Jegyzőkönyv

**Web Technológiák féléves feladat**

Téma: PC Webshop

<https://github.com/GaborCsicsely/C7H5VB_WebTech/tree/main>

Csicsely Gábor

C7H5VB

2024.01.09

**Feladat leírás:**

A beadandóra, egy adatbázissal kezelt gépeket árusító webshopot csináltam, ahol meglévő gépek és saját magunknak összerakott gépek közül lehet vásárolni.

Az adatok tárolására főként SQL adatbázist használtam, de néhány funkció megbízható működéséhez ezt kombináltam localstorage-al. Ez éles környezetben nem biztonságos, de a beadandóhoz, tökéletes megoldást nyújt a bejelentkezés megjegyzéséhez, valamint a lekérdezések könnyítéséhez.

Az adatbázis létrehozásához MSSMS (Microsoft SQL Server Management Studio) volt használva.

**Használt programozási nyelvek, IDE:**

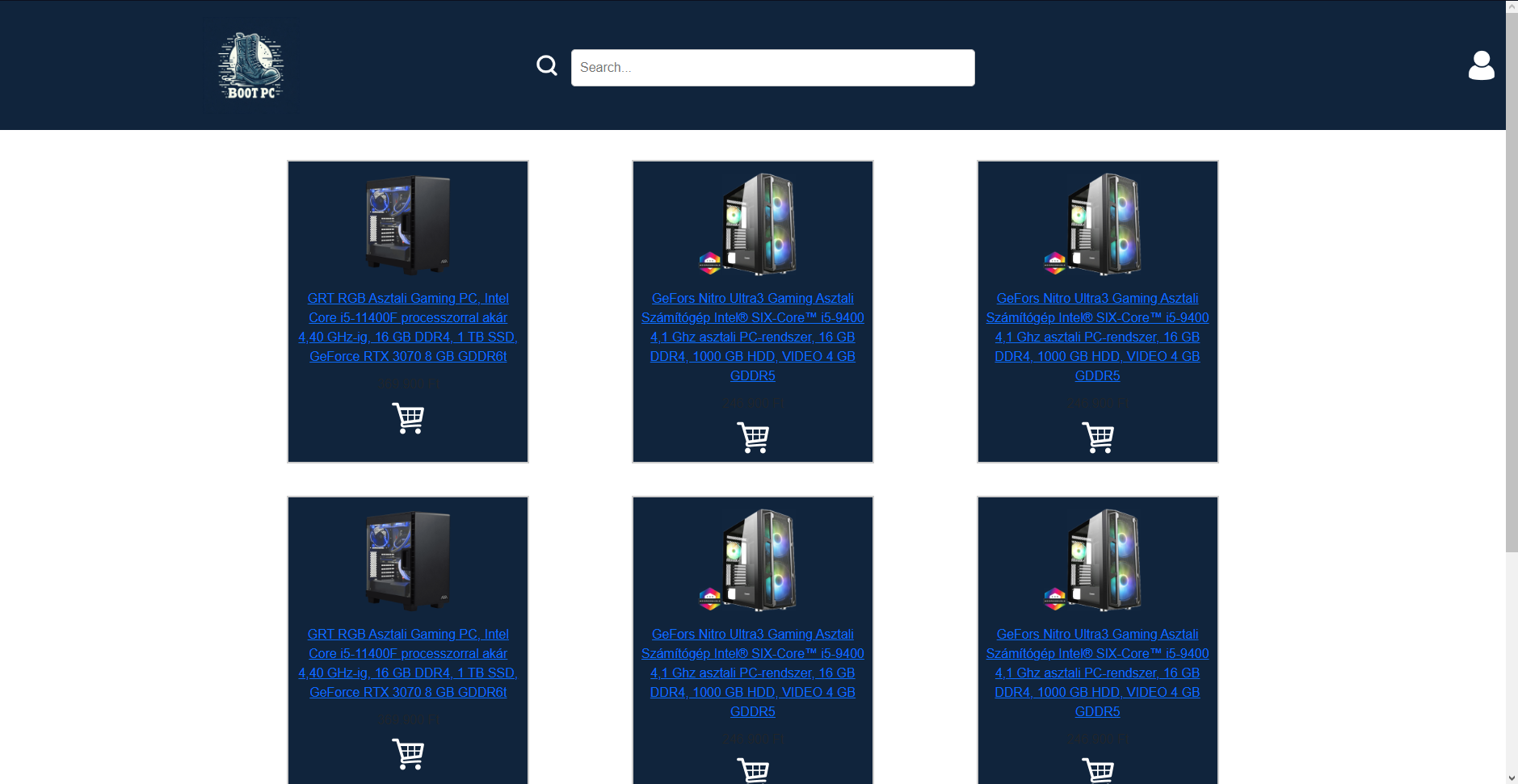
A Projekt maga az ASP.NET-en alapul és Visual Studio IDE-n lett megírva.

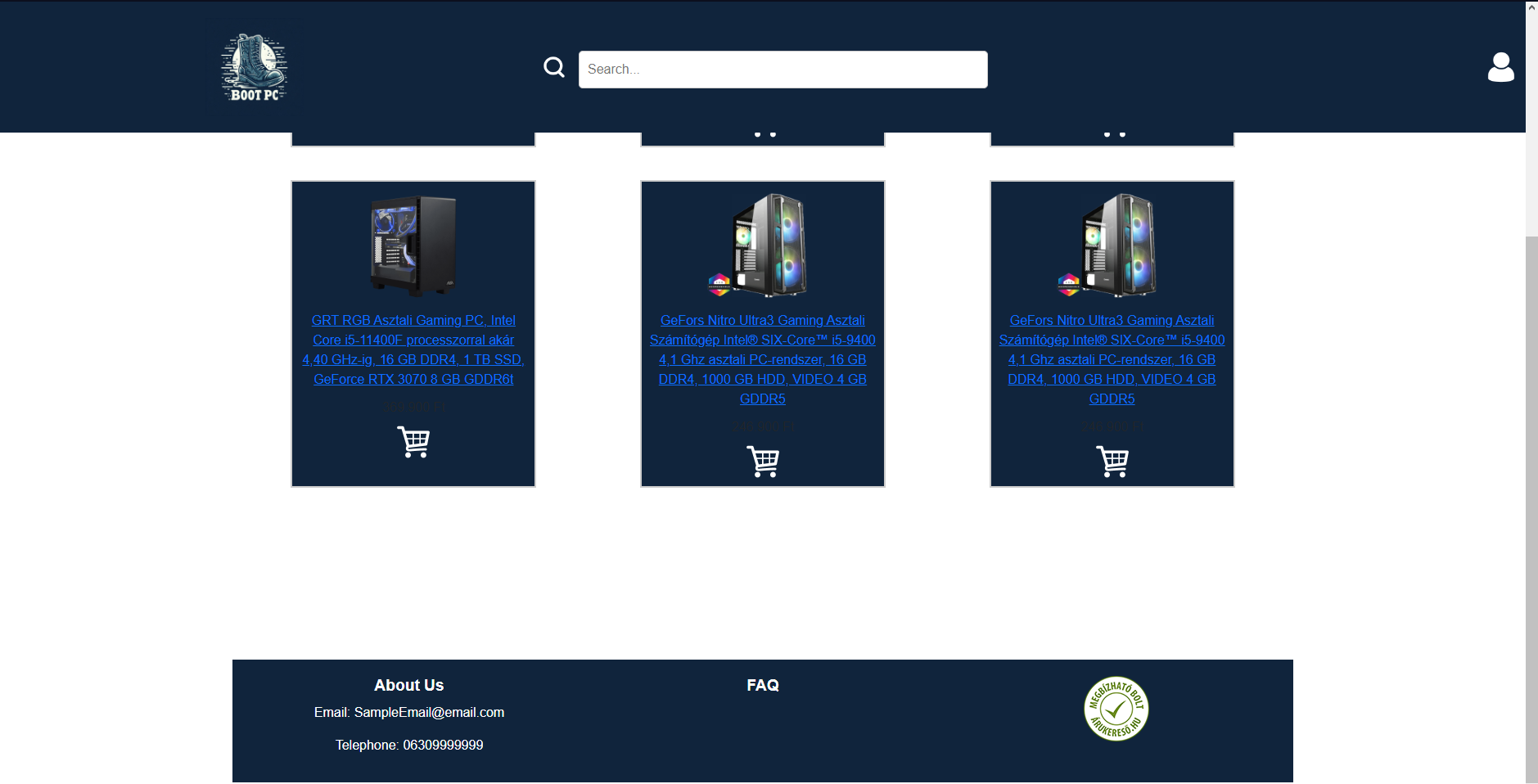
Az ASP.NET-et azért választottam, mert lehetővé tette az adatbázis használatát, és annak kezelését C# segítségével.

A weboldalak HTML-ben voltak megírva, ennek formázására pedig CSS-t használtam és legtöbb funkcióhoz pedig JavaScriptet.

A C# csak a DB kezelésére volt használva.

**A Main Page, Header és Footer:**



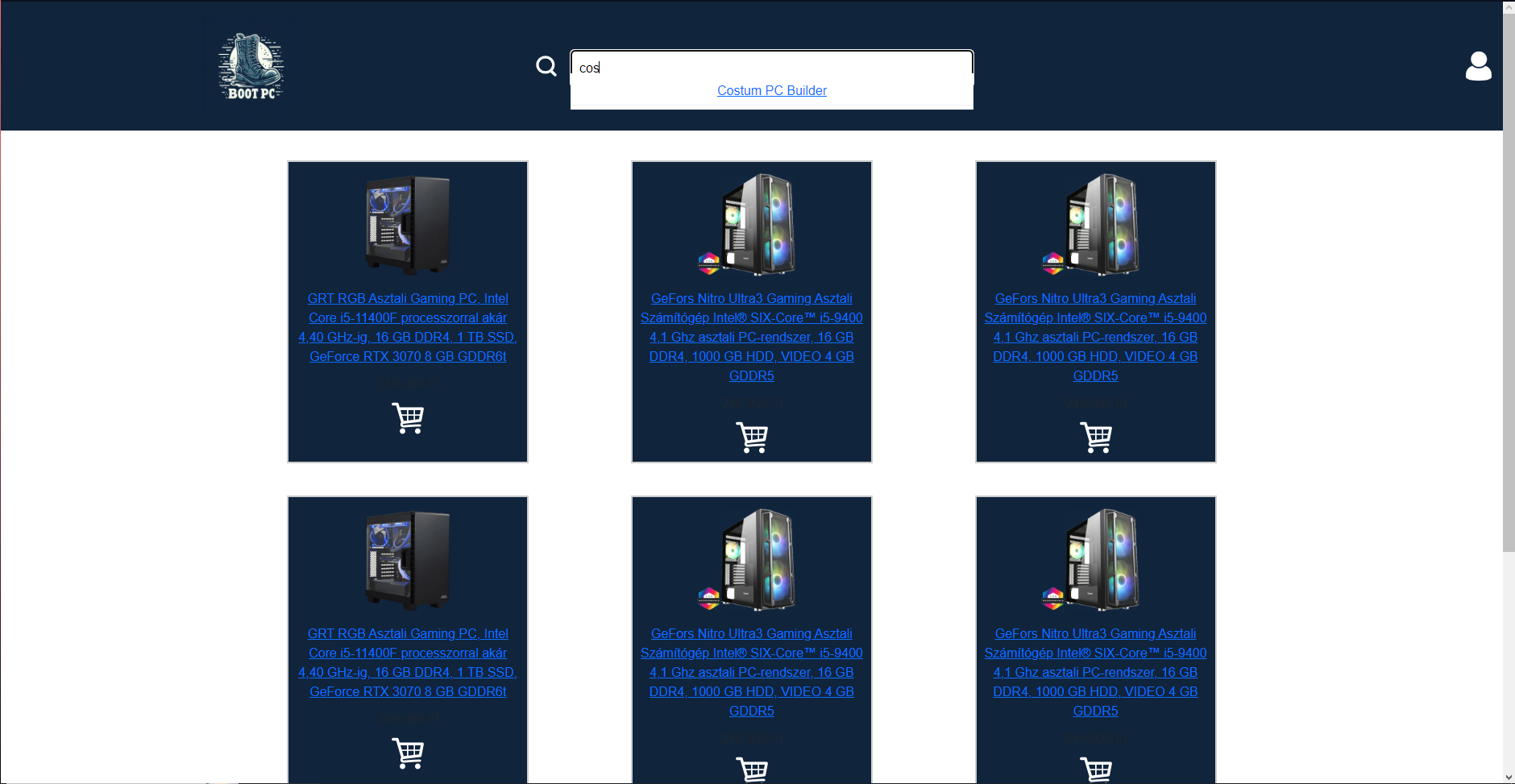
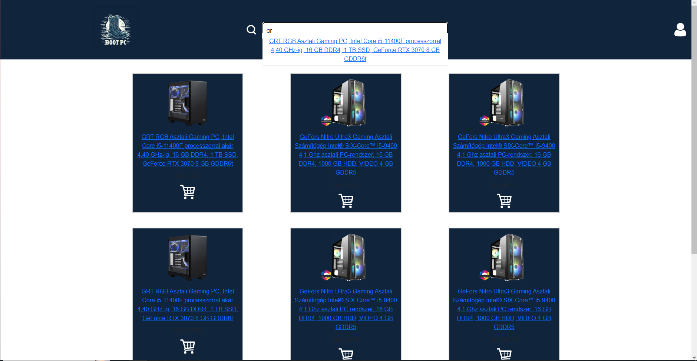
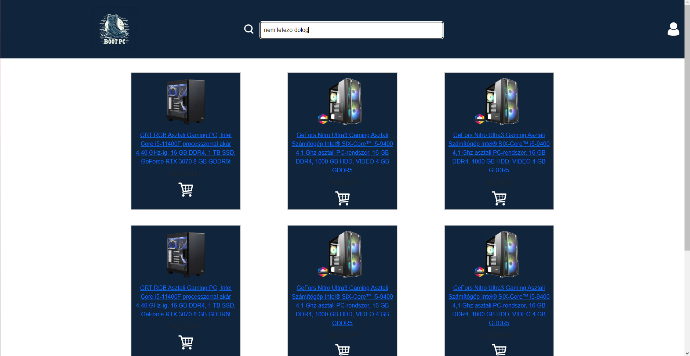


Ami az összes oldalon egyezik, a felül lévő sticky header, valamint a footer a weboldalak alján.

Ezeken található a search bar, a weboldal logó (és egyben a fő oldal linkje) és az Account Page/Login page.

A Fő oldal lényegében 2 gép linkjeit tartalmazza árakkal, képpel és címmel, viszont ezeket a tér kitöltése érdekében többszörösen hozzá adtam.

**A Search Bar:**

****

A Search bar lényegében egy olyan Input mező melyekhez hozzá van kapcsolva egy JS Function.

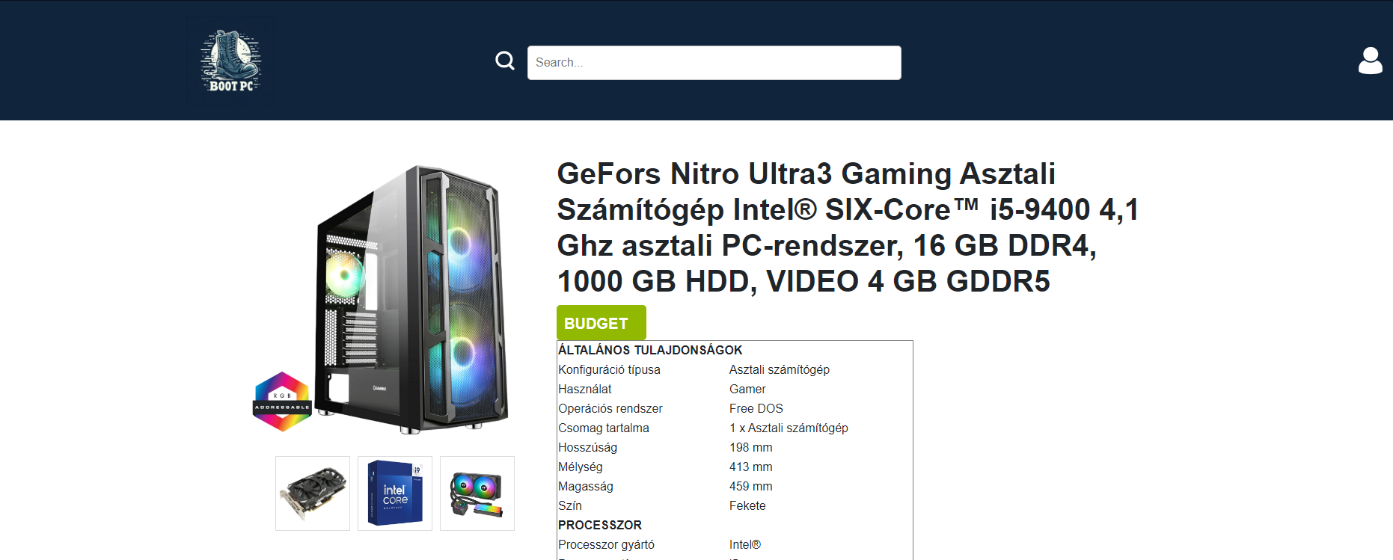
Elsőnek htmlPages tömbbe létrehozzuk azokat az oldalakat amiket elérhetővé szeretnénk tenni keresére(Én a costum PC builder és a Computer1 oldalt adtam hozzá)

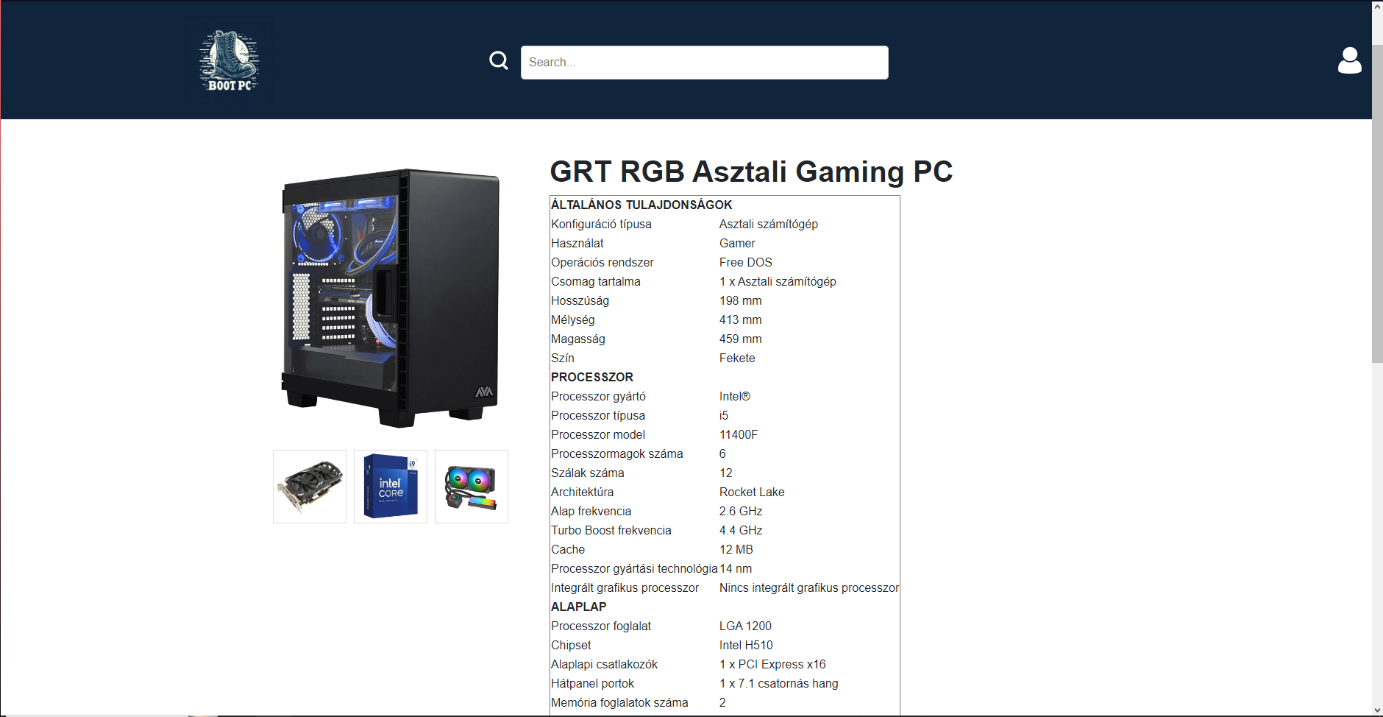
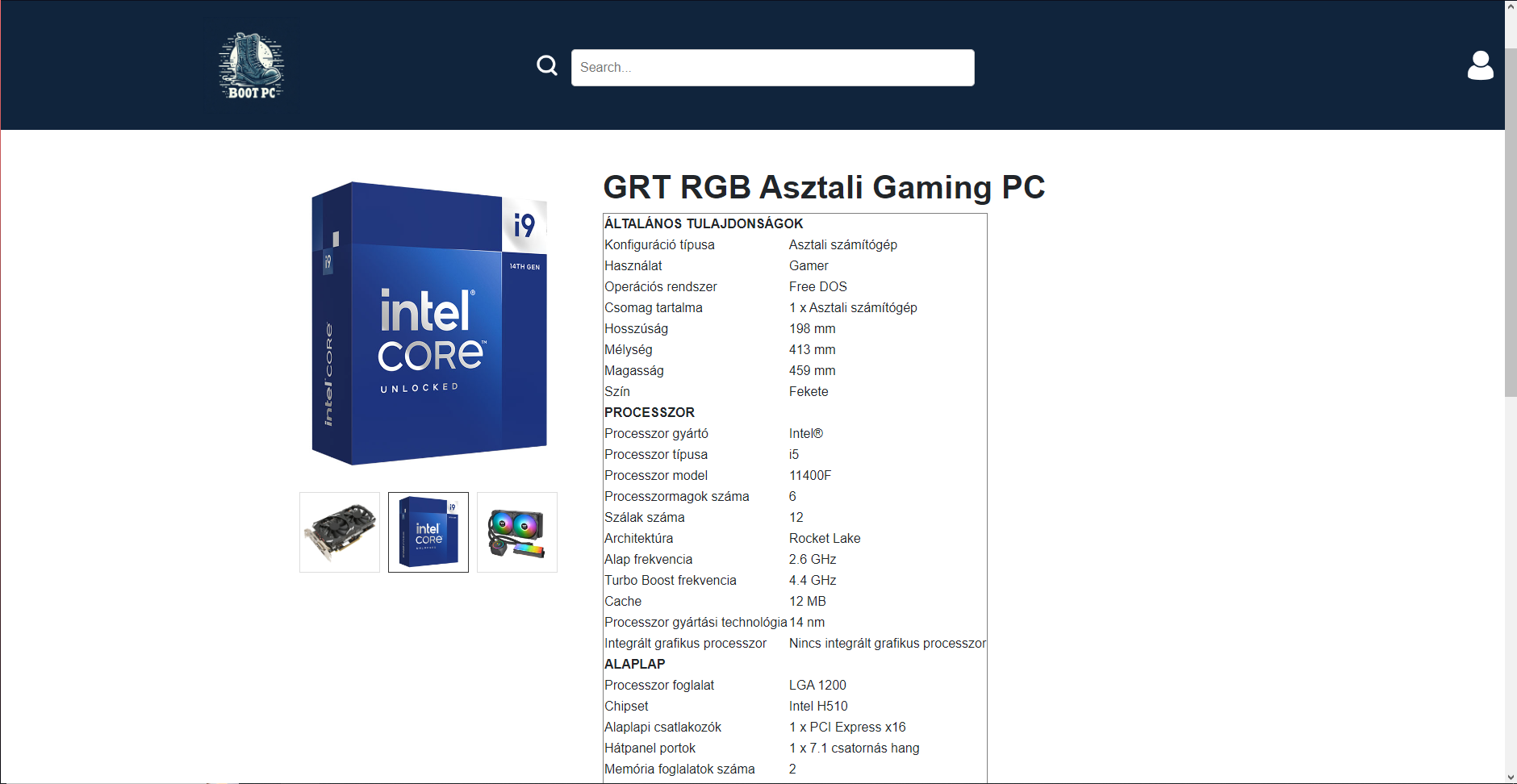
Ezeknek elsőnek megadjuk az elérését, majd a keresési nevét.

Amikor elkezdünk írni bele a searchTerm változó segítségével keresünk a htmlPages-ben a filter függvénnyel.

A talált elemekre létrehoz egy lista elemet, linkkel az elérhetőségével.

**Computer1 és Computer2:**



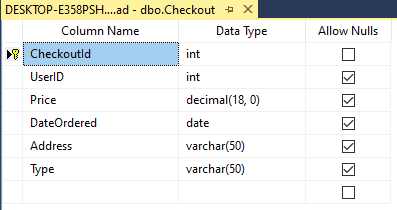


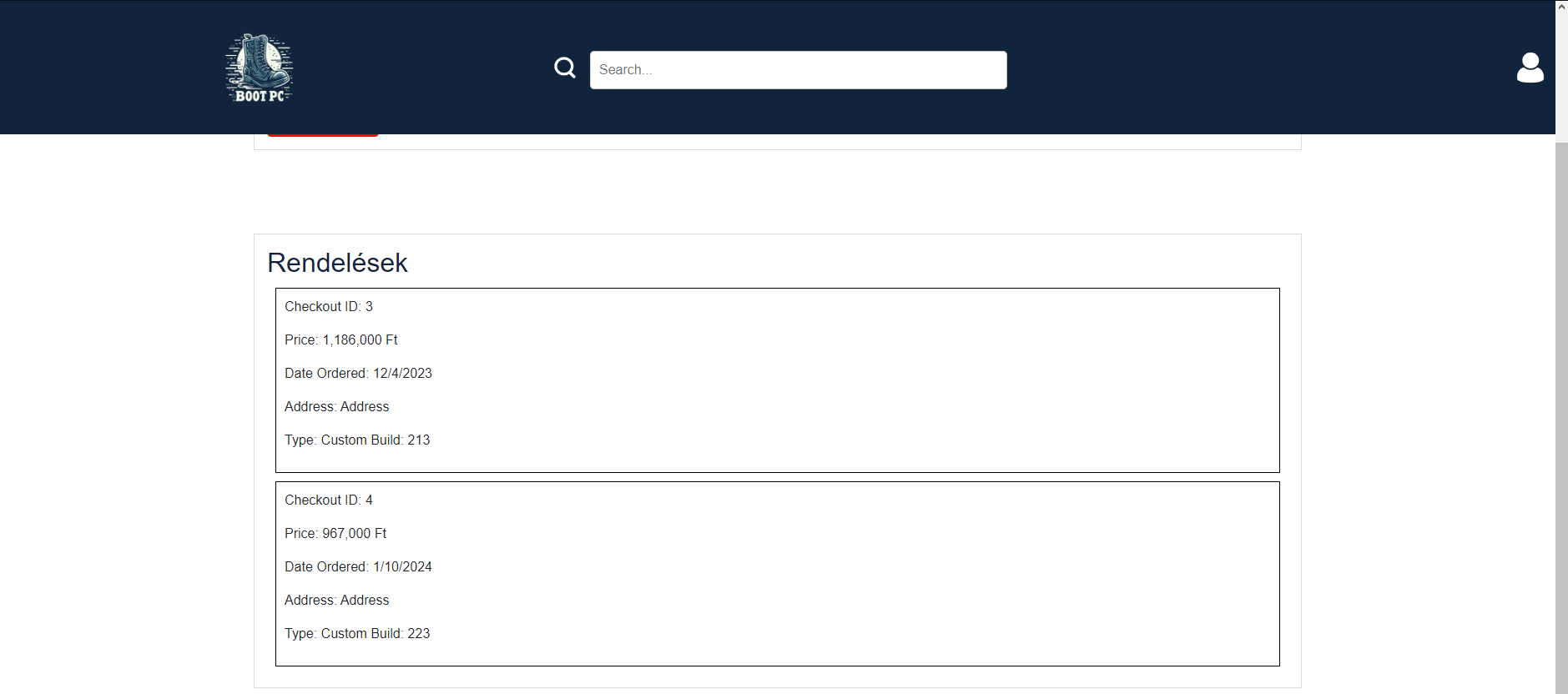
Computer2 oldalon animált BUDGET felirat: <https://github.com/GaborCsicsely/C7H5VB_WebTech/blob/main/animation.mp4>

Ezen az oldalon láthatunk egy táblázatos leírást a megvásárolható gépről, valamint a JS segítségével választhatunk a képek között az egerünkkel.

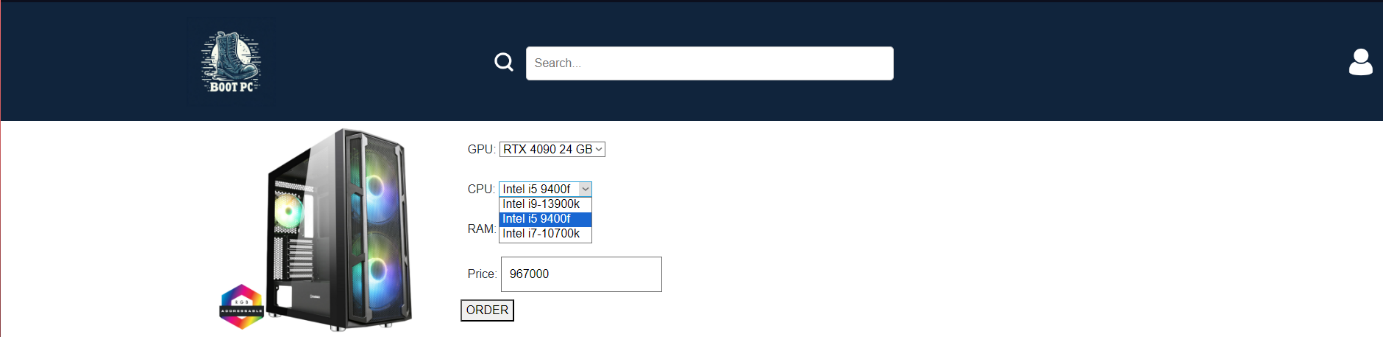
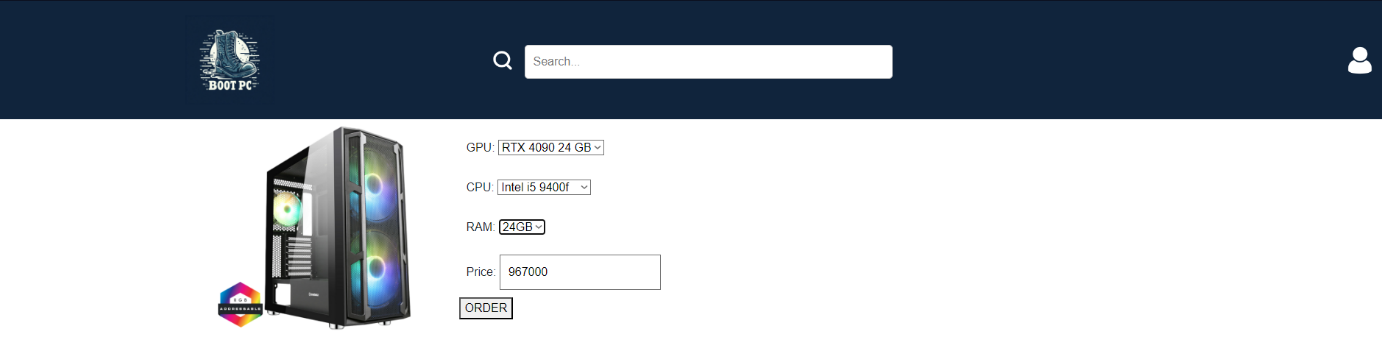
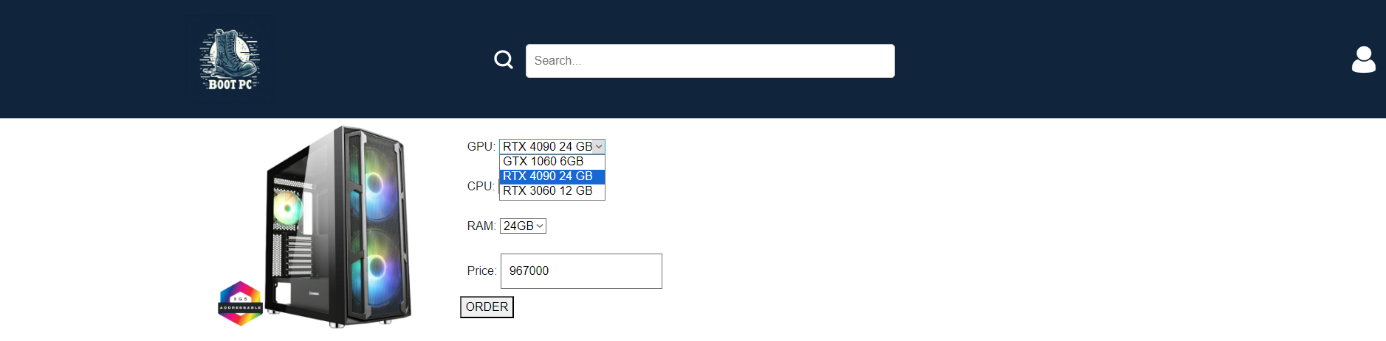
Ha letekerünk az Order gombbal megtudjuk rendelni a terméket.

Ezt az „addToCart()” functionnel oldottam meg, melyben a DB felé küldünk egy kérést a CheckoutController C# kód segítségével.





**Costum PC Builder:**



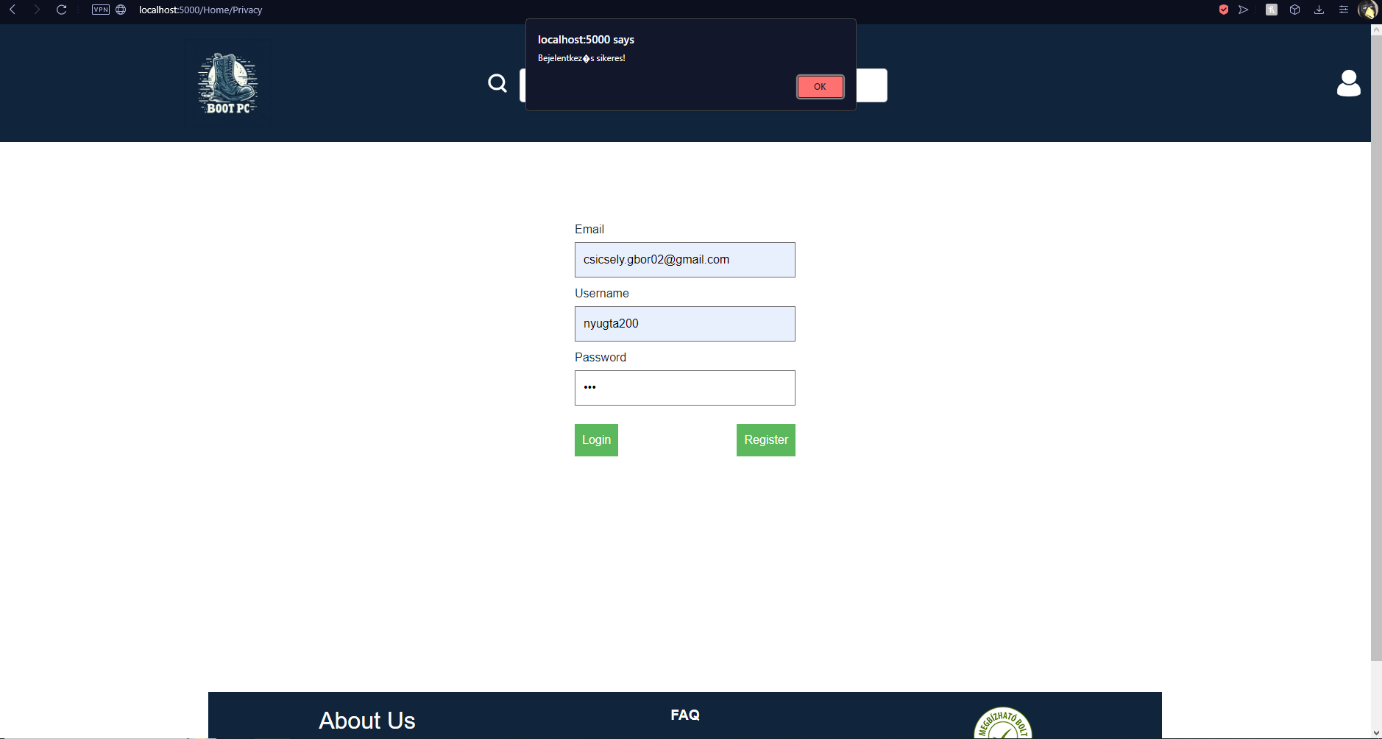
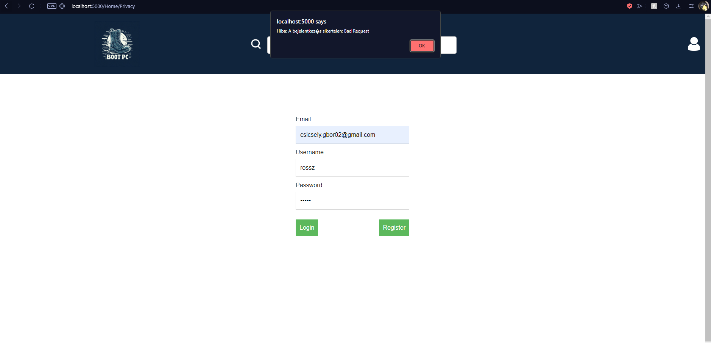
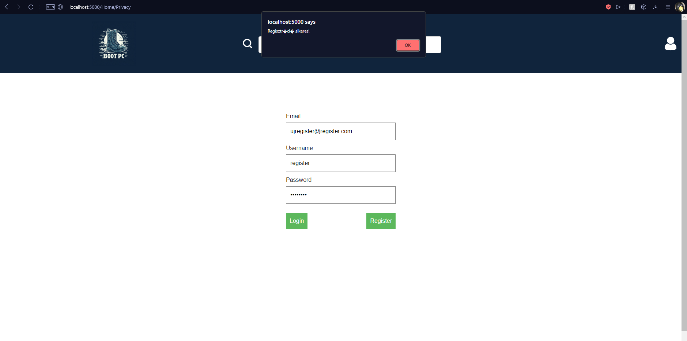
Ennek a felépítéséhez a html forms-át használtam, select-el.

Az árát a kiválasztott elemek alapján egy JS function kiszámolja. Ez úgy lett megoldva h minden opciónak adtam egy értéket, amiket összead és kiszámol.

Amikor megrendeljük a gépet, szerettem volna, hogy az adatbázisban/felhasználói oldalon látható legyen, hogy milyen opciókat választott a felhasználó. Ezt is egy JS function segítségével oldottam meg, melyeben a kiválasztott elemeknek egy bizonyos számot adtam meg 1-3ig.

Így az adatbázisba például, ha az 1. GPU-t 3. CPU-t és 2. RAMot választotta, a kódja Custom Build: 132 lesz.

**Login/Register:**

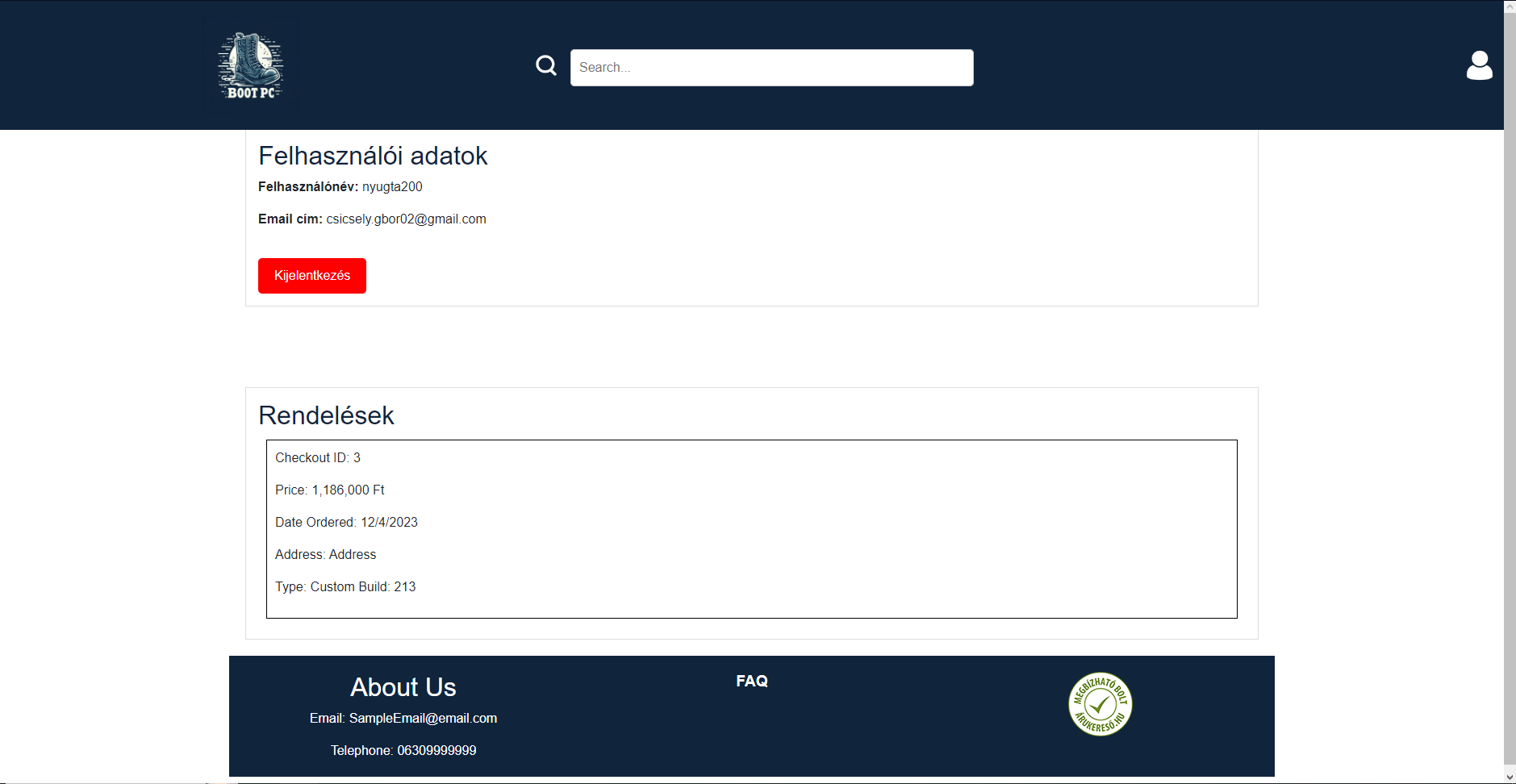


Amikor rákattintunk a Headerben az account ikonra, az oldal ellenőrzi, hogy éppen bevan-e jelentkezve a felhasználó LocalStorage-ban (userLoggedIn = True/False) és ez alapján visz vagy a Login, vagy az Account oldalra.

Az oldal itt is Formsot használ, melyben az összes mező required.

Ez után, ha kitöltjük ellenőrzi az DB-ben lévő adatokat és az alapján vagy hiba üzenetet ad és nem engedi a bejelentkezést/regisztrációt (Ez lehet azért mert már foglalt az email, vagy mert helytelenek az adatok), vagy engedélyezi és továbbít minket az account page-re, és userLoggedIn = True-t, más fontosabb adatokat felvisz a LocalStorage-ba.

**Account Page:**

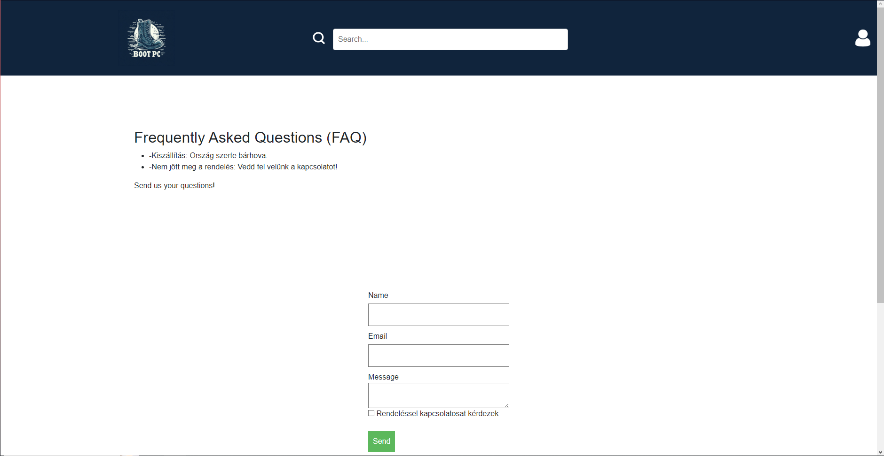
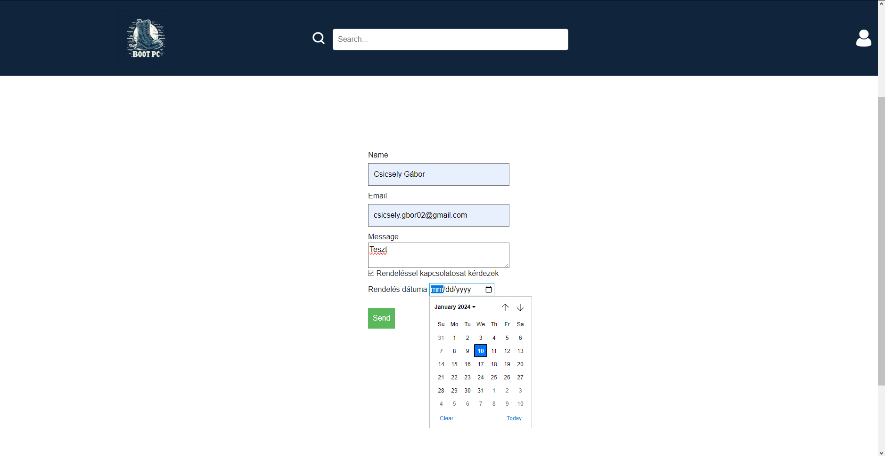
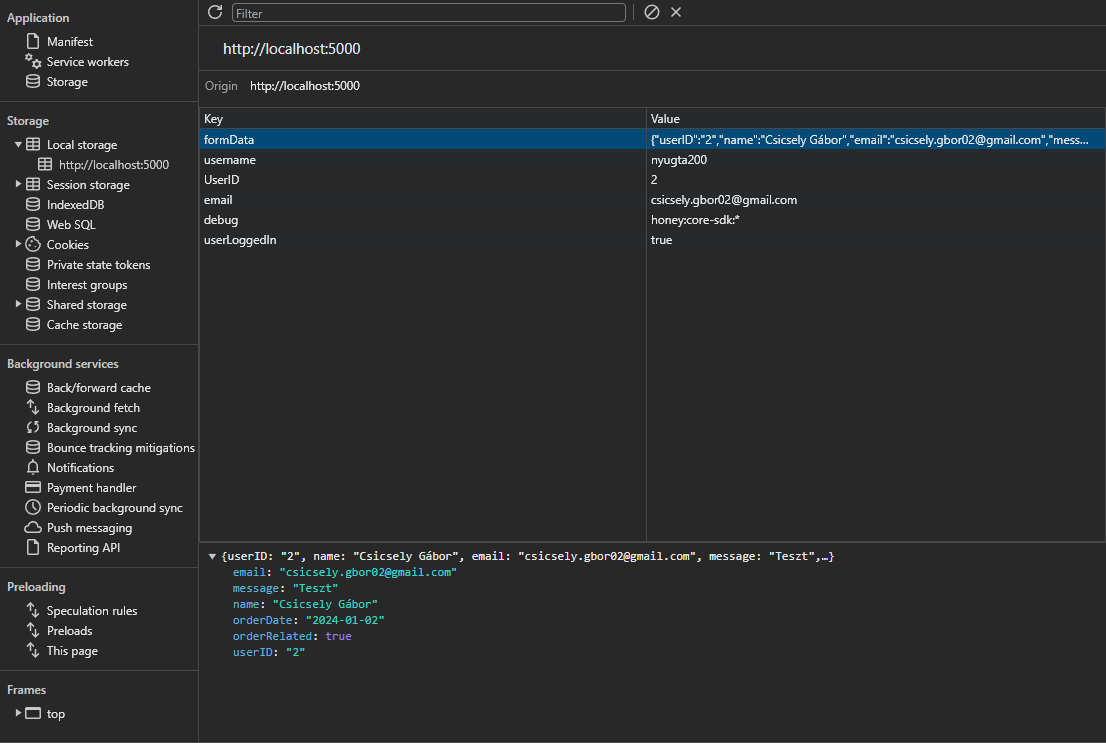


Ezen az oldalon találhatjuk meg a leadott rendeléseket, valamint a felhasználói adatainkat és a kijelentkezés gombot.

A felhasználói adatokat egyszerűen kivesszük LocalStoragebol, ez kevésbé biztonságos, mintha SQL hívással szereztük volna meg őket, viszont gyorsabb és sokkal megbízhatóbban működik.

A kijelentkezés gombunk, annyit csinál, hogy törli a LocalStoragebol a felhasználói adatokat, valamint a userLoggedIn változót aztán visszadob a főoldalra.

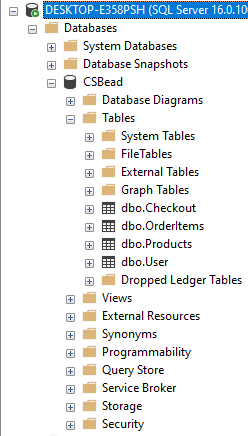
A rendelések viszont DB-böl vannak lekérdezve, a bejelentkezett felhasználó UserID-ja alapján

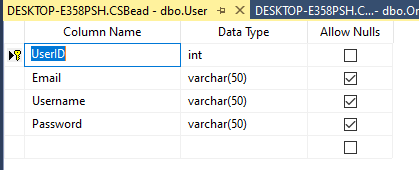
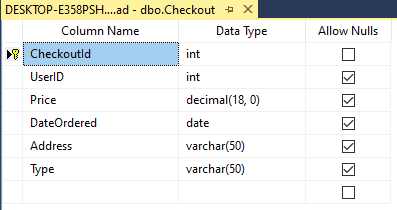
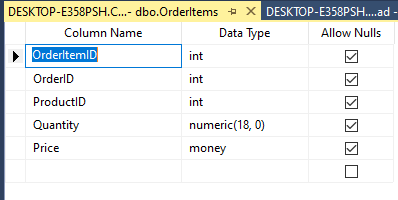
**FAQ oldal:**

Itt is listát használva van egy rövid gyakori kérdések, rá válaszok.

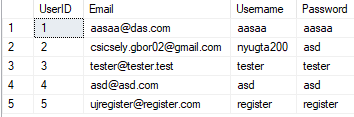
Alatta van egy form, melyben a felhasználó üzenetet küldhet felénk, amit utána LocalStorage-ban tárolunk.

Itt belehet pipálni, hogy a kérdés rendeléssel kapcsolatos-e, és ha igen akkor megjelenik egy dátum választó, melyben a rendelés dátumát kell megadni. Ha a felhasználó bevan jelentkezve UserID-val együtt van mentve az üzenet.

**Az adatbázis:**

****

****

****

**Megjegyzés:**

Sajnos a Visual Studio sokszor véletlenszerűen nem akarta frissíteni a változtatásokat a JS/CSS kódjaimban, ezért 4 különböző .js és 3 különböző .css fájlt tartalmaz a projekt, ezek között kisebb, nagyobb különbségek vannak, emellett néhány weboldal csak a saját maga a html-oldalba írt js kódját használja.

A DB működéséhez C# segítségével DBContextet kellett létrehozni melyben a táblákat megneveztük/létrehoztuk, emellett az összes tábláról kellett készíteni egy modellt amiben a tábla felépítését hoztuk létre.