1. Bevezetés: 2

2. Specifikáció: 3

3. Alkalmazott technológiák: 4

3.1 A PHP fogalma 4

3.2 A PHP működése 5

3.3 Laravel 5

3.3.1 Függőségek kezelése 5

3.3.2 Modularitás 6

3.3.3 Hitelesítés 6

3.3.4 Útvonaltervezés 6

3.3.5 Biztonság 6

Felhasznált források: 11

EZ ITT MEG NEM JO, NE NÉZZED, KI KELL JAVÍTANI

# Bevezetés:

Manapság 2022-ben és az elmúlt években egyre nagyobb teret hódítottak az online vásárlási felületek és az IT szféra fejlődésének köszönhetően egyre jobban arra felé hajlunk, hogy amit csak lehet, azt valamilyen online felületen végezzük, ez az ügyintézéstől fogva a meetingek és találkozók megtartásán keresztül egészen a vásárlásig terjedt ki. Ezért döntöttünk úgy mi sem állunk a változás újtába és a növekvő igénynek megfelelve, mi ezen a területen szeretnénk létrehozni átfogó porjektet, amely teljesen életképes a ma, azaz 2022 igényeinek, teljes mértékben megfelel és kielégítő felhasználói élményt nyújt.

Fejünkben többféle remek ötlet is megfordult, példának okáért egy futárszolgálat weboldala, online kaszinó ötlete és egy mezőgazdasági videómegosztó platform is, de végül egy webáruház gondolata tűnt a leg ideálisabbnak és a leg kézenfekvőbbnek.

A webshop tárgyát/témáját illetően több lehetőség is megfogalmazódott;

Például:

* Bútorbolt
  + Indoklás:
    - A környezetünkben több irányultságú cég található, a projektet megpróbáltunk volna náluk értékesíteni
  + Elhatározás:
    - Végül is úgy döntöttünk, hogy más ötleteket is megvizsgálunk
* „OnlySelfies” online tartalommegosztó oldal
  + Indoklás:
    - Magas kereslet a hasonló monotizálható tartalommegosztó oldalak iránt
  + Elhatározás:
    - Ezt a praktikát etikátlannak találtuk és így az ötletet elvetettük
* Online kaszinó
  + Indoklás
    - Az online tartalomgyártók promócióiknak köszönhetően, ez az iparág világszerte nagy teret hódított az elmúlt pár év alatt.
  + Elhatározás
    - A piac telítettsége miatt más ötlet mellett döntöttünk

# Specifikáció:

Vésősoron egy webáruház megvalósítása mellett döntöttünk. Mivel napjainkban az online vásárlás nagy teret hódított, ezért kézenfekvőnek tűnt, hogy mi is egy webshopot készítsünk. Véleményünk szerint ez a projekt kellően összetett és a modern igényekhez mérten aktuális ahhoz, hogy a vizsga követelményeinek megfeleljen.

A projektünk egy teljes értékű, adatbázissal támogatott webáruház, különválasztott backend és frontend használatával.

A webshop teljesértékű felhasználói élményt nyújt: a vásárló tud böngészni a feltűntetett termékek között, rá tud keresni konkrét termékre, vagy kategóriák szerint szűrni. A kiválasztott termékről tud bővebb információhoz jutni és akár regisztráció nélkül is a kiválasztott terméket kosárban tudja tárolni, továbbá a felhasználó regisztráció nélkül is képes rendelést leadni.

Lesz lehetőség regisztrált felhasználóknak, wishlist létrehozására, abban az esetben, ha egy wishlist-en lévő termék leárazásra kerül, akkor a felhasználót emailben értesítjük az új leárazásokról.

Lehetőség lesz arra is, hogy a felhasználó kérjen értesítő emailt, hogy mindig naprakész információja legyen a legújabb akciókról és új termékekről.

Regisztrált felhasználók továbbá tudnak hűségpontokat gyűjteni, ha egy bizonyos érték felett vásárolnak.

# Alkalmazott technológiák:

Szükségünk van keretrendszerekre, illetve azok alkalmazására, hiszen ezeknek a segítségével, jóval gyorsabban dolgozhatunk, valósíthatjuk meg az adott funkcionalitást. A keretrendszer lényege az, hogy a különböző alkalmazásokban legnagyobb gyakorisággal használt elemeket egy helyre csoportosítja, ezáltal gyakorlatilag készen kínálva ezeket a webfejlesztők, illetve a különböző programok számára.

A programozók így nagyon sok elvégzendő munkától szabadulnak meg. A keretrendszerek egységesen önmagukban nem használhatóak, de nagymértékben könnyítik az egyes tipikus feladatok elvégzését. A komplexebb folyamatokat leegyszerűsíti, tiszta és újra felhasználható kódot írhatunk velük. Megkönnyíti a hibakeresést és a tesztelést, rugalmasabb alkalmazásokat eredményez, illetve webshopkészítés során is igen hasznos lehet.

Ennek következtében, ami a backendet illeti a natív nyelven írt backend ötletét elvetettük, a szükségtelen bonyolultsága miatt.

Ezen felül, gondoljunk csak a böngésző kompatibilitásra: ez a probléma nem csupán CSS esetén merülhet fel. A megírt kódunk nem egyformán fog működni minden böngésző, illetve azok verziói alatt.

Elsősorban két lehetőséget vettünk fontolóra: A PHP alapú Laravel framework-öt és a JavaScript alapú Express.js –t.

## A PHP fogalma

A PHP egy általános szerveroldali scriptnyelv, melynek segítségével dinamikus weboldalakat készíthetünk. Az első scriptnyelvek egyike, melyet HTML oldalakba ágyazhatunk.

Hogy pontosan mire is jó ez? Képzeljünk el egy weboldalt, ami csak HTML fájlokból áll, ezzel a weboldallal a képek, és szövegek megjelenítésén kívül mást nem tudunk megjeleníteni, csak megnézni. Ha szerkeszteni szeretnénk, csak a forrás módosításával tehetjük meg.

Ahhoz, hogy a weboldalunkat rávegyük űrlapok feldolgozására, e-mailek küldésére, webshopok üzemeltetésére, PHP-re (vagy más programnyelvre) lesz szükségünk.

A PHP népszerűsége rendkívül gyorsan nőtt, és nem sokkal később már minden nagyobb weboldal ezt használta, többek között a Facebook, Google, Amazon, eBay és még sokan mások is.

Népszerűségét az előre definiált függvényekkel és könyvtárakkal, gyorsaságával, és egyszerű használatával érte el.

## A PHP működése

A PHP a HTML-el ellentétben nem kliens (felhasználó) oldali, hanem szerver oldali nyelv, ami azt jelenti, hogy a kiszolgáló nem küldi el a PHP kódot az kliens-nek, hanem a kiszolgáló oldalán a PHP-értelmező motor dolgozza fel.

A PHP kódok végezhetnek adatbázis műveleteket, e-mail műveleteket, fájlokat hozhatnak létre, módosíthatnak, törölhetnek, képeket szúrhatnak be, de akár módosíthatják a HTML kódokat is. A PHP kódok kimenetele, csak a HTML elemekkel együtt kerül a felhasználóhoz.

## Laravel

A Legelterjedtebb PHP keretrendszer Laravel névre hallgat. Ami a Laravel-t illeti mindenképpen előnyt jelentett, hogy maga a framework nagyon jól dokumentált és napjainkban továbbra is széles körben elterjedt és nagy népszerűségnek örvend. A Laravel keretrendszerének hatalmas ökoszisztémája van, azonnali telepítéssel, útválasztással, sablonnal, ORM, DB lekérdezéssel és listázással.

### 3.3.1 Függőségek kezelése

A függőségek vezetése a Laravel egyik legnagyobb tulajdonsága. A Laravel osztályfüggéseinek kezelésére a leghatékonyabb eszköz az IoC (Inversion of Control) vagy a szerviztartály. A függőségi injekció egy eszköz, amellyel keményen kódolt (hard-coded) modulokat lehet eltávolítani és befecskendezni composer segítségével.

### 3.3.2 Modularitás

A modularitás a webalkalmazás részeinek elválasztásának és rekombinációjának mértékét jelenti. A programlogikát fel lehet osztani különféle összetevőkre, amelyek együttműködnek. A Laravel segítségével könnyen megtervezhetünk és létre is hozhatunk széles körű webalkalmazást.

### 3.3.3 Hitelesítés

A hitelesítés minden modern webalkalmazás elvárt eleme. A fejlesztés sok időt vehet igénybe, és nem is egyszerű megvalósítani. A Laravel már egy kész modult kínál, amely lehetővé teszi egy teljesen működőképes hitelesítési séma létrehozását egy egyszerű paranccsal. A dokumentáció segítségével meglehetősen könnyen testre is szabhatjuk.

### 3.3.4 Útvonaltervezés

A Laravel útvonala-rendszere meglehetősen hatékony és egyszerű. Csoportosíthatjuk, elnevezhetjük, szűrhetjük és összekapcsolhatjuk a modellinformációkat/vezérlőket az utakkal, amitől rendezettebbé, átláthatóbbá válik a projectünk.

### 3.3.5 Biztonság

A Laravel intuitív módszert kínál biztonságos webes alkalmazások létrehozására. A sima szöveges jelszavak helyett az összes jelszót kivonatként menti el. A jelszavak kivonásához BCrytp-t használ. SQL injektálási támadás biztonságot nyújt, és elkerüli az összes felhasználói bejegyzést, hogy megakadályozza a szkriptcímkék befecskendezését.

Viszont jelenleg a JavaScript alapú rendszerekben több tapasztalattal rendelkeztünk ezért máshogy döntöttünk.

-

-

-

Az Express.js –re esett végül a választásunk.

A framework szintén az MVC modellt követi, defőleg az miatt döntöttünk az express mellett, mert ez egy JavaScript alapú framework melynek syntax-át jelentősan jobban kedveltünk a PHP-val szemben. AZ express szintén egy elterjedt megoldás a backendek terén és nekünk is több tapasztalatunk van ennek a megoldásnak a használatában és nagyobb óraszámban is tanultuk, bár az utóbbi nemfeltétlenül volt szempont mivel funkcióiban mind a két framework használatával megtudjuk csinálni ugyanazt a projektet.

A frontend terén is több lehetőséget vettünk fontolóra:

A Typescript alapú Angular és a Javascript alapú Vue.js lehetőségét mérlegeltük. Egyedül natív kódolással hasonló okok miatt most sem szerettünk volna dolgozni, mindenképpen most is az imént említett lehetőségekből szerettünk volna választani és natív Javascript kódot csak akkor szerettünk volna alkalmazni, amikor olyan dolgot szerettünk volna megvalósítani, amelyet a framework nem tartalmaz.

Az Angular bár napjainkban egy elég elterjedt megoldás, a környezetünkben is elismert cégek programoznak benne, mi személyes preferencia alapján a Vue.js mellett döntöttünk.

Szeretnénk megjegyezni, hogy bár most nem az Angulárra esett a választásunk, de mindenképpen megéri most a közös projekt mellett és a későbbiekben is, nagy hangsúlyt fektetni rá, hiszen a jelenlegi ismereteink alapján elmondhatjuk, hogy sok lehetőség van benne. A Typescript bár egyesek mondhatják, hogy nehezebb nyelv, mint a Javascript, mindenképpen megéri a bele fektetett időt, melyet később majd a személyes gyakorló projektek által igenis bele fogunk fektetni.

A frontedet Vue.js-ben írjuk!

A Vue.js egy **nagyon könnyen megtanulható, és használható JavaScript könyvtár**. Alacsony Learning Curve-vel rendelkezik.

A Vue.js-t annak ellenére, hogy könnyebben elsajátítható, rengeteg feladatra könnyedén fel lehet használni. Kifejezetten szeretik startupok is használni, valamint kiváló párosítást alkot [Laravel](https://gremmedia.hu/edukacio/bejegyzes/laravel-keretrendszer-mit-erdemes-tudni-rola-tenyleg-legnepszerubb-keretrendszer)keretrendszerrel.

Igaz most ezt az előnyét nem fogjuk kihasználni.

Szintén pozitívumnak könyveltük el, hogy a Vue.js-ben minden eddiginél könnyebben tudtunk teszt adatokat felvenni olyan struktúrában, amilyenben majd már élesben fogja kapni az adatokat és így a kezdetektől fogva koncentrálni tudtunk egy dinamikus frontend létrehozására.

Ami az adatbázis típusát illeti, szintén két lehetőség közül választottunk: a MySQL és a NoSQL között.

Bár a MySQL tűnhet az egyértelmű megoldásnak, a NoSQL lehetőségét komolyan fontolóra vettük és végül az mellett is döntöttünk.

Erre a döntésre nem voltak hasonló indokaink, mint az előzőekben, hiszen mind a két megoldás teljesen valid és széles körben használt megoldás.

Az adatstruktúrák jelentősen különböznek, de mind a két megoldás teljesen átlátható tehát ez nemvolt szempont. A NoSQL az adatokat BSON formátumban tárolja majd küldés előtt alakítja át JSON-be, ez miatt az adatforgalom gyorsabb, bár ez az előny leginkább nagyobb adathalmazoknál mutatkozik meg, a mi projektünknél ez az előny elhanyagolható.

Végsősoron a döntésünket személyes preferencia alapján hoztuk meg, így tehát NoSQL-t fogunk használni.

Fejlesztői dokumentáció:

Kommunikáció a csapattagok között:

A csapat tagjainak a fejlesztés során volt lehetősége személyes kapcsolattartásra így a közös munka zavartalanul tudott zajlani a fejlesztés túlnyomó részében, ám amikor ez az állapot nem állt fen, több különböző módot is felhasználtunk a kapcsolattartásra.

Az elsődleges módja a kommunikációnak a facebook használata volt, illetve az azon való chatelés, de a fejlesztés első fázisaiban más módszereteket is fontolóra vettünk.

A Discord is nyújtott volna egy stabil platformot a zavartalan kommunikáció fenntartására. A Discord egy kommunikációs platform, amely lehetőséget nyújt több fél közti chatelés, streamelés és meetek indítására, továbbá képek fájlok és egyéb kisebb tartalmak közzétételére.

Bár ezek mind előnyösek lehetnek a zavartalan munka fenntartásában, a facebook lehetőséget biztosított az egymás között folytatott üzenetküldésre, a fájlmegosztásra pedig a github szolgáltatásait vettük igénybe. A streamelésre és a konferenciák indítására nemvolt szükség.

A projektet a github-on vezettük ahová a projekten történő változtatásainkat és munkáinkat commiteltük. Ez és hogy visszatudjuk állítani a projekt egy korábbi verzióját hiba észlelése esetén arra indított minket, hogy a github verziókezelő segítségével fejlesszük és tároljuk a projektet, továbbá ezt használjuk fel a projekt beadására is.

Frontend fejlesztés dokumentációja:

Mielőtt a frontend fejlesztésének neki láttunk volna, megkellett határoznunk, hogy milyen legyen a dizájn, és hogy milyen komponenseket szeretnénk belerakni a projektbe.

A dizájn és a komponensek meghatározásának az érdekében, egy meeting keretein belül a döntéseinket egy dizájn demo elkészítésével örökítettük meg.

Ez nem a végleges verzió, sokkal inkább egy iránymutató, arról hogy a dizájn terén milyen irányba menjen el a frontend. Ami a dizájn milyenségét illeti több oldalról is inspirációt merítettünk: többek között a tchibo oldaláról.

Már a projekt létrehozásakor tudtuk, hogy a bootstrap külső komponenseket, stílus elemeket és a grid rendszert szeretnénk majd használni ezért ez már a projekt elején telepítésre került. További a Primevue által felkínált listákat és formokat is fontolóra vettük.

Navigation komponens:

Bár az imént emlétett források tartalmaznak előre elkészített navigation panelt, úgy döntöttünk, hogy nem vagyunk teljes mértékben megelégedve a felkínált lehetőségekkel ezért itt, egy külső komponens használata helyett, saját komponens megírása mellett döntöttünk.

A komponensel kapcsolatban több alap kikötésünk is volt.

Az első kikötés a reszponzivitás volt, jól kell mutatnia mind mobilon mind nagyobb felbontású képernyőkön. Ezt a kritériát és a tényt, hogy mobilon nézetben nehezebb több információt megjeleníteni anélkül, hogy a felhasználót elárasztanánk üzenetekkel, a weboldal átláthatóságát csökkentenénk és a felhasználói élményt rontanánk, ez a feladat olyan megoldást igényelt, amely lehetővé teszi, hogy a felhasználó kontrollálni tudja a megjelenített információ mennyiségét.

Az imént említett gondolatokat figyelembe véve, felismertük, hogy a felhasználónak nemszükséges minden pillanatban látnia a menüt mobil nézetben, ezért annak és a navigációs menühöz kapcsolódó paneleknek a megjelenítését dinamikussá, láthatóságát egy fix pozíciójú gomb által testre szabhatóvá tettük.

# Felhasznált források:

[PHP programozás alapjai 1. rész (phpmuhely.hu)](https://phpmuhely.hu/muhely/segedletek/olvas/php-programozas-alapjai-1-resz)