

# DANKA3D Webshop

Készítette:

Danka András (Backend, Frontend)

Gintli Máté (Admin panel, Szoftverteszt)

Konzulens:

Soós Gábor

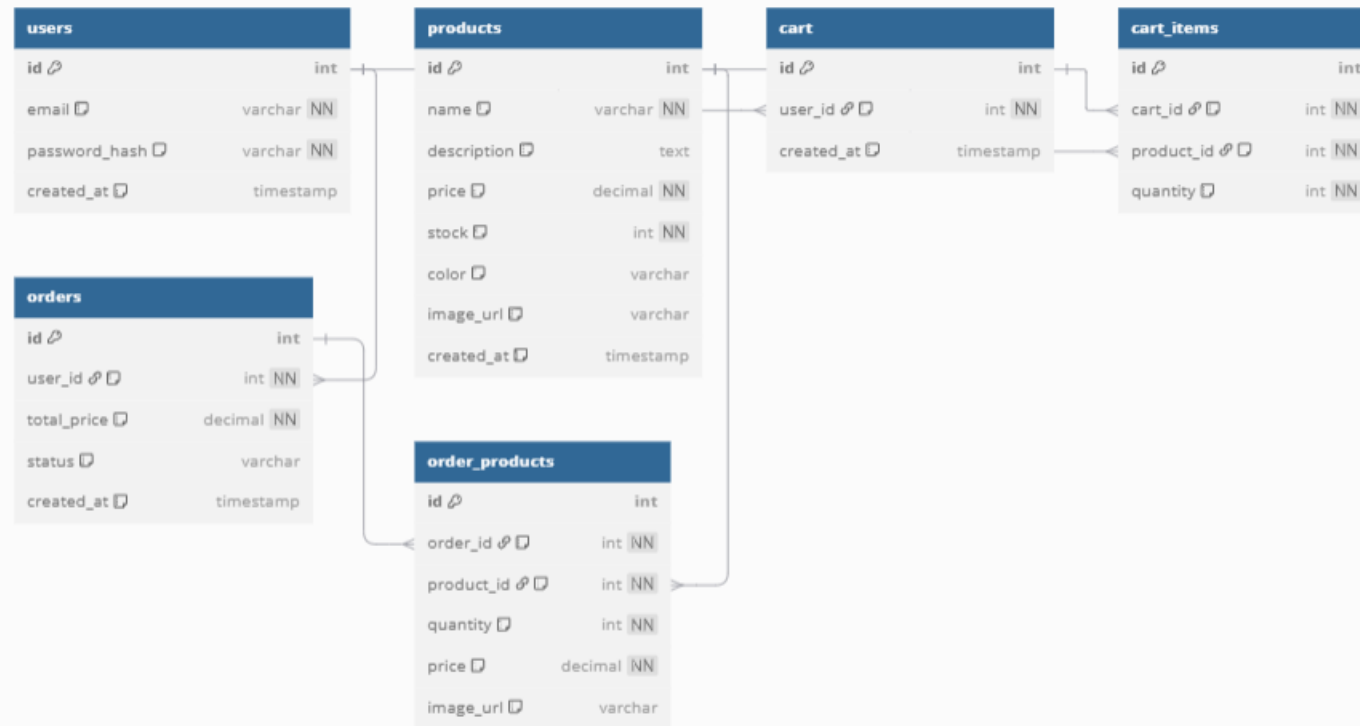
# 1 Bevezetés

A vizsgaremekünk témája egy webshop, amit elsősorban édesapámnak kezdtem el csinálni. A program elkészítésekor ügyeltünk a könnyen kezelhetőségre, egyszerűségére. A webshopban lehet létrehozni saját fiókot, de nélküle is lehet böngészni a termékek között. Bejelentkezett fiókkal lehet a kosárba rakni termékeket, aminek a rendelését aztán le is lehet adni. A kosár tartalmát elmenti, ha a felhasználó egy másik gépen szeretne bejelentkezni. A weboldal teljes mértékben reszponzív, így akár telefonról, vagy éppen tabletről is kiválóan működik.

# 1.1 Bevezetés (a program célja)

- Termékek böngészése, keresőmezővel
- Regisztrálás és Bejelentkezés (session alapú)
- Termékek hozzáadása a kosárhoz, kosárban való szerkesztéssel, és valós idejű végösszeg számlálóval
- Termékek megrendelése, mely a rendelések menün megjelenik, valamint az adatbázisban is elmenti
- Admin panel célja a felhasználók által megrendelt termékek megjelenítése és azok státuszának szerkeztése. Valamit maguk a felhasználók megjelenítése és új termékek hozzáadása a már meglévőkhöz.

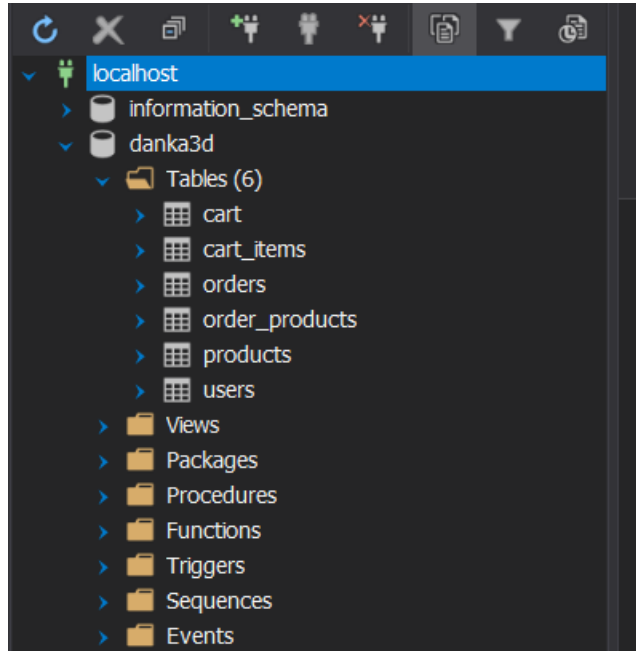
# 2 Adatbázis



Ez az adatbázis-struktúra egy e-kereskedelmi rendszer alapját képezi. Az alábbi főbb elemeket tartalmazza:

- **Felhasználók (users):** Tárolja a regisztrált felhasználók adatait, például e-mail címeket és jelszó-hashekeket.
- **Termékek (products):** A kínált termékek részleteit tartalmazza, beleértve az árakat, készletet és színeket.
- **Kosár (cart):** Minden felhasználóhoz kapcsolódik, és a vásárlás előtti termékeket gyűjti össze.
- **Kosár elemei (cart\_items):** A kosárban lévő termékek részleteit tárolja, például mennyiséget.
- **Rendelések (orders):** A vásárlások adatait tartalmazza, például összárát és státuszt.
- **Rendelés termékei (order\_products):** A rendeléshez tartozó termékek részleteit tárolja, például mennyiséget és árat.

## 2.1 Adatbázis (host)



A projektünk helyi környezetben fut, ahol az XAMPP biztosítja a MySQL adatbázist, és a dbForge segíti az adatbázisok kezelését. Ez gyors hibakeresést, rugalmas fejlesztést és kényelmes adatbázis-műveleteket tesz lehetővé.

# 3 Backend

A backend az ASP.NET keretrendszeren alapul, és a következő NuGet csomagokat használja:

- Pomelo.EntityFrameworkCore.MySql: MySQL adatbázisok hatékony kezeléséhez.
- Entity Framework Core: Objektum-relációs leképezés (ORM) az adatbázis-séma és lekérdezések egyszerűsítésére.
- Entity Framework Core Tools: Eszközök a migrációk kezelésére és az adatbázis műveletekhez. Ezek együtt gyorsítják a fejlesztést és az adatkezelést.

## 3.1 Backend (Model-ek)

```
namespace backend.Models
{
    [Table("users")]
    2 references
    public class UserModel
    {
        [Key]
        [Column("id")]
        3 references
        public int Id { get; set; }

        [Required]
        [MaxLength(100)]
        [Column("email")]
        6 references
        public string? Email { get; set; }

        [Required]
        [MaxLength(255)]
        [Column("password_hash")]
        2 references
        public string? PasswordHash { get; set; }

        [Column("created_at")]
        1 reference
        public DateTime CreatedAt { get; set; } = DateTime.UtcNow;
    }
}
```

Az alkalmazásban 6 modell található, amelyek megfelelnek az adatbázisban lévő 6 táblának. Ezeket a modelleket a Pomelo.EntityFrameworkCore.MySql csomag segítségével definiáltad, amely megkönnyíti a MySQL adatbázissal való munkát. Az Entity Framework Core ORM segítségével a modellek és táblák közötti kapcsolatok könnyedén kezelhetők, így a fejlesztési folyamat hatékony és egyszerű maradt. Ez a megközelítés biztosítja az adatbázis-struktúra és az alkalmazás logikájának szoros összhangját.

## 3.2 Backend ( User Controller )

User	
POST	/api/User/register
POST	/api/User/login
POST	/api/User/logout
GET	/api/User/me

### Register:

- Ellenőrzi, hogy az e-mail cím létezik-e.
- Hasheli a jelszót biztonságosan.
- Létrehozza a felhasználót, és automatikusan egy hozzá tartozó kosarat is generál.

### Login:

- Ellenőrzi az e-mailt és a hashelt jelszót.
- A felhasználói munkamenetben (session) elmenti az e-mail címet.

### Logout:

- Eltávolítja a felhasználó adatait a munkamenetből.

### GetCurrentUser:

- Visszaadja az aktuálisan bejelentkezett felhasználó adatait a munkamenet alapján.

### Biztonságos jelszókezelés:

- HashPassword: SHA256 algoritmussal hasheli a jelszavakat.
- VerifyPassword: Ellenőrzi a beírt jelszó és a tárolt hash egyezését.



## 3.3 Backend (Product Controller)

### Product

GET /api/Product

POST /api/Product

GET /api/Product/{id}

PUT /api/Product/{id}

DELETE /api/Product/{id}

### GetProducts

- Termékek listázása oldalakra bontva
- Ellenőrzött lapozási paraméterekkel (page, pageSize)

### GetProduct

- Egy termék lekérése id alapján

### CreateProduct

- Új termék létrehozása, aktuális dátum beállítása, ha hiányzik
- Visszaadja a létrehozott terméket id-vel

### UpdateProduct

- Termék frissítése id alapján
- Ellenőrzi az id egyezést, kezeli a konkurenciát

### DeleteProduct

- Termék törlése id alapján

### ProductExists

- Ellenőrzi egy termék létezését id alapján

## 3.4 Backend (Order Controller)

### Orders

POST

`/api/Orders`

GET

`/api/Orders`

GET

`/api/Orders/user/{userId}`

#### CreateOrder

- Új rendelés létrehozása tranzakcióval
- Ellenőrzi, hogy minden termék id-je létezik-e
- Létrehozza a rendelést és kapcsolt rendelési termékeket

#### GetOrdersByUser

- Rendelések lekérése felhasználó id alapján
- Rendelési termékekkel együtt adja vissza az adatokat

#### GetOrders

- Összes rendelés lekérése termékekkel és lapozással

Ellenőrzi a lapozási paramétereket (page

## 3.5 Backend (Cart Controller)

### Cart

GET

`/api/Cart/{userId}`

POST

`/api/Cart/add-item`

PUT

`/api/Cart/update-item/{cartItemId}`

DELETE

`/api/Cart/remove-item/{cartItemId}`

DELETE

`/api/Cart/clear-cart/{userId}`

#### GetCart

- Felhasználó kosarának lekérése
- Létrehoz új kosarat, ha nem létezik

#### AddItemToCart

- Termék hozzáadása a kosárhoz
- Ellenőrzi, hogy a termék létezik-e
- Frissíti a darabszámot, ha a termék már szerepel a kosárban

#### UpdateCartItem

- Kosárban lévő termék darabszámának frissítése

#### RemoveCartItem

- Termék eltávolítása a kosárból
- Ellenőrzi, hogy a termék megtalálható-e

#### ClearCart

- Teljes kosár ürítése
- Ellenőrzi, hogy van-e termék a kosárban

## 3.6 Backend (Képek)

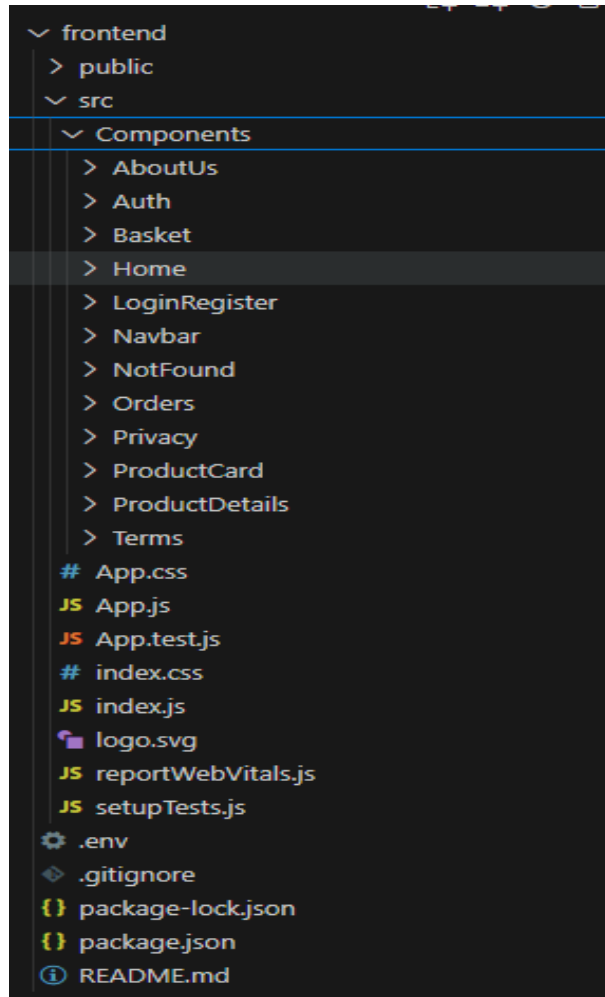
A backend alkalmazás az `app.UseStaticFiles` segítségével szolgálja ki az `images` mappában tárolt képeket, amelyeket az ügyféloldalon a `/Images` útvonalon lehet elérni. Ez egyszerűsíti a képek kezelését és hozzáférhetőségét.

```
app.UseStaticFiles(new StaticFileOptions
{
    FileProvider = new PhysicalFileProvider(
        Path.Combine(Directory.GetCurrentDirectory(), "images")),
    RequestPath = "/Images"
});
```

## 4 Frontend

A webshop frontendjét React keretrendszerrel valósítjuk meg, amely rugalmas és hatékony megoldást kínál az interaktív felhasználói felületek kialakításához. A React biztosítja az összetevők alapú fejlesztést, így könnyen kezelhetők a webshop különböző funkciói, például a termékek megjelenítése, kosár kezelése és rendelés lebonyolítása. Ezáltal a felhasználói élmény gördülékeny és modern lesz.

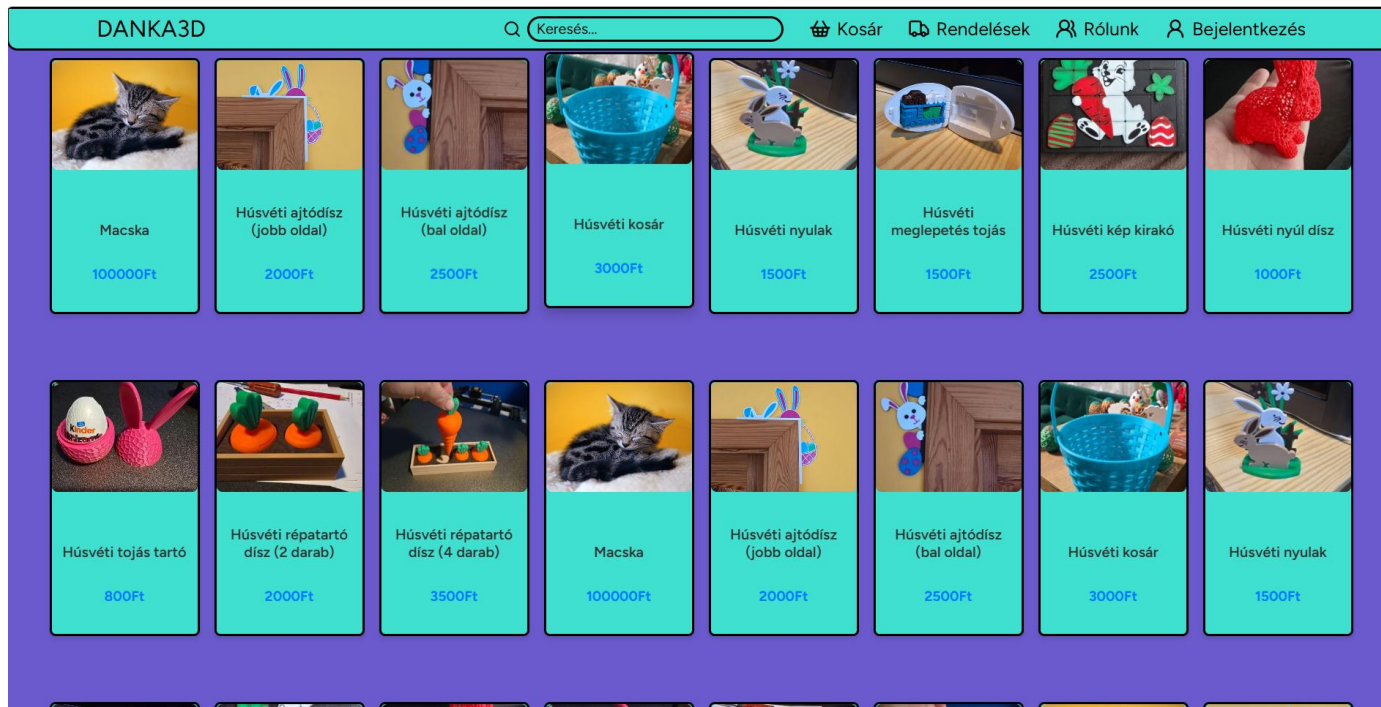
## 4.1 Frontend (Felépítés)



A frontend szintűgy, mint a backend localhost-on fut.

A képem látható a komponensek elrendezése, mindegyiknek külön mappája van, külön jsx és css file-okkal.

## 4.1 Frontend (Főoldal)



A főoldal egyben a product oldal is, ahol az összes termék látszik (azért látszódik egy-két tárgy többször, mert az adatbázisban duplikálva lettek). Az oldal legalján lapozó rendszer is van. A keresés funkció segíti a könnyebb böngészést.

A navbar-nál törekedtünk rá, hogy letisztult, egyértelmű legyen.

## 4.2 Frontend (Regisztrálás / Bejelentkezés)

### Regisztráció

☒ Elfogadom a feltételeket és az adatvédelmet.

Regisztráció

Van már fiókod? [Jelentkezz be!](#)

[← Vissza a főoldalra](#)

### Bejelentkezés

☒ Adatok megjegyzése

Bejelentkezés

Nincs még fiókod? [Regisztrálj!](#)

[← Vissza a főoldalra](#)

Regisztrálni, csak megfelelő email címmel és jelszóval lehet, a feltételek elfogadása után. Regisztráció után egyből átvizet a bejelentkezés oldalra, ahol ezután ezekkel az adatokkal be is lehet jelentkezni. Egy cookie session segítségével bejelentkezve is marad a személy, de 30p inaktivitás után kijelentkezik automatikusan.



## 4.3 Frontend (Termék adatok)

### Húsvéti kosár



Kiváló minőségű 3D nyomtatott kosár.

Ár/db: 3000Ft

Szín: kék

Mennyiség:

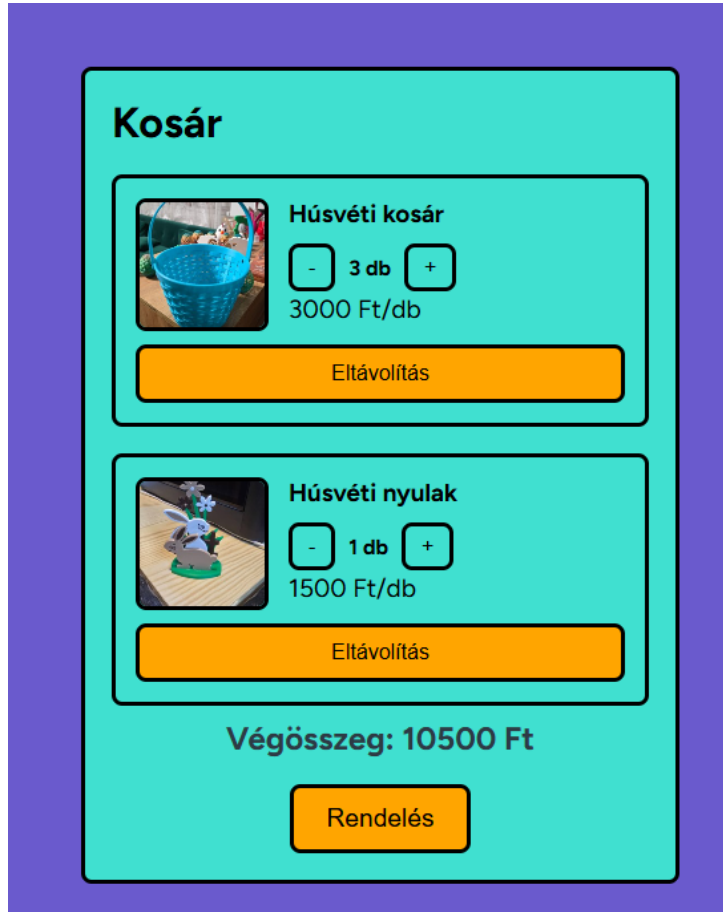
Kosárhoz adás

A főoldalon egy terméket kiválasztva megnyit egy teljesen új lapot, saját címmel Id alapján. A példában használt cím:

localhost:3000/product-details/4

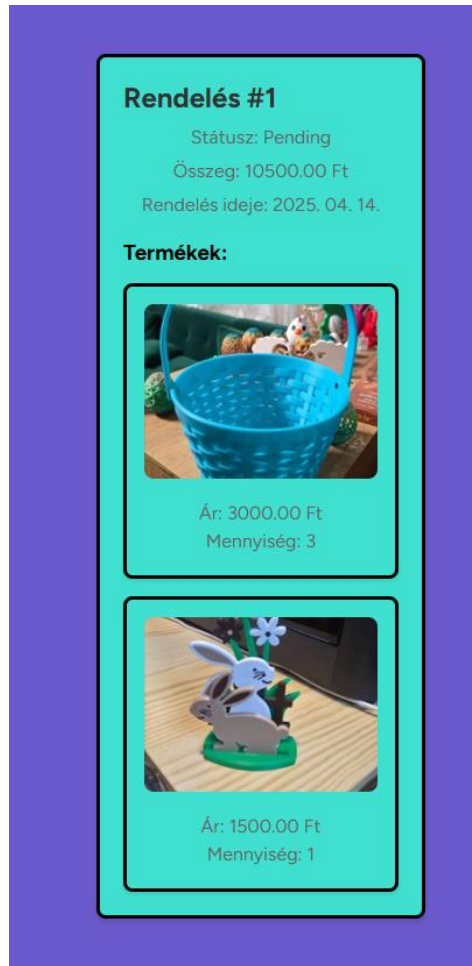
Itt például a 4-es id-vel létező terméket nyitottam meg.

## 4.4 Frontend (Kosár)



A kosárba rakott termékeket lehet törölni, valamint a mennyiségüket csökkenteni / növelni. Ráadásként valós időben írja a végösszeget is, ami azonnal frissül. A rendelés gombbal egyértelműen rendelést lehet leadni.

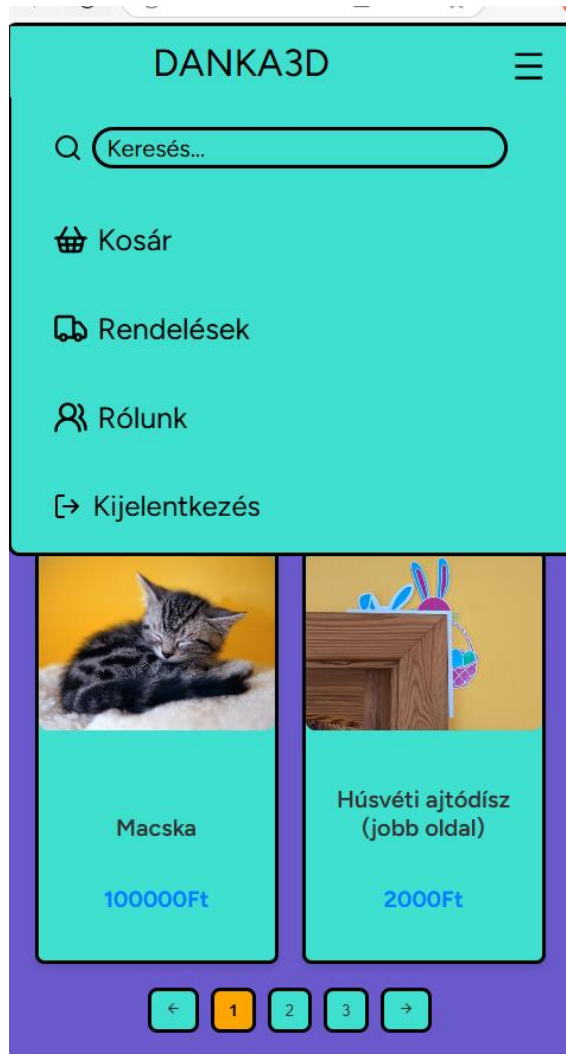
## 4.5 Frontend (Rendelések)



A rendelés leadás után a rendelés fülben megtekinthető az előző rendelések. A rendelés összegét, időpontját és a státuszát is írja, valamint eltárolja az adatbázisban.

A rendelés státuszát az admin panelben lehet majd kezelni.

## 4.6 Frontend (Rugalmas weboldal)



A képen jól látható, hogy telefonon is kiválóan működik, a navigációs felület egy hamburger menüvé változik, valamint képernyőmérettől függően jelennek meg a termék kártyák.

## 4.7 Frontend (Hibás link)



Hibás link esetén saját hiba képernyőnk van

## 4.8 Frontend (Rólunk oldal)



Az utolsó nem bemutatott oldal, a „Rólunk oldal”, itt csak egy-két információ van a csapatról.

## 5. Adminpanel

Az adminpanelhez a Visual Studio 2022 alkalmazást használatam és WPF-ben készítettem el. Ezt a módszert találtam a számomra a legkönnyebb és legjobban kezelhetőnek. Sok jó módszert sikerült beleépítenem a programba, és az admin felület könnyen kezelhető és adminbarát élményt biztosít.

Illetve a segítségemre volt néhány NuGet csomag is:

- Pomelo.EntityFrameworkCore.MySql
- Entity Framework Core
- Entity Framework Core Tools

# 5.1 Adminpanel (Főoldal)

A fő oldal célja a felhasználók listázása és a felhasználók által megrendelt termékek és azok árainak megjelenítése. A fő oldalon több funkció elérhető többek között egy termékek gomb is helyett kapott amivel a már meglévő termékekhez lehet új terméket/termékeket hozzáadni. Valamit a megrendelések státuszát is lehet szerkeszteni egy külön gombbal.


Felhasználók/Megrendelések

Felhasználók

Termékek

ID	Email	Regisztrálva
1	gintlimat@gmail.com	2025. April 27.

Megrendelések

Rendelés ID	ID	Végösszeg	Státusz	Létrehozva	Módosítás
1	1	305000.00	Készül	2025. April 27.	

<

>



## 5.2 Adminpanel (Termékek ablak)

A termékek oldalon találhatóak a termékek, illetve ezekhez lehet új termékeket hozzáadni. Több dolgot is figyelembe kell venni, mivel több adatot is be kell gépelni. Ha az admin valamit nem ír bele akkor hiba üzenet lép elő.

Termékek

Új Termék

ID	Név	Leírás	Ár	Szín	Kép URL	Létrehozva
1	Macska	Szórós	100000.00	fekete	https://th.bing.com/th/i	4/27/2025 9:43:08 AM
2	Húsvéti ajtódísz (jobb o	Kiváló minőségű 3D nyomtatott a	2000.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
3	Húsvéti ajtódísz (bal olc	Kiváló minőségű 3D nyomtatott a	2500.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
4	Húsvéti kosár	Kiváló minőségű 3D nyomtatott k	3000.00	kék	http://localhost:5277/in	4/27/2025 9:43:08 AM
5	Húsvéti nyulak	Kiváló minőségű 3D húsvéti nyula	1500.00	fehér, barna	http://localhost:5277/in	4/27/2025 9:43:08 AM
6	Húsvéti meglepetés toji	Kiváló minőségű 3D nyomtatott n	1500.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
7	Húsvéti kép kirakó	Kiváló minőségű 3D nyomtatott k	2500.00	fekete	http://localhost:5277/in	4/27/2025 9:43:08 AM
8	Húsvéti nyúl dísz	Kiváló minőségű 3D nyomtatott c	1000.00	piros	http://localhost:5277/in	4/27/2025 9:43:08 AM
9	Húsvéti tojás tartó	Kiváló minőségű 3D nyomtatott t	800.00	rózsaszín	http://localhost:5277/in	4/27/2025 9:43:08 AM
10	Húsvéti répatartó dísz (	Kiváló minőségű 3D nyomtatott r	2000.00	barna	http://localhost:5277/in	4/27/2025 9:43:08 AM
11	Húsvéti répatartó dísz (	Kiváló minőségű 3D répatartó dísz	3500.00	barna	http://localhost:5277/in	4/27/2025 9:43:08 AM
12	Macska	Szórós	100000.00	fekete	https://th.bing.com/th/i	4/27/2025 9:43:08 AM
13	Húsvéti ajtódísz (jobb o	Kiváló minőségű 3D nyomtatott a	2000.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
14	Húsvéti ajtódísz (bal olc	Kiváló minőségű 3D nyomtatott a	2500.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
15	Húsvéti kosár	Kiváló minőségű 3D nyomtatott k	3000.00	kék	http://localhost:5277/in	4/27/2025 9:43:08 AM
16	Húsvéti nyulak	Kiváló minőségű 3D húsvéti nyula	1500.00	fehér, barna	http://localhost:5277/in	4/27/2025 9:43:08 AM
17	Húsvéti meglepetés toji	Kiváló minőségű 3D nyomtatott n	1500.00	fehér	http://localhost:5277/in	4/27/2025 9:43:08 AM
18	Húsvéti kép kirakó	Kiváló minőségű 3D nyomtatott k	2500.00	fekete	http://localhost:5277/in	4/27/2025 9:43:08 AM
19	Húsvéti nyúl dísz	Kiváló minőségű 3D nyomtatott c	1000.00	piros	http://localhost:5277/in	4/27/2025 9:43:08 AM
20	Húsvéti tojás tartó	Kiváló minőségű 3D nyomtatott t	800.00	rózsaszín	http://localhost:5277/in	4/27/2025 9:43:08 AM

Új Termék Ho

Név:

Leírás:

Ár:

Szín:

Kép URL:

Mentés

Hiba

Kérlek töltsd ki a kötelező mezőket helyesen!

OK

## 5.3 Adminpanel (Státusz szerkesztő ablak)

Ebben az ablakban a rendelések státuszát lehet szerkeszteni amit a megrendelő is lát. 4 státuszt lehet beállítani.

Státusz

Rendelés ID:  
1

Teljes ár:  
305000.00

Új státusz:  

Készül

Mentés

Termék ID	Termék név	Darab	Ár / Db
4	Húsvéti kosár	1	3000.00
12	Macska	3	100000.00
24	Húsvéti ajtódísz	1	2000.00

Státusz mód

Rendelés ID:  
1

Teljes ár:  
305000.00

Új státusz:  

Készül

Termék ID	Termék név	Darab	Ár / Db
4	Húsvéti kosár	1	3000.00
12	Macska	3	100000.00
24	Húsvéti ajtódísz	1	2000.00

## 5.4 Adminpanel (Főoldal kód)

Amikor az admin rákattint a rendeléshez tartozó státusz szerkesztő gombra, megnyílik egy külön ablak, ahol módosíthatja a rendelés állapotát. Ha a módosítás sikeres, a rendelés automatikusan frissül, hogy az új adatok azonnal láthatók legyenek.

```
private void SzerkesztStatusz_Click(object sender, RoutedEventArgs e)
{
    if (sender is Button button && button.DataContext is OrderModel selectedOrder)
    {
        OrderStatusWindow statusWindow = new OrderStatusWindow(selectedOrder, context);
        bool? result = statusWindow.ShowDialog();

        if (result == true)
        {
            OrdersDataGrid.Items.Refresh();
        }
    }
}
```

## 5.5 Adminpanel (Termékek ablak kód)

A LoadProducts betölti az adatbázisból az összes terméket egy listába. Amikor az admin az "Új termék" gombra kattint, megnyílik egy ablak új termék hozzáadására, és ha sikeresen menti, az új termék bekerül az adatbázisba, majd frissül a terméklista a képernyőn.

```
private void LoadProducts()
{
    _products = context.products.ToList();
}

1 reference
private void UjTermek_Click(object sender, RoutedEventArgs e)
{
    AddProductWindow addWindow = new AddProductWindow();
    bool? result = addWindow.ShowDialog();

    if (result == true && addWindow.NewProduct != null)
    {
        context.products.Add(addWindow.NewProduct);
        context.SaveChanges();
        LoadProducts();
        ProductsDataGrid.ItemsSource = _products;
    }
}
```

## 5.6 Adminpanel (Státusszerkesztő ablak kód)

Amikor megnyílik a rendelés státusz szerkesztő ablak, az ablak betölti a kiválasztott rendeléshez tartozó adatokat és termékeket. Ha az admin új státuszt választ és elmenti, akkor az új státusz frissül az adatbázisban, az ablak bezárul, és visszajelzi, hogy a mentés sikeres volt.

```
public OrderStatusWindow(OrderModel selectedOrder, DatabaseContext dbContext)
{
    InitializeComponent();
    context = dbContext;
    SelectedOrder = selectedOrder;
    context.orderproducts.Include(op => op.Product).Load();
    OrderProducts = new ObservableCollection<OrderProductModel>(
        context.orderproducts.Where(op => op.OrderId == selectedOrder.Id).ToList()
    );
    DataContext = this;
}

1 reference
private void Mentek_Click(object sender, RoutedEventArgs e)
{
    if (statuszBox.SelectedItem is string newStatus)
    {
        SelectedOrder.Status = newStatus;
        var orderInDb = context.orders.SingleOrDefault(o => o.Id == SelectedOrder.Id);
        if (orderInDb != null)
        {
            orderInDb.Status = SelectedOrder.Status;
            context.SaveChanges();
        }

        DialogResult = true;
        Close();
    }
}
```

## 6. Tesztelés

A teszteket Cypress és Postmanben valósítottam meg. Több dolgot is figyelembe vettem a tesztelésnél. A weboldalon sok lehetőséget megnéztem és teszteltem le. A postman teszteléseknél pedig kéréseket teszteltem le. ***A 2 logót innen használtam fel:***

<https://brandfetch.com/cypress.io>

<https://brandfetch.com/getpostman.com>



# 6.1 Cypress tesztelés kód

Ez a tesztállomány E2E teszteket tartalmaz a webalkalmazás alapvető működésének ellenőrzésére Cypress segítségével. A tesztek lefedik a főoldal betöltését, a bejelentkezés folyamatát, jogosultságellenőrzést a kosár oldalnál, valamint a különböző menüpontokra (Bejelentkezés, Kosár, Rendelések, Rólunk) történő navigációt. A tesztek célja, hogy szimulálják a felhasználói interakciókat és biztosítsák, hogy az alkalmazás megfelelően működjön a felhasználó szemszögéből.

```
/// <reference types="cypress" />

describe('DANKA3D Alap működés teszt', () => {
  it('Megnyitja a főoldalt', () => {
    cy.visit('http://localhost:3000');
    cy.contains('DANKA3D').should('exist');
  });

  it('Navigál a bejelentkezés oldalra a legördülő menüből', () => {
    cy.visit('http://localhost:3000');
    cy.contains('Bejelentkezés').click({ force: true });
    cy.url().should('include', '/login');
  });

  it('Kitölti a bejelentkezés űrlapot', () => {
    cy.visit('http://localhost:3000/login');
    cy.get('input[name="email"]').type('teszt@pelda.hu');
    cy.get('input[name="password"]').type('jelszo123');
    cy.get('button[type="submit"]').click();
  });

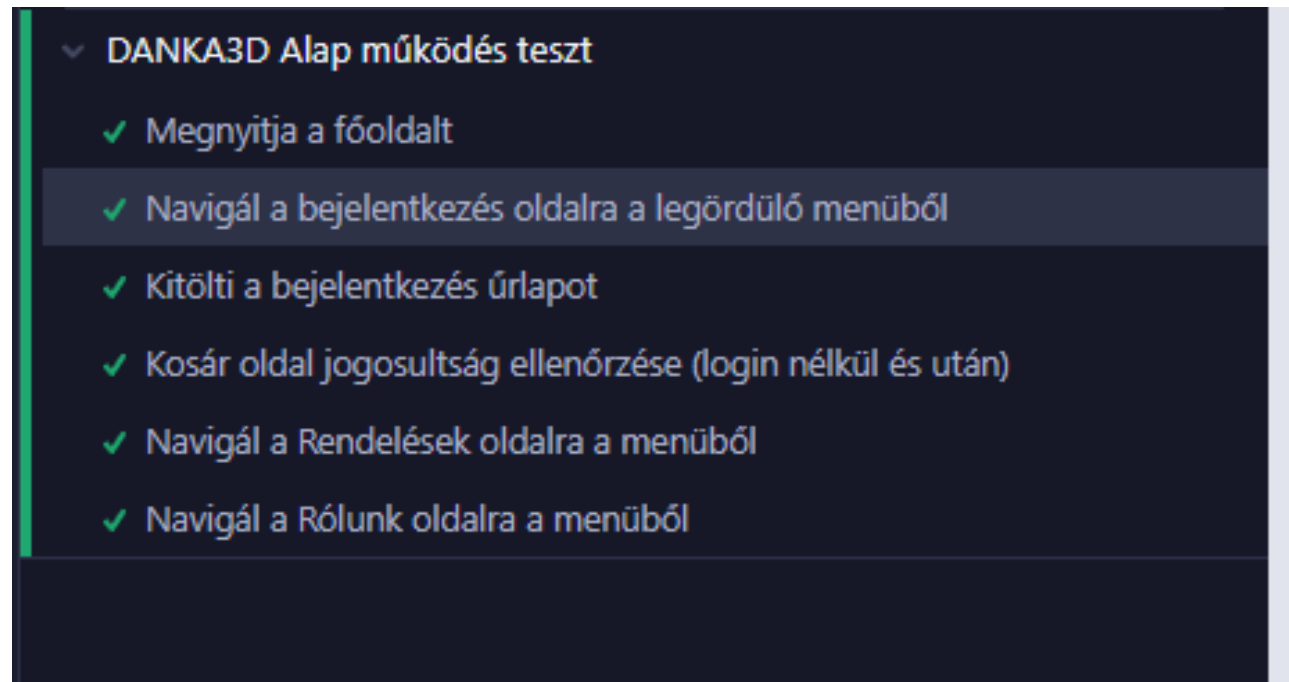
  it('Kosár oldal jogosultság ellenőrzése (login nélkül és után)', () => {
    cy.visit('http://localhost:3000');
    cy.contains('Kosár').click({ force: true });
    cy.get('input[name="email"]', { timeout: 10000 }).should('exist');
    cy.url().should('include', '/login');
    cy.get('input[name="email"]').type('teszt@pelda.hu');
    cy.get('input[name="password"]').type('jelszo123');
    cy.get('button[type="submit"]').click();
    cy.visit('http://localhost:3000');
    cy.contains('Kosár').click({ force: true });
    cy.url().should('include', '/basket');
  });

  it('Navigál a Rendelések oldalra a menüből', () => {
    cy.visit('http://localhost:3000');
    cy.contains('Rendelések').click({ force: true });
    cy.url().should('include', '/orders');
  });

  it('Navigál a Rólunk oldalra a menüből', () => {
    cy.visit('http://localhost:3000');
    cy.contains('Rólunk').click({ force: true });
    cy.url().should('include', '/aboutus');
  });
});
```

## 6.2 Cypress tesztelés

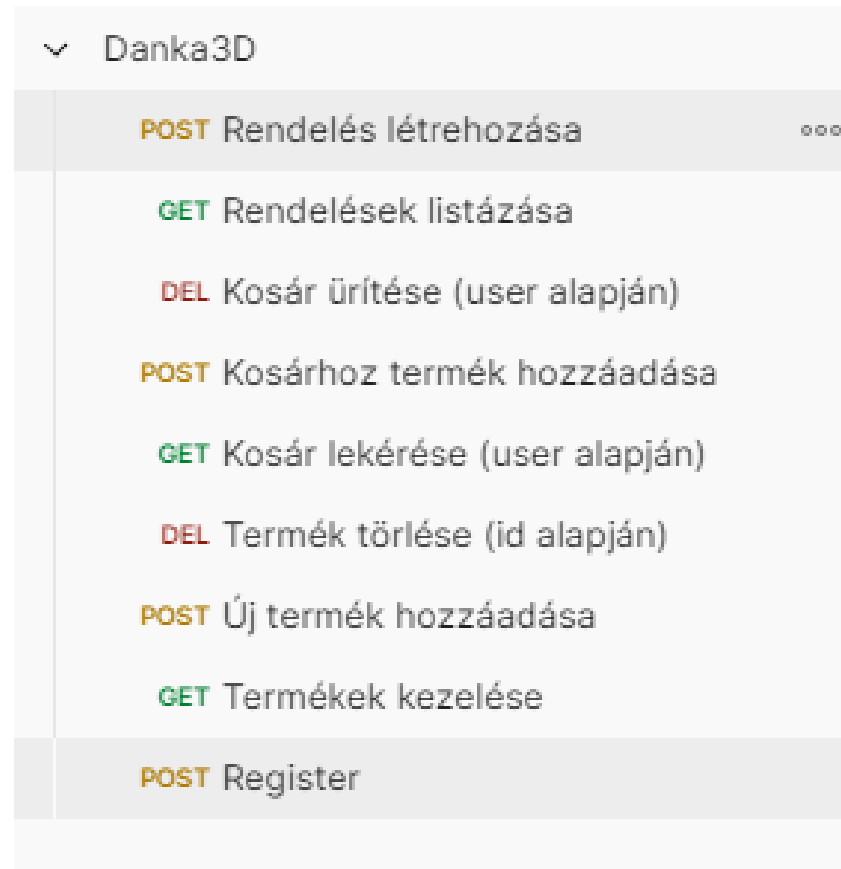
A képen a Cypress tesztfutó felülete látható, ahol a DANKA3D Alap működés teszt nevű E2E tesztcsoporthoz eredményei szerepelnek. Minden teszt sikeresen lefutott, például a főoldal megnyitása, a bejelentkezési oldalra navigálás, az űrlap kitöltése, valamint a Kosár, Rendelések és Rólunk oldalak ellenőrzése is.





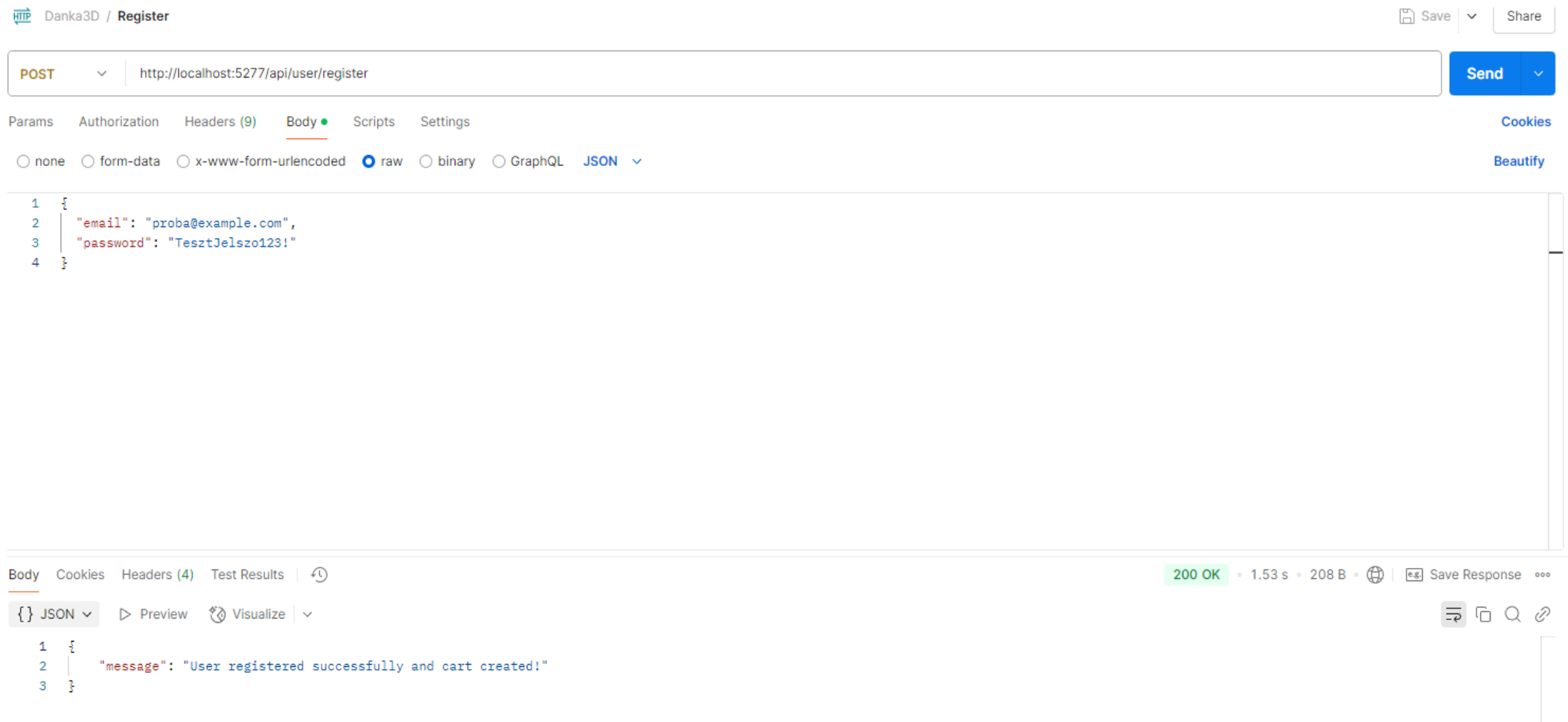
## 6.3 Postman kérések

A képen látható a backendhez tartozó kérések. Postman-ben oldottam meg mivel a programot nagyon könnyű és kényelmes használni és sok kérést lehet vele tesztelni.



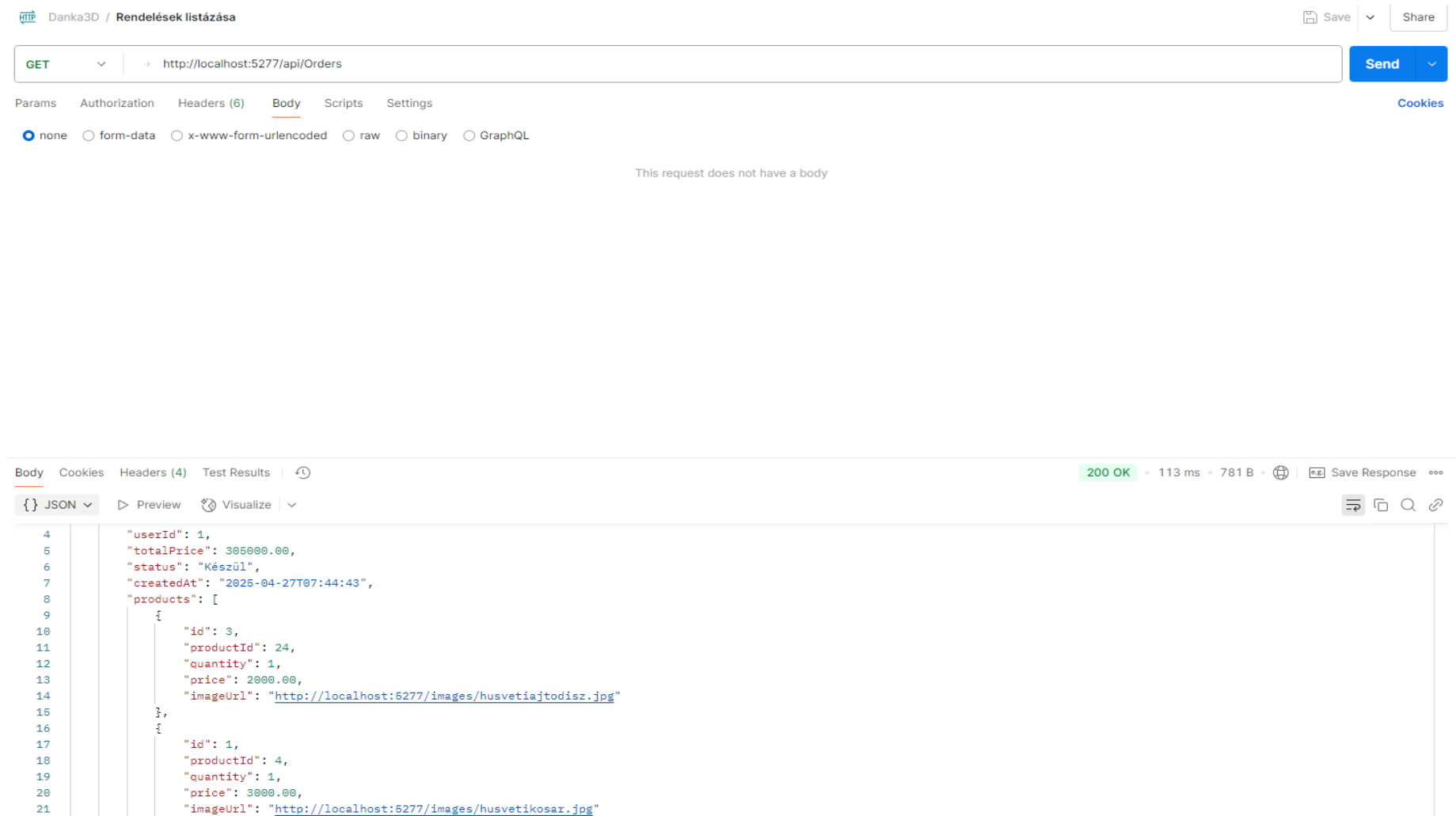
## 6.4 Postman tesztelés

A képen egy POST kérést teszteltem le, mégpedig a regisztrációt. Teszt e-mailt és jelszót használtam, hogy megnézzem rendesen működik e.



# 6.5 Postman tesztelés

A képen egy GET kérést teszteltem le, mégpedig a rendelések listázását. Itt maguk a rendelések jelennek meg, minden adattal.



## 6.6 Postman tesztelés

A képen egy DELETE kérést teszteltem le, mégpedig a kosárban lévő termékek törlése. Törli egy adott felhasználó összes kosárban lévő tartalmát.

The screenshot displays the Postman interface for an API request. At the top, the breadcrumb navigation shows 'Danka3D / Kosár ürítése (user alapján)'. The request method is set to 'DELETE' and the URL is 'http://localhost:5277/api/cart/clear-cart/1'. The 'Body' tab is selected, showing a message: 'This request does not have a body'. Below the request configuration, the response section is visible, showing a '200 OK' status with a response time of 160 ms and a size of 204 B. The response body is displayed in JSON format, indicating that all items have been removed from the cart.

HTTP Danka3D / Kosár ürítése (user alapján) Save Share

DELETE http://localhost:5277/api/cart/clear-cart/1 Send

Params Authorization Headers (6) Body Scripts Settings Cookies

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary ☐ GraphQL

This request does not have a body

Body Cookies Headers (4) Test Results 200 OK • 160 ms • 204 B Save Response

{ } JSON Preview Visualize

```
1 {
2   "message": "All items have been removed from the cart."
3 }
```

# 7 Használati utasítás (localhost-on)

1. XAMPP MySql indítása:
2. dbForge-ban execute-olni kell az adatbázis fájlt, utána ott látszik minden adat.
3. A backend mappában lévő fájlt a zöld gombra kattintva Visual Studio-n belül el lehet indítani, ezután bejön egy swagger környezet.
4. A frontendben a frontend mappán belül a konzolra be kell írni, hogy „npm i”, majd miután ez települt ugyan ide „npm start”
5. Adminpanel indítása:
6. Megnyitjuk az .sln fájlt majd ha a program rendesen betöltött és a build is kész akkor a fent lévő kis zöld nyílra kattintva el lehet indítani a programot. Természetesen ehhez is kell az adatbázis amit dbForge-ban kell elindítani.
7. A cypress tesztelést külön a frontendben kell elindítani terminál ablakban kell egy npm i cypress majd utána egy npx cypress open. Majd amint elő jött a cypress akkor ki kell választani hogy E2E majd a böngésző típusát. És ott majd látunk egy cypress teszt fájlt abban meg lehet nézni a tesztek.
8. A postman-es kérésekhez pedig kell a backend amit szintén Visual Studio-ban a zöld gombra kattintva lehet elindítani. Majd ha a swagger elindult akkor a postmant is el lehet indítani majd meg lehet nézni a tesztek.