

Debreceni Szakképzési Centrum
Baross Gábor Középiskolája és Kollégiuma
Debrecen, 4030
Budai Ézsaiás u. 8/A.

A SZAKKÉPESÍTÉS AZONOSÍTÓ SZÁMA:54 213 05
SZAKKÉPESÍTÉS MEGNEVEZÉSE: SZOFTVERFEJLESZTŐ

Záródolgozat címe:

Brutal Barbers

Záródolgozat készítője:

Balogh Benedek

Konzulens: Kovács István Zoltán

Konzultációs időpontok:

Debrecen, 2023.

Tartalomjegyzék

Bevezetés.....	4
Témaválasztás indoka.....	4
A program jelentősége.....	4
A szakdolgozat célja.....	4
A program létrehozásának elképzelése.....	4
Felhasznált programok	5
Felhasználói dokumentáció	7
Az oldal elérése	7
XAMPP.....	7
Visual Studio Code	9
Hardver és szoftver igény	12
Minimális rendszerkövetelmények	12
Ajánlott rendszerkövetelmények	13
Lehetőségek a regisztráció után a weboldalon:	15
A regisztráció és bejelentkezés ismertetése.....	15
Regisztráció.....	16
Belépés.....	17
Belépés az oldalra adminként	18
A navigációs sáv felhasználók részére	20
A főoldal és tartalma.....	21
Az időpontfoglalás menüpont.....	22
A referenciáink menüpont	22
Fejlesztői dokumentáció.....	24
Fejlesztői környezet.....	24
Algoritmusok az oldalon	25

Regisztráció.....	25
Belépés.....	27
Admin oldal*	30
Időpont feltöltése és lekérdezése	34
Az adatbázis különböző táblái és azok mezői	38
A felhasználók tábla.....	38
Az időpontok tábla	39
A képek tábla	39
Tesztelői dokumentáció.....	40
Programozásban nem jártas felhasználó véleménye	40
Az oldal visszajelzései.....	41
Hibaüzenetek	41
Általános üzenetek.....	43
Összefoglalás.....	46
Az oldal dizájn része.....	46
Célok, amelyeket nem sikerült teljesíteni.....	46
Fejlesztési lehetőségek	47
Irodalomjegyzék.....	48
Ábrajegyzék	49

Bevezetés

Témaválasztás indoka

A weboldal témáját egy közeli időben befutott Berettyóújfaluban található férfifodrászat ihltette, melynek munkásságát már a megnyitás előtt értékelhettem, és vehettem igénybe szolgáltatásait. Az évről-évre nagyobbodó ügyfelek száma a boltban már elérte azt a szintet, hogy nem könnyű egy hétköznapi naptárban tárolniuk az időpontokat, így ezzel a weboldallal szeretném megkönnyíteni a munkásságukat.

A program jelentősége

A fodrászat dolgozóinak akár egy hétre előre, de legfontosabban egy nap minden érkező vendég időpontjait kell egy kis egyszerű naptárból figyelembe tartaniuk. Így ezáltal a program elősegíti az időpontok átláthatóságát. Ezek mellett az oldalon található egy oldalt ahol a fodrászat munkásságait követhetik a vendégek.

A szakdolgozat célja

A szakdolgozatom célja, hogy a Brutal Barbers fodrászatban dolgozó fodrászok mindennapját könnyíthessem meg az által, hogy egy helyen követhetik az ügyfelek saját maguk által rögzített időpontokat, és a fodrászok tölthessék fel saját maguk munkásságait.

A program létrehozásának elképzelése

Adatbázis megtervezése és létrehozása, majd ezt követően felületek elkészítése HTML-ben minimális CSS beállításokkal. Az első az adatfeltöltéses felületek elkészítése, a beléptetés, majd ezt követően a regisztráció és az ehhez szükséges feldolgozó programok. Az adatfeltöltéses felületeken belül érdemes lehet inkább a regisztrációval kezdeni, hogy az embernek teszteléskor ne azzal kelljen kezdeni, hogy hozzon létre adatbázison belül magának egy teszteléshez szükséges profilt. Amikor egy weboldalban regisztrációt várunk felhasználóinktól, érdemes azt eleinte rövidre venni, mert az emberek nagy része nem kedveli a kötelező regisztrációt. Ezután következne az időpontfoglalás rész elkészítése, majd az időpontok kiírásához szükséges programrész. A legutoljára hagynám az adminisztrátori felületet, benne az összes felhasználó által foglalt időpontok kiírásához szükséges kóddal.

Felhasznált programok



1. ábra Windows 10 logó

A Windows 10 operációs rendszert használtam szakdolgozatom elkészítésénél, mert ez az operációs rendszer kompatibilis a később felsorolt programokkal.



2. ábra MySQL logó

A MySQL többfelhasználós, többszálú, SQL-alapú relációs adatbázis-kezelő szerveret használtam a weboldalhoz kapcsolódó adatok tárolásához.



3. ábra Google Chrome logó

Google Chrome: A program teszteléséhez és böngészéshez egyaránt, Mozilla Firefox-ot használtam. Ennek a böngészőnek a segítségével tudtam kapcsolódni az internetre és tesztelni vagy formázni az eddig elkészült weboldalt.



4. ábra Total Commander logó

A Total Commander shareware fájlkezelő programot használtam a szakdolgozatom fájlkezelés részéhez. Ez a program könnyen átláthatóvá teszi az állományokat, így megkönnyítve a fájlok megkülönböztetését egymástól



5. ábra Visual Studio Code

A Visual Studio Code egy olyan ingyenes kódszerkesztő program melyet a Microsoft fejleszt. Azért is ezt választottam mert nem csak személyre szabható a környezet amiben dolgozni szeretnénk, hanem akár olyan kiegészítőket telepíthetünk melyek segítenek, könnyebbé teszik a programozást.



6. ábra NodeJS logó

A Node.js egy szoftverrendszer, melyet skálázható internetes alkalmazások, mégpedig webszerverek készítésére hoztak létre.



7. ábra Bootstrap logó

A bootstrap egy olyan nyíltforráskódú rendszer amely elősegíti az oldal reszponzivitását. A Bootstrap számos előre elkészített felhasználói felület "darabkát" tartalmaz, úgymint navigációs menük, táblázatok, gombok, űrlapok, satöbbi ezek az úgynevezett komponensek.

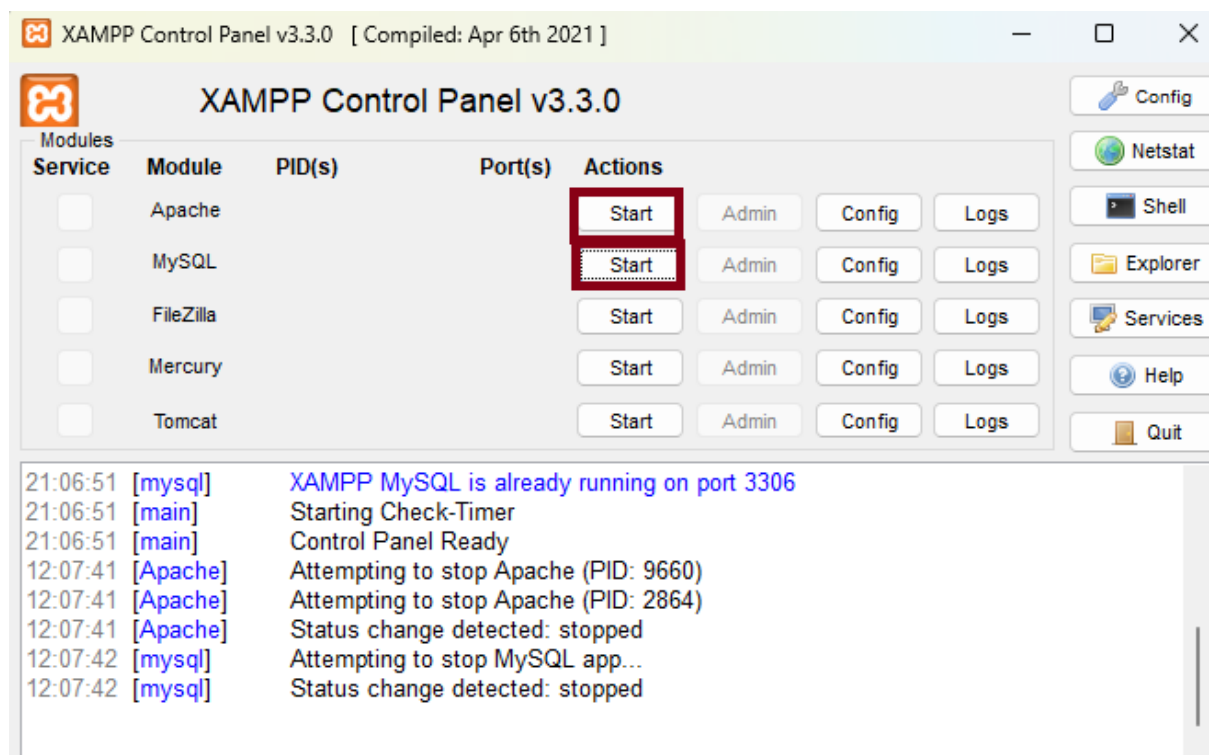
Felhasználói dokumentáció

Az oldal elérése

Ehhez rendelkezünk kell a Visual Studio Code programmal, a xampp programmal és egy böngészővel.

XAMPP

Ha feltelepítettük a xampp programot akkor indítsuk el a „*XAMPP Control Panel*” alkalmazást és kattintsunk a „*Apache*”, és „*MySQL*” mellett látható start gombokra



8. ábra XAMPP control panel

Ezt követően a böngészőnkbe keressük meg a <http://localhost/phpmyadmin/> címet. Ha ezzel megvagyunk nincs más hátra hogy a felső sorban kattintsunk az „Importálás gombra” és tallózzuk a „*Fodrászat.sql*” fájlt, majd az oldal alján kattintsunk az importálás gombra.

Az importálandó fájl:

A fájl lehet tömörített (gzip, bzip2, zip) vagy tömörítetlen.
A compressed file's name must end in .[format].[compression]. Example: .sql.zip

Számítógép tallózása: (Max: 40MB)

Fájl kiválasztása fodraszat.sql

Továbbá behúzhat és beejthet egy fájlt bármely oldalra.

A fájl karakterkészlete:

utf-8

Részleges importálás:

☒ Allow the interruption of an import in case the script detects it is close to the PHP timeout limit.
This might be a good way to import large files, however it can break transactions.

Ennyi számú lekérdezés (SQL esetén) kihagyása az elsőől kezdve:

0

Other options

☒ Időben kulcsok ellenőrzésének engedélyezése

Formátum

SQL

Formátum specifikus opciók:

SQL kompatibilitási mód:

NONE

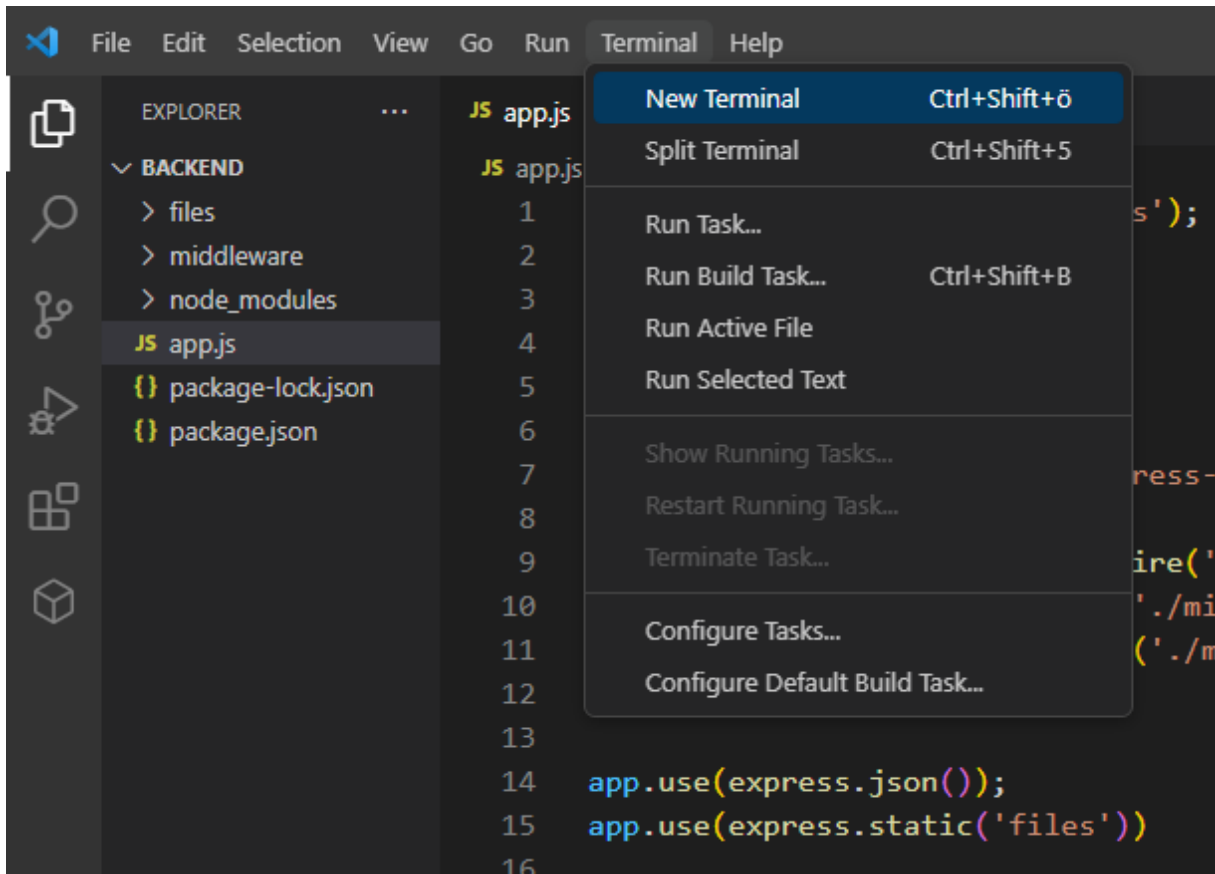
☒ Ne használja az AUTO_INCREMENT-et nulla értékekhez

Importálás

9. ábra PhPmyadmin importálás

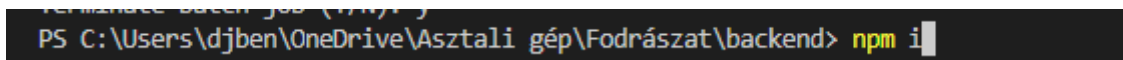
Visual Studio Code

Indítsuk el a Visual Studio Code-ot két különböző ablakban és nyissuk meg a „Fodrászat.rar” fájlban található „Frontend”, „Backend” mappát. Ezt követően a felső menüpontok közül válasszuk ki a „Terminal-New Terminal” lehetőséget abban az ablakban amiben a backend mappa van megnyitva.



10. ábra Terminal megnyitás

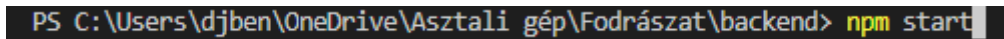
A gomb megnyomása után a képernyő alján megjelenik a Terminál ahova a következőket kell beírunk: `npm i` [ENTER]. Ez telepíteni fogja a felhasznált hozzáadott csomagokat.



```
PS C:\Users\djben\OneDrive\Asztali gép\Fodrászat\backend> npm i
```

11. ábra csomagok feltelepítése

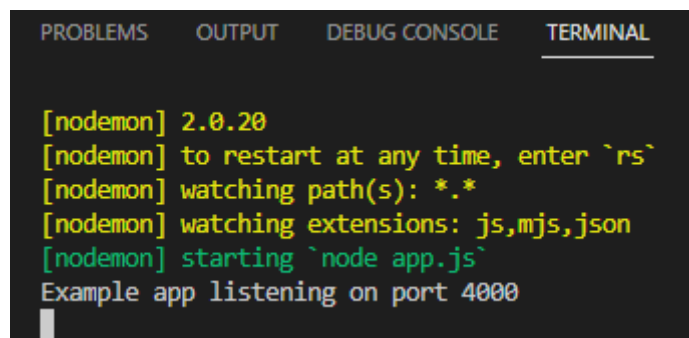
Ezt követően írjuk be, hogy `npm start` [ENTER]. Ez elfogja indítani a program „háttérét”.



```
PS C:\Users\djben\OneDrive\Asztali gép\Fodrászat\backend> npm start
```

12. ábra "háttérprogram" elindítása

Ha sikeresen elindul akkor a következőket kell látnunk a terminálban:

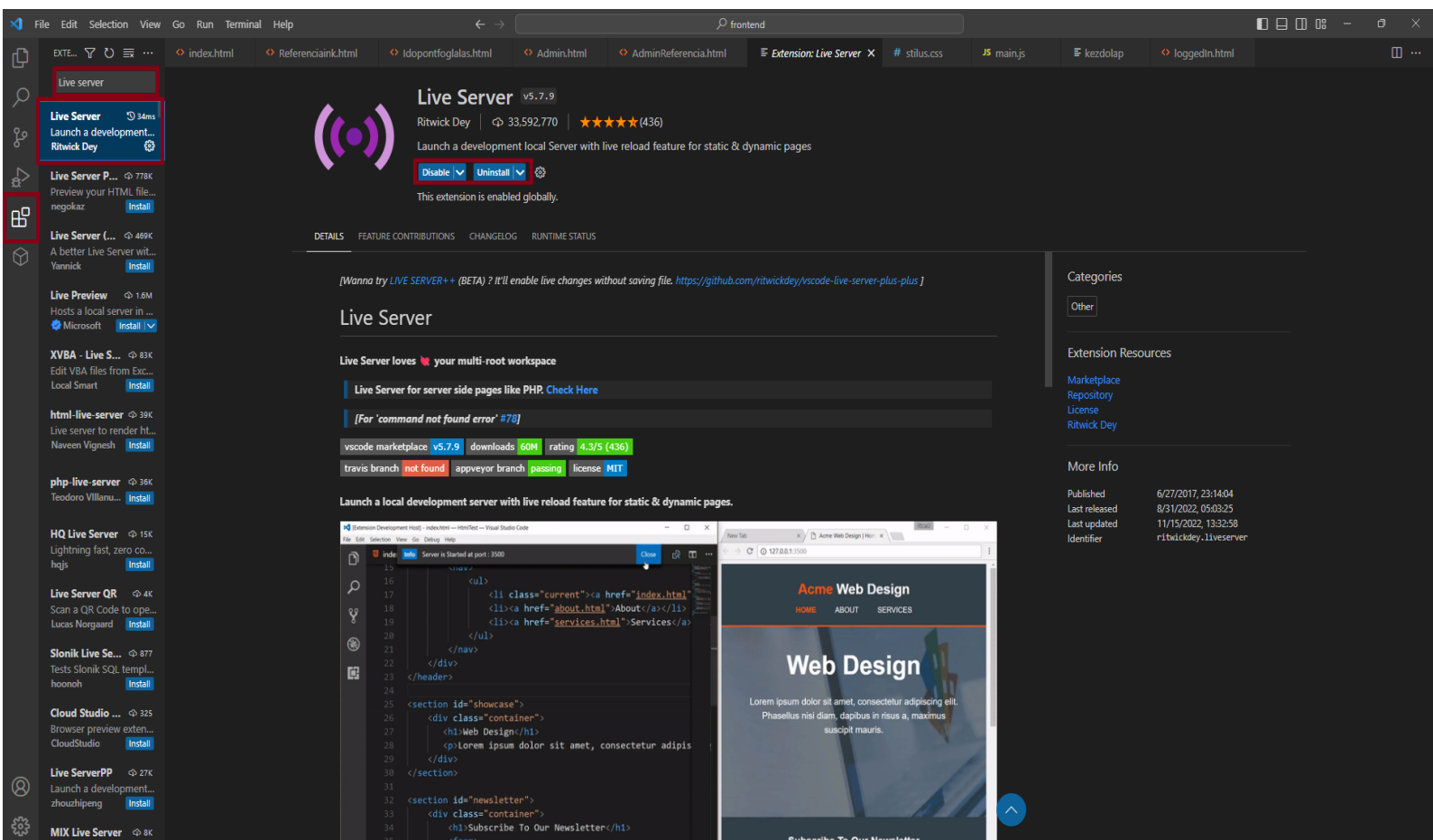


```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node app.js`
Example app listening on port 4000
```

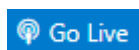
13. ábra Sikeres futtatás

Ezután következhet az az ablak amiben a frontend mappa van megnyitva. Itt hozzáadunk egy kiegészítőt. Ehhez a bal oldalon keressük meg és kattintsunk arra az ikonra amin 4 négyzet látható. A mellette megjelenő keresőbe írjuk be, hogy Live Server és az első találatra kattintsunk rá. Megfog jelenni egy oldal ahol az „Install” gombra kattintva elindul a kiegészítő telepítése.



14. ábra Live Server telepítése

Ezentúl a jobb alsó sarokban megjelent egy „Go Live” gomb amire kattintsunk is rá és már a weboldalon találhatjuk magunkat.



15. ábra Go Live gomb

Hardver és szoftver igény

Mivel ez egy webes felület ezért igényel egy eszközt, amivel lehet internetezni, ez lehet laptop vagy asztali számítógép. Továbbá szükséges egy szerver, amire a weboldal kerül, ezen felül szükséges internet hozzáférés, valamint a következő böngészők egyike: Google Chrome, Firefox, Internet Explorer.

Minimális rendszerkövetelmények

Windows operációs rendszer (32-bit és 64 bit):

- CPU: Pentium 4 vagy újabb processzor, ami támogatja a SSE2 technológiát
- RAM: 512MB (32 bites rendszernél) / 2GB (64 bites rendszernél)
- HDD: 128gb
- VGA: Integrált Videó kártya 32MB RAM
- Operációs Rendszer: MS Windows XP SP2 vagy újabb
- Internet

Macintosh operációs rendszer:

- Macintosh számítógép egy Intel x86 processzorral vagy jobb
- RAM: 512 MB • HDD: 200 MB
- VGA: Integrált Videó kártya 32MB RAM
- Monitor: 800x600-as felbontás vagy nagyobb
- Operációs Rendszer: macOS 10.9, macOS 10.10, macOS 10.11, macOS 10.12, macOS 10.13
- Internet

Linux operációs rendszer:

- CPU: Pentium 4 vagy újabb processzor ami támogatja a SSE2 technológiát
- RAM: 512MB (32 bites rendszernél) / 2GB (64 bites rendszernél)
- HDD: 200 MB
- VGA: Integrált Videó kártya 32MB RAM
- Monitor: 800x600-as felbontás vagy nagyobb
- Operációs Rendszer: Ubuntu, Linux Mint, Fedora, Debian vagy hasonló linux alapú
- Internet

Ajánlott rendszerkövetelmények

Windows operációs rendszer(32-bit és 64 bit):

- CPU: Intel Core i5- @ 2.60GHz vagy jobb
- RAM: G.SKILL 16GB DDR4 3333Mhz vagy jobb
- HDD: WD Red Pro 500GB 7200 rpm 256 MB SATA 3 WD101KFBX vagy jobb
- VGA: NVIDIA GeForce GTX 1050 , 4GB GDDR5X vagy jobb
- Monitor: 3840 x 2160 (4K UHD) vagy nagyobb
- Operációs Rendszer: Windows 10 vagy újabb
- Internet

Macintosh operációs rendszer:

- CPU: Intel Xeon E5 25 MB L3 gyorsítótárral, 3,9 GHz vagy újabb
- RAM: 16 GB (4 x 4 GB) 1866 MHz-es DDR3 ECC memória vagy több
- HDD: 512 GB-os, PCIe-alapú SSD
- VGA: AMD FirePro D700 6 GB GDDR5 vagy több
- Monitor: 3840 x 2160 (4K UHD) vagy nagyobb
- Operációs Rendszer: macOS 10.9, macOS 10.10, macOS 10.11, macOS 10.12, macOS 10.13
- Internet

Linux operációs rendszer:

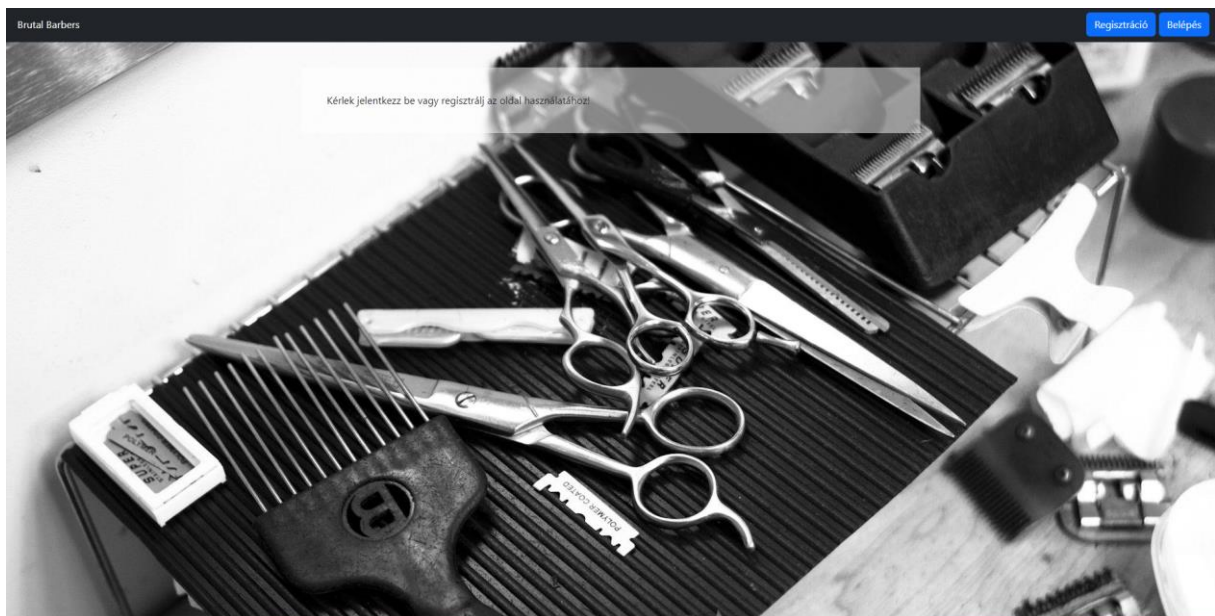
- CPU: Intel Core i5- @ 2.60GHz vagy jobb
- RAM: G.SKILL 16GB DDR4 3333Mhz vagy jobb
- HDD: WD Red Pro 500GB 7200 rpm 256 MB SATA 3 WD101KFBX vagy jobb
- VGA: NVIDIA GeForce GTX 1050 , 4GB GDDR5X vagy jobb
- Monitor: 3840 x 2160 (4K UHD) vagy nagyobb
- Operációs Rendszer: Ubuntu, Linux Mint, Fedora, Debian vagy hasonló linux alapú
- Internet

Lehetőségek a regisztráció után a weboldalon:

- Főoldal megnyitása
- Időpont lefoglalása
- Referencia munkák megtekintése
- Instagramm oldal megnyitása

A regisztráció és bejelentkezés ismertetése

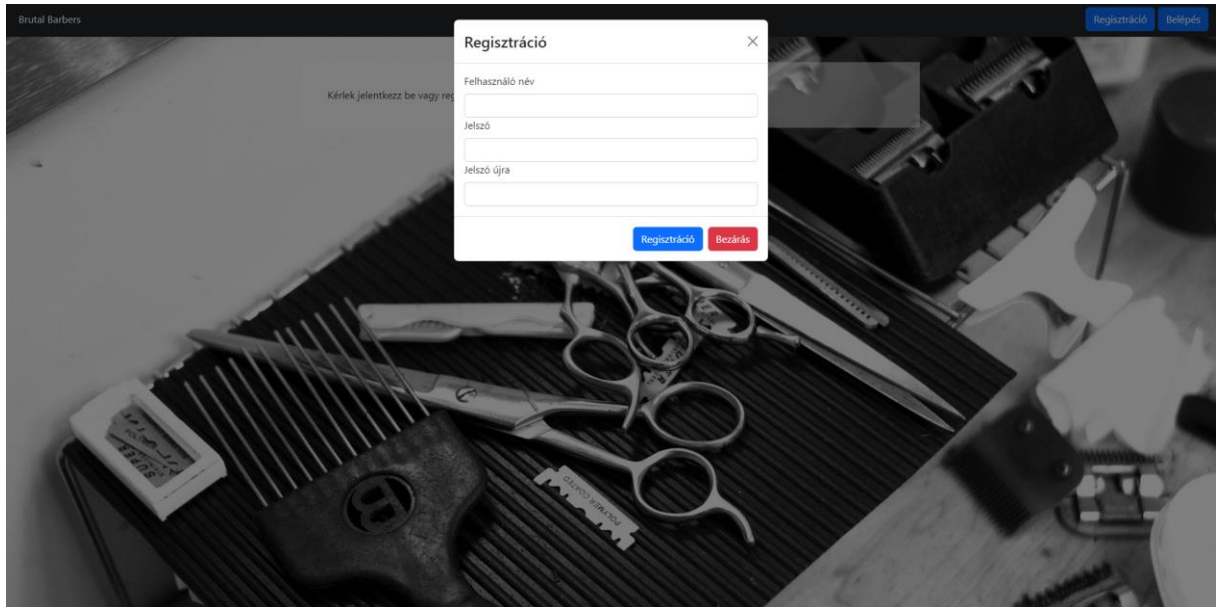
A weboldalon a legelsőnek, amivel szembe találkozunk, az a bejelentkező/regisztrációs oldal. Ennek fontossága az oldalra belépő rosszakarók kiszűrése.



16. ábra A weboldalra érkezéskor

Regisztráció

A jobb felső sarokban találhatjuk azt a két gombot, amivel ha új fiókot szeretnénk regisztrálni, akkor azt a regisztráció gombbal tudjuk megtenni.

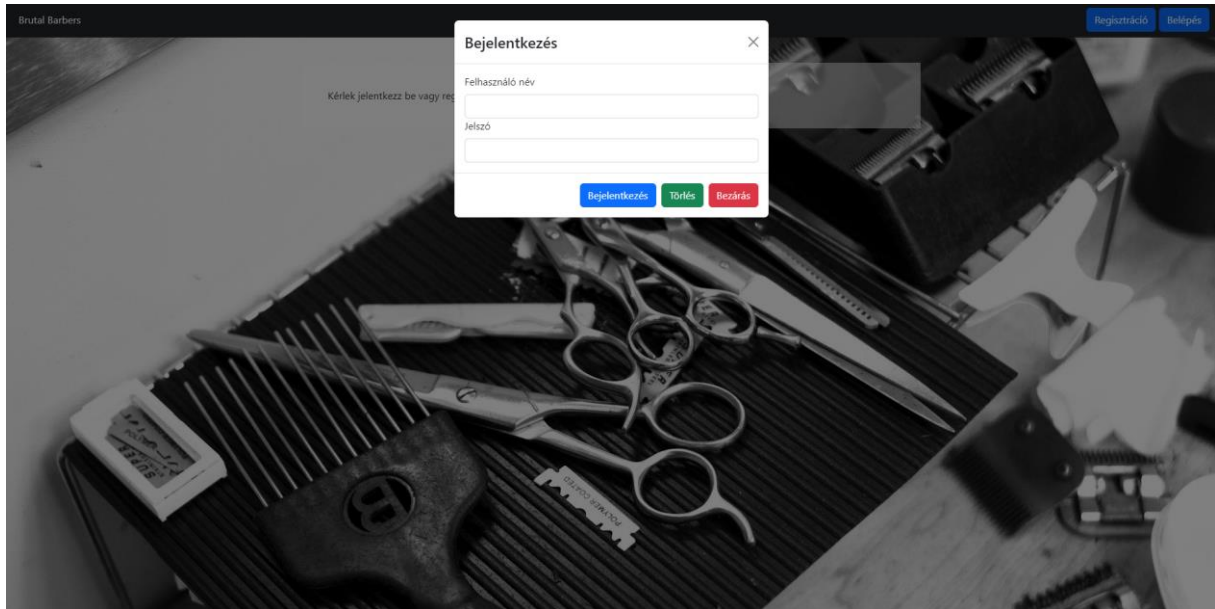


17. ábra Regisztráció

A gomb megnyomását követően egy ablak jelenik meg, ahol egy felhasználó nevet és egy jelszót kér, amit kétszer helyesen kell megadnunk saját magunk érdekében.

Belépés

Ha most hoztuk létre, vagy már van meglévő fiókunk, akkor nincs más hátra, mint a Belépés gombra kattintanunk.



18. ábra Bejelentkezés

Itt a fiókunk Felhasználónevére és jelszávára lesz szükségünk. Amint megadtuk ezeket, három lehetőség közül választhatunk :



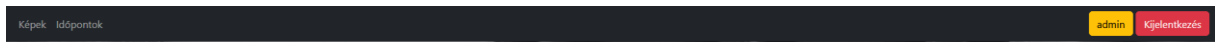
19. ábra Bejelentkezés gombjai

A bejelentkezés gombra kattintva tovább lépünk a főoldalra. A törlés gombbal töröljük a fiókunkat az oldalról. A bezár gombbal pedig bezárjuk a felugró ablakot.

Belépés az oldalra adminként

Ha tudjuk adminként a felhasználó nevünket és jelszavunkat, akkor nincs más hátra, minthogy egyszerűen csak belépünk. Miután ezt megtettük, akkor már láthatjuk is, hogy két, -az átlagos felhasználók számára nem elérhető - navigációs gombbal fogunk találkozni.

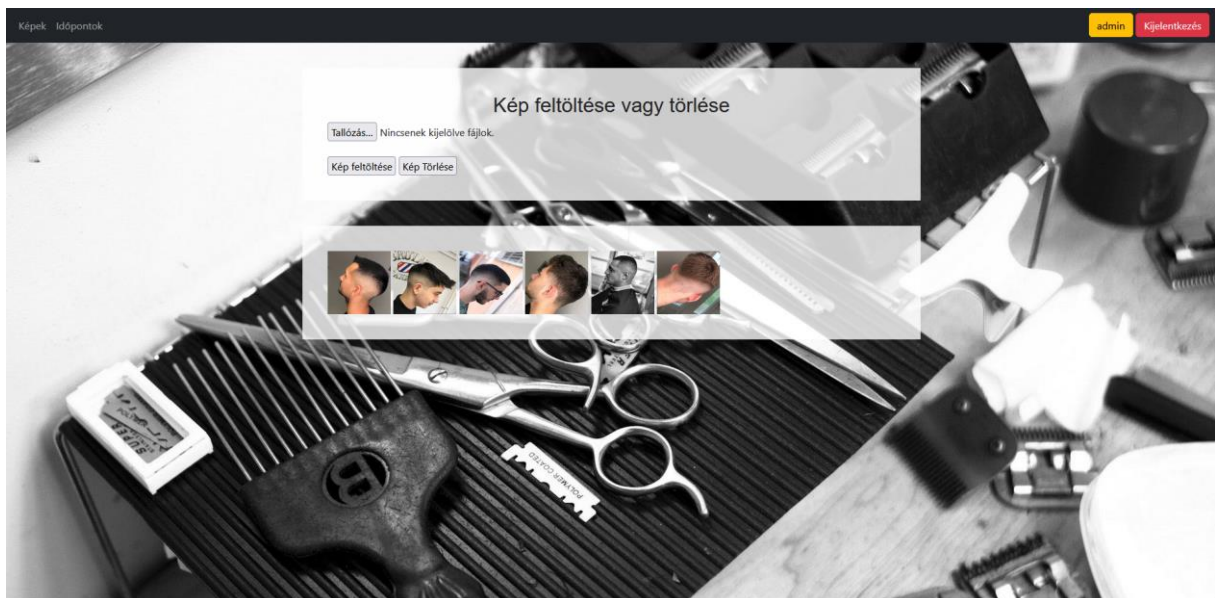
Az egyik az „időpontok”, és a másik a „képek” menüpont



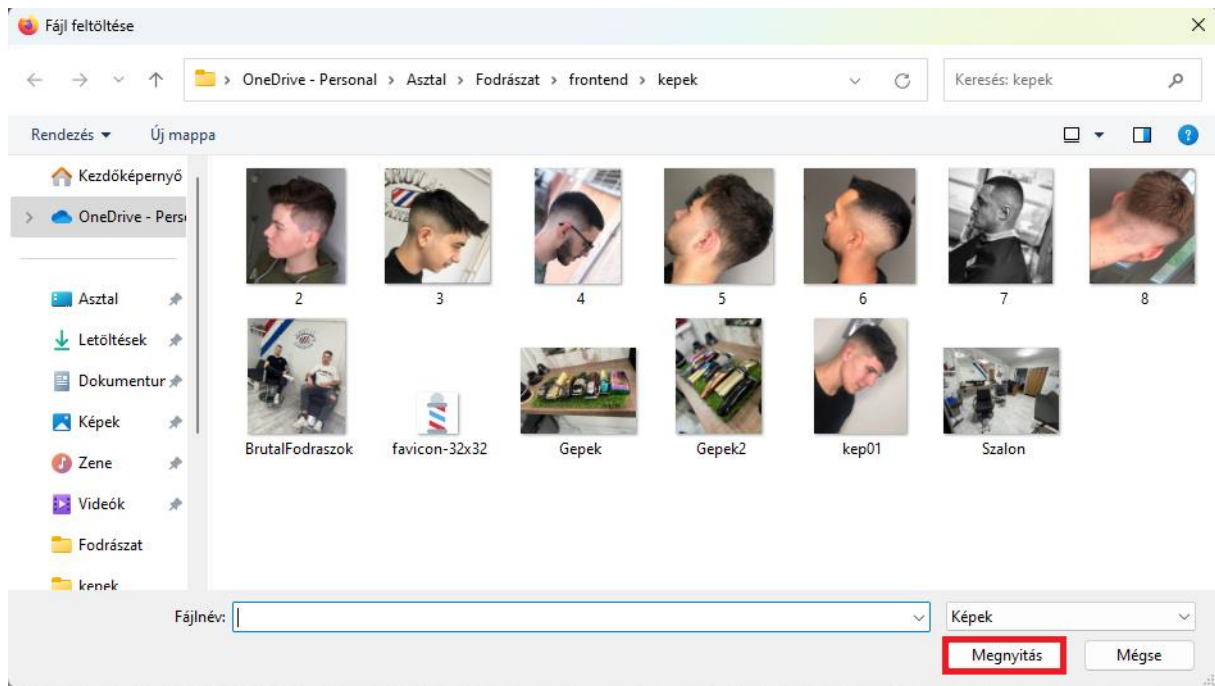
20. ábra Admin navigációs sáv

Referenciamunka feltöltése

Ha szeretnénk feltölteni egy referencia munkáról készült képet, akkor nincs más teendőnk, csak az, hogy rá kattintsunk a „Tallózás” gombra, kiválasszuk a feltölteni kívánt képet, és megnyomjuk a „megnyitás” gombot. Itt a félrekattintás elkerülése érdekében láthatjuk majd az adott kép nevét, majd ezt követően csak meg kell nyomjuk a „Kép feltöltése” gombot. Erre leginkább azért van szükség, hogy az oldalra újonnan látogatók mindig láthassák a minél frissebb munkákat, vagy sok esetben történik meg az is, hogy az adott illető nem tudja, hogy milyen frizurát szeretne a későbbiekben magán látni és ezek közül választhat is.



21. ábra Referenciamunka feltöltése



22. ábra Kép tallózása és kiválasztása

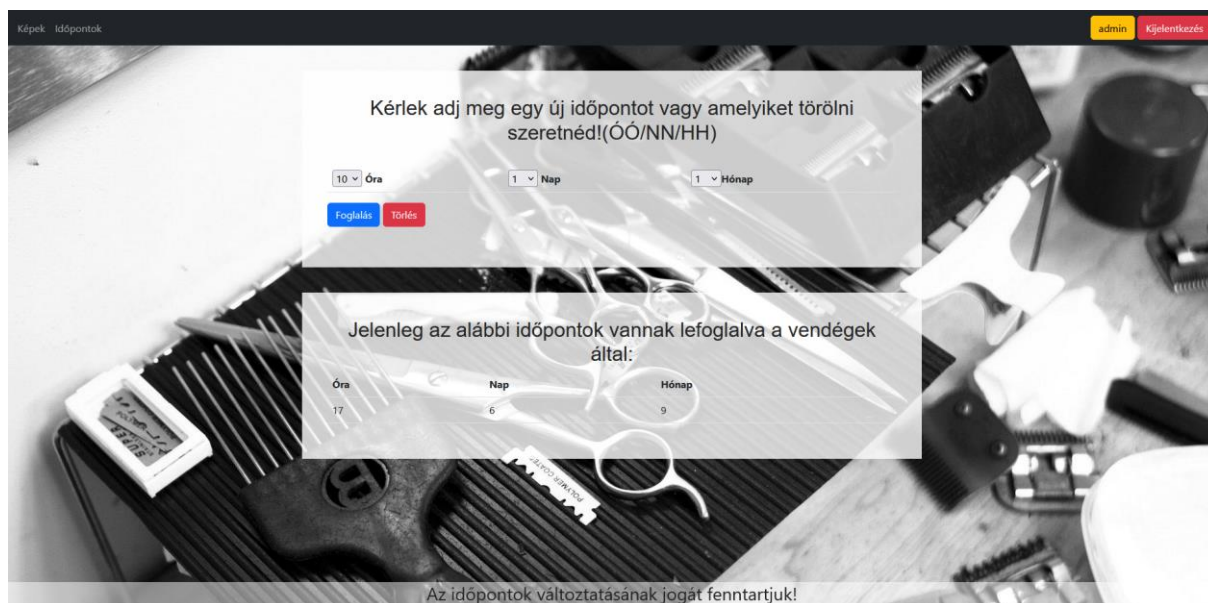
Referenciamunka törlése

Ha esetlegesen egy rossz képet, vagy csak szimplán nem szeretnénk a referenciamunkák között látni egy bizonyos képet, akkor nem sokban tér el a művelet, amit végre kell hajtanunk ez érdekében a kép feltöltéstől. Egyszerűen válasszuk a „Tallózás” gombot, majd válasszuk ki ugyan úgy, ahogy az a „6.ábrán” látszik és kattintsunk a „Kép Törlése” gombra.

Időpont hozzáadása vagy törlése adminként

Ha hozzászeretnél adni, vagy törölni egy időpontot, az „időpontok” oldal alján lévő táblázatban, akkor csak ki kell töltened az „Óra”, „Nap” és „Hónap” rubrikát, és kiválasztani, hogy Feltölteni vagy Törölni szeretnéd az általad megadott időpontot. Ha sikeres volt a feltöltés vagy a törlés, akkor azt egy felugró ablak fogja követni. Ez azt a célt fogja szolgálni, hogy ha egy felhasználó bármilyen formában jelezte feléd, hogy nem tud menni az adott időpontra, esetleg már távozott is a fodrásszékből, vagy legrosszabb esetben nem ment el a számára fenntartott időpontra, akkor tudd azt onnan eltávolítani.

Az alábbi képen az időpontok oldalt láthatjuk az admin oldalról:



23. ábra Admin oldali időpont foglalás és törlés

A navigációs sáv felhasználók részére

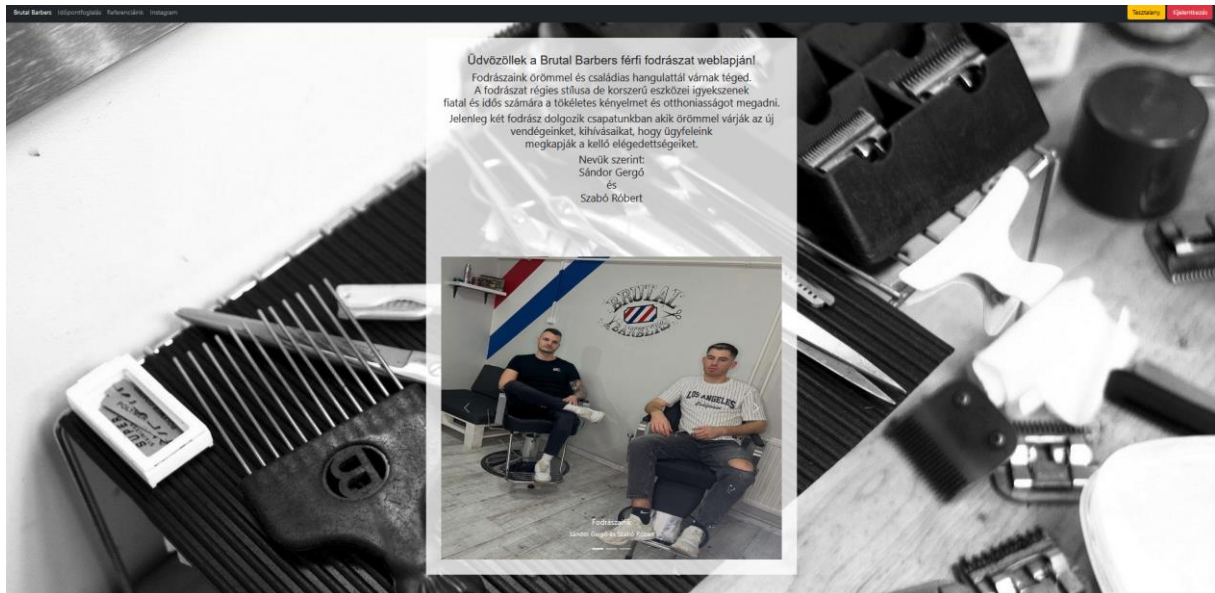
A főoldalon legfelül egy navigációs sávot láthatunk, ami a weboldal bármely pontján ott lesz. Baloldalon váltakoztathatjuk, hogy éppen melyik menüpontban szeretnénk lenni, a jobb oldalon a felhasználónevünk és egy Kijelentkezés gomb található.



24. ábra Navigációs sáv

A főoldal és tartalma

Az oldal törzsében egy rövid bemutatkozás után láthatod a fodrászok neveit, azon belül a nevükre kattintva pedig a közösségi oldalukat.



25. ábra Főoldal és tartalma

Az időpontfoglalás menüpont

Itt foglalhatsz magadnak időpontot a Brutal Barbers fodrászatba, és itt láthatod a már más vendégek által lefoglalt időpontokat.

Brutal Barbers Időpontfoglalás Referenciáink Instagram

Tesztalány Kijelentkezés

Kérlek add meg hogy mikor szeretnél hozzánk érkezni!
(ÓÓ/NN/HH)

Boltunk mindennap 10 és 19 óra között tart nyitva!

10 Óra 1 Nap 1 Hónap

Foglalás

Jelenleg az alábbi időpontok vannak lefoglalva a vendégek által:

Óra	Nap	Hónap
17	6	9

Az időpontok változtatásának jogát fenntartjuk!

26. ábra Időpontfoglalás menüpont

Itt megadhatod a számodra legalkalmasabb időpontot, amikor szeretnél érkezni.

Kérlek add meg hogy mikor szeretnél hozzánk érkezni!
(ÓÓ/NN/HH)

Boltunk mindennap 10 és 19 óra között tart nyitva!

10 Óra 1 Nap 1 Hónap

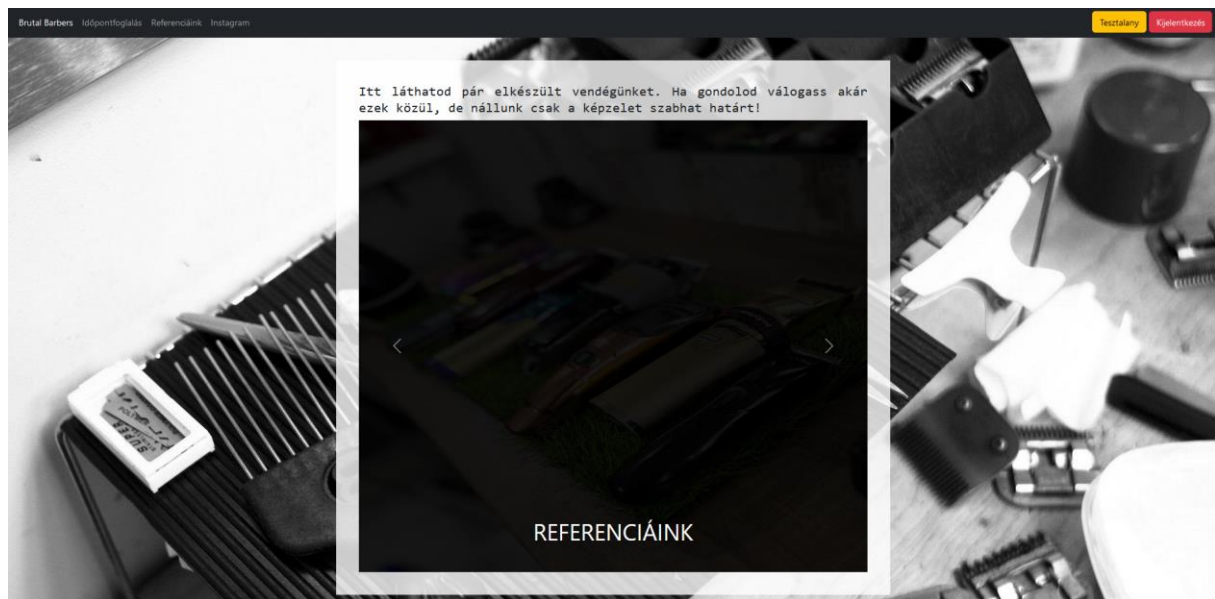
Foglalás

27. ábra Időpont lefoglalása

Az óra nap és hónap megadása után nincs már más hátra, mint megnyomd a „Foglalás” gombot.

A referenciáink menüpont

Itt láthatod pár fodrászunk által közölt kész hajvágásokat. A képen a jobb és a bal oldalt látható nyilak kattintásának segítségével könnyedén lehet nézelődni a feltöltött képek között.



28. ábra Referencia munkák menüpont

Fejlesztői dokumentáció

Fejlesztői környezet

Mivel egy weboldal készítését választottam ezért az ehhez szükséges alap eszközöket használtam. A felhasznált eszközök pedig a következők:

- Visual Studio Code magához a kódoláshoz
- az Apache Web Server
- a MySQL Database, ami az oldalak háttérében lévő adatbázist szolgáltatja
- phpMyAdmin Database Manager ami az adatbázis létrehozásához és kezeléséhez szükséges

Algoritmusok az oldalon

Regisztráció

Feladata a felhasználó által megadott felhasználó név és jelszó hozzáadása az adatbázishoz és majd lehetőséget biztosítson a bejelentkezésre.

A regisztráció algoritmus lépésenként:

A felhasználó kitölti mind a három kötelezően kitöltendő mezőt.

Ha valami üresen maradt, akkor azt az oldal hibaüzenettel jelzi.

A mezők kitöltése után a „Regisztráció” gomb lenyomásával a megadott adatok tárolódnak az adatbázisban. A folyamat sikerességét felugró ablakban lévő „Sikerült a regisztráció!” üzenet igazolja.

Először a lekérdezéssel kezdtem a backend oldalon a foglalt felhasználó nevek ellenőrzésével:

29. ábra Foglalt felhasználónév ellenőrzés backend

```
129 //Már foglalt felhasználói nevek lekérdezése
130 app.post('/felhasznaloLetezikE', (req, res) => {
131   var connection = mysql.createConnection({
132     host: 'localhost',
133     user: 'root',
134     password: '',
135     database: 'fodraszat'
136   });
137   connection.connect();
138   connection.query('SELECT felhasznalo.felhasznalo FROM felhasznalo WHERE felhasznalo.felhasznalo='${req.body.user}', function (err, rows) {
139     if (err) throw err
140     //console.log('Felhasználó check: ${rows}')
141     res.send(rows);
142   });
143   connection.end();
144 });
```

Ezt követte a tényleges regisztrációs adatok feltöltése az adatbázisba:

30. ábra Regisztrációs adatok feltöltése adatbázisba

```
146 //Regisztráció
147 app.post('/regisztracio', (req, res) => {
148   var connection = mysql.createConnection({
149     host: 'localhost',
150     user: 'root',
151     password: '',
152     database: 'fodraszat'
153   });
154   connection.connect();
155   connection.query('INSERT INTO felhasznalo VALUES (NULL, '${req.body.user}', '${req.body.jelszo}', '0', '0')',
156     function (err, rows, fields) {
157       if (err) throw err
158       //console.log('Regisztráció check: ${rows}');
159       res.send("Sikerült a regisztráció!");
160     });
161   connection.end();
162 });
```

Az ábrán azt láthatjuk, hogy a regisztráció ellenőrzésére szolgáló programrész két csoportra van szedve. Először ellenőrizzük, hogy minden mező ki van-e töltve.

```
1 //a regisztráció ellenőrzéséhez:
2 function felhasznaloVanEMar() {
3     //1. először megnézzük, hogy minden mező ki van-e töltve
4     if (document.getElementById('regUser').value != "" && document.getElementById('regPass1').value != "" && document.getElementById('regPass2').value != "") {
5         if (document.getElementById('regPass1').value == document.getElementById('regPass2').value) {
6             let bemenet = {
7                 user: document.getElementById('regUser').value
8             };
9             let url = "http://localhost:4000/felhasznaloLezerikE";
10            let fetchOptions = {
11                method: "POST",
12                body: JSON.stringify(bemenet),
13                headers: { "Content-type": "application/json; charset=UTF-8" }
14            };
15
16            fetch(url, fetchOptions)
17                .then(x => x.json())
18                .then(y => { //felhasznaloInev_ell(y)
19                    //alert(y[0].fnev);
20                    if (y.length != 0) {
21                        alert("Már foglalt felhasználói név!!!!");
22                    }
23                    else {
24                        regisztracio();
25                        document.getElementById('regUser').value = "";
26                        document.getElementById('regPass1').value = "";
27                        document.getElementById('regPass2').value = "";
28                    }
29                });
30        }
31        else {
32            alert("A két jelszó nem egyezik!");
33        }
34    }
35    //2. Ha nincs minden mező kitöltve
36    else {
37        alert("Minden mezőt kötelező kitölteni!");
38    }
39 }
```

31. ábra Regisztráció programrész 1.

Ezt követően jön a tényleges regisztráció, amit a következő program rész hajt végre az alábbi módszerrel:

```
98 //a tényleges regisztráció
99 function regisztracio() {
100     let bemenet = {
101         user: document.getElementById("regUser").value,
102         jelszo: document.getElementById("regPass1").value,
103     };
104     let url = "http://localhost:4000/regisztracio";
105     let fetchOptions = {
106         method: "POST",
107         body: JSON.stringify(bemenet),
108         headers: { "Content-type": "application/json; charset=UTF-8" }
109     };
110     fetch(url, fetchOptions)
111         .then(x => x.text())
112         .then(y => {
113             alert(y);
114         });
115 }
```

32. ábra Regisztráció programrész 2.

Belépés

A célja, hogy a felhasználó biztonságosan be tudjon lépni az oldalra a regisztráció után és ezt követően használhassa a weboldalt.

A felhasználó megadja a regisztrációnál használt felhasználónevet és jelszavát.

Ha valami üresen maradt, vagy hibásan lett beírva azt hibaüzenettel jelzi.

A jelszó ellenőrzése szintén az adatbázisban tárolt adat összehasonlításával történik.

```
201 //User ellenorzes bejelentkezéshez
202 app.post('/felhasznaloHelyesE', (req, res) => {
203     var connection = mysql.createConnection({
204         host: 'localhost',
205         user: 'root',
206         password: '',
207         database: 'fodraszat'
208     });
209     connection.connect();
210     connection.query('SELECT felhasznalok.felhasznalo, felhasznalok.jelszo FROM felhasznalok WHERE felhasznalok.jelszo='${req.body.jelszo}'
211     AND felhasznalok.felhasznalo='${req.body.felhasznalo}'');
212     function (err, rows) {
213         if (err) throw err
214         //console.log(`felhasználó: ${rows[0].felhasznalo}, jelszó: ${rows[0].jelszo}`);
215         res.send(rows);
216     };
217     connection.end();
218 });
```

33. ábra Felhasználó és jelszó lekérdezés belépéshez

Abban az esetben, ha a beléptetés sikeresen zárult, akkor az oldal átirányítódik a bejelentkezett oldalra, ahol a navigációs sáv jobb sarkában látható a felhasználónévünk és a „Kijelentkezés” gomb jelzi.

A következő képen már beregisztrált, de még be nem lépett felhasználó, ha helyesen kitölti a bejelentkezéshez szükséges adatokat akkor a bejelentkezés gombra kattintva az alábbi programrész kerül futtatásra:

Első sorban lekérdezzük a backend oldalról, hogy létezik-e a felhasználónév, és a hozzá tartozó jelszó az adatbázisban:

```
app.post('/felhasznaloHelyesE', (req, res) => {
  var connection = mysql.createConnection({
    host: 'localhost',
    user: 'root',
    password: '',
    database: 'fodraszat'
  });
  connection.connect();
  connection.query('SELECT felhasznalo.felhasznalo, felhasznalo.jelszo FROM felhasznalo WHERE felhasznalo.jelszo='${req.body.jelszo}' AND felhasznalo.felhasznalo='${req.body.felhasznalo}';',
    function (err, rows) {
      if (err) throw err;
      //console.log('felhasználó: ${rows[0].felhasznalo}, jelszó: ${rows[0].jelszo}');
      res.send(rows);
    });
  connection.end();
});
```

34. ábra Felhasználó létezik-e az adatbázisban

Ezt követően a frontend oldalról kezeljük azt:

```
311 function userLogin() {
312   if (document.getElementById('loginUser').value != "" && document.getElementById('loginPass').value != "") {
313     let bemenet = {
314       felhasznalo: document.getElementById('loginUser').value,
315       jelszo: document.getElementById('loginPass').value,
316     };
317     let url = "http://localhost:4000/felhasznaloHelyesE";
318     let fetchOptions = {
319       method: "POST",
320       body: JSON.stringify(bemenet),
321       headers: { "Content-type": "application/json; charset=UTF-8" }
322     };
323
324     fetch(url, fetchOptions)
325       .then(x => x.json())
326       .then(y => userStateUpdate(y))
327   }
328   else {
329     alert("Minden mezőt kötelező kitölteni!");
330   }
331 };
```

35. ábra Belépés programrész

A következő ábrán már a bejelentkezett felhasználó nevének kiírásában segít:

```
208 //Bejelentkezett felhasznalo kiirasa
209 function melyikNev() {
210     let url = "http://localhost:4000/whoIs";
211     let fetchOptions = {
212         method: "POST",
213     };
214
215     fetch(url, fetchOptions)
216         .then(x => x.json())
217         .then(y => nevKiir(y))
218 };
219
220 function nevKiir(nev) {
221     //console.log(nev);
222     for (var i of nev) {
223         document.getElementById('loggedInUser').innerHTML = i.felhasznalo;
224     }
225 };
```

36. ábra Felhasználó név kiíratása bejelentkezett felületeken

A következő ábrákon a kijelentkezéshez használt programrészek láthatók backend és frontend részről:

```
302 app.post('/kijelentkezés', (req, res) => {
303     var connection = mysql.createConnection({
304         host: 'localhost',
305         user: 'root',
306         password: '',
307         database: 'fodnaszat'
308     });
309     //console.log(req.body.felhasznalo);
310     connection.connect();
311     connection.query(`UPDATE felhasznalok SET bejelentkezve = '0' WHERE felhasznalok.felhasznalo = '${req.body.felhasznalo}'`,
312         function (err, rows) {
313             if (err) throw err
314             res.send(rows);
315         });
316     connection.end();
317 });
```

37. ábra Kijelentkezés az oldalról backend

```

228 //Kijelentkezés
229 function logout() {
230     let bemenet = {
231         felhasznalo: document.getElementById('loggedInUser').innerHTML,
232     };
233     let url = "http://localhost:4000/kijelentkezés";
234     let fetchOptions = {
235         method: "POST",
236         body: JSON.stringify(bemenet),
237         headers: { "Content-type": "application/json; charset=UTF-8" }
238     };
239
240     fetch(url, fetchOptions)
241         .then(x => x.json())
242         .then(y => {
243             alert("Sikeres kijelentkezés");
244             window.location.href = "index.html";
245         });
246 };

```

39. ábra Kijelentkezés a weboldalról frontend

Admin oldal*

Az oldal sajátosságát az adatbázisban található „rang” mező biztosítja, hogy ezt csak admin láthassa és tudja kezelni azt.

Időpontok hozzáadása és törlése*

A törlés funkció, ami az időpontfoglalás oldalon egy átlagos felhasználó számára nem látható, hogy egy átlagos felhasználó ne tudja más felhasználók időpontját törölni.

A felhasználói oldalról is elérhető foglalás gomb bent maradt az adminok részére, ugyanis ha valaki nem az oldalt használja időpont kérésére, hanem befárad az üzletbe vagy valamilyen formában tudtára adja egy fodrász számára, akkor azt a fodrász is tudja feltölteni.

fodraszat felhasznalok	
id : int(11)	
felhasznalo : varchar(255)	
jelszo : varchar(255)	
# rang : int(11)	
# bejelentkezve : int(11)	

38. ábra Felhasználók tábla, rang mező

A képen a „Törlés” gomb működése látható:

```
71 // Foglalt időpontok törlése admin részére
72 function idopontTorles(){
73     let bemenet = {
74         ora: document.getElementById("dateOra").value,
75         nap: document.getElementById("dateNap").value,
76         honap: document.getElementById("dateHonap").value,
77     };
78     let url = "http://localhost:4000/datumTorles";
79     let fetchOptions = {
80         method: "POST",
81         body: JSON.stringify(bemenet),
82         headers: { "Content-type": "application/json; charset=UTF-8" }
83     };
84     fetch(url, fetchOptions)
85         .then(x => x.text())
86         .then(y => {
87             alert(y);
88         });
89 };
90
```

40. ábra Időpont törlése programrész

Referenciamunka feltöltése

Az admin felhasználók részére lehetőséget adtam arra, hogy bármikor, amikor egy vendég haját elkészülés után lefényképezik, akkor azt a képet fel tudják tölteni az felhasználók számára. Ahhoz hogy ez létrejöjjön több szempontot használtam fel a megvalósításhoz. Elsőként a fájlok lementése egy mappába backend részről, másodszor a kép nevének felküldése az adatbázisba:

```
319 app.post('/upload',
320   fileUpload({ createParentPath: true }),
321   filesPayloadExists,
322   fileExtLimiter(['.png', '.jpg', '.jpeg']),
323   fileSizeLimiter,
324   (req, res) => {
325     const files = req.files
326
327     Object.keys(files).forEach(key => {
328       const filepath = path.join(__dirname, 'files', files[key].name)
329       files[key].mv(filepath, (err) => {
330         if (err) return res.status(500).json({ status: "error", message: err })
331         console.log(files[key].name)
332       })
333     })
334     res.header("Access-Control-Allow-Origin", "*")
335     return res.status(200).json({message: "Sikeres feltöltés!"})
336   }
337 )
```

41. ábra Képek mentése mappába backend részről

```
339 app.post('/kepnev', (req, res) => {
340   var connection = mysql.createConnection({
341     host: 'localhost',
342     user: 'root',
343     password: '',
344     database: 'fodraszat'
345   });
346   //console.log(req.body.felhasznalo);
347   connection.connect();
348   connection.query(`INSERT INTO kepek VALUES (NULL, '${req.body.kep}')`,
349     function (err, rows) {
350       if (err) throw err
351       res.send("Sikeres feltöltés!");
352     });
353   connection.end();
354 });
```

42. ábra Képek nevének feltöltése az adatbázisba

Az alábbi képen az a kódsor látható, ami a backend rész mappába mentését éri el, majd a kép nevének elküldése a backend részre:

```
// Képfeltöltés Files mappába
function kepfeltoltes(){

    // Object
    const myFiles = document.getElementById('refKep').files
    if(myFiles.length>0){
        const sendFiles = async () => {
            const formData = new FormData()

            Object.keys(myFiles).forEach(key => {
                formData.append(myFiles.item(key).name, myFiles.item(key))
            })

            const response = await fetch('http://localhost:4000/upload', {
                method: 'POST',
                body: formData
            })

            const json = await response.json()
            document.getElementById('myFiles').value = ''

        }
        sendFiles()
    }
}
```

43. ábra Kép feltöltése programrész 1.

```
409 ~ function kepev() {
410   let url ="http://localhost:4000/kepev"
411   let kepev = document.getElementById('refKep').value.split("\\")
412   ~ let bemenet={
413     kepev: kepev[2]
414   }
415   ~ let fetchOptions={
416     method: "POST",
417     body: JSON.stringify(bemenet),
418     headers: {"Content-type": "application/json; charset=UTF-8",
419               "Access-Control-Allow-Origin" : "*" }
420   }
421   fetch (url, fetchOptions)
422   .then(x=>x.text())
423   ~ .then(y=> {
424     alert(y)
425     kepfeltoltes()
426   })
427 }
```

44. ábra Kép feltöltése programrész 2.

Referenciamunkák megjelenítése

A felhasználó oldali referenciamunkák egy javascript kóddal kerül az oldalra:

```
function kepek3() {
  fetch("http://localhost:4000/kepek")
    .then(x=>x.json())
    .then(y=>megjelenit(y))

  function megjelenit(adatok) {
    var sz=""
    for(var sor of adatok) {
      sz+=`
      <div class="carousel-item c-item ">
        
        <div class="carousel-caption d-none d-md-block">
          <div>
            </div>
          </div>
        </div>
      `
    }
    document.getElementById("ref4").innerHTML+=sz;
  }
}
```

45. ábra Referenciamunkák megjelenítése

Időpont feltöltése és lekérdezése

Időpont ellenőrzése és feltöltése

Első sorban le kell futtatnunk egy lekérdezést, amivel leellenőrizzük, hogy a feltöltendő időpont foglalt-e, majd csak ezt követően történhessen meg a tényleges időpontfeltöltés.

46. ábra Időpontok ellenőrzése backendről

```
41 app.post('/FoglaltE', (req, res) => {
42   var connection = mysql.createConnection({
43     host: 'localhost',
44     user: 'root',
45     password: '',
46     database: 'fodraszat'
47   });
48   connection.connect();
49   connection.query(`SELECT idopontok.ora, idopontok.nap, idopontok.honap FROM idopontok WHERE idopontok.ora='${req.body.ora}'
50 AND idopontok.nap='${req.body.nap}' AND idopontok.honap='${req.body.honap}';`, function (err, rows) {
51     if (err) throw err
52   });
53   res.send(rows);
54 });
55 connection.end();
56 });
```

```

76 //Idopontfeltöltés
77 v app.post('/datumFel', (req, res) => {
78 v   var connection = mysql.createConnection({
79     host: 'localhost',
80     user: 'root',
81     password: '',
82     database: 'fodraszat'
83   });
84   connection.connect();
85 v   connection.query(`INSERT INTO idopontok VALUES (NULL, '${req.body.ora}', '${req.body.nap}', '${req.body.honap}')`,
86 v     function (err, rows, fields) {
87       if (err) throw err
88       //console.log(`Regisztráció check: ${rows}`);
89       res.send("Sikerült az időpontfoglalás!");
90     });
91   connection.end();
92 });

```

47. ábra Időpont feltöltése az adatbázisba

A foglalás lekérdezéseit a következő JavaScript részlet dolgozza fel és kezeli:

```
72 function FoglaltE(){
73     let bemenet = {
74         ora: document.getElementById("dateOra").value,
75         nap: document.getElementById("dateNap").value,
76         honap: document.getElementById("dateHonap").value,
77     };
78     let url = "http://localhost:4000/FoglaltE";
79     let fetchOptions = {
80         method: "POST",
81         body: JSON.stringify(bemenet),
82         headers: { "Content-type": "application/json; charset=UTF-8" }
83     };
84     fetch(url, fetchOptions)
85         .then(x => x.json())
86         .then(y => {
87             if (y.length != 0) {
88                 alert("Már foglalt időpont, kérlek válassz másikat!");
89             }
90             else {
91                 let bemenet = {
92                     ora: document.getElementById("dateOra").value,
93                     nap: document.getElementById("dateNap").value,
94                     honap: document.getElementById("dateHonap").value,
95                 };
96                 let url = "http://localhost:4000/datumFel";
97                 let fetchOptions = {
98                     method: "POST",
99                     body: JSON.stringify(bemenet),
100                     headers: { "Content-type": "application/json; charset=UTF-8" }
101                 };
102                 fetch(url, fetchOptions)
103                     .then(x => x.text())
104                     .then(y => {
105                         alert(y);
106                     });
107             }
108         });
109 };
```

48. ábra Foglalt időpontok ellenőrzése Frontend

Foglalt időpontok lekérdezése

Ahhoz hogy a weblapon láthassuk a már lefoglalt időpontokat, először is le kell kérdeznünk őket az adatbázisból

```
111 // foglalt időpontok kiíratása
112 app.get('/foglalt', (req, res) => {
113     var mysql = require('mysql')
114     var connection = mysql.createConnection({
115         host: 'localhost',
116         user: 'root',
117         password: '',
118         database: 'fodraszat'
119     })
120     connection.connect()
121     connection.query("select * from idopontok", function (err, rows, fields) {
122         if (err) throw err
123         res.send(rows);
124     })
125     connection.end()
126 })
```

49. ábra Foglalt időpontok lekérdezése adatbázisból

Itt azt a programrészt láthatjuk, ahol a már feltöltött időpontokat lekérdezi, és tölti fel a legfoglalt időpontokat táblázatba:

```
42 //foglaltak lekérdezése
43 function foglalt(){
44     fetch("http://localhost:4000/foglalt")
45     .then(x => x.json())
46     .then(y => {
47         //alert(y);
48         //alert(JSON.stringify(y))
49         var b=""
50         y.forEach(elem => {
51             b+=''<tr>'
52             b+=''<td>'+elem.ora+'</td>'
53             b+=''<td>'+elem.nap+'</td>'
54             b+=''<td>'+elem.honap+'</td>'
55             b+=''</tr>'
56         });
57         document.getElementById("foglalt").innerHTML=b;
58     });
59 };
```

50. ábra Időpontok lekérdezése és táblázat feltöltése

Az adatbázis különböző táblái és azok mezői

A felhasználók tábla



fodraszat_felhasznalok	
id	int(11)
felhasznalo	varchar(255)
jelszo	varchar(255)
rang	int(11)
bejelentkezve	int(11)

51. ábra Felhasználók tábla

Elsősorban a felhasználók adatainak tárolására szolgál. Az első mezője az id, ez egy szám típusú mező. Ez a tábla elsődleges kulcsa, automatikusan növekszik és generálódik, ahányszor egy új felhasználó regisztrál. Ebből a számból akár megállapíthatjuk a regisztrált felhasználók számát is. Ez a mező azért fontos, mert a beléptetés és a saját adatok lekérdezése során tudjuk a felhasználót beazonosítani.

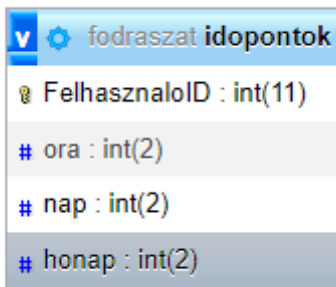
A második mező felhasznalo (varchar) mező, ezért szöveget és számot is tartalmazhat. Ez egy másodlagos megkülönböztetése a felhasználónak regisztrációnál kötelező megadni, mely felhasználó barátságosabbá teszi az oldalt.


A harmadik mező a jelszo, ami szintén egy vegyes karakterű mező (számot és szöveget is tartalmazhat). Itt nem adtam meg sem karakteri béli, sem hossz béli bonyolultsági összetételt, ugyanis az oldal nem tartalmaz bizalmas adatokat. Így a feltörés veszélye sem áll fent. A felhasználó bármit megadhat jelszóként, annak egyeznie kell a bejelentkezésnél használt jelszóval. Elfelejtett jelszó esetén új regisztráció szükséges.

A negyedik mező a rang, ami egy int típusú mező. Itt azt tudhatjuk meg a bejelentkezett személy rendelkezik-e egyes típusú ranggal, ami admint, vagy 0-val ami, átlagos felhasználót jelent.

Az ötödik mező egy olyan int típusú mező, amely arra szolgál, hogy a felhasználó éppen be van-e jelentkezve (1) vagy nincs (0).

Az időpontok tábla



fodraszat idopontok	
	FelhasználóID : int(11)
#	ora : int(2)
#	nap : int(2)
#	honap : int(2)

52. ábra Időpontok tábla

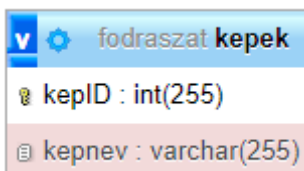
Az első mező összeköttetésben áll a felhasználók tábla első „id” mezejével. Mivel ezáltal tudjuk majd a későbbiekben beazonosítani azt, hogy ki töltött fel új időpontot.



Az ora, a felhasználó által megadott órát tárolja, amit a későbbiekben felhasználhatunk.

A nap mező az aktuális hónap napját tárolja az adatbázisban.

Utolsó sorban a honap mező, egy int típusú mező, ami a hónapot fogja tárolni a későbbiekben.

A képek tábla



fodraszat kepek	
	kepID : int(255)
	kepnev : varchar(255)

53. ábra Képek tábla

Az első mezőben a „KepID” látható, ami int típusú, és Auto Increment-tel van ellátva. Ez a mező annak érdekében lett készítve, hogy ha a későbbiek fejlesztés során lesz használva.

„kepnev”: ez a mező varchar típusú, mivel ide a kép neve kerül feltöltésre, azért , hogy a html kódba való beillesztésekor csak az elérési út megadása után csak a kép neve kerüljön lekérdezésre.

Tesztelői dokumentáció

Az oldal a következő böngészőkben lett tesztelve:

Google Chrome (Version 65.0.3325.181 (Official Build) (64-bit))

Mozilla Firefox Developer Edition (Version 59.0b11 (64-bit))

Microsoft Edge (Version 41.16299.371.0)

A különböző böngészők között nem tapasztaltam semmilyen megjelenítési rendellenességet. Ezt a bootstrap keretrendszernek köszönhető, ugyanis a fenti böngészők mindegyike támogatja.

Programozásban nem jártas felhasználó véleménye

Az oldal felhasználók által való tesztelésében egy fodrász és egy barátom segítségét kértem, akik az informatikai dolgokhoz kevésbé értenek.

A fodrász meglátása:

Az oldal megnyitásakor, könnyen zökkenőmentesen teljesült a folyamat. Gondolok itt arra, hogy a felhasználó akadály nélkül tudott létrehozni egy új fiókot és tudott belépni. A belépést követően saját nevére kattintva meglátta a közösségi médiában megtalálható weboldalát. Majd ezek után egyből átugrott az időpontfoglalás földre és egyértelműen ki is töltötte azt. Viszont arra lett figyelmes, hogy ugyan azt az időpontot többször is le tudja foglalni. Ez viszont egy súlyosabb hiba volt, aminek azonnal neki is láttam, hisz ez is fő kritériuma egy időpont lefoglalásának.

Meglátása szerint egy átlagos felhasználói fölről a saját lefoglalt időpontunkat kellene tudjuk törölni.

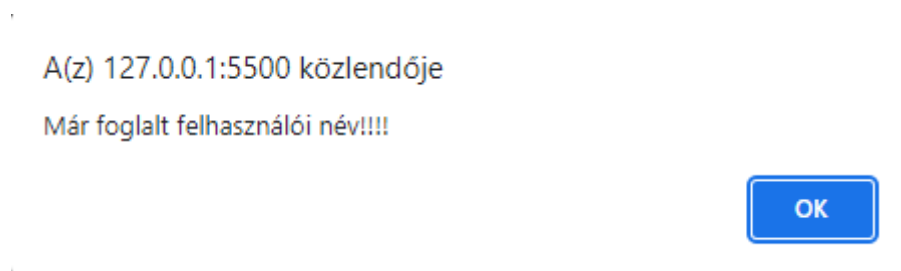
A barátom meglátása:

Neki kissé nehézkes volt, mivel már a regisztrációnál fentakadt azon problémán, hogy helytelenül adta meg másodjára a jelszavát, és így nem történt semmi, és bejelentkezni sem tudott. Majd ezt követően ismét a regisztráció gombra, hiszen nem tudta, hogy ténylegesen sikerült-e a regisztráció. Ekkor jutott eszembe, hogy kelleni fog egy olyan algoritmus, amely azt szolgálja, hogy a „jelszó1”-es és a „jelszó2” es mező ugyan azt tartalmazza- e a regisztráció gomb lenyomását követően, és ha minden helyesen van kitöltve, és sikeres a feltöltés az adatbázisba, akkor azt egy „Sikeres regisztrációs”-s üzenet jelezze.

Az oldal visszajelzései

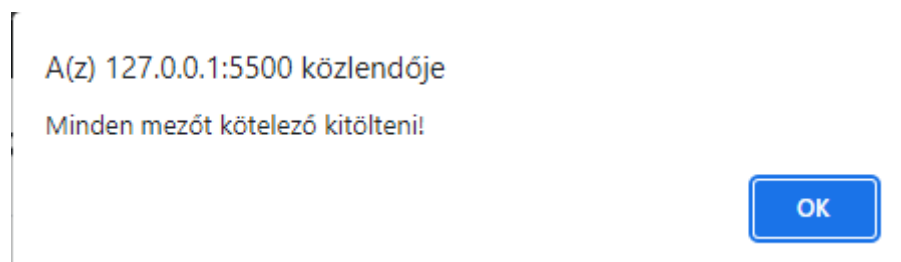
Hibaüzenetek

Ha már foglalt felhasználó névvel próbálunk regisztrálni:



54. ábra Hibaüzenetek 1.

Ha nem töltöttünk ki minden mezőt a regisztráció során:



55. ábra Hibaüzenetek 2

Ha a regisztrációkor megadott jelszót nem helyesen írjuk be másodszor:

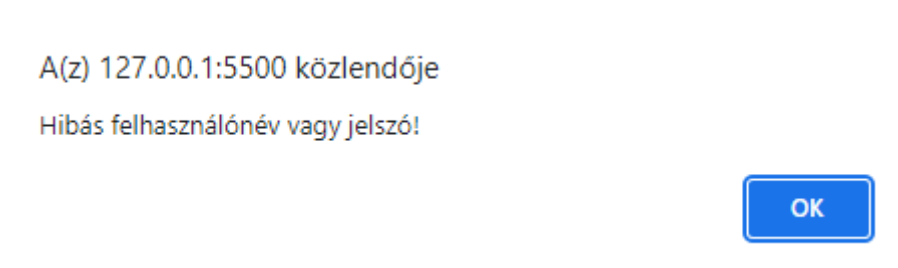
A(z) 127.0.0.1:5500 közlendője

A két jelszó nem egyezik!

OK

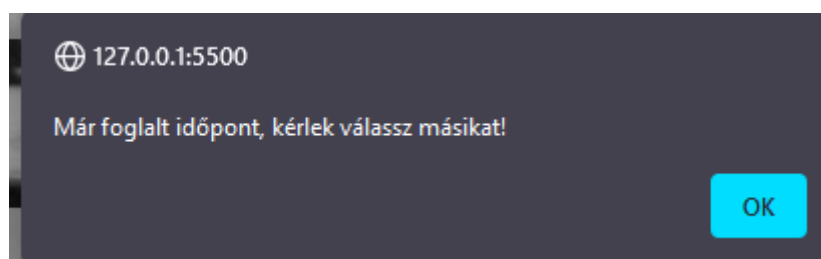
56. ábra Hibaüzenetek 3

Bejelentkezéskor ha hibásan adjuk meg vagy a felhasználónevünket vagy a jelszavunkat:



57. ábra Hibaüzenetek 4

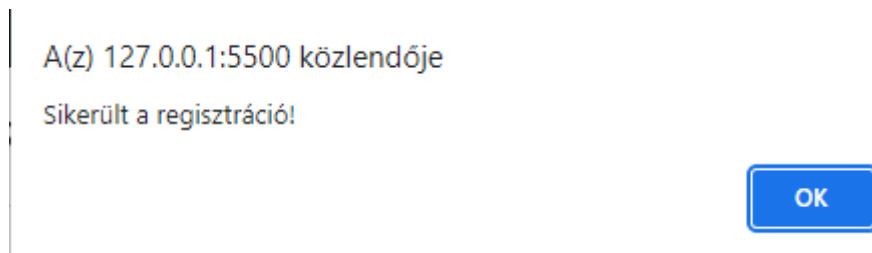
Ha már foglalt időpontot szeretnénk az oldalon lefoglalni:



58. ábra Hibaüzenetek 5

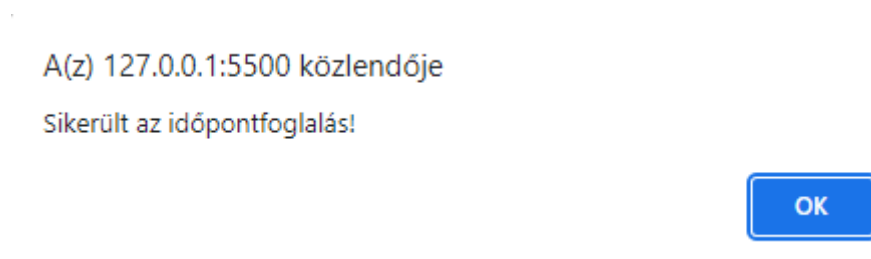
Általános üzenetek

Ha sikeres a regisztráció akkor a gomb megnyomása után a következőt kell látnunk:



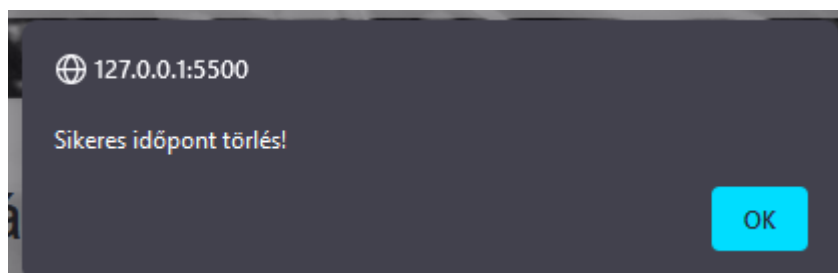
59. ábra Általános üzenetek 1

Ha sikeresen töltöttünk fel egy új időpontot az adatbázisba:



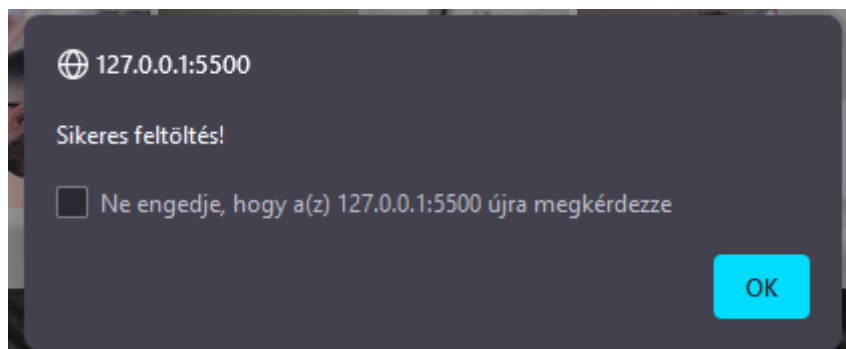
60. ábra Általános üzenetek 2

Ha sikeresen törölünk egy időpontot az admin felületről:



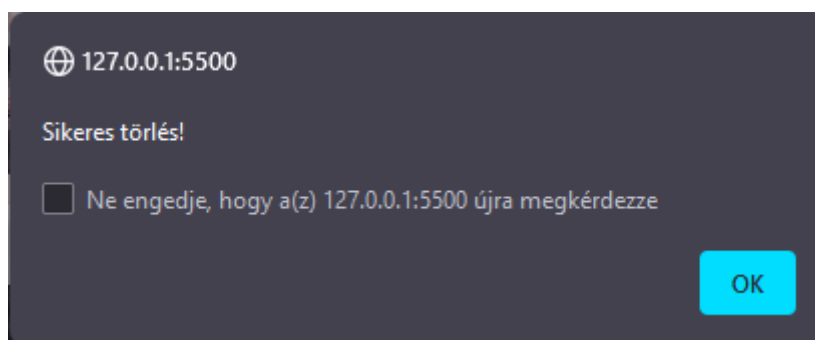
61. ábra Általános üzenetek 3

Ha sikeresen töltünk fel képet a referenciamunkák közzé az admin oldalról:



62. ábra Általános üzenetek 4

Ha sikerült a kép törlése az admin oldalról a refrenciamunkák közül:



63. ábra Általános üzenetek 5

Összefoglalás

Az oldal dizájn része

Az oldal a mai kor elvárásainak megfelelő, a felülete letisztult, mely áttekinthető, és mindig a lényegi, a felhasználókat leginkább érdeklő információkra fókuszál.

Nincsenek színes, vibráló felületek, képek, esetleg szembeszökő elemek, nélkülözi a hatásvadász elemeket, továbbá azt a minőséget, valamint színvonalat tükrözni, amit az oldal készítője, továbbá az oldal célkitűzése reprezentál.

Ezzel az egyszerű megjelenéssel érhető el a legkönnyebben, hogy a felhasználó azt kapja, amiért az oldalt betöltötte. A legszembetűnőbb részen, a fejlécben megtalálható minden lényegi információ, a főoldaltól kezdve a regisztráción át a kijelentkezésig. Ezek a gombok minden oldalon elérhetőek és ugyan ott találhatóak.

Célok, amelyeket nem sikerült teljesíteni

Egy admin belépésekor szerettem volna, hogy az ő rangja által az tudjon létrehozni egy új admin felhasználót. Ez idő hiányában nem sikerült sajnos.

A felhasználók a saját maguk által lefoglalt időpontok szerkesztése és törlése. Ezt a későbbiek során szeretném még megoldani.

Fejlesztési lehetőségek

A fent említettek megvalósításán túl egy új menüpontban a fodrászok kiértékelése, egy-egyől 5-ig terjedő skálán, és szöveges megjegyzést is hozzáfűzni.

Egy olyan kapcsolattartó fület létrehozni, ami egyből a fodrász és felhasználó közötti beszélgetést tud kezelni, végrehajtani.

Elhelyezni egy email-és egy telefonszámot a fodrászoktól az időpontfoglalás fülben annak érdekében, hogy ha esetlegesen bármilyen probléma merülne fel az időponttal.

A későbbiekben a fodrászoknak (adminoknak), egy olyan email küldése, amelyben értesülnek az újabb lefoglalt időpontokról.

Regisztrációkor telefonszám vagy email megadása kötelező legyen, hogy a fodrász könnyedén fel tudja venni a kapcsolatot az ügyféllel.

Irodalomjegyzék

<https://www.w3schools.com>

<https://getbootstrap.com>

<https://prog.hu>

<https://expressjs.com/>

<https://nodejs.org/en>

Ábrajegyzék

1. ábra Windows 10 logó -----	5
2. ábra MySQL logó-----	5
3. ábra Google Chrome logó -----	5
4. ábra Total Commander logó -----	5
5. ábra Visual Studio Code-----	6
6. ábra NodeJS logó-----	6
7. ábra Bootstrap logó-----	6
8. ábra XAMPP control panel -----	7
9. ábra Phpmyadmin importálás -----	8
10. ábra Terminal megnyitás -----	9
11. ábra csomagok feltelepítése -----	10
12. ábra "háttérprogram" elindítása-----	10
13. ábra Sikeres futtatás -----	10
14. ábra Live Server telepítése-----	11
15. ábra Go Live gomb -----	11
16. ábra A weboldalra érkezéskor -----	15
17. ábra Regisztráció -----	16
18. ábra Bejelentkezés -----	17
19. ábra Bejelentkezés gombjai-----	17
20. ábra Admin navigációs sáv -----	18
21. ábra Referenciamunka feltöltése-----	18
22. ábra Kép tallózása és kiválasztása -----	19
23. ábra Admin oldali időpont foglalás és törlés -----	20
24. ábra Navigációs sáv -----	20
25. ábra Főoldal és tartalma-----	21
26. ábra Időpontfoglalás menüpont-----	22
27. ábra Időpont lefoglalása -----	22
28. ábra Referencia munkák menüpont-----	23
29. ábra Foglalt felhasználónév ellenőrzés backend -----	25
30. ábra Regisztrációs adatok feltöltése adatbázisba -----	25
31. ábra Regisztráció programrész 1. -----	26

32. ábra Regisztráció programrész 2. -----	27
33. ábra Felhasználó és jelszó lekérdezés belépéshez -----	27
34. ábra Felhasználó létezik e az adatbázisban -----	28
35. ábra Belépés programrész -----	28
36. ábra Felhasználó név kiírtatása bejelentkezett felületeken -----	29
37. ábra Kijelentkezés az oldalról backend -----	29
39. ábra Kijelentkezés a weboldalról frontend -----	30
38. ábra Felhasználók tábla, rang mező -----	30
40. ábra Időpont törlése programrész -----	31
41. ábra Képek mentése mappába backend részről -----	32
42. ábra Képek nevének feltöltése az adatbázisba -----	32
43. ábra Kép feltöltése programrész 1. -----	33
44. ábra Kép feltöltése programrész 2. -----	33
45. ábra Referenciamunkák megjelenítése -----	34
46. ábra Időpontok ellenőrzése backendről -----	34
47. ábra Időpont feltöltése az adatbázisba -----	35
48. ábra Foglalt időpontok ellenőrzése Frontend -----	36
49. ábra Foglalt időpontok lekérdezése adatbázisból -----	37
50. ábra Időpontok lekérdezése és táblázat feltöltése -----	37
51. ábra Felhasználók tábla -----	38
52. ábra Időpontok tábla -----	39
53. ábra Képek tábla -----	39
54. ábra Hibaüzenetek 1. -----	41
55. ábra Hibaüzenetek 2 -----	41
56. ábra Hibaüzenetek 3 -----	42
57. ábra Hibaüzenetek 4 -----	43
58. ábra Hibaüzenetek 5 -----	43
59. ábra Általános üzenetek 1 -----	43
60. ábra Általános üzenetek 2 -----	44
61. ábra Általános üzenetek 3 -----	44
62. ábra Általános üzenetek 4 -----	44
63. ábra Általános üzenetek 5 -----	45

Eredetiségnyilatkozat

Alulírott **Balogh Benedek** nyilatkozom, hogy záró dolgozatom a saját szellemi termékem.

Debrecen, 2023. április. 12

Balogh Benedek

aláírás