**D O K U M E N T Á C I Ó**

**Vizsgaremek**

**SZOFTVERFEJLESZTŐ ÉS -TESZTELŐ TECHNIKUS SZAKMA**

Tartalomjegyzék

[1. Bevezetés, a téma ismertetése, témaválasztás indoklása 1](#_Toc101201241)

[2. Felhasználói dokumentáció 4](#_Toc101201242)

[2.1 Telepítési útmutatás Windows eszközre: 4](#_Toc101201243)

[2.2 A Web App telepítése és konfigurálása 6](#_Toc101201244)

[3. Fejlesztői dokumentáció 27](#_Toc101201245)

[3.1 A fejlesztési környezet, terminológiák bemutatása 27](#_Toc101201246)

# Bevezetés, a téma ismertetése, témaválasztás indoklása

A megvalósított projekt egy hotel honlapot és annak foglalási rendszerét, valamint termékeinek kezelését valósítja meg. A téma választás oka egy barátunk kérése, mivel jelen pillanatban épít egy hotelt-panziót a Balaton-felvidéken. Kérésének megfelelően a prioritásunk egy full-stack web létrehozása volt.

Kérésére láttunk neki a projektnek, mint referencia értékű feladatnak, mivel mindannyian tudjuk, hogy a mai világban rendkívül könnyű egy WordPress sablonnal egy SPA-t vagy bármilyen honlap sablont létre hozni és publikálni, viszont valós idejű foglalásokat az airbnb.com, a szallas.hu és a Booking.com legalább 10%-os költséggel kezel a teljes foglalás árából, még csak nem is a szállásadó profitjából.

Ezért manapság már a legtöbb önmagára adó hotel tulajdonos a saját honlapján is szeretné, ha a vendégek, felhasználók tudnák a foglalásaikat kezelni.

Ez nem azt jelent, hogy nem hirdetnek a legfőbb platformokon és nem érkezik ott foglalásaik nagy része, hanem azt, hogy saját platformokon is szeretnék kiadni szobáikat a közvetítő cégek költségei nélkül. Ezért kulcsfontosságú volt számunkra a foglalások kezelése adminisztrátor felhasználok számára, ezután a ’User’ -azaz vendég- felhasználók saját foglalásaik kezelése, majd mindezek mellett szerettük volna, ha a szálloda termékinek kezelés is megvalósul a web applikáción.

A termék kezelés is valósan live adatokkal történik a honlapon, valamint a termékek adatai elérhetőek 3rd party applikációk és API-k számára is.

Az adminisztrátor tud termékeket felvinni, módosítani, törölni és ahogy végre hatja bármelyik műveletet azt a bejelentkezett felhasználók is látják a honlapon valós időben.

Ez azért is font mert az összes modern üzlet árai gyorsan változnak a mai világban köszönhetően a hirtelen és magas inflációs nyomásnak, valamint a magyar Forint deviza elértéktelenedésének köszönhetően, hiszen napjainkban a magyar vállalkozások termékeik több mint 60-70%-át importálják, azaz euróban vagy dollárban fizetnek a beszerzés közben az áruért. Ha pedig Forint gyengül akkor a magyar cégeknek többet kell fizetni az árért dollárban ezért kénytelenek növelni az itthon eladott termékek árait.

Mindezek fényében fontos volt számunkra, hogy a hotel tulajdonos/ vezetőség / adminisztrátornak egy egyszerű mégis dinamikus és valós idejű lehetősége legyen a szálloda termékeinek kezelésére, főleg úgy, hogy későbbi terveink között szerepel a hotel foglalást is kibővíteni további funkciókkal, valamint a termék kezelést egy kisebb ERP (Enterprise Resource Planning - Vállalati Erőforrás Tervezés) rendszerré változtatni.

Fontos megemlítenem, hogy Vállalati Erőforrás Tervezés azaz ERP világában az SAP SE német cég piacvezető több évtizedes tapasztalattal, így nem kívánnánk ekkor konglomerátumokkal versenyezni, hanem a kisebb hoteleket, panziókat céloznánk meg, mint lehetséges jövőbeli megrendelők.

Fontos megemlítenünk, hogy az elkészült hotel projekt egyfajta sablonként is szolgál majd számunkra a jövőben, mivel terveink között szerepel fodrászatok, műkörmösök, autószerelők, orvosi rendelők foglalási rendszerinek is a kiszolgálása és az, hogy a hotelben napokat kezelünk szobánkként, míg a többi iparban napok és órákat kezelünk per kiszolgáló személyzet nem igazán nagy különbség egy fejlesztő számára.

Mindezeknek a fényében logikusan esett választásunk a Laravel-re, mivel az iskolai oktatás Back end fejlesztés tananyagában is hosszasan tanulmányoztuk a framework-ot.

A Laravel egy PHP-alapú és nyílt forráskódú háttér-keretrendszer, amelyet egyéni webalkalmazások széles skálájának létrehozására használnak. Ez egy teljes mértékben szerveroldali keretrendszer, amely a Model-View-Controller (MVC) tervezés segítségével kezeli az adatokat, amely logikai részekre bontja az alkalmazás-háttér-architektúrát.

Asztali alkalmazásunkban- mivel az üzlet felhasználók több mint 70%-a Windows operációs rendszert használ, a magán személyek pedig még magasabb százalékban- a .NET framework-re esett választásunk, a Visual Studio IDE által nyújtott Windows-ra célzott Microsoft termékre.

Az asztali alkalmazás jelenlegi funkciója, hogy begyűjti a termék adatokat Http kliens-el, a Laravel REST API által adott json-t feldolgozza és az API által adott adatokat megjeleníti egy táblában, majd egy gomb megnyomásával az adatokat .csv formátumban el is lehet menteni.

# Felhasználói dokumentáció

## 2.1 Telepítési útmutatás Windows eszközre:

**Xampp telepítése:**

1. Nyissa meg az Apache Friends webhelyet: <https://www.apachefriends.org/download.html>
2. Kattintson a Letöltés gombra az XAMPP Windows-verziójához, és mentse a fájlt a számítógépére.
3. Gyors megjegyzés: A projekt a PHP 8.0.12 verzióval lett készítve és tesztelve. Amennyiben másik verzió van telepítve a gépén, telepítse a PHP 8.0.12 vagy 8.0.2 verziót, de mindenképpen a PHP 8.0.12-es verzió az ajánlott.
4. A telepítő elindításához kattintson duplán a letöltött fájlra.
5. Kattintson az OK gombra.
6. Kattintson a Tovább gombra.
7. Az XAMPP különféle telepíthető összetevőket kínál, mint például a MySQL, phpMyAdmin, PHP, Apache stb. A legtöbb esetben ezeket az összetevőket fogja használni, ami azt jelenti, hogy ajánlott hagyni az alapértelmezett beállításokat.
8. Kattintson a Tovább gombra.
9. Használja az alapértelmezett telepítési helyet. (Vagy válasszon másik mappát a szoftver telepítéséhez a „Mappa kiválasztása” mezőben.)
10. Kattintson a Tovább gombra.
11. Válassza ki az XAMPP vezérlőpult nyelvét.
12. Kattintson a Tovább gombra.
13. Törölje a További információ a Bitnami for XAMPP-ról lehetőséget.
14. Kattintson a Tovább gombra.
15. Kattintson ismét a Tovább gombra.
16. Kattintson a Hozzáférés engedélyezése gombra, hogy engedélyezze az alkalmazást a Windows tűzfalon (ha van).
17. Kattintson a Befejezés gombra.
18. A lépések végrehajtása után elindul az XAMPP Vezérlőpult, és megkezdheti a webszerver környezet konfigurálását.

**Az XAMPP konfigurálása Windows 10 rendszeren**

1. Az XAMPP vezérlőpult három fő részből áll. A Modulok részben megtalálja az összes elérhető webszolgáltatást. Az egyes szolgáltatásokat a Start gombra kattintva indíthatja el.
2. Amikor elindít néhány szolgáltatást, beleértve az Apache-t és a MySQL-t, a jobb oldalon, az egyes szolgáltatások által használt folyamatazonosító (PID) és TCP/IP-port (Port) száma is megjelenik. Például alapértelmezés szerint az Apache a 80-as és a 443-as TCP/IP portot, míg a MySQL a 3306-os TCP/IP portot használja.
3. Az Adminisztrálás gombra kattintva elérheti az egyes szolgáltatások adminisztrációs irányítópultját, és ellenőrizheti, hogy minden megfelelően működik-e.
4. Az alapértelmezett beállításoknak működniük kell a legtöbb felhasználó számára, aki XAMPP-t használ a webhely futtatásához szükséges tesztelési környezet létrehozásához. A beállítási konfigurációtól függően azonban előfordulhat, hogy módosítania kell az Apache szerver TCP/IP portszámát, az adatbázis feltöltési méretét, vagy be kell állítania a phpMyAdmin jelszavát.
5. A beállítások módosításához a megfelelő szolgáltatás Config gombját kell használnia. Például meg kell nyitnia a httpd.conf fájlt az Apache-kiszolgáló beállításainak módosításához, a my.ini fájlt pedig a MySQL-beállítások módosításához.
6. Xampp Control panel indítása
7. Apache és MySQl rákattintani a Start gombra
8. MYSQL gombon az admin gombra kattintva eljutunk a <http://localhost/phpmyadmin/> oldalra

## 2.2 A Web App telepítése és konfigurálása

**phpMyAdmin konfigurálás:**

1. Új adatbázis létrehozása: “laravel” címen
2. Ez fontos, mivel az env állományban van meghatározva az adatbázis neve és a migrációs séma a projektben előre meg van határozva, azaz ebbe az adatbázisba fogjuk migrálni az adatbázis táblákat és azoknak a Laravel php-ban megadott relációs sémáját is.

**Composer telepítése:**

1. Composer telepítése exe fájl formátumban az alábbi webhelyről: <https://getcomposer.org/download/>
2. Letöltött exe fájl futtatása és telepítése a legújabb php verzóval
3. Parancssor megnyitása és a “composer” szó beírásával majd az “Enter” szó leütésével ellenőrizzük, hogy a composer sikeresen települt a Windows 10 környezetben

**Laravel telepítése:**

1. composer global require laravel/installer

**Projekt mappa készítése és Laravel telepítése:**

1. composer create-project --prefer-dist laravel/laravel **ProjektNév**
2. Ezután cd az az be kell lépni a laravel **ProjektNév** projekt mappába és ott Git bash vagy Új terminál VS Code programban
3. php artisan migrate

**Jetstream telepítése:**

1. composer require laravel/Jetstream
2. php artisan jetstream:install livewire
3. php artisan jetstream:install inertia
4. npm install
5. npm run dev
6. php artisan migrate
7. php artisan vendor:publish --tag=jetstream-views
8. npm run dev

**Laravel Sanctum telepítése:**

1. composer require laravel/sanctum
2. php artisan vendor:publish --provider="Laravel\Sanctum\SanctumServiceProvider"
3. php artisan migrate

**Laravel Fortify telepítése**

1. composer require laravel/fortify
2. php artisan vendor:publish --provider="Laravel\Fortify\FortifyServiceProvider"
3. php artisan migrate

**Doctrine Dbal telepítése**

1. composer require doctrine/dbal

**Clone Laravel projekt:**

1. A file explorerben job egér gomb klikkel vagy Git bash here vagy open with VS code-al meg kell nyitni egy mappát és oda klónozni a git clone parancsal a project mappát, avagy azt letölteni a Github URL-ről zip formátumban. A klónozáshoz új Terminál ablakot kell létesíteni a VS Code-ban vagy a projekt mappában job klikk és a GIT bash here parancsal lehet Terminált létesíteni.
2. **Egy Terminálban beírni a következő parancsokat:**
3. git clone 'url'
4. ha megvan a projekt a lokális mappában akkor be kell lépni az AuthFinal-HMS klónozott mappába és annak az egész tartalmát Ctrl+ A paranccsal kijelölni és Ctrl-C / Ctrl +V billentyűk segítségével át kell másolni a korábban composer-el létrehozott **ProjektNév** mappába, ahol már a teljes környezet kiépítésre került (composer create-project --prefer-dist laravel/laravel **ProjektNév**)

Table

Description automatically generated

1. Be kell lépni a laravel **ProjektNév** mappába, amit korábban megadott a Laravel project készítésénél
2. A **ProjektNév** mappát megnyitni VS Code alkalmazással
3. Itt **New Terminal** parancsal új terminált létesítünk
4. **composer install**
5. **npm install**
6. **npm run dev**
7. **Env file konfigurálása a lejjeb található env paraméterek bemásolása**

**Env file konfigurálása:**

**Fontos az alábbi változók bemásolása az env fájlba (kivéve az APP\_KEY változót, mivel azt a “php artisan key:generate” parancs generálja majd) mivel ezek a pamatéterek adják a környezet konfigurációját.**

APP\_NAME="HMS System"

APP\_ENV=local

APP\_KEY=base64:ET5PUEUNqm6u1V7I1eqgIsqTdkinieqkAhwTm6wLdKM=

APP\_DEBUG=true

APP\_URL=http://127.0.0.1:8000

ASSET\_URL=/public/assets

LOG\_CHANNEL=stack

LOG\_LEVEL=debug

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=laravel

DB\_USERNAME=root

DB\_PASSWORD=

BROADCAST\_DRIVER=log

CACHE\_DRIVER=file

QUEUE\_CONNECTION=sync

SESSION\_DRIVER=database

SESSION\_LIFETIME=120

MEMCACHED\_HOST=127.0.0.1

REDIS\_HOST=127.0.0.1

REDIS\_PASSWORD=null

REDIS\_PORT=6379

MAIL\_MAILER=smtp

MAIL\_HOST=smtp.mailtrap.io

MAIL\_PORT=2525

MAIL\_USERNAME=474d8b7be5ef44

MAIL\_PASSWORD=6b25bed91ad7a9

MAIL\_ENCRYPTION=tls

MAIL\_FROM\_ADDRESS= register@budapesthotels.com

MAIL\_FROM\_NAME="${APP\_NAME}"

AWS\_ACCESS\_KEY\_ID=

AWS\_SECRET\_ACCESS\_KEY=

AWS\_DEFAULT\_REGION=us-east-1

AWS\_BUCKET=

PUSHER\_APP\_ID=

PUSHER\_APP\_KEY=

PUSHER\_APP\_SECRET=

PUSHER\_APP\_CLUSTER=mt1

MIX\_PUSHER\_APP\_KEY="${PUSHER\_APP\_KEY}"

MIX\_PUSHER\_APP\_CLUSTER="${PUSHER\_APP\_CLUSTER}"

1. **php artisan key:generate**
2. **php artisan migrate (migrációs hiba esetén törölni kell a duplikált migrációs táblát (database/migrations útvonaon a VS doce-banl) és a** [**http://localhost/phpmyadmin**](http://localhost/phpmyadmin) **oldalon, például ebben az esetben**

**Graphical user interface, text, chat or text message

Description automatically generated**

**a ’sessions’ táblát, majd újra kell migrálni a ’php artisan migrate’ parancsal**

Text

Description automatically generated

**Duplikált session table migration esetén akár törölhetjük a korábbi verziót, persze előtte érdemes ellenőrizni, hogy tartalmilag megegyezik a két fájl.**

1. **php artisan migrate:fresh**
2. **npm run watch**
3. **Adatbázis feltöltése:**
   1. [**http://localhost/phpmyadmin/**](http://localhost/phpmyadmin/) **oldalra látogatni:**
   2. **Drop database laravel az adatbázis eldobása**
   3. **Újra csinálni egy laravel nevű adatbázist**
   4. **A Laravel.sql fájlt a http://localhost/phpmyadmin/ laravel adatbázisban az Import gombra kattintra, majd a ‘Choose file’ gombra kattintva be tudjuk illeszteni, majd a ‘Go’ gomb megnyomásával be is importáljuk az összes korábban bevitt, megadott és exportált adatot.**
      1. **Opciónálisan: A user táblát fel tudjuk tölteni random user adatokkal a AuthFinal-HMS\database\seeders\DatabaseSeeder.php elérési útvonalon található fájl alapján a faker segítségével, a “php artisan db:seed” parancsal**
4. **A Laravel fejlesztési szerver indítása: php artisan serve**
5. **Xampp Control Panel start Apach és MySQL ellenőrzése**
6. [**http://127.0.0.1:8000/**](http://127.0.0.1:8000/) **URL-t beírni a böngészőbe**

**Kezdőlap:**

**A screenshot of a video game

Description automatically generated**

A jobb felső sarokban található Login illetve Register gombokkal a /login:

Graphical user interface, application

Description automatically generated

Illetve a /register oldalra érkezik:

Graphical user interface, application

Description automatically generated

A /login oldalról a No account Yet? -re kattintva a /register oldalra tud eljutni a felhasználó, míg a Forgot your password? -re kattintás esetén a /forgot-password nézet jelenik meg:

Graphical user interface, text, application

Description automatically generated

Mind regisztráció mind jelszó elfelejtés esetén email-es jóváhagyásra van szükség.

**Email küldés:**

A web appon történt regisztráció után a regisztrációt jóvá kell hagyni email-en keresztül, egy felhasználó csak azután tudja használni a programot, hogyha a regisztrációt követően igazolta email címét, valamint jelszó visszaállítás, helyre állítás is emailben küldött linken keresztül működik.

Az email funkciók teszteléséhez és használatához a **MailTrap** szolgáltatását vettük igénybe.

**Mailtrap bemutatása**:

A Mailtrap hamis SMTP-kiszolgálót biztosít a fejlesztőcsapat számára, amellyel tesztelheti, megtekintheti és megoszthatja a gyártás előtti környezetekből küldött e-maileket, és valós adatokkal tesztelheti anélkül, hogy fennállna a valódi ügyfelek spamküldésének veszélye. A Railsware készítette, és számos fejlesztési feladathoz a Mailtrap használata ingyenes.

Lényegében regisztrál a Mailtrap szolgáltatásra, és minden e-mailt elküld a gyártás előtti környezetből a hamis Mailtrap SMTP-kiszolgálón keresztül.

A MailTrap segítségével e-maileket rögzíthet a tesztelési fejlesztői és állomásozó környezetekből

Ezután minden levele a Mailtraphez tartozik. Megtekintheti és hibakeresheti e-mailjeit a Mailtrap barátságos grafikus felületén.

A Mailtrap segítségével akár valódi felhasználói e-maileket tartalmazó kiíratásokat is elhelyezhet a termelési adatbázisában, tesztek segítségével az átmeneti kiszolgálón. Automatizált tesztjei ellentmondhatnak a valós adatoknak – ha e-mailt küldenek a Mailtrap-on keresztül, így kiküszöbölhető annak a kockázata, hogy a teszte-mailek valódi ügyfelek e-mail címére menjenek.

**URL:** [**https://mailtrap.io/**](https://mailtrap.io/)

**Ahhoz, hogy regisztrálni tudjunk a honlapon, a fenti URL-en be kell jelentkezni az alábbi paraméterekkel.**

**A projekthez tartozó felhasználónév és jelszó, ezekkel kell belépni a honlapon.**

Felhasználónév**: emailsenderz53@gmail.com**

Jelszó**: Email23$**

**Amennyiben Xampp Apache és MySQl még mindig aktív, valamint a Laravel development server is az akkor:** [**http://127.0.0.1:8000/**](http://127.0.0.1:8000/) **vagy a** [**http://127.0.0.1:8000/index**](http://127.0.0.1:8000/index) **oldalakon be tud jelentkezni, mint felhasználó a honlapon vagy regisztrálni is tud. Az alábbi fiókokat is lehet használni tesztelésre:**

**Adminisztrátor fiók:**

felhasználónév/ email**:** [**admin@gmail.com**](mailto:admin@gmail.com)

jelszó**: Admin111$**

**Vendég fiók:**

felhasználónév/ email**: user2@gmail.com**

jelszó**: User222$**

**Minden esetben új regisztráció esetén figyelni kell, hogy a jelszó megadásánál a jelszónak minimum 8 karakterből kell állnia, legyen benne betű, szám és speciális karakter és egy nagy betű is.**

Mindkét URL igazából egy welcome page, azaz ahol a felhasználók meg tudják ismerni a hotelt, tudnak képeket nézni és információt gyűjteni a hotelről ( a navigációs menü ’Gallery’ szövegre kattintva) , szobákról ( a navigációs menü ’Room’ szövegre kattintva), a szolgáltatásokról( a navigációs menü ’Services’ szövegre kattintva) , az ételről ( a navigációs menü ’Food’ szövegre kattintva), gyakran ismételt kérdésekről(GYIK) ( a navigációs menü ’FAQ’ szövegre kattintva), a felső vezetésről ( a navigációs menü ’Our Team’ szövegre kattintva), aktuális álláslehetőségről( a navigációs menü ’Career’ szövegre kattintva),szobát foglalni ( a navigációs menü ’Book Now’ szövegre kattintva).

A ’Book now’ szövegre kattintva vagy be kell jelentkeznie, vagy regisztrálnia kell emailes validálással a felhasználónak.

Admin felhasználó esetén a következő lehetőségek állnak rendelkezésre:

A navigációs sávban a Profile-ra kattintva a profil beállítások válnak elérhetővé: Graphical user interface, application

Description automatically generated

/user/profile a URL cím:

Itt akár Adminnak akár vendég, azaz sima User felhasználónak is lehetősége a profileban a nevét, email címét, lakcímét és telefonszámát változtatni.

Ezen az oldalon tud az összes felhasználó jelszavat változtatni.

Lejjebb Két kulcsos azonosítást tudnak a felhasználók bekapcsolni Google Authenticator Applikáció segítségével egy egyedi QR kódót tudnak beolvasni, valamint recovery kódokat elmenteni.

Az ezalatti konténerben pedig, ha esetleg több böngésző session lenne nyitva akkor a többi böngészőből a user ki tudja léptetni magát.

És végül, de nem utolsó sorban a felhasználok GDPR-nak megfelelően tudják a saját fiókjukat törölni is.

Graphical user interface, application

Description automatically generated

A home page a bejelentkezett felhasználóknak mindig /user/home URI a kezdőlap, ám ez másik nézetet ad vissza annak alapján, hogy a felhasználó jogosultsága admin vagy user.

Admin felhasználó esetén itt található három konténer.

Egy a foglalások kezelésére való átvitelért felelős (Bookings Admin Center konténer, azaz: /transactions útvonal), a második a termékek kezelésére való átvitelért felelős (Products Admin Center, azaz: /products útvonal), a harmadik pedig a felhasználók kezelésére való átvitelért felelős (Users Admin Center, azaz: /users útvonal).

Legfelül van egy admint üdvözlő üzenet, valamint két kék színű link, ahol az egyiken a program összes útvonala jelenik meg (Routes list), a másikra (Download Routes list in CSV) kattintva pedig egy .csv-ben letöltődik a webről a program által használt összes útvonal.

A navigációs menüben a Home page vissza visszavisz a logged-out home page-re, ebben az esetben a Dashboard ikon jelenik meg a logged out navigációs menü jobb felső sarkában a Login és Register helyett hiszen a felhasználó be van jelentkezve és ha a felhasználó a Dashboard-ra kattint akkor újra a logged in view-ban és navbaron lesz a felhasználó a /dashboard nézetben.

Graphical user interface, text, application, email

Description automatically generated

A dashboard nézetben van egy útvonal (Google Maps redirect link) választó gomb (Direction) és egy Book Now gomb, ahol a felhasználó a /my-bookings útvonalra jut el ahol csak magának tud foglalni szobát a belépett felhasználó a New gombra kattintva. A media konténer Our instagram az Instagramra visz át, míg a watch a video a YouTube-ra, mindkettő új oldalon jelenik meg.

Amennyiben a bejelentkezett felhasználónak van foglalása akkor a táblában ez meg is jelenik:

Graphical user interface, application

Description automatically generated

A user tudja módosítani, illetve törölni is saját foglalását az Edit illetve a Delete gombra kattintva.

Admin felhasználó a /transactions útvonalon tud foglalásokat módosítani az Edit gombbal, törölni a Delete gombbal és a New gombbal új foglalást rögzíteni. Delet gombra kattintás esetén van egy figyelmeztetés Graphical user interface, text, application

Description automatically generated

Amíg az admin itt nem kattint az OK gombra addig nem törlődik a foglalás.

Edit gombra kattintva az összes választható paraméter (Guest Name, Guest email, Guest phone, Select Room, Checkin, Checkout, Halfbooard, Number of guests) állíthatóvá válik az admin számára. kivéve a Days és a Bill field, mivel ezek auto generáltak a day a Check-in és Check-out napok különbége, míg a Bill ez a különbség megszorozva a napok számával:

Graphical user interface, text, application

Description automatically generated

Természetesen mind a /transactions, /products, /users middleware által vannak biztosítva és letiltva a vendégek számára, tehát csak jóváhagyott admin felhasználók tudják ezeket a szenzitív adatokat látni és állítani.

Az admin a /transactions oldalon a New gombra kattintva tud új foglalást rögzíteni.

Itt amennyiben még nincs ilyen nevű felhasználó, mint akit az admin felvisz akkor a foglalás rögzítésével együtt a user (azaz vendég) neve, email címe és egy random generált bcrypt titkosítással ellátott jelszó is rögzítésre kerül a user táblában a transaction táblában pedig tároljuk az admin által megadott user telefonszámot.

A Search field-ben, azaz kereső mezőben lehetséges keresni avagy szűrni a foglalások között, szoba név(Room name) vagy email vagy telefonszám alapján.

Az összes foglalást foglalási idő alapján dátum szerint növekvő sorrendben mutatjuk alapértelemezve.

Az oldalon pagination van érvényben, jelenleg 10 találatot mutat egy oldal.

Az admin a /products oldalon tudja a termékeket kezelni. Fel tud vinni új terméket a New gombbal, a terméket tudja kategorizálni, tud leírást adni és árat is, a Save gombbal elmenteni ezt. A Delete gombbal tud törölni terméket az admin, itt is megjelenik egy pop-up warning Graphical user interface, application

Description automatically generated

Amíg ezt le nem OK-za az admin addig nem törlődik a termék, Cancel gombra kattintva lehet visszalépni és nem törölni a terméket.

Az Edit gombra kattintva lehet a termék nevét, árát, leírását és kategóriáját változtatni (Name, Category, Description, Price).

Minden változtatás azonnal frissül az adatbázisban itt a web útvonalon megadva is, ahonnan a REST API is nyeri a termék adatokat.

Minden termék itt is pagination által jelenik meg 10 találatot mutatva egy oldalon, kategória név alapján növekvő sorrendben.

A Search field-ben, azaz kereső mezőben lehetséges keresni avagy szűrni a termékek között, név(name) vagy category(Kategória) vagy description(leírás) alapján.

Minden terméket, ami az admin oldalon megjelenik az elérhető a vendégek számára is egy hasonló tábla nézetben csak teljes CRUD operátorok nélkül, azaz szerkeszteni, törölni, avagy új terméket itt felvinni nem lehet, csak Read opció adott, azaz a GET metódus, hogy tudják a vendégek élőben látni a termékeket és keresni közöttük, mindezt a /restaurantandbar útvonalon a Menu & bar navigációs gombra kattintva.

Minden termék itt is pagination által jelenik meg, itt 15 találatot mutatva egy oldalon, kategória név alapján növekvő sorrendben.

A Search field-ben, azaz kereső mezőben lehetséges keresni avagy szűrni a termékek között, név(name) vagy category(Kategória) vagy description(leírás) alapján.

Minden, ami látható egy szépen rendezett táblában a /restaurantandbar útvonalon azt egy nyers json formátumban is meg tudjuk kapni a /jsonproducts útvonalon a weben.

Az asztali alkalmazás az úgynevezett GetProducts is ezeket aza adatok szedi le a webről egy Http kliens segitségvel majd az asztali alkalmazás is táblába rendezi ezeket a Get Product data-ra kattintva:

Graphical user interface, application, Word

Description automatically generated

Az asztali alkalmazást a Products.exe fájlra kétszer kattintva tudjuk telepíteni és indítani a NET\_Desktop-master mappából.

Az indító alkalmazás a NET\_Desktop-master\GetProducts\bin\Debug\ GetProducts.exe útvonalon érhető el. A letöltött zip fájl kibontása után.

Graphical user interface, table

Description automatically generated

A Get Product data az alábbi nézetet adja vissza:

Table

Description automatically generated

Miután megkapta az asztali alkalmazás az adatokat és táblába rendezi azokat a CSV Export gombra kattintva tetszőleges néven lehet menteni az exportált csv fájlt.

Kétszer kattintva a tetszőleges néven mentett .csv fájlra megnyílik a Microsoft Excel és csv nézetben kapjuk vissza az összes termék adatot, amit tetszés szerint lehet szerkeszteni az Excel-ben.

Text

Description automatically generated

Eddig csak a weben és az asztali alkalmazás kapcsán említettük a Termékeket (Products), ám ezeket az api útvonalon is elhelyeztük egy REST API-ban ami Postman testing tool-al tesztelhető, a Postman kollekció megtalálható a következő utvonalon a projekt mappában: api\_documentation & docs\ ProductControllerREST.postman\_collection.json

Itt ugyanazokat a műveleteket tudják a userek és adminok végrehajtani, mint a weben a /products útvonalon, annyi különbséggel, hogy a REST API tesztelése folyamán konfigurálni kell a Postman applikációt.

Legkönyebben ezt a Postman megnyitása után a Collections -> import gombra kattintva -> importálni kell a mappában található kollekciót.

Fontos megemlíteni, hogy a login után kapott token-t fontos másolni,

Graphical user interface, text, application, email

Description automatically generated

majd a következő http metódus pl. POST <http://localhost:8000/api/products>

Esetén az Authorization alatt a Type: Bearer Token-t kiválasztása után a Token field-be bemásolni.

A token( Sanctum) és / vagy Admin szerep által tiltott, avagy nem engedélyezett metódusok csak a token megadása után lesznek engedélyezve.

Az összes metódus ami POST, PUT, DELETE tehát a CRUD-on belül a READ-et kivéve mind szigorúan authorizációhoz vannak kötve

Graphical user interface, text, application, email

Description automatically generated

Visszatérve a web-re a /users útvonalon van lehetősége az adminoknak a felhasználó kezelésre.

Az admin a /users oldalon tudja a usereket kezelni. Fel tud vinni új user-t a New gombbal, a usereknek tud role-t adni (pl. Admin), tud nevet, emailt, címet és telefonszámot bevinni. A Save gombbal tudja elmenteni ezt. Ha az admin új usert visz fel akkor a bycrpt-el egy titok enkriptált jelszót is beinjektálunk az adatbázis user táblájába. Amennyiben egy user bajba kerülne és admin segítséget kérne mert jelszavát megszerezték, és a user nincs gép előtt, akkor az admin az Edit -> Reset Password gombra kattintva tud új bcrypt enkryptált jelszavat generálni.

Biztonsági okok miatt az adminok senkinek se tudják a jelszavát, és az adtbázisban is bcrypt technológia által enkriptált formátumban tároljuk a jelszavakat az adat biztonság nevében. A user táblában nincsen Delete gomb, az Admin nem tud törölni usereket, csak a userek tudják magukat, biztonsági okokból. Az admin az Edit gombra kattintva tudja a userek adatait változtatni (name, role, email, address, phone number). Mint az összes többi edit pop-up modal boksz-nál úgy itt is a Save gombbal lehet menteni, míg a Cancel gombbal visszalépni és meg nem történté tenni a változtatásokat.

Minden a /users oldalon is pagination által jelenik meg, itt 20 találatot mutatva egy oldalon, ID alapján növekvő sorrendben.

A Search field-ben, azaz kereső mezőben lehetséges keresni, avagy szűrni a userek között, név(name), cím(address), szerep(role), email, telefonszám (phone number) alapján.

# 3. Fejlesztői dokumentáció

## 3.1 A fejlesztési környezet, terminológiák bemutatása

A fejlesztés webes applikáció esetén az MVC modell alapján valósult meg a Laravel framework használatával.

**API dokumentációk elérése a weben:**

1. Swagger:
   1. <https://app.swaggerhub.com/apis-docs/wokres/product-controller_rest/1.0.0>
2. Postman
   1. <https://documenter.getpostman.com/view/19126249/UVyxRZv2#ea026aee-00a1-4287-9517-85a8d58b8977>

**ProductController OPENAPI 3.0 specifikáció:**

openapi: 3.0.0

info:

  title: ProductControllerREST

  version: 1.0.0

servers:

  - url: http://localhost:8000

components:

  securitySchemes:

    bearerAuth:

      type: http

      scheme: bearer

    noauthAuth:

      type: http

      scheme: noauth

paths:

  /api/products:

    post:

      tags:

        - default

      summary: http://localhost:8000/api/products

      requestBody:

        content:

          application/x-www-form-urlencoded:

            schema:

              properties:

                name:

                  type: string

                  example: Gingerbread

                category:

                  type: string

                  example: Food

                description:

                  type: string

                  example: Sweet bread, allergic

                price:

                  type: integer

                  example: '955'

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '201':

          description: Created

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:04:53 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '58'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                name: Ketchup

                category: Food

                description: Small portion of ketchup

                price: '248'

                updated\_at: '2022-04-08T20:04:53.000000Z'

                created\_at: '2022-04-08T20:04:53.000000Z'

                id: 31

        '401':

          description: Unauthorized

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:46:25 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Unauthenticated.

        '403':

          description: Forbidden

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Sat, 09 Apr 2022 18:40:36 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Forbidden

    get:

      tags:

        - default

      summary: http://localhost:8000/api/products

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:02:33 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                - id: 1

                  name: Pina Colada

                  category: Cocktail

                  description: Pineapple, rum, creamy cocktail

                  price: '2399.00'

                  created\_at: '2022-03-20T19:10:10.000000Z'

                  updated\_at: '2022-03-31T21:44:06.000000Z'

                - id: 2

                  name: Beef steak

                  category: Food

                  description: Argentinian beef

                  price: '9900.00'

                  created\_at: null

                  updated\_at: '2022-03-27T15:02:11.000000Z'

                - id: 3

                  name: Chicken steak

                  category: Food

                  description: Countryside chicken

                  price: '3000.00'

                  created\_at: null

                  updated\_at: null

                - id: 4

                  name: Beef burger

                  category: Food

                  description: 330g meat, salad, mayo, burger

                  price: '3999.99'

                  created\_at: null

                  updated\_at: '2022-03-27T14:46:37.000000Z'

                - id: 5

                  name: Long Island

                  category: Cocktail

                  description: Rum, Tequila, Vodka, Coke

                  price: '3999.00'

                  created\_at: '2022-03-21T02:03:35.000000Z'

                  updated\_at: '2022-03-23T08:02:03.000000Z'

                - id: 8

                  name: Mliano spaghetti

                  category: Food

                  description: Meatballs with pasta

                  price: '2699.00'

                  created\_at: '2022-03-23T07:59:36.000000Z'

                  updated\_at: '2022-03-23T07:59:36.000000Z'

                - id: 9

                  name: Chilli con carne

                  category: Food

                  description: Mexican spicy minced meat

                  price: '2699.00'

                  created\_at: '2022-03-23T08:00:06.000000Z'

                  updated\_at: '2022-03-23T08:00:06.000000Z'

                - id: 10

                  name: Tequila Sunrire

                  category: Cocktail

                  description: Lemony tequila cocktail

                  price: '2999.00'

                  created\_at: '2022-03-23T08:25:54.000000Z'

                  updated\_at: '2022-03-23T08:25:54.000000Z'

                - id: 11

                  name: Sparkling water

                  category: Beverage

                  description: Still water sparkling

                  price: '499.00'

                  created\_at: '2022-03-23T08:34:32.000000Z'

                  updated\_at: '2022-03-23T08:34:32.000000Z'

                - id: 13

                  name: Coca-Cola

                  category: Beverage

                  description: Cola 0.5l

                  price: '499.00'

                  created\_at: '2022-03-23T08:35:55.000000Z'

                  updated\_at: '2022-03-23T08:35:55.000000Z'

                - id: 14

                  name: Pepsi Coke

                  category: Beverage

                  description: Coke 0.5l

                  price: '499.00'

                  created\_at: '2022-03-23T08:36:09.000000Z'

                  updated\_at: '2022-03-23T08:36:09.000000Z'

                - id: 15

                  name: Fanta Lemon

                  category: Beverage

                  description: Lemon drink 0.5l

                  price: '499.00'

                  created\_at: '2022-03-23T08:36:36.000000Z'

                  updated\_at: '2022-03-23T08:36:36.000000Z'

                - id: 16

                  name: Coca-Cola can

                  category: Beverage

                  description: Coke drink 0.33l

                  price: '499.00'

                  created\_at: '2022-03-23T08:36:56.000000Z'

                  updated\_at: '2022-03-23T08:36:56.000000Z'

                - id: 18

                  name: Lemonade

                  category: Beverage

                  description: Lemon juice 0.33 glass

                  price: '499.00'

                  created\_at: '2022-03-23T08:37:46.000000Z'

                  updated\_at: '2022-03-23T08:37:46.000000Z'

                - id: 21

                  name: Töpörtyű

                  category: Food

                  description: Hungarian pork specialty

                  price: '1999.00'

                  created\_at: '2022-03-27T14:52:15.000000Z'

                  updated\_at: '2022-03-27T14:58:50.000000Z'

                - id: 22

                  name: Chicken wings

                  category: Food

                  description: Spicy grilled wings

                  price: '2350.00'

                  created\_at: '2022-03-27T14:53:50.000000Z'

                  updated\_at: '2022-03-27T14:53:50.000000Z'

                - id: 23

                  name: Ham and eggs

                  category: Food

                  description: Grilled ham and eggs

                  price: '2290.00'

                  created\_at: '2022-03-27T14:56:18.000000Z'

                  updated\_at: '2022-03-27T15:00:16.000000Z'

                - id: 24

                  name: Vodka Soda

                  category: Cocktail

                  description: Kalinka Vodka and Soda 2dl

                  price: '2999.00'

                  created\_at: '2022-03-27T15:01:10.000000Z'

                  updated\_at: '2022-03-27T15:01:22.000000Z'

                - id: 25

                  name: Goulash soup

                  category: Food

                  description: Pork chops, potato, vegetable, thick soup

                  price: '2999.00'

                  created\_at: '2022-03-27T15:13:49.000000Z'

                  updated\_at: '2022-03-27T15:13:49.000000Z'

                - id: 26

                  name: Ginger tea

                  category: Drink

                  description: Hot natural ginger tea

                  price: '799.00'

                  created\_at: '2022-03-27T15:15:24.000000Z'

                  updated\_at: '2022-03-27T15:15:24.000000Z'

                - id: 27

                  name: Black tea

                  category: Drink

                  description: Earl gray black tea

                  price: '599.00'

                  created\_at: '2022-03-27T15:16:43.000000Z'

                  updated\_at: '2022-03-27T15:16:56.000000Z'

                - id: 29

                  name: Grilled chicken

                  category: Food

                  description: Spicy chicken grilled

                  price: '2850.00'

                  created\_at: '2022-04-08T19:07:48.000000Z'

                  updated\_at: '2022-04-08T19:07:48.000000Z'

                - id: 30

                  name: Honey

                  category: Food

                  description: small box of honey

                  price: '699.00'

                  created\_at: '2022-04-08T19:29:14.000000Z'

                  updated\_at: '2022-04-08T19:30:29.000000Z'

  /api/logout:

    post:

      tags:

        - default

      summary: http://localhost:8000/api/logout

      requestBody:

        content:

          application/x-www-form-urlencoded:

            schema:

              properties:

                name:

                  type: string

                  example: Andras2

                email:

                  type: string

                  example: andras2@gmail.com

                password:

                  type: string

                  example: xyz23

                password\_confirmation:

                  type: string

                  example: xyz23

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:02:07 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: You have succesfully logged out

        '401':

          description: Unauthorized

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:02:23 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Unauthenticated.

  /api/login:

    post:

      tags:

        - default

      summary: http://localhost:8000/api/login

      requestBody:

        content:

          application/x-www-form-urlencoded:

            schema:

              properties:

                name:

                  type: string

                  example: Admin

                email:

                  type: string

                  example: admin@gmail.com

                password:

                  type: string

                  example: Admin111$

                password\_confirmation:

                  type: string

                  example: Admin111$

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '201':

          description: Created

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:47:16 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                user:

                  id: 1

                  name: admin

                  email: admin@gmail.com

                  role: Admin

                  email\_verified\_at: '2022-03-20T19:07:07.000000Z'

                  current\_team\_id: null

                  profile\_photo\_path: null

                  created\_at: '2022-03-20T19:06:58.000000Z'

                  updated\_at: '2022-03-20T19:07:07.000000Z'

                  address: Rakoczi utca 85

                  phone\_number: '+36201111111'

                  profile\_photo\_url: >-

                    https://ui-avatars.com/api/?name=a&color=7F9CF5&background=EBF4FF

                token: 15|3XKkWLZA90yQ5sPjViOE6fUYKKNb6P4NnxPYMnCP

        '401':

          description: Unauthorized

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:48:13 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '60'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Wrong credentials entered

  /api/products/4:

    get:

      tags:

        - default

      summary: http://localhost:8000/api/products/25

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:02:54 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '58'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              examples:

                example-0:

                  summary: http://localhost:8000/api/products/25

                  value:

                    id: 25

                    name: Goulash soup

                    category: Food

                    description: Pork chops, potato, vegetable, thick soup

                    price: '2999.00'

                    created\_at: '2022-03-27T15:13:49.000000Z'

                    updated\_at: '2022-03-27T15:13:49.000000Z'

                example-1:

                  summary: http://localhost:8000/api/products/4

                  value:

                    id: 4

                    name: Beef burger

                    category: Food

                    description: 330g meat, salad, mayo, burger

                    price: '3999.99'

                    created\_at: null

                    updated\_at: '2022-03-27T14:46:37.000000Z'

  /api/products/30:

    put:

      tags:

        - default

      summary: http://localhost:8000/api/products/30

      requestBody:

        content:

          application/x-www-form-urlencoded:

            schema:

              properties:

                name:

                  type: string

                  example: Honey

                category:

                  type: string

                  example: Food

                description:

                  type: string

                  example: small box of honey

                price:

                  type: integer

                  example: '499'

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:06:41 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                id: 30

                name: Honey

                category: Food

                description: small box of honey

                price: '899'

                created\_at: '2022-04-08T19:29:14.000000Z'

                updated\_at: '2022-04-08T20:06:41.000000Z'

        '401':

          description: Unauthorized

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:06:13 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Unauthenticated.

        '403':

          description: Forbidden

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Sat, 09 Apr 2022 18:39:54 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '58'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Forbidden

    delete:

      tags:

        - default

      summary: http://localhost:8000/api/products/4

      security:

        - bearerAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:49:14 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Content-Type:

              schema:

                type: string

                example: text/html; charset=UTF-8

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            text/plain:

              schema:

                type: string

              example: '1'

        '401':

          description: Unauthorized

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:50:23 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Unauthenticated.

        '403':

          description: Forbidden

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Sat, 09 Apr 2022 18:39:31 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                message: Forbidden

  /api/products/search/long:

    get:

      tags:

        - default

      summary: http://localhost:8000/api/products/search/chicken

      security:

        - noauthAuth: []

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '200':

          description: OK

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:32:46 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '57'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              examples:

                example-0:

                  summary: http://localhost:8000/api/products/search/chicken

                  value:

                    - id: 3

                      name: Chicken steak

                      category: Food

                      description: Countryside chicken

                      price: '3000.00'

                      created\_at: null

                      updated\_at: null

                    - id: 22

                      name: Chicken wings

                      category: Food

                      description: Spicy grilled wings

                      price: '2350.00'

                      created\_at: '2022-03-27T14:53:50.000000Z'

                      updated\_at: '2022-03-27T14:53:50.000000Z'

                    - id: 29

                      name: Grilled chicken

                      category: Food

                      description: Spicy chicken grilled

                      price: '2850.00'

                      created\_at: '2022-04-08T19:07:48.000000Z'

                      updated\_at: '2022-04-08T19:07:48.000000Z'

                example-1:

                  summary: http://localhost:8000/api/products/search/chicken

                  value:

                    - id: 5

                      name: Long Island

                      category: Cocktail

                      description: Rum, Tequila, Vodka, Coke

                      price: '3999.00'

                      created\_at: '2022-03-21T02:03:35.000000Z'

                      updated\_at: '2022-03-23T08:02:03.000000Z'

  /api/register:

    post:

      tags:

        - default

      summary: http://localhost:8000/api/register

      requestBody:

        content:

          application/x-www-form-urlencoded:

            schema:

              properties:

                name:

                  type: string

                  example: tester

                email:

                  type: string

                  example: user@gmail.com

                password:

                  type: string

                  example: Testing111

                password\_confirmation:

                  type: string

                  example: Testing111

      parameters:

        - name: Accept

          in: header

          schema:

            type: string

          example: application/json

      responses:

        '201':

          description: Created

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 20:01:06 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              example:

                user:

                  name: tester

                  email: tester1234@gmail.com

                  updated\_at: '2022-04-08T20:01:04.000000Z'

                  created\_at: '2022-04-08T20:01:04.000000Z'

                  id: 7

                  profile\_photo\_url: >-

                    https://ui-avatars.com/api/?name=t&color=7F9CF5&background=EBF4FF

                token: 16|l3cQirhIWpgDpe7vi6ngDMAxnIsjXMObGWmTtWCt

        '422':

          description: Unprocessable Content

          headers:

            Host:

              schema:

                type: string

                example: localhost:8000

            Date:

              schema:

                type: string

                example: Fri, 08 Apr 2022 19:50:43 GMT

            Connection:

              schema:

                type: string

                example: close

            X-Powered-By:

              schema:

                type: number

                example: PHP/8.0.12

            Cache-Control:

              schema:

                type: string

                example: no-cache, private

            Content-Type:

              schema:

                type: string

                example: application/json

            X-RateLimit-Limit:

              schema:

                type: integer

                example: '60'

            X-RateLimit-Remaining:

              schema:

                type: integer

                example: '59'

            Access-Control-Allow-Origin:

              schema:

                type: string

                example: '\*'

          content:

            application/json:

              schema:

                type: object

              examples:

                example-0:

                  summary: http://localhost:8000/api/register

                  value:

                    message: The given data was invalid.

                    errors:

                      email:

                        - The email has already been taken.

                      password:

                        - The password confirmation does not match.

                example-1:

                  summary: http://localhost:8000/api/register

                  value:

                    message: The given data was invalid.

                    errors:

                      email:

                        - The email has already been taken.