectare și implementare a bazei de date, reducând considerabil timpul necesar acestei etape și facilitând o integrare rapidă și sigură a componentelor esențiale ale aplicației web.

4. Concluzii

4.1 Concluziile proiectului

Prin proiectul „Ghid Turistic” am realizat o aplicație web complet funcțională, intuitivă și sigură, destinată promovării turismului local și național. Aplicația oferă utilizatorilor acces rapid și facil la informații relevante despre destinații turistice, obiective de vizitat, restaurante, unități de cazare și alte puncte de interes. De asemenea, integrează funcționalități moderne precum filtre de căutare, recenzii, evaluări și localizare pe hartă, contribuind astfel la o experiență de utilizare plăcută și eficientă.

Pe parcursul dezvoltării proiectului, ne-am îmbunătățit considerabil competențele tehnice în domenii precum programarea web, arhitectura bazelor de date, securitatea aplicațiilor și designul interfeței. În plus, am dobândit experiență valoroasă în ceea ce privește munca în echipă, organizarea sarcinilor, comunicarea eficientă și respectarea termenelor, ceea ce ne-a permis să colaborăm productiv și să livrăm un produs final de calitate.

Pe parcursul dezvoltării proiectului, ne-am îmbunătățit considerabil competențele tehnice în domenii precum programarea web, arhitectura bazelor de date, securitatea aplicațiilor și designul interfeței. În plus, am dobândit experiență valoroasă în ceea ce privește munca în echipă, organizarea sarcinilor, comunicarea eficientă și respectarea termenelor, ceea ce ne-a permis să colaborăm productiv și să livrăm un produs final de calitate.

4.2 Lista contribuțiilor

Contribuțiile echipei au fost bine împărțite, fiecare membru asumându-și responsabilități clare în funcție de abilitățile proprii. Un membru a lucrat la dezvoltarea frontend-ului folosind tehnologii precum HTML, CSS și JavaScript, concentrându-se pe crearea unei interfețe atractive și ușor de utilizat. Dar și de partea de backend, utilizând phpMyAdmin pentru a gestiona logica aplicației, autentificarea utilizatorilor și comunicarea cu baza de date.

O altă componentă importantă a fost proiectarea și implementarea bazei de date MySQL, care a permis stocarea structurată a informațiilor despre locații turistice, recenzii și utilizatori. De asemenea, s-a acordat o atenție deosebită testării aplicației, optimizării performanței și asigurării securității datelor utilizatorilor. În final, am colaborat pentru redactarea documentației și pregătirea prezentării proiectului, evidențiind atât funcționalitățile tehnice, cât și impactul asupra utilizatorilor.

Pentru a ne ușura colaborarea și a eficientiza procesul de testare, am decis să folosim GitHub ca platformă principală pentru gestionarea codului sursă. Utilizarea unui repository comun ne-a permis să salvăm progresul realizat în timp real, să urmărim modificările efectuate de fiecare membru al echipei și să evităm suprascrierea neintenționată a codului. Prin intermediul funcționalităților de versionare, pull request-uri și issue tracking, am reușit să lucrăm simultan la diferite componente ale aplicației, să revizuim codul înainte de integrare și să remediem rapid eventualele erori sau conflicte. GitHub ne-a oferit astfel un mediu de lucru organizat, colaborativ și sigur, contribuind semnificativ la succesul proiectului.

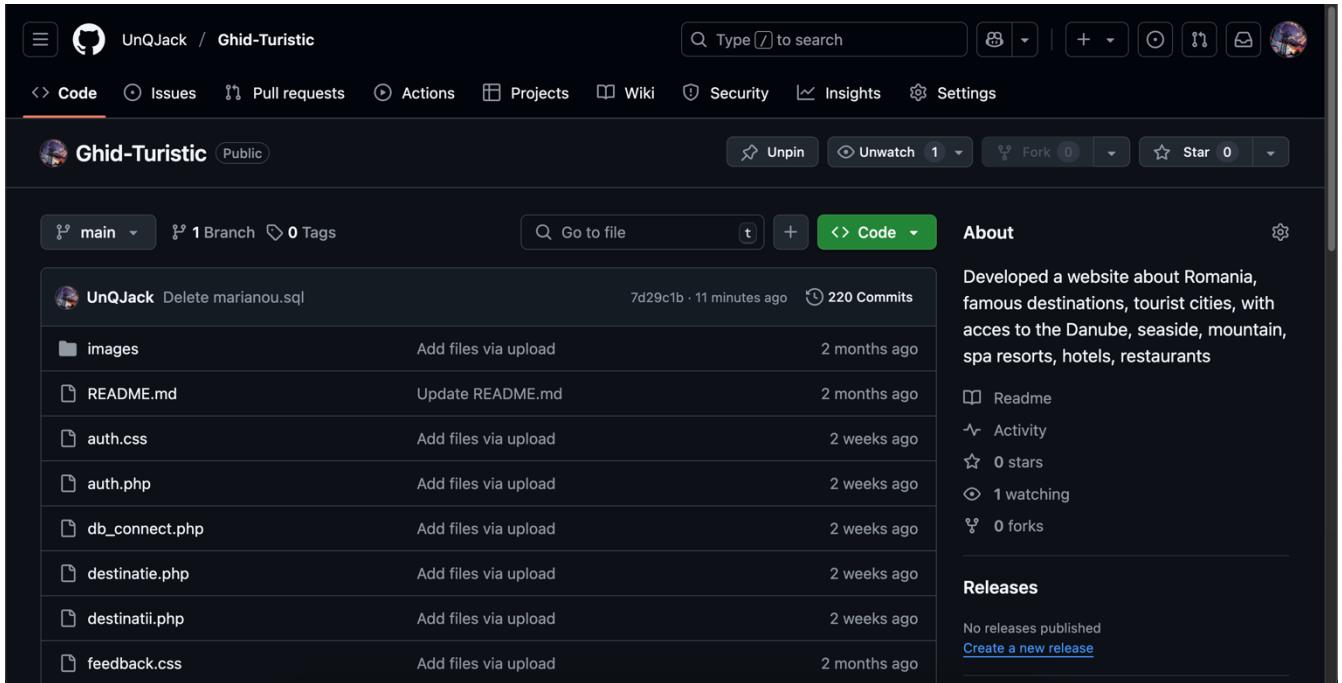


Figura 16. Utilizarea platformei GitHub

4.3 Perspective de dezvoltare ulterioara

- **Sistem de filtrare avansat:** Utilizatorii pot filtra sau căuta destinațiile în funcție de criterii multiple, precum: Tipul locației (cultural, natură, aventură, gastronomie etc.), interval de preț (gratuit, buget mediu, premium).
- **Rating-uri sau recenziile altor turiști:** Acest sistem facilitează găsirea rapidă a celor mai relevante obiective în funcție de preferințele personale.
- **Recenziile și evaluările avansate:** Fiecare destinație, hotel sau restaurant poate fi evaluat de către utilizatori printr-un sistem de rating (stele sau scor numeric) și comentarii detaliate. Se pot adăuga și fotografii din partea vizitatorilor pentru a oferi o perspectivă autentică asupra experienței.
- **Listă personală de locații favorite:** Utilizatorii autentificați au posibilitatea de a salva destinațiile preferate într-o listă personalizată. Aceasta le permite să-și planifice călătoria mai ușor și să revină rapid la locațiile de interes.
- **Notificări personalizate:** Aplicația trimit notificări în timp real în funcție de preferințele selectate de utilizator, precum: oferte speciale la hoteluri sau restaurante, evenimente locale sau festivaluri
Alerta privind modificările programului sau închideri temporare
- **Suport multilingv:** Aplicația este disponibilă în mai multe limbi de circulație internațională, precum engleză, franceză, germană, italiană etc. Această funcționalitate îmbunătățește accesibilitatea și utilitatea aplicației pentru turiștii internaționali.
- **Optimizare completă pentru mobil:** Dezvoltarea unei aplicații mobile native pentru platformele Android și iOS asigură o experiență fluidă și intuitivă pentru utilizatori. Designul este responsiv, cu încărcare rapidă și adaptare automată la diferite rezoluții și dimensiuni de ecran.
- **Autentificare prin social media:** Pentru a simplifica procesul de înregistrare și autentificare, utilizatorii pot accesa aplicația folosind conturile lor existente de Google, Facebook sau alte

platforme de socializare. Acest lucru contribuie la o experiență de onboarding rapidă și familiară.

- **Sistem de rezervări integrat:** Utilizatorii pot face rezervări direct prin aplicație pentru hoteluri, restaurante, tururi ghidate sau alte activități. Sistemul este conectat la parteneri locali și permite afișarea disponibilității în timp real, confirmări rapide și eventual integrare cu platforme externe (ex: Booking.com, OpenTable).

5. Bibliografie

<https://github.com/UnQJack/Ghid-Turistic>

<https://www.infoghidromania.com/>

<https://romaniatravel.guide/ro/ghid-de-calatorie/romania/1036991/>

<https://www.locurifaine.ro/ghid-turistic/>

<https://www.booking.com/>

<https://www.tripadvisor.com/>

<https://chatgpt.com/>