KÜLKERESKEDELMI
TECHNIKUM

Fejlesztői dokumentáció

Szoftverfejlesztő és –tesztelő technikus szakma

2025

Készítette: Sövér Benjámin, Tóth Dániel

Tartalomjegyzék

Bevezetés

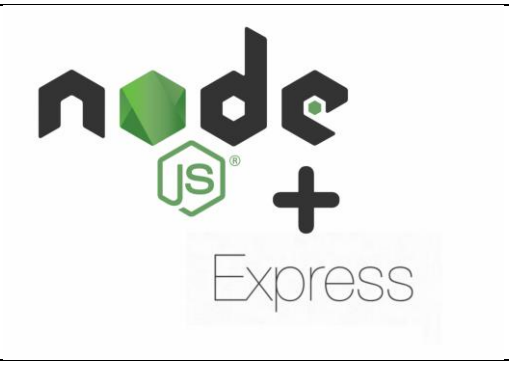
A projekt egy esküvői csokrokat és egyéb mint kerti és dekorációs növényeket árusító magán vállalkozásnak készült, amire az igény az ismeretségi körünkön belül jelentkezett. A cél az volt hogy egy olyan komplett webshopot készítsünk amiben van egyaránt felhasználói és admin felület. A felhasználók szemszögéből a legfontosabb és egyben elvárt funkció az volt hogy az oldalon lévő termékek között lehessen válogatni, kategóriák és típusok szerint egyben lehetőség nyíljon a rendelésre, az üzlet és meghirdetett termékek adatainak a megismerésére. Az Admin oldali funkciók lényege az volt hogy termékeket lehessen feltölteni, szerkeszteni, a felhasználókat és a rendeléseket kezelni.

A Virág Webshop projektünkben olyan alap funkcióknak kellet megvalósulniuk mint a CRUD műveletek (Create, Read, Update, Delete) amelyek működése egy MongoDB alapú adatbázisban zajlik. A Webshop egy Mern Stack Developing néven elhíresült módszerrel lett létrehozva, ahol a backend Node.js keretrendszer alapján, az adatbázis MongoDB, a frontend pedig React keretrendszerrel dolgozik és működik. Az módszer kiválasztásának fő oka a fejlesztő környezetek nyújtotta egyszerűség, gyorsaság, kompaktság és az ezen rendszer által alkotható modern szép kinézetű weboldalakról eredeztethető. Fejlesztésünk során ezen előítéleteink be is bizonyosodtak. Arra jöttünk rá hogy a főként JavaScript alapú rendszerekkel való munka tényleg egy sokkal egyszerűbb és modernebb fejlesztési környezetet adott.

A projektünkhöz sok segítséget sikerült találni különböző weboldalakról, így ha időközben bármilyen probléma is ütköztünk volna akkor általában sikerült olyan módszereket találni amiket felhasználva viszonylag gyorsan orvosolni sikerült az adott problémát.

Technológiák és készítéshez használt programok

Backend

A **Node.js** egy aszinkron JavaScript futtatókörnyezet, amely lehetővé teszi a szerveroldali fejlesztést. Az **Express.js** pedig egy könnyen használható webes keretrendszer, amely egyszerűsíti a webalkalmazások és API-k fejlesztését. Együtt gyors és hatékony megoldásokat kínálnak.	

Frontend

A **React** egy népszerű JavaScript könyvtár, amelyet a felhasználói felületek (UI) építésére használnak. Komponens-alapú megközelítést alkalmaz, lehetővé téve az újrafelhasználható UI elemek létrehozását. A React gyors renderelést biztosít a virtuális DOM használatával, ami növeli az alkalmazások teljesítményét. Kiválóan alkalmas dinamikus, interaktív webalkalmazások fejlesztésére.	

Adatbázis

MongoDB A **MongoDB** egy dokumentum-orientált NoSQL adatbázis, amely rugalmas és skálázható tárolást kínál. JSON-szerű adatstruktúrával dolgozik, így könnyen kezelhetők a változó adatmodellek. Nagyszerű választás nagy mennyiségű adat kezelésére és dinamikus alkalmazásokhoz.	

Kódolási környezet

Microsoft Visual Studio Code A **Visual Studio Code** egy ingyenes, gyors és testreszabható kódszerkesztő, amely számos nyelvet támogat. Hibakeresést, verziókezelést és bővítményeket kínál. Ideális webfejlesztőknek és szoftverfejlesztőknek.	

Webböngészők

Google Chrome A **Google Chrome** egy gyors, fejlesztők számára számos hasznos eszközt kínáló böngésző. Beépített fejlesztői eszközei (DevTools) lehetővé teszik a kód hibakeresését, a teljesítmény optimalizálását és a webalkalmazások tesztelését. Támogatja a legújabb webes technológiákat és könnyen bővíthető különböző kiegészítőkkel.	
Firefox A **Firefox** egy fejlesztők számára is kiemelkedő böngésző, amely erőteljes beépített fejlesztői eszközöket (DevTools) kínál. A JavaScript hibakeresés, a CSS szerkesztés és a teljesítményfigyelés könnyedén elérhetők benne. Támogatja a legújabb webes szabványokat, és nagy hangsúlyt fektet a biztonságra és a magánélet védelmére.	

Tesztelés

Postman A **Postman** egy népszerű API-fejlesztő eszköz, amely lehetővé teszi API-k tesztelését, dokumentálását és monitorozását. Könnyen készíthetők HTTP kérések, és az eszköz segítségével ellenőrizhetők a válaszok és hibák. Támogatja az automatizált tesztelést és a munkafolyamatok optimalizálását API fejlesztéskor.	
Insomnia Az **Insomnia** egy API-fejlesztő eszköz, amely lehetővé teszi RESTful és GraphQL API-k könnyű tesztelését és hibakeresését. Támogatja a HTTP kérések gyors küldését, válaszok megjelenítését és a környezeti változók kezelését. Felhasználóbarát felülete és fejlett funkciói segítik az API fejlesztést és integrációt.	

Felhőalapú média-kezelő szolgáltatás

Cloudinary A **Cloudinary** egy felhőalapú média kezelő szolgáltatás, amely lehetővé teszi képek és videók tárolását, optimalizálását és kezelését. Automatikusan méretezhetők, formátumra konvertálhatók és tömöríthetők a fájlok. Fejlesztők számára API-kat kínál a médiafájlok egyszerű integrálásához web- és mobilalkalmazásokba.	

E-mail kiszolgáló

G-mail A **Gmail API** lehetővé teszi a fejlesztők számára, hogy integrálják a Gmail szolgáltatásait saját alkalmazásaikba. Az API segítségével automatizálhatók az e-mailek küldése, olvasása, címkék kezelése és a fiók kezelésének különböző aspektusai. Biztonságos hozzáférést biztosít OAuth2 autentikációval, és rugalmas lehetőségeket kínál a Gmail fiókokkal való interakcióra.	

Kártyás fizetés

Stripe A **Stripe** egy fizetési platform, amely egyszerűsíti online tranzakciók kezelését web- és mobilalkalmazásokban. Támogatja a kártyás fizetéseket, előfizetéseket és egyéb módszereket. Az API-k könnyen integrálhatók és biztonságosak.	

Deploy

Vercel A **Vercel** egy felhőalapú platform, amely lehetővé teszi webalkalmazások és statikus oldalak gyors telepítését és skálázását. Különösen jól integrálódik a Next.js-el, de más front-end keretrendszerekkel is használható. Az automatikus deploy, a gyors betöltési idő és a globális CDN segíti az alkalmazások teljesítményét.	

Kódolási konvenciók

A kódot Git verziókezelővel használjuk és a Clean Code alapelvei szerint készült el.

Alapkönyvtárak

Frontend

```
├── App.jsx
├── /src
│   ├── /components
│   │   ├── BestSeller.jsx
│   │   ├── CartTotal.jsx
│   │   ├── Footer.jsx
│   │   ├── Hero.jsx
│   │   ├── LatestCollection.jsx
│   │   ├── Navbar.jsx
│   │   ├── NewsLetterBox.jsx
│   │   ├── Policy.jsx
│   │   ├── ProductItem.jsx
│   │   ├── RelatedProducts.jsx
│   │   ├── SearchBar.jsx
│   │   ├── ScrollToTop.jsx
│   │   ├── Title.jsx
│   ├── /context
│   │   ├── ShopContext.jsx
│   ├── /pages
│   │   ├── About.jsx
│   │   ├── Cart.jsx
│   │   ├── ResetPassword.jsx
│   │   ├── Contact.jsx
│   │   ├── DataProtection.jsx
│   │   ├── Delivery.jsx
```

```
│   │   ├── Home.jsx
│   │   ├── Login.jsx
│   │   ├── Orders.jsx
│   │   ├── PlaceOrder.jsx
│   │   ├── Product.jsx
│   │   ├── Products.jsx
│   │   ├── Title.jsx
│   │   ├── Verify.jsx
│   ├── /assets
│   │   ├── assets.js
│   │   ├── about_img.png
│   │   ├── bin_icon.png
│   │   ├── cart_icon.png
│   │   ├── contact_img.png
│   │   ├── cross_icon.png
│   │   ├── dropdown_icon.png
│   │   ├── exchange_icon.png
│   │   ├── hero_img.png
│   │   ├── logo.png
│   │   ├── menu_icon.png
│   │   ├── profile_icon.png
│   │   ├── quality_icon.png
│   │   ├── search_icon.png
│   │   ├── star_dull_icon.png
│   │   ├── star_icon.png
│   │   ├── stripe_logo.png
│   │   ├── support_img.png
```

Backend

```
├── /config
```

```
│   ├── cloudinary.js
│   ├── mongodb.js
├── /controllers
│   ├── cartController.js
│   ├── orderController.js
│   ├── productController.js
│   ├── userController.js
├── /middleware
│   ├── adminAuth.js
│   ├── auth.js
│   ├── multer.js
├── /models
│   ├── orderModel.js
│   ├── productModel.js
│   ├── userModel.js
├── /routes
│   ├── cartRoute.js
│   ├── orderRoute.js
│   ├── productRoute.js
│   ├── userRoute.js
```

Admin

```
├── /src
  ├── /pages
  │   ├── Add.jsx
  │   ├── List.jsx
  │   ├── Orders.jsx
  │   ├── Update.jsx
  │   ├── PopUp.jsx
  │   ├── ManageUsers.jsx
```

```
├── /components
│   ├── Login.jsx
│   ├── Navbar.jsx
│   ├── ScrollToTop.jsx
│   ├── Sidebar.jsx

├── /assets
│   ├── add_icon.png
│   ├── assets.js
│   ├── logo.png
│   ├── order_icon.png
│   ├── parcel_icon.svg
│   ├── upload_area.png
│   ├── user_icon.png
```

Felülettervek

Adatmodell

(Megjegyzés: Ide kéne egy képernyőkép a MongoDB-ről szerintem.)

Végpontok: Összes fájl

Backend

```
├── /routes
│   ├── cartRoute.js
│   ├── orderRoute.js
│   ├── productRoute.js
│   ├── userRoute.js
```

Végpontok: cartRoute.js

Végpont	Metódus	Azonosítás	Leírás
/get	post	authUser	getUserCart
/add	post	authUser	addToCart
/update	post	authUser	UpdateCart

Végpontok: orderRoute.js

Megjegyzés	Végpont	Metódus	Azonosítás	Leírás
Admin funkciók	/list	post	adminAuth	allOrders
	/status	post	adminAuth	updateStatus
Fizetési funkciók	/place	post	authUser	placeOrder
	/stripe	post	authUser	placeOrderStripe
Felhasználó lehetőségei	/userorders	post	authUser	userOrders
Fizetési authentikáció	/verifyStripe	post	authUser	verifyStripe

Végpontok: productRoute.js

Végpont	Metódus	Azonosítás	Leírás
/add	post	adminAuth	addProduct

/remove	post	adminAuth	removeProduct
/list	get	nincs	listProducts
/single/:productId	get	nincs	singleProduct
/update	put	adminAuth	updateProduct

Végpontok: userRoute.js

Végpont	Metódus	Azonosítás	Leírás
/register	post	nincs	registerUser
/login	post	nincs	loginUser
/admin	post	nincs	adminLogin

Általános működés

A REST API http kéréseket fogad, melyek tartalmazzák a műveletekhez szükséges megfelelő adatokat. A kényes műveletek végpontjai védettek, authentikáció szükséges a használatukhoz.

Az adatokat JSON formátumban fogadja és feldolgozza. A vezérlést kontrollerek valósítják meg, minden adatkezelési csoportnak külön kontrollere van, itt történik az adatfeldolgozás.

A kontrollerek modellekkel vannak kapcsolatban, amelyek az adatkezelésért felelősek. Minden adatkezelési csoportnak külön modellje van, itt történik az adatok adatbázisból kiolvasása, illetve az adatok kiírása adatbázisba.

A modellek adatbázis táblákkal vannak kapcsolatban, melyek az adatok tárolásáért felelősek. Az adatbázis táblák adatait a modellek kezelik.

Osztályok

?

Tesztelés

?

Ismert hibák

?

Feljesztési lehetőségek

?