**Baranya Megyei Szakképzési Centrum**

**Baranya Megyei SzC Radnóti Miklós Közgazdasági**

**Technikum**

Szakma megnevezése: Szoftverfejlesztő és –tesztelő

A szakma azonosító száma: 5 0613 12 03

**Vizsgaremek**

Pagoda Travel Utazási iroda

**Készítették: Hideg Felícia, Garai Szabolcs, Odrobina Róbert.**

**Pécs**

**2025**

# Tartalomjegyzék

[Tartalomjegyzék 1](#_Toc548412107)

[1. Bevezetés 2](#_Toc105038116)

[1.1 Feladat megosztás: 3](#_Toc891760523)

[1.2 Részletes feladat megosztás: 3](#_Toc808456202)

[2. A projekt célja 3](#_Toc1153187463)

[A webalkalmazás fő céljai: 4](#_Toc1166963182)

[3. Rendszerkövetelmények 4](#_Toc1338272479)

[3.1 Hardver és szoftver követelmények: 4](#_Toc1308414153)

[3.2 A program telepítése és futtatása 5](#_Toc2070924973)

[4. A program használatának részletes leírása 6](#_Toc643630371)

[4.1 Kezdőlap 7](#_Toc734192835)

[4.2 Bejelentkezés és regisztráció 7](#_Toc338640442)

[4.3 Utazások oldal 8](#_Toc823273495)

[4.4 Foglalásaim 8](#_Toc1747385087)

[4.5 Kapcsolat oldal használata 9](#_Toc1308135983)

[4.6 Kijelentkezés 10](#_Toc1741282509)

[4.7 Foglalás menete 10](#_Toc1536415949)

[4.8 A foglalás lépései: 10](#_Toc1740111689)

[5. Admin felület használata 11](#_Toc1741108816)

[5.1 Foglalások kezelése 11](#_Toc817841036)

[5.2 Új utazás felvitele 12](#_Toc1700590708)

[5.3 Új szállás vagy úticél felvitele 12](#_Toc1393673035)

[5.4 Felhasználók kezelése 12](#_Toc284807332)

[5.5 Felhasználói és egyéb üzenetek kezelése 12](#_Toc1427042865)

[6. Hibakezelés / hibaüzenetek 13](#_Toc851213974)

[7. Használt technológiák: 13](#_Toc1407445647)

[7.1 Frontend (Felhasználói felület) 13](#_Toc207557625)

[7.2 Backend (Szerveroldali logika) 13](#_Toc1512365761)

[7.3 Fájlszerkezet 14](#_Toc1751089996)

[7.4 Főbb funkciók működése 14](#_Toc777759998)

[7.5 Biztonsági szempontok 15](#_Toc729202660)

[7.6 Adatbázis 15](#_Toc587602862)

[7.7 Fejlesztési eszközök 15](#_Toc1666530544)

[8. Rendszerfelépítés 15](#_Toc315316869)

[8.1 Mappa szerkezet 16](#_Toc677570520)

[8.2 Kérésfeldolgozás menete 17](#_Toc1098111615)

[8.3 A működés általános folyamata 17](#_Toc1816228755)

[8.4 Működési példák funkciókra bontva 18](#_Toc861490016)

[9. Adatmodell 19](#_Toc917169632)

[9.1 Adatbázis Dokumentáció 19](#_Toc837452313)

[9.2 DB diagram Modell: 22](#_Toc926180249)

[10. A program főbb függvényei és vezérlő logikája 22](#_Toc1698399669)

[11. Tesztelés: 24](#_Toc916414851)

[11.1 Tesztelési módszerek 24](#_Toc1746493079)

[11.2 Biztonsági ellenőrzések 24](#_Toc2124145237)

[11.3 Tesztelés során tapasztalt hibák: 24](#_Toc1262219239)

[12. Továbbfejlesztési lehetőségek 25](#_Toc1348143356)

[12.1 Online fizetés integrálása 25](#_Toc577581523)

[12.2 Értékelési és visszajelzési rendszer 25](#_Toc231269862)

[12.3 E-mail értesítések 25](#_Toc1784665510)

[12.4 Foglalás törlése / módosítása felhasználói oldalon 25](#_Toc819653256)

[12.5 Teljes reszponzivitás és mobil optimalizálás 25](#_Toc660114646)

[12.6 Több szerepkör bevezetése 26](#_Toc238112266)

[12.7 Kétlépcsős bejelentkezés 26](#_Toc2031764148)

[13. Képernyőképek 26](#_Toc1212947956)

[14. A projekt forráskódja: 30](#_Toc1589812697)

[15. Forrásmegjelölés, irodalomjegyzék 30](#_Toc2028304383)

# 1. Bevezetés

A PagodaTravel egy fiktív utazási iroda weboldala, amely lehetőséget kínál a felhasználóknak repülős utak és szállások böngészésére, foglalására, valamint regisztrációra és saját foglalásaik nyomon követésére. Az admin felület külön lehetőséget ad új utazások, szállások és úticélok rögzítésére, valamint foglalások kezelésére. Továbbá felhasználók módosítására és törlésére.

Azért döntöttünk az utazási iroda weboldal mellett, mert a későbbiekben szeretnénk tovább fejleszteni és élesíteni. A projekt témájára azért esett a választásunk, mert a kapcsolati körömben egy barátom turizmussal foglalkozik és a közel jövőben szeretnénk egyesíteni erőinket. Az ő ötletei alapján feltöltve weblapunkat a megfelelő utazásokkal.

A fejlesztés során sok megoldandó hibával és feladattal találkoztunk, melyek megmutatták mennyit kell még tanulnunk, hogy sikeres Szoftverfejlesztők legyünk! Emellett egy-egy témában mélyebb ismereteket is szereztünk, mint például egy szépen struktúrált MVC modell kialakítása, vagy a PDO működése terén.

A projekt a kezdetek kezdetén nehézkesen indult, de miután lefektettük az alapokat szinte már szárnyaltunk. Az elején az ötletelési fázisban lassan haladtunk. Mindenkinek volt egy elképzelése, hogy milyen irányban kellene elindulnunk, hogy milyen vizsga remeket is kellene elkészítenünk. Ekkor nagyon sokat találkoztunk az iskola mellett, kötetlen beszélgetések során ötleteltünk egy-egy kávé mellett. Ilyenkor szinte észre sem vettük milyen gyorsan telik az idő. Néhány ilyen beszélgetés rádöbbentett milyen fontos egy személyes találkozás és ötletelés, rohanó világunkban. Sok-sok tervezés után végre találtunk egy mindenkinek tetsző formátumot. Ez lett a Pagoda Travel utazási iroda.

A feladatkörök szétosztáskakor figyeltünk arra, hogy mindenkinek az erősségét kihasználjuk és mindenki olyan feladatrészt kapjon ahol kihasználhatja kreativitását. Mindegyikünknél voltak olyan területek, ahol több tudással rendelkezett a másikunknál. Természetesen ez nem azt jelenti, hogy nem segítettük egymást a részfeladatok megoldásában! Ha valaki segítségre szorult közösen oldottuk meg az elénk gördülő akadályokat.

## 1.1 Feladat megosztás:

A feladatok fő megosztása e formában valósult meg:

|  |  |  |
| --- | --- | --- |
| Frontend | Backend | Adatbázis |
| Hideg Felícia | Odrobina Róbert | Garai Szabolcs |

## 1.2 Részletes feladat megosztás:

|  |  |
| --- | --- |
| Feladat | Felelős |
| Projekt terv és adatmodell | Hideg Felícia, Odrobina Róbert, Garai Szabolcs |
| Frontend fejlesztés(HTML/CSS/JS) | Hideg Felícia |
| Backend logika (PHP, MVC) | Odrobina Róbert |
| Adatbázis létrehozása és kezelése | Garai Szabolcs |
| Admin felület kialakítása | Hideg Felícia, Odrobina Róbert, Garai Szabolcs |
| Felhasználói funkciók fejlesztése | Odrobina Róbert |
| Dokumentáció és tesztelés | Hideg Felícia, Odrobina Róbert, Garai Szabolcs |

# 2. A projekt célja

A projekt célja egy olyan **modern, dinamikus webes platform** létrehozása volt, amely lehetővé teszi a felhasználók számára, hogy kényelmesen és hatékonyan böngésszenek különböző utazási ajánlatok között, és azokat azonnal le is foglalhassák. A rendszer megtervezésekor külön figyelmet fordítottunk a **felhasználóbarát felület kialakítására**, valamint az **adminisztrációs oldal egyszerű és biztonságos kezelésére**.

A főbb szempontok:

* a különböző utazások böngészése,
* szállások kiválasztását az utazás helyszíne szerint,
* foglalások utasszám alapján (a helyek számának csökkenésével),
* a foglalások tárolását és megjelenítését,
* az Admin felület biztosítását tartalomkezelés céljából.

## A webalkalmazás fő céljai:

* **Különböző utazások böngészése:** A látogatók átlátható kártyanézetben tekinthetik meg az elérhető utakat, rövid és részletes leírással, képekkel, árakkal és indulási-időpont információkkal. A kezdőlapon csupán a három legfrissebb utazás jelenik meg, de egy külön aloldalon az összes elérhető ajánlat listázva van.
* **Szállások kiválasztása a célállomás alapján:** A foglalási folyamat során a felhasználó csak azok közül a szállások közül választhat, amelyek az adott utazás célhelyéhez kapcsolódnak. Ez egy backend szűréssel történik, amely kizárja a nem releváns szállásokat, így biztosítva a releváns kínálatot és az adatkonzisztenciát.
* **Foglalás utasszám alapján, dinamikus férőhelykezeléssel:** A foglalási űrlap nem engedélyezi több utas lefoglalását, mint amennyi szabad hely még elérhető az adott utazáson. A foglalás rögzítése után az utazáshoz tartozó elérhető férőhelyek száma automatikusan csökken az adatbázisban. Ezzel elkerülhetők a túlfoglalások, és pontosan nyomon követhető a kapacitás.
* **Foglalások tárolása és megjelenítése:** A rendszer rögzíti a felhasználó által végzett foglalásokat a bookings adatbázistáblába. A bejelentkezett felhasználó egy külön „Saját foglalásaim” oldalon keresztül bármikor megtekintheti az összes eddigi foglalását, beleértve a választott utazást és szállást is.
* **Admin felület biztosítása tartalomkezelés céljából:** Az adminisztrátorok számára külön felület készült, amely lehetővé teszi új utazások, szállások és úticélok rögzítését, meglévők törlését, valamint a beérkezett foglalások és felhasználók kezelését. A bal oldali navigációs menü révén könnyen elérhetők az admin funkciók, külön nézetekre bontva.

Összegzés

Ez az utazási webalkalmazás nem csupán egy foglalási platform, hanem egy jól szervezett, **többfunkciós rendszer**, amely egyszerre szolgálja ki a felhasználók, az adminisztrátorok és a fejlesztési követelmények igényeit. A projekt célkitűzései között szerepelt a **reszponzív design**, a **valós idejű adatkezelés**, valamint az egyszerű **felhasználói élmény** biztosítása minden látogatói szinten.

# 3. Rendszerkövetelmények

## 3.1 Hardver és szoftver követelmények:

|  |  |  |
| --- | --- | --- |
| Kategória | Minimális követelmény | Ajánlott követelmény |
| Processzor(cpu) | 1.6 Ghz dual core vagy gyorsabb | 2.4 Ghz quad core vagy gyorsabb |
| Memória(RAM) | 4 Gb RAM | 8 Gb RAM |
| Háttértár | 500 Mb szabad hely az alkalmazás és a XAMP számára | SSD háttértát 1 Gb szabad hellyel |
| Képernyő | 1280x720 felbontás | 1920x1080 felbontás vagy magasabb |
| Operációs rendszer | Windows 10/Linux/ MacOs | Windows 11/Ubuntu 22.04 MacOs |
| Böngésző | Mozilla Firefox, Google Chrome | Mozilla Firefox legújabb verzió |
| Internetkapcsolat | Stabil internet a dinamikus működéshez, minimum 10 Mbps | 1 Gb/s internet kapcsolat |
| Szerverkörnyezet | XAMP, PHP, MYSQL | XAMP 8.2, Apache, phpMyadmin |

## 3.2 A program telepítése és futtatása

**A program éles környezetben való használatához nincs szükség a szerver környezet szimulálását szolgáló szoftverekre, mint a XAMP. Azonban a projekt jelenlegi stádiumában szükségesek ezek a szoftverek.**

Tehát nézzük át mit kell tenni ahhoz, hogy a projekt a jelenlegi formában megtekinthető legyen akármilyen asztali számítógépen.

1. lépés: A XAMP legfrissebb verziójának a feltelepítése.

2.lépés: A Xamp feltelepítése után a projekt főkönyvtárát minden tartalmával együtt átmásoljuk a C:\xampp\htdocs mappába.

3.lépés: A C:\xampp\apache\conf mappában a httpd.conf fájlban módosítani kell ezeket az értékeket:

**Ez az alap beállítás:**

*<Directory />*

*AllowOverride none*

*Require all denied*

*</Directory>*

**És erre kell módosítani:**

*<Directory />*

*AllowOverride All*

*Require all granted*

*</Directory>*

**Ugyanebben a fájlban, még ezt kell módosítani:**

**Ez a gyári érték:**

*DocumentRoot "C:/xampp/htdocs" <Directory "C:/xampp/htdocs">*

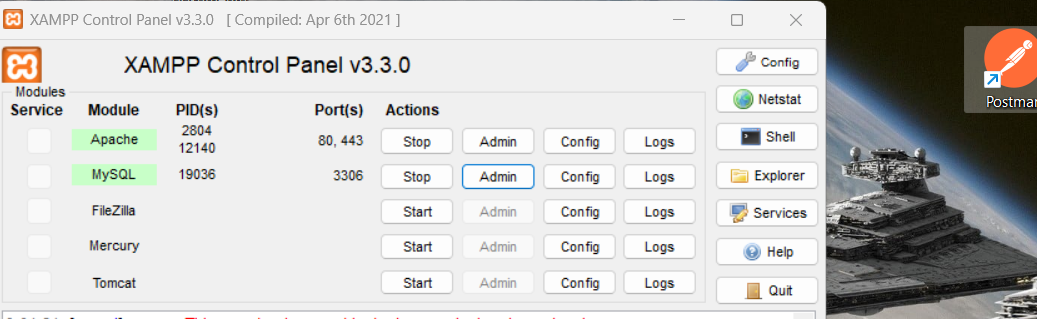
**És erre kell módosítani:**

DocumentRoot "C:/xampp/htdocs/ev\_vegi\_mvc\_szerint/public" <Directory "C:/xampp/htdocs/ev\_vegi\_mvc\_szerint/public">

4.lépés: A C:\xampp\apache\conf\extra mappában módosítani kell a: httpd-vhosts.conf fájlt.

**Itt csak bele kell írni a fájl végére:** *<VirtualHost \*:80> DocumentRoot "C:/xampp/htdocs/ev\_vegi\_mvc\_szerint/public" ServerName evvegi.local <Directory "C:/xampp/htdocs/ev\_vegi\_mvc\_szerint/public"> AllowOverride All Require all granted*

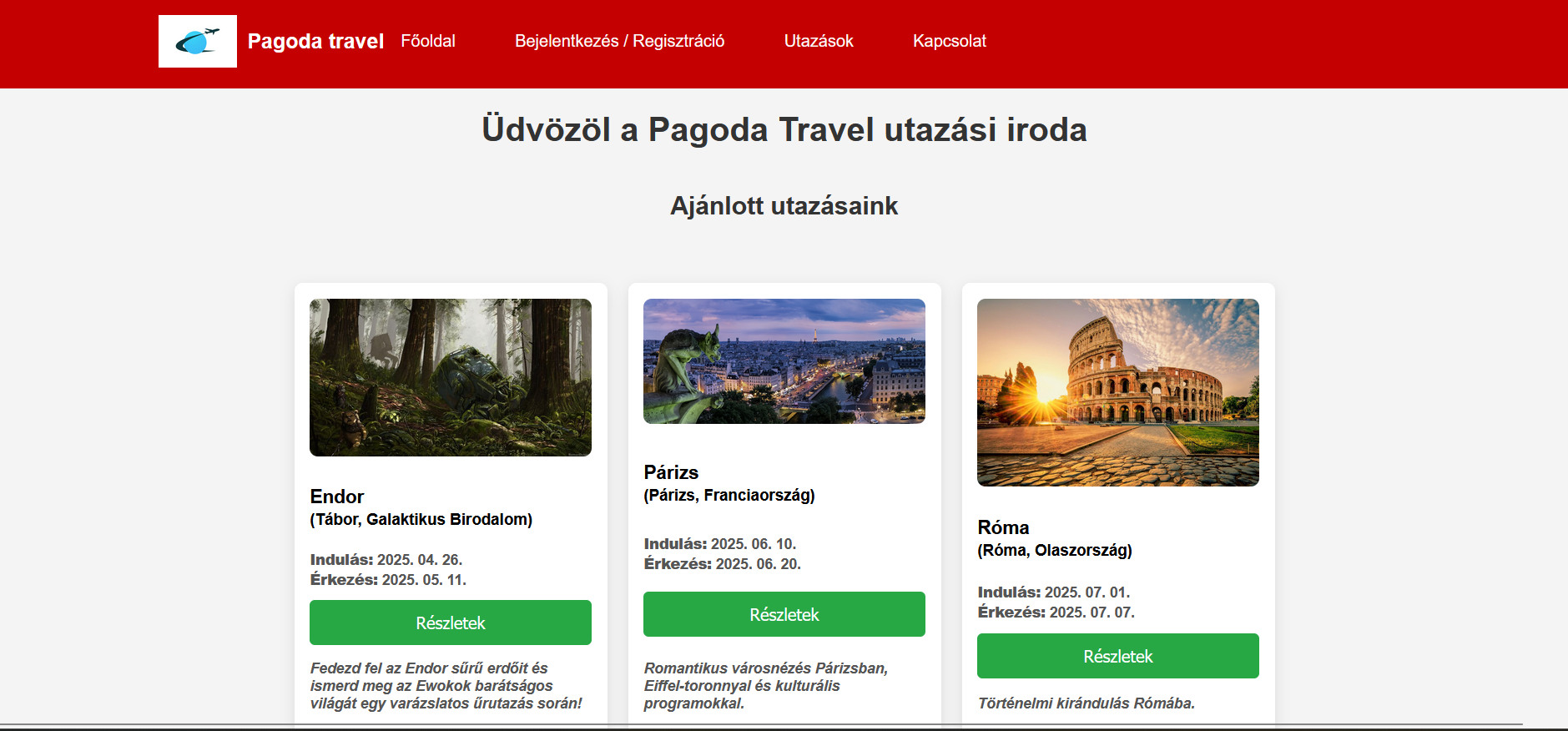
5.lépés: Indítsuk el a XAMP-ot, majd a XAMP control panelen indítsuk el az Apache és a MySql-t.



6.léps: A böngésző címsorába írd be a localhost kifejezést. Ekkor megérkezel a projekt kezdőoldalára.

# 4. A program használatának részletes leírása

A webalkalmazás reszponzív, böngészőből elérhető felületen működik, de mobil készülékeken is kiválóan elérhető és működőképes. Az oldal **bejelentkezés után** teljeskörűen használható, és több funkciót kínál mind a felhasználók, mind az adminisztrátorok számára. Az alábbiakban a felhasználói élmény lépéseit és lehetőségeit mutatjuk be.



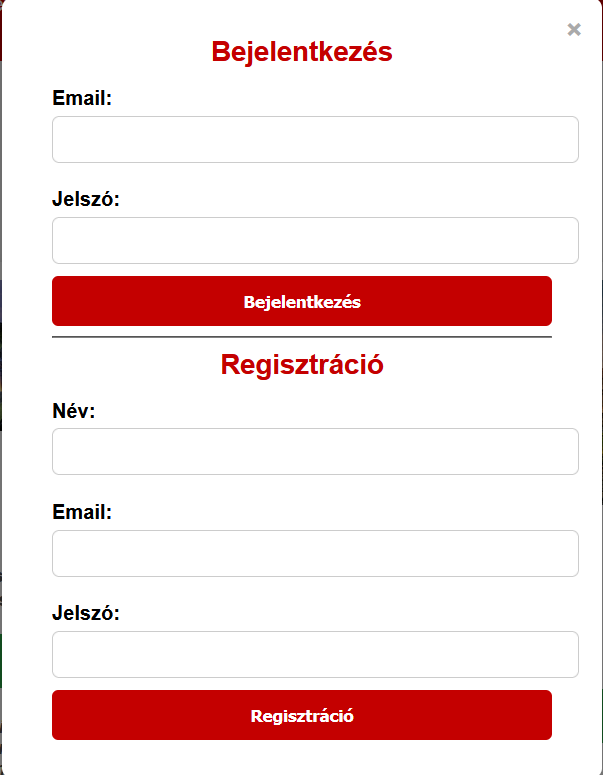
## 4.1 Kezdőlap

* A bejelentkezett felhasználót a kezdőoldal fogadja, ahol **három kiemelt utazás** jelenik meg kártyák formájában. A három megjelenített utazás a legújabb vagy legnépszerűbb ajánlatok közül kerül kiválasztásra, hogy a látogatók gyorsan képet kapjanak a kínálatról.
* Minden kártya tartalmazza:
  + az úticél nevét,
  + az indulási és vissza érkezési dátumát,
  + az árát,
  + egy rövid leírást,
  + valamint egy „Részletek” gombot.
* A „Részletek” gombra kattintva **modálablakban** jelenik meg a teljes leírás, hosszabb szöveggel és nagyobb képpel.

Mint a fenti képen is látszik a webalkalmazás megtekintéséhez nem szükséges regisztrálnunk vagy bejelentkeznünk. Ebben a formában megtekinthetjük az utazásokat. De csak regisztrált illetve bejelentkezett felhasználóink foglalhatnak illetve ekkor érhető el az utakhoz tartozó árajánlat is.

## 4.2 Bejelentkezés és regisztráció

* A felhasználó a bejelentkezés/regisztráció menüpont(fül) kiválasztásakor egy **modál ablakban megjelenő űrlap** fogadja. Tehát nem irányítja a felhasználót egy új oldalra, hanem egy felugró ablakban intézheti a belépést vagy a regisztrációt.
* Új felhasználók a „Regisztráció” részen adhatják meg nevüket, email címüket, jelszavukat.
* Sikeres regisztráció után automatikusan be lehet jelentkezni.
* A rendszer a megadott adatok formai helyességét (pl. email formátum) ellenőrzi, és hibaüzenetet jelenít meg szükség esetén.



## 4.3 Utazások oldal

* A navigációs menüből elérhető „Utazások” oldalon **az összes elérhető utazás** megtekinthető.
* Itt is ugyanaz a kártyás megjelenítés használatos, azonban minden aktív utazás listázva van, nem csak a legújabb három.
* A felhasználó bármelyik utazás kártyájáról tovább léphet a foglalási folyamatba.

## 4.4 Foglalásaim

* A foglalásaim menüpontban a felhasználó megtekintheti:
  + az általa korábban rögzített foglalásokat,
  + az utazás célállomását, időpontját, szállás nevét,
  + az utasok számát és a foglalás dátumát.
* Ez az oldal **csak bejelentkezett felhasználók** számára érhető el! Amennyiben nincs bejelentkezve a felhasználó nem is látszik a navigációs menüben ez a lehetőség.



## 4.5 Kapcsolat oldal használata

A „Kapcsolat” menüpont a navigációs sávból érhető el, és lehetőséget biztosít a látogatók számára, hogy **üzenetet küldjenek az oldal üzemeltetőinek**. Ez különösen hasznos, ha:

* kérdésük van egy adott utazással kapcsolatban,
* problémát tapasztaltak a foglalás során,
* javaslatot szeretnének tenni az oldal működésével kapcsolatban.

Kapcsolati űrlap mezői

A kapcsolatfelvételi űrlap egyszerű, átlátható mezőket tartalmaz:

|  |  |  |
| --- | --- | --- |
| Mező | Kötelező | Leírás |
| Név | ✅ | A felhasználó neve, teljes neve |
| Email cím | ✅ | A válaszadásra használt email cím. Formailag ellenőrzött. |
| Üzenet szövege | ✅ | A tényleges kérdés, kérés vagy észrevétel. Több soros szövegmező. |
| Üzenet tárgya(fejlesztés alatt) | -- | Az üzenet rövid címe. |

Üzenetküldés folyamata

1. A felhasználó kitölti az összes kötelező mezőt.
2. Az űrlap ellenőrzi, hogy minden adat helyesen van-e megadva (pl. email formátum).
3. Az üzenet a **contact\_messages** adatbázistáblába kerül mentésre.
4. Sikeres mentés után a rendszer visszajelzést küld:  
    **„Köszönjük, hogy kapcsolatba lépett velünk. Hamarosan válaszolunk!”**

Adatvédelem

* A kapcsolatfelvétel során megadott adatokat **nem tároljuk véglegesen**, csak a kapcsolattartás céljából kerülnek rögzítésre.
* Az email címeket nem osztjuk meg harmadik féllel.

Üzenetek kezelése

Az admin oldalon történik az üzenetek kezelése és listázása.

## 4.6 Kijelentkezés

* A navigációs sávban található „Kijelentkezés” gomb lehetővé teszi a biztonságos kilépést.
* Kijelentkezés után a felhasználót visszairányítja a kezdőoldalra.

## 4.7 Foglalás menete

A foglalás funkció csak **bejelentkezett felhasználók számára elérhető**. A kiválasztott utazás részleteit megtekintve, a felhasználó egy gomb segítségével elindíthatja a foglalási folyamatot.

## 4.8 A foglalás lépései:

1. **Utazás kiválasztása**  
    A „Utazások” menüpontban szereplő ajánlatok közül a felhasználó kiválaszt egy számára szimpatikus utat, és a „Részletek” gombra kattintva megtekinti a hosszabb leírást.
2. **Foglalási űrlap kitöltése** Az űrlapon a következő mezők találhatók:
   1. **Utasok száma:** A rendszer figyeli, hogy ne lehessen többet választani, mint az elérhető férőhelyek száma.
   2. **Szállás kiválasztása:** Csak azok a szállások jelennek meg, amelyek a kiválasztott úticélhoz tartoznak.
   3. (A felhasználó adatai már előre kitöltésre kerülnek, de ez rejtett.)
3. **Foglalás véglegesítése** és a Foglalás gomb megnyomása :Az adatok megadása után a felhasználó elküldi a foglalást. Ekkor:
   1. A foglalás bekerül a bookings adatbázistáblába.
   2. Az utazás available\_spots értéke csökken az utasok számával.
   3. Megjelenik egy sikeres visszajelzés:  
       **„Köszönjük! A foglalása sikeresen rögzítésre került.”**
4. **Foglalás megtekintése** A felhasználó a „Foglalásaim” oldalon visszanézheti a lefoglalt utazásait és szállását, valamint az utasok számát.

Hibakezelés:

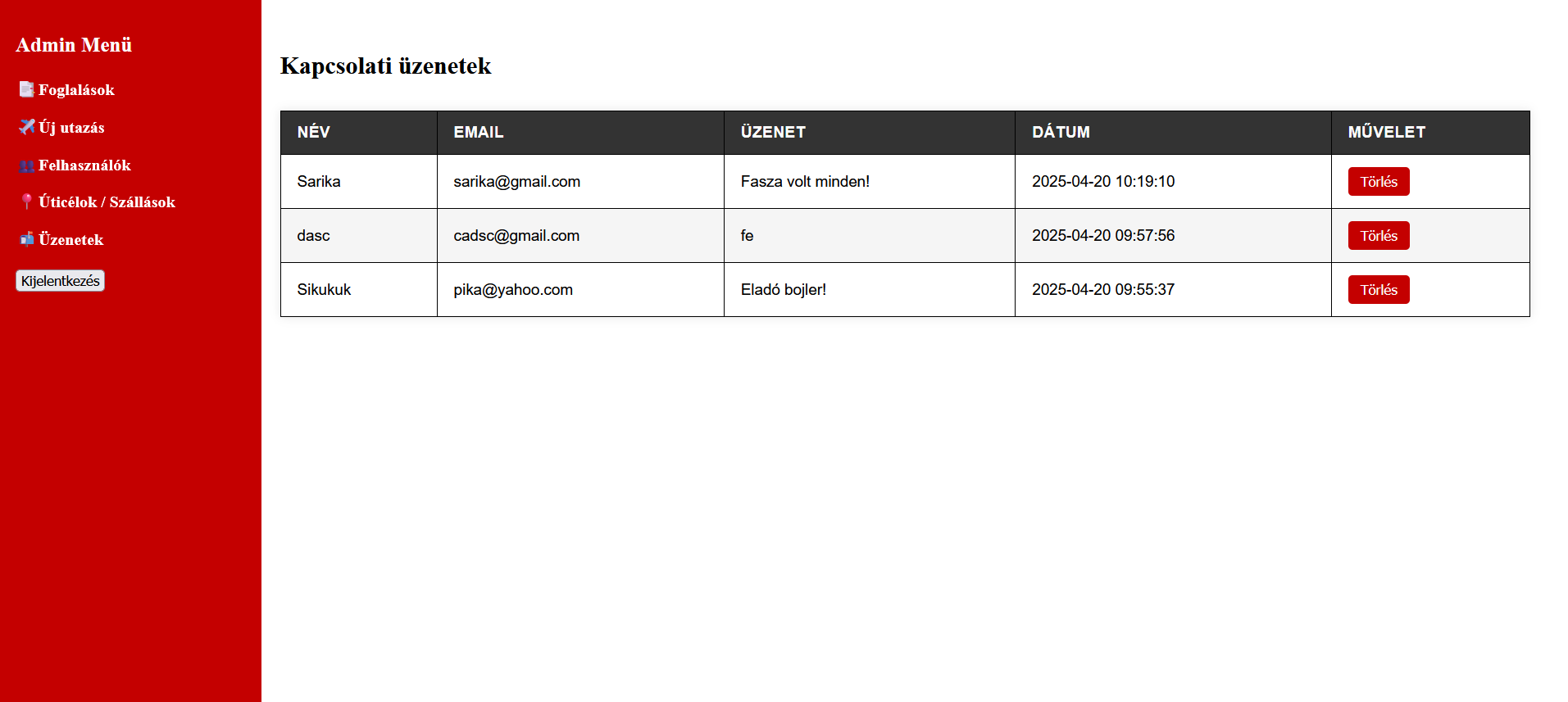
* A rendszer nem engedi a foglalást, ha a kiválasztott utasszám meghaladja az elérhető férőhelyeket.
* Az űrlap minden mezőjének kitöltése kötelező.
* Érvénytelen adatok (pl. negatív utasszám) esetén hibaüzenet jelenik meg.
* Csak szám vihető fel az utasok számánál.

# 5. Admin felület használata

* Az adminisztrátorok külön belépés után férnek hozzá a **public/admin** által betöltött admin felülethez.
* Az oldal bal oldalán található **oldalsó menü**, amely az alábbi lehetőségeket kínálja:

## 5.1 Foglalások kezelése

* Az összes rögzített foglalás megtekinthető felhasználókra lebontva.
* A foglalások törölhetők, ha szükséges, illetve módosíthatóak.



## 5.2 Új utazás felvitele

* A „Utazás felvétele” menüpont alatt az admin rögzíthet:
  + célállomást,
  + árakat,
  + indulási/érkezési dátumokat,
  + elérhető férőhelyeket,
  + rövid és hosszú leírást
  + képfájlt az utazáshoz.

## 5.3 Új szállás vagy úticél felvitele

* Az admin megadhat új országot/várost (úticélt), amelyhez később utazásokat lehet kapcsolni.
* Szállásokat is lehet rögzíteni, amelyek hozzákapcsolódnak egy célállomáshoz.

## 5.4 Felhasználók kezelése

* A regisztrált felhasználók listája megtekinthető.
* Szükség esetén törölhetők és módosíthatóak.

## 5.5 Felhasználói és egyéb üzenetek kezelése

A beérkezett kapcsolatfelvételek az admin felületen **külön menüpontban** „Üzenetek” pontban találhatóak.

* Minden bejegyzés tartalmazza a feladó nevét, email címét, üzenetet és beküldési dátumot.
* Az admin jogosult a nem releváns vagy megválaszolt üzenetek **törlésére**.
* Az admin felület célja az átlátható ügyintézés és gyors reagálás elősegítése.

# 6. Hibakezelés / hibaüzenetek

Mi történik, ha pl. hibás email-cím, nincs elérhető férőhely, üres űrlap mező stb.?

**„A rendszer minden fontos felhasználói műveletet (foglalás, regisztráció, kapcsolatfelvétel) során ellenőrzi a bevitt adatokat, és figyelmeztető üzenetet jelenít meg, ha hiányzó vagy hibás adatot talál. Például: 'Érvénytelen e-mail cím', 'Nincs elérhető férőhely', vagy 'Töltse ki az összes mezőt!'”**

# 7. Használt technológiák:

## 7.1 Frontend (Felhasználói felület)

* **HTML5**  
   A weboldal szerkezeti felépítéséhez HTML5 nyelvet használtunk. Ez biztosítja a korszerű és szemantikailag jól strukturált tartalom kialakítását, amely a keresőmotorok és a segédeszközök számára is optimális.
* **CSS3**  
   A stílusozáshoz modern CSS3-at használtunk, beleértve:
  + saját stíluslapokat (pl. style.css, modal.css),
  + responsív megoldásokat (rugalmas rácsszerkezet, média lekérdezések),
  + popup (modál) megjelenítés formázását.
* **JavaScript**  
   Az interaktív elemek (modálablakok megnyitása/zárása, dinamikus tartalom megjelenítése) JavaScript segítségével valósultak meg.  
   Fő fájl: modal.js – ez felelős a regisztrációs/bejelentkezési és utazás részleteihez tartozó popup ablakok kezeléséért.

## 7.2 Backend (Szerveroldali logika)

A backend oldal PHP nyelven készült, és egy egyszerű, de strukturált MVC (Model–View–Controller) architektúrára épül, amely segíti a logikus felépítést, a könnyű bővíthetőséget és a kód újrafelhasználhatóságát is biztosítja. Végül megfelel a “Tiszta Kód” elvének.

* **MVC struktúra**
  + models/ – az adatbázis műveleteket végző osztályokat tartalmazza (pl. Trip, Booking).
  + controllers/ – vezérlők, amelyek feldolgozzák a kéréseket (pl. TripController, BookingController).
  + views/ – sablonok, amelyek megjelenítik az adatokat (pl. trips.php, index.php, modals).
  + router.php – központi fájl az útvonalak kezelésére.

## 7.3 Fájlszerkezet

* **controllers/ mappa:**  
   Itt található minden vezérlőosztály, amely a felhasználói kérések feldolgozásáért felel.  
   Példák:
  + HomeController – főoldali tartalom betöltése
  + TripController – utazások listázása, részletek, új utak rögzítése
  + BookingController – foglalások kezelése, foglalás mentése
  + AuthController – regisztráció, bejelentkezés, kijelentkezés
  + AdminController – admin funkciók
* **models/ mappa:**  
   Az egyes adatbázistáblákhoz tartozó osztályok találhatók itt. Ezek az osztályok felelősek a SELECT, INSERT, DELETE stb. műveletekért.
  + Pl. Trip.php, User.php, Booking.php, Accommodation.php
  + A Trip modell például JOIN-t is használ a kapcsolódó szállások lekérésére.
* **views/ mappa:**  
   Itt kerülnek elhelyezésre a felhasználónak megjelenő sablonok (HTML + PHP vegyesen), pl. a kezdőlap (index.php), utazások listája (trips.php), részletmodálok stb.
* **router.php:**  
   A központi útválasztó fájl, amely feldolgozza az URL-ben érkező útvonalakat, és meghívja a megfelelő vezérlőt és metódust.  
   Példa:
  + /trips → TripController::index()
  + /book → BookingController::store()

## 7.4 Főbb funkciók működése

* **Foglalás logika (BookingController::store)**
  + Ellenőrzi, hogy az utazáshoz elérhető férőhelyek megfelelnek-e a kért utasszámnak.
  + Ha igen, létrehozza a foglalást, és csökkenti az available\_spots értéket.
  + Az adatokat $\_POST-on keresztül kapja meg, és tisztítva, validálva kerülnek feldolgozásra.
* **Regisztráció és bejelentkezés**
  + A jelszavakat password\_hash() és password\_verify() segítségével biztonságosan kezeljük.
  + A bejelentkezett felhasználó adatai $\_SESSION változókban tárolódnak.
  + Az admin jogosultság szerepkör alapján kerül ellenőrzésre (role mező).
* **Kapcsolatfelvétel mentése**
  + Az üzenetek bekerülnek a contact\_messages táblába.
  + A mentés előtt minden mező tartalma ellenőrzésre kerül.
  + Az admin felületen külön nézetből kezelhetők.

## 7.5 Biztonsági szempontok

* **SQL injection ellen védekezés:**
  + Az adatbázis lekérdezések előkészített SQL utasításokkal (prepared statement) történnek, így megakadályozzuk a veszélyes adatbevitelt.
* **XSS védelem:**
  + Az űrlapoknál minden felhasználói bemenetet tisztítunk (htmlspecialchars, filter\_input stb.).
* **Session kezelés:**
  + Minden bejelentkezett művelethez ellenőrzés történik, hogy a felhasználó valóban be van-e jelentkezve.
  + Az admin funkciók kizárólag akkor érhetők el, ha a felhasználó role = 'admin'.

## 7.6 Adatbázis

* **MySQL (MariaDB)**  
   Az adatbázis kezelése XAMPP-on keresztül történt, a táblák közötti kapcsolatok idegen kulcsokkal lettek kialakítva. A táblák közé tartozik:  
   accommodations, admins, bookings, contact\_messages, clients, destination, trips.
* **phpMyAdmin**  
   Az adatbázis böngészésére, manuális módosításaira, valamint a táblák létrehozására a phpMyAdmin felületét használtuk.

## 7.7 Fejlesztési eszközök

* **Visual Studio Code**  
   A forráskód fejlesztését a VS Code szerkesztőben végeztük, amely a beépített fájlrendszer-nézet, kódkiemelés és Git-integráció révén nagyban megkönnyítette a munkát.
* **XAMPP**  
   A projekt lokális futtatásához a XAMPP csomagot használtuk, amely tartalmazza a PHP-t, Apache szervert és MySQL adatbázist.
* **Git és GitHub**  
   Verziókövetéshez a Git rendszerét használtuk, a távoli tároláshoz pedig GitHub-ot. Ez lehetővé tette a folyamatos mentést és a projekt biztonságos archiválását.
* **dbdiagram.io** (vizuális adatmodell készítéshez)

# 8. Rendszerfelépítés

## 8.1 Mappa szerkezet

Az alábbiakban bemutatjuk a főbb technikai struktúrát és a mappaszerkezetet.

Pagoda\_travel/

├── public/ # Nyilvánosan elérhető fájlok (index.php, admin.php)

│ ├── index.php # A nyitóoldal belépési pontja

│ ├── admin.php # Admin felület belépési pontja

│ ├── css/ # Stíluslapok (pl. style.css, modal.css)

│ ├── js/ # JavaScript fájlok (pl. modal.js)

│ └── uploads/ # Feltöltött képek

├── controllers/ # Kéréseket kezelő vezérlők

│ ├── HomeController.php

│ ├── TripController.php

│ ├── BookingController.php

│ ├── AuthController.php

│ └── AdminController.php

├── models/ # Az adatbázissal kommunikáló osztályok

│ ├── Trip.php

│ ├── User.php

│ ├── Booking.php

│ └── Accommodation.php

├─└──Admin.php

├── views/ # Megjelenítő sablonok (HTML + PHP)

│ ├── home/ # Kezdőlap, utazások stb.

│ ├── admin/ # Admin oldalak

│ │ └── pages/ # Admin aloldalak (pl. bookings.php, users.php)

│ ├── partials/ # Közös elemek (header.php, footer.php, modals.php)

│ └── auth/ # Bejelentkezés, regisztráció modál

│ ├──contacts #Kapcsolati oldal

│ ├──errors #Hiba sablonok(pl 404)

├── router.php # Központi útválasztó fájl

├── config.php # Adatbáziskapcsolat, konfigurációs adatok

└── .htaccess # URL átirányítás

## 8.2 Kérésfeldolgozás menete

1. A böngésző egy URL-t kér le (pl. site.com/trips)
2. A public/index.php vagy router.php fájl elemzi a kérést.
3. A router.php meghatározza, hogy melyik vezérlő osztály melyik metódusát kell meghívni.
4. A vezérlő (Controller) szükség esetén lekéri az adatokat a Model osztályon keresztül.
5. Az adatok megjelenítése a View fájlok (HTML + PHP) segítségével történik.

Háromféle felhasználói szint

* **Vendég felhasználó:** utazások megtekintése, kapcsolat üzenet írása
* **Regisztrált felhasználó:** utazások megtekintése, foglalás, kapcsolatfelvétel
* **Adminisztrátor:** külön admin felület, oldalsó menüvel, tartalomkezelési lehetőségekkel

Fejlesztői előnyök

* **Könnyű karbantartás:** a fájlok funkció szerint szétválasztva
* **Átláthatóság:** minden oldal sablon és logika külön fájlban
* **Bővíthetőség:** új vezérlők, modellek vagy nézetek könnyen hozzáadhatók

A webalkalmazás működését a **kliens–szerver architektúra** és az **MVC (Model–View–Controller)** elv biztosítja. A felhasználói interakciók (pl. bejelentkezés, foglalás, kapcsolatfelvétel) egy jól szervezett folyamat során kerülnek feldolgozásra a háttérben, majd az eredmények visszajelzésként jelennek meg a felhasználónak.

## 8.3 A működés általános folyamata

1. **A felhasználó egy oldalt kér le a böngészőből** (pl. <https://pagodatravel/trips>)
2. A kérés a public/index.php vagy router.php fájlhoz érkezik, amely **elemzi az URL-t**, és meghatározza, hogy melyik vezérlő (Controller) és melyik metódus kezelje azt.
3. A vezérlő szükség esetén kapcsolatba lép a megfelelő modell osztállyal (Model), hogy adatokat kérjen le vagy mentsen az adatbázisban.
4. Az eredmény (pl. utazások listája) egy nézeten (View) keresztül kerül megjelenítésre.
5. A válasz visszakerül a böngészőbe, ahol a felhasználó azt megtekintheti vagy további műveletet végezhet.

## 8.4 Működési példák funkciókra bontva

Bejelentkezés

* A felhasználó kitölti a bejelentkezési űrlapot (modálban).
* Az adatok a POST metódussal továbbításra kerülnek az AuthController::login() metódushoz.
* A rendszer ellenőrzi az email cím és jelszó párosát (password\_verify()).
* Sikeres bejelentkezés esetén a felhasználó adatai bekerülnek a $\_SESSION tömbbe.
* A rendszer visszairányítja a kezdőlapra, és a navigációs sáv ennek megfelelően változik.

Utazások listázása

* A TripController::index() lekéri az utazásokat az adatbázisból a Trip modellen keresztül.
* Az utazások adatai átadásra kerülnek a views/index.php vagy views/trips/list.php nézetnek.
* A kártyák a frontenden HTML és CSS segítségével jelennek meg.

Foglalás

* A felhasználó kiválasztja az utazást, majd rákattint a „Foglalás” gombra.
* A BookingController::store() metódus:
  + ellenőrzi a kiválasztott férőhelyek számát,
  + csökkenti az available\_spots mezőt a trips táblában,
  + rögzíti a foglalást a bookings táblában.
* A visszajelzés (sikeres foglalás) megjelenik a felhasználónak.

Foglalásaim

* A rendszer ellenőrzi, hogy a felhasználó be van-e jelentkezve.
* A BookingController::userBookings() lekéri az adott felhasználó foglalásait (JOIN a bookings, trips, accommodations táblák között).
* A nézet egy listát jelenít meg a foglalás részleteiről.

Kapcsolatfelvétel

* A látogató kitölti a kapcsolat űrlapot (név, email, üzenet).
* Az adatok a ContactController::store() vagy hasonló metóduson keresztül bekerülnek a contact\_messages táblába.
* A felhasználó visszajelzést kap a sikeres elküldésről.
* Az üzenetek az admin felületen listázhatók.

Admin funkciók működése

* Az admin bejelentkezés után a public/admin.php fájl alapján töltődnek be az aloldalak a GET['page'] paraméter segítségével.
* Például: admin.php?page=bookings → betölti a views/admin/pages/bookings.php fájlt.
* Az admin felület külön kezeli:
  + új utazás rögzítését,
  + szállások és úticélok hozzáadását,
  + foglalások és felhasználók kezelését,
  + kapcsolatfelvételek listázását.

Biztonsági működés

* **Session ellenőrzés** minden olyan oldalnál, ami csak bejelentkezett felhasználók számára elérhető.
* **Role alapú hozzáférés** az admin funkciókhoz.
* **Validáció és szűrés** minden beviteli mezőnél (pl. email, szám mezők).
* **Adatbázis biztonság:** az SQL lekérdezések prepared statement vagy mysqli\_real\_escape\_string segítségével készülnek, így megelőzhetők az SQL injection támadások.

# 9. Adatmodell

## 9.1 Adatbázis Dokumentáció

**Cél**

A projekt célja egy olyan utazásszervező rendszer létrehozása, amely lehetővé teszi az ügyfelek számára utazások foglalását, szállásválasztást, valamint a fizetések kezelését. Az adatbázis célja ezen adatok strukturált, összefüggő és hatékony tárolása, kezelése és lekérdezése. Az adatmodell biztosítja az adatok logikus, integrált és visszakereshető tárolását, valamint az entitások közötti kapcsolatok átlátható kezelését.

**Eszközök és technológiák**

Az adatbázis létrehozásához és működtetéséhez az alábbi technológiákat alkalmaztuk:

* **XAMPP** (Apache + MariaDB)
* **MariaDB** (MySQL-kompatibilis relációs adatbázis-kezelő)
* **MySQL nyelv** az adatbázis lekérdezésekhez és táblák létrehozásához
* **dbdiagram.io** a vizuális adatmodell megtervezéséhez

**Adatbázis szerkezete**

Az adatbázis relációs modellre épül, normalizált szerkezettel. Az alábbi táblákat tartalmazza:

**1. clients**

A regisztrált felhasználók adatai.

* id (PK) – egyedi azonosító
* name, email, password – alapadatok (a jelszó jelszóhash formátumban, VARCHAR(255))

**2. admins**

Az adminisztrátorok belépési adatai.

* id (PK)
* username, password

**3. destinations**

Az elérhető úticélok (ország, város, név).

* id (PK)
* destination, city, country

**4. trips**

Az elérhető utazások adatai.

* id (PK)
* destination\_id (FK) – kapcsolódik a destinations táblához
* departure, arrival, price\_per\_person, available\_spots
* image\_url, short\_description, full\_description

**5. accommodation**

A szálláshelyek, amelyek úticélhoz kapcsolódnak.

* id (PK)
* name\_of\_accommodation, address, rating, price\_per\_night\_per\_per
* destination\_id (FK)

**6. bookings**

A felhasználók foglalásai.

* id (PK)
* clients\_id (FK), trips\_id (FK), accommodation\_id (FK)
* booking\_date, number\_of\_passengers, total\_price (DECIMAL)

**7. payments**

A foglalásokhoz kapcsolódó fizetések.

* id (PK)
* bookings\_id (FK)
* date\_of\_payment, amount\_paid, method\_of\_payment (amount: DECIMAL)

**8. contact\_messages**

A felhasználók által küldött kapcsolatfelvételi üzenetek.

* id (PK)
* user\_id (FK – opcionális)
* name, email, message, created\_at (DATE vagy DATETIME)

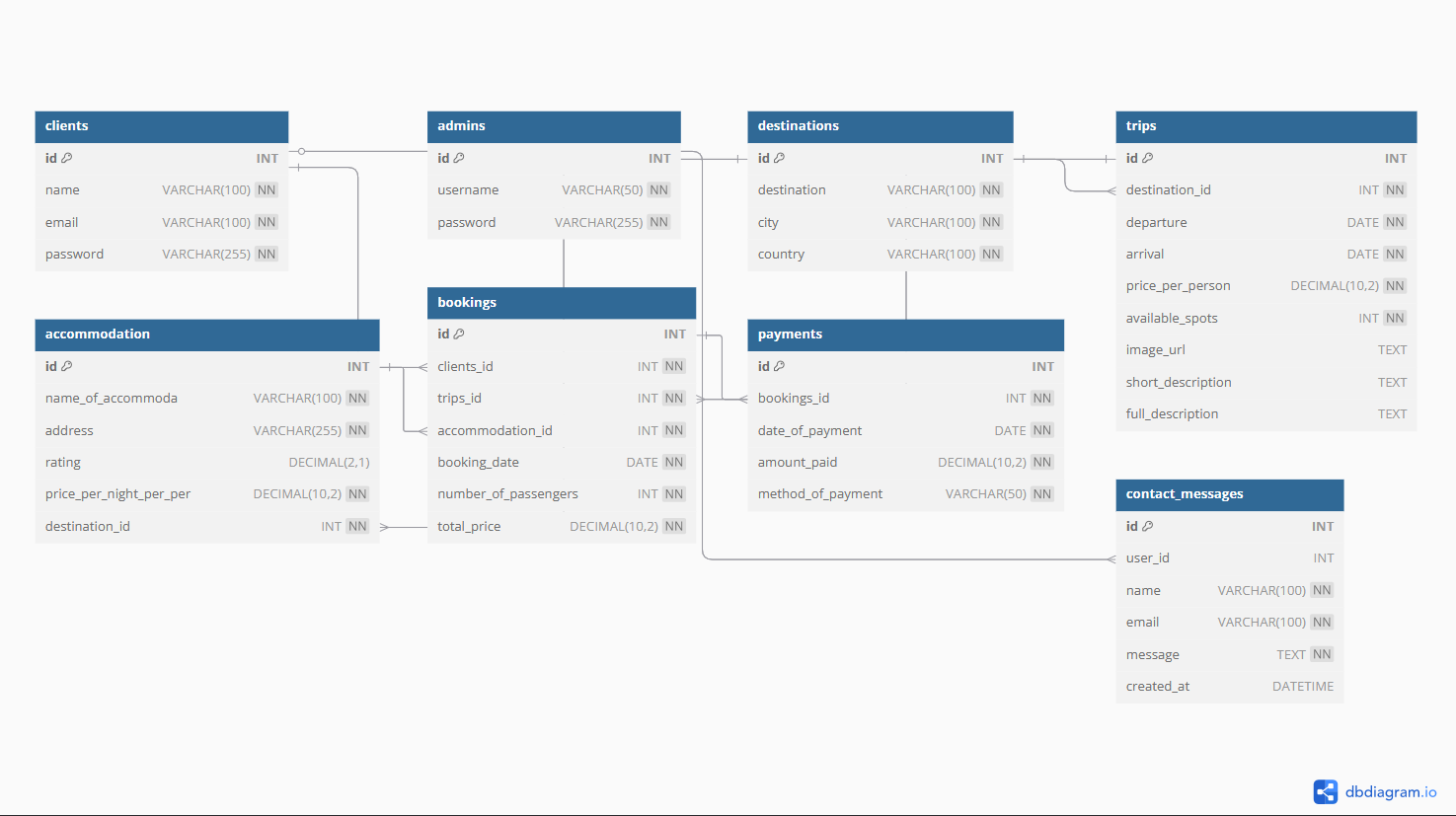
**Kapcsolatok a táblák között**

* Egy felhasználónak (clients) több foglalása (bookings) lehet. (1:N)
* Egy utazás (trips) több foglaláshoz kapcsolódhat. (1:N)
* Minden foglalás egy adott utazáshoz és szálláshoz kapcsolódik.
* Egy szálláshely egy adott úticélhoz tartozik.
* Minden utazás egy adott úticélhoz tartozik. (1:N)
* Egy foglaláshoz pontosan egy fizetés (payments) kapcsolódik. (1:1)
* A contact\_messages tábla opcionálisan kapcsolódik a clients táblához. (N:1)

**További megjegyzések**

* A jelszavak VARCHAR(255) mezőben tárolódnak, titkosított formátumban.
* A pénzügyi mezők DECIMAL típusúak a pontos számábrázolás érdekében.
* Az időbélyeg mezők (created\_at, booking\_date, date\_of\_payment) DATE vagy DATETIME típusúak.
* Az adatmodell lehetőséget biztosít a jövőbeni bővítésre (pl. utasértékelés, kedvezmények).

## 9.2 DB diagram Modell:



# 10. A program főbb függvényei és vezérlő logikája

Az alkalmazás a Model–View–Controller (MVC) architektúra alapján működik. Minden egyes funkció egy meghatározott vezérlőosztályban (Controller) található, amely a **felhasználói kéréseket kezeli**, majd szükség esetén kapcsolatba lép az adatbázist kezelő Model osztályokkal, és az eredményt továbbítja a megfelelő View sablonoknak.

Az alábbiakban bemutatjuk a legfontosabb vezérlők és azok kulcsfüggvényeinek működését.

AuthController – Bejelentkezés, regisztráció, kijelentkezés

* **register()**  
   Feldolgozza a regisztrációs űrlapot:
  + Ellenőrzi az email cím formátumát és jelszó hosszát.
  + A jelszót password\_hash() függvénnyel titkosítja.
  + Sikeres regisztráció után a felhasználó automatikusan bejelentkezik.
* **login()**
  + Megkeresi a felhasználót email alapján.
  + Ellenőrzi a jelszót password\_verify()-vel.
  + Siker esetén beállítja a session-öket ($\_SESSION['user']), és átirányítja a főoldalra.
* **logout()**
  + Megsemmisíti a session-t, ezzel kijelentkezteti a felhasználót.

HomeController – Kezdőlap, 3 utazás listázása

* **index()**
  + Lekéri az adatbázisból a legfrissebb 3 utazást.
  + Ezeket továbbítja a index.php nézetnek.
  + Vendégként is elérhető.

TripController – Utazások megjelenítése és kezelése

* **index() / list()**
  + Lekéri az összes utazást, valamint hozzájuk tartozó úticélt és szállásokat.
  + A lekérdezés során SQL JOIN-t alkalmaz, hogy a kapcsolódó adatok egy lekérdezésben visszatérjenek.
* **show($id)**
  + Egy adott utazás részleteit tölti be (pl. hosszú leírás, kép, időpontok).
  + Az eredmény modálablakban jelenik meg.
* **store() [Admin]**
  + Új utazás rögzítése az admin felületen keresztül.
  + Feldolgozza az űrlap mezőit és a képfájlt, majd beszúrja az adatokat a trips táblába.

BookingController – Foglalások

* **store()**
  + Foglalási adatok mentése: utasok száma, utazás, szállás.
  + Ellenőrzi, hogy van-e elég elérhető férőhely.
  + Csökkenti az available\_spots értéket a trips táblában.
  + A teljes foglalás bekerül a bookings táblába, majd visszajelzést ad.
* **userBookings()**
  + A bejelentkezett felhasználó összes foglalását listázza.
  + A bookings, trips és accommodation táblákat össze JOIN-olja.
* **destroy($id)** [opcionális]
  + Egy foglalás törlése (admin jogosultság szükséges hozzá).

AccommodationController (vagy TripController része)

* **getByDestination($destination\_id)**
  + Lekérdezi az adott úticélhoz tartozó szállásokat.
  + A foglalási űrlapon ez alapján szűrt listát lát a felhasználó.

Admin funkciók (általában külön fájlokban vagy admin.php alatt dinamikusan)

* A public/admin.php fájl a GET['page'] paraméter alapján betölti a megfelelő admin nézetet (bookings.php, users.php, trip\_form.php, stb.).
* A műveletek különböző vezérlőkhöz kapcsolódnak, de a logika ugyanaz:
  + Adatok lekérése az adatbázisból
  + Táblázatos megjelenítés
  + Törlés, szerkesztés, új elem rögzítése

Hibakezelés és validáció

* Minden űrlap tartalmaz **frontenden** ellenőrzést (pl. kötelező mezők, email formátum).
* A backend oldalon a függvények empty(), filter\_var() vagy saját validáló logikát alkalmaznak.
* Hibás vagy hiányos adatok esetén hibaüzenet jelenik meg.

# 11. Tesztelés:

A projekt jelen stádiumában több lépcsőben is tesztelésre került, hogy biztosítsuk a rendszer **megbízható működését**, a **hibák kiszűrését**, és a **felhasználói élmény stabilitását**. A tesztelés során figyelmet fordítottunk a funkcionális, logikai és biztonsági ellenőrzésekre is.

## 11.1 Tesztelési módszerek

* **Manuális tesztelés**  
   A funkciókat különböző felhasználói szerepkörökből (vendég, regisztrált, admin) manuálisan ellenőriztük a böngészőben.
* **Adatbázis ellenőrzés**  
   A phpMyAdmin segítségével figyeltük, hogy a regisztráció, foglalás, üzenetküldés során az adatok ténylegesen bekerülnek-e a megfelelő táblákba.
* **Hibakezelés tesztelése**  
   Direkt hibás adatokat próbáltunk beadni a rendszerbe (pl. rossz formátumú email, negatív utasszám), hogy ellenőrizzük a validációk működését.
* **Jogosultság tesztelés**  
   Olyan oldalakat próbáltunk megnyitni nem bejelentkezett felhasználóként, amelyeket csak belépve lenne szabad elérni (pl. foglalás, admin oldal).

## 11.2 Biztonsági ellenőrzések

* Jelszavak titkosítása password\_hash() segítségével.
* SQL injection elleni védelem: a lekérdezések paraméterezetten, tisztítva történnek.
* E-mail és input mezők szűrése filter\_var() és htmlspecialchars() használatával.

## 11.3 Tesztelés során tapasztalt hibák:

* A kapcsolatfelvételi űrlap nem adott visszajelzést – **javítva**.
* Az admin oldalon nem jelent meg egy újonnan hozzáadott utazás – **frissítés után javult**.

# 12. Továbbfejlesztési lehetőségek

A rendszer jelenlegi formájában egy működő, jól használható online utazási platformot biztosít, azonban a jövőben számos funkcióval szeretnénk tovább bővíteniu, hogy még kényelmesebb, informatívabb és interaktívabb legyen a felhasználók és az adminisztrátorok számára.

## 12.1 Online fizetés integrálása

* **Cél:** Lehetővé tenni, hogy a felhasználók közvetlenül az oldalon keresztül tudjanak bankkártyával fizetni.
* **Technológia:** Stripe
* **Előny:** Automatizált foglalási folyamat, kevesebb adminisztráció

## 12.2 Értékelési és visszajelzési rendszer

* **Cél:** A felhasználók értékelhessék az utazásokat és szállásokat (csillagozás, szöveges vélemény).
* **Tábla:** reviews (user\_id, trip\_id, rating, comment)
* **Előny:** Több információ új látogatóknak, közösségi élmény

## 12.3 E-mail értesítések

* **Cél:** Automatikus email küldése:
  + foglalás után,
  + kapcsolatfelvételre válaszként,
  + admin számára új foglalás vagy üzenet esetén.
* **Technológia:** PHPMailer, SMTP, SendGrid API

## 12.4 Foglalás törlése / módosítása felhasználói oldalon

* **Jelenleg:** csak admin tud törölni
* **Bővítés:** a felhasználók egy határidőn belül visszavonhatják vagy módosíthatják foglalásukat.

## 12.5 Teljes reszponzivitás és mobil optimalizálás

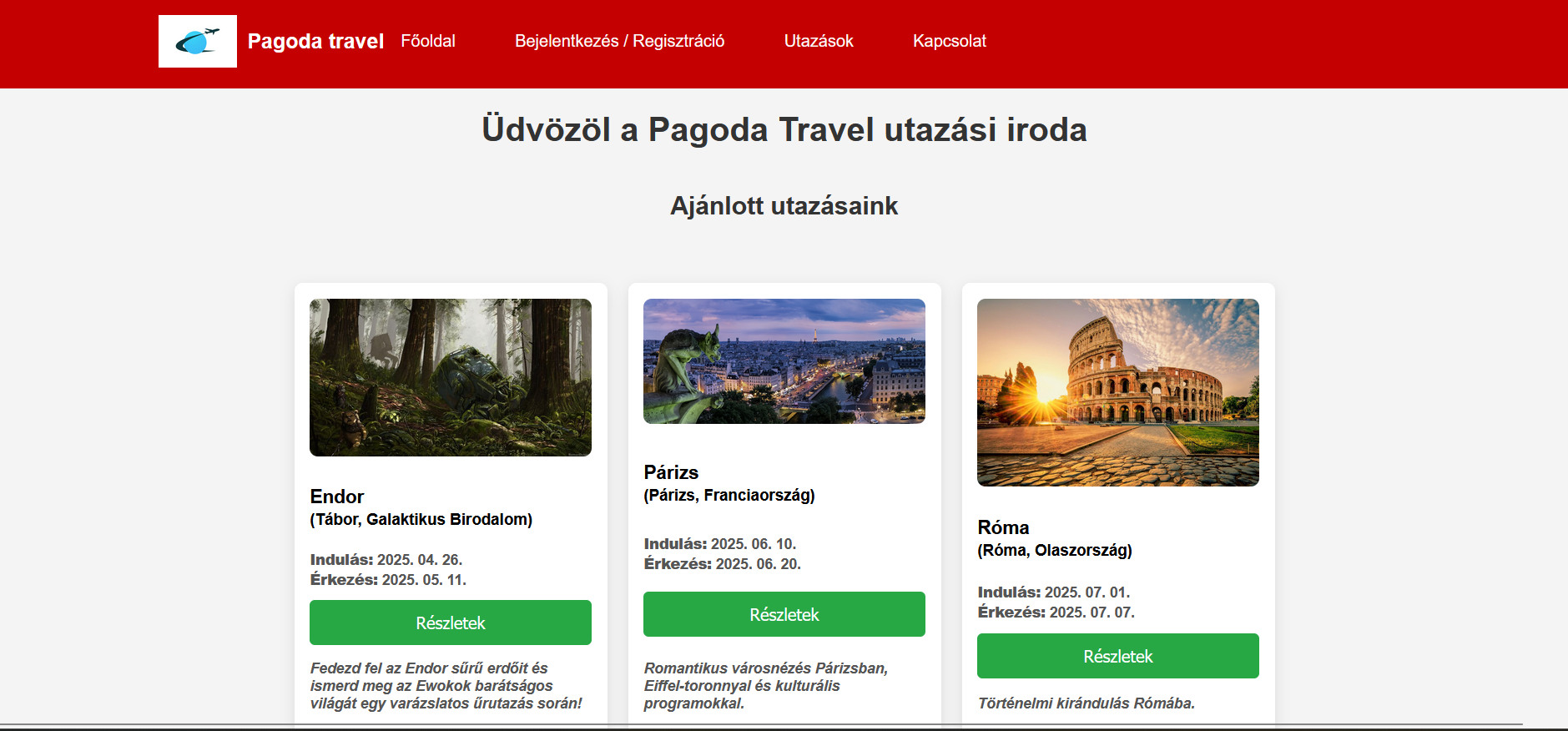
* **Jelenlegi állapot:** alap szintű reszponzivitás
* **Fejlesztés:** mobilmenü, nagy gombok, érintőbarát UI
* **Eszköz:** Bootstrap, Flexbox, Tailwind CSS

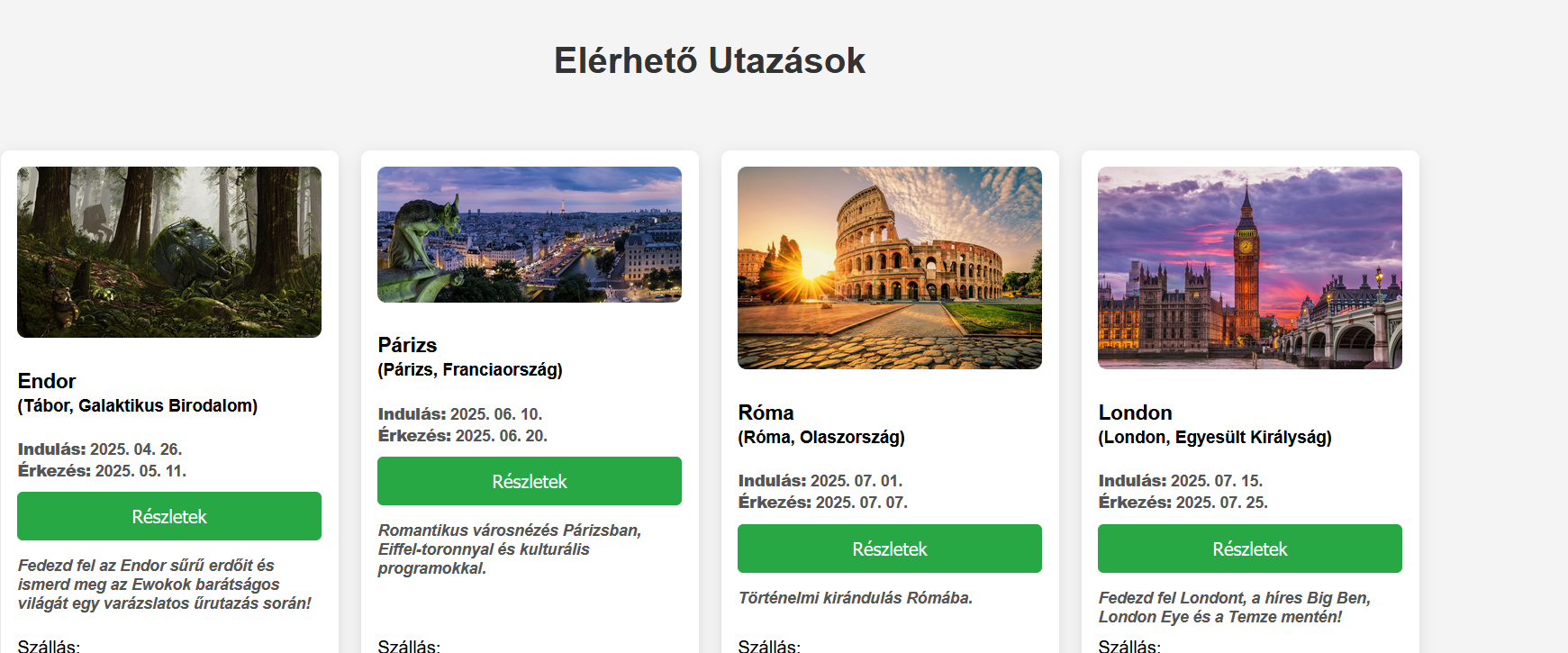
## 12.6 Több szerepkör bevezetése

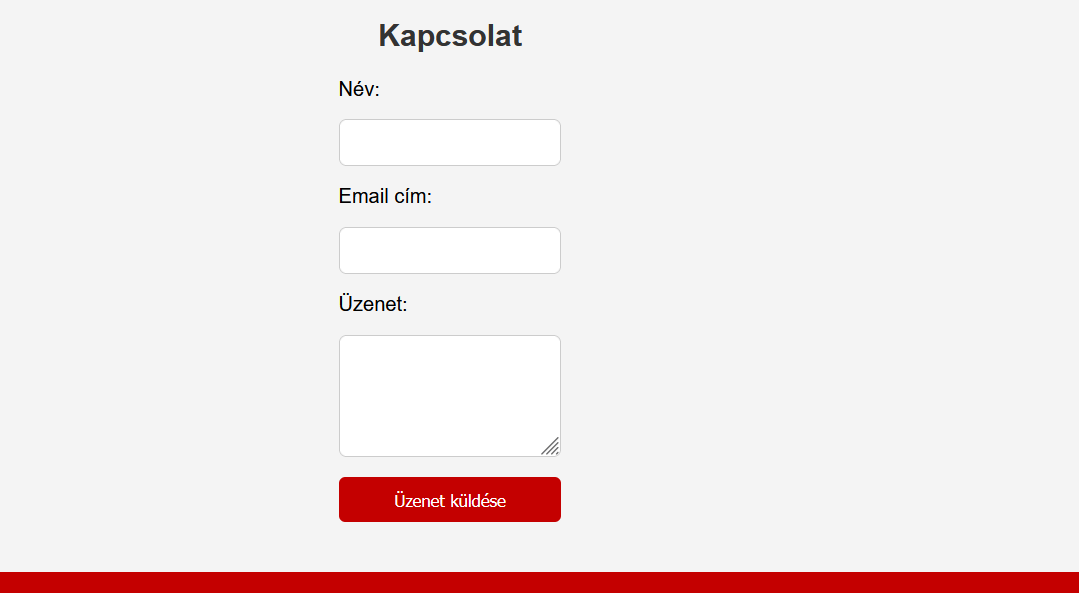
* Jelenleg: vendég, regisztrált, admin
* Bővítés: „moderátor” vagy „ügyintéző” szerepkör, aki csak foglalásokat kezelhet

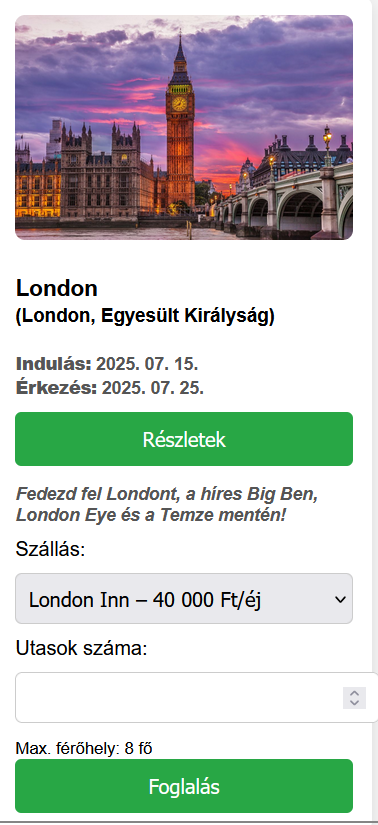
## 12.7 Kétlépcsős bejelentkezés

* Extra biztonság regisztrált vagy admin felhasználóknak
* SMS vagy email alapú megerősítés

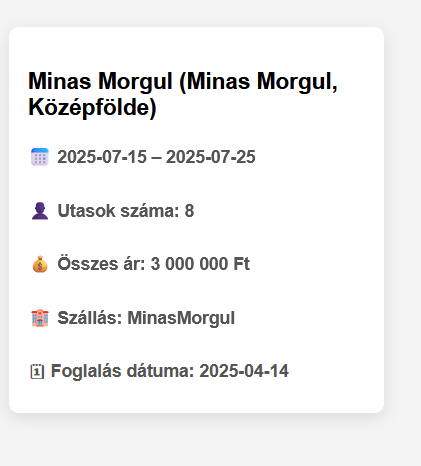
13. Képernyőképek*Index, a webalkalmazás főoldala, új látogatóknak.*

*Összes utazás, az utazások menüpontban.*

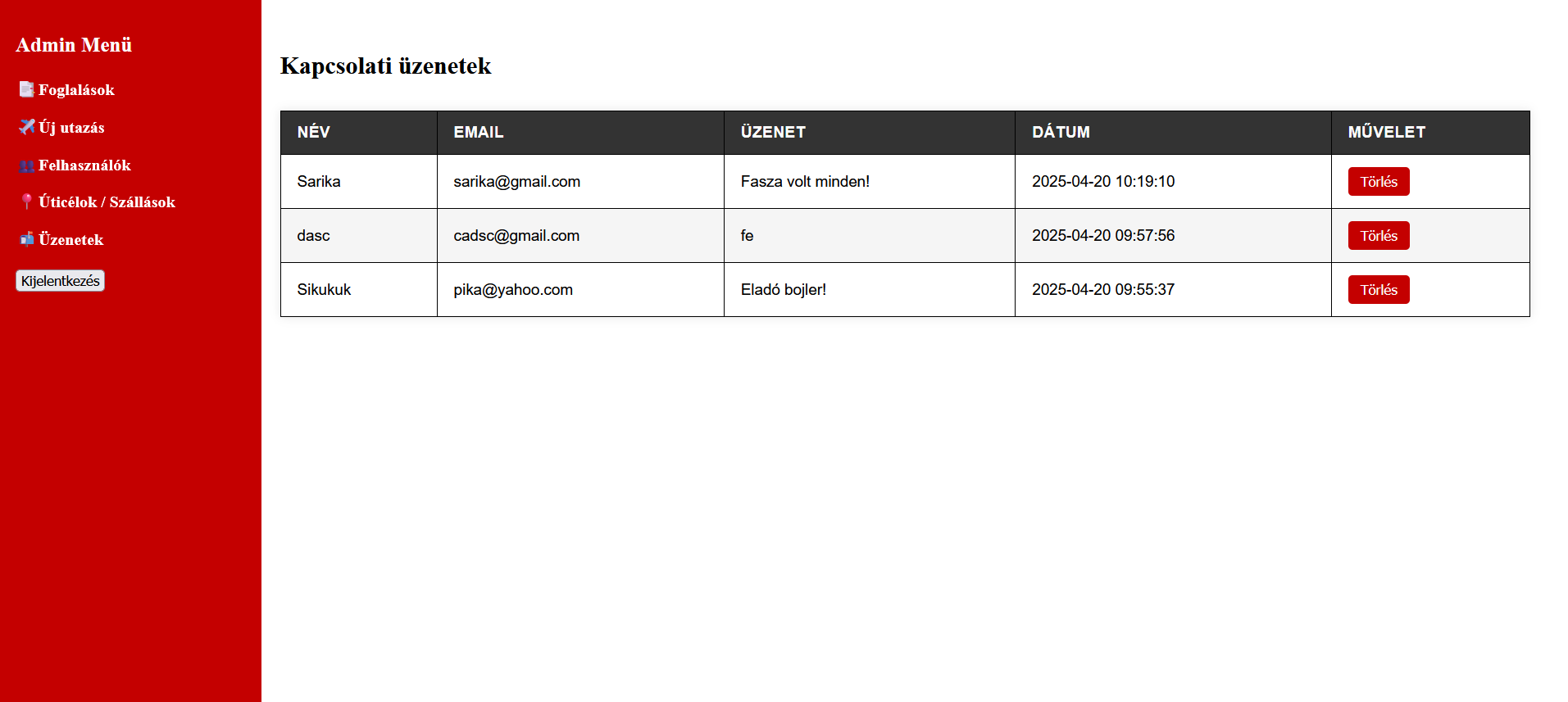
*Kapcsolat menüpont.*

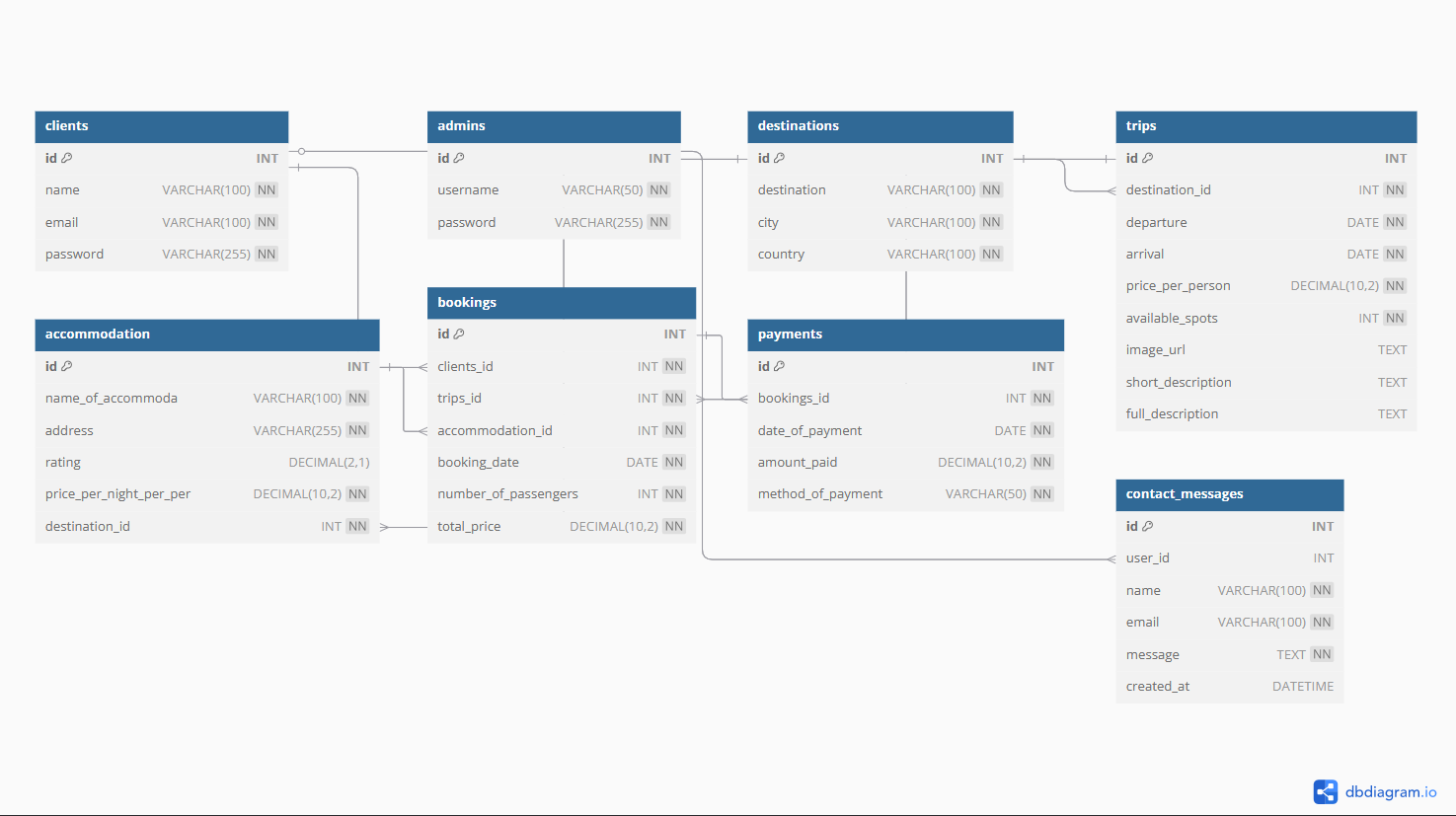


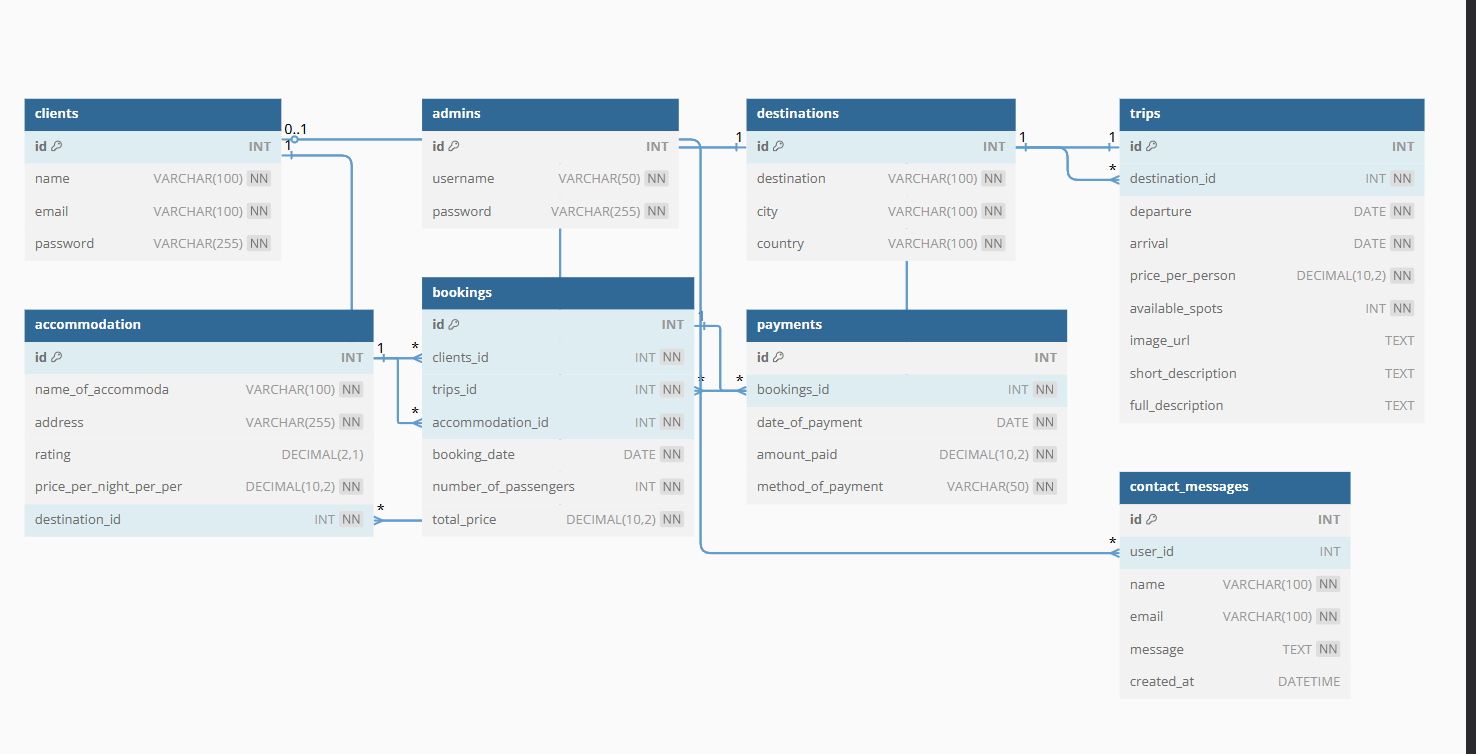
*Utazási kártya.*



*Foglalásaim menüpont alatt, kártya nézetben a foglalt utak.*

*Admin felület, felhasználói üzenetek listázása adatbázisból. Törlés funkcióval.*

*Db modell.*



# 14. A projekt forráskódja:

<https://github.com/oneilez/Pagoda-Travel-vizsgaremek>

# 15. Forrásmegjelölés, irodalomjegyzék

Minden részhez használt:

<https://www.w3schools.com/>

PDO:

<https://deadlime.hu/2006/02/11/mi-is-az-a-pdo/>

PDO 2:

<https://www.youtube.com/watch?v=Li90TQap1bc>

<https://www.youtube.com/watch?v=wcD5HlEHrnc>

PHP

<https://stackoverflow.com/questions>

Routing

<https://www.youtube.com/watch?v=siQUekpmImw>

<https://www.youtube.com/watch?v=caUXERCGZtI>

<https://stackoverflow.com/questions/42838053/post-route-via-form-vs-get-route-post-via-url-params>

Modal:

<https://www.youtube.com/watch?v=kjIS3JnrE9E>

<https://www.w3schools.com/howto/howto_css_modals.asp>