ZÁRÓDOLGOZAT

Dreilinger Vanessza Maja

Családi Alexandra

2024

**Kertészeti Webárúház**

**Dreilinger Vanessza Maja  
Családi Alexandra**

**SZÁMALK-Szalézi Technikum és Szakgimnázium**

**Szoftverfejlesztő**

**Nyilatkozat**

a záródolgozat eredetiségéről

Alulírott \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (név) {\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. (anyja neve) \_\_\_\_\_\_\_\_\_\_\_\_. (szem. ig. szám)} büntetőjogi és fegyelmi felelősségem tudatában kijelentem és aláírásommal igazolom, hogy a záródolgozat saját munkám eredménye. A felhasznált irodalmi és egyéb információs forrásokat az előírásoknak megfelelően kezeltem, a záródolgozat készítésre vonatkozó szabályokat betartottam.

Kijelentem, hogy ahol mások eredményeit, szavait vagy gondolatait idéztem, azt a záródolgozatomban minden esetben, beazonosítható módon feltüntettem, a dolgozatban található fotók és ábrák közlésével pedig mások szerzői jogait nem sértem.

Kijelentem, hogy a záródolgozatom elektronikus változata teljes egészében megegyezik a nyomtatott formával.

Hozzájárulok ahhoz, hogy az érvényben lévő jogszabályok és a Számalk-Szalézi Szakgimnázium belső szabályzata alapján az iskola saját könyvtárában megtekinthető (olvasható) legyen a záródolgozatom.

A záródolgozat titkos/nem titkos.

Budapest, 2022. április …..

Tanuló aláírása

Tartalom

[1 Bevezetés 7](#_Toc165027764)

[1.1 Témaválasztás 7](#_Toc165027765)

[2 Tervezés, drótvázak 8](#_Toc165027766)

[2.1 Tervezés folyamata 8](#_Toc165027767)

[2.2 Drótvázak 9](#_Toc165027768)

[3 Felhasználói dokumentáció 12](#_Toc165027769)

[3.1 A program célja és lényegesebb funkciói 12](#_Toc165027770)

[3.2 Futtatási környezet 12](#_Toc165027771)

[3.3 A program fő funkcióinak leírása 12](#_Toc165027772)

[3.3.1 A vásárló funkciói 12](#_Toc165027773)

[3.3.2 Az admin funkciói: 13](#_Toc165027774)

[4 Fejlesztői dokumentáció 14](#_Toc165027775)

[4.1 Fejlesztői környezet 14](#_Toc165027776)

[4.2 Program struktúra 14](#_Toc165027777)

[4.3 Adatbázis 14](#_Toc165027778)

[4.3.1 Egyed-kapcsolati diagram 14](#_Toc165027779)

[4.3.2 Az adatrendszer ismertetése 14](#_Toc165027780)

[4.3.3 Az ügyviteli funkciók 15](#_Toc165027781)

[4.3.4 Az adatbázis-tervezés a szükséges egyedek kialakításával 15](#_Toc165027782)

[4.3.5 Táblák 17](#_Toc165027783)

[4.3.6 Kapcsolati ábra 20](#_Toc165027784)

[4.3.7 Hasznos Lekérdezések 20](#_Toc165027785)

[4.3.8 Kiemelt hasznos lekérdezések 21](#_Toc165027786)

[4.3.9 Beépítendő megszorítások 22](#_Toc165027787)

[4.3.10 Technikai táblák (nem kötelezően megvalósítandó) 22](#_Toc165027788)

[4.4 Backend 23](#_Toc165027789)

[4.4.1 Modellek és Controllerek 23](#_Toc165027790)

[4.4.2 API végpontok 24](#_Toc165027791)

[4.5 Frontend 26](#_Toc165027792)

[4.6 Reszponzivitás 26](#_Toc165027793)

[5 Tesztelés 27](#_Toc165027794)

[6 Fejlesztési lehetőségek 28](#_Toc165027795)

[7 Irodalomjegyzék 1](#_Toc165027796)

**Ábra jegyzék**

**Nincs ábrajegyzék-bejegyzés.**

# Bevezetés

A Szakdolgozat témája egy elképzelt Kertészeti webáruház, ahol az elérhető növényeket lehet megtekinteni és megvásárolni.

Az oldalon a vásárlás **regisztrációval** történik, enélkül nem lehet vásárolni.

A regisztráláshoz kell megadni az *email címet* és *nevet*.

A vásárláshoz meg kell adni a felhasználó nevét, számlázási és rendelési címét. Lehetőség van az adatok módosításra is, ha a szállítási adatok változnak. (Továbbá lehetősége van “Kedvencek” listára, ahol elmentheti kedvenc termékeit vagy megvásárolni kívánt termékeit.)

Admin-ként az admin felületen képes új termékek felvételére, azok árának változtatására, és új készlet beszerzésére adott termékből. Ki tudja listázni az összes terméket.

A SuperAdmin kezeli a felhasználókat, **törölheti** őket, továbbá adhat **admin** **jogosultságot**. Mindenre képes, mint amire az admin is.

## Témaválasztás

A témaválasztás a következők alapján történt. Közösen kerestünk számunkra is megfelelő és érdekes témát. Beszélgettünk különböző közös érdeklődési körökről, mi az amit szívesen látnánk viszont. Akadt pár téma, amit szívesen csináltunk volna: virágok, kertészeti cikkek. Végül ezeknek az ötvözetéből készült el a kertészeti webáruház. A végső döntést mellett szólt, hogy sok kertészeti cég weboldala nem megfelelő a felhasználó szempontjából. Például a kényelmetlen elrendezések, a kategorizálások furcsasága. Ezek miatt szerettünk volna egy szép és kényelmes, felhasználóbarát webáruházat létrehozni. Illetve az adatbázis nagysága is döntőérv volt, az elejétől kezdve ötleteltünk, a későbbiekben ez csak bővült.

# Tervezés, drótvázak

## Tervezés folyamata

**Témaválasztás**

A tervezés egy rendkívül fontos folyamat bármely alkalmazás elkészítéséhez.

A mi programunk elkészítésének legelső lépése a témaválasztás volt.

Ezután összeszedtük a legfontosabb alap funkcióit készülendő alkalmazásunkhoz. Itt fontos volt, hogy megvalósítható célokat tűzzünk ki. Ezek után létrehoztunk kettő darab GITHUB projectet, hogy számon tudjuk követni a majd elvégzendő feladatainkat . Ezek a Kertészeti Backend, és Kertészeti Frontend elnevezést viselik.

**Adatbázis tervezés**

A következő nagy lépés az adatbázis szerkezetének megtervezése következett. Először is a legfontosabb adattáblákat kellet meghatározni, ezeknek ki kellet szolgálnia a már megtervezett funkcióink szükségleteit. Egy másik fontos lépésezzel párhuzamosan a táblák egymással való kapcsolatának kialakítása volt. Továbbá eldöntöttük, hogy milyen adatokat szeretnénk ezekben eltárolni, illetve, hogy ezek milyen típusúak.

Természetesen ezeket később normalizáltuk, mivel a tervezés fázisában még nem voltak a legmegfelelőbbek egy élő adatbázishoz. Ezen lépés után elkészítettünk hasznos segéd, és szótártáblákat.

**Látvány tervezés**

**A Whimsical nevű online oldalt használtuk a drótvázak tervezéséhez. Itt először is a főoldal megtervezésével kezdtük, hiszen ez az az oldaldal, amit bármely jövőbéli felhasználó legelőször fog látni. A következő a termékeket kilistázó felület volt a konkrét termék oldalával együtt. A tervek között volt a bejelentkezés és a regisztráció oldalainak megtervezése, ezek hasonló sémára épülnek, nagyon kicsi eltérés van közöttük. A látvány tervezés legvégén elkészítettük az admin oldalak kezdetleges drótvázát.**

**A tervezés közben igyekeztünk egy könnyen kezelhető felhasználó barát oldalt kialakítani, amit intuitívan lehet majd használni. Az**

## Drótvázak

FőoldalA képen szöveg, képernyőkép, diagram, szoftver látható

Automatikusan generált leírás

A képen szöveg, képernyőkép, diagram, kör látható

Automatikusan generált leírásA képen szöveg, képernyőkép, Betűtípus, szám látható

Automatikusan generált leírás

Termékek oldalA képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

Termék oldalA képen szöveg, képernyőkép, szoftver, Weblap látható

Automatikusan generált leírás

Admin és SuperAdmin oldalakA képen szöveg, képernyőkép, szoftver, Operációs rendszer látható

Automatikusan generált leírás A képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

BejelentkezésA képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

RegisztrációA képen szöveg, képernyőkép, szoftver, Számítógépes ikon látható

Automatikusan generált leírás

# Felhasználói dokumentáció

A felhasználó meglátogatja a weboldalt. Az üdvözlőoldal után a legördülő menüből választhat tetszőleges menüpontot.

A főoldal menüpont minden esetben visszavisz a kezdőlapra. Válassza a "minden termék" menüpontot. Itt lehetősége van végigtekinteni a jelenlegi árukészleten. Amennyiben az "Aktualitások"-at választja, megtekintheti az éppen kiemelt ajánlatokat. Ha a növényeket választja, megtekintheti a nem virágos készletet, ennek ellenkezője pedig a Virágok menüpont, ahol csak a virágzótermékek közül válogathat.

A Bejelentkezés/Regisztráció menüpontnál lehetősége van bejelentkezni vagy regisztrálni. Ha az előbbit választja, úgy már az elmentett adatokkal szabadon vásárolhat. A Profilom menüpont alatt, ha kiválasztja az Adatlapom menüpontot, akkor áttekintheti a regisztrációnál felvitt adatait vagy módosíthatja azokat. A kosár menüpontra kattintva áttekintheti a kosárba rakott termékeket, azok mennyiségét, igény szerint törölheti őket.

## A program célja és lényegesebb funkciói

A program célja az, hogy legyen egy szemre könnyű, emészthető és könnyen kezelhető oldal. A lényegesebb funkciói a vásárlás, bejelentkezés, regisztráció, admin. Az admin fontos feladatokat lát el: adatokat visz fel (például: növény, termék), adminjogosultságot ad a felhasználóknak, törölhet felhasználót.

## Futtatási környezet

## A program fő funkcióinak leírása

### A vásárló funkciói

Regisztráció: regisztrálhatja magát a vendég, adatait a felületen rögzíti. Ezzel biztosítja a lehetőséget a vásárlásra.

Bejelentkezés: számontartja a vásárlót és annak adatait, elérhetővé teszi a vásárlást.

Kosár: számontartja a vásárolni kvánt mennyiséget adott termékből. A felhasználó ürtheti a kosarat kívánsága szerint. (Session lejárta után ürül, nem tárolja a végtelenségig a vásárlást).

Vásárlás: kívánt termékeket kosárba rakhatunk és megvásárolhatjuk igény szerint.

### Az admin funkciói:

Adatfelvitel: különböző adatokat különböző menüpontok alatt felvihet. Ezek bekerülnek az adatbázisba. Ezeket az új adatokat a későbbiekben másik menüpont alatt tudja szerkeszteni: módosíthatja az adathoz tartozó információkat, például elírás esetén, vagy akár teljesen törölheti azokat.

Adminjogosultság: adminjogosultságot adhat a regisztrált felhasználónak, például egy új kolléga esetén a kollégát regisztráljuk és az adminunk ad neki jogosultságot, így tudja használni a fontos funkciókat a cég részéről.

Felhasználók kezelése: a felhasználók kezelése elsősorban a törlésre vonatkozik, de a későbbiekben lehetőség szerint kérésre módosthatja a felhasználó adatait. (FONTOS! CSAK FELHASZNÁLÓI KÉRELEM ESETÉN MÓDOSTUNK ADATOT).

# Fejlesztői dokumentáció

## Fejlesztői környezet

**Laravel 10 verzió**

A Laravel egy nyílt forráskódú PHP alapú keretrendszer, amelyet webes alkalmazások készítésére használunk. Az általunk választott 10. verzió a legfrissebb funkciókat és fejlesztéseket kínálja, lehetővé téve a gyors és hatékony alkalmazásfejlesztést.

**React 18.3.1 verzió**

A React egy népszerű JavaScript könyvtár, amelyet felhasználói felületek készítésére használunk. Az általunk választott 18.3.1 verzió a legújabb funkciókat és fejlesztéseket kínálja, lehetővé téve a modern, dinamikus és skálázható felhasználói élmények létrehozását.

**MySQL adatbázis**

Az adatbázis-kezelő rendszerünk a MySQL, amely egy ingyenes és nyílt forráskódú relációs adatbázis-kezelő rendszer. A MySQL legfrissebb verzióját alkalmazzuk, hogy biztosítsuk az adatbázisrendszer stabilitását és biztonságát. Segítségével hatékonyan tárolhatunk és kezelhetünk nagy mennyiségű adatot a projektünk számára.

## Program struktúra

MVC

Az MVC egy architekturális tervezési minta, melyet a program strukturált és jól szervezett felépítésére alkalmazunk. A Modell réteg az adatok kezeléséért és az üzleti logika végrehajtásáért felelős, a Nézet réteg pedig a felhasználói felület megjelenítését és az interakció lehetőségét biztosítja. A Vezérlő réteg pedig összeköti a Modellt és a Nézetet, kezelve a felhasználói interakciókat.

Az MVC struktúra segítségével könnyen megkülönböztethetjük az egyes részegységeket, és lehetőségünk van azokat elkülönítve fejleszteni és tesztelni. Ez átláthatóbbá teszi az alkalmazás kódját, hozzájárulva annak tisztaságához és könnyebb karbantarthatóságához. Emellett segít az alkalmazás kódjának hatékonyabb fejlesztésében és skálázhatóságában.

## Adatbázis

### Egyed-kapcsolati diagram

A képen szöveg, sor, diagram, képernyőkép látható

Automatikusan generált leírás

### Az adatrendszer ismertetése

A tárgyalt rendszerben igyekszünk kényelmes és átlátható termékrendszert biztosítani mind a vásárlónak, mind az adminoknak.

* A *termék* nyilvántartja a növényt és annak tulajdonságait, árral és készlet mennyiséggel eltárolva. A *termék*ben számontartjuk, hogy hány darabot foglaltak már le a készletből, így az aktuális készletből kivonva ellenőrizhetjük, hogy van-e elég készletünk a teljesítéshez.
* A *növény*ben megtalálhatóak a fajták latinnévvel azonosítva és növénykategóriával számon tartva.
* *A növénykategóriá*ban tároljuk a növény fajtáját.
* A *kiszerelés* tartja számon a növény méretét.
* A *termék ár* követi a termék árának változását dátummal, rögzíti adott dátumtól az új árat.
* A *felhasználó* tárolja a regisztrált felhasználóinkat, admint és vásárlót egyaránt, és a hozzáférésük szintjét.
* A *beszerzés* tábla kezeli a beszerzés dátumát darabszámmal és beszerzési árral.
* *Eladása tétel* tábla számontartja az adott nyugtához tartozó terméket és a rendelt mennyiséget az adott tételből.
* Az *eladás* a kiszámolt végösszeg dátummal ellátva szerepel, a végösszeget az *eladástételen* szereplő termék kódhoz tartozó darab és a *termék*ben szereplő azonos termékkódhoz tartozó ár alapján számoljuk ki.

### Az ügyviteli funkciók

Termékbeszerzés

* napibeszerzések rögzítése, a pillanatnyi készlet frissítésével

Növénydefiniálás

* növények számontartása
* növényfajta meghatározása

Termékdefiniálás

* termék karbantartás
* készlet és lefoglalt(kosárba helyezett) db nyomonkövetése

Termékár

* termék árának nyomon követhető változtatása

Nyugtázás/ Eladás

* a kosár alapján létrejött vásárlás nyugtázása
* egy nyugtán szereplő összes tétel darabszám alapján kiszámolt árának összesítése

Felhasználók

* felhasználók karbantartása
* jogosultság megítélése

### Az adatbázis-tervezés **a szüks**éges egyedek kialakításával

(jelölés: TÁBLA (kulcs, leíró, *kapcsolathordozó*)

**FELHASZNÁLÓ: a felhasználók eltárolása**

FELHASZNÁLÓ(felhasználó\_azon, név, email\_cím, jelszó, felhasználó\_szint)

A felhasználólk rendelkezhetnek jogosultsági szinttel, ezt a szuperadmin tudja csak kezelni. Bárkinek adhat admin jogosultságot. Adminok kezelik a beszerzést, a termékek felvitelét.

**NÖVÉNY\_KATEGÓRIA: növények hovatartozásának meghatározása elnevezéssel**

NÖVÉNY\_KATEGÓRIA(növény\_kategória, elnevezés)

Ezek rövid elnevezések lesznek, mint pl.: Egynyáriak, Rózsák ,Bogyós gyümölcsök…

**NÖVÉNY: latinnévvel, névvel és növény kategóriájának besorolásával számontartva**

NÖVÉNY(tudományos\_név, név, növény\_kategória)

**TERMÉK: az eladható termékek törzse az eladási árral és a már lefoglalt mennyiséggel**

TERMÉK (termék\_kód, növény, állapot, tipus, szín, kiszerelés, ár, készlet, lefoglalt\_mennyiség, szezonális)

Az állapot igen-nem típussal dönti el, hogy az adott növény milyen állapotban van, az igen az élőnövényt jelöli, ha nem akkor a magállapotot. Hasonlóan a típus is így dönti el, hogy dísz vagy haszon növényről van-e szó. Ha igen akkor dísznövény, ha nem akkor haszonnövény. Az alapállapot: mag és haszonnövény.

A készlet a termék éppen aktuális mennyiségét mutatja meg. Ha vásárlás történik, akkor elvesszük a mennyiséget, és ha beszerzés akkor automatikusan hozzáadódik az új árú.

A lefoglat mennyiség akkor kerül használatba, ha egy vásárló kosárba tesz bizonyos mennyiséget egy adott termékből. Itt lefoflalja a megadott mennyiséget, hogy mindenképpen meg lehessen vásárolni, ez úgy történik hogy levonja az aktuális készletből a megadott mennyiséget.

A szezonális egy igen nem típusú mezővel dönti el, hogy az adott termék szezonális e.

**KISZERELÉS: termékek kiszerelésének törzse, rövid megnevezéssel**

KISZERELÉS(kiszerelés, név)

**TERMÉK\_ÁR: termékek tervezett árai sávosan tárolva**

TERMÉK\_ÁR(termék, mikortól, új\_ár)

**BESZERZÉS: termékek napi beszerzése**

BESZERZÉS(termék\_kód, beszerzés\_dátuma, darabszám, beszerzési\_ár)

**ELADÁS: Az eladás törzsadatai, vásárlási dátummal és a kiszámolt fizetendő végösszeggel**

ELADÁS(eladás\_szám, vásárlás\_dátuma, végösszeg)

**ELADÁS\_TÉTEL: az eladás\_számhoz tartozó tételek darabszámmal**

ELADÁS\_TÉTEL(eladás\_szám, termék\_kód, darabszám)

**NÖVENY\_LEÍRAS: a nyövényekhez tartozó leírásokat tartalmazza**

NÖVÉNY\_LEÍRAS (leírás\_id, növény\_kód, leírás)

**KÉPEK: a termékekhez tartozó fényképek elérési útvonalát tárolja, a cél dönti el egy konkrét kép felhasználási célját.**

A cél mező lehet például B ami borítóképre utal.

KÉPEK (termék\_kód, elérési\_útvonal,cél)

### Táblák

K: kötelező

G: generált

**Felhasznalo**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| felhasznalo\_azon | int | elsődleges kulcs G | 10000 <= |
| nev | varchar(30) | leíró, K | - |
| email | nvarchar(50) | leíró, K | - |
| jelszo | varchar(30) | K | - |
| felhasz\_szint | int | leíró, K | default(2), 0,1, 2 |

**Noveny Kategoria**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| id | int | elsődleges kulcs G | 10<= |
| elnevezes | nvarchar(30) | leíró, K | - |

**Noveny**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| tudomanyos\_nev | nvarchar(30) | elsődleges kulcs | - |
| nev | nvarchar(30) | leíró, K | - |
| noveny\_kategoria | int | külső kulcs, K | - |

**Termek**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| termek\_kod | int | elsődleges kulcs G | 1000 <= |
| noveny | nvarchar(30) | külső kulcs | - |
| allapot | bit | leíró, K | -- |
| tipus | bit | leíró, K |  |
| szin | nvarchar(30) | leíró, | - |
| kiszereles | int | külső kulcs | - |
| ar | int | leíró, K | 1<= ar |
| keszlet | int | leíró, K | 0 <= keszlet |
| lefoglalt\_mennyiseg | int | leíró, K | 0<= lefoglalt\_mennyiseg |
| szezonlis | bit | leíró, K | - |

**Kiszereles**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| id | int | elsődleges kulcs G | 200< = |
| nev | nvarchar(30) | leíró, K | - |

**Termek Ar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| termek | int | külső kulcs | - |
| mikortol | date | leíró, K | - |
| uj\_ar | int | leíró, K | 1<= uj\_ar |

**Beszerzes**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| termek | int | külső kulcs | - |
| besz\_datum | date | leíró, K | - |
| darabszam | int | leíró, K | 1<= darabszam  default(1) |
| besz\_ar | int | leíró, K | 1<= besz\_ar |

**Eladas**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| eladas\_szam | int | elsődleges kulcs | 100000 <= |
| vasarlo | int | külső kulcs |  |
| vas\_datum | date | leíró, K |  |
| vegosszeg | int | leíró, K | 0 < vegosszeg |

**Eladas Tetel**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| eladas\_szam | int | külső kulcs | - |
| termek | int | külső kulcs | - |
| darabszam | int | leíró, K | 1 <= darabszam |

**Noveny Leiras**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| leiras\_id | int | elsődleges kulcs G | 1<= |
| noveny | nvarchar(30) | külső kulcs | - |
| leiras | nvarchar(MAX) | leíró, K | - |

**Kepek**

|  |  |  |  |
| --- | --- | --- | --- |
| **Oszlopnév** | **Típus** | **Szerep** | **Korlátozás** |
| termek | int | külső kulcs | - |
| eleresi\_utvonal | nvarchar(60) | leíró, K | - |
| cel | nvarchar(2) | leíró, K | - |

### Kapcsolati ábra

A képen szöveg, diagram, képernyőkép, Tervrajz látható

Automatikusan generált leírás

### Hasznos Lekérdezések

1. Adott felhasználó mely termékeket vásárolta meg eddig. (Mikor mit mennyit mennyiért)
   1. ezt a bejelentkezett felhasználó és az admin láthatja  
      (lehet nézet is, és akkor egyszerűbb megoldani, és majd le lehet rendezni frontenden)
2. Egy adott termékből mely színek voltak a legkelendőek az elmúlt évben?
3. Egy adott termékből mely kiszerelések voltak a legkelendőbbek az elmúlt évben?
4. Adott termék mely hónapban volt a legkelendőbb az elmúlt évben
5. A top 3 legkelendőbb termék az elmúlt hónapban.
6. A legkelendőbb termék a mai nap folyamán.
7. A hét legjobban fogyó terméke.
8. Mely termékből nem fogyott az elmúlt hónapban.
9. Mely termékekből fogyott kevesebb mint az előző hónapban a legutolsó árváltozás óta?
10. Adott regisztrált felhasználónak mely növénykategóriák reklámozhatók az eddigi rendelései alapján?
    1. adjon vissza konkrét terméket(ket).
11. Adott regisztrált felhasználó mely növénykategóriákból nem rendelt még soha?
12. Aktuális átlagár kiszámítása a beszerzési ár alapján, termékekre lebontva, tartalmazza a beszerzési árat, a jelenlegi eladási árat
13. A(z) \_\_\_ évi kiadások összege hónapra lebontva.
14. Az idei kiadások összege havi bontásban
15. Átlagosan hány százalékkal változtak meg a termékárak az előző évhez képest?
16. Mely termékekhez mely képek tartoznak?

### Kiemelt hasznos lekérdezések

**5. A top 3 legkelendőbb termék az elmúlt hónapban.**

Select TOP 3 with ties et.termek, SUM(et.darabszam)

from eladas\_tetel et

inner join eladas e on e.eladas\_szam=et.eladas\_szam

where MONTH(e.vas\_datum)=MONTH(GETDATE())-1

group by et.termek

order by 2 desc

**8. Mely termékből nem fogyott az elmúlt hónapban.**

go

create view multhonapi\_eladasok

as

select termek, vas\_datum

from eladas\_tetel et

inner join eladas e on e.eladas\_szam=et.eladas\_szam

where MONTH(e.vas\_datum)=MONTH(GETDATE())-1

go

Select t.termek\_kod

from termek t

left outer join multhonapi\_eladasok m on t.termek\_kod=m.termek

where m.vas\_datum is NULL

**13. A(z) \_\_\_ évi kiadások összege hónapra lebontva.**

go

CREATE function melyik\_evi\_kiadas

(

@ev int

)

returns table

as

return

(

select MONTH(besz\_datum) as hónap, SUM(besz\_ar \* darabszam)

from beszerzes

where year(besz\_datum)=@ev

group by MONTH(besz\_datum)

)

go

**14. Az idei kiadások összege havi bontásban**

Select MONTH(besz\_datum) as hónap, SUM(besz\_ar \* darabszam)

from beszerzes

where year(besz\_datum)=year(getdate())

group by MONTH(besz\_datum)

### Beépítendő megszorítások

1. Egyediség (kulcsok)
2. Hivatkozások (külső kulcsok)
3. Árak, készlet, új ár és a lefoglalt mennyiség nem negatív egész számok
4. Végösszegek a nyugtákon számított mezők
5. Egy termék aktuális árának megváltozásakor a régi ár elmentésre kerül
6. Ha egy termék bekerült a kosárba, akkor azt a termék táblába le kell vonni a készletből, és el kell tárolni a lefoglal mennyiségben a megvásárlás pillanatáig.
7. Ha egy terméket megvásárolnak, akkor az le legyen vonva a készletből.
8. Kosárba tételkor ellenőrizni kell, hogy van-e a megvásárolni készült mennyiség raktáron.
9. Termék ne kerülhessen ki az oldalra készlet nélkül. (vagy jelezze hogy nincs készlet)
10. Termékek nem törölhetők, ha van termék a kosárban.
11. Egy előre meghatározott esemény (vásárlás gomb) történésére létrehozza az eladás számot az aktuális dátummal.

### Technikai táblák (nem kötelezően megvalósítandó)

* kosár tábla és kosár tétel tábla, hogy az aktuális éppen zajló vásárlást számon tudja tartani a vásárló, de egy idő elteltével(session) a kosarat és a kosárban szereplő tételeket ürítjük.
* mentés, archiválás módja (a táblák sorainak szabályozott törlése)
* felkészülés a többnyelvűségre (az alapértelmezett nyelvű felh. funkcióknak és megjelenítendő adatoknak a más nyelvű megfelelői)

## Backend

A backend elkészítéséhez Laravel keretrendszert használtunk, ami PHP alapú webes keretrendszer. Ez az MVC (Model-View-Controller) tervezési mintát használja, aminek célja a szerepkörök szétválasztása.

### Modellek és Controllerek

#### Model

Feladata az adatok kezelése, adatstruktúrák meghatározása, létrehozása. Itt hajtjuk végre az adatokon elvégzendő műveleteket, mint a tárolásuk, és visszaolvasásuk . A model kommunikál az adatbázissal.

**Saját Modeljeink**

|  |  |
| --- | --- |
| Model | Feladata/Leírás |
| User | A felhasználók adatait tárolja, a permission mező határozza meg az adott személy szerepkörét: Felhasználó, Admin, SuperAdmin. |
| PlantCategory | A növényekhez tartozó kategóriákat tárolja el, mint például: rózsafélék. |
| Plant | A növényeket tárolja, a latin nevük által vannak megkülönböztetve. |
| Unit | Az egyes termékek lehetséges kiszereléseit tárolja el. Mint például: cserepes. |
| Product | A konkrét terméket tárolja el. |
| Purchase | A vásárlás adatait tárolja és kezeli, meghatározza mely Felhasználó vásárolt, és mikor. |
| PurchaseItem | Egy konkrét Terméket köt össze egy Vásárlással, tárolja az adott Termék darabszámát. |
| Suppliance | A Termékek beszerzését kezeli. |
| ProductPrice | A Termékek árváltozásait kezeli. |
| Description | A Termékekhez biztosít leírást. |
| Picture | A Termékekhez tartozó fényképek elérési útvonalát tárolja. |
|  |  |

#### Controller

A Controller feladata a Model és a View közötti kapcsolat biztosítása. Egy API hívása során megadjuk, hogy mely Controller mely metódusa fusson le.



**Saját Controllereink**

* ProductController

A ProductController egy olyan osztály, amely az alkalmazásban a termékekkel kapcsolatos műveletek végrehajtásáért felelős.

* + index(): Ez a függvény visszaadja az összes terméket egy listában.
  + show($id): Ezzel a függvénnyel egy adott termék adatait lehet lekérni az azonosítója alapján.
  + update(Request $request, $id): A függvény frissíti a megadott azonosítójú termék adatait a kapott kérés alapján.
  + store(Request $request): Ez a függvény új terméket hoz létre a kapott adatok alapján és elmenti az adatbázisba.
* UserController

A UserController felelős az alkalmazás felhasználókkal kapcsolatos műveletek végrehajtásáért. A következő funkciók lehetővé teszik a felhasználók kezelését az alkalmazásban.

* index(): Visszaadja az összes felhasználót egy listában.
* show($id): Ezzel a függvénnyel egy adott felhasználó adatait lehet lekérni az azonosítója alapján.
* update(Request $request, $id): Frissíti a megadott azonosítójú felhasználó adatait a kapott kérés alapján.
* store(Request $request): Új felhasználót hoz létre a kapott adatok alapján és elmenti az adatbázisba.
* PlantController

A PlantController egy olyan osztály, amely az alkalmazásban a növényekkel kapcsolatos műveletek végrehajtásáért felelős.

* + index(): Ez a függvény visszaadja az összes növényt egy listában. Az adminisztrációs felületen a növények listázásához használjuk.
  + show($id): Ezzel a függvénnyel egy adott növény részletes adatait lehet lekérni az azonosítója alapján. Például egy specifikus növény adatait lehet megjeleníteni a felhasználói felületen.
  + update(Request $request, $id): Frissíti a megadott azonosítójú növény adatait a kapott kérés alapján. Ez lehetővé teszi például a növény nevének, tudományos nevének vagy kategóriájának frissítését.
  + store(Request $request): Új növényt hoz létre a kapott adatok alapján és elmenti az adatbázisba. Egy űrlap kitöltésekor új növényt ad hozzá az adminfelületen.
* PurchaseController

A PurchaseController egy olyan osztály, amely az alkalmazásban a vásárlásokkal kapcsolatos műveletek végrehajtásáért felelős.

* index(): Visszaadja az összes vásárlást egy listában. Ez hasznos lehet például az adminisztrációs felületen a vásárlások listázásához.
* show($id): Ezzel a függvénnyel egy adott vásárlás részletes adatait lehet lekérni az azonosítója alapján. Például a vásárlás részleteinek megjelenítésére a felhasználói felületen.
* update(Request $request, $id): Frissíti a megadott azonosítójú vásárlás adatait a kapott kérés alapján. Ez lehetővé teszi például a vásárló nevének vagy a vásárlás dátumának frissítését.
* store(Request $request): Új vásárlást hoz létre a kapott adatok alapján és elmenti az adatbázisba. Hasznos lehet például egy űrlap kitöltésekor az új vásárlás hozzáadásához.

Ezek a funkciók lehetővé teszik a vásárlások kezelését az alkalmazásban.

* PurchaseItemController

A PurchaseItemController felelős a vásárlási tételekkel kapcsolatos műveletek végrehajtásáért.

* index(): Visszaadja az összes vásárlási tételt egy listában. Ez lehetővé teszi a vásárlási tételek listázását.
* show($purchase\_number, $product\_id): Ezzel a függvénnyel egy adott vásárlási tétel részletes adatait lehet lekérni a vásárlás száma és a termék azonosítója alapján.
* destroy($purchase\_number, $product\_id): Törli a megadott vásárlási tételt az adatbázisból, és visszaállítja a kapcsolódó termék raktárkészletét.
* update(Request $request, $purchase\_number, $product\_id): Frissíti a megadott vásárlási tétel adatait a kapott kérés alapján, és frissíti a kapcsolódó termék raktárkészletét.
* store(Request $request): Új vásárlási tételt hoz létre a kapott adatok alapján és elmenti az adatbázisba. Ellenőrzi, hogy van-e elegendő készlet a termékből, és frissíti a készletet a megfelelő mennyiséggel.
* SupplianceController

A SupplianceController az alkalmazásban a beszerzésekkel kapcsolatos műveletek végrehajtásáért felelős.

* index(): Visszaadja az összes beszerzést egy listában. Hasznos lehet például az adminisztrációs felületen a beszerzések listázásához.
* show($product, $suppliance\_date): Ezzel a függvénnyel egy adott beszerzés részletes adatait lehet lekérni a termék és a beszerzés dátumának alapján.
* destroy($product, $suppliance\_date): Törli a megadott termékhez és beszerzés dátumához tartozó beszerzést az adatbázisból.
* update(Request $request, $product, $suppliance\_date): Frissíti a megadott termékhez és beszerzés dátumához tartozó beszerzés adatait a kapott kérés alapján.
* store(Request $request): Új beszerzést hoz létre a kapott adatok alapján és elmenti az adatbázisba.
* ProductPriceController

A ProductPriceController felelős az alkalmazásban a termékek árváltozásainak kezeléséért.

* index(): Visszaadja az összes termékár-változást egy listában. Hasznos lehet például az adminisztrációs felületen a termékárak listázásához.
* show($id): Ezzel a függvénnyel egy adott termékár-változás részletes adatait lehet lekérni az azonosítója alapján.
* destroy($id): Törli a megadott termékár-változást az adatbázisból.
* update(Request $request, $id): Frissíti a megadott termékár-változás adatait a kapott kérés alapján.
* store(Request $request): Új termékár-változást hoz létre a kapott adatok alapján és elmenti az adatbázisba.
* UnitController

A UnitController felelős a növények kiszereléseivel kapcsolatos műveletek kezeléséért az alkalmazásban.

* index(): Visszaadja az összes kiszerelést egy listában. Hasznos lehet például a kiszerelések listázásához az adminisztrációs felületen.
* show($id): Ezzel a függvénnyel egy adott kiszerelés részletes adatait lehet lekérni az azonosítója alapján.
* update(Request $request, $id): Frissíti a megadott kiszerelés adatait a kapott kérés alapján.
* store(Request $request): Új kiszerelést hoz létre a kapott adatok alapján és elmenti az adatbázisba.
* PlantCategoryController

A PlantCategoryController felelős az növénykategóriák kezeléséért az alkalmazásban. Nézzük meg részletesebben a funkciókat:

* index(): Visszaadja az összes növénykategóriát egy listában. Ez hasznos lehet például a kategóriák listázásához a felhasználói felületen.
* show($id): Visszaadja az adott növénykategória részletes adatait az azonosítója alapján.
* update(Request $request, $id): Frissíti az adott növénykategória adatait a kapott kérés alapján.
* store(Request $request): Új növénykategóriát hoz létre a kapott adatok alapján, majd elmenti az adatbázisba.

Ezen funkciók lehetővé teszik a növénykategóriák kezelését az alkalmazásban, így könnyen csoportosíthatod és kezelheted az egyes növényeket a megfelelő kategóriákba.

**Laravel Api Breeze Controllerei**

* RegisteredUserController
* AuthenticatedSessionController
* ConfirmablePasswordController
* PasswordController

### API végpontok

Az **Application Programming Interface** kifejezés röviden **API** egy olyan alkalmazásprogramozási interfész, vagy felület, amely lehetővé teszi más szoftverek számára, hogy adatokat kérdezzenek le módosítsanak vagy töröljenek, illetve funkciókat hívjanak elő.

**GET:** Lekéri az erőforrás reprezentációját a megadott URI-n keresztül. A válaszüzenet törzse tartalmazza a kért erőforrás részleteit.

**POST:** Egy új erőforrást hoz létre a megadott URI-n. A kérésüzenet törzse tartalmazza az új erőforrás részleteit.

**PATCH:** Egy erőforrás részleges frissítését hajtja végre. A kérés törzse megadja az erőforrásra alkalmazni kívánt módosításokat.

**DELETE:** Eltávolítja az erőforrást a megadott URI-n

**PUT:** A megadott URI-n létrehoz egy új erőforrást, vagy cseréli a meglévőt. A kérésüzenet törzse meghatározza a létrehozni vagy frissíteni kívánt erőforrást.

**Útvonalak a Product-hoz**

|  |  |  |
| --- | --- | --- |
| **Kérés típusa** | **URI** | **Az útvonalhoz tartozó függvény feladata** |
| GET | api/product\_w\_pictures | Az összes Terméket lekéri a borítóképekkel. Rendesen kiírja a kiszerelést, a növény nevét, és leírást is tartalmaz. |
| GET | api/konkret\_product/{product\_id} | Egy konkrét Termék releváns adatait adja vissza képek nélkül. |
| GET | api/k\_product\_pictures/{product\_id} | Egy konkrét Termék összes képét adja vissza. |
| GET | api/products | Az összes terméket adja vissza a Termék törzsadataival. |
| GET | api/products/{product\_id} | Egy konkrét Terméket ad vissza a törzsadatokkal. |
| GET | api/boritokep | Az összes Termék borítóképként megjelölt fényképét adja vissza. |
| GET | api/description/{product\_id} | Egy konkrét Termékhez tartozó leírást adja vissza. |
| POST | api/products | Létrehoz egy új Terméket. |
| PATCH | api/products/{product\_id} | Egy konkrét létező Termék frissítését teszi lehetővé. |
|  |  |  |

**Útvonalak a User-hez**

|  |  |  |
| --- | --- | --- |
| **Kérés típusa** | **URI** | **Az útvonalhoz tartozó függvény feladata** |
| GET | api/users | Az összes létező User-t vissza adja minden adatával eggyüt. |
| GET | api/users/{$user\_id} | Egy konkrét User-t ad vissza minden adtával. |

**Útvonalak a Purchase-hez**

|  |  |  |
| --- | --- | --- |
| **Kérés típusa** | **URI** | **Az útvonalhoz tartozó függvény feladata** |
| GET | api/purchases | Lekéri az Összes Vásárlást. |
| GET | api/ purchases /{ purchase\_id } | Lekér egy specifikus Vásárlást. |
| POST | api/ purchases | Új vásárlást hoz létre. |
| PATCH | api/ purchases/{purchase\_id} | Egy meglévő vásár |
|  |  |  |

**Útvonalak a PurchaseItem-hez**

|  |  |  |
| --- | --- | --- |
| **Kérés típusa** | **URI** | **Az útvonalhoz tartozó függvény feladata** |
| GET | api/purchase\_items | Visszaadja az összes létező Vásárlási tételt. Tartalmazza mely terméket és mennyit vásároltak. |
| GET | api/purchase\_items/ {purchase\_number}/{product\_id} | Visszaad egy specifikus Vásárlási tételt |
| GET | api/rendelesek/{user\_id} | Egy Adtott Felhasználó összes rendelését adja vissza. |
| DELETE | api/purchase\_items/ | Egy létező Vásárlási tételt töröl. |
| POST | api/purchase\_items | Létrehoz egy új Vásárlási tételt. |
| PATCH | api/purchase\_items/ {purchase\_number}/{product\_id} | Egy meglévő Vásárlási tételt módosít. |

**Útvonalak a Suppliance-hez**

|  |  |  |
| --- | --- | --- |
| **Kérés típusa** | **URI** | **Az útvonalhoz tartozó függvény feladata** |
| GET | api/ suppliances | Megjeleníti az összes Beszerzést. |
| GET | api/ suppliances/{product}/{suppliance\_date} | Megjelenít egy specifikus Beszerzést. |
| DELETE | api/ suppliances/{product}/{suppliance\_date} | Egy létező Beszerzést töről. |
| POST | api/ suppliances | Új Beszerzést hoz létre. |
| PATCH | api/ suppliances{product}/{suppliance\_date} | Egy létező Beszerzést modosít. |

## Frontend

## Reszponzivitás

**Bootstrap**:

A Bootstrap egy népszerű, nyílt forráskódú front-end keretrendszer. Célja, hogy segítsen a webfejlesztőknek gyorsan és egyszerűen létrehozni reszponzív, mobilbarát weboldalakat. A Bootstrap előre megírt CSS és JavaScript komponenseket tartalmaz, mint például gombok, űrlapok, navigációs menük és táblázatok, amelyeket könnyen integrálhatunk a saját projektjeinkbe. Ezáltal minimalizálja a fejlesztési időt és növeli az alkalmazások egységességét. A Bootstrap a CSS keretrendszer, amely könnyen testre szabható és bővíthető saját stílusok és komponensek hozzáadásával.

**CSS (Cascading Style Sheets):**

A CSS egy stíluslap nyelv, amelyet weboldalak és webes alkalmazások megjelenésének formázására használnak. A CSS segítségével definiálhatjuk az elemek stílusát, mint például a betűtípust, a színeket, a méreteket, a távolságokat és a pozíciókat. Az elnevezés ("Cascading") azt jelenti, hogy a stílusok hierarchikusan öröklődnek az elemek között, így lehetővé téve az egységes és könnyen karbantartható stílusok létrehozását. A CSS alapvető fontosságú minden webfejlesztő számára, mivel lehetővé teszi a weboldalak megjelenésének testreszabását és az egyedi dizájn létrehozását.

# Tesztelés

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Az összes teszteset** | **Tesztelési egység** | | | |
|  | | | |
| Teszteset / Bemeneti paraméterek | Várt eredmény | Kapott eredmény | Státusz |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Fejlesztési lehetőségek

A következő fejlesztési lehetőségekre gondoltunk.

* A következőkben lehetne fájl beküldéssel feltölteni az egyes táblákat, például a növényeket vagy akár a kiszerelést.
* Kedvencek tábla és menüpont a felhasználónak. A felhasználó eltárolhatná kedvenc vagy megvásárolni kívánt termékeit későbbi vásárláshoz.
* Email küldés, hogy értesíthessük a felhasználókat.
* Készlet bővítés: egyéb termékek eladása növényeken kívül.
* Különleges eladás: az admin létrehozna egy eladást, ezek különleges eladások lennének: például saját részre eladni vagy áruházak közötti eladás zajlana.
* Keresések részletezése, hogy a felhasználó a lehető legpontosabban kereshessen egy termékre.
* Alkalmazottak kezelése: az alkalmazottakat külön kezelnénk az egyszerű felhasználóktól. Ezt csak az admin vihetné fel és nem kellene a kommunális oldalon regisztrálni a kollégának.
* Selejtezés: hibás vagy kifutott termékek elkülönítése, ezeket tudnánk kedvezményesen kinálni és eladni
* a TERMÉK-et mozgathatjuk a raktárból a bemutató térbe, vagy leselejtezzük
* Hűség jutalmazása a felhasználünak: Pont gyűjtés/ törzsvásárlói kártya, pontok felhasználhatók ahhoz, hogy a végösszegből X összeg lejönnne.

# Irodalomjegyzék

Hivatalos Dokumentációk:

<https://react.dev/learn>

<https://laravel.com/docs/10.x>

<https://getbootstrap.com/docs/5.0/getting-started/introduction/>

Egyéb

<https://kiszervezettmarketing.hu/weboldal-keszites/api-jelentese/>

<https://ade.web.elte.hu/wabp/lecke7_lap1.html>