Budapesti Műszaki Szakképzési Centrum

Verebély László Technikum

5-0613-12-03 Szoftverfejlesztő és-tesztelő szakképesítés

RushWave Retreat  
Hotel foglaló weboldal

Készítette: Kovács Mihály Dániel, Vajda Dániel, Bárczi Bence

Konzulenstanárok: Somogyi Erika, Horváth Attila

2025. április

nyilatkozat a szakdolgozat eredetiségéről

Alulírott Kovács Mihály Dániel, Vajda Dániel és Bárczi Bence a BMSZC Verebély László Technikum 5-0613-12-03 Szoftverfejlesztő és-tesztelő képzésében részt vevő tanulók büntetőjogi felelősség tudatában nyilatkoznak és aláírásukkal igazolják, hogy a RushWave Retreat című vizsgaremek saját, önálló munkánk, és abban betartottuk az iskola által előírt, a vizsgaremek készítésére vonatkozó szabályokat

Tudomásul veszem, hogy a szakdolgozatban plágiumnak számít:

* Szó szerinti idézet közlése idézőjel és hivatkozás nélkül,
* tartalmi idézet hivatkozás megjelölése nélkül
* más publikált gondolatainak saját gondolatként való feltűntetése.

E nyilatkozat aláírásával tudomásul veszem továbbá, hogy plágium esetén szakdolgozatom visszautasításra kerül.

Budapest, 2025. április 16 \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Hallgatók aláírása

Tartalomjegyzék

[1. Bevezetés 4](#_Toc194591674)

[1.1. Követelményspecifikáció: 4](#_Toc194591675)

[2. Fejlesztői dokumentáció 5](#_Toc194591676)

[2.1. Alkalmazott fejlesztői eszközök 5](#_Toc194591677)

[2.2. Alkalmazott technológiák 5](#_Toc194591678)

[2.3. Futtatási környezet 5](#_Toc194591679)

[2.4. Fejlesztői ütemterv 5](#_Toc194591680)

[2.5. Fejlesztés menete 5](#_Toc194591681)

[2.5.1. Piackutatás 5](#_Toc194591682)

[2.5.2. Adatbázis megtervezése 6](#_Toc194591683)

[2.5.3. Főbb algoritmusok 11](#_Toc194591684)

[2.5.3.1. Weboldal 11](#_Toc194591685)

[2.5.3.2. Asztali alkalmazás 17](#_Toc194591686)

[2.5.4. Tesztelés 18](#_Toc194591687)

[3. Felhasználói dokumentáció, felhasználói kézikönyv 19](#_Toc194591688)

1. Bevezetés

Vizsgaremekünk témájának kiválasztása egy személyes problémából adódott, hiszen egyik közeli ismerősünk megemlítette, hogy milyen nehéz manapság egy koherens szálloda foglaló weboldalt találni, ami nem komplikálja túl a foglalás menetét.

Manapság a hotelláncok rengeteg mindent nyomnak a felhasználó arcába, ezzel néha átláthatatlanná teszik saját weboldalukat, ezért mi úgy döntöttünk, hogy készítünk egy letisztult és könnyen kezelhető weboldalt melynek segítségével az átlagember könnyedén tudja elintézni következő nyaralását.

Ezen felül szerettünk volna egy olyan kihívást kitűzni magunk elé, amely lehetőséget ad arra, hogy próbára tegyük saját tudásunkat és fejlesszük képességeinket. Úgy döntöttünk, hogy egy szálloda foglaló oldal erre egy tökéletes feladat, hiszen a koncepciója egyszerű, viszont számos extra kihívással bír.

Szeretnénk létrehozni egy olyan weboldalt, amely átlátható mindenki számára, miközben megőrzi a szálloda foglalás komplexitását, ezzel megnyitva a kaput különböző hotelláncoknak, hogy kérvényezzék saját hoteljeiket az oldalra, valamint lehetőséget adni a felhasználóknak, hogy találjanak egy számukra kedvező szokásos nyaralóhelyet.

* 1. Követelményspecifikáció:

Célunk, hogy legyen olyan weboldalunk, amely funkcionalitás terén képes lenne megállni a helyét a modern piacon. További célunk volt, hogy a legjobb élményt nyújtsuk a felhasználónak. A weboldal elsőszámú célja hogy a felhasználó tudjon csinálni egy fiókot, ahol követni tudja saját foglalásait illetve, hogy képes legyen foglalni az oldalon jelen lévő szállodák valamelyikébe. Ezek mellett fontosnak tartottuk, hogy egy felhasználó képes legyen véleményt nyílvánítani egy értékelés formájában.

A weboldalt két féle módon lehet megtekinteni, mint vendég, és mint fiókkal rendelkező felhasználó. A vendég ugyanúgy megtudja tekinteni a különböző szállodákat illetve azoknak értékeléseit, viszont ő maga nem tud foglalni vagy értékelést írni, ellentétben egy regisztrált felhasználóval. Ezentúl, a weboldalt számos böngészőben meglehet tekinteni, illetve mobil és táblagép eszközökkel is kompatibilis.

A C sharp alkalmazás lényege, hogy a különböző szállodák alkalmazottjai tudják kezelni a szállodák különböző adatait, mint például a szolgáltatások elérhetőségét, a szobák árát vagy a foglalások különböző aspektusait. Az alkalmazottak kizárólag azokat a hoteleket tudják kezelni, ahol dolgoznak, ezzel elkerülve a lehetséges szabotálást

1. Fejlesztői dokumentáció
   1. Alkalmazott fejlesztői eszközök

* Kódszerkesztő: Visual Studio Code, Visual Studio
* Fejlesztői környezet: Laravel, .NET
  1. Alkalmazott technológiák
* Programozási nyelvek: PHP, JavaScript, C#
* Webes technológiák: HTML, CSS
* Adatbázis motor: MariaDB (MySQL)
  1. Futtatási környezet

A weboldal bármely modern böngészőben működőképes, akár mobileszközön is megtekinthető.

A C# alkalmazás asztali számítógépeken elérhető, vagy egyéb olyan eszközön, amely Windows 10 vagy afeletti operációs rendszerrel rendelkezik.

* 1. Fejlesztői ütemterv

|  |  |
| --- | --- |
| Határidő | Feladat |
| 2024.12.19 | Adatlap beadása, vizsgaremek témájának eldöntése |
| 2025.01.27 | Végleges sablon bemutatása |
| 2025.02.21 | Adatbázis terv bemutatása |
| 2025.02.24 | Működőképes Authentikáció |
| 2025.03.24 | Weboldal és asztali alkalmazás főbb funckióinak kialakítása |
| 2025.04.10 | Stílus csiszolgatása, végső simítások |
| 2025.04.16 | Végső leadási határidő |

* 1. Fejlesztés menete
     1. Piackutatás

A projekt megtervezése előtt számos hasonló weboldalt kutattunk át, mind magyart, mind nemzetközit. Kutatómunkánk során gyűjtöttük ki hogy milyen fő funkcionalitások kellenek egy ilyen oldal működéséhez, illetve hogy mi ezeken hogyan tudnánk fejleszteni. Ennek a végeredménye az lett, hogy weboldalunk hasonlít a már piacon lévőkre, ezzel megadva a felhasználónak egy már meglévő komfortot, viszont, különbözik annyira, hogy ne vegyüljön el a már meglévő piac rengetegében.

* + 1. Adatbázis megtervezése

Az adatbázisunk MySQL nyelven, MariaDB-vel működik.

Az adatbázis 12 táblával rendelkezik, viszont számos egyéb adattag kiszámítható ezekből az adatokból. Törekedtünk arra, hogy az adatbázis megfeleljen a harmadik normálformának, illetve célunk volt a redundancia (azaz a szükség nélküli adatismétlődés) elkerülése. A következő oldalakon a táblák felépítése található:

1. ’User’ tábla

Ez a tábla tárolja a regisztrált felhasználó minden adatát

* user\_id: A felhasználó egyedi azonosítója, egész szám, elsődleges kulcs
* username: A felhasználó egyedi felhasználó neve, maximum 32 karakter hosszú.
* lastName: A felhasználó vezetékneve, maximum 50 karakter hosszú.
* firstName: A felhasználó keresztneve, és amennyiben van akkor harmadik neve, maximum 50 karakter hosszú.
* birthDate: A felhasználó születési dátuma.
* phonenumber: A felhasználó telefonszáma.
* email: A felhasználó saját email címe.
* password: A felhasználó jelszavának titkosított változata.
* created\_at: A pontos időpontja, hogy egy felhasználó mikor regisztrált az oldalra.
* updated\_at: A pontos időpontja, hogy egy felhasználó mikor módosította valamelyik adatát. (például email vagy telefonszám)
* active: logikai változó, mutatja, hogy egy felhasználó fiókja törlésre került-e már.
* profilePic: A felhasználó profilképének azonosítója, egész szám.

1. ’loyaltyrank’ tábla

Ez a tábla tárolja a felhasználó hűségprogram adatait.

* rank\_id: A tábla egyedi azonosítója, elsődleges kulcs.
* rank: a hűségszint elnevezése
* discount: tört szám, amely megadja, hogy az adott rang hány százalékos kedvezményt kap a vásárlásaira.
* minPoint: A minimális pontszám, ami szükséges, hogy a felhasználó erre a szintre lépjen
* perks: A hűségszint nyújtotta előnyök

1. ’loyalty’ tábla

Ez a tábla tárolja a felhasználó hűségprogram adatait.

* loyalty\_id: A tábla egyedi azonosítója, elsődleges kulcs.
* user\_id: Idegen kulcs a ’user’ táblához.
* rank\_id: Idegen kulcs a ’loyaltyrank’ táblához.
* points: A felhasználó hűségpontjai, egész szám.
* updated\_at: A hűségpontok frissülésének pontos dátuma.

1. ’city’ tábla

Ez a tábla tárolja el azokat a városokat, ahol az oldal szállodákat üzemeltet.

* city\_id: A város egyedi azonosítója, elsődleges kulcs
* cityName: A város neve, maximum 100 karakter hosszú
* country: Az ország, amelyben a város elhelyezkedik, maximum 100 karakter hosszú
* description: A város hosszú leírása, amelyet kiírunk az oldalra.
* description\_short: A város leírásának rövid változata.

1. ’hotel’ tábla

Ez a tábla tárolja el az oldalon szereplő szállodák adatait

* hotel\_id: A szálloda egyedi azonosítója, elsődleges kulcs
* city\_id: Idegen a kulcs a ’city’ táblához, mutatja, hogy melyik városban található a város.
* hotelName: A szálloda elnevezése, maximum 150 karakter hosszú
* address: A szálloda pontos címe, szöveges.
* phoneNumber: A hotel recepciójának telefonszáma
* email: A hotel saját email címe, szöveges.
* description: A hotel leírása, amelyet betöltünk az oldalra, szöveges.

1. ’reviews’ tábla

Ez a tábla tárolja el a felhasználók által írt értékeléseket.

* review\_id: Az értékelés egyedi azonosítója, elsődleges kulcs.
* user\_id: Idegen kulcs a ’user’ táblához
* hotel\_id: Idegen kulcs a ’hotel’ táblához
* rating: Az értékelés csillagértéke, egész szám.
* reviewText: Az értékelés szövegi része, szöveges.
* created\_at: Az értékelés létrehozásának/módosításának pontos időpontja.
* active: Logikai, mutatja, hogy az értékelés törlésre került-e
* edited: Logikai, mutatja, hogy az értékelés szerkesztve lett-e.

1. ’employee’ tábla

Ebben a táblában vannak eltárolva azok a felhasználók, akik valamely szállodában dolgoznak.

* hotel\_id: Idegen kulcs a ’hotel’ táblához
* user\_id: Idegen kulcs a ’user’ táblához
* userType: Mutatja, hogy a felhasználó milyen pozícióban van a szállodában, értékei „employee”, azaz alkalmazott, „manager” vagy pedig „owner” aki a szálloda tulajdonos.

1. ’servicecategory’ tábla

Ebben a táblában vannak eltárolva a különböző szolgáltatásai nevei.

* serviceCategory\_id: A szolgáltatás egyéni azonosítója, elsődleges kulcs
* serviceName: A szolgáltatás megnevezése.

1. ’service’ tábla

Ebben a táblában vannak eltárolva a szálloda szolgáltatásainak pontos adatai

* service\_id: A szolgáltatás egyéni azonosítója, elsődleges kulcs.
* hotel\_id: Idegen kulcs a ’hotel’ táblához
* category\_id: Idegen kulcs a servicecategory táblához
* price: Egész szám, a szolgáltatás igénybevételének ára
* available: Logikai, mutatja, hogy a szolgáltatás az aktuális pillanatban elérhető-e
* allYear: Logikai, mutatja, hogy a szolgáltatás egész évben elérhető-e
* startDate: Amennyiben nem egész évben elérhető, a szolgáltatás kezdeti dátuma
* endDate: Amennyiben nem egész évben elérhető, a szolgáltatás végső dátuma
* openTime: Amennyiben a szolgáltatás nem érhető el egész nap, a nyitó ideje.
* closeTime: Amennyiben a szolgáltatás nem érhető el egész nap, a záró ideje.

1. ’room’ tábla

Ebben a táblában vannak eltárolva a szállodák különböző szobáinak az adatai

* room\_id: A szoba egyéni azonosítója, elsődleges kulcs
* hotel\_id: Idegen kulcs a ’hotel’ táblára.
* roomNumber: A szoba megnevezése, maximum 25 karakter hosszú
* floor: Az emelet, amelyen a szoba elhelyezkedik, egész szám
* capacity: A szoba férőhelye, egész szám.
* pricepernight: A szoba éjszakánkénti foglásának az ára
* available: logikai, mutatja, hogy a jelenlegi pillanatban elérhető-e a szoba

1. ’booking’ tábla

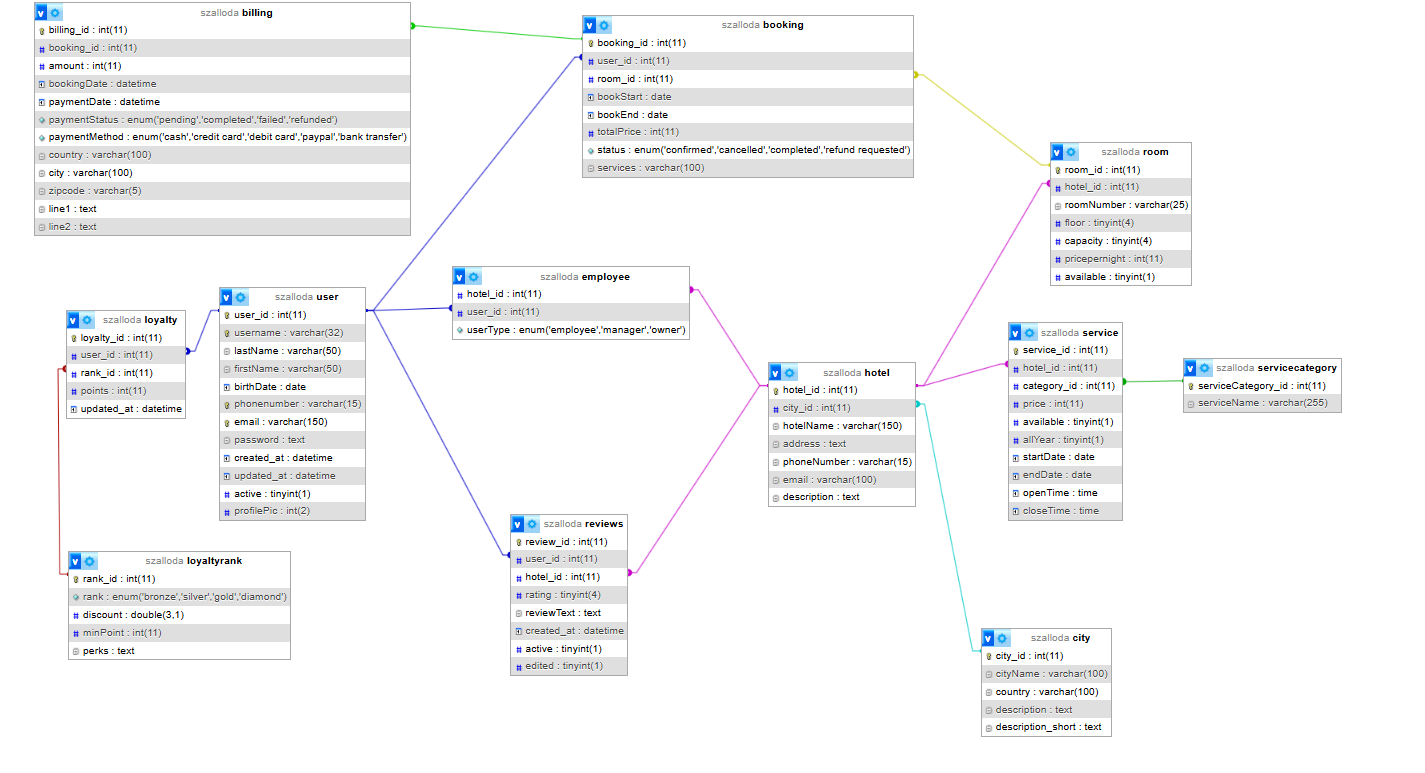
Ez a tábla tartalmazza az oldalon történt foglalásokat

* booking\_id: A foglalás egyéni azonosítója, elsődleges kulcs.
* user\_id: Idegen kulcs a ’user’ táblához
* hotel\_id: Idegen kulcs a ’hotel’ táblához
* bookStart: A foglalás kezdetét jelölő dátum
* bookEnd: A foglalás végét jelölő dátum
* totalPrice: A foglalásért fizetendő összeg forintban, egész szám
* status: A foglalás jelenlegi állapotát jelzi, értékei ’confirmed’ azaz megerősített, ’cancelled’ azaz lemondott, ’completed’ azaz befejezett és ’refund requested’ azaz visszatérést igényelt.
* services: A foglalás mellé nyújtott szolgáltatások listája.

1. ’billing’ tábla

Ez a tábla tartalmazza a foglalás számlázásával kapcsolatos adatait.

* billing\_id: A számla egyéni azonosítója, elsődleges kulcs.
* booking\_id: idegen kulcs a ’booking’ táblához.
* amount: A fizetett összeg forintban, egész szám.
* bookingDate: A foglalás pontos időpontja.
* paymentDate. A fizetés pontos időpontja.
* paymentStatus: A fizetés jelenlegi állapotát jelzi, értékei ’pending’ azaz függőben lévő, ’completed’, azaz befejezett, ’failed’ azaz sikertelen volt a fizetés és ’refunded’, azaz visszatérített
* paymentMethod: A fizetés módszerét jelzi, értékei: ’cash’ azaz készpénz, ’credit card’ azaz, hitelkártya, ’debit card’ azaz betéti kártya, ’paypal’ illetve’bank transfer’, azaz banki átutalás.
* country: A számlázott felhasználó országa.
* city: A számlázott felhasználó városa.
* zipcode: A város irányítószáma.
* line1: A lakcím első sora, az utca és házszámot tartalmazza
* line2: A lakcím második sora, a cím egyéb adatait tartalmazza mint például az emeletet és az ajtószámot



ábra 1: EER diagram a 'szalloda' adatbázisról

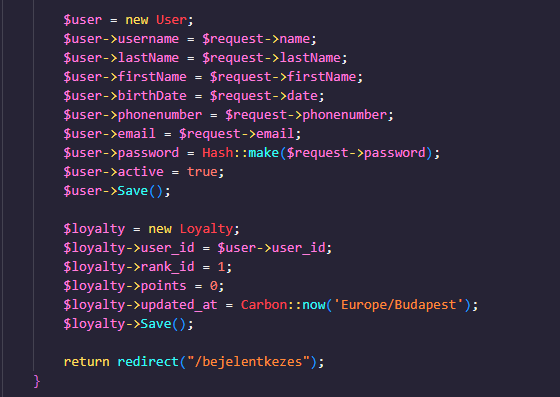
* + 1. Főbb algoritmusok
       1. Weboldal

A weboldal backendje Laravelt használ, hiszen a csapat tagjainak ez volt a legkényelmesebb fejlesztői környezet, továbbá a beépített funkciók segítségével képesek voltunk többet megvalósítani, mint amit eredetileg terveztük. Erre az egyik legjobb példa a regisztráció, hiszen itt a laravel segítségével könnyedén tudtuk validálni hogy a felhasználó megfelelően töltötte-e ki az adatait, illetve egyértelműen tudtuk jelezni, hogy mi a probléma a bevitt adatokkal.



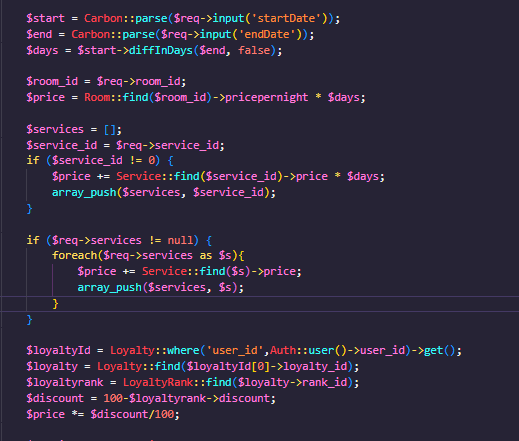
ábra 2: a regisztráció függvény validáló része

Miután meggyőződtünk róla hogy a felhasználó helyesen vitte be az adatait, lefut a függvény további része amely létrehozza a felhasználó fiókját, illetve a hozzátartozó hűségprogram adatokat. A felhasználó jelszava titkosítva van eltárolva, így amennyiben illetéktelen hozzáférés történik az adatbázishoz, nem fognak tudni hozzáférni a felhasználók jelszavához.



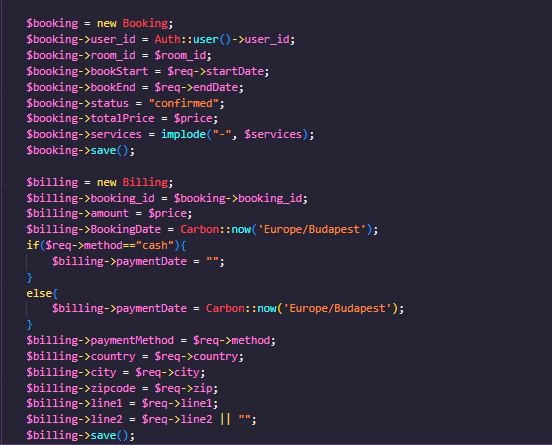
ábra 3: a regisztráció függvény adatbázisba illesztő része

Az oldal leglényegesebb funkciója a szobafoglalás, így természetesen ez a függvény az egyik leghosszabb, így ezt szeretném több részre felbontani. Elsősorban a függvény ellenőrzi, hogy a felhasználó bevitte-e a megfelelő adatokat. Majd ezután egyesével kiszámolja a költségeket. Először kiszámolja, hogy mennyi a szoba ára az ott töltött éjszakákra, majd ezután hozzáadja a kiválasztott szolgáltatások árát. A végső árra ezután alkalmazza a felhasználó hűségprogramjából származó akciót is.



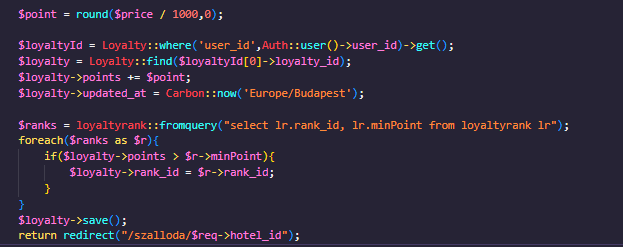
ábra 4: A foglalás függvény árat számoló része

Ezek után a függvény feltölti az adatbázisba a felhasználó által bevitt adatokat mind a foglalás, mind a számlázás táblába.



ábra 5: A foglalás függvény adatbázisba író része

Végezetül, a függvény frissíti a felhasználó hűségpontjait, illetve le ellenőrzi, hogy a felhasználó elért-e egy új hűségszintet, és amennyiben igen, annak megfelelően frissíti a felhasználó adatait.



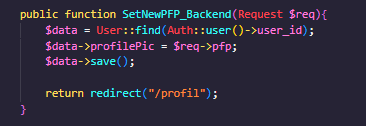
ábra 6: A foglalás függvény hűségpontszámító része

A felhasználó tudja az oldalon módosítani saját adatait, amennyiben azok változtak regisztráció óta. A felhasználó tudja módosítani a felhasználónevét, email címét vagy akár a polgári nevét is. A függvény figyel az email cím helyességére, a felhasználónév foglaltságára, illetve arra hogy a felhasználó beírta-e a vezeték és keresztnevét is.



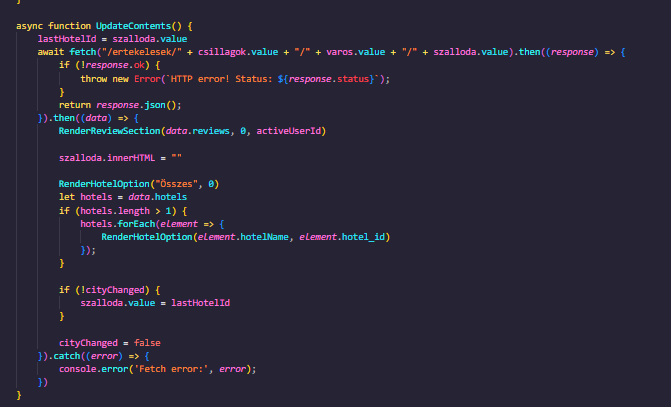
ábra 7: A felhasználói adatokat módosító függvény

Ezentúl a felhasználó képes módosítani a profilképét, a weboldal szereplő profilképek közül bármelyiket képes kiválasztani.



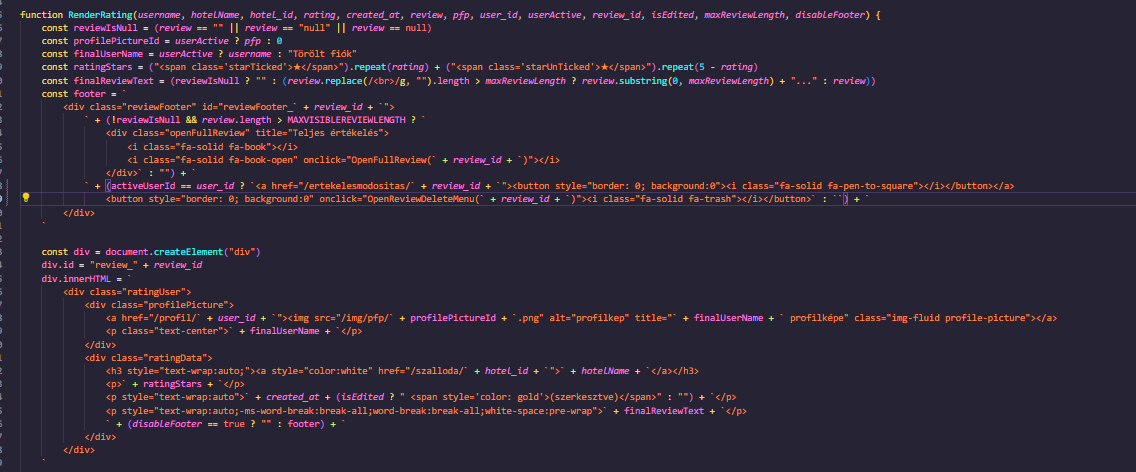
ábra 8: A felhasználó profilképét módosító függvény

Sajnos, mivel a Laravel nem képes dinamikusan frissíteni az oldalt bizonyos változások bekövetkezésekor, ezért a JavaScript is fontos szerepet játszik a weboldal működésében. A JavaScript függvényeink kényelmet biztosítanak a felhasználónak, hiszen így nem kell folyamatosan újra frissíteni az oldalt, amikor valamilyen változás történik annak tartalmában. Erre a legjobb példa az értékelések szűrése, amely a szűrésnek megfelelő adatok alapján tölti be a megfelelő értékeléseket, amiket az adatbázisból kap vissza.



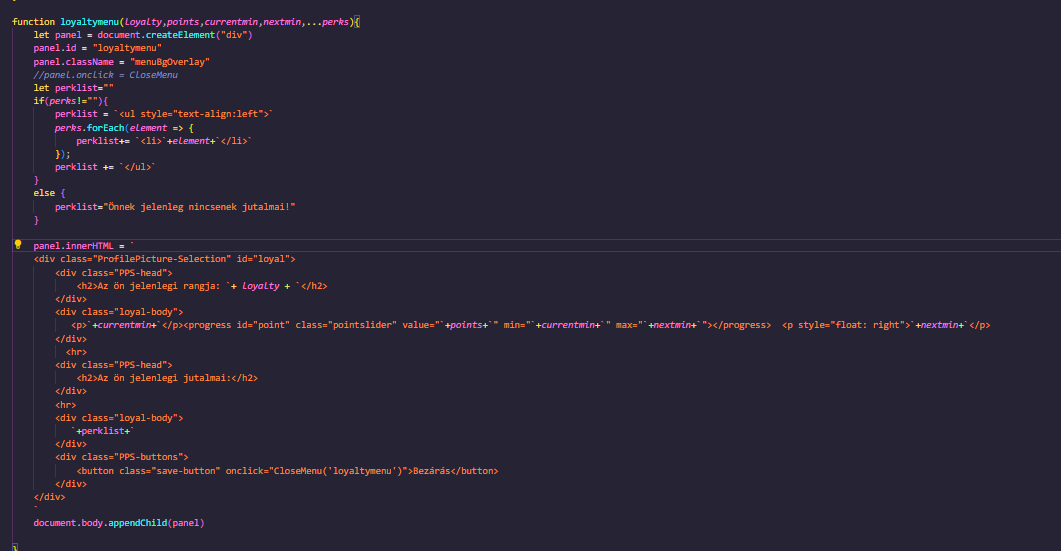
ábra 9: Az értékelés szűrést végző függvény

A szűrés után minden egyes értékelés, ami megfelelt a követelményeknek, kap egy saját szekciót, amelynek létrehozásáért egy másik függvény felel.



ábra 10: Az értékelés szekciók létrehozásáért felelős függvény

A JavaScriptet viszont nem csak oldalfrissítésre használjuk, hanem arra is, hogy megjelenítsünk a felhasználónak új menüpontokat kattintásra, mint például a hűségszint vagy a profilkép változtatás menüje. Ezek a függvények hozzák létre a felhasználó által látott menüt, minden működésével együtt



ábra 11: A hűségmenü ablakát létrehozó függvény

* + - 1. Asztali alkalmazás

Valamikor folytasd geci mert kurvára elfogod felejteni

* 1. Tesztelés

A weboldal fejlesztése alatt dinamikusan teszteltünk, ami azt takarja, hogy a kód közvetlen megírása után teszteltük a működését is. Az oldalt manuálisan volt tesztelve, azaz saját kézzel ellenőriztük az oldal megfelelő működését.

A legtöbb tesztelést a foglalási oldal igényelte, hiszen itt vannak a legtöbb adatbeviteli felület, illetve a legbonyolultabb függvények is. Eleinte a szobák elérhetőségi szűrési kizárólag azt vizsgálta, hogy a foglalás kezdeti és vég dátuma alatt foglalt-e a szoba, így amennyiben valaki úgy foglalt, hogy a foglalása közepén valaki más már kivette-e a szobát, az oldal nem jelzett volna hibát és engedte volna a kettős foglalást. Szerencsére a folytonos tesztelés miatt erre nagyon hamar rájöttünk

További nehézséget okozott az adatbázisba való helyes adat bevitel is, hiszen sokszor nem egyenesen azt visszük be az adatbázisba, amit a felhasználó begépel. Erre egy tökéletes példa a profil oldalon használt „polgári név” mező, hiszen itt a felhasználó csak beírja a teljes nevét, viszont az adatbázisban a vezeték és keresztnevek külön vannak tárolva. Sokáig nem találtunk arra megoldást, hogy hogyan tudnánk leellenőrizni azt, hogy a felhasználó legalább 2 nevet beleírt-e a mezőbe, hiszen amennyiben ez nem teljesül az adatbázis hibásan lenne feltöltve, viszont szerencsére sikerült megoldani ezt a problémát.

Sajnos az asztali alkalmazás fejlesztése se ment zökkenő mentesen. Szerettük volna megcsinálni, hogy minden kezelendő adat generáljon saját magának egy panelt, amin rajta vannak a különböző műveletek. Rengeteg munka volt azzal, hogy ezek működően, egymás alá generálódjanak, hiszen rengetegszer kilógtak a képből, egymásra generálódtak vagy a gombok nem működtek rendeltetésszerűen. Szerencsére minden hibát ezzel kapcsolatban sikerül kiköszörülni és minden kellően működik.

A folytonos tesztelésnek köszönhetően sikerült létrehoznunk egy olyan oldalt, ami javarészt hibamentes. Természetesen előfordulhat, hogy valamelyik apró zugban egy apró hiba fellelhető, viszont a csapat minden tőle telhetőt megtelt, hogy egy közel tökéletes oldalt hozzon össze.

1. Felhasználói dokumentáció, felhasználói kézikönyv

A weboldal két féle módban tekinthető meg. vendégként, ami körbetekintés céljából hasznos, hiszen a felhasználó megtudja nézni a kínálatunkat regisztráció előtt, illetve felhasználóként, akiknek a teljes oldal elérhetővé válik.

* 1. Telepítési útmutató.

Először is, bizonyosodjon meg róla, hogy az eszközén rajta van a XAMPP Control Panel ,illetve a Visual Studio. Indítsa el a XAMPP Control Panelt, majd indítsa el az „Apache” és „MySQL” modulokat. Nyomja meg a „MySQL” modul „admin” feliratú gombját, és ellenőrizze hogy létezik-e a „szalloda” című adatbázis, benne a megfelelő adatokkal. amennyiben nem, hozza létre az adatbázist és pótolja tartalmát az „sql” mappában szereplő „szalloda.sql” file segítségével.