győri szakképzési centrum

Jedlik Ányos
Gépipari és Informatikai
Technikum és Kollégium

9021 Győr, Szent István út 7.
+36 (96) 529-480
+36 (96) 529-448

OM: 203037/003
jedlik@jedlik.eu
www.jedlik.eu

Záródolgozat feladatkiírás

Tanulók neve:	Vangel Celina Hanna, Letenyei Szilárd
Képzés:	esti
Szak:	506131203 Szoftverfejlesztő és -tesztelő

A záródolgozat címe:

MenuGenius

Konzulens:	Kottra Richárd
Beadási határidő:	2024.04.15.

Győr, 2023. 10. 01

Módos Gábor
igazgató

győri szakképzési centrum

Jedlik Ányos
Gépipari és Informatikai
Technikum és Kollégium

9021 Győr, Szent István út 7.
+36 (96) 529-480
+36 (96) 529-448

OM: 203037/003
jedlik@jedlik.eu
www.jedlik.eu

Konzultációs lap

	A konzultáció		Konzulens aláírása
	ideje	témája	
1.	2023.10.21.	Témaválasztás és specifikáció	
2.	2024.03.14.	Záródolgozat készültségi fokának értékelése	
3.	2024.04.11.	Dokumentáció véglegesítése	

Tulajdonosi nyilatkozat

Ez a dolgozat a saját munkánk eredménye. Dolgozatunk azon részeit, melyeket más szerzők munkájából vettünk át, egyértelműen megjelöltük.
Ha kiderülne, hogy ez a nyilatkozat valótlan, tudomásul vesszük, hogy a szakmai vizsgabizottság a szakmai vizsgáról kizár minket és szakmai vizsgát csak új záródolgozat készítése után tehetünk.

Győr, 2024. április 09.

_____
tanuló aláírása

_____
tanuló aláírása

Tartalom

1. Bevezetés....4
2. Fejlesztői dokumentáció....6
2.1. Munkamegosztás....6
2.1.1. Feladatok elosztása....6
2.2. Verziókezelés....6
2.3. Frontend....7
2.3.1. Dizájn tervek....7
2.4. Használt technológiák....8
2.4.1. Angular....8
2.4.2. A projekten belül használt „service”-ek....9
2.4.3. Navigáció a projekten belül....12
2.4.4. Angular és Bootstrap 5.3....12
2.4.5. Reszponzivitás....12
2.4.6. Főoldal....13
2.5. Backend....13
2.5.1. Használt technológiák....13
2.5.2. Laravel....13
2.5.3. Laravel config mappa és a .env fájl a projekten belül....14
2.5.4. Database.php....14
2.5.5. A .env fájl a projekten belül....15
2.5.6. HTTP-protokoll....15
2.5.7. Laravel Sanctum....16
2.5.8. Adminisztrátori grafikus kezelőfelület....17
2.5.9. MySQL....25
2.6. Adatbázismodell....25

2.6.1. Allergens tábla 27
2.6.2. Categories tábla 28
2.6.3. Desks tábla 28
2.6.4. Event_logs tábla 28
2.6.5. Images tábla 29
2.6.6. Ingredients tábla 30
2.6.7. Ingredient_allergen kapcsolati tábla 30
2.6.8. Products tábla 30
2.6.9. Product_ingredient kapcsolati tábla 31
2.6.10. Product_purchase kapcsolati tábla 32
2.6.11. Product_user kapcsolati tábla 32
2.6.12. Purchases tábla 33
2.6.13. Reservations tábla 34
2.6.14. Users tábla 35
3. Projekt tesztelése 36
3.1. Playwright - End To End tesztek 36
3.2. ThunderClient - API tesztek 37
3.3. Swagger UI 38
4. Felhasználói dokumentáció 40
4.1. Felhasználói fiók 40
4.1.1. Regisztráció 40
4.1.2. Bejelentkezés 41
4.1.3. Vendég felhasználók 43
4.1.4. Regisztrált és bejelentkezett felhasználok számára elérhető funkciók 44
4.1.5. Beállítások oldal 45

4.1.6. Kedvenc ételek és italok oldal....47

4.1.7. Ételek....47

4.1.8. Italok....48

4.1.9. Foglalások oldal....49

4.1.10. Főoldal....50

4.1.11. Asztalfoglalás oldal....51

4.1.12. Kínálatunk oldal....52

4.1.13. Termék részletei oldal....54

4.1.14. Kosár oldal....56

5. Összefoglalás....58

6. Idézett forrásmunkák....61

1. Bevezetés

A Menu Genius egy ételrendelő program elsősorban, amely lehetőséget biztosít a felhasználó számára, hogy előre tudjon asztalt foglalni saját magának, illetve a vele érkező személyeknek egyaránt. Fontos szempont volt a projekt tematikájának kiválasztását illetően, hogy sok embernek nyújtson egyszerű és korszerű megoldást. Sok étteremben, főként franchise-okban fellelhető már ez az ötlet, viszont azok az adott cégnek a saját maguk által fejlesztett egyedi, zárt rendszerek. Amit mi készítettünk az egy „template", azaz egy sablon, amelyet az adott étteremnek a saját ételeivel, megjelenésével és egyéb paramétereivel egyetemben vagyunk képesek átalakítani.

Az általunk készített programnak az elsődleges célja, hogy bárki számára átlátható és könnyen kezelhető felületet biztosítson, az asztalnak a lefoglalásától kezdve, egészen az allergének böngészéséig és szűréséig. Mindenki számára elérhető az asztalfoglalás és böngészés funkció is. Így aki csak hirtelen pillantást szeretne vetni az adott étteremben levő kínálatra, az regisztráció nélkül is gond nélkül megteheti, ahogyan akár egy asztalnak a lefoglalását is egyszerűen megoldhatja mindenféle előzetes regisztrációt, illetve bejelentkezést „átugorva".

A könnyű használhatóságot rendkívüli mértékben segíti, hogy a felhasználói kezelőfelület letisztult és könnyen értelmezhető minden egyén számára. Felül a navigációs sávon vannak az oldalon való navigációhoz szükséges gombok nagyrésze. Ide tartozik a regisztrációs gomb, és a bejelentkezést követően a felhasználó saját adatihoz irányító gomb. Továbbá olyan beépített eszközök segítik a zökkenőmentes kezelést, mint például az ételkategória szerinti keresés, valamit allergének által szűrhető böngészés.

A MenuGenius letöltés nélkül elérhető. A program webes technológiára épül, így amennyiben az okoseszköz rendelkezik modern böngészővel, a program használható. Gyengébb rendszereken is gond nélkül futtatható és semmilyen felhasználói élmény romlás nem tapasztalható.

Kiemelt szerepet kapott a projekt megvalósításának során, hogy naprakész technológiákat használjunk, ezzel is fejlesztve a saját tudásunkat a jövőt illetően a

piacon. A választásunk - a kliensoldali részt illetően - az Angular-ra esett. Az Angular egy JavaScript alapú keretrendszer, amelyet világszerte rengetegen használnak a fejlesztéseik során. Jelenleg a 2. legnépszerűbb webfejlesztői környezet. Szerver oldalon a világ egyik legkeresetteb fejlesztői környezetére, a Laravel-re hagyatkoztunk. Előszeretettel alkalmaznak startup-ok, kis- és nagyvállalatok is Laravel fejlesztőket, az igény erre a tudásra évről-évre nő. 743.470 oldal készült ennek a technológiának a segítségével 2023-ban.

Sok új dologgal találkoztunk a projekt megvalósításának során, betekintést nyertünk, hogy hogyan is kell adatbázist felépíteni egy weboldalhoz, ezekhez API erőforrásokat kell kötni, vagyis a szerveroldallal összekapcsolni. Az Angular-nak köszönhetően - fejlesztői oldalról is - egy jól áttekinthető, könnyen értelmezhető és sok komponensre bontott eredményt voltunk képesek létrehozni.

Több ötletet is merítettünk az elkészítés során, jelenleg is üzemelő nagyobb étterem láncoknak weboldalaikról. (megjelenés struktúrája, ételrendelés menete, asztalfoglalás és különböző szolgáltatások)

A MenuGenius weboldal forráskódjának elérhetősége:

https://github.com/Celinayy/menuGenius

Bejelentkezéshez szükséges adatok:

User login web:

- user@gmail.com
- password

Admin login web:

- menugenius@gmail.com
- password

Admin login az admin felületen:

- admin
- password

2. Fejlesztői dokumentáció

2.1. Munkamegosztás

2.1.1. Feladatok elosztása

A MenuGenius projektet első körben 3 fős csapattal terveztük elkezdeni, de végül 2 fővel mentünk tovább. A projekt legelején döntöttünk arról, hogy ki milyen szerepet fog vállalni a fejlesztés során. Letenyei Szilárd a backendért és az adatbázisért felelt, illetve az admin felületet készítette el. A backend Laravel-ben készült, amely egy PHP keretrendszer. Az adatbázis egy MySQL alapú relációs adatbázis, és a dbForge segítségével tudtuk vizuálisan is nyomon követni az adatokat benne. Vangel Celina Hanna felelt a frontendért, illetve a dokumentációért a projekt során. A frontendet az Angular keretrendszer 16-os verziójával valósítottuk meg. Különböző előzetes dizájn terveket szerkesztette Vangel Celina Hanna.

A használt grafikai szoftverek:

- Paint 3D
- GIMP
- Wireframe

2.2. Verziókezelés

Párhuzamosan dolgoztunk ketten a projekt során. A munkánkat egy GitHub nevezetű verziókezelő szolgáltatással tudtuk zökkenőmentesen végezni. A GitHub segítségével figyelemmel tudtuk kísérni a projekt fejlődését és a módosításokat is könnyen nyomon lehetett követni.

A verziókezelés fogalma: *„A verziókezelés alatt több verzióval rendelkező adatok kezelését értjük.”*. A verziókezelő programok a fejlesztendő dokumentumok, forráskódok, tervek és egyéb olyan adatok verzióinak kezelésére szolgálnak, amelyeken egyidejűleg számos fejlesztő dolgozik. A fájlok módosulását naplózza a

rendszer. A nyilvántartás tartalmazza a módosító nevét, a módosítás idejét és egy „commit” üzenetet, amely egy összefoglaló leírás a módosításról. Megjegyzendő, hogy egy egyszemélyes fejlesztés során is hatalmas segítséget jelent egy verziókezelő szoftver használata.

A piacon több verziókezelő szoftver áll rendelkezésünkre, amelyek bizonyos funkciókban eltérnek, de az alapműködés és az alapfogalmak megegyeznek.

A GitHub segítségével lehetőség van több modul egyidejű fejlesztésére is, amelyeket „merge” funkció segítségével közös „branch” alá tudtunk hozni.

2.3. Frontend

A frontend feladata az adatok megjelenítése, befogadása a felhasználó (ritkábban akár egy másik rendszer) felől. A programoknak, weboldalaknak az a része, amely a felhasználóval közvetlen kapcsolatban van.

A frontend járul hozzá a felhasználó élményéhez elsősorban. A fő cél frontend oldalról, hogy könnyen lehessen kezelni, esztétikus legyen és gond nélkül át tudja látni a felhasználó a megjelenített adatokat. Ezen séma mentén készült el a MenuGenius weboldal is.

2.3.1. Dizájn tervek

Első lépés a weboldal elkészítése során a tervezés volt.

A használt grafikai szoftverek:

- Paint 3D
- GIMP
- Wireframe

Vázlatokat és rétegeket készítettünk elsősorban, hogy hogyan képzeljük el a weboldal különböző komponenseit. (1. számú melléklet) Ezeket a folyamatokat követően kezdtük meg a programozást, a dizájn tervek mellett az adatbázis is már a kezdeteken elején kidolgozásra került nagyrészt.

A weboldalon látható logó kidolgozására a NameCheap nevű oldalt választottuk, mivel ez ingyenes és az általunk megadott paraméterekkel számos variációt tudott ajánlani, amelyek közül a jelenlegire esett a választásunk. Ez a logó az adott étteremnek megfelelően változik.

1. ábra: Jelenlegi logó a főoldalon

2.4. Használt technológiák

2.4.1. Angular

Az Angular egy webalkalmazás-keretrendszer, amelyet a Google fejleszt. TypeScript programozási nyelv alapú, ingyenes és nyílt forráskód segítségével érhető el a rendszer, ami meglehetősen nagy népszerűségnek örvend. Az Angular alkalmazások felépítésében az MVC, vagyis Model View Controller játssza a fő szerepet. A HTML fájl tehát maga a nézet, a modellek és a kontrollerek pedig JavaScript használatával készülnek el.

Ez a frontend framework a HTML eszköztárát kiterjeszti és kiegészíti, az alkalmazás felépítésében helyet kapó elemek pedig követhetően elkülönülnek.

Ezek mellett az Angular keretrendszer alapjához különféle modulok csatolhatók, amelyek az eszköztár bővítését szolgálják, de természetesen új, saját kiegészítéseket is lehet írni és hozzákapcsolni a rendszerhez.

Az Angular előnyei összesítve:

- Típushibákra figyelmeztet
- Meghatározott keretek között zajlik a kódolás, mivel kötött felépítésű
- Jól szeparált fájlrendszerrel rendelkezik, egy adott komponens részekre van bontva, remekül strukturált a jobb átláthatóság és érthetőség szempontjából (üzleti logika, CSS, template)
- Komponensek és modulok a fő alkotóelemei, így könnyen módosítható

- A központi hibakezelés is biztosított az *Interceptor* által, az összes szerverről érkező hibaüzenet általános kezeléssel bír. (422 validációs, 403 autentikációs)

Nagy mértékben az is befolyásolta a választást frontend oldalról, hogy a tanórán ezt tanultuk, valamint a jó dokumentáltsága sem volt utolsó tényező.

2.4.2. A projekten belül használt „service”-ek

A „service”-eket komponensekhez használjuk fel. Remek lehetőség a „service”-be kiszervezni a tényleges logikákat, mivel ezeket újra és újra fel lehet használni a különböző komponensekben, amennyiben szükség van rájuk. Ez preferencia kérdése, de így jobban elválnak a dolgok az alkalmazásokon belül és sokkal tisztább, értelmezhetőbb kódot kapunk. Ennek nyomán a projekt ezen logikája több „service”-be lett végül kiszervezve, amelyeket alább megtekinthetünk:

- auth.service.ts

Itt történnek a felhasználó azonosításával kapcsolatos logikák, mint például regisztráció, bejelentkezés vagy akár az adatainak a frissítése.

```
75    public update(data: Partial<{
76      email: string,
77      phone: string,
78      password: string,
79      current_password: string,
80      password_confirmation: string,
81    }>) {
82      return this.connection.put<{ token: string }>(`${this.url}/user`, data,
83        {
84          headers: { Authorization: `Bearer ${localStorage.getItem("token")}` }
85        })
86    }
87
88    public deleteUser(current_password: string) {
89      return this.connection.delete(`${this.url}/user/${this.user!.id}`, {
90        headers: { Authorization: `Bearer ${localStorage.getItem("token")}` },
91        body: { current_password}
92      })
93    }
94  }
```

2. ábra Frissítés és felhasználó törlése a projekten belül

- cart.service.ts

Ezen „service”-en belül kezeljük a *„Kosárral”* kapcsolatos tevékenységeket, mint a termék hozzáadása vagy eltávolítása.

```
19  private persist() {
20    sessionStorage.setItem("cart", JSON.stringify(this.products));
21  }
22
23  public addProduct(product: ProductModel) {
24    this.products.push(product);
25    this.persist();
26  }
27
28  public removeItem(index: number) {
29    this.products.splice(index, 1);
30    this.persist();
31  }
32
```

3. ábra Termék hozzáadása illetve eltávolítása a projekten belül

- categories.service.ts

Ez a „service” felel a kategóriák eléréséért.

```
export class CategoriesService {

  private url = "http://localhost:8000/api/category"

  constructor(private connection: HttpClient) { }

  public getAllCategory() {
    return this.connection.get<CategoryModel[]>(this.url)
  }
}
```

4. ábra Kategóriák elérése a projekten belül

- favourite.service.ts

Itt található az összes *„Kedvencek”* oldalhoz tartozó interakciók, mint például a kedvenc ételekhez adás, eltávolítás vagy akár csak azon termékek megjelenítése, amelyeket a felhasználó megjelölt, mint *„Kedvenc Termék”*.

```
22  private loadFavourites() {
23    return this.connection.get<{favorites: ProductModel[]}>(`${this.url}/product/userFavorites`, {
24      headers: {
25        "Authorization": `Bearer ${localStorage.getItem("token")}`,
26      },
27    })
28  }
29
30  public addToFavourite(product: ProductModel) {
31    this.favProducIds.push(product.id)
32
33    this.connection.post<{ message: string }>(`${this.url}/product/${product.id}/addToFavorites`, {}, {
34      headers: {
35        "Authorization": `Bearer ${localStorage.getItem("token")}`,
36      },
37    }).subscribe((result) => {
38      this.toast.success(result.message);
39    });
40  }
```

5. ábra Kedvenc termékek betöltése és hozzáadása a projekten belül

- product.service.ts

A „product.service.ts” felel a termékekkel kapcsolatos logikáért, mint akár a külön étel vagy ital listázása.

```
22  public listProducts() {
23    return this.connection.get<ProductModel[]>(this.url)
24  }
25
26  public listDrinks() {
27    return this.products.filter((product) => {
28      return !product.is_food
29    })
30  }
31        You, 2 weeks ago • Uncommitted changes
32  public listFoods() {
33    return this.products.filter((product) => {
34      return product.is_food
35    })
36  }
```

6. ábra Összes termék, italok illetve ételek listázása a projekten belül

- reservation.service.ts

Ez a „service” felelős a felhasználók által leadott asztalfoglalásoknak a tényleges megvalósításáért, illetve a felhasználó le is tudja kérni a korábbi foglalásainak a részletes információit.

```
15  public createReservation(data: {
16    phone: string;
17    name: string;
18    number_of_guests: number;
19    date: Date;
20    check_in_datetime: string;
21    check_out_datetime: string;
22    comment: string;        You, last month • feat: create reservation as unaut
23  }) {
24    const token = localStorage.getItem("token");
25
26    return this.connection.post(`${this.url}`, {
27      name: data.name,
28      phone: data.phone,
29      number_of_guests: data.number_of_guests,
30      comment: data.comment,
31      checkin_date: this.formatDateTime(data.date, data.check_in_datetime),
32      checkout_date: this.formatDateTime(data.date, data.check_out_datetime)
33    }, {
34      headers: token ? {
35        "Authorization": `Bearer ${token}`,
36      } : {
37
38      },
39    })
40  }
```

7. ábra Foglalás létrehozása a projekten belül

- user.service.ts

A jelenleg bejelentkezett felhasználó adatait kérdezi le.

```
8   export class UserService {
9
10    constructor(private connection: HttpClient) { }
11
12    public get() {
13      return this.connection.get<UserModel>("http://localhost:8000/api/user",
14      {headers: {Authorization: `Bearer ${localStorage.getItem("token")}`}})
15    }
16  }
```

8. ábra Felhasználó elérése a projekten belül

2.4.3. Navigáció a projekten belül

A felhasználó számára az „app-routing.module.ts” fájl biztosítja a navigációt az oldalak között. Annak érdekében, hogy az *Angular router* működőképes legyen, legalább kettő komponenst tartalmaznia kell a programnak, hogy lehetséges legyen a komponensek közötti oda-vissza navigálás. Alább láthatóak a projektben felhasznált összes útvonal.

```
17  const routes: Routes = [
18    {path: "", component: MainWindowComponent},
19    {path: "tableReservation", component: TableReservationWindowComponent},
20    {path: "profile", component: UserProfileComponent},
21    {path: "profile/settings", component: SettingsWindowComponent},
22    {path: "profile/history", component: HistoryWindowComponent},
23    {path: "profile/favFood", component: FavouriteFoodWindowComponent},
24    {path: "products", component: ProductsWindowComponent},
25    {path: "products/:id", component: ProductsDetailsWindowComponent},
26    {path: "cart", component: CartWindowComponent},
27    {path: "profile/foods", component: FoodsWindowComponent},
28    {path: "profile/drinks", component: DrinksWindowComponent},
29    {path: "stripe/payment/success", component: SuccesfullPaymentWindowComponent},
30    {path: "stripe/payment/cancel", component: CancelPaymentWindowComponent},
31
32  ];
```

9. ábra Útvonalak a projekten belül

2.4.4. Angular és Bootstrap 5.3

Az előre definiált stílusokhoz a Bootstrap 5.3-as verzióját használtuk, mivel ez egy könnyen átlátható és kezelhető felületet tud biztosítani a felhasználóknak és a dokumentáltsága is megfelelő volt számunkra.

2.4.5. Reszponzivitás

A kész projekt kiszolgálja az alapvető elvárást, a reszponzivitást, ennek megfelelően minden eszközön használható, bármilyen megjelenítő eszközt támogat. A Bootstgrap Grid System segítségével tudtuk ezt elérni a kész weboldal esetében.

2.4.6. Főoldal

A főoldal célja, hogy meghozza a felhasználó kedvét és étvágyát, ezért a rövid leírás mellett egyből látható egy Bootstrap 5.3 által biztosított Carousel, amelyben a termékek fotói láthatóak véletlenszerűen megjelenítve, a felhasználó egy ilyen megjelenítésnek köszönhetően tízszer nagyobb eséllyel kattint a látott termékre.

2.5. Backend

2.5.1. Használt technológiák

A backendet a nyílt forráskódú, PHP alapú Laravel keretrendszerrel valósítottuk meg. Az adatbázis felépítéséhez a MySQL-t használtunk. Különböző engedélyezések kezelésére pedig a Sanctum megoldásra esett a választásunk.

2.5.2. Laravel

A Laravel egy ingyenes és nyílt forráskódú PHP webes keretrendszer, amelyet Taylor Otwell fejlesztette ki. Az első verziója 2011-ben jelent meg, és azóta számos verziófrissítésen és fejlesztésen esett át. A projekt fejlesztése során a legfrissebb, Laravel 10-es verziójával dolgoztunk.

Az egyik legnépszerűbb, legbiztonságosabb és legkönnyebb architektúra a Laravel. Rengeteg alapvető és ismétlődő funkciót old meg a fejlesztő helyett. Akár kezdőként is használható és gyorsan megtanulható, mivel kellő támpontokat tud adni, ahhoz, hogy jól strukturált kódot kapjunk a beépített terminál (Artisan) segítségével.

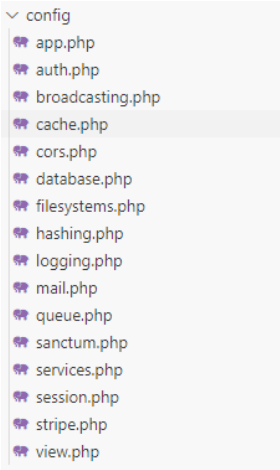
A Laravel előnyei és főbb jellemzői:

- Többnyelvű alkalmazások készítésére is alkalmas, így azok könnyen lokalizálhatóak
- Remekül testreszabható és bővíthető, így az egyedi igényekhez is alkalmazkodni tud
- Optimalizált kódbázissal bír, így nagy forgalmú honlapokhoz is tökéletes
- Jól dokumentált keretrendszer, számos segítséget találni az interneten hozzá

- Rengeteg beépített biztonsági funkcióval rendelkezik, amelyek segítenek megvédeni az alkalmazásokat a támadásoktól.
- Nagyon könnyen és gyorsan lehet vele dolgozni.
- Könnyen bővíthető és fejleszthető

2.5.3. Laravel config mappa és a .env fájl a projekten belül

A config mappán belül találhatóak azok a fájlok, amelyek a backend konfigurálását végzik. A mappa különféle konfigurációs fájlokat tartalmaz, amelyek megkönnyítik az olyan tevékenységek elvégzését, mint például az adatbázis illesztőprogram vagy a külső szolgáltatások meghatározását. A .env fájl tartalmazza a legfontosabb, helyi környezeti beállításokat. Az alábbi alfejezetekben röviden, a teljesség igénye nélkül bemutatunk egyes konfigurációs fájlokat.



10. ábra A config mappában lévő fájlok a projektben

2.5.4. Database.php

A Laravel segítségével rendkívül egyszerű az adatbázishoz való kapcsolódás és a lekérdezés egyaránt. Az adatbázis konfigurációs fájlja a *database.php.* Ebben a fájlban van eltárolva az összes adatbázis kapcsolat. A Laravel összesen négy adatbázisrendszert támogat: SQLite, MySQL, Postgres, SQL Server. Mi MySQL-t használunk és ebben a fájlban ezt is állítottuk be alapértelmezett kapcsolatnak.

```
'default' => env('DB_CONNECTION', 'mysql'),
```

11. ábra Adatbázis kapcsolat a database.php fájlban

Láthatjuk az env() segédmetódust, ami kapcsolódik a .env fájlhoz, azon belül pedig a metódus első paraméteréhez. Ha ez meg van határozva a .env fájlban, akkor onnan veszi az értéket. Ha nincs meghatározva, akkor az env() metódus második paraméterét veszi alapértelmezettnek, amit az első paraméter szerinti változóba állít be.

2.5.5. A .env fájl a projekten belül

A .env fájlon belül meghatároztuk, hogy milyen adatbázist használunk és magának az adatbázisnak a nevét is itt adtuk meg. Ezek az adatbázis specifikus beállítások.

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=menugenius
DB_USERNAME=root
DB_PASSWORD=
```

12. ábra Adatbázis konfigurálása a .env fájlban

Az alábbi kódunkban láthatjuk, hogy egy MySQL adatbázis-kezelőhöz csatlakozunk, vagyis a MySQL driver-t használjuk, az adatbázis szerverünk IP címe 127.0.0.1, vagyis localhost. A port a MySQL alapértelmezett portja, az adatbázis neve menugenius, a MySQL motor eléréséhez pedig a root felhasználónév van beállítva, illetve nincs jelszó.

2.5.6. HTTP-protokoll

A *http* protokoll olyan metódusokat határoz meg, amelyek szemantikai jelentést rendelnek a kérésekhez. Ilyen gyakrabban használt metódusok:

GET: lekéri az erőforrás adott reprezentációját a megadott URL-n keresztül. A válaszüzenetet a body-ban kapjuk meg. Lehetőség van információ küldésére is, amikor az erőforrás számára adatokat tudunk átadni.

POST: új erőforrás létrehozására szolgál az adott URL-n, ilyenkor a body-nak tartalmaznia kell a létrehozandó objektum adatait.

PUT: új erőforrás létrehozására, illetve meglévő frissítésére alkalmas.

DELETE: a megadott URL-n keresztül a megnevezett erőforrást törli.

PATCH: egy erőforrás részleges frissítését hajtja végre. A body tartalmazza az erőforrásra alkalmazni kívánt módosításokat.

```
Route::get("allergen", [AllergenController::class, 'index']);
Route::get("category", [CategoryController::class, 'index']);
Route::get("ingredient", [IngredientController::class, 'index']);
Route::get("desk", [DeskController::class, 'index']);

Route::get('/charge', function () {
    return view('charge');
});

Route::get('/', [StripeController::class, 'index'])->name(name:'index');
Route::post('/checkout', [StripeController::class, 'checkout'])->name(name:'checkout');
```

13. ábra Néhány HTTP metódus a kódból

2.5.7. Laravel Sanctum

A fejlesztés során figyelnünk kellett arra, hogy a rendszer képes legyen felügyelni a kiállított token-ek használatát.

A Laravel Sanctum egy nagyon egyszerű hitelesítési rendszert biztosít az egyoldalas, mobilalkalmazások és egyszerű, token alapú API-k számára. A Sanctum az alkalmazás minden felhasználójának biztosítja, hogy több API-token jöjjön létre a fiókjához. Ezeket a token-eket olyan hatókörrel ruháztuk fel, amelyek meghatározzák, hogy azok milyen műveleteket hajthatnak végre.

A Laravel Sanctum egyszerűsége abban rejlik többek között, hogy egyetlen adatbázisban tárolja a felhasználói API-token-eket és hitelesíti a bejövő http kéréseket az Authorization fejlécen keresztül, amelyeknek tartalmaznia kell egy érvényes API token-t. Továbbá a Sanctum egyszerű használatát erősíti az is, hogy az egyoldalas alkalmazások (SPA) hitelesítésekor, amelyek a Laravel által üzemeltetett API-val kommunikálnak, nem használ semmilyen token-t. Ugyanis a Sanctum a Laravel beépített cookie alapú hitelesítési szolgáltatásait használja. A Sanctum általában Laravel web hitelesítési védelmét használja, amely biztosítja a CSRF védelmet, a munkamenet hitelesítést, valamint védelmet nyújt az XSS-en keresztüli hitelesítéskor fellépő kiszivárgás ellen.

A Sanctum csak akkor kísérli meg a hitelesítést cookie-val, ha a bejövő kérés a saját SPA frontend felől érkezik. Amikor a Sanctum megvizsgál egy bejövő http

kérelmet, először azt nézi, hogy van-e hitelesítési cookie és ha nincs, akkor megvizsgálja, hogy van-e Authorization érvényes API-token a fejlécben.

A Sanctum-ot használtuk SPA hitelesítésre is. Ez látható a Kernel.php fájlban, a middleware szoftvercsoportban.

```
'api' => [
    \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,
    \Illuminate\Routing\Middleware\ThrottleRequests::class.':api',
    \Illuminate\Routing\Middleware\SubstituteBindings::class,

],
```

14. ábra SPA hitelesítés

A projektben a Sanctum alapértelmezett migrációit használtuk.

Az API token-ek kiadásával hitelesítettük az alkalmazás API kéréseit. Amikor API token kérés érkezik, a token-nek szerepelni kell az Authorization fejlécben, mint Bearer token. Ez látszik a User.php fájlban, amikor token-eket adunk a weboldal felhasználóinak.

```
class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable, SoftDeletes;
```

15. ábra Token küldése felhasználónak

2.5.8. Adminisztrátori grafikus kezelőfelület

A **C#** (C Sharp) a Microsoft által fejlesztett, a .NET keretrendszeren belül használt objektumorientált programozási nyelv.

Az alkalmazás felhasználói felületét WPF segítségével hoztam létre. A **WPF (Windows Presentation Foundation)** egy olyan keretrendszer a Windows alkalmazásokhoz, amely lehetővé teszi a felhasználói felületek létrehozását. A XAML (eXtensible Application Markup Language) segítségével építhetünk felületeket, amelyeket a WPF C# kódban lehet vezérelni.

A **Database First** megközelítés egy adatbázis-orientált fejlesztési módszer, amelyben az adatbázis struktúrája először van definiálva, és a programkód ezen struktúrához igazodik. Ebben az esetben az adatbázis sémát először készítjük el, majd ebből generáljuk a C# osztályokat.

Az **Entity Framework (EF)** egy objektum-relációs leképező (ORM) keretrendszer a .NET keretrendszerhez, amely lehetővé teszi az adatbázis-objektumok közötti átjárást. Az EF segítségével könnyedén kezelhetjük az adatbázisból származó adatokat C# objektumokként, és egyszerűen elvégezheted a CRUD (Create, Read, Update, Delete) műveleteket.

A **Pomelo.EntityFrameworkCore.MySql** egy nyílt forráskódú .NET alapú MySQL adatbázis szolgáltató, amely az Entity Framework Core-hoz készült. Ennek segítségével lehetőségünk van az Entity Framework Core-t használni MySQL adatbázissal, így könnyedén kezelhetjük a MySQL adatbázisból származó adatokat.

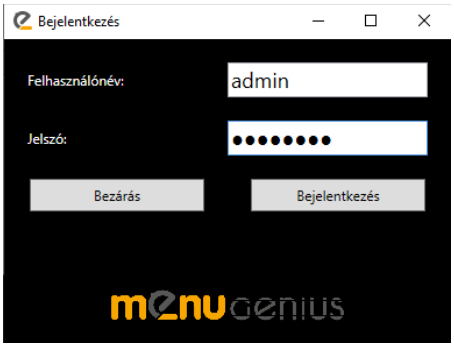
Ezeket a technológiákat kombinálva az alkalmazás a felhasználói felület megjelenítésére és az adatok kezelésére szolgál, lehetővé téve a felhasználók számára az adatok megtekintését, módosítását és törlését a WPF felületen keresztül, miközben az adatokat hatékonyan kezeljük az Entity Framework segítségével a MySQL adatbázisban.

A szoftver használatával karbantartható a MenuGenius-t kiszolgáló adatbázis. Alkalmas az adatbázisba új termékeket, kategóriákat, alapanyagokat, stb felvenni, törölni, módosítani. Segítségével követhetők és szerkeszthetők az étteremben leadott foglalások, rendelések. Valamit lehetőség van benne a program eseménynaplójának megnézésére és a felhasználók felvételére, törlésére módosítására is.

Főbb funkciói:

- **Bejelentkezés**

A program kezelőfelületét egy beléptető ablakon keresztül érhetjük el, ahol meg kell adnunk egy admin jogosultságú felhasználónevet és a hozzá kapcsolódó jelszót.



16. ábra: MenuGenius GUI Bejelentkezés

- **Rendelések fül**

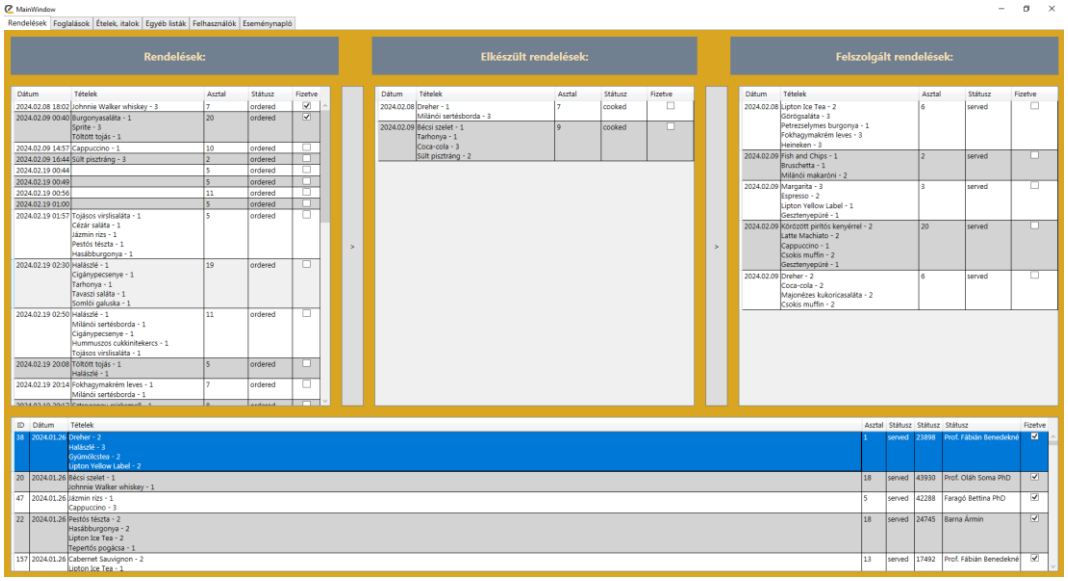
Ezen az oldalon lehet nyomon követni az étteremben leadott rendeléseket.

Az első oszlopba kerülnek be a honlapon leadott rendelések időrendi sorrendben. A szakácsok itt láthatják mit kell elkészíteniük. A első és a második lista közötti gombra kattintva a „Rendelések” oszlopban kiválasztott rendelés státusza megváltozik és átkerül az „Elkészült rendelések” oszlopba, ahol a pincérek láthatják mely ételek készültek el a konyhán és készek a felszolgálásra.

Amennyiben fel is szolgálták az ételt a második és harmadik lista közötti gombbal áttehetik a rendelést a „Felszolgált rendelések” oszlopba, ahol azok a rendelések láthatók, amelyek már fel lettek szolgálva, de még nem lettek kifizetve. Amennyiben egy étel nem lett kifizetve a honlapon a rendeléskor, akkor a pincérnél kell rendezni a számlát.

A rendelés akkor is átkerül az alsó datagrid-be, ha a „Felszolgált rendelések” között kipipáljuk a rendeléshez tartozó „Fizetve” checkboxot.

A „Rendelések” fülön alul található datagrid elemben vannak felsorolva a korábbi rendelések időrendi sorrendben. Csak akkor kerülhet ide egy rendelés, ha már fel lett szolgálva és ki lett fizetve.



17. ábra: MenuGenius GUI Rendelések fül

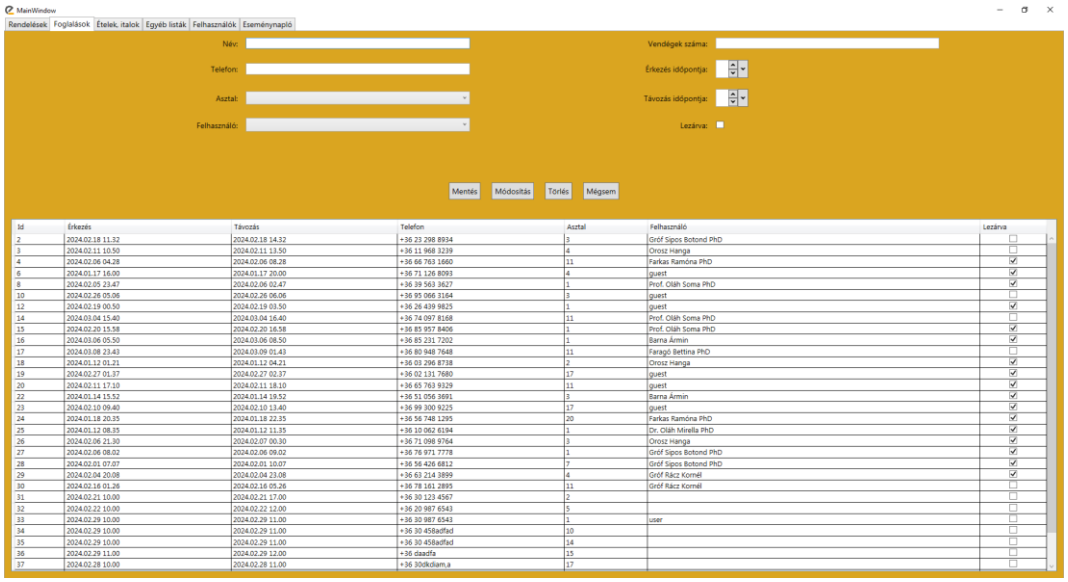
- **Foglalások fül**

A „Foglalások” fül alatt találhatóak az étteremben leadott asztalfoglalási információk.

Az oldal alján látható datagridben időrendi sorrendben vannak felsorolva. Kiválasztva egy elemet a kiválasztott elem tulajdonságai jelennek meg az oldal tetején található szövegdobozokban.

A gombok funkciói:

1. Mentés – Új foglalás létrehozása és mentése az adatbázisba
2. Módosítás – A datagridben kiválasztott foglalás adatait módosítja és menti a változásokat az adatbázisba
3. Töröl – A datagridben kiválasztott foglalás törlése.
4. Mégsem – Törli a kiválasztást a datagridben, és a foglalás adatait megjelenítő felületi elemek tartalmát is.



Id	Érkezés	Távozás	Telefon	Asztal	Felhasználó	Lezárva
2	2024.02.18 11.32	2024.02.18 14.32	+36 23 298 8934	3	Gróf Sipos Botond PhD	☐
3	2024.02.11 10.50	2024.02.11 13.50	+36 11 968 3239	4	Orosz Hanga	☐
4	2024.02.06 04.28	2024.02.06 08.28	+36 66 763 1660	11	Farkas Ramóna PhD	☑
6	2024.01.17 16.00	2024.01.17 20.00	+36 71 126 8093	4	guest	☑
8	2024.02.05 23.47	2024.02.06 02.47	+36 39 563 3627	1	Prof. Oláh Soma PhD	☑
10	2024.02.26 05.06	2024.02.26 06.06	+36 95 066 3164	3	guest	☐
12	2024.02.19 00.50	2024.02.19 03.50	+36 26 439 9825	1	guest	☑
14	2024.03.04 15.40	2024.03.04 16.40	+36 74 097 8168	11	Prof. Oláh Soma PhD	☐
15	2024.02.20 15.58	2024.02.20 16.58	+36 85 957 8406	1	Prof. Oláh Soma PhD	☑
16	2024.03.06 05.50	2024.03.06 08.50	+36 85 231 7202	1	Barna Ármin	☑
17	2024.03.08 23.43	2024.03.09 01.43	+36 80 948 7648	11	Faragó Bettina PhD	☐
18	2024.01.12 01.21	2024.01.12 04.21	+36 03 296 8738	2	Orosz Hanga	☑
19	2024.02.27 01.37	2024.02.27 02.37	+36 02 131 7680	17	guest	☑
20	2024.02.11 17.10	2024.02.11 18.10	+36 65 763 9329	11	guest	☑
22	2024.01.14 15.52	2024.01.14 19.52	+36 51 056 3691	3	Barna Ármin	☑
23	2024.02.10 09.40	2024.02.10 13.40	+36 99 300 9225	17	guest	☑
24	2024.01.18 20.35	2024.01.18 22.35	+36 56 748 1295	20	Farkas Ramóna PhD	☑
25	2024.01.12 08.35	2024.01.12 11.35	+36 10 062 6194	1	Dr. Oláh Mirella PhD	☑
26	2024.02.06 21.30	2024.02.07 00.30	+36 71 098 9764	3	Orosz Hanga	☑
27	2024.02.06 08.02	2024.02.06 09.02	+36 76 971 7778	1	Gróf Sipos Botond PhD	☑
28	2024.02.01 07.07	2024.02.01 10.07	+36 56 426 6812	7	Gróf Sipos Botond PhD	☑
29	2024.02.04 20.08	2024.02.04 23.08	+36 63 214 3899	4	Gróf Rácz Kornél	☑
30	2024.02.16 01.26	2024.02.16 05.26	+36 78 161 2895	11	Gróf Rácz Kornél	☐
31	2024.02.21 10.00	2024.02.21 17.00	+36 30 123 4567	2		☐
32	2024.02.22 10.00	2024.02.22 12.00	+36 20 987 6543	5		☐
33	2024.02.29 10.00	2024.02.29 11.00	+36 30 987 6543	1	user	☐
34	2024.02.29 10.00	2024.02.29 11.00	+36 30 458adfad	10		☐
35	2024.02.29 10.00	2024.02.29 11.00	+36 30 458adfad	14		☐
36	2024.02.29 11.00	2024.02.29 12.00	+36 daadfa	15		☐
37	2024.02.28 10.00	2024.02.28 11.00	+36 30dkdiam,a	17		☐

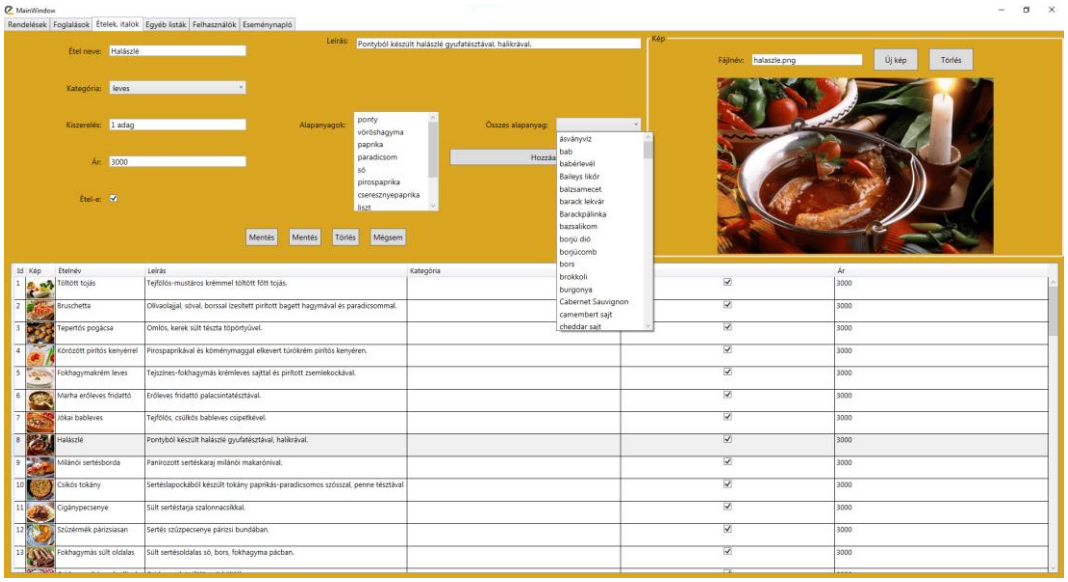
18. ábra MenuGenius GUI Foglalások fül

- **Ételek és italok fül**

Az „Ételek, italok” fülön találhatóak az étteremen levő kínálatok.

A gombok funkciói:

1. Mentés – Új termék létrehozása és mentése az adatbázisba
2. Módosítás – A datagridben kiválasztott termék adatait módosítja és menti a változásokat az adatbázisba
3. Töröl – A datagridben kiválasztott termék törlése.
4. Mégsem – Törli a kiválasztást a datagridben, és a termék adatait megjelenítő felületi elemek tartalmát is.
5. Van lehetőség az új termékhez hozzáadni az alapanyagokat is az „Összes alapanyag”-ból kiválasztva a kívánt összetevőt, majd a „Hozzáad” gombot megnyomva. A „Kép” groupboxban adhatunk hozzá képet is az új termékhez, vagy cserélhetjük le a már meglévő termék képét.



19. ábra MenuGenius GUI Ételek és italok fül

- **Egyéb listák**

Ezen a fülön láthatjuk az étteremben használt alapanyagokat, a kategóriákat, amelyekbe az étlapon található termékeket sorolhatjuk be, az allergének listáját, illetve az étterem asztalait.

Az alsó részen lévő datagridekben tekinthetőek meg az adatbázisban tárolt vonatkozó listák (ételkategóriák, alapanyagok, allergének, asztalok). A gombok itt is hasonlóan működnek, mint a program egyéb helyein. Minden adatbázistáblát megjelenítő részhez saját gombok vannak rendelve, melyekkel új tételt lehet menteni, meglévőt módosítani, törölni, illetve a mégsem gombok a kiválasztás megszüntetésére és a felületi elemek üressé tételére szolgálnak.

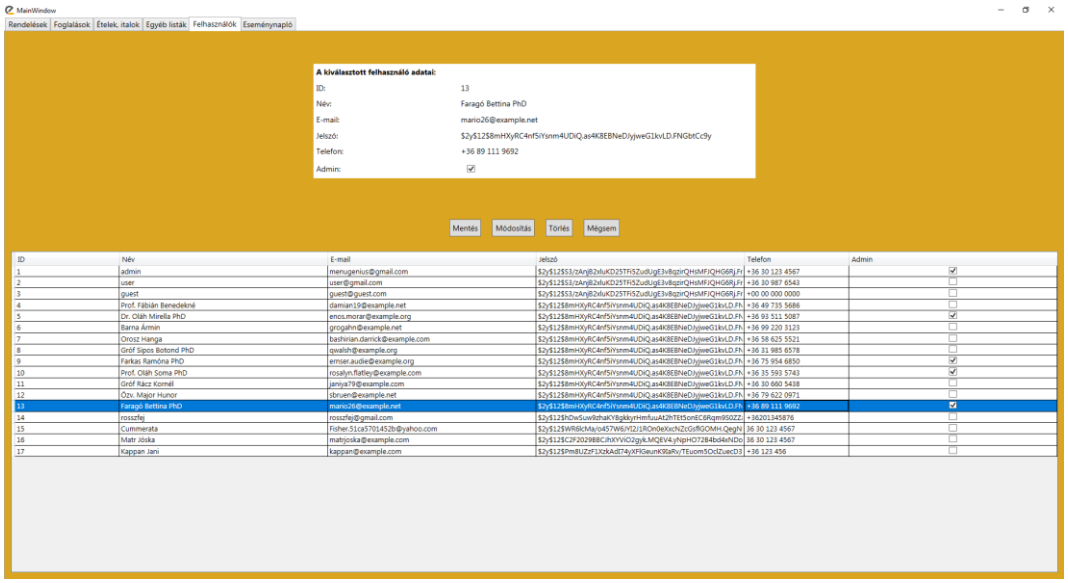
20. ábra MenuGenius GUI Egyéb listák fül

- **Felhasználók**

A „Felhaszánlók” fülön az étteremben regisztrált felhasználók (vendégek, alkalmazottak, adminisztrátorok) vannak listázva.

A gombok funkciói:

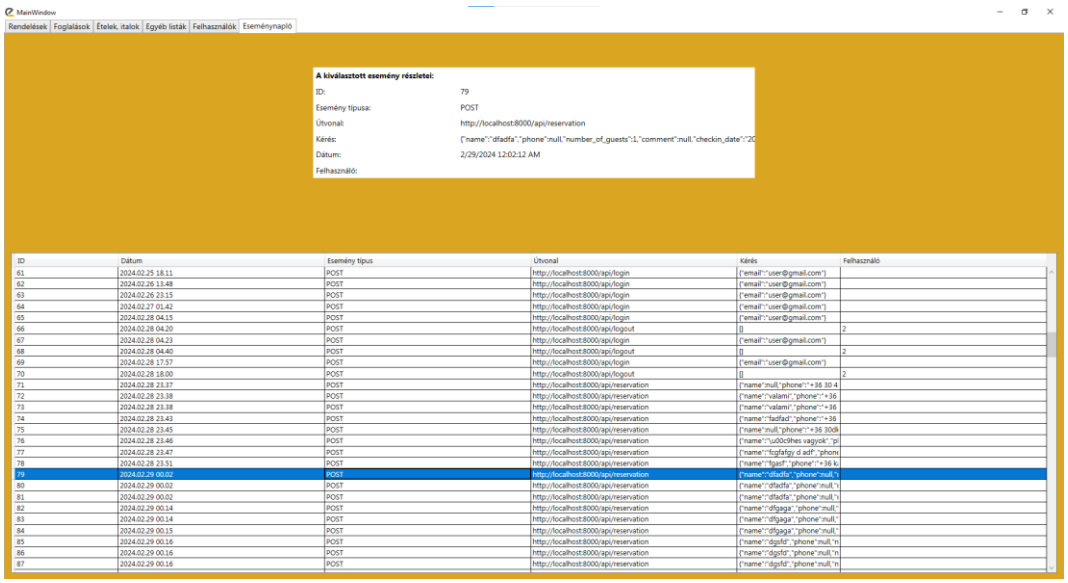
1. Mentés – Új felhasználó létrehozása és mentése az adatbázisba
2. Módosítás – A datagridben kiválasztott felhasználó adatait módosítja és menti a változásokat az adatbázisba
3. Töröl – A datagridben kiválasztott felhasználó törlése.
4. Mégsem – Törli a kiválasztást a datagridben, és a felhasználó adatait megjelenítő felületi elemek tartalmát is.



21. ábra MenuGenius GUI Felhasználók fül

- **Eseménynapló**

Itt látható a felhasználók által kiváltott összes tevékenység az oldalon belül.



22. ábra MenuGenius GUI Eseménynapló fül

Hardverkövetelmények:

- Processor: x86 or x64
- RAM: 512 MB (minimum), 1 GB (recommended)

Operációs rendszer:

- Windows XP
- Windows Vista SP2
- Windows 7 SP1
- Windows 8, 8.1
- Windows 10
- Windows 11

Fejlesztői környezet:

- Microsoft Visual Studio 2022
- .NET Framework 4.0/4.5/4.5.1/4.6
- .NET 6.0/.NET 7.0/.Net 8.0

2.5.9. MySQL

A MySQL egy nyílt forráskódú, többszálú, SQL (Structured Query Language) alapú relációs adatbázis-kezelő szerver. A szoftvert eredetileg a svéd MySQL AB cég fejlesztette ki. A MySQL az egyik legelterjedtebb adatbázis-kezelő, költséghatékony és egyszerűen beállítható megoldást ad dinamikus webhelyek szolgáltatására. A relációs adatbázis az egy csoportba tartozó adatokat táblákba rendezi, az adatokat mezőkben tárolja. Ahhoz, hogy az egyes táblák adatai között létrejöjjön a kapcsolat idegen kulcsokat (foreign key) kell használnunk. Tulajdonképpen az idegen kulcsok egy másik tábla elsődleges kulcsai (primary key).[1]

A MySQL adatbázis előnyei:

- Magas biztonsági jogosultságokkal rendelkezik.
- Működéséhez nem szükséges nagy mennyiségű erőforrás
- Nagy stabilitás, mivel rétegeket és modulokat foglal magába
- Egyszerű adatimportálás és exportálás
- Nagy mennyiségű adatkezeléssel bír

2.6. Adatbázismodell

[1] https://www.awh.hu/kb/webtarhely/mi-az-a-mysql 2024.02.09.

A dizájnt megelőzően, már a projekt első körében az adatbázis megtervezésével kezdtük. Ez egy iterációs folyamat, ahol a legelső lépés az üzleti igények felmérése és megértése, ezek után több modellt hoztunk létre. A projekt során ezek a modellek folyamatos átalakításon, finomításon és esetekben elvetésre szorultak. Miután újra átgondoltuk az üzleti igényeket, megkapta a végleges formáját az üzleti logika.

Az üzleti logika megértése után, a logikai modell elkészítésére törekedtünk. Az adat mezők típusainak meghatározására itt került sor.

A mezők típusai:

- BIGINT
- DECIMAL
- VARCHAR
- TIMESTAMP
- INT
- TEXT
- DATETIME
- DATE
- LONGTEXT
- TINYINT
- ENUM
- LONGLOB

A foglalások állapotaihoz létrehoztunk egy ENUM-ot, amelynek előre meghatározott állapotai lehetnek a projekten belül: *„ordered”, „served”, „cooked”*

Az entitások kapcsolatát is meg kellett határoznunk, melyeket az alábbi táblázatban ábrázolunk is a fő kapcsolati formák alapján:

egy-az-egyhez (1-1)	egy-a-többhöz (1-N)	több-a-többhöz (N-M)
Egy egyedtípus egy egyedéhez egy másik egyedtípus csak egyetlen egyede kapcsolódhat és fordítva is igaz. (osztály – osztályfőnök)	Egy egyedtípus egy egyedéhez egy másik egyedtípus több egyede is kapcsolódhat, de fordítva NEM igaz. (osztály – tanuló)	Egy egyedtípus egy egyedéhez egy másik egyedtípus több egyede is kapcsolódhat, de fordítva is igaz. (osztály – tanár)

Ekkor létrejöttek a kapcsolati táblák, melyek definiálják a kapcsolatokat az entitások között. Itt már egy teljes struktúra bontakozik ki, amely láttatja, leképezi az üzleti logikát. A logikai modellben létrehozott mezőket valós adattárolásra alkalmas mezőkké definiáljuk, például: purchases adatmezőt lebontottuk valós vásárlási adatok tárolására alkalmas mezőkre: (dátum, teljes összeg, státusz, kinek a rendelése, melyik asztal stb).

Utolsó lépésként a fizikai modell létrehozása volt a feladatunk, amely adatbázis specifikus elemekkel bír és már az adatbázisban kerül megalkotásra. Ez már teljesen adatbázis függő. Ennek a szakasznak az eredménye a megvalósított adattáblák a kapcsolatokkal együtt.

Az adatok tárolására a MySQL adatbázist használtuk, ahol az adatok tárolása táblákban, mezőkben történt. Ezeket a táblákat a Laravel Eloquent nevű adatbázis leképezővel hoztuk létre a backend oldalon már megalkotott modellek alapján. A migrációt minden modell változásakor le tudtuk futtatni, így tudtuk az adatbázist folyamatosan frissíteni, így lett mindig napra kész az aktuális verzióhoz.

Az alábbi alfejezetekben bemutatásra kerülnek az adatbázis táblái a jellemzőkkel együtt: (teljes adatbázismodellt a 2. számú melléklet tartalmazza)

2.6.1. Allergens tábla

Az allergéneket tartalmazó tábla.

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- code
  - Típus: DECIMAL (8, 2)
  - Egyedi kód az allergénekhez.
- name
  - Típus: VARCHAR (255)
  - Az allergén neve.

- deleted_at
  - Típus: TIMESTAMP

2.6.2. Categories tábla

A kategóriákat tartalmazó tábla

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- name
  - Típus: VARCHAR (255)
  - A kategória neve.
- deleted_at
  - Típus: TIMESTAMP

2.6.3. Desks tábla

Az asztalokat tartalmazó tábla

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- number_of_seats
  - Típus: INT (11)
  - Az asztal ülőhelyének száma.
- deleted_at
  - Típus: TIMESTAMP

2.6.4. Event_logs tábla

A felhasználó által kiváltott különböző eseményeket tárolja

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- event_type
  - Típus: VARCHAR (255)
  - Az interakció típusát jelöli
- user_id
  - Típus: BIGINT (20)
- route
  - Típus: VARCHAR (255)
  - Az adott elérési útvonal
- body
  - Típus: TEXT
  - A küldött adat
- date_time
  - Típus: DATETIME
  - Az esemény időpontja
- deleted_at
  - Típus: TIMESTAMP

2.6.5. Images tábla

Az oldalon található összes fotót tárolja.

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- img_name
  - Típus: VARCHAR (255)
  - Fotó neve
- deleted_at
  - Típus: TIMESTAMP

- img_data
  - Típus: LONGLOB
  - A fotó adatait tartalmazza

2.6.6. Ingredients tábla

A hozzávalókat tartalmazza.

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- name
  - Típus: VARCHAR (255)
  - A hozzávaló neve
- deleted_at
  - Típus: TIMESTAMP

2.6.7. Ingredient_allergen kapcsolati tábla

Kapcsolatot biztosít a hozzávalók és az allergének között ez a tábla

Jellemzők:

- ingredient_id
  - Típus: BIGINT (20)
- allergen_id
  - Típus: BIGINT (20)
- deleted_at
  - Típus: TIMESTAMP
  - Típus: INT (11)

2.6.8. Products tábla

A termékeket tartalmazza.

Jellemzők:

- name
  - Típus: VARCHAR (255)
  - Termék neve
- description
  - Típus: VARCHAR (255)
  - A termékről való rövid leírás
- category_id
  - Típus: BIGINT (20)
  - Category táblára mutató idegen kulcs (Foreign key)
- packing
  - Típus: VARCHAR (255)
  - Az adott termék adagolása
- price
  - Típus: INT (11)
  - A termék ára
- is_food
  - Típus: TINYINT (1)
  - Étel vagy sem az adott termék
- image_id
  - Típus: BIGINT (20)
  - Images táblára mutató idegen kulcs (Foreign key)
- deleted_at
  - Típus: TIMESTAMP

2.6.9. Product_ingredient kapcsolati tábla

Kapcsolatot biztosít a termékek és a hozzávalók tábla között

Jellemzők:

- product_id
  - Típus: BIGINT (20)
  - Products táblára mutató idegen kulcs (Foreign key)
- ingredient_id

  - Típus: BIGINT (20)
  - Ingredients táblára mutató idegen kulcs (Foreign key)
- deleted_at
  - Típus: TIMESTAMP

2.6.10. Product_purchase kapcsolati tábla

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- product_id
  - Típus: BIGINT (20)
  - Products táblára mutató idegen kulcs (Foreign key)
- purchase_id
  - Típus: BIGINT (20)
  - Purchases táblára mutató idegen kulcs (Foreign key)
- quantity
  - Típus: INT (11)
  - Adott termék mennyiségét jelöli
- deleted_at
  - Típus: TIMESTAMP

2.6.11. Product_user kapcsolati tábla

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- product_id
  - Típus: BIGINT (20)
  - Products táblára mutató idegen kulcs (Foreign key)
- user_id

  - Típus: BIGINT (20)
  - User táblára mutató idegen kulcs (Foreign key)
- favorite
  - Típus: TINYINT (1)
  - Kedvenc termék vagy sem jelölésére szolgál
- stars (ez a projekten belül nincs használatban, de a jövőben való fejlődés során igénybe vehető)
  - Típus: INT (11)
  - A felhasználó által értékelve 1-5 skálán
- deleted_at
  - Típus: TIMESTAMP

2.6.12. Purchases tábla

Az oldalon végbemenő vásárlások tárolására alkalmazzuk a projekten belül.

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- date_time
  - Típus: DATETIME
  - A vásárlás időpontját jelöli
- total_pay
  - Típus: INT (11)
  - Az az érték, amelyben a vásárlás történt
- status
  - Típus: ENUM ('ordered', 'cooked', 'served')
  - Az adott étel státuszát jelöli
- paid
  - Típus: TINYINT (1)
  - A vásárlás fizetett vagy sem
- user_id

  - Típus: BIGINT (20)
  - User táblára mutató idegen kulcs (Foreign key)
- desk_id
  - Típus: BIGINT (20)
  - Desks táblára mutató idegen kulcs (Foreign key)
- deleted_at
  - Típus: TIMESTAMP

2.6.13. Reservations tábla

A foglalásokat tartalmazza ez a tábla.

Jellemzők:

- checkout_date
  - Típus: DATETIME
- name
  - Típus: VARCHAR (255)
  - A foglalásnál megadott nevet tárolja
- phone
  - Típus: VARCHAR (255)
  - A foglalásnál megadott telefonszámot tárolja
- desk_id
  - Típus: BIGINT (20)
  - Desks táblára mutató idegen kulcs (Foreign key)
- user_id
  - Típus: BIGINT (20)
  - Users táblára mutató idegen kulcs (Foreign key)
- closed
  - Típus: TINYINT (1)
  - A foglalás lezárult-e vagy sem
- comment
  - Típus: TEXT
  - A felhasználó által megadott megjegyzés a foglalás során

- deleted_at
  - Típus: TIMESTAMP

2.6.14. Users tábla

Jellemzők:

- id
  - Típus: BIGINT (20)
  - Automatikusan létrehozott azonosító (Primary Key)
- name
  - Típus: VARCHAR (255)
  - A felhasználó által megadott név
- email
  - Típus: VARCHAR (255)
  - A felhasználó által megadott email cím
- email_verified_at
  - Típus: TIMESTAMP
- password
  - Típus: VARCHAR (255)
  - A felhasználó által megadott jelszó
- phone
  - Típus: VARCHAR (255)
  - A felhasználó által megadott telefonszám
- admin
  - Típus: TINYINT (1)
  - Admin a felhasználó vagy sem
- remember_token
  - Típus: VARCHAR (255)
  - Ezzel tárolható az adott felhasználó bejelentkezési adatai
- deleted_at
  - Típus: TIMESTAMP

3. Projekt tesztelése

3.1. Playwright - End To End tesztek

A projekthez különböző teszteket is készítettünk, a Playwright segítségével, amely alapértelmezetten TypeScript alapú, de a telepítés során állítható JavaScriptre is. A Playwright Test kifejezetten az end-to-end tesztelés igényeinek kielégítésére lett létrehozva. A Playwright támogatja az összes modern renderelő motort, beleértve a Chromiumot, a WebKitet és a Firefoxot. Tesztelhetünk Windows-on, Linuxon és macOS-en, helyileg vagy CI-n keresztül, valamint natív mobil emulációval a Google Chrome-hoz Androidon és a Mobile Safari-hez.

A munkánk során használt end to end tesztek megtalálhatóak ezen az útvonalon:

- Frontend\tests

Számos teszt található a projekten belül:

1. Regisztráció
2. Bejelentkezés
3. Kijelentkezés
4. Felhasználó törlése
5. Jelszó megváltoztatása
6. Email cím megváltoztatása
7. Telefonszám megváltoztatása

Néhány konkrét end to end teszt a projektből:

```
test("Regisztráció", async ({ page }) => {
    await page.goto("/");

    const fullName = faker.person.fullName();
    const email = faker.internet.email();
    const phone = faker.phone.number();
    const password = faker.internet.password();

    await register(page, fullName, email, phone, password);

    await expect(page.getByText("Sikeres regisztráció!")).toBeVisible();
});
```

23. ábra: Playwright teszt Regisztrációhoz

```
test("Telefonszám megváltoztatása", async ({ page }) => {
    await page.goto("/");

    const fullName = faker.person.fullName();
    const email = faker.internet.email();
    const phone = faker.phone.number();
    const password = faker.internet.password();
    const newPhone = faker.phone.number();

    await register(page, fullName, email, phone, password);
    await login(page, email, password);
    await openSettings(page);

    await expect(page.getByTestId("current-phone-input")).toHaveValue(phone);
    await expect(page.getByTestId("current-phone-input")).toBeDisabled();

    await page.getByTestId("new-phone-input").fill(newPhone);
    await page.getByTestId("new-phone-again-input").fill(newPhone);

    await page.getByTestId("save-phone-change-button").click();
    await expect(page.getByText("Sikeres mentés!")).toBeVisible();
});
```

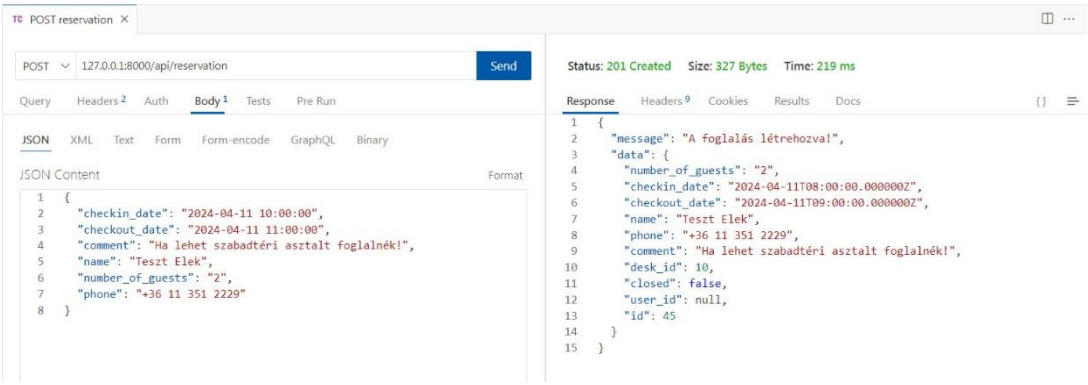
24. ábra: Playwright teszt Telefonszám megváltoztatása

3.2. ThunderClient - API tesztek

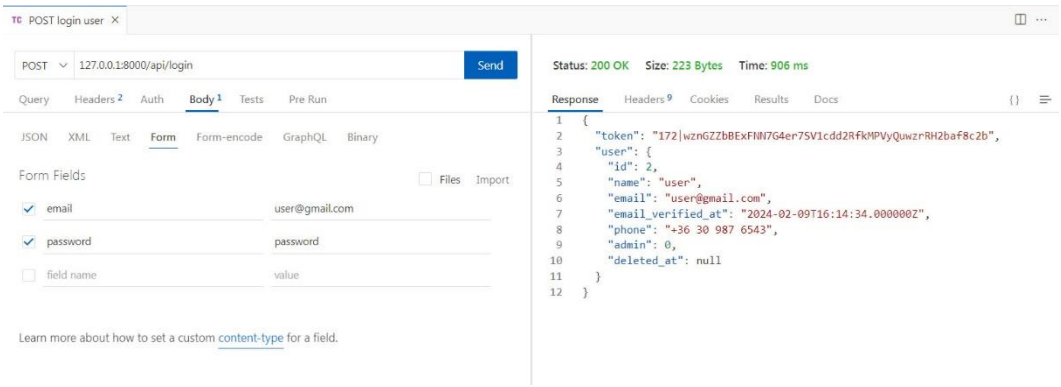
A ThunderClient egy hatékony eszköz a Laravel backend route-jainak tesztelésére. Ez egy könnyen használható API-kliens, amely lehetővé teszi HTTP kérések küldését és válaszok fogadását a backenddel. A tesztelés során lehetőség van különböző HTTP módszerek, például GET, POST, PUT, DELETE stb. használatára, hogy ellenőrizni lehessen a különböző funkciókat és viselkedéseket. A ThunderClient intuitív felhasználói felülettel rendelkezik, amely segít a kérések könnyű konfigurálásában és a válaszok vizsgálatában. Ezen felül, a tesztek könnyen automatizálhatók, így a fejlesztők gyorsan és hatékonyan tudják ellenőrizni a Laravel backend működését és stabilitását. Ezáltal a ThunderClient jelentős segítséget nyújt a Laravel alkalmazások fejlesztése és karbantartása során.

Munkánk során használt teszt route-ok kollekciója megtalálható a projekt GitHub Repository-jában "thunder-collection_menugenius.json" néven.

Néhány konkrét API teszt a projektből



25. ábra: ThunderClient teszt Asztal foglalásra

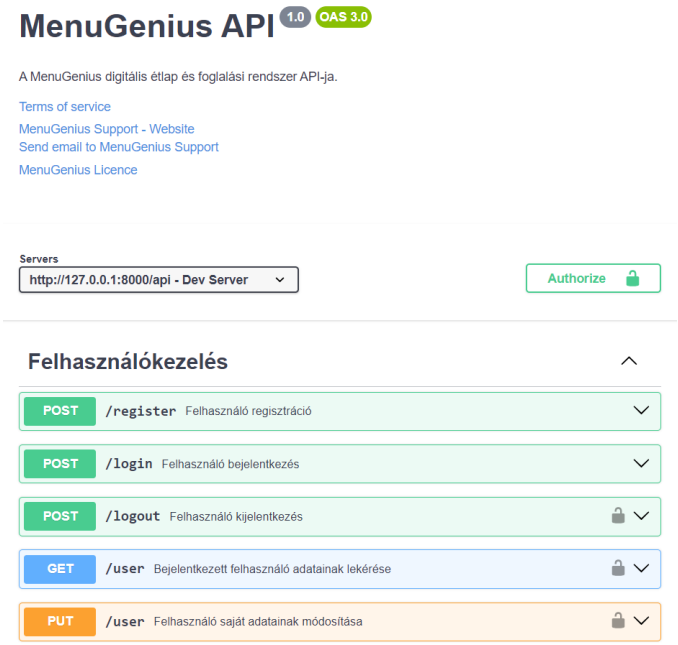


26. ábra: ThunderClient teszt Felhasználó bejelentkezés

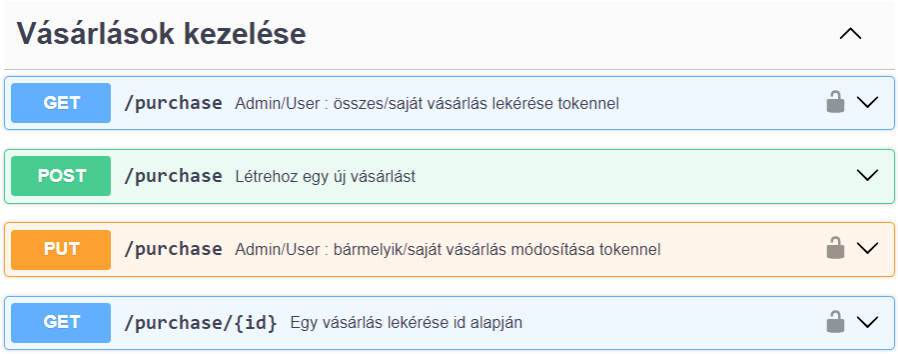
3.3. Swagger UI

A Swagger segít az API-k dokumentálásában és egyúttal egy interaktív felhasználói felületet biztosít az API-k teszteléséhez és használatához. Amikor Swaggerből generálunk egy felhasználói felületet a backend route-jaihoz, az egy könnyen navigálható, áttekinthető és tesztelhető platformot teremt az API-nk számára. A Swagger által generált dokumentáció részletes információkat nyújt az egyes végpontokról, beleértve azok elérhetőségét, paramétereit, válaszait stb. Ezáltal a fejlesztők gyorsan megismerhetik az API funkcionalitását és használatát anélkül, hogy a forráskódot részletesen kellene tanulmányozniuk. Emellett a Swagger generált felülete lehetővé teszi a különböző kérések elküldését és a válaszok vizsgálatát, így könnyen és hatékonyan lehet tesztelni az API-t. Összességében a Swagger által generált felhasználói felület segít a fejlesztőknek jobban megérteni és tesztelni a Laravel backend route-jait, ezáltal hozzájárulva az alkalmazás minőségének és stabilitásának javításához.

Néhány konkrét példa a projektből a Swagger UI segítségével



27. ábra: Swagger UI Alapinformációk és felhasználó kezelés



28. ábra: Swagger UI Vásárlások kezelése

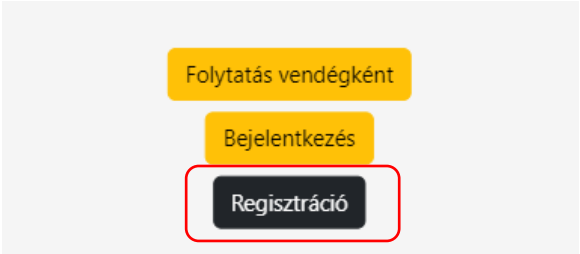
4. Felhasználói dokumentáció

4.1. Felhasználói fiók

4.1.1. Regisztráció

A MenuGenius weboldalon lehetőség van a regisztrációra, amelyet a bal felső sarokban található kis ikon segítségével tudunk megoldani. Ez egy felugró ablakot jelenít meg és lehetőségünk lesz a Regisztráció gombra kattintani.

29. ábra: Menüsor

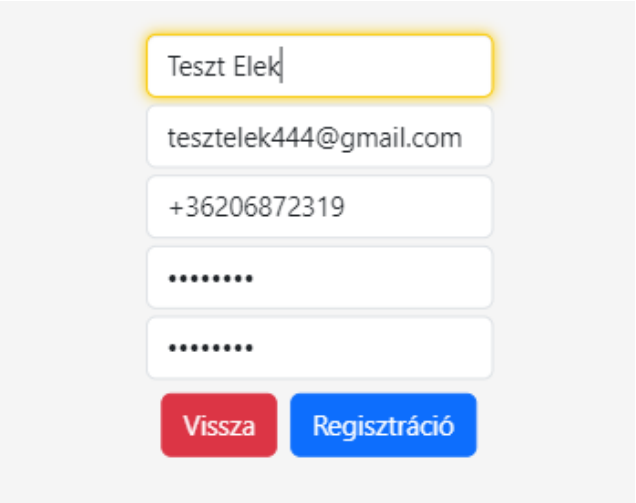


30. ábra: Elérhető opciók

Hibás adatok megadása nem lehetséges, mivel a regisztrációs űrlapon mindenre kiterjedő hibakezelés található. Amennyiben a felhasználó rossz adatot adna meg, azt a rendszer a megfelelő hibaüzenettel jelzi.

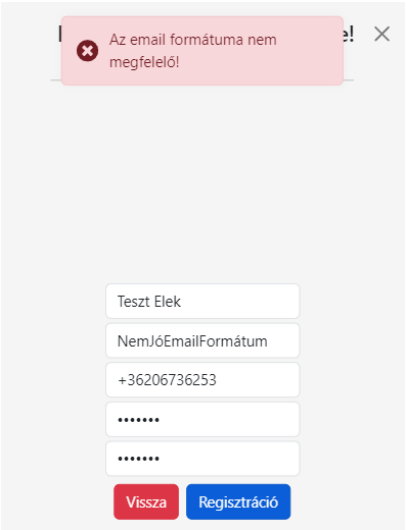
A regisztrációhoz a következő adatok szükségesek:

- Teljes név
- E-mail cím
- Telefonszám
- Jelszó
- Jelszó újra

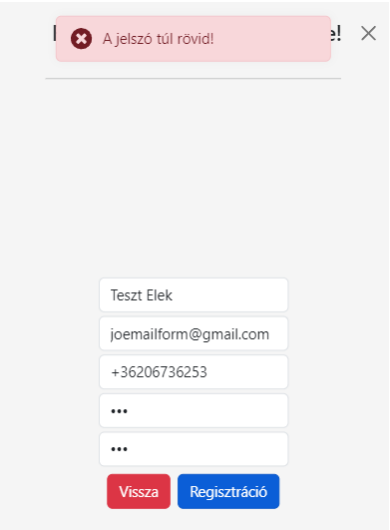


31. ábra Regisztrációhoz szükséges adatok

Példa két rosszul kitöltött űrlap hibaüzenetét illetően, ahol először az e-mail címet, majd a jelszót töltöttük ki nem megfelelő adatokkal:



32. ábra: Regisztráció rossz email



33. ábra: Regisztráció rossz jelszó

4.1.2. Bejelentkezés

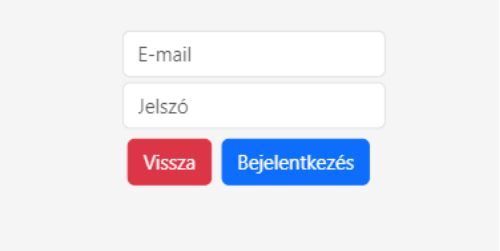
A weboldalra való bejelentkezéshez a bal felső sarokban levő kis ikont kell használnunk, ami aztán egy felugró ablakot jelenít meg számunkra, ahol lehetőség lesz kiválasztani a Bejelentkezés gombot.

34. ábra: Menüsor

35. ábra: Elérhető opciók

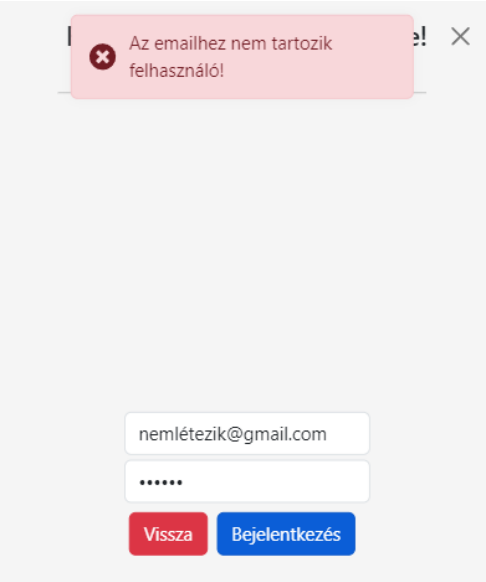
A bejelentkezéshez szükséges adatok:

- E-mail cím
- Jelszó



36. ábra: Bejelentkezéshez szükséges adatok

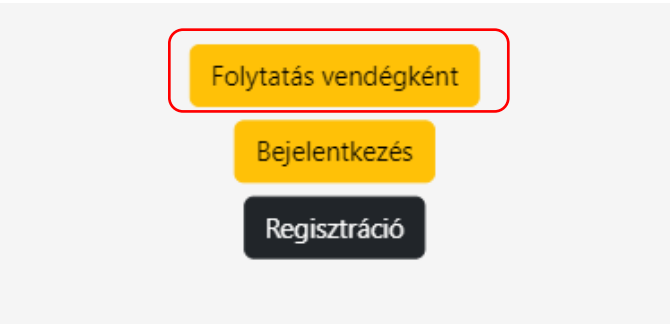
Kizárólag regisztrált felhasználók tudnak csak bejelentkezni a MenuGenius weboldalára, erről visszajelzést is kap hibaüzenet formájában, amennyiben még nem tette volna meg a regisztrációt:



37. ábra: Nem létező felhasználó

4.1.3. Vendég felhasználók

Vendég felhasználóként nem kell regisztrálni, illetve bejelentkezni sem a weboldalra. A főbb funkciói az oldalnak így is elérhetőek. Lehetséges az asztalfoglalás és a kínálatban levő keresés, az összes szűrési funkcióval együtt. „Folytatás vendégként” gombra kattintva tudjuk elérni ezeket a funkciókat.

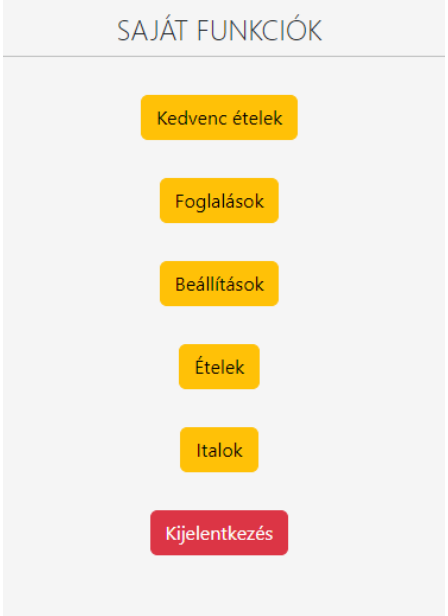


38. ábra: Elérhető opciók

4.1.4. Regisztrált és bejelentkezett felhasználok számára elérhető funkciók

A MenuGenius által nyújtott plusz funkciók a bejelentkezés után elérhetővé válnak a felhasználó számára, megjelenik a felső navigációs sávban, jobb felül, egy hamburger ikon, amelyen belül a *Kedvenc ételek, Foglalások, Beállítások* és a *Kijelentkezés* gomb lesz látható, illetve az *Étlapra* és az *Itallapra* mutató gombok.

39. ábra: Navigációs sáv Hamburger ikon



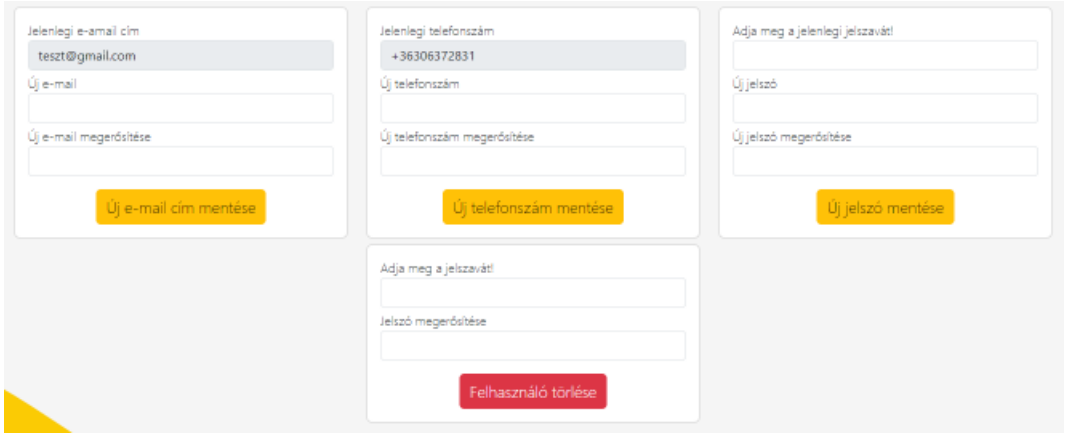
40. ábra: Elérhető plusz funkciók

A navigációs sávon megjelenik egy bevásárlókosár ikon is, ezáltal a felhasználó tudja, hogy bejelentkezett állapotban van.

41. ábra: Bejelentkezett állapot

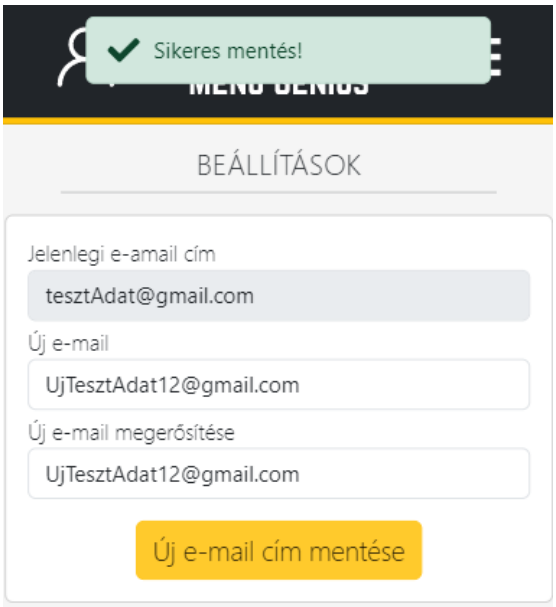
4.1.5. Beállítások oldal

A *Beállítások* gombra kattintva megjelennek a regisztráció során, a felhasználó által megadott adatok automatikusan. Ezen oldalon belül a felhasználónak jogosultsága van az e-mail címének, a telefonszámának, valamint a jelszavának a megváltoztatására, illetve törölheti saját fiókját.



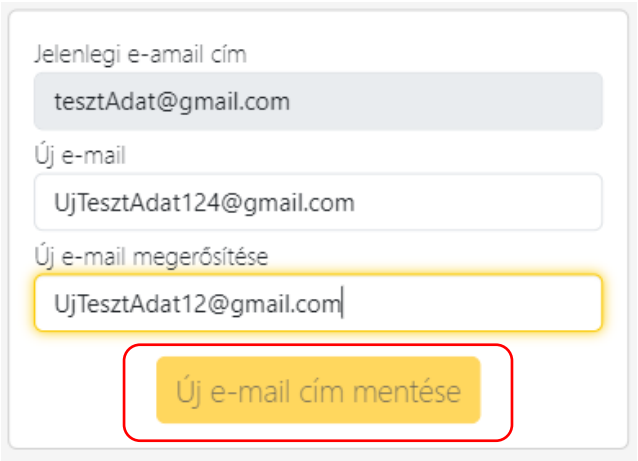
42. ábra: Beállítások

Amennyiben az adott személyes adat megváltoztatása sikeresen végbe ment, arról a felhasználó egy visszajelzést is kap, amely az alábbi képernyőfotón látható:



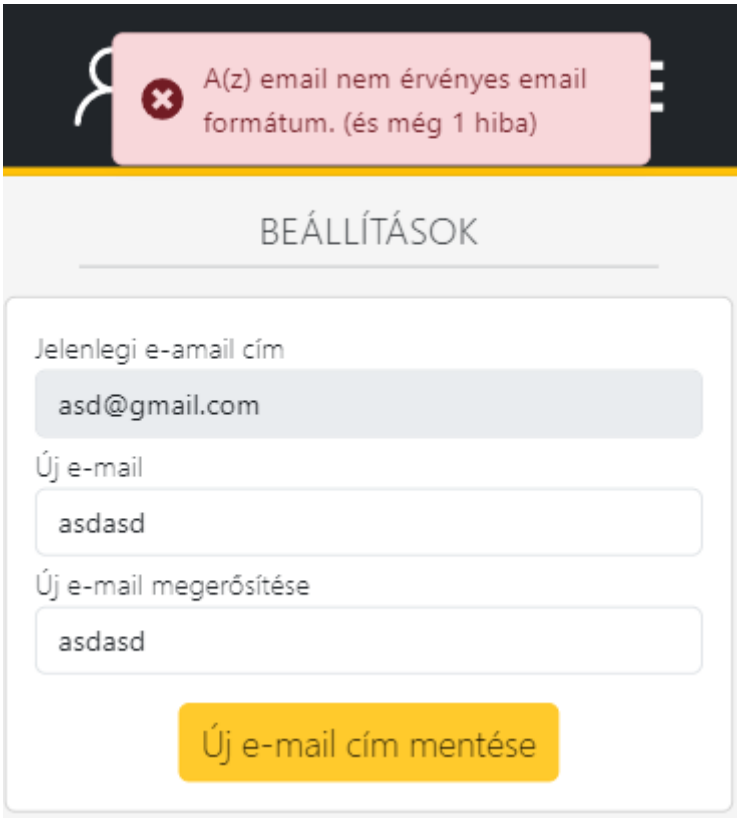
43. ábra: Sikeres mentés

Nem azonos adatok megadásáról azonnali visszajelzést kap a felhasználó, azáltal, hogy nem lesz kattintható számára az új adat mentése gomb:



44. ábra: Nem azonos adatok

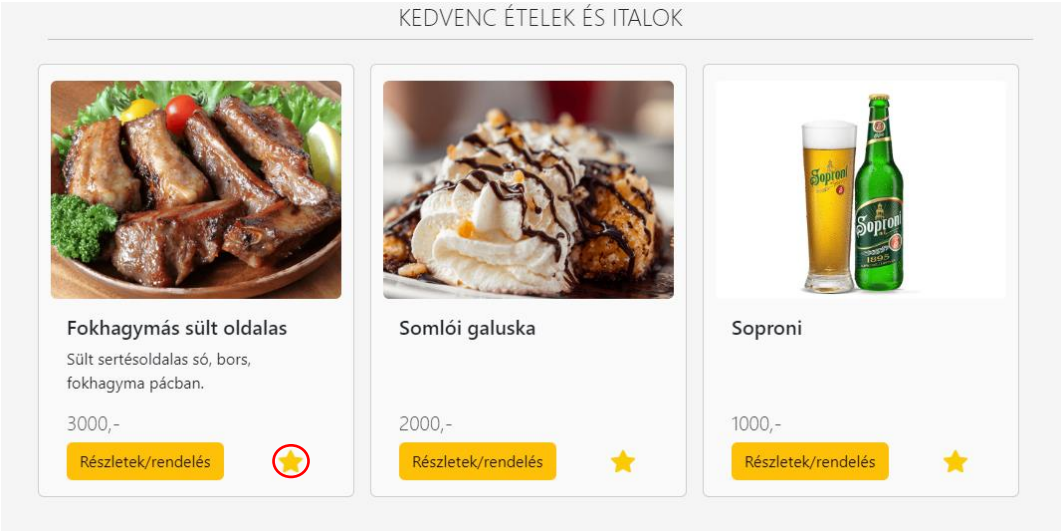
Amennyiben a felhasználó rosszul adta meg az adott adatot, arról is visszajelzés látható egy hibaüzenet formájába. Minden mezőnek egyedi hibaüzenetei vannak a hibának megfelelően, így a felhasználó egyből tudni fogja, hogy mit kell javítania:



45. ábra: Egyedi hibaüzenet

4.1.6. Kedvenc ételek és italok oldal

A *Kedvenc ételek és italok* gombra kattintva megjelennek a felhasználó által kiválasztott ételek és italok a kínálatból, ez segíthet a későbbiekben a gyors étel és ital rendelésben.



46. ábra: Kedvenc ételek oldal

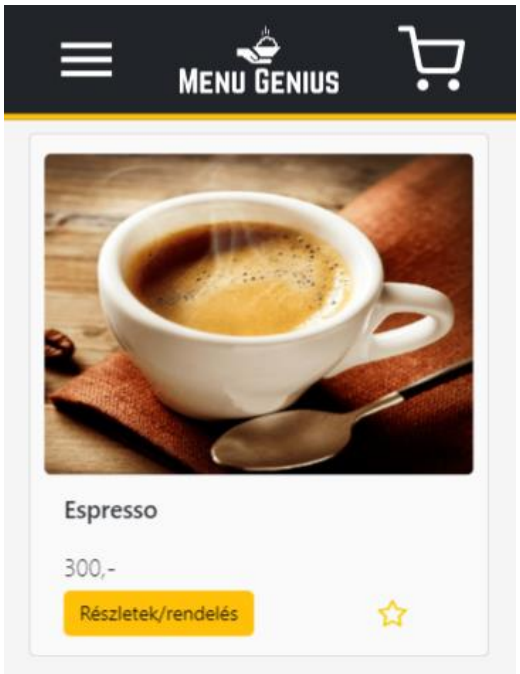
A csillag embléma segítségével egy kattintással tudja a *Kedvencekhez* adni az ételeit és italait a felhasználó, és ugyanígy eltávolítani is.

4.1.7. Ételek

Az Ételek gombra kattintva csak az ételek jelennek meg a felhasználó számára, ezzel is megkönnyítve a gyors keresést.

4.1.8. Italok

Az Italok gombra kattintva csak az italok jelennek meg a felhasználó számára, ezzel is megkönnyítve a gyors keresést.



4.1.9. Foglalások oldal

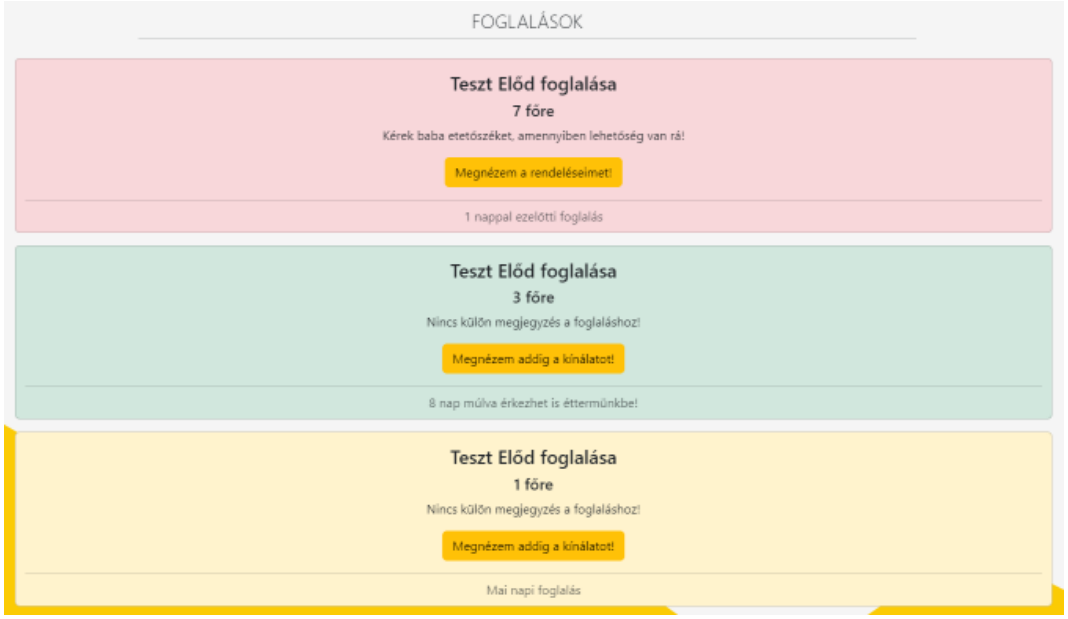
Amennyiben a felhasználó foglalt már asztalt, akkor az a *Foglalások oldalon* fog megjelenni az általa megadott különböző adatokkal, innen könnyen elnavigálhat a kínálathoz, ameddig várakozik.

47. ábra: Foglalás egyedi adatokkal

A *Foglalások oldalon* továbbra is megmaradnak az adatok, ha már elmúlt az adott időpont, így könnyen vissza lehet nézni, hogy miket rendelt azon a napon és milyen értékben fogyasztott.

48. ábra: Elmúlt foglalás

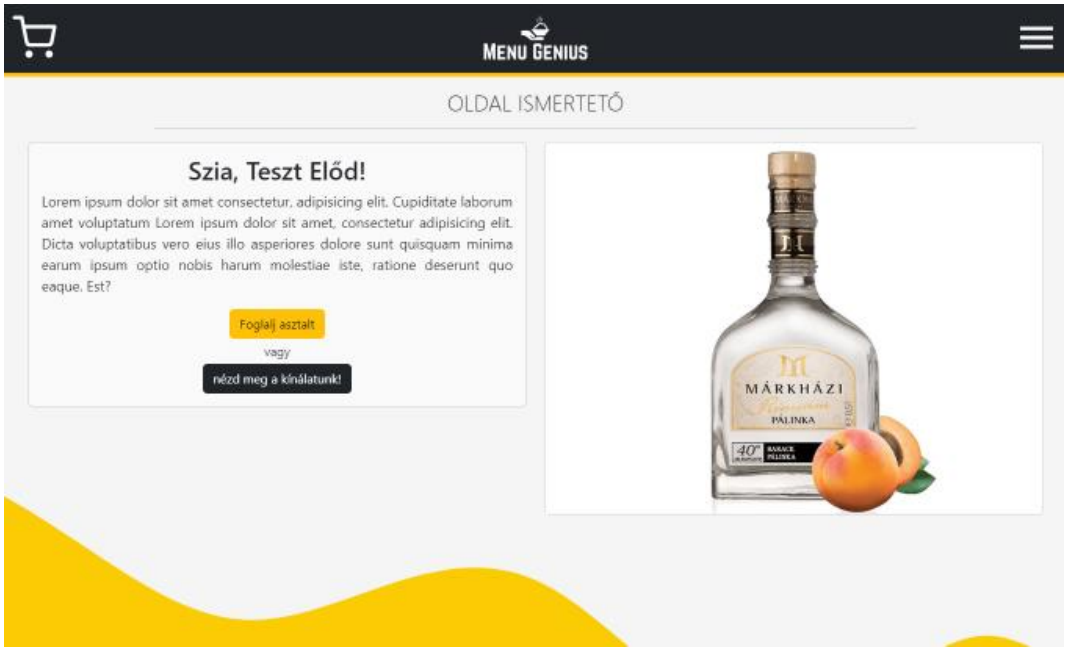
A könnyű átláthatóság érdekében az elmúlt foglalások kártyáinak a háttere pirosra, a jövőben levőknek zöldre, míg az azon a napon levő foglalásoknak sárgára van állítva.



49. ábra: Színjelzéses foglalások

4.1.10. Főoldal

A főoldalon a megjelenés és a funkciók is mind a bejelentkezett és a nem bejelentkezett, vendég felhasználók számára is ugyanaz, a *Hamburger* és a *Bevásárlókosár* ikon kivételével.

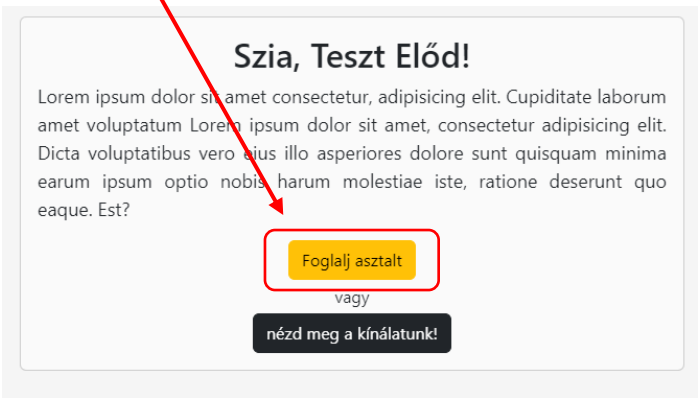


50. ábra: Főöldal

Itt két darab kártya található, az egyik kártya egy rövid és könnyen értelmezhető leírást ad az oldalról, és a kettő gomb segítségével egyből el tudunk navigálni az *Asztalfoglaláshoz,* illetve a *Kínálatot* is meg tudjuk tekinteni egy kattintás segítségével. A másik kártyában pedig véletlenszerűen jelennek meg a termékekről fotók, amelyek akár a választásban is segíthetnek, ugyanis, ha rákattint a felhasználó, akkor egyből a *Termék részleteihez* navigálja az oldal.

4.1.11. Asztalfoglalás oldal

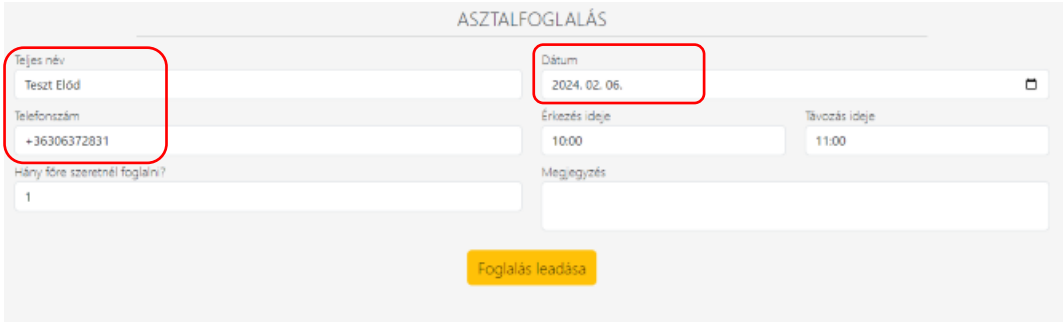
A felhasználó a ”*Foglalj asztalt”* gombra kattintva eljut az *Asztalfoglalás* oldalra, ahol asztalt tud foglalni.



51. ábra: Navigációs gomb asztalfoglaláshoz

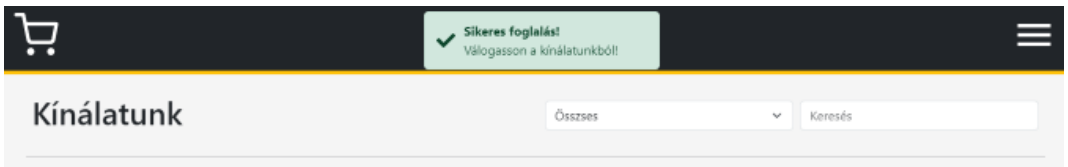
Amennyiben a felhasználó bejelentkezett állapotban van, úgy az általa megadott adatok automatikusan kitöltésre kerülnek, valamint az adott napi dátum is betöltésre kerül, ezzel is segítve a zökkenőmentes és gyors űrlap kitöltést.

Az alábbi fotón jelölve van az automatikusan kitöltött rész:



52. ábra: Automatikus mező kitöltés

Ha a foglalás sikeres, akkor arról azonnali visszajelzést kap a felhasználó, ez egy kis felugró üzenetben jelenik meg a képernyő tetején, eközben az oldal el is irányítja a kínálathoz a felhasználót.



53. ábra: Sikeres foglalás utáni állapot

4.1.12. Kínálatunk oldal

A felhasználó a főoldalon levő, *„nézd meg a kínálatunk”* gombra kattintva elnavigál a *Kínálatunk* oldalra az oldalon belül.

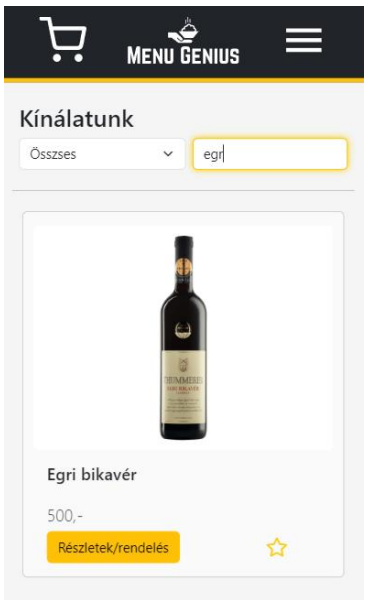
54. ábra: Navigációs gomb a kínálathoz

A *Kínálatunk* oldalon található az összes termék az adott éttermen belül, két lehetőség is segíti a felhasználót abban, hogy meg tudja találni a számára megfelelő terméket gyorsan és egyszerűen.

Egyik ilyen funkció, a kategóriák szerinti szűrés, amely alább látható:

55. ábra: Kategória általi szűrés

A másik funkció pedig a *Keresés* mezőbe való gépeléssel érhető el, minden új karakter beírásánál frissül az oldal.



56. ábra: Szűrés keresés funkcióval

A *Kínálatunk* oldalon a termékek kis kártyákba vannak kiszervezve, amelyben minden termékről egy kisebb ismertető látható, elsősorban a termék neve, leírása majd az ára. Ha a *„Részletek/rendelés”* gombra kattint a felhasználó, akkor elirányítja az oldal az adott terméknek a részletes leírásához, és plusz információkat

tudhat meg az étellel kapcsolatosan, mint például az összetevőket és nem utolsó sorban azokról az allergénekről is információt szerezhet, amelyeket az adott termék tartalmaz.

57. ábra: Termék kártya és a hozzátartozó gomb

4.1.13. Termék részletei oldal

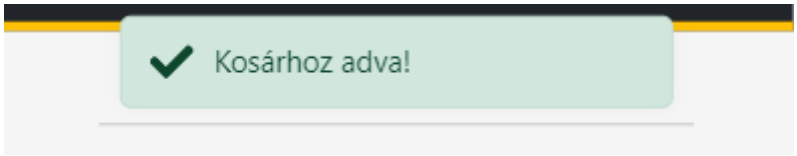
Ezen az oldalon sok új és hasznos információhoz juthat a felhasználó az adott termék részletes leírása, összetevői és allergénjei által.

58. ábra: Termék részletei

További lehetőségek elérhetőek, ilyen a már ismert *Kedvencekhez* adás a csillag embléma által, valamint a *„Kosárhoz adás”* és a termék *„Azonnali rendelése”*. Ezek mellett természetesen, ha az adott termék nem nyerte el a tetszését a felhasználónak, akkor a *„Vissza a kínálathoz!”* gomb segítségével egyből újra a *Kínálatoknál* találja magát és tovább válogathat a termékek között.

A *„Kosárhoz adás!”* gombra kattintva egy üzenet fogadja a felhasználót, amennyiben sikeresen végbement a művelet. Annyiszor adódik a termék a kosárhoz, ahányszor a felhasználó rákattint a *„Kosárhoz adás!”* gombra, ez természetesen a *Kosár* megnyitásával módosítható lesz utólagosan, amennyiben

mégsem kívánja az adott terméket vagy esetlegesen a termék mennyiségét szeretné módosítani.



59. ábra: Sikeres kosárhoz adás visszajelzés

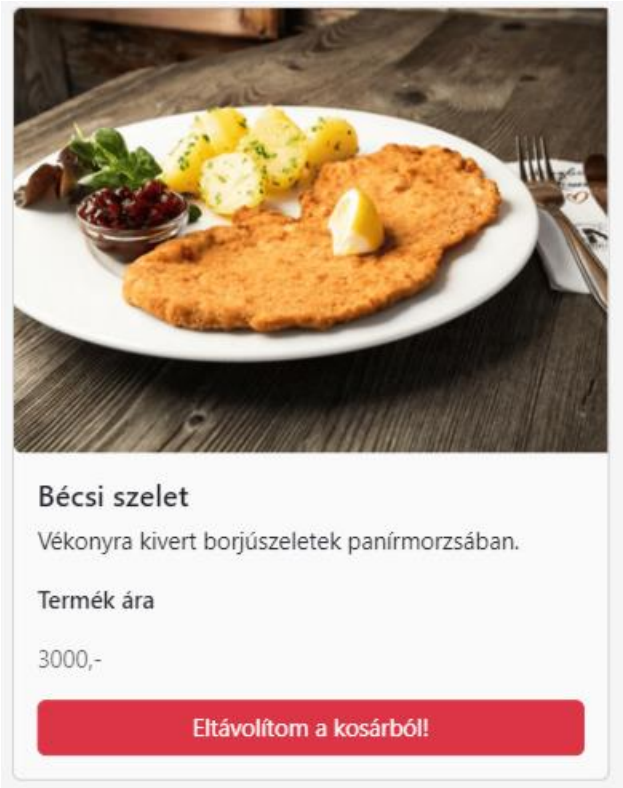
Amennyiben az *„Azonnali rendelés!”* gombra kattint a felhasználó, úgy az oldal egyből a *Kosárhoz* navigálja.

4.1.14. Kosár oldal

A „*Kosár*” oldalra lépve, ha még nem adott hozzá a felhasználó terméket, akkor az annak megfelelő üzenet fogadja és egy gomb segítségével egyből a *Kínálathoz* tud navigálni.

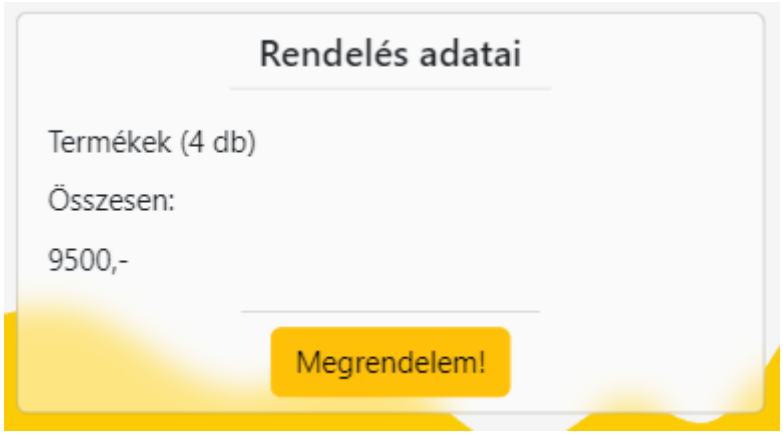
60. ábra: Üres kosár

Termék hozzáadása utána *Kosárban* megjelennek a kiválasztott ételek és italok, amelyeket el tud távolítani a felhasználó az *„Eltávolítom a kosárból!”* gombbal, amennyiben mégsem kívánja azt megrendelni.



61. ábra: Termék eltávolítás a kosárból

A *Kosár* oldalon a felhasználó, nyomon követheti egy kis összegző kártya által, hogy milyen mennyiségben és mekkora összegben válogatott össze termékeket.



62. ábra: Rendelés részletes adatai

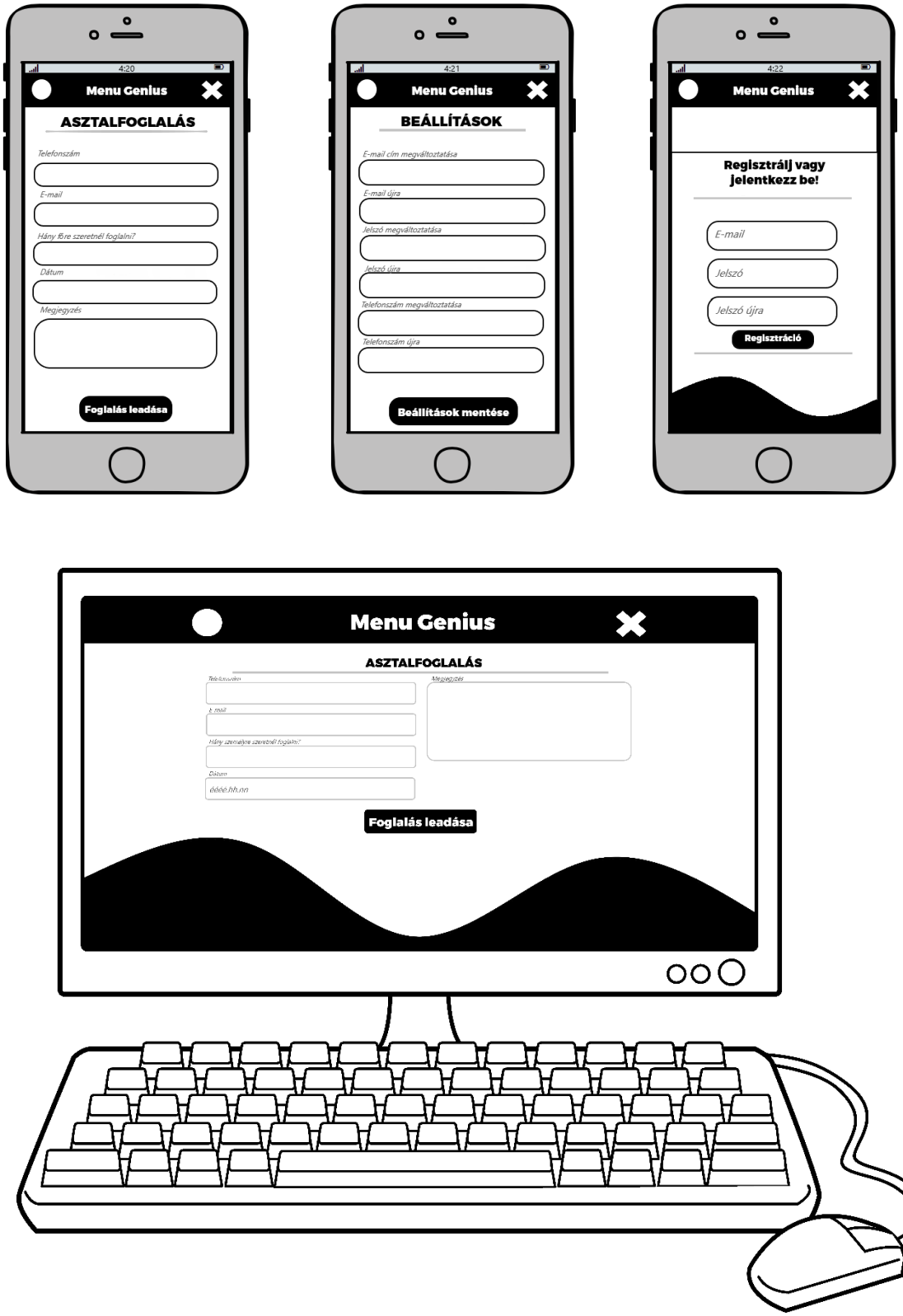
5. Összefoglalás

A szakdolgozatunkon belül bemutattuk, hogy hogyan lehet egy olyan alkalmazást megvalósítani, amely különböző éttermeknek nyújt megoldást az étel- és italrendelésre, valamint különböző plusz funkciókat is tud biztosítani a felhasználók számára, amely által ez az alkalmazás egyedi lesz a piacon.

A tervezési fázisban megfogalmaztuk az üzleti logikát és a pontos feladatokat kiosztottuk egymás között, amelyet mindenki teljesíteni tudott a projekt végére. Az alkalmazás megvalósítására egyértelmű volt, hogy PHP-t fogunk használni, Laravel-lel kiegészítve. Ez egy nyílt forráskódú, terjedelmes dokumentációjú, PHP keretrendszer.

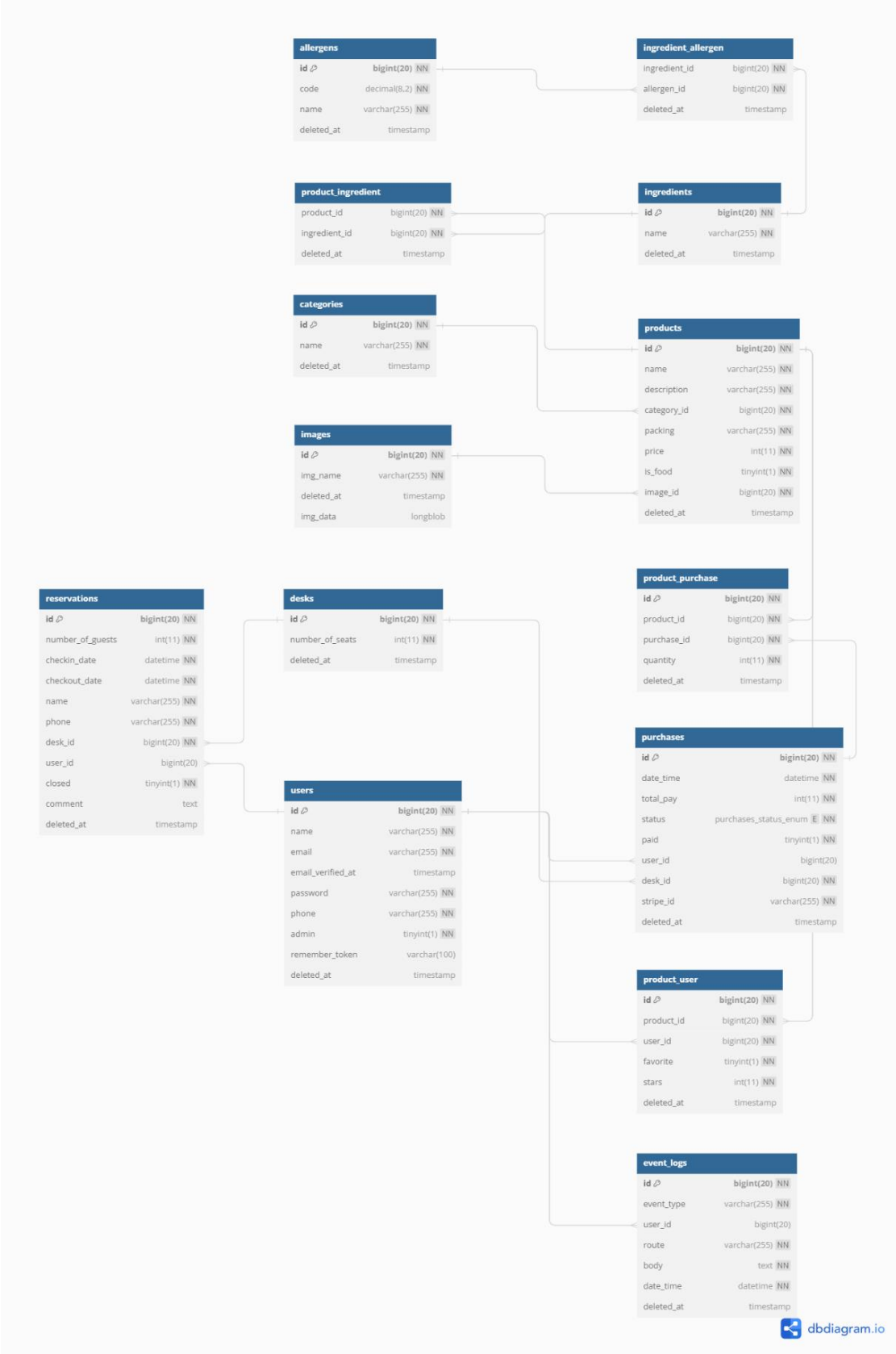
A fejlesztés során nagymértékben támaszkodtunk a Laravel és Angular hivatalos dokumentációira, amelyek segítségével a legtöbb problémára gond nélkül találtunk megoldást. Az üzleti és a logikai modell, valamint az adatbázisszerkezet létrehozása volt az első és legfontosabb, hogy egy átfogó képet kaphassunk az alkalmazásról. Ezt követte párhuzamosan a felhasználói felület és a backend folyamatos fejlesztése.

Mellékletek



1. számú melléklet

2. számú melléklet



6. Idézett forrásmunkák

https://stackdiary.com/front-end-frameworks/ 2024. 01 29.

https://techjury.net/blog/laravel-statistics/ 2024. 01 29.

https://www.trofea.hu/ 2024. 01 29.

https://www.mcdonalds.com/hu/hu-hu.html 2024. 01 29.

https://kfc.hu/ 2024. 01 29.

PHP: Hypertext Preprocessor 2024.02.07

MySQL 2024.02.07

Database Tools for Development and Management | dbForge (devart.com) 2024.02.07

Git és GitHub gyorstalpaló. A fejlesztők körében leginkább… | by Laszlo Fazekas | ENVIENTA Magyarország | Medium 2024.02.07

Verziókezelés – Wikipédia (wikipedia.org) 2024.02.07.

Front-end jelentése (lexiq.hu) 2024.02.07.

https://www.namecheap.com/logo-maker/ 2024.02.08.

MVC Framework Introduction - GeeksforGeeks 2024.02.08.

https://angular.io/guide/architecture-services 2024. 03. 08.

https://www.hrenko.hu/blog/a-carousel-hirdetesek-elonyei 2024.02.08.

https://webcapital.hu/hu/blog/laravel-bemutatasa-mukodese 2024.02.09.

https://laravel.com/docs/10.x/artisan 2024.02.09.

https://kiszervezettmarketing.hu/weboldal-keszites/http-protokoll/ 2024.03.05.

https://www.awh.hu/kb/webtarhely/mi-az-a-mysql 2024.02.09.

https://playwright.dev/docs/intro 2024.04.05.

https://www.youtube.com/watch?v=h0j0QN2b57M&list=PL_c9BZzLwBRK0Pc28IdvPQizD2mJlgoID 2023.09.15.

https://www.youtube.com/watch?v=zvyQNuuTqks 2023.10.29.

https://www.youtube.com/playlist?list=PLA8ZIAm2I03hS41Fy4vFpRw8AdYNBXmNm 2023.11.02.

https://www.youtube.com/playlist?list=PL4alkMYFMfltGP3P5c_IU65GGiwNR-eFn 2023.12.10.

https://www.youtube.com/playlist?list=PL1BztTYDF-QNlGo5-g65Xj1mINHYk_FM9 2024.01.01.

https://www.youtube.com/playlist?list=PLFDH5bKmoNqxKwOIghq-c9CGtJbyjQ24p 2024.01.06.

https://stackoverflow.com/questions/16506653/accessing-a-property-in-one-viewmodel-from-another 2024.01.09.

https://stackoverflow.com/questions/31826221/laravel-cashier-how-to-check-if-stripe-payment-was-successful 2024.01.17.

https://www.youtube.com/playlist?list=PLLhM616BHkss3VLQdFqmEOrq0aktOItqC 2024.01.21.

https://www.youtube.com/playlist?list=PLZlA0Gpn_vH8jbFkBjOuFjhxANC63OmXM 2024.02.01.

https://www.youtube.com/playlist?list=PLOIq79MWqv84BLdSq8avEj5mMtxT2zWLE 2024.02.11.

https://www.youtube.com/playlist?list=PLXC_gcsKLD6n7p6tHPBxsKjN5hA_quaPI 2024.02.13.

https://www.youtube.com/watch?v=J13Xe939Bh8 2024.02.18.

https://www.youtube.com/playlist?list=PLlJeO89e07g6HaiIEw7pYll4DQCFb6SFJ 2024.02.22.

https://www.youtube.com/watch?v=cWRG2gaZYQw 2024.03.03.

https://www.youtube.com/playlist?list=PLgkLZS2sgBf_5um4uNGBrrItRTKGWMjYT 2024.03.07.

https://stackoverflow.com/questions/10667002/wpf-datagrid-deselect-selected-items-when-clicking-whitespace-in-the-datagri 2024.03.09.

https://stackoverflow.com/questions/7833425/wpf-datagrid-selecteditem-null-in-mvvm 2024.03.10.

https://stackoverflow.com/questions/6813440/stuck-with-asp-net-mvc-3-0-scaffolding-in-case-of-many-to-many-relationship 2024.03.10.