BALASSAGYARMAT RÉTSÁG SZÉCSÉNY VÁROS MULTINACIONÁLIS ÉS ORSZÁGOS KERESKEDELMI EGYSÉGEINEK BEMUTATÁSA WEBFELÜLETEN

**1. Bevezetés**

**1.1 Témaválasztás indoklása**

A vizsgaremeket team project-ben készítettük el. A témát hiánypótló elgondolásból, az ötlet hasznosságát szem előtt tartva választottuk. Nagyon érdekel minket a webfejlesztés és webprogramozás. Talán az informatikai világában, a hálózatok és infokommunikáció mellett ez a másik nagyobb ága a számítástechnikának, amit úgy gondolunk, hogy a jövőben is tudnánk hasznosítani. Más kontextusból közelítve a dolgot, ezt a témát úgy gondoljuk, hogy akár hasznosítani is lehet egy esetleges online térben történő megjelenésre. A témánk a kereskedelmi áruházakat és az ott forgalmazott termékeket mutatja be webes felületen. Bár tudjuk a nagy multi cégekkel versenyezni nehéz lehet, de ha a „Gondolkodj globálisan cselekedj lokálisan” elvet követjük, és következetesen kis lépéssel haladunk előre akkor talán egy nagyobb volumenű project is kialakulhat a témánkból! Nagy inspirációt adott nekünk Jeff Bezos és Mark Zuckerberg az Amazon és a Facebook atyjai. Életrajzi cikkeit olvasva rá jöttünk, hogy kis dolgokból lehetnek a világmegváltó tettek. A Facebook alapítói is egy egyetemi kollégium helyi hálózatából kiindulva jutottak el a világ meghatározó pontjaiig. Jeff Bezos pedig számunkra jó példát állított azzal, hogy édesanyja házának garázsából globális céget alapított. Számunkra nagy előnyt jelent, hogy az informatikát mint szakmát akár home-office módszerrel is lehet gyakorolni és végezni! Témánk választásánál azt is szem előtt tartottuk, hogy olyat válaszunk, ami szükséges lehet, emellett soha nem évül el, és soha nem megy ki a divatból sem. Következésképp ezért is választottuk az áruházak és néhány termékének bemutatását, mert tudjuk ezekre a dolgokra még sokáig lesz kereslet és kínálat is egyben. Persze egy kis kutató munka után rájöttünk, hogy Nógrád megye e részében nem nagyon létezik ilyen hiánypótló weboldal. Nagyjából ismerve a helyi viszonyokat azért is tartjuk jónak a témát, mert így helyi szinten tudnának tájékozódni az emberek a hozzájuk közel lévő áruházak választékából. Tudva azt, hogy hiába globális és országos szintű cégek ezek, egy adott országrész áruháza mindig az ott előállított alapanyagokból és helyi termelőtől is beszerezheti az előállításnál hasznosított árukat. Ezért is tartottuk fontosnak, hogy helyi szinten helyi megközelítésben készítsük el a hozzá tartozó webes felületet és programot. A vizsgaremek témájából adódóan kézenfekvőnek tűnt, hogy ezt válasszuk. A munkánkat nagy leterheltség, és folyamatos egyéb beadandó dolgozatok és munka mellett végeztük. A témánk kiválasztásánál több előzetes videómegbeszélés zajlott, amelyben szép lassan körvonalazódott mit is szeretnénk végül. A fő ötlet Gergő elgondolása volt, amelyet közösen gondoltunk tovább. Először más városokat és több kis települést is bele akartunk vinni a programba, de sajnos az idő rövidsége és egyéb feladatok miatt a 3 nyugat-nógrádi járásszékhely került kiválasztásra. Azért esett ezekre a választás, mert ezek a városok központi szerepet töltenek be nyugat-nógrádban. Emellett igen sokan fordulnak meg ezekben a városokban, akár turisták akár helyi lakosok. Ebből adódóan, amit a vidéki településeken a helyi lakosok nem tudnak megvenni kénytelenek nagyobb városokba elutazni és az ott forgalmazott árukat megvásárolni. Témaválasztásunk lezárásaként pedig meg kell, említsük, hogy mindketten ezekben a járásokban élünk dolgozunk. Ebből adódóan tudjuk, hogy az adott településen milyen áruházak vannak.

**1.2** **A Dokumentáció tartalmának rövid bemutatása**

A szakdolgozati dokumentáció két fő részből áll. Az első részben található a bevezetés, amelyben a témáról van szó. Az első rész második részében maga a dolgozat tartalmi ismertetője olvasható. A második fő részben a project munka bemutatása került kidolgozásra. Ez a fejezet a fejlesztői környezet (alkalmazott technológiák, fejlesztői segédprogram, segédprogramok), felhasználói dokumentáció (az oldal bemutatása, megjelenése), fejlesztői dokumentáció (a kód bemutatása), egyéb kutatások a project elvégzéséhez, tesztelés, és összegzés, alfejezetekből áll. Erre a két fő részre épül a szakdolgozat.

**2. A weboldal/webalkalmazás bemutatása**

**2.1. Fejlesztői környezet**

**2.1.1. Visual Studio Code**

Maga a program megírásához Visual Studio Code fejlesztői segédprogramot használtunk.

A Visual Studio Code (rövidítve: VSCode vagy VS Code) egy ingyenes, nyílt forráskódú kódszerkesztő, melyet a Microsoft fejleszt Windows, Linux és OS X operációs rendszerekhez. Támogatja a hibakeresőket, valamint beépített Git támogatással rendelkezik, továbbá képes az intelligens kódkezelésre (intelligent code completion) az IntelliSense segítségével. Ezen felül testre szabható, így a felhasználók megváltoztathatják a kinézetet (témát), megváltoztathatják a szerkesztő gyorsbillentyű-kiosztását, az alapértelmezett beállításokat és még sok egyebet.

A Visual Studio Code az Electron nevű keretrendszeren alapszik, amellyel asztali környezetben futtatható Node.js alkalmazások fejleszthetőek. Ugyanakkor a Visual Studio Code nem az Atom forkja, hanem a Visual Studio Online szerkesztőn alapszik (fejlesztési neve: "Monaco").

A programot 2015. április 29-én jelentették be a Build 2015 konferencián (a Microsoft éves fejlesztői konferenciája). Ezt követően pedig 2015. november 18-án MIT licence alatt megnyitotta a forráskódját.

Program nyelvek amelyekkel használható:

Batch, C++, Clojure, CoffeeScript, DockerFile, F#, Go, JADE, Java, HandleBars, Ini, Lua, Makefile, Objective-C, Perl, PowerShell, Python, R, Razor, Ruby, Rust, SQL, Visual Basic, XML

„

forrás: <https://hu.wikipedia.org/wiki/Visual_Studio_Code>

A Visual Studio Code-ban több bővítményt is használtunk, hogy megkönnyítsük a munkánkat.

Ilyenek:

* vscode-icons: a mappastruktúra fájljainak megkülönböztetésére
* Prettier – Code formatter: a kód megformázására
* open-in-browser: a weboldal kiválasztható és rendszerre telepített böngészőiben való megjelenítésére
* Emmet Live: automatikus kódkiegészítő bővítmény (Ez nagyban segítette a munkánkat, igen hasznos bővítmény amellyel fel lehet gyorsítani a gépelést, egyben a fejlesztést is)
* Barcket Pair Colorizer: a kód egyes és különálló részeinek megkülönböztetésére
* code snippets

A fejlesztői segédprogramot használva a gépelés során több billentyűkombinációt is alkalmaztunk:

SHIFT-ALT-lefele nyíl = a kód lemásolása lenti irányban kijelölés után

SHIFT-ALT-A = kijelölt kódrészlet kommentbe helyezése

A Back-end oldal programozásánál szintén Visual Studio Code programot használtunk. Eleinte Node modulok telepítésével oldottuk meg aztán később ezen változtattunk.

**2.1.2. XAMPP**

„

A XAMPP egy szabad és nyílt forrású platformfüggetlen webszerver-szoftvercsomag, amelynek legfőbb alkotóelemei az Apache webszerver, a MariaDB (korábban a MySQL) adatbázis-kezelő, valamint a PHP és a Perl programozási nyelvek értelmezői (végrehajtó rendszerei). Ez a szoftvercsomag egy integrált rendszert alkot, amely webes alkalmazások készítését, tesztelését és futtatását célozza, és ehhez egy csomagban minden szükséges összetevőt tartalmaz. A rendszer egyik nagy előnye az összehangolt elemek könnyű telepíthetősége.

A XAMPP egy betűszó ill. rövidítés, betűi a következő kifejezésekből származnak:

X – eredeti olvasatban az angol cross-platform szót helyettesíti, amely a platformfüggetlenséget jelenti

Apache webszerver

MariaDB adatbázis-kezelő (korábban MySQL)

PHP szerveroldali szkriptnyelv

Perl általános célú szkriptnyelv

A XAMPP egyetlen tömörített (zip, tar, 7z vagy exe formátumú) állományba van csomagolva, telepítéséhez mindössze ezt a fájlt kell letölteni és futtatni. A telepítés elvégzi az alapbeállításokat, azokon csak nagyon keveset vagy éppen semmit nem kell változtatni, ezután a rendszer készen áll a webszerver és a mintaalkalmazások futtatására. A XAMPP-csomagot rendszeresen frissítik, így az mindig az Apache, MariaDB, PHP és Perl legújabb változatát tartalmazza, valamint más kiegészítőket is, mint pl. az OpenSSL és a phpMyAdmin. Verziószáma a csomagban található PHP-értelmező verziójával egyezik meg (jelenleg az 5.5.x, 5.6.x és 7.0.x PHP-verziókhoz egyaránt elérhető különálló csomag).

A XAMPP-csomag több példányban is telepíthető a gazdagépre, a különálló telepítések képesek önmagukban, a többi csomag megzavarása nélkül működni, ráadásul az installált példányok egyszerűen átmásolhatók egy másik gépre.

A szoftver három változatban is elérhető: teljes, általános és kisméretű csomagok léteznek.

Hivatalosan a XAMPP tervezői az eszközt egy fejlesztőrendszernek szánták, amellyel a web-tervezők és programozók internetes kapcsolat nélkül fejleszthetik és tesztelhetik alkalmazásaikat. Ennek érdekében több fontos biztonsági funkció alapértelmezésben ki van kapcsolva a csomagban, ennek ellenére a XAMPP szoftvert valódi webes szolgáltatóként is használják. A csomag egy külön eszközt tartalmaz a legfontosabb részek jelszavas védelmének beállítására.

A XAMPP többféle adatbázis-kezelő használatát is támogatja, ilyenek pl. a MySQL és az SQLite és mások.

A XAMPP telepítése után a helyi gép (a localhost) hálózati gépként is hozzáférhetővé válik, pl. FTP kliensprogrammal elérhető. Alkalmazható pl. FileZilla fájlkezelő, telepíthetők tartalomkezelő rendszerek, mint a WordPress vagy a Joomla!. A localhost közvetlenül is kezelhető egyes HTML vagy általános szövegszerkesztőkből, az FTP protokoll használatával. Az alapbeállítás szerinti FTP felhasználónév a „newuser”, jelszava „wampp”. Az alapbeállítás szerinti MySQL felhasználónév a „root”, jelszó nélkül.

„

forrás: <https://hu.wikipedia.org/wiki/XAMPP>

Igen hasznos volt az adatbázisunk szerver oldalról történő lehívásához. Mi Apache és mysql-t használtunk. Sajnos eleinte egyéni munka során volt akit ledobott egy adott portól de újratelepítés megoldotta a problémát. Egy másik probléma volt viszont a fejlesztés során amikor phpmyadmin felületen próbáltunk exportálni adatbázist vagy táblákat. Ekkor hálózati hibát írt ki! Hosszas utána járás után kiderült, hogy a xampp újabb verziója bugot (hibát) rejt, és ilyen esetben át kell állni egy régebbi verzióra és az megoldja a problémát az exportálás során. Így letelepítettünk egy régebbi verziót és ez után ez a hiba megszűnt, könnyedén ki tudtuk exportálni az adatbázisunkat. Portok tekintetében otthoni feladat végzés közben 8080-as localhost phpmyadmin, és 3306-os mysql portot használtunk!

**2.1.3 DBforge**

A phpmyadmin felületén létrehozott adatbázist a DBforge nevű adatbázis kezelő programban teszteltük le, több lekérdezést is írtunk az adatbázishoz, hogy lássuk minden úgy működik ahogy mi azt elképzeltük, az egyik ilyen lekérdezést pedig a későbbiekben felhasználtuk a backend részében. A DBforge egy adatbázis kezelő program amellyel adatbázisokat lehet létrehozni, módosítani, és táblákat lehet összekapcsolni. Előnye a grafikus megjelenítés és megjelenítés, ami szintén nagyban megkönnyíti a fejlesztést és az adatbázisok tervezését és létrehozását. Bár korlátozottak a képességei a programnak de adatbázisok létrehozására alkalmas.

**2.1.4 MySql**

A MySQL egy széles körben használt relációs adatbázis-kezelő rendszer.

A MySQL ingyenes és nyílt forráskódú.

A MySQL ideális kis és nagy alkalmazásokhoz egyaránt.

**Mi az a MySQL?**

A MySQL egy relációs adatbázis-kezelő rendszer.

A MySQL nyílt forráskódú.

A MySQL ingyenes.

A MySQL ideális kis és nagy alkalmazásokhoz egyaránt.

A MySQL nagyon gyors, megbízható, méretezhető és könnyen használható.

A MySQL többplatformos.

A MySQL kompatibilis az ANSI SQL szabvánnyal.

A MySQL először 1995-ben jelent meg.

A MySQL-t az Oracle Corporation fejleszti, terjeszti és támogatja.

A MySQL nevét a társalapító Monty Widenius lányáról kapta: My

**Ki használja a MySQL-t?**

Hatalmas webhelyek, mint a Facebook, Twitter, Airbnb, Booking.com, Uber, GitHub, YouTube stb.

Tartalomkezelő rendszerek, mint a WordPress, Drupal, Joomla!, Contao stb.

Nagyon sok webfejlesztő szerte a világon.

Adatok megjelenítése a webhelyén

**2.1.5. HTML**

A HTML a szabványos jelölőnyelv a weboldalak létrehozásához.

**Mi az a HTML?**

A HTML a Hyper Text Markup Language = Hiperszöveges jelölő nyelv rövidítése.

A HTML a szabványos jelölőnyelv a weboldalak létrehozásához.

A HTML egy weboldal szerkezetét írja le.

A HTML egy sor elemből áll.

A HTML elemek megmondják a böngészőnek, hogyan jelenítse meg a tartalmat.

A HTML-el weboldalunk vázát tudjuk kialakítani ez írja le a weboldalnak mire épüljön az alapstruktúrája a kódnak.

Szemléltetésként elmondható:

Gondoljunk, az emberi testre a HTML jelenti az ember csontozatát a CSS bemutatja az ember megjelenítését (mint pl bőr szín, emberi hússzövetek, szem stb.), a Javascript pedig az ember mozgásáért-beszédéért hallásáért felel, tehát az interakciókért. Ez a 3 alapnyelv a fő mozgató rugója egy weboldalnak. Ezek aztán egymásra épülnek legalsó szinten a HTML-el. Mint a CSS a HTML is leírónyelv, amely bemutat bizonyos elemeket a weboldalon. Mind a HTML mind pedig a CSS-nek vannak korlátai, amit viszont Javascriptel tovább tudjuk gondolni. E 3 nyelvhatással van egymásra. Ezek közötti viszonyok mutatják be a weboldalunk megjelenését!

Mivel a HTML az alapstruktúrája a weboldalunknak ezért ezt egy állományba kell megírnunk pl: **terkep.html** amit html kiterjesztéssel kell elmentenünk. Egy weboldal közzétevéséhez vagy szerverre feltöltéséhez index.html az elfogadott szabvány, amely ha vannak aloldalak vagyis pl aloldal.html lesz az első amit betölt a szerver a kívánt címről.

**2.1.6. CSS**

A CSS az a nyelv, amelyet a weboldal stílusának meghatározására használunk.

**Mi az a CSS?**

A CSS a Cascading Style Sheets rövidítése.

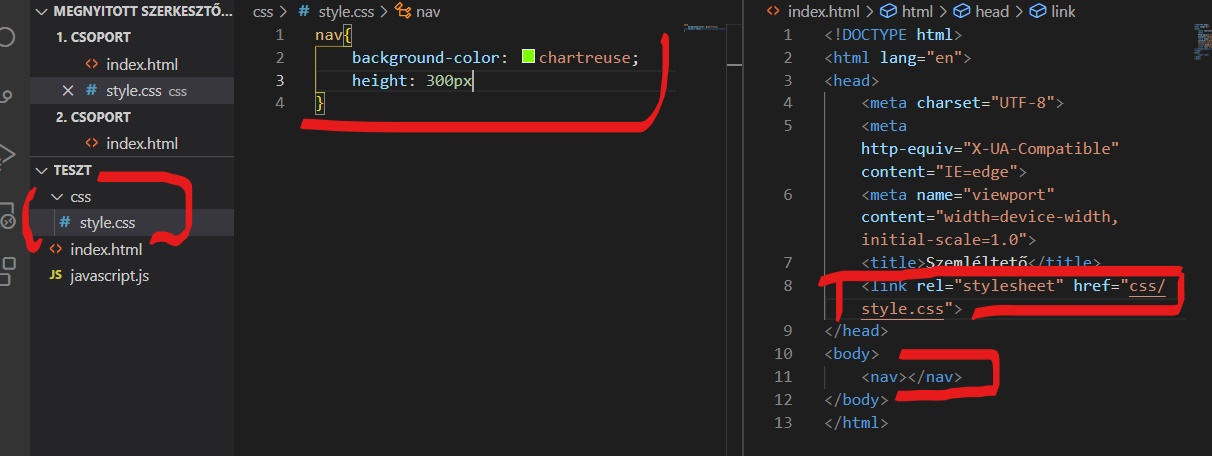
A CSS leírja, hogyan kell a HTML-elemeket megjeleníteni a képernyőn, papíron vagy más adathordozón.

A CSS rengeteg munkát takarít meg. Egyszerre több weboldal elrendezését is vezérelheti

A külső stíluslapok CSS-fájlokban tárolódnak.

CSS kódot többféle módon írhatunk meg és mindig hivatkoznunk kell egy adott HTML tag-re vagy osztályra, azonosítóra, amelyre aztán megírjuk a tulajdonságokat. Nem hatékony, de alkalmazható megoldás amikor, a CSS-t a HTML állományunk HEAD részébe írjuk be <STYLE>…..</STYLE>. Ezzel a megoldással aztán az a gond, hogy mivel felülről lefelé fordul le a program, ezért ha változtatást akarunk a kódunkon ezért utólag mindig az alsó rész fut le és az lesz, ami alapján megjelenik a kódunk a böngészőben. A másik megoldás, amikor a HTML tag-ek közé írjuk bele a CSS tulajdonságokat, ami azért nem előnyös mert ha mi meg akarjuk változtatni több dolognak is például a betűszínét akkor mindenhova külön oda kell írni, ami egy nagyobb projektnél akár több száz sor is lehet. A harmadik és egyben leghatékonyabb és legjobb módszer, amikor külön állományba szervezzük a tulajdonságokat. Ezt is mind a HTML állományokat külön css kiterjesztéssel menthetjük el. pl: **termekek.css**

A CSS állományunkat a HTML head részében hivatkozzuk, ami alapján már látja a HTML hogy hol található a mappa struktúrában a CSS állományunk.



A képen látszik a létrehozott css állományunk, annak a html tag-ből való tulajdonságainak megadása, és a head részben hivatkozott css elérési útvonal LINK tag-ek között.

Egy weboldalhoz több CSS állomány is tartalmazhat. Célszerű az egyes HTML oldalakhoz külön CSS fájlt vagy fájlokat rendelni. Kisebb animációk és események, kölcsönhatások elvégzéséhez kiválóan alkalmas a CSS. Ennek ellenére főként az oldal megjelenéséért felel.

**2.1.7 Javascript**

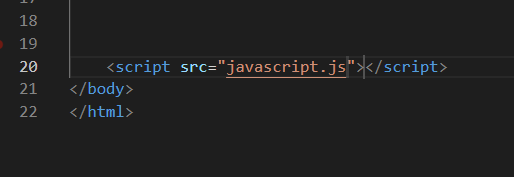
A JavaScript a világ legnépszerűbb programozási nyelve.

A JavaScript a web programozási nyelve.

A javascript felel a weboldalunk kölcsönhatásainak, eseményeiről, mozgásáról, változásáról. A javascriptel animációkat (jobbról balra eltolás, színmegváltoztatás stb.), összegzéseket, összeadásokat, számolási műveleteket is lehet végezni! A javascript programozási nyelv egyaránt használható front és back-end részen is. Javascript kódot többféle képpen tehetünk a weboldalunkhoz. Az első amikor a HTML állományunk HEAD címke részébe <SCRIPT>…\*valami javascript kód\*….</SCRIPT> címkék közé írjuk meg a kódot. A második formája a megadásnak szintén hasonlatos az elsőhöz, csak ilyenkor a HTML állományunk végére vagy a BODY címke közé illesztjük be kódunkat. Mindkettő módszernél SCRIPT tag-ek közé írjuk a kódot. Ezeknél a megoldásoknál viszont figyelni kell a fentről lefele való program lefordulást, mikor éri el a fordító a javascript kódot. Előfordulhat ugyan is az, mint a CSS –nél is hogy előbb fut le a javascript kódunk és nem annál a HTML vagy CSS beállításnál éri el a program. Ennek a következménye viszont az lehet , hogy másképpen jelenik meg a weboldalunk vagy éppen nem azt csinálja amit szeretnénk. A harmadik módszer a javascript létrehozására tehát amikor külön állományba szervezzük az egészet. Javascript állományt a VISUAL STUDIO CODE programmal is tudunk készíteni szintén JS kiterjesztés megadásával.



Javascript állomány létrehozása!



Javascript kód megadása a body alján külön fájlba szervezve.

**2.1.8. NodeJs**

A NodeJs egy nyílt forrás kódú szerverkörnyezet. Ingyenes. Gyakran használják a weboldalak háttéroldali programozására. Több Operációs rendszeren is fut, mint például:

* Windows
* Linux
* Unix
* MAC OS X

Előnye az aszinkron, dinamikus műveletek végrehajthatósága. A NodeJs alkalmas a szerverről állományok, illetve adatbázisok (MySql) betöltésére, törlésére, megváltoztatására űrlapadatok elmentésére, a szerverre. Ezt a működést a weboldal a háttérben végzi ebből a felhasználó semmit sem lát. Ezt nevezik Back-end programozásnak, amit bár fejlesztők használnak legfőképpen, de ki lehet alakítani az ügyfeleknek saját belső használatra különböző felületeken, amit viszont csak az oldal illetve alkalmazás üzemeltetője lát. Tegyük fel, hogy van egy webshop amit megrendeltek, ebben lehet vásárolni, bankártyával fizetni, módosítani a forgalmazott termékek árát, megjelenését! Ennek az alkalmazásnak két fő része van. Az első, amikor a vásárló látja a termékeket, az oldal megjelenését és meg is tudja vásárolni. A másik, ami ez mögött van a háttér ahova az árú fizetésének összege, illetve az adatok a felhasználóról megérkeznek. Itt tudja a webshop üzemeltetője fogadni, illetve az adminisztrációs felületen módosítani a termékek árát, képeit, nevét. Ezt a részt viszont csak az oldal tulajdonosa és a fejlesztők látják. A NodeJs –t a Visual Studio Code programmal telepíthetjük fel.

„

A Node.js a Google-féle V8 JavaScript-motorból, a libUV-ből és számos beépített könyvtárból áll.

A Node.js-t Ryan Dahl hozta létre 2009 januárjában, a növekedését pedig a Joyent, Dahl munkaadója támogatja.

Dahl eredeti célja az volt, hogy lehessen weboldalakat push technológiával létrehozni, ahogy például a Gmail alkalmazásban is látható. Számos egyéb, más programnyelvekben való próbálkozás után a JavaScriptet választotta a meglévő I/O API hiánya miatt. Ez megengedte neki az eseményalapú, nem-blokkoló I/O API definiálását.

Hasonlókat már írtak más programnyelvekre is, így például a Twistedet Pythonra, a Perl Object Environmentet Perlhez, a libeventet C nyelvre és az EventMachine-t Rubyhoz. A legtöbb JavaScript programmal ellentétben nem a webböngésző futtatja, hanem a szerveroldalon alkalmazásként fut. A Node.js implementálja a CommonJS specifikációk egy részét. Tartalmaz továbbá egy REPL rendszert az interaktív teszteléshez.

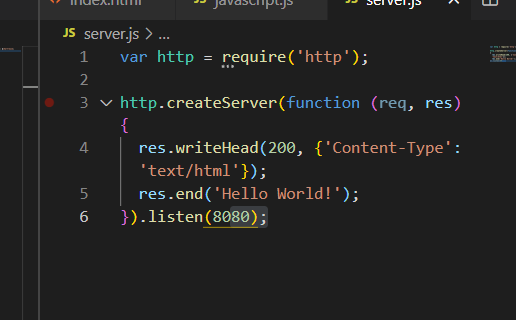
„

forrás: https://hu.wikipedia.org/wiki/Node.js

NodeJs –t szintén JS kiterjesztéssel hozzuk létre. Az alábbi képen egy szerver kód látható.

Most számítógépünk a 8080-as porton szerverként működik.

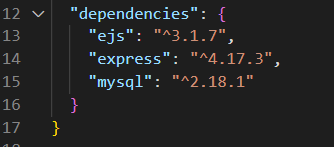
Szemléltető Forráskód: <https://www.w3schools.com/nodejs/nodejs_get_started.asp>



**2.2. Fejlesztői Dokumentáció**

**2.2.1 Felhasznált technológiák**

A webalkalmazás alapstruktúráját html technikával írtuk meg amit **EJS** modullal alakítottunk ki JSON-ben később pedig PHP fájlokat kellett alkalmaznunk az adatbázis végett. Emellett még mysql, express modul is fel van telepítve.

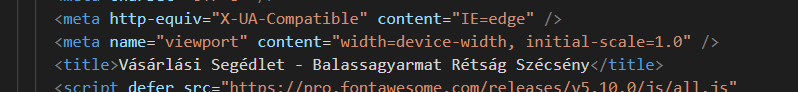


Az oldal megjelenítését Bootsrap 4.6 verzió segítségével alakítottuk ki. Az oldal és az aloldalak responsive megjelenésűek, ez azt jelenti hogy normál, közepes, illetve kis (mobil) méretben is tökéletesen látni minden olyan információt ami elengedhetetlen a felhasználó számára. Az adatbázis létrehozást Mysql típussal hoztuk létre

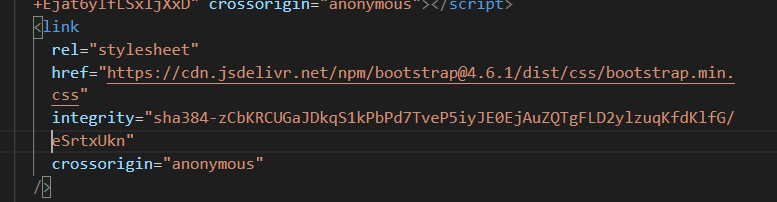
Az adatok lekérését phpMyadmin felületről kérjük mysql szerver 3306-os portján. Az oldalunkat úgy tudjuk elindítani ha a böngésző keresőjébe beírjuk, hogy: „[localhost:8080/project/php/fooldal.php](http://localhost:8080/project/php/fooldal.php)” , ez csak akkor működik ha a weboldalunk mappáját a megfelelő helyre mentettük el, ha rossz helyre mentettük akkor a szerver nem fogja találni a mappát és hibát fog mutatni.

**2.2.2 html**

A html head részében van egy title tag elhelyezve amely minden oldalon megtalálható:



Bootstrap hivatkozásokat adtunk hozzá, amit a bootstrap saját szerveriről emeltünk be:



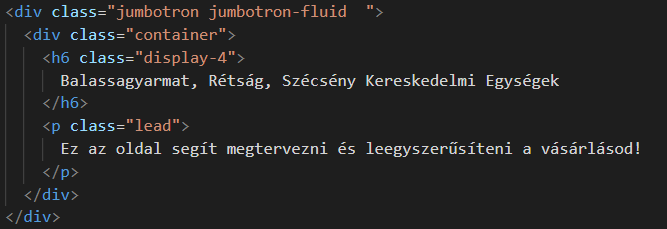
A Fő tartalmak értelemszerűen a body –ban vannak elhelyezve:

**fooldal.php fájl tartalma:**

Legfelső részének az oldalakon egy JUMBOTRON van elhelyezve, amely minden aloldalon és a fő oldalon ugyan úgy néz ki, törekedve az egységes letisztult, egyszerűbb, felhasználóbarát megjelenésre. A háttérnek egy sötétebb sárgából átlóval a világosabb sárga felé haladó átmenetet készítettünk.



Az jumbotron egy jumbotron osztállyal divben elhelyezett container osztályú responsive div-el van elkészítve:



A jumbotron alatt egy keskeny rész található amely egy animáció az oldal készítőiről:

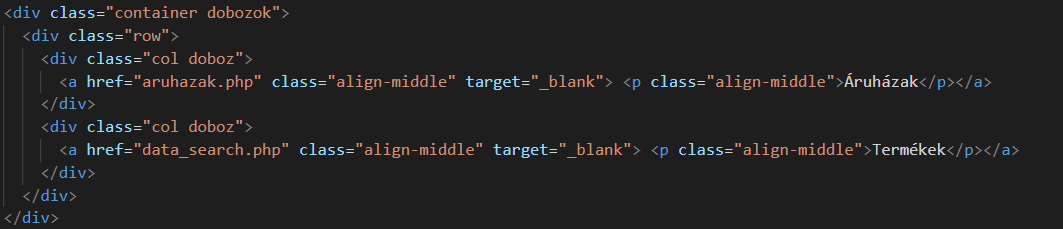
*Ezt a végleges oldalból végül kivettük tanáraink tanácsára, de meg akartuk említeni a dokumentációban.*



Ez szintén divben van elhelyezve! A belsejében egy Nav címke van amely tartalmazza az anchor címkébe helyezett szöveget. Itt ugyan nem állítottuk be de ebbe az anchor címkébe tovább fejlesztésként lehet egy hivatkozást készíteni amely rámutat az oldal készítőinek weboldalára. A mi dolgozatunkban ezt nem készítettük el!

A fő oldal fő tartalma egy konténer mely két kártyát tartalmaz anchor címkékkel, amelyek az egyes al-oldalakra mutatnak. A háttérkép egy fixált áruház kép, a fixált az azt jelenti, hogy az oldal le és fel tekerése közben nem mozog az oldallal hanem fixen a háttérben marad.





A main konténer az anchor címkékkel.

**aruhazak.php tartalma:**

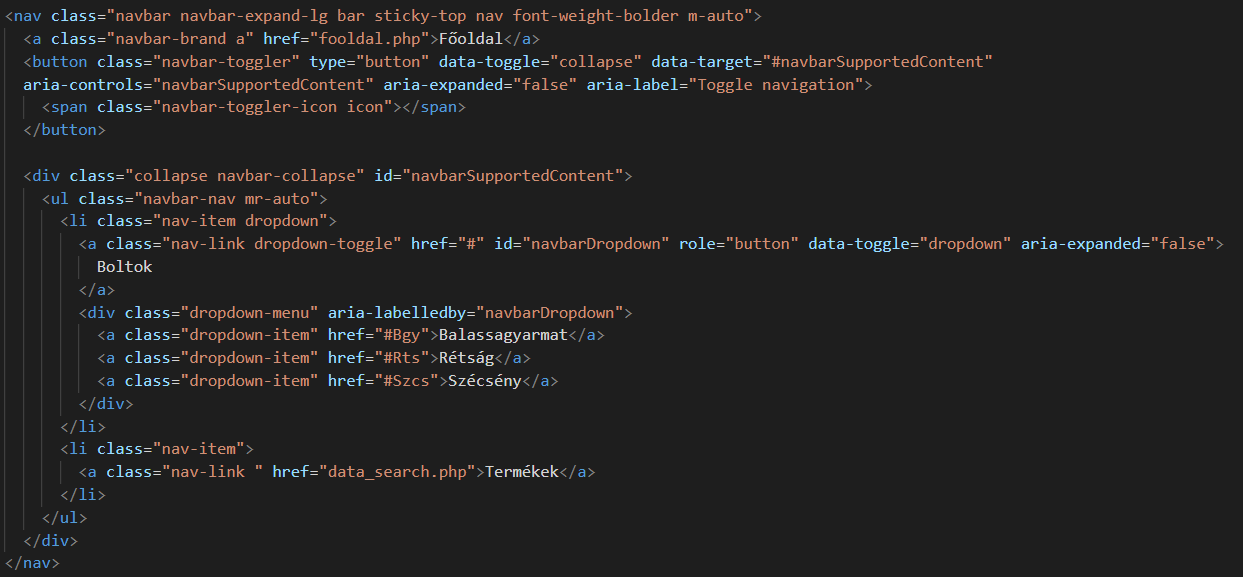
Az aruhazak.php aloldal felső része szintén ugyanaz a jumbotron ami minden oldalon található.

 Ezen oldalon a felső jumbotron alatt egy navigációs menü található:



A navigációs menün 3 fő anchor címke található:

* Főoldal visszaviszi a felhasználót a kezdőoldalra
* Boltok lenyíló menü, alpontjain az egyes azonosítókra visz kattintás után az oldal (mi esetünkben a 3 város áruházaira)
* Termékek átvisz minket a termékek aloldalra, ahol a felhasználó kikeresheti a kívánt terméket

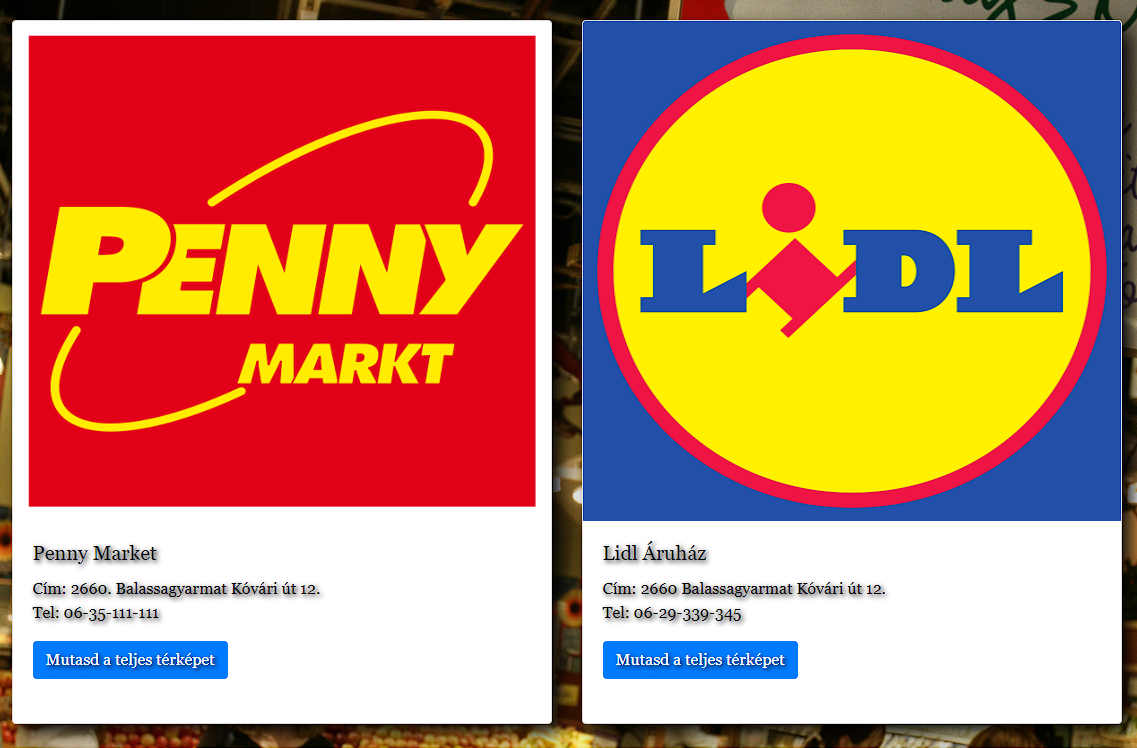


A navigációs menü nav címke amelyben egy div és több lista elem anchor címkékkel fent

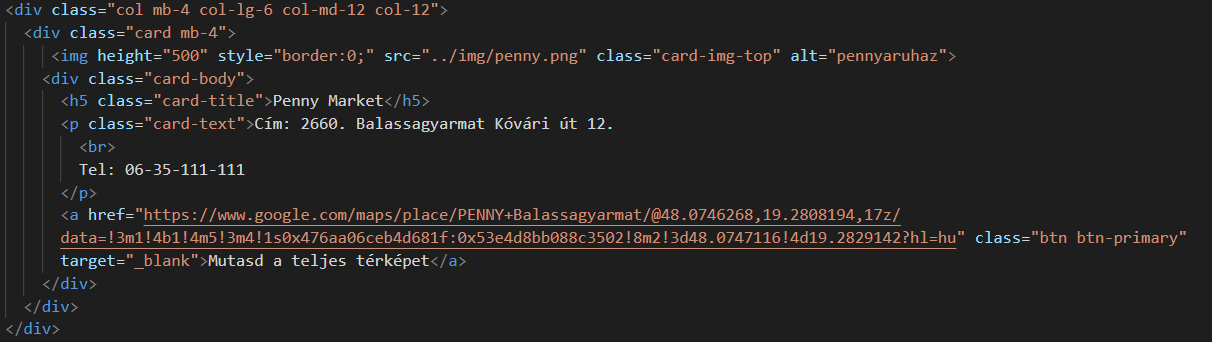
Ezen aloldal fő tartalma a footer rész és navigációs sáv között jelenik meg. Ez az oldal mutatja meg nekünk, hogy a 3 városban milyen kereskedelmi áruházak találhatóak. Témánkban csak az országos és multinacionális vállalatokat egy részét dolgoztuk fel. Helyi szintű kisebb vállalkozásokat nem mutatunk be. A fő tartalom több jumbotronból és abban elhelyezett kártyákból áll 3 részre bontva a body ezen részét. (Balassagyarmat-Rétság-Szécsény). A kártyák száma az áruházak számától függ.



A kép az alap elrendezést mutatja.



A kártyák elhelyezkedése, amelyek tájékoztatást nyújtanak az áruházak adatairól, mint például név, cím. A kártyák két fő részre vannak bontva egy fejléc és egy törzs. A fejlécben minden kártyának külön img címke van elhelyezve az áruházak logóival. A törzs alján egy horgony található, ami a google map térképén külön oldalon megmutatja, térképen hol is helyezkedik el maga az áruház.



A kártyák kódja bootstrap osztályokkal, gridel, képpel, p címkével h5-ös címsorral és horgonnyal a térképes megjelenítéshez.

A város 3 szekciója azonos módon épül fel. A különbség az egyes városokban jelen lévő áruházak neve, és mennyisége között van.

**data\_search.php tartalma**

Ezen az oldalon a felhasználó rákereshet olyan termékre amiről meg szeretné tudni, hogy mennyibe kerül, illetve itt megtudhatja azt is, hogy melyik városban és boltban tudja megvásárolni a legkedvezőbb áron.

Az oldalon van egy navbar aminek a segítségével visszajuthatunk a főoldalra, illetve itt is megtalálható az áruházak oldalra átirányító gomb is, abban az esetben ha a felhasználó meg szeretné tudni hogy pontosan hol van az a bolt ahol meg szeretné vásárolni a megkeresett terméket.

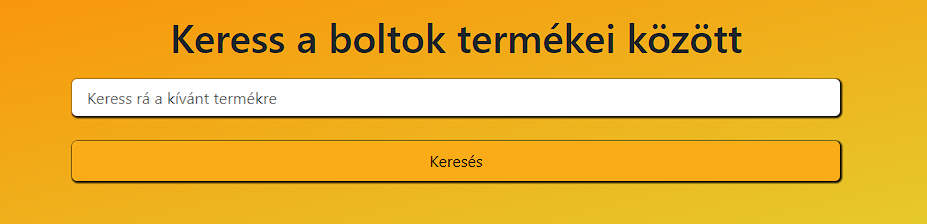


A navbar ami az oldalon látható

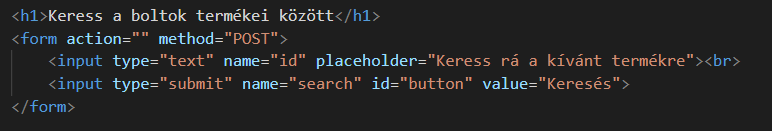


A kódsor ami Bootstrap segítségével lett megírva

Az oldal lényeges funkciója a navbar alatt helyezkedik el, ez nem más mint maga a kereső, ez tartalmaz kettő darab input mezőt. Az első input mező egy text típusú mező, ide tudja a felhasználó beírni a termék nevét vagy a nevének egy részletét, kulcsszavát. A második input mező egy submit típusú mező, ez gyakorlatilag egy gomb amit ha megnyom a felhasználó kilistázza azt a terméket amire rákeresett.



A kereső alaphelyzetben



A kódsor



Az oldal miután rákerestünk egy termékre

A termékek nevének a kiszelektálását a következő képpen oldottuk meg:

Létrehoztuk a kapcsolatot a localhost-al és az adatbázissal, majd egy „if” elágazásban megírtunk egy adatbázis lekérdezést, ami kikeresi a termék, a bolt és a település nevét illetve hogy mennyibe kerül az adott termék akkor, ha a felhasználó olyanra keresett rá ami szerepel az adatbázis termékei között.

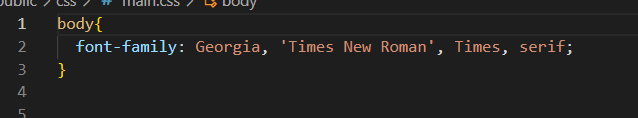
Majd egy „while” ciklussal kiíratjuk az eredményeket.

Próbálkoztunk egy „else” elágazást is bevezetni, abban az esetben, ha valami olyasmit ír be a felhasználó, amit nem tartalmaz az adatbázis. Ezt sok utánajárás és sok videó megnézése után sajnos nem tudtuk kivitelezni, mert egyik példa sem hasonlított a mi kódunkra. Próbálkoztunk mi magunk megírni egy „alert”-et (felhívást), de olyan hibába ütköztünk, hogy nem akkor jelent meg a felhívás, amikor beírtunk valamit a keresőbe és nem talált olyan terméket, hanem minden egyes alakalommal, amikor betöltöttük vagy újratöltöttük az oldalt. Tudtuk, hogy valamit mi csinálunk rosszul, vagy valamit nem veszünk észre, ezért a php-val kapcsolatos tudásunk és idő hiányában ezt az igen fontos és hasznos funkciót sajnos el kellett vetnünk.



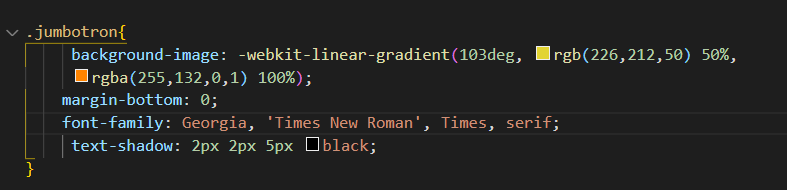
**2.2.3 CSS**

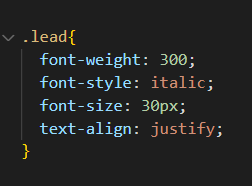
Minden egyes css tartalomnak külön állományt hoztunk létre az aloldalak számát figyelembe véve. Ettől eltér a footer amelynek egy önálló állománya és ami minden oldalon ugyanúgy jelenik meg. Minden egyes oldalon a css elején kiszelektáltuk a body címkét amely tulajdonságának 3 fajta betűtípust adtunk meg.



  
  
A 4 db css állomány

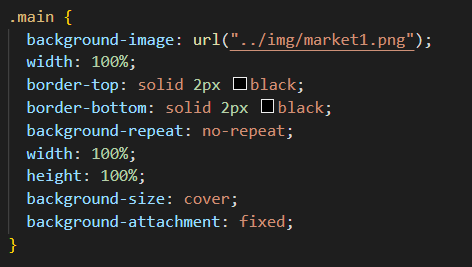
A fejlécekben sárgás keresztirányú átmenetet van beállítva háttérképnek. Georgia, Times New Roman, Times, serif betűcsaláddal. A szövegek kaptak egy fekete árnyékot.

A jumbotron beállításai



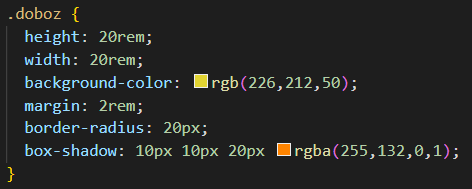
A jumbotron szövegének beállításai.

A kezdőoldal fő tartalmát teljes szélességűre állítottuk be 100%-ra fekete 2px-es határral. Az itt található tartalom képének egy fix beállítású áruházról szóló fotót állítottunk be. Emellett az ismétlődést kikapcsoltuk. A háttér méretét borító (cover) tulajdonsággal ruháztuk fel.

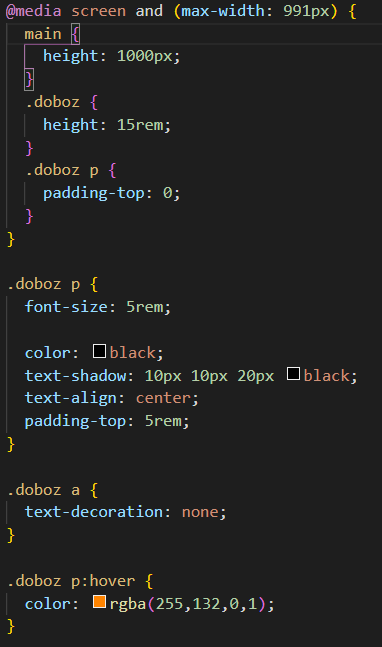


**main.css:**

A két doboznak 20-20rem szélességű magasságú méretet állítottunk be. Háttérszíne citromsárga színű. A dobozok le vannak kerekítve 20px méretben:



média lekérdezéssel 991px méretig illeszkedik a képernyő méretéhez a két doboz illetve az őket tartalmazó div címkék. Mivel eltolódtak volna a dobozokban található szavak ezért a média beállításnál patkó beállítás felső részét 0-ra állítottuk. Az Áruházak és Termékek szavak képernyőméret illeszkedésűek és **fekete** színűek. Középre igazítottak és fekete árnyékkal vannak ellátva. Ha az egérmutatót a szavakra irányítjuk egy lebegő (hover) lekérdezést tapasztalhatunk. Ilyenkor a szavak színe **narancssárg**ára vált át.



A dobozok beállítása

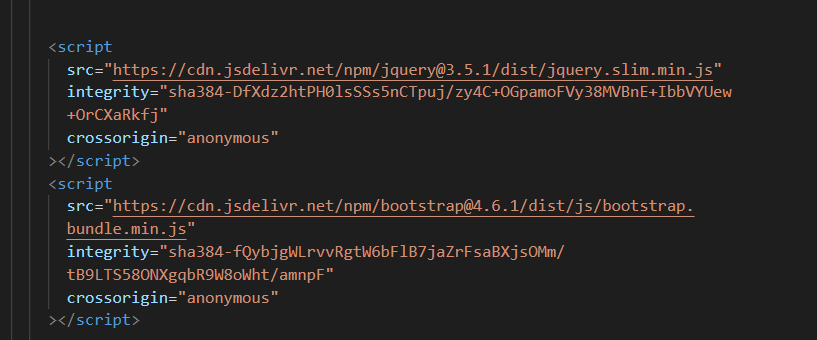


A képen a lebegő (hover) lekérdezés látható működés közben az fooldal.php kezdőoldal középső részén

**2.2.4. Javascript**

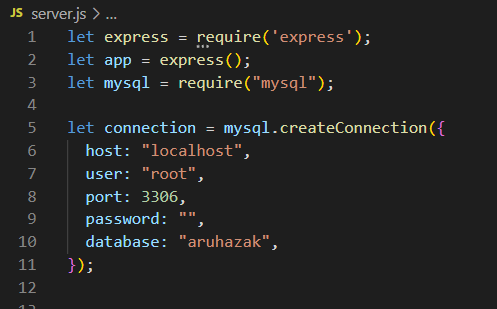
Bár igaz, hogy végül php fájlban csináltuk a projektünket, sokat foglalkoztunk a javascriptel, illetve a NodeJS-el ezért szerettük volna megemlíteni a dokumentációban, hogy eredetileg hogy csináltuk volna js/ejs fájlban.

Alapjában az összes ejs fájlnak be van emelve a booststrap szervereiről a script kód!



**server.js tartalma:**

A server.js tartalmazza a server beállításokat és az egyes aloldalak elérhetőségeit a böngészőben a localhost 8000 portján. A kód elején látható az express és a mysql meghívása, majd egy kapcsolat létrehozása a mysql 3306-os portján. A változóba mentett adatokban látható melyik adatbázisból hívjuk meg az adatokat.

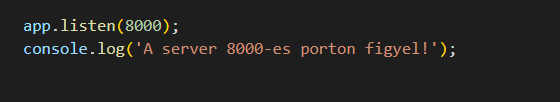


Ezután 2 get metódus és egy set metódus látható. A 2 get metódus függvénnyel meghívja az egyes aloldalak és a kezdőoldal címeit(/, /aruhazak, /termekek). A függvényben a **res** paraméter meghívása és az ejs állományok elérési útjai láthatóak:



A termékek aloldal get metódusának függvénye egy lekérdezést valósít meg az oldal betöltése során. Az adatokat a mysql 3306-os portjáról és phpmyadmin felületről szerzi be. Miután a termekek táblát beimportáltuk a phpmyadmin felületére.

A kódsor végén az app változó révén egy figyelő metódus lett rákapcsolva a rendszerre, amely a 8000-es porton figyel a node futtatása után.



**2.2.5 PHP**

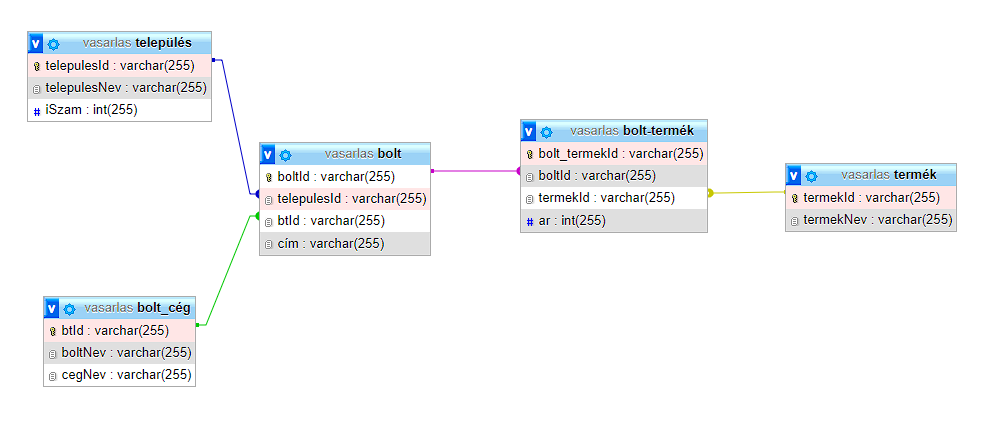
Az adatbázis megtervezést a phpmyadmin felületén készítettük el. Próbáltunk egy olyan adatbázist létrehozni ami nem túl nagy, de mégis benne van számunkra illetve a felhasználó számára a megfelelő mennyiségű adat.

Az adatbázis tartalmaz 4 db egyedet:

* Település
* Bolt
* Bolt\_Cég
* Termék

Az egyedek közötti kapcsolatok:

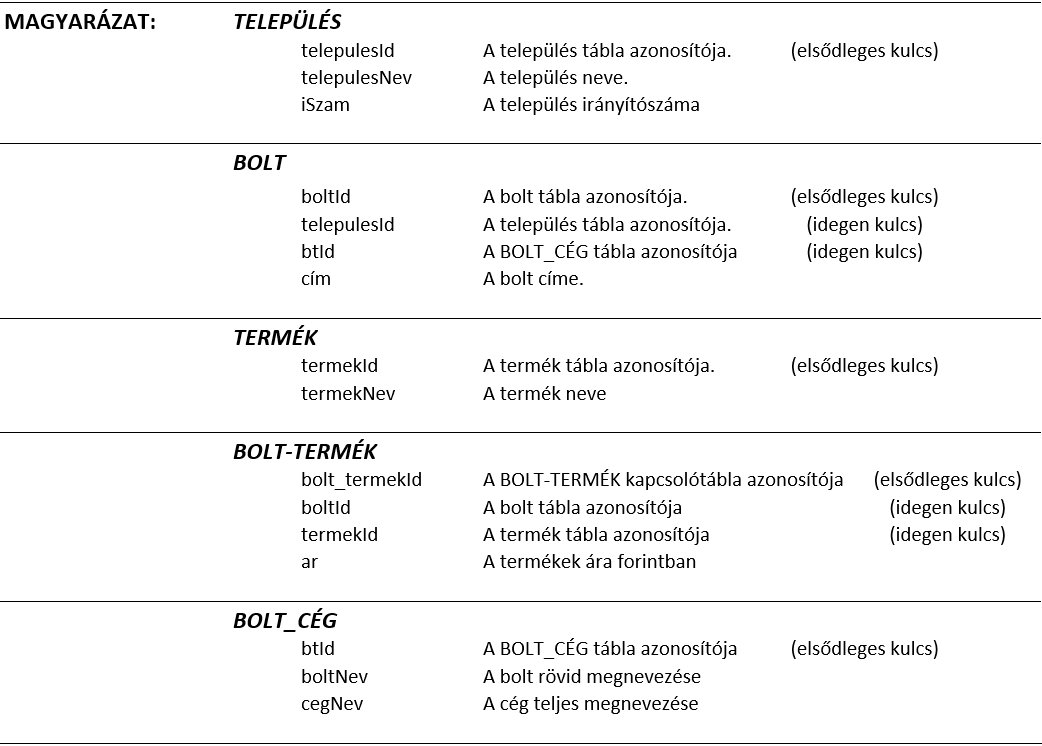
* Település-Bolt: 1:N mert 1 településen több bolt lehet de 1 bolt csak egy településen lehet
* Bolt-Termék: N:M mert 1 boltban több termék lehet és 1 termék több boltban is lehet
* Bolt\_Cég-Bolt: 1:N mert 1 bolt cégnek több boltja is lehet de egy bolt csak egy bolt céghez tartozhat



Mint minden adatbázist és programot lehet tovább fejleszteni, igy ezt is lehet, mi a következőkre gondoltunk:

* időhöz kötni a termékek árát, ezáltal előre tervezhető és utólag követhető a termékek árának változása
* akciók bevezetése, a termékek akciós státuszának beállítása
* raktárkészlet bevezetése, követni a készlet változását, különös tekintettel arra, ha egy termék elfogy
* termékkategóriák bevezetése pl.: péksütemény, ital, édesség
* végül, de nem utolsó sorban az áruházak és a termékek listájának bővítése

Az adatbázishoz tartozó magyarázat



**2.3. Kutatások a kívánt eredmény eléréséhez**

**2.3.1 A téma kutatómunkája**

Mivel biztos és nagy helyismerettel rendelkezünk a 3 járási székhelyén ezért tudtuk hol is milyen kereskedelmi áruházak, bevásárló központok helyezkednek el az adott településen. Következésképpen nem volt akadály hozzáadni a kívánt áruházak nevét az alkalmazáshoz. A google maps segítségével gyorsan és hatékonyan meg tudtuk találni és bejelölni az áruházak helyzeteit a térképeken. Az internet segítségével próbáltuk a telefonos elérhetőségeket is megtalálni, de úgy döntöttünk sok helyre csak kitalált telefonszámot adunk meg. A termékek képeit és az egyes oldalak háttérképeit a google segítségével kerestük ki. A források a szakdolgozat végén láthatóak.

**2.3.2 Az alkalmazáshoz végzett kutatómunka**

Eleinte probléma volt a megfelelő szerver környezet kialakításával. A youtube segítségével különböző csatornákon kutattunk az után, hogy milyen módon kell összekapcsolni a NodeJs-t a Xammp szerver Mysql 3306-os portjával. Mivel egy kicsit meggyűlt vele a bajunk elkezdtünk más megoldásokat keresni, ekkor találtunk rá egy videóra amiben php fájlban csinálták meg az egész lekérdezést és az összekapcsolást és mivel elég követhető volt és nagyjából értettük is, ezért döntöttünk végül úgy, hogy php-ban csináljuk meg mi is. Az egyéb kutatásokat, mint például valamilyen kódrészlet problémájának megtalálása. Különböző fórumokon és közösségi platformokon találtuk meg.

Nagy segítségünkre volt:

* StackOverflow
* Facebook programozói csoportjai
* google

Sajnos a [www.prog.hu](http://www.prog.hu) igen körülményes volt számunkra. Illetve sajnos úgy nem tudtuk leírni problémáinkat ahogy az másoknak is megérthető és az oldal szabályainak is megfelel.

**3. Tesztelés**

Alkalmazásunkat a Visual Studio Code program segítségével Xammp szerveren tezsteltük. Minden egyes fázis vagy akadály megoldása után mindig elindítottuk a rendszert és betöltöttük az oldalakat. Minden egyes indításkor az összes aloldalt megjelenítettük. Teszteléseink jól sikerültek a végső kialakításnál a program a kívánt módon jelenik meg és megfelelően fut le.

**4. Tovább fejlesztési lehetőségek**

Az adatbázis tovább fejlesztéseit leszámítva, rengeteg egyéb más fontos funkciót szerettünk volna még kivitelezni. Mi a következőkre gondoltunk:

A navbar tovább fejlesztett verziója valahogy így nézett volna ki:

(a most következő képek tervek/prototípusok voltak, természetesen más színnel és dizájnnal lettek volna megvalósítva)



A „logo” értelem szerűen egy logó helye lenne, ez saját kezűleg és tervezésileg lett volna elkészítve, előre nem meghatározott szerkesztő program segítségével.

A „sales” az egy olyan aloldal lett volna, ahol a felhasználó rá tud nézni az esetleges akciókra vagy népszerű termékekre. Itt tudomást nyerhet olyan termékekről is, amire eleinte nem gondolt vagy esetleg pont azért nem keresett rá, mert tudta, hogy alapból drága.

A „cart” szintén egy aloldal lett volna, ahol azok a termékek jelennek meg, amikre a felhasználó rákeresett, és kiválasztott. Itt összegezve lennének a termékek árai, valamint csoportosítva lennének a különböző boltok szerint.

A főoldal egy két lépcsős keresőből épülhetne fel:

Az első kereső: Városok szűrése



A városok leszűrése azért lenne egy nagyon fontos funkció, mert ha nem csak 3 várossal dolgozunk, mint például a mi esetünkben, hanem egy megye összes városával, falujával, vagy akár egy országnyi településsel, akkor a felhasználó rengeteg fölösleges adatot látna, amire ő nem kíváncsi.

Vegyünk példának egy balassagyarmati felhasználót, aki rákeres például a tejre, a tej mondhatni az összes élelmiszer boltban jelen van. Ha nem lenne a szűrő, ő azt is látná, hogy Budapesten az Auchan-ban vagy számára egyéb más nem érdekelt boltban mennyibe kerül, ez fölösleges információ a számára.

A városok leszűrését kettő féle képpen lehetne megcsinálni. Az egyik lehetőség főként telefonos felhasználóknak előnyös, ez GPS alapján betájolná, hogy az illető melyik városban tartózkodik.

A második lehetőség, hogy a felhasználó beírja a település nevét.

A városok leszűrése után már rákereshetünk bármilyen termékre, amire csak szeretnénk és csak abban a városban fogja keresni, amit mi megadtunk neki.

A második kereső: a termékek keresése



Ez a kereső már a meglévő termék szűrőn alapszik. Annyi újítással, hogy rá lehetne keresni nem csak bizonyos termékekre, hanem csak simán kategóriákra is például péksüteményekre, alkoholos italokra, zöldségre, gyümölcsre, stb. Ez a funkció is kilistázná ugyan úgy a termékeket annyi változtatással, hogy itt semmi mást nem látna, csak a termék nevét és esetlegesen egy képet a termékről. A képre vagy az áru nevére rá tudna kattintani, és a kattintás után látná csak, hogy mennyibe kerül és, hogy hol.

Ez a változtatás azért kell bele, mert ha kategóriára keresünk rá például a gyümölcsre, akkor itt felsorolná a banánt, almát, körtét, stb… ezek a termékek egyszer lennének megjelenítve és mivel az árak minden boltban különböznek, ha ott lenne az ára is, akkor rengetegszer lennének kiíratva, ami megnehezítené az átláthatóságot és a böngészést. Itt eleinte nem az árára vagyunk kíváncsiak, hanem a kategórián belüli kínálatra. (Remélem érthetően fogalmaztam).

Tovább fejlesztési lehetőségnek gondoltuk még a fentieknél említett „else” elágazással megírt „alert”-et amit nem sikerült megoldanunk. Ehhez a funkcióhoz lehetne csatolni egy adatbázist, ami feljegyezi azokat a termékeket, amikre a felhasználók rákerestek és nem találhatók az adatbázisban. Ez a funkció megkönnyíteni és felgyorsítaná az adatbázis kezelő munkáját, mert így könnyebben és hatékonyabban tudná bővíteni, frissíteni az adatbázis termékeinek a listáját.

Természetesen vannak egyéb lehetőségek is, amiket már fentebb említettünk vagy majd lentebb említeni fogunk. Tovább fejlesztési lehetőségnek ezeket gondoltuk fontosabbaknak, ezért fejtettük ki ezeket bővebben.

**5. Összegzés**

Sajnos a vizsgaremek elkészítéséhez egyéb tanulmányi teendőink mellett igen későn tudtunk hozzá kezdeni kapacitás és nagy leterheltség mellett. Elmondható, hogy problémáink megoldására az idő szűkössége mellett igen sok idő ment. Sokszor nem tudtuk megbeszélni a munkamenetet és, hogy hogyan csináljuk, valamint sokszor nem tudtuk összeegyeztetni az időpontokat, mert valamelyikünknek nem volt alkalmas az az időpont, ami a másikunknak megfelelő lett volna. Ezt az alkalmazást tovább lehet fejleszteni plusz mysql táblák felvezetésével, és még több aloldal hozzáadásával. Egyéb funkciókat is lehet hozzá írni mint például űrlap beküldő rendszer amely fogadja a felhasználók adatait és vagy esetleges észrevételeit. Elmondható, hogy a munkánk során törekedtünk a letisztult színvilágra. Megjelenést tekintve minél egyszerűbb, mondhatni felhasználóbarát formákat alkalmaztunk. A megjelenés kialakításánál tanultunk a régebbi kormányzati portálok hibáiból, amelyekben össze voltak zsúfolva a tartalmak. Rájöttünk ennek a következménye az lehet, hogy egy olyan magyar állampolgár, aki el szeretne valamit intézni, annak nagy nehézséget okoz és okozhat egy-egy ilyen nagy kidolgozottságú portálon a kívánt dolog megtalálása vagy elintézése. Ezért mi próbáltuk felülről lefele haladó elrendezést alkalmazni. Emellett nagyobb méreteket alkalmaztunk a könnyebb olvashatóság, és vizualitás érdekében. Természetesen tudjuk egy országos szintű portálon több tartalomnak is meg kell jelennie, de mivel kutattunk a felújított állami portálokon is, arra a következtetésre jutottunk, hogy a mai trend inkább már a nagyobb könnyen értelmezhető, és navigálható oldalakat részesítik előnyben, szemben a régi, túlzsúfolt, kis méretű szöveggel ellátott oldalakkal. A színeknél fontos volt számunkra a meleg színösszeállítás, amely úgy gondoljuk egyezhetett a témával és a megjelenített képekkel.

**Továbbá szeretnék köszönetet mondani Szilágyi Péter Mechatronikai Mérnök barátomnak, aki igen sokat segített a vizsgaremek elkészítésében. Nagy öröm volt számomra vele együtt dolgozni. Ő útmutatásával és egy – egy probléma megoldásához vezető szakasz megsegítésével sikerült elvégezni a kívánt alkalmazást.**

**Tartalomjegyzék**

**1. …........................................................................Bevezetés**

**1.1. ….........................................................Témaválasztás Indoklása**

**1.2. …............................A Dokumentáció tartalmának rövid bemutatása**

**2. .….....................A weboldal/webalkalmazás bemutatása**

**2.1. .…..............................................................Fejlesztői Környezet**

**2.1.1 …................................................................Visual Studio Code**

**2.1.2. ….............................................................................XAMPP**

**2.1.3. ….............................................................................Dbforge**

**2.1.4. …..............................................................................MySql**

**2.1.5. …..............................................................................HTML**

**2.1.6. ….................................................................................CSS**

**2.1.7. …..........................................................................Javascript**

**2.1.8. ….............................................................................NodeJs**

**2.2. …..........................................................Fejlesztői Dokumentáció**

**2.2.1. ….......................................................Felhasznált technológiák**

**2.2.2. ….................................................................................html**

**2.2.3. …..................................................................................CSS**

**2.2.4. …..........................................................................Javascript**

**2.2.5. …………………………….…………………………PHP**

**2.3. ….......................................Kutatások a kívánt eredmény eléréséhez**

**2.3.1 …...........................................................A téma kutatómunkája**

**2.3.2.......................................... Az alkalmazáshoz végzett kutatómunka**

**3. ….............................................................................Tesztelés**

**4. ……………………………..Tovább fejlesztési lehetőségek**

**5. …...........................................................................Összegzés**

**Források**

**Áruházak**

**Lild logó:**

<https://www.google.hu/search?q=Lidl+logo&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjk-M3rg6z3AhUKwKQKHaNKBI4Q_AUoAXoECAIQAw&biw=1536&bih=722&dpr=1.25#imgrc=15wxQK5riqfDEM>

**Penny Logó:**

<https://www.google.hu/search?q=penny+logo&tbm=isch&ved=2ahUKEwjKs9bug6z3AhVP86QKHeExCm4Q2-cCegQIABAA&oq=Penny+&gs_lcp=CgNpbWcQARgAMgQIABBDMgQIABBDMgQIABBDMggIABCABBCxAzIFCAAQgAQyBQgAEIAEMgUIABCABDIFCAAQgAQyBQgAEIAEMgUIABCABDoKCAAQsQMQgwEQQzoGCAAQBxAeOgQIABADOggIABCxAxCDAToHCAAQsQMQQ1CcD1jLG2DwLmgAcAB4AIABvgGIAYIIkgEDMC43mAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=k-hkYoqeIs_mkwXh46jwBg&bih=722&biw=1536#imgrc=8UOjig_LExFPEM>

**Coop Logó:**

https://www.google.com/search?q=coop+logo&sxsrf=ALiCzsbUcX-bTkkkfs0QvHn\_oDDdUdLduQ:1651534203643&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjEm6us\_MH3AhXsAxAIHdPbCKIQ\_AUoAXoECAIQAw&biw=1920&bih=937&dpr=1#imgrc=v5S2TinpgWagVM&imgdii=2Ilxqgjg-UNZgM

**Reál Logó:**

[https://www.google.hu/search?q=Re%C3%A1l+logo&tbm=isch&ved=2ahUKEwjwuYyBhaz3AhUNuaQKHRUGBu8Q2-cCegQIABAA&oq=Re%C3%A1l+logo&gs\_lcp=CgNpbWcQAzIFCAAQgAQyBggAEAgQHjoECAAQQzoGCAAQBxAeOgsIABCABBCxAxCDAToICAAQsQMQgwE6CAgAEIAEELEDOgQIABADOgQIABAYUMOxBFjNyQRgoMsEaAFwAHgAgAGmAYgBxgmSAQM0LjaYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=xulkYvC-Jo3ykgWVjJj4Dg&bih=722&biw=1536#imgrc=WYMxY9Rbz29y1M](https://www.google.hu/search?q=Reál+logo&tbm=isch&ved=2ahUKEwjwuYyBhaz3AhUNuaQKHRUGBu8Q2-cCegQIABAA&oq=Reál+logo&gs_lcp=CgNpbWcQAzIFCAAQgAQyBggAEAgQHjoECAAQQzoGCAAQBxAeOgsIABCABBCxAxCDAToICAAQsQMQgwE6CAgAEIAEELEDOgQIABADOgQIABAYUMOxBFjNyQRgoMsEaAFwAHgAgAGmAYgBxgmSAQM0LjaYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=xulkYvC-Jo3ykgWVjJj4Dg&bih=722&biw=1536#imgrc=WYMxY9Rbz29y1M)

**Tesco Logó:**

<https://www.google.hu/search?q=tesco+logo&tbm=isch&ved=2ahUKEwjHuOv2hqz3AhUMyaQKHfG5B40Q2-cCegQIABAA&oq=tesco+logo&gs_lcp=CgNpbWcQAzILCAAQgAQQsQMQgwEyBQgAEIAEMgUIABCABDIGCAAQBxAeMgYIABAHEB4yBggAEAcQHjIGCAAQBxAeMgYIABAHEB4yBggAEAcQHjIGCAAQBxAeOgQIABBDOgQIABATOggIABAHEB4QE1DOCFj4G2COIWgAcAB4AIABoAGIAZYGkgEDMy40mAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=yetkYoewPIySkwXx857oCA&bih=722&biw=1536#imgrc=-6Wi4KbxdMaKcM>

**Spar Logó:**

https://www.google.com/search?q=spar+logo+wallpaper&tbm=isch&ved=2ahUKEwitkpqt\_MH3AhWXhP0HHdwcAPwQ2-cCegQIABAA&oq=spa&gs\_lcp=CgNpbWcQARgAMgcIIxDvAxAnMgoIABCxAxCDARBDMgQIABBDMgsIABCABBCxAxCDATIFCAAQgAQyBAgAEEMyBAgAEEMyCAgAEIAEELEDMggIABCABBCxAzIECAAQQzoGCAAQBxAeOgcIABCxAxBDUN0HWJUKYIQPaABwAHgAgAFhiAH8ApIBATSYAQCgAQGqAQtnd3Mtd2l6LWltZ8ABAQ&sclient=img&ei=fWlwYu3fG5eJ9u8P3LmA4A8&bih=937&biw=1920#imgrc=ebd2O1jrYa03yM

**YouTube**:

<https://www.youtube.com/watch?v=n5NYVpFRSB0&t=468s>

https://www.youtube.com/watch?v=wtgiEluCbhc

**Egyéb:**

**Visual Studio Code:**

<https://hu.wikipedia.org/wiki/Visual_Studio_Code>

**Xammp:**

<https://hu.wikipedia.org/wiki/XAMPP>

**NodeJs:**

<https://hu.wikipedia.org/wiki/Node.js>

**NodeJs szerverkód a szemléltetésre:**

<https://www.w3schools.com/nodejs/nodejs_get_started.asp>