

**Trefort Ágoston Technikum,
Szakképző Iskola és Kollégium
Szoftverfejlesztő - 5421305**

Záródolgozat

WebX

**Készítette: Fekete Károly Richárd 13/A
Konzulens: Ferenczi Tímea**

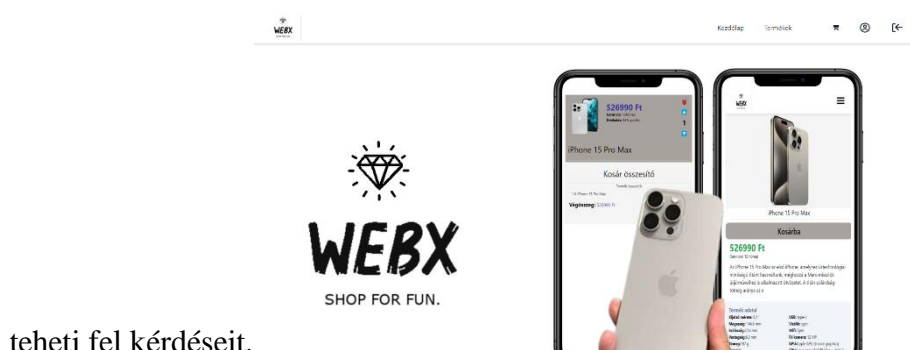
Békéscsaba 2024.

Tartalomjegyzék

A program általános specifikációja	1
Rendszerkövetelmények.....	2
A program telepítése	3
A program használatának a részletes leírása	8
Választott téma indoklása.....	14
Az alkalmazott fejlesztői eszközök	15
Adatmodell leírása.....	16
Részletes feladat-specifikáció, algoritmusok.....	17
Forráskód.....	20
Tesztelési dokumentáció	24
Továbbfejlesztési lehetőség.....	26
Irodalomjegyzék, forrásmegjelölés	27

A program általános specifikációja

A WebX egy telefonokat árusító weboldal, amely legfőbb és még egyetlen funkciója a telefonok eladása. Esetleg később a telefon felvásárlás is szóba jöhet, ha igényt tartanak rá az felhasználók. Egy széles termékkatalógus melyben bizalommal böngészhetnek, olvashatnak a termékekről leírást. Egyszerű böngészést biztosít mindenki számára. Könnyű, átlátható, kezelhető felületének köszönhetően teljes mértékben felhasználóbarát. A rendelés lehetőség beregisztrált felhasználóhoz van kötve, így leszűrve a kamu rendeléseket. A termékeket folyamatos felügyelet alatt tartják az adminok. Ők töltik fel a weboldalt termékekkel, akár személyes tapasztalat beleírására is megvan a lehetőségük. Későbbiekben lehetőség lesz a pénztárcához igazítani a termékek keresését, kifejezetten egy adott termékre szűrni a keresést. Minden termék amit a kosárba helyezünk a felhasználó saját fiókja alapján mentésre kerül, későbbi bejelentkezés esetén is elérhető a felhasználó számára, ezzel egy kényelmi funkciót kiváltva. Fizetést kiváltja egy E-számla amiben leírja részletesen a rendelés részleteit. Sok telefon csupán az eredeti ár töredékéért, akciós áron is vannak termékek, melyeket egy akciós jelzővel ellátva jól el lehet különíteni a normál áras termékektől. Illetve ha valaki a legújabb modellre vágyik, kezdőlapon legalul megtalálja a legújabb termékünket is. Dizájn szempontjából a legfontosabb volt, hogy jól átlátható és könnyen kezelhető legyen a felület. Későbbiekben vélemények, és visszajelzések vannak tervben. Termék leírás minden termékhez tartozik, itt elolvasható, hogy milyen specifikációi vannak a terméknek, miket érdemes tudni megvásárlás előtt. Termék megvásárlása egy egyszerű folyamat, szállítási mód, szállítási cím, és fizetési mód megadásával már indítható is egy vásárlás. Amennyiben a megvásárlás sikeres volt egy letölthető számla fogja írni a pontos adatokat a rendeléssel kapcsolatban. Itt továbbá látható a WebX hivatalos elérhetőségei. Itt lehet tájékozódni a rendeléssel kapcsolatban ha bármi elakadás adódna. Kérjük őrizze meg a számlát minden lehetséges probléma elkerülése végett. Továbbá a **+36 31 785 6067** telefonszámon vagy a **feketekaroly@taszi.hu** email címen



teheti fel kérdéseit.

Rendszerkövetelmények

Hardverkövetelmények

Minimum:

CPU	GPU	RAM	OS	Tárhely	Internet
Intel Core I3-530	Nvidia GeForce 9600GT 1GB Vram	2GB+	Win 10 +	1GB+	Szükséges

Ajánlott:

CPU	GPU	RAM	OS	Tárhely	Internet
Intel Core i5-3300	Nvidia GeForce 560	4GB+	Win 10 +	1GB+	Szükséges

Szoftver követelmények

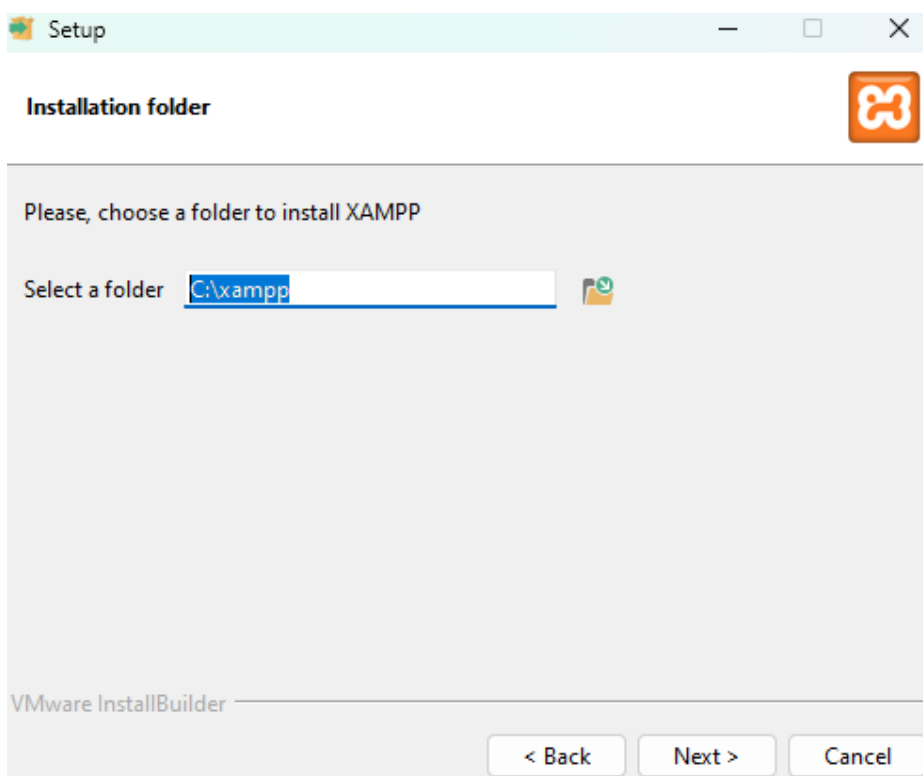
Szükséges szoftverek a program működéséhez: XAMPP, NODEJS. Illetve bármilyen Windows rendszer képes a működésére amire elérhető a XAMMP.

A program telepítése

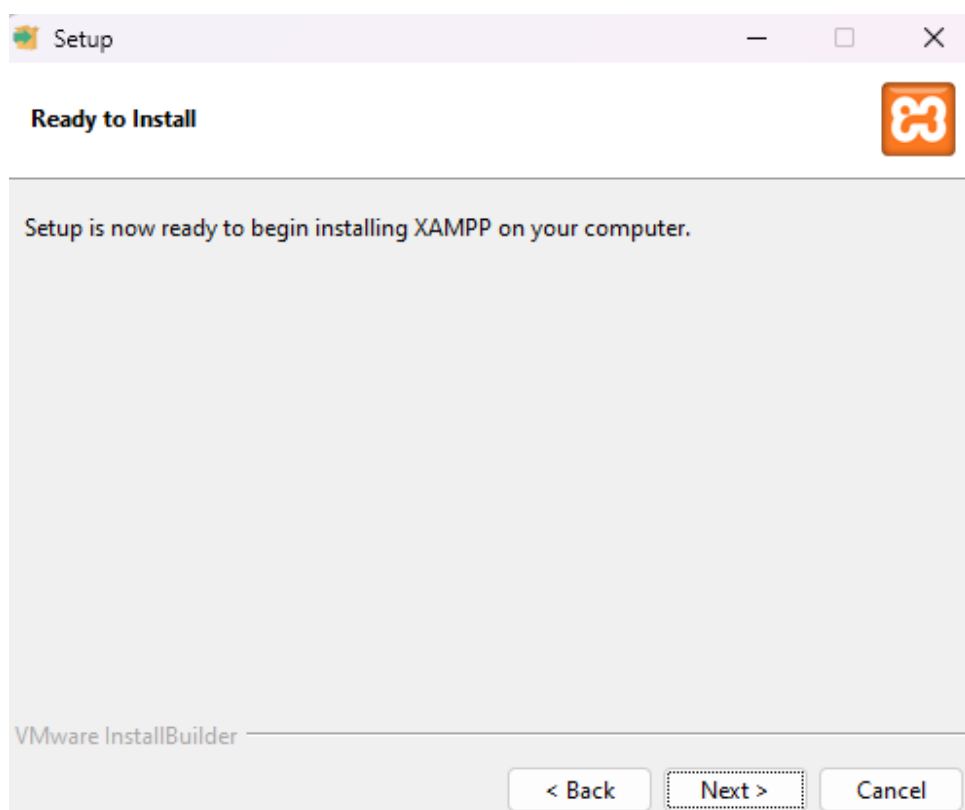
XAMPP:

Leírás: XAMPP egy ingyenes és nyílt forráskódú szoftvercsomag, amely tartalmazza az Apache HTTP szerver, MySQL adatbázis-kezelőt és PHP-t. Verzió: A legfrissebb stabil verzió ajánlott. Letöltés és telepítés: A szoftver letölthető és telepíthető a hivatalos weboldalról: <https://www.apachefriends.org/index.html> vagy a telepítő CD-n található XAMPP mappából.

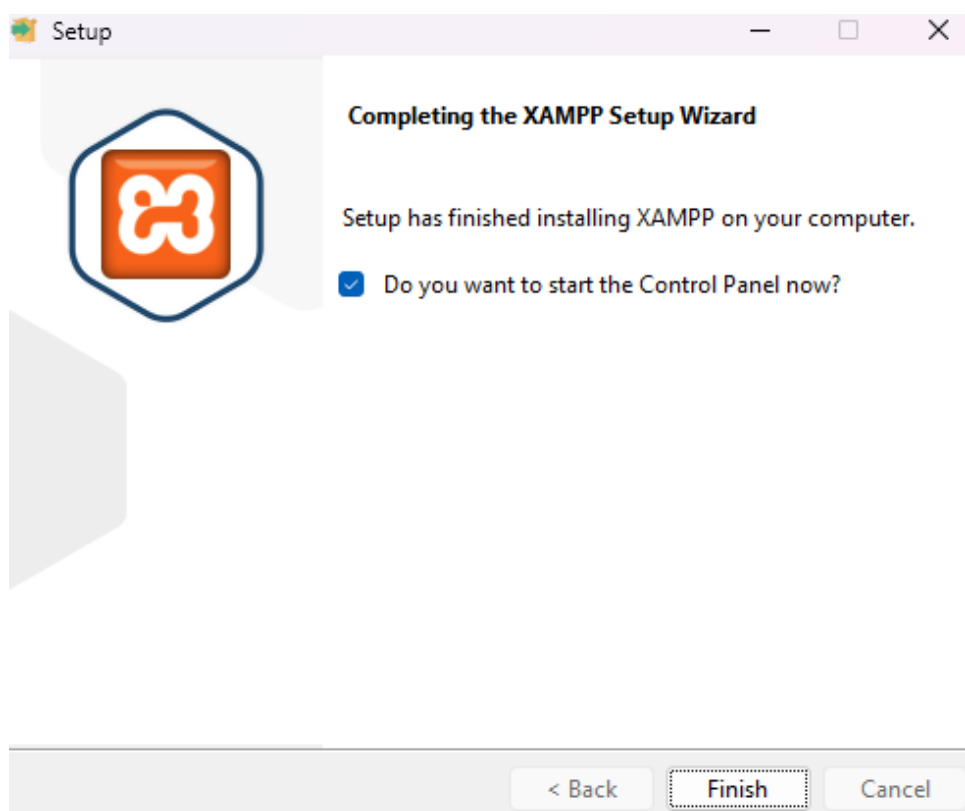
Telepítés: A letöltött .exe fájlt megnyitva, adunk rendszergazdai jogot, megnyílik a setup, itt a Next gombra kattintva folytathatjuk. Itt komponenseket adhatunk hozzá, nem szükséges semmi plusz így irány a Next gomb. Megadjuk a fájl helyét, a legideálisabb az alapértelmezetten hagyni. Ez után next



Majd kiválasztjuk a nyelvet, nyelvet követően jön egy Ready to Install szöveg.



Tovább lépve elkezdje telepíteni a XAMPP-ot. Megvárjuk a telepítés végét, majd a Finish gombra kattintva indítható a XAMPP.

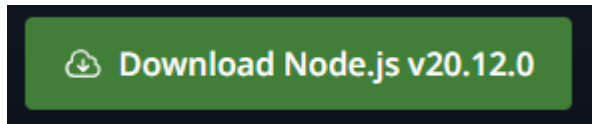


NODEJS:

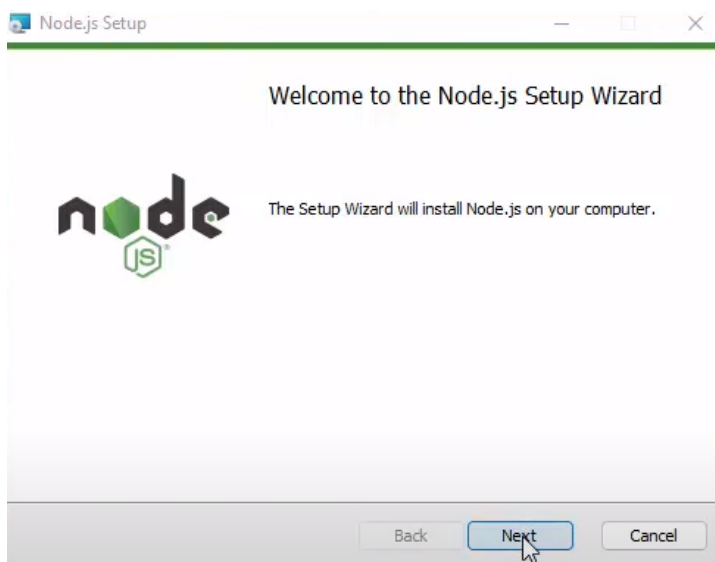
A NodeJS szükséges a program működéséhez, ezzel lehet leszedni a komponenseket melyek életre keltik a projektet(react, prisma stb...).

Telepítése:

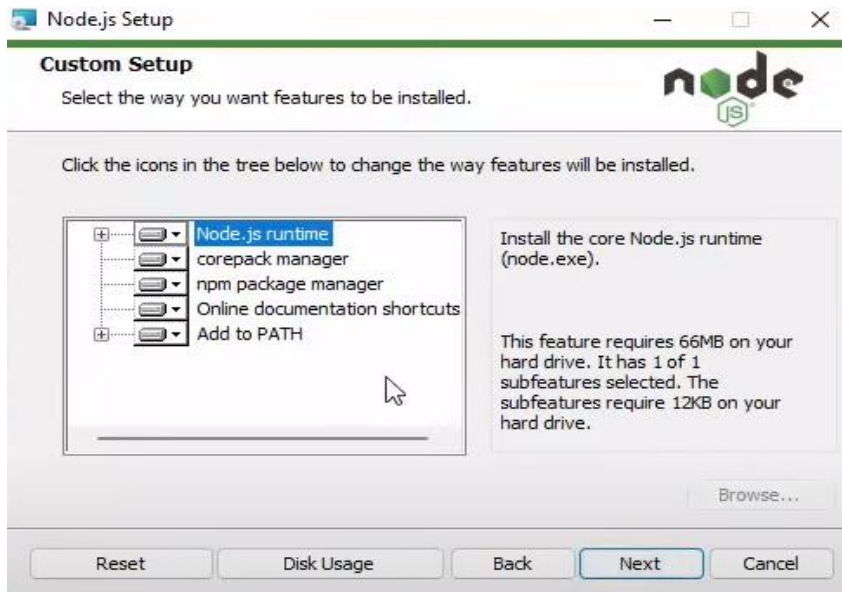
[Node.js — Download Node.js® \(nodejs.org\)](#) oldalról letölthető egy nagy zöld gombra kattintva: (ajánlott a legfrissebb verzió letöltése).



Ezt követően kapunk egy installer .msi filet. Ezt megnyitva elkezdődhet az installálás. Ezt a filet ez a kép fogad:



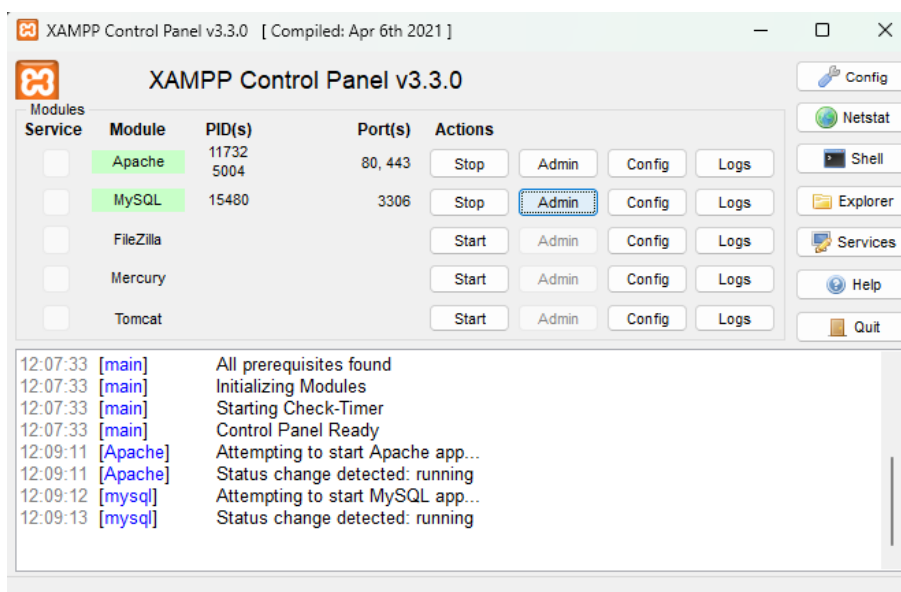
Itt a next-re kattintva haladhatunk tovább. Elfogadjuk a felhasználói feltételeket. Azt követően kiválasztjuk a file helyét (érdemes alapértelmezetten hagyni). Majd egy Custom Setup ba berakhatunk további komponenseket, nincs szükség semmi pluszra.



Ezt követi egy Tools menü, itt sincs semmi plusz dolgunk. Next gombra kattintva haladunk tovább. Ez után kezdődhet a tényleges installálás a Next gombra kattintva. Megvárjuk a telepítést, az után Finish és készen is vagyunk.

Projekt:

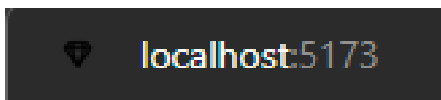
A telepítő CD-n található CD-t a gépbe helyezve, megtalálható minden fájl, ezt kicsomagolva, megnyitva Visual Studioval, vagy egy parancssorral is működtethető. Visual Studion belül a terminál fül legfelül, nyitni kell 2 terminált. Majd el kell navigálni egyikben a backend mappába, másikba a WebX mappába. WebX es terminálba egy “npm i” parancs segítségével telepíthető a projekthez szükséges komponensek. Majd futtatható a frontend része az “npm run dev” parnacsal. Innentől weben a <http://localhost:5173> porton elérhető. A backendes mappa egy kicsit összetettebb folyamat lesz. El kell indítani a XAMPP ot, majd a listába az Apache és a MySQL modulokat kell Startolni.



Itt az admin gombra kattintva átirányít a MySQL admin panelbe. A terminálra visszatérve “npm i” majd “npx prisma migrate dev” parancsal létrehozható a minden szükséges komponens. Ezek után létrejön az adatbázis. A MySQL oldalon a “projekt” nevű adatbázisra kattintva fent az importálás gombra kattintva be kell importálni az “sqlparancs.sql” nevű fájlt, ami szintén a CD-n található.

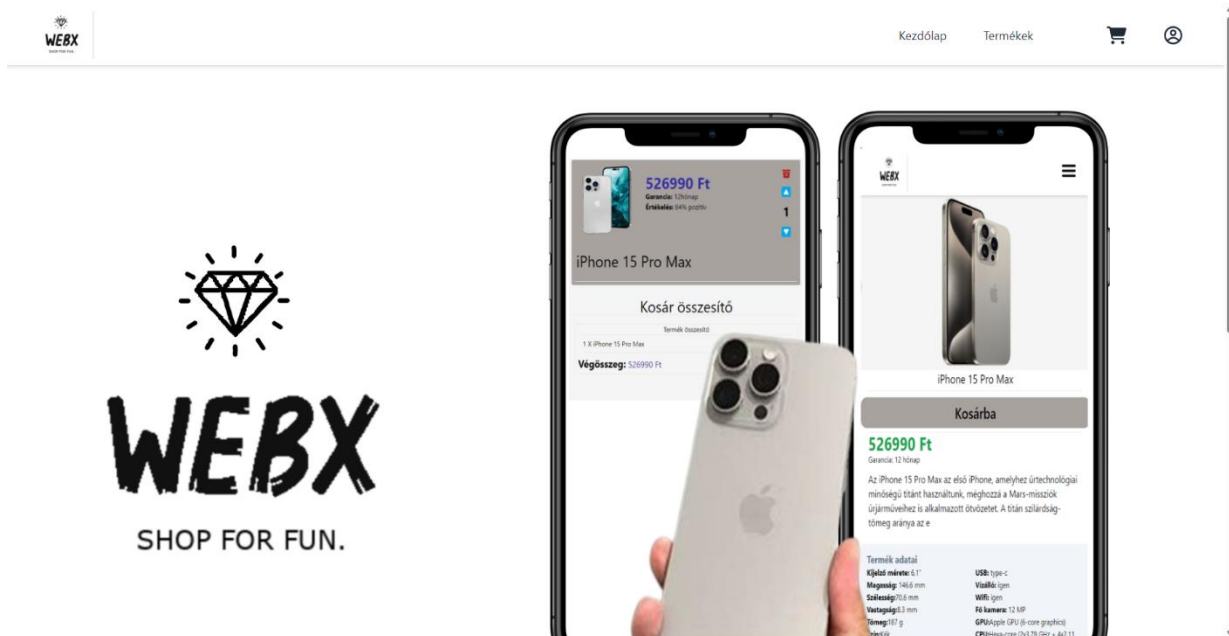


Innentől vissza backend terminálra, ”npm start” parancsal indítható is a backend. Innentől teljes egészében fut a projekt.



A program használatának a részletes leírása

A WebX fel fellépve a következő kép fogad:



Ez a kezdőlap. Mint látható jobb oldal a görgethető sáv, nem csak ennyiből áll, letekerve megtalálhatóak az akciós termékek egy listában. Alatta rögtön a legújabb termék található a kínálatban. Felépítésre fent található egy navigációs sáv ahonnan könnyedén el lehet érni a kívánt oldalakat. Navigációs sáv bal oldalán a cég logója foglal helyet. Nagy tér, majd a navigáció gombok. Alapból a kezdőlapra kezd a felhasználó. Majd ezt követi a Termékek gomb. Egy kosár gomb, és egy fiók menü. Ez a sáv alatt található egy kép, ez a kép egy kis összesítő, hogy mi is várható a weboldalon, természetesen logóval ellátva. Illetve bizonygatja a projekt telefonos nézetre való rezponzivitását is. Alatta az akciós termékek menüsáv. Itt egy oldalra görgethető sávban megjelennek az akciós termékek. Minden termék 1-1 kártyában található. Ezzel könnyítve az átláthatóságot, és weboldal komolyságát. Illetve az akciós jelző, azaz a bal felső sarokban található “%” jel. Ezzel jelzi, hogy a termék az akciós termék kategóriába tartozik. Ez a termékek fölön is feltűnik. Továbbá a kártyákon megjelenik a termék pontos neve, illetve az ára. Következő oldal a Termékek.

Termékek

Itt található meg a készlet összes terméke. Navigációs sáv minden oldalon változatlan, kivéve, ha bejelentkezik a felhasználó, de ezt majd a profil fölön. Dizájnrá egy egyszerű, letisztult kártyákkal ellátott termékek vannak. Bal oldalt pedig egy szűrő foglal helyet, ahol rendezni lehet a későbbiekben ár, név, megjelenés szerint, illetve keresni 1 fix termékre. A

kártya dizájnya megegyezik a kezdőlapon látottakkal. Név, ár, vásárlás gomb. A vásárlás gombra kattintva átkerülünk a termék adatlapjára.

Termék adatlapja

Itt kapunk egy részleges leírást a termék vásárlásával kapcsolatban. Elsőnek egy rövid leírással találjuk magunkat szembe. Majd ezt követi a specifikációk (paraméterek). Itt elolvasható egy-egy termék pontos processzora, videokártyája, memória mennyisége, Kijelző átmérője, pontos magassága, szélessége, tömege, vastagsága. Színe, töltő port bemenete. És a wifi illetve vízállóság szabványa. Továbbá a fő kamera paramétere. Továbbá fogad egy kosárba gomb, amire rákattintva kosárba lehet helyezni a kiválasztott terméket. Amennyiben többször kívánja kosárba helyezni a terméket nyomja meg újra a Kosárba gombot. Így az bekerül a kosárba megnövelt mennyiséggel. Amennyiben véletlen rányomott többször, azt később a kosár fülbe tudja módosítani. Minden termék saját képével van ellátva. Fixen 1 kép található minden termékről, jelenleg. Következő opció a kosár, navigációs sávban a bevásárlókosár ikonra kattintva érhető el.

Kosár

Itt jelennek meg a kosárba helyezett termékek. Amennyiben nincs bejelentkezve a felhasználó egy üzenetet kap, amiben írja, hogy a kosár tartalmához jelentkezzen be. Amennyiben a bejelentkezés már megtörtént úgy a kosárba helyezett termékeket fognak megjeleníteni. Itt lesz lehetőség megrendelni a terméket. Látunk minden lényeges információt amit a rendelés előtt tudni kell. Majd kiválasztjuk a kívánt mennyiségeket, ha valamit mégsem szeretnénk rendelni a kuka ikonra kattintva lehetőségünk van törölni. Maximális rendelhető mennyiség 10 db minden egyes termékből, Amennyiben többre lenne szükségünk az E-mail-es úton megtehetjük. Amin meggyőződünk minden adat pontosságáról rámehetünk a Vásárlás gombra. Egeret ráhúzva átíródik a szöveg Tovább az adatok megadáshoz szövegre. Erre rányomva átirányít egy adatok megadása fülre.

Rendelés adatainak megadása

Itt kiválasztjuk a szállítási módot és a fizetési módot. Majd megadjuk a szállítási nevet és címet. Innen a vásárlás gombra kattintva tovább mehetünk, Ezzel már sikeres a rendelés ha minden adat jól lett megadva. Ha nem hibaüzenetet kapunk, hogy hiányos adat vagy hibás adat. Sikeres rendelés után kiírja a rendelés azonosítót, és két gombot, az egyik a Számla letöltése amivel lehetőségünk lesz a számla letöltésére. Letölt egy pdf fájlt, amibe bele lesz írva a rendelés adatai, mit mennyinek számolt fel, kiírja az áfa nélküli és az áfával ellátott árakat.

Továbbá a rendelő adatait és a cég adatait is. Ha még nincs termék a kosárban akkor egy üzenet jelenik meg amiben leírja, hogy még nincsenek termékek a kosárban. Ezt követi a profil fül, navigációs sávon a profil ikonra kattintva átirányít a bejelentkezéshez.

Bejelentkezés

Innen elérhető a regisztráció is amennyiben nem rendelkezik fiókkal. Ha mégis lenne felhasználói fiókja adja meg az E-mail címét és jelszavát amivel előzőlegesen regisztrált. Regisztrálni a Belépés gomb fölött található kék “Regisztráció” szövegre kattintva teheti meg.

Regisztráció

Itt egy több lépésből álló regisztrációs folyamaton kell végig menni a sikeres regisztrációhoz. Első lépésben egy valódi E-mail címet szükséges megadni. Majd a tovább gombra kattintva ha létező, és még nem regisztrált E-mailt adott meg továbbmegy a felhasználónév illetve a jelszó megadására biztonsági okok miatt a jelszót kétszer szükséges megadni. A jelszó egy minimum 8 karakterből álló betű és vagy szám sorozat is lehet. Majd a “Következő” gombbal amennyiben minden rendben volt a jelszóval átkerül a születési dátum és a telefonszám megadásához. A telefonszám az elérhetőség miatt szükséges, az életkor pedig, hogy biztosak legyünk benne, hogy valódi személyeknek adunk el termékeket. Ez után létrejön a regisztráció. Innentől van egy fiókunk. Későbbiekben bármikor be tudunk jelentkezni a bejelentkezés fülnél az E-mail, jelszó párossal. Bejelentkezés követően az első szembetűnő változás a navigációs sávban megjelenő plusz gombok:



Kezdőlap

Termékek



Profil fül

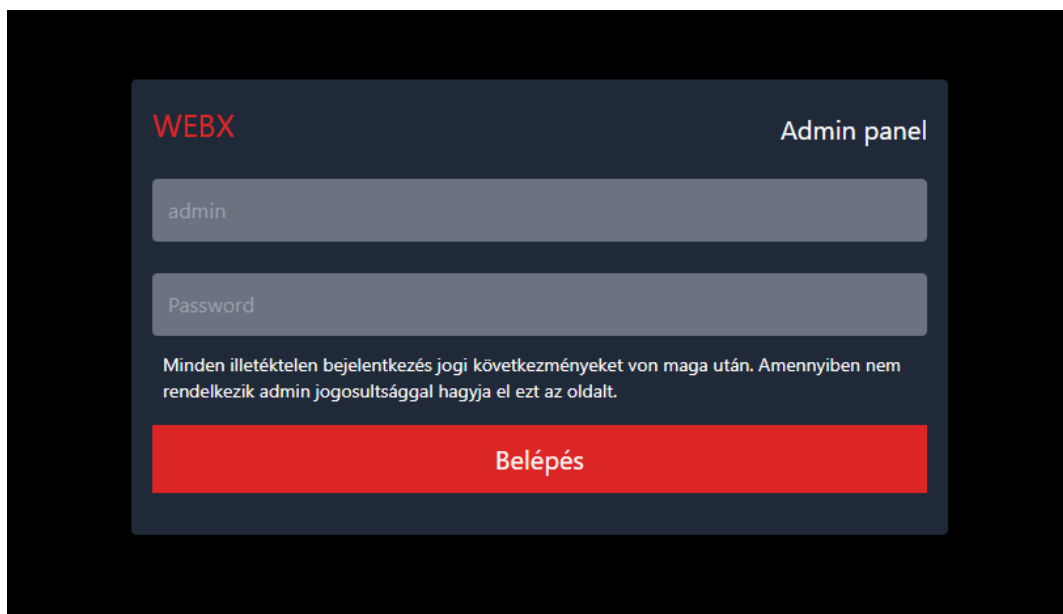
Egy profil gomb, amivel elérhető a profil fül, itt lehet átnevezni, adatokat módosítani. Vagy esetleg tartózkodási címet hozzáadni. Lehetősége van módosítani a felhasználónevét, az e-mail címet nem, mivel ez egy egyedi érték, ez alapján vannak azonosítva a felhasználók. Továbbá a telefonszámát, születési dátumát. Minden adatot helyes értékekkel kell megadni. Azaz a telefonszámot 11 karakteres formátumba, a születési dátumhoz pedig egy pontos napot kell kiválasztani. Pontos lakcímet is lehet megadni, ezzel elmentheti későbbiekre a lakcímet, meggyorsítja a folyamatot.

Kijelentkezés

Másik gomb egy kijelentkezés gomb, erre a gombra kattintva kijelentkezteti a felhasználót.

Admin panel kezelése

Admin panelhez csak és kizárólag a webx adminok kapnak hozzáférést (admin, admin). Így admin joggal nem rendelkező felhasználó ne is közelítse meg a végportot, és ne is olvassa el a következő fejezetet. Admin belépésre a “localhost:5173/admin” port érhető el. Itt szembe találkozunk egy belépő menüvel. Itt tud az admin bejelentkezni a fiókjába. Az admin panelen végig egy fekete szürke oldal fog vezetni. Minden egyes oldal ezt a dizájnt követi tovább.

The image shows a login interface for 'WEBX Admin panel'. It features a dark blue background with a central light blue box. Inside this box, there are two input fields: the first is labeled 'admin' and the second is labeled 'Password'. Below these fields is a line of text in Hungarian: 'Minden illetéktelen bejelentkezés jogi következményeket von maga után. Amennyiben nem rendelkezik admin jogosultsággal hagyja el ezt az oldalt.' At the bottom of the box is a prominent red button with the white text 'Belépés'. The top left of the box has the 'WEBX' logo in red, and the top right has the text 'Admin panel' in white.

Admin fiókba bejelentkezés után megnyílik az admin dashboard.

Dashboard

Itt van lehetősége az adminoknak, hogy lássák az oldalon történő aktivitásokat, különböző diagrammok formájában. További lehetőségek az admin panelon “Üzenetek”, “Eladások”, ”Felhasználók”, ”Termék hozzáadása”, ”Termék eltávolítása”, ”Paraméter szerkesztése”. Ezekkel a fülekkel lehet navigálni a weboldalon . Az Üzenetek gomb átirányít az admin chat részére. Itt tudnak egymás között üzengetni az adminok.

Admin chat



Itt lent az üzenet írása beviteli mezőbe írva van lehetőségünk megírni a kívánt üzenetet. Majd a jobb oldalon található küldés gombbal küldhető el az üzenet. Kiírja hogy ki, mit és mikor írt ki. Így van lehetőség beszélgetni két adminnak egymás között. Következő az eladások menü.

Eladások

Itt láthatóak az eddigi rendelések. Egy egyszerű táblázat szerű nézetben kilistázza a rendeléseket. A rendelés összes adatát visszaadja, felhasználó nevét, szállítási címének ID értékét, rendelés dátumát, rendelt termék nevét, mennyiségét, árát. Következő lehetőség a Felhasználók fül.

Felhasználók

Amiben a felhasználók listája található. Itt látják az adminok, hogy hány regisztrált felhasználó van. A neve, Email címe, telefonszáma, születési dátuma. Ez után a Termék menü, ezen belül elérhető a termék hozzáadása, termék törlése és a paraméterek szerkesztése. Ezekkel a gombokkal lehet navigálni. Átírányítanak az adott névnek megfelelő oldalra. Első a Termék hozzáadása.

Termék hozzáadása

Itt van lehetőség új termék felvitelére, egy egyszerű fekete, szürke dizájn fogad. Egy termék neve, leírása, ára, eldöntés, hogy akciós-e, az akció értékét. Illetve egy kép beszúrása a termékről. Nincs semmilyen visszajelzése, hogy képet töltünk fel. Minden adat megadása kötelező. Minden adat megadása után rányomhatunk a feltöltés gombra, ezzel amennyiben minden adat megfelel kapunk egy „Sikeres feltöltés” szöveget.

Termék törlése

Amennyiben elrontottunk valamilyen adatot átléphetünk a termék törlése gombra, itt van lehetőség törölni a terméket. Amennyiben később vesszük észre a hibát, és már rendelt valaki a termékből sincs probléma, mivel minden adatot ahol megtalálható a termék kitöröl. Ennek az oldalnak csak ennyi a funkciója. Amennyiben viszont sikeresen hozzáadtuk a terméket átmegyünk a Paraméter hozzáadásához.

Paraméter hozzáadása

Itt lesz egy kiválasztó menü, amiben kiválasztjuk a kívánt terméket. Majd itt megadjuk az összes kívánt paraméterét ami a termékre jellemző. Kijelző mérete inch-ben megadva. Magasságát, szélességét, vastagságát egy számmal cm-ben megadva. Tizedes érték esetén '.'-al elválasztva. Majd a tömegét grammra. A színét, USB portját. Vízálló eszköz esetén a vízállóságát, elég egy igen vagy nem. Wifi értékét, ide is igen, nem. Fő kamera MP-ben megadott értékét. Az eszköz processzorát, videokártyáját. Majd a Paraméter hozzáadása gombra kattintva feltölthetőek az adatok. Bármelyik adat hiányzik egy hiba üzenetet kapunk, hogy „hiányos adatok!”. Ha minden megfelelt egy „Sikeres paraméter hozzáadása” üzenetet kapunk.

Választott téma indoklása

Webshopokból alaptól sokszor rendelek, mivel nagyon megkönnyítik a vásárlást. Egyszerűbb weben megkeresni és házhoz rendelni bármilyen terméket, mintsem elmenni a boltba, megkeresni és hosszas időt a kasszába tölteni. A webshop számomra egy hatalmas kényelmi előre lépés. Illetve ha saját termékekről van szó mi sem jobb megoldás a termékek árusítására mint egy webshopban árulni őket. A szívem a webshop felé húzott, így ezt a témát találtam a legjobbnak. Egy egyszerű, de mégis összetett, nagyon sokfelé túlgondolható téma a webshop. A végtelenségig lehetne növelni az adatbázisát, a termékek mennyiségét, paraméterek hosszúságát. Sok gondolkodás volt mire ténylegesen rájöttem, hogy a webshop áll hozzám legközelebb. Sikerült egy teljesen automatizált irányba elhúzzam ezt az egész projektet, díszíteni egy számlával és kiszínezni. Termékek szempontjából volt kérdéses , hogy pontosan mit is árusítson a webshopom. Többszöri újratekésztés, többszöri tervezgetés és hosszas gondolkodás után megtaláltam a hozzám legközelebb álló termékeket, a mobiltelefonokat. Volt tervben , hogy minél több fajta terméket árusítsak, de ez megbukott mivel még most is a termékek paramétereit és új-új adatbázis táblákat írnék az újabb termékekhez. Következő a bútorok lettek volna, de az egy unalmas téma, senki nem akar bútorokat venni. Így elektronikai cikkekre gondoltam. Elsőre itt is túl sok ötletem lett volna, külön fül monitornak, külön CPU, GPU, billentyűzetek stb. Így egy adott termékre gondoltam, ezzel egyben a mobiltelefonokra esett a választás. Céлом elérni egy olyan webshopot, ami tökéletes szintre viszi a vásárlói élményt, állandó visszajelzésekkel, véleményekkel, és folyamatosan tisztába lenni a legfrissebb hírekkel a piacon, tisztába lenni a telefon minden apró részletével, ami jelenleg nem sok webshopnál látható.

Az alkalmazott fejlesztői eszközök

Felhasznált fejlesztői eszközök: A projekt alapötlete egy webes alkalmazás amely HTML alapon íródott. Segítséget nyújtott dizájn szempontjából a CSS, funkcionalitás miatt pedig a JS. A PHP nyelvet próbáltam elengedni, elkerülni. Helyette express JS környezetbe íródott a program backendje. Mind ezt Visual Studio Code-ba kezdtem el írni. Frontendhez Reactet backendhez expressst használtam. Az adatbázist a XAMPP biztosította. Adatbázis kezeléshez prisma-t használtam, így lett egy saját API okat tartalmazó backend. Továbbá egy NodeJs szükséges a pluginok installálásához.

Backend szinten felhasznált pluginok:

1. Prisma : Ez felelt az adatbázis és az API ok megírásáért.
2. Argon2 : Jelszó titkosításához.
3. Cors : feloldja a webes kapcsolatot, engedélyezi az API ok működését.
4. Jsonwebtoken : felhasználóhoz tokent rendelt.
5. Multer : képek feltöltéséért felelt.
6. Uuid : Egyedi azonosító a rendelés azonosítóhoz stb.
7. Nodemon : Állandóan frissíti a szerveret.

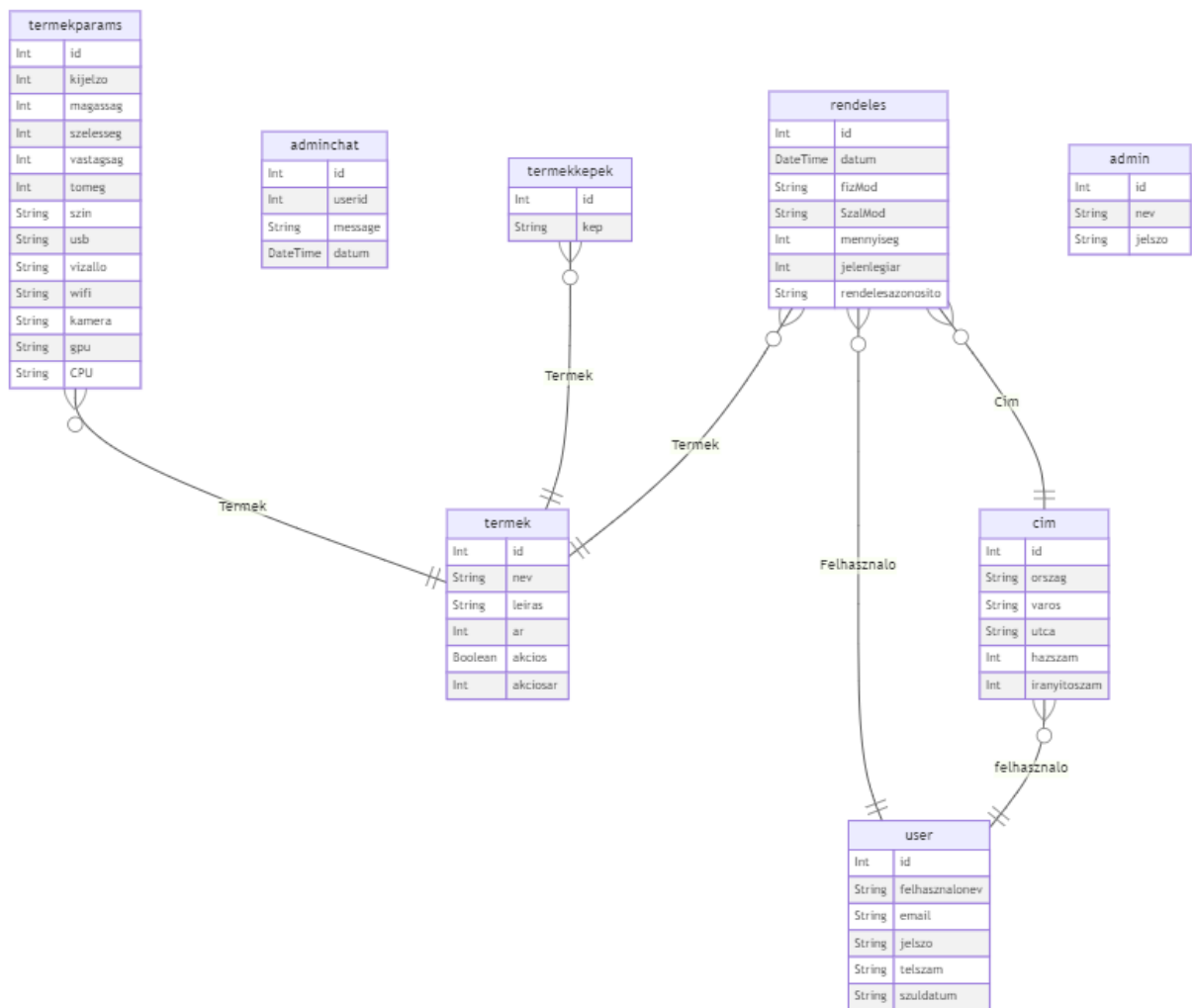
Frontend szinten használt pluginok:

1. Axios : Axios segítségével kellett a fetch lekérdezések megírva.
2. React : React js re épült az egész projekt.
3. React-data-table-components : Ezzel lettek a táblázatok megformázva.
4. React-hot-toast : Felugró üzenetekért felel.
5. React-icons : ikonok működését hozza életbe.
6. React-router-dom : a navigációt érvényesíti.
7. Uuid : egyedi azonosító létrehozásához.
8. Jspdf : a számlához.
9. Vite : A react Vite csomag lett felhasználva a szerkezet létrehozásához.
10. Tailwind : Minden stílus tailwindba íródott meg.

Visual Studio ban készült a projekt, a dokumentáció Wordben íródott, és a PPT PowerPoint-ban jött létre.

Adatmodell leírása

Az adatbázist prisma segítségével csináltam, ez megkönnyíti a folyamatot. Sokkal leegyszerűsíti a folyamatokat. Első kiindulási alap a felhasználó tábla volt. Itt tárolja a felhasználó adatait. Minden felhasználóhoz van kötve egy cím tábla, alapértelmezetten null értékekkel hozza létre. Későbbiekben, ha rendel a felhasználó ez íródik felül. Termékek táblában az termékkatalógus összes terméke található. Minden termékhez rendel 1 termékparams és egy termékképet. Termékképben a kép elérhetősége található. Termékparams-ba pedig a termék pontos paraméterei. Ezeket olvassák ki későbbiekben a lekérések. A rendelések tábla tartalmazza a rendelés össze értékét. A felhasználó adatait, címét, és a rendelt termék összes adatát, továbbá a mennyiségét és a rendelés időpontját, időpontjában levő adott árát. Az admin tárolja az admin belépő adatait. Továbbá az adminchat ami egy üzenőfal az adminok számára.



Részletes feladat-specifikáció, algoritmusok

Navigálás:

```
function App() {
  return (
    <>
      <BrowserRouter>
        <ArContext>
          <UserContext>
            <FizetesContext>
              <Toaster />
              <Routes>
                <Route index element={<HomePage />} />
                <Route path='/home' element={<HomePage />} />
                <Route path='/login' element={<Belepes />} />
                <Route path='/signup' element={<SignUp />} />
                <Route path='/products' element={<Products />} />
                <Route path='/info/:id' element={<TermekOldal />} />
                <Route path='/profile' element={<Profiles />} />
                <Route path='/logout' element={<Logout />} />
                <Route path='/kosar' element={<Kosar />} />
                <Route path='/placeorder' element={<PlaceOrder />} />
                <Route path='/success/:id' element={<SuccessOrderPage />} />
                <Route path='/admin' element={<AdminPanel />} />
                <Route path='/admindashboard' element={<AdminDashBoard />} />
                <Route path='*' element={<NoPage />} />
              </Routes>
            </FizetesContext>
          </UserContext>
        </ArContext>
      </BrowserRouter>
    </>
  )
}
```

A weboldal navigálására a React Router szolgál. A React Router komponens alapú navigálást biztosít az egész weboldalon, ez azt jelenti, hogy a felső url cím mindig szinkronban van a weboldallal, és a megfelelő oldalakat vagy komponenseket jeleníti meg. Egész komponenseket is megtud jeleníteni, én például a felső menüsört (Navbar) beleírtam a BrowserRouter komponens közé és így minden oldalon megjelenik. A Link komponens felel az oldalon történő tényleges navigációért. A Link komponens használata során meghívjuk a következő oldalt az egész weboldal frissítése nélkül, ennek az egyik előnye, hogy a weboldalon számos értesítést használtam a React-hot-toast nevű komponenssel és ha nem a Link komponenst használnám a navigálásra akkor az értesítések a navigáció használatát követően nem jelennének meg, mivel az egész weboldal újra betölt.

Termékek API:

```
const [termek, setTermek] = useState(null)
useEffect(() => {
  const termekContainer = document.querySelector("#termek");
  const getAllProducts = async () => {
    const response = await fetch('http://localhost:8000/product/all');
    const data = await response.json();
    setTermek(data)
  }
  getAllProducts();
}, [])
```

A képen látható lekérdezés kiolvassa egy saját API ból az összes terméket. Majd ezt beállítja egy termékek változóba. Ez a lekérdezés visszatér a termék tábla összes adatával, a termékhez kapcsolt kép táblával és a termékhez kapcsolt paraméter táblával. Minden termékkártyán megjeleníti a termék nevét, árát, amennyiben akciós kivonja az árból az akció mértékét, és jelzi a bal felső sarokban egy ikon, hogy akciós a termék. Kiolvas minden adatot, és ezek alapján jelenik meg minden adat a termékről.

Kosár:

A kosár alaphoz egy Cart Contextből jött létre. Itt tárol egy végösszeget, egy egységárat és a termék nevét. Későbbiekben a kosárba helyezve egy terméket hozzá ad hozzá, amennyiben a termék már a kosárban van így megnöveli a mennyiség értékét 1-el. A Ezt mindet a localStoragebe menti. A kosár page innen olvassa ki az adatokat. Átadja a képet, termék nevet, árát, akciót, leírást. Ezek alapján összerakja a kosár oldalt.

```
import { createContext, useMemo, useState } from "react";

const arContext = createContext();

export const ArContext = ({ children }) => {
  const [ar, setAr] = useState(0);
  const [total, setTotal] = useState(0);
  const [cart, setCart] = useState(null);

  useMemo(()=>{
    setCart(JSON.parse(localStorage.getItem('cart')) || []);
  },[])

  useMemo(() => {
    localStorage.setItem('cart', JSON.stringify(cart))
    setTotal(cart?.reduce((all,item)=> all = all + (item.ar * item.darab),0))
  }, [cart])
  return <arContext.Provider value={
    {
      ar,
      setAr,
      total,
      setCart,
      cart
    }
  }>{children}</arContext.Provider>
}

export default arContext
```

```
const {cart,setCart}=useContext(arContext);
const kosarba = () => {
  const isItemInCart= cart?.find((item)=>item.id === data.id);
  if (isItemInCart) {
    setCart(
      cart.map((cartItem)=>
        cartItem.id === data.id
          ? {...cartItem, darab: cartItem.darab>=10? 10 : cartItem.darab + 1, kep:data?.termekkepek.kep }
          : cartItem
        )
    )
  } else {
    setCart([...cart,{ ...data,darab:1}])
  }
  toast.success("Termék sikeres a kosárhoz adva!");
}
```

Képfeltöltés:

Az adatbázisban nem konkrétan a képeket tárolom, hanem csak is az elérési útvonalukat. Ennek az az előnye, hogy a képek lekérésére lehet írni egy api szerű lekérdezést. Egy adott termékhez egy képet lehet feltölteni még. Ekkor a „termék id”-hez csatolt kép elérési útvonalával fog megjelenni. A képeket a backend mappán belül az uploads mappában tárolom. Erre a multer nevű ingyenes csomagot használom. Ez tűnt a legjobbnak. A képek uuid egyedi azonosítóval vannak ellátva. Ezzel elkerülve hogy két kép ugyan azon a néven fusson. Multer csomag segítségével lehet kép méretét, felbontását, pixel számát lehet állítani. Én ezeket alapon hagytam. Az API végeredményben visszaadja a termékid és a kép útvonalát. ezt használom fel a kép megjelenítéséhez. a kép src tulajdonságát az api eredményére állítom be. Kép feltöltés után a képet áthelyezi az uploads mappába az uuid csomag által megadott néven. Ezt a nevet viszi fel adatbázisba. Hivatkozáshoz szükséges még a /uploads/ végpont.

```
INSERT INTO `eleresek` (`id`, `TmkID`, `kep`) VALUES
(1, 1, '4292f893-ab12-40a0-9989-a3fa02f85612.jpeg'),
(2, 2, '9d3dc8fa-09bd-4b91-a47f-646df37b17a4.jpeg'),
(3, 3, 'c24318a9-e365-4494-a756-b3396b517405.jpeg'),
(4, 4, '6cfdd66e-34ca-44cf-a7b6-da3cd21f7949.jpeg'),
(5, 5, '25b621bf-ce23-499e-9ba7-717754c447a6.jpeg'),
(6, 7, 'f23dcd55-c24d-4732-b12e-d441beba8e82.png'),
(7, 8, '00821486-2c49-4829-bade-ce5e01728875.png'),
(8, 9, '53cb3f06-d9d1-4871-80a7-5d9c577ee788.png'),
(9, 10, 'daaae2a0-d549-4638-90a0-3786758d2dc1.jpeg'),
(10, 6, 'c7199b80-c104-4337-8ee9-0a2edb9a91cd.png');
```

1. így néz ki adatbázisban a képek elérési útja.

Forráskód

Navigációs sáv:

A navigációs sávom az oldal legtetején található. Fix pozícióban van, így tekerés esetén is a helyén marad. A react egyik sajátossága a useState, ezzel lett létrehozva egy MenuBar nevű változó, ennek az értékét módosítom gomb kattintásra, így van megoldva a telefonos nézetre a legördülő sáv. A gombot megnyomva lenyílik egy menü a navigációs gombokkal. Gépes nézetben a gombok állandó helyen maradnak, az navigációs sáv jobb oldalán. Ezeket a gombokat is kódból hozom létre. Van egy tömb ami tárolja a gombok linkjét, nevét. Egy függvény bejárja ezt a tömböt ami létrehozza egyesével a gombokat.

```
let [MenuBar, setMenuBar] = useState(false);
return (
  <>
    <div className='w-full fixed bg-white z-50 h-24 top-0 left-0 shadow-md'>
      <div className='md:flex md:justify-between py-4 px-7 md:px-10'>
        <div className='text-2xl flex items-center border-gray-400'>
          <img src='../public/logo.png' className='w-24 border-r-2' />
        </div>
        <div onClick={() => { setMenuBar(!MenuBar); }} className='text-5xl absolute right-8 top-6 cursor-pointer md:hidden'>
          <ion-icon name={MenuBar ? 'close' : 'menu'}></ion-icon>
        </div>
        <ul className={`md:flex md:items-center pb-12 md:pb-0 absolute md:static bg-white shadow-md md:shadow-none md:z-auto z-[-1] left-0 w-full md:w-auto pl-9 md:pl-0 transition-all duration-500 ease-in ${MenuBar ? 'top-20 opacity-100' : 'top-[-490px]'} md:opacity-100 opacity-0`}>
          {
            Links.map((gombok) => (
              <li key={gombok.id} className='md:ml-8 text-lg pr-6 my-7 md:my-0'>
                <Link to={gombok.link} className='text-gray-800 hover:text-gray-400 cursor-pointer duration-500'>{gombok.name}</Link>
              </li>
            ))
          }
        </ul>
      </div>
    </>
  )
)
```

Termék kártya:

A termékek kártya egy olyan komponens ami bekapja a termék adatait. Ez alapján építi fel a kis kártyát, egy teljes linear-gradients kártyát. Ha akciós a termék feltűnik egy “%”-jel a bal felső sarokba. Kirakja a termék képét felül, ez alatt a termék neve, egy eldöntő függvény ami ha akciós a termék kivonja az akció mértékét és kiírja az árát. Ez alatt egy vásárlás gomb ami átirányít a termék adatlapjára.

```
const TermekCard = ({ termék }) => {
  return (
    <div className='main hover:scale-95 transition flex flex-col p-5 gap-5 rounded-2xl md:ml-8 md:pb-10 drop sm:w-1/2 md:size-80 md:gap-1 size-11/12 shadow-2xl justify-center items-center bg-gradient-to-t from-stone-500'>
      <div className={`w-full ${termék.akcios ? "" : "hidden"} `}>
        <AiOutlinePercentage title='Akciós' className='absolute z-50 size-10 hover:animate-pulse flex justify-end items-end' />
      </div>
      <div className='kep'>
        <img src={`http://localhost:8000/uploads/${termék.termekkepek[0].kep}`} className='size-60 md:size-40 rounded-xl hover:scale-110 transition drop-shadow-2xl' />
      </div>
      <div className='szoveg flex flex-col gap-2 w-full'>
        <span className='font-bold flex w-full items-center justify-center text-2xl'>{termék.nev}</span>
        <span className='flex w-full text-center items-center justify-center'>
          {Math.round(termék.akcios ? termék.ar - (termék.ar * (termék.akciosar / 100)) : termék.ar)} Ft
        </span>
      </div>
      <div className='buy w-full'>
        <Link to={`/${info}/${termék.id}`} className='flex hover:font-bold hover:bg-stone-300 rounded-lg p-2 justify-center items-center bg-stone-200 w-full font-medium'>Vásárlás</Link>
      </div>
    </div>
  )
}
```

Vásárlás véglegesítés Modal:

Ez egy lenyíló menü a vásárlás véglegesítésénél. Itt megjelennek a kosárba helyezett termékek képei, nevei, és az egységáruk. Majd az alján írja a végárat is.

```
<div className='fixed flex-col right-0 top-24 z-10 flex bg-sky-200 w-64 justify-center items-center ring-2 rounded-b'>
  <div onClick={() => { setarrow(!arrow) }} className='flex flex-col justify-center items-center'>
    <h1>Termékek <span className='text-xs align-top'>{cart.length}</span></h1>
    <h1>{arrow ? <IoIosArrowUp /> : <IoIosArrowDown />}</h1>
  </div>
  <div className={` ${arrow ? "flex" : "hidden"} flex-col `}>
    {
      cart && cart.map((termék, index) => (
        <div key={index} className='flex'>
          <img src={`http://localhost:8000/uploads/${termék.termekkepek[0].kep}`} className='w-10 h-10' />
          <div>
            <a href={`/${info}/${termék.id}`} key={index}>{termék.nev}</a>
            <h2 className='text-indigo-800'>
              {Math.round(termék.akcios ? (termék.ar - (termék.ar * (termék.akciosar / 100))) * termék.darab :
            </h2>
          </div>
        </div>
      ))
    }
    <hr />
    <div className='-mt-0'>
      <h1 className='text-indigo-800 text-xl'>{Math.round(vegar)}ft</h1>
    </div>
  </div>
</div>
```

Termék fül bal oldali kereső sáv:

Itt lesz lehetőség a szűrő használatához. A bal oldalt lévő nyílra kattintva kijön balról egy sáv. Kinyitva lesz lehetőségünk keresni egy adott termékre név alapján, beállítani a minimum

és a maximum árat. És listázni lehet az összes termék szerint, vagy csak az akciós termékek szerint, továbbá tervben lesz egy rendezés ár szerint növekvő, csökkenő sorrendbe. A Keresés! gombra kattintva frissül a lista, és aktiválja a szűrőket.

```
<div className={flex z-40 ${oldalBar ? 'w-14' : 'w-1/3'}} >
  <div className={(`${oldalBar ? 'w-14' : 'w-1/2 md:w-1/3'}`) fixed z-40 duration-300 transition-all ease-in border-r-2 bg-slate-100 h-screen`} >
    <div className='flex justify-between pl-3' >
      <h1 className={(`${text-center items-center justify-center text-3xl ${oldalBar ? 'hidden' : 'flex'}`)} >Szűrő</h1>
      <button className='ring-2 ring-stone-400 rounded-md p-1 justify-center items-center flex text-2xl m-3 ml-0 bg-stone-200' onClick={() => { setOldalBar(!oldalBar)}} >Szűrő</button>
    </div>
    <hr className={(`${oldalBar ? 'hidden' : 'flex'}`) mx-3 bg-black} />
    <div className={(`${oldalBar ? 'hidden' : 'flex'}`)} >
      <div className='flex flex-col w-full justify-center items-center' >
        <h1>Termék neve:</h1>
        <input type='text' id='termeknev' className='ring-1 ring-stone-400 w-11/12 rounded-md text-center' />
        <h1>Termék ára:</h1>
        <div className='flex w-full justify-between' >
          <div id='min' className='w-full mx-5 justify-between flex rounded-md text-center' >
            <input type='text' id='minar' placeholder='Min' className='ring-1 ring-stone-400 w-full mx-2 justify-between rounded-md text-center' />
            <label htmlFor='minar'>Ft</label>
          </div>
          <div id='max' className='w-full mx-5 rounded-md text-center' >
            <input type='text' id='maxar' placeholder='Max' className='ring-1 ring-stone-400 w-full mx-2 rounded-md text-center' />
            <label htmlFor='maxar'>Ft</label>
          </div>
        </div>
      </div>
      <div className='w-full flex-col' >
        <h1 className='text-2xl ml-2' >Listázás</h1>
        <select name='lista' className='bg-gray-50 border flex mx-auto text-xl border-gray-300 text-gray-900 rounded-lg focus:ring-blue-500 focus:border-blue-500' >
          <option className='rounded-lg' value='all' >Összes termék</option>
          <option className='rounded-lg ring-2' value='dis' >Akciós termékek</option>
        </select>
      </div>
    </div>
    <hr className='w-11/12 flex ml-5 mx-5 mt-5' />
    <div>
      <button onClick={kereso} className='text-2xl m-6 p-3 rounded-md bg-stone-300 shadow-sm' >Keresés!</button>
    </div>
  </div>
```

Kosár elemek:

Ezek a kosár oldalon belül található kis kártyák, ahol lehet állítani a mennyiséget és törölni terméket a kosárból. Gépes nézetben a képe a bal oldalt helyezkedik el, mellette fent a termék neve, alatta a garancia és az értékelése (ezek még továbbfejlesztési lehetőségek). Jobb oldalt a mennyiség, mennyiség növelése, csökkentése és a piros kuka gomb amivel lehet törölni a terméket a kosárból. Illetve a termék ára látszik a kártyán. Telefonos nézetben fent van a kép, alatta pedig ugyan ezek a szövegek.


```

return (
  <div className='bg-stone-400 justify-between mb-4 flex flex-col rounded h-72'>
    <div className='flex justify-between'>
      <div className='flex p-3'>
        <img src={`http://localhost:8000/uploads/${termek.termekkepek[0].kep}`} className='size-40' />
        <div className='flex flex-col'>
          <h2 className='px-5 pt-5 justify-between text-4xl text-indigo-800 font-bold'>
            {Math.round(termek?.akcios ? (termek.ar - (termek.ar * (termek.akciosar / 100))) * termek.darab : (termek.ar * termek.darab)} Ft
          </h2>
          <h2 className='px-5'><span className='font-bold'>Garancia:</span> 12hónap</h2>
          <h2 className='px-5'><span className='font-bold'>Értékelés:</span> 84% pozitív</h2>
        </div>
      </div>
      <div className='flex pr-5 pt-5 flex-col gap-3 items-center text-2xl mb-3 justify-center'>
        <button onClick={() => { deleteTermek() }}><IoTrashBin className='size-6 text-red-700' /></button>
        <button onClick={MennyisegTobb}><IoPlus className='size-6 text-blue-700' /></button>
        <span className='text-3xl font-bold'>{mennyiseg}</span>
        <button onClick={MennyisegKevesebb}><IoMinus className='size-6 text-blue-700' /></button>
      </div>
    </div>
    <div className='p-3'>
      <h1 className='text-4xl'>{termek.nev}</h1>
    </div>
    <hr className='m-2' />
  </div>
)

```

Admin belépő:

Ez az admin felületre való belépést teszi lehetővé. Itt kell megadni az admin nevet és jelszót. Egy fekete alapon egy szürke kocka, amibe lehetőség van 1-1 mezőbe megadni a nevet és a jelszót. Ez után a Belépés gombra kattintva amennyiben helyes adatokat adott meg tovább enged az admin panelre.

```

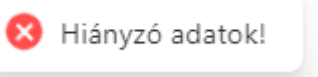
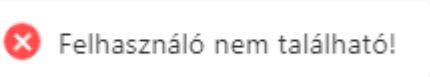
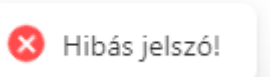
return (
  <div className='w-screen flex justify-center items-center h-screen bg-black'>
    <div className='bg-gray-800 text-white rounded p-4 mb-4 mx-3 w-full sm:w-1/2 lg:w-1/3'>
      <div className='flex items-center justify-between mb-3'>
        <a href='home' className=''>
          <h3 className='text-red-600 text-2xl'>WEBX</h3>
        </a>
        <h3 className='text-xl mt-2'>Admin panel</h3>
      </div>
      <div>
        <input onChange={() => {setNev(event.target.value)}} type='email' className='bg-gray-500 rounded p-3 my-3 w-full border-spacing-2' placeholder='ad' />
        <input onChange={() => {setJelszo(event.target.value)}} type='password' className='bg-gray-500 rounded p-3 my-3 w-full border-spacing-2' placeholder='jelszo' />
      </div>
      <div className='d-flex align-items-center justify-content-between mb-4'>
        <span className='px-2 flex text-sm'>Minden illetéktelen bejelentkezés jogi következményeket von maga után. Amennyiben nem rendelkezik admin jogosítással, a bejelentkezés nem fog működni.</span>
        <button onClick={login} className='bg-red-600 w-full py-3 mb-4 text-xl'>Belépés</button>
      </div>
    </div>
  </div>
)

```

Tesztelési dokumentáció

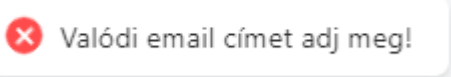
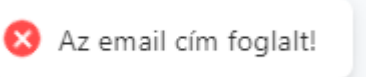
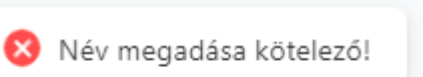
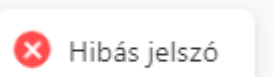
Belépés

Belépés esetén adatok megadása nélkül „Hiányos adatok!” üzenetet kapunk. Nem létező felhasználó esetén a „Felhasználó nem található” üzenetet kapjuk. Hibásan megadott jelszó esetén pedig „Hibás jelszó” üzenet fogad. Sikeres belépés esetén továbblép, és „Sikeres bejelentkezés” üzenetet látunk, illetve átirányít a kezdőlapra.

1.  Hiányzó adatok!
2.  Felhasználó nem található!
3.  Hibás jelszó!

Regisztráció

Üresen hagyott E-mail esetén „Valódi email címet adj meg!”¹ üzenet fogad. Illetve minden hibásan megadott, @ és . karakter nélküli e-mail címet ad meg ez az üzenet fogad. Foglalt E-mail cím esetén az „Az email cím foglalt!”² üzenet fogad. Sikeresen megadott E-mail cím esetén továbblép a név és jelszó megadásra. Itt üres mezők esetén „Név megadása kötelező!”³ üzenet fogad. Név megadása, de hibás vagy üresen hagyott jelszó esetén „Hibás jelszó”⁴ üzenetet kapunk. Jól megadott adatok esetén továbblép, Itt a telefonszám és a Születési dátumot kell megadni. Üres adatok esetén „Nem valós adatok”⁵ üzenetet kapunk. Illetve minden hibásan megadott adat esetén a Nem valós adatokat adja vissza. Sikeres regisztráció esetén átirányít a kezdőlapra.

1.  Valódi email címet adj meg!
2.  Az email cím foglalt!
3.  Név megadása kötelező!
4.  Hibás jelszó

 Nem valós adatok!

5.

Rendelés véglegesítése

Üresen hagyott adatok esetén a „Minden adat kitöltése kötelező!”¹ üzenet fogad. Amennyiben nincs minden adat megadva minden esetben ez az üzenet fogad. Sikeresen megadott adatok után átirányít a Sikeres rendelés oldalra. Innen tölthető le a számla is. Itt a „Sikeres rendelés”² üzenetet látva kerülünk át.

 Minden adat kitöltése kötelező!

1.

 Sikeres rendelés!

2.

Továbbfejlesztési lehetőség

A projekt létrejötté elején sok-sok gondolkodást követően minél nagyobbra akartam összerakni, de nem volt mindre idő, így lehetne mit sorolni. Első nagy előrelépés a fizetés normális módra való megcsinálása. A számla átírása UTF-8 magyar betűkre. Kisebb dizájnbeli átalakítások. Tervben volt egy működő szűrő bevezetése. Erre nem volt elég idő megvalósítani, de el lett kezdve, megcsináltam az alapjait, a kinézetét, de nagyon alap szinten. Tervben van, hogy lehessen a termékeket sorba rendezni ár szerint növekvő és csökkenő sorrendbe. Keresni név szerint a termékre, Ár szerinti szűrőt beállítani, ahol le tudja szűrni egy adott keret közötti termékeket. Minden oldalon átírni teljes egészében a dizájnt, nem szimpla fekete fehéren hagyni mindent. Követni szeretném a mai weboldal „trendeket”. Modernebb megjelenés lenne a cél. Akár egy „about us” fül bekerülhetne valahova, vagy ezt felváltaná egy footer (lábléc) hozzáadása a weboldalhoz. Itt lenne feltüntetve az elérhetőség. Több termék felvitele az adatbázisba, jelenleg 10 termék van benne alapból. Nem sok, de a projekt beláthatóságához szerintem elég. Valamilyen további visszajelzést szeretnék adni a bejelentkezett felhasználónak. Navigációs bárban jelenleg csak megjelenik 2 gomb ami másra irányít, ugye a kijelentkezés és a profil menü, ezt szeretném valamilyen egyértelműbb visszajelzéssel megoldani. Termékek árát szeretném feltüntetni ÁFÁ-val és nélküle. Továbbá több képet rendelni a termékekhez. A számlának kéne számolnia a szállítási díjat, illetve átvételi díjat is. Kezdőlapra egy teljes átalakítás, mint dizájn, mint elrendezés alapján. Profil adatok módosítása fül megcsinálása egy szebb oldalra. Admin panel titkosabbá tétele, üzenőfal fixálása, dátumok normalizálása. Admin dashboardon több kártya, több adat, több diagramm megjelenítése. Több hasznos adat hozzáadása. Diagrammokkal próbálkoztam, de nem sikerültek. Admin panel bővítése egy termék módosítása gombbal, ahol lenne lehetőség az akció mértékét vagy a képet, képeket hozzáadni, módosítani. Paramétereket be szeretném vonni valami más formában. Teljes admin panel reszponzivitását fixálni. Jelen állapotban a Termék hozzáadása, Paraméter hozzáadása telefonos nézet esetén teljesen egybecsúszik.

Irodalomjegyzék, forrásmegjelölés

React alapok:

[Getting Started | Vite \(vitejs.dev\)](#)

Prisma alapok:

[Prisma Documentation | ORM, Accelerate, Pulse & More](#)

JSPDF:

[jspdf - npm \(npmjs.com\)](#)

Tailwind:

[Installation - Tailwind CSS](#)

Argon2:

[argon2 - npm \(npmjs.com\)](#)

Cors:

[cors - npm \(npmjs.com\)](#)

Nodemon:

[nodemon - npm \(npmjs.com\)](#)

Axios:

[Getting Started | Axios Docs \(axios-http.com\)](#)

React-data-table:

[react-data-table-component - npm \(npmjs.com\)](#)

React-hot-toast:

[react-hot-toast - npm \(npmjs.com\)](#)

React-icons:

[react-icons - npm \(npmjs.com\)](#)

React-router:

[Tutorial v6.22.3 | React Router](#)

Uuid:

[uuid - npm \(npmjs.com\)](#)

Multer:

[multer - npm \(npmjs.com\)](#)

Plágium-nyilatkozat

Alulírott (név)

..... (szül.hely)

..... (szül.idő)

..... (anyja neve)

jelen nyilatkozat aláírásával kijelentem, hogy a

..... című záródolgozat

(a továbbiakban: dolgozat) önálló munkám, a dolgozat készítése során betartottam a szerzői jogról szóló 1999. évi LXXVI. tv. szabályait, valamint a Békéscsabai Szakképzési Centrum által előírt, a dolgozat készítésére vonatkozó szabályokat.

Tudomásul veszem, hogy a dolgozat esetén plágiumnak/szerzői jogsértésnek számít:

- szó szerinti idézet közlése idézőjel és hivatkozás megjelölése nélkül;
- tartalmi idézet hivatkozás megjelölése nélkül;
- más szerző publikált gondolatainak saját gondolatként való feltüntetése

Kijelentem továbbá, hogy a dolgozat készítése során az önálló munka kitétel tekintetében a konzulenszt, illetve a feladatot kiadó oktatót nem tévesztettem meg.

Jelen nyilatkozat aláírásával tudomásul veszem, hogy amennyiben bizonyítható, hogy a dolgozatot nem magam készítettem vagy a dolgozattal kapcsolatban szerzői jogsértés ténye merül fel, a Békéscsabai Szakképzési Centrum a dolgozatot elégtelennek minősíti.

Békéscsaba, 20 hó nap

Tanuló aláírása