

Projektmunka 2.

GKNB\_INTM005

Dokumentáció a Recept e-book weboldal felépítéséhez

2023/24/1

Készítette:

Rába Róbert AY6761

Szücs Inez Kíra GKH0WC

Kiss Luca Regina CCINJX

Tartalom

[Bevezetés 3](#_Toc152404572)

[Alkalmazott technológiai háttér / Fejlesztési környezet 4](#_Toc152404573)

[Visual Studio Code 4](#_Toc152404574)

[Laravel 4](#_Toc152404575)

[Github: 6](#_Toc152404576)

[Design: (UI/UX) 7](#_Toc152404577)

[Figma 7](#_Toc152404578)

[Alapvető Design Irányelvek 9](#_Toc152404579)

[Adatbázis-kezelés 9](#_Toc152404580)

[Docker 9](#_Toc152404581)

[DBeaver 10](#_Toc152404582)

# Bevezetés

A dokumentáció bemutatja weboldalunk felépítéséhez, létrehozásához szükséges programokat, illetve megoldásokat. Az oldalunk megalkotásának célja az volt, hogy egy olyan webalkalmazást szerettünk volna létrehozni, ahol az emberek megoszthatják egymással kedvenc receptjeiket, ezáltal tudnak inspirálódni. A felület lehetővé teszi a felhasználók számára, hogy létrehozzanak egy saját profilt, ahol elmenthetik kedvenc receptjeiket, illetve fel is tölthetik, meg is oszthatják azokat.

A weboldal megvalósítása során számos lépést kellett megtenni. Az első lépés a tervezés és specifikálás volt. Ebben a fázisban a funkcionalitás, a felhasználói felület tervezése, az adatbázis tervezése, a weboldal architektúrájának kialakítása, valamint a technológiák és keretrendszerek kiválasztása voltak az alapvető feladatok.

A tervezés befejeztével a következő lépés a fejlesztés volt. Ehhez szükség volt különböző eszközökre, mint például a Figma a felhasználói felület tervezéséhez, a Docker az alkalmazások konténerizációjához és könnyű telepítéséhez, a dBeaver az adatbázis tervezéséhez és kezeléséhez, valamint a Visual Studio Code a fejlesztési munkafolyamatokhoz.

A backend technológiaként a Laravel került kiválasztásra. Ez a fázis lehetővé tette a tervezés alapján kiválasztott technológiák és eszközök összehangolt alkalmazását a weboldal fejlesztése során. Ennek eredményeként egy dinamikus és hatékony webalkalmazást sikerült létrehozni.

# Alkalmazott technológiai háttér / Fejlesztési környezet

## Visual Studio Code

A Visual Studio Code (rövidítve: VSCode vagy VS Code) egy ingyenes, nyílt forráskódú kódszerkesztő, amelyet a Microsoft fejlesztett a Windows, Linux és macOS operációs rendszerekhez. A szoftver számos fejlett funkciót kínál a fejlesztők számára, többek között hibakereső támogatást, beépített Git integrációt és intelligens kódkiegészítést az IntelliSense által.

A VSCode lehetővé teszi a felhasználók számára, hogy testreszabják a kódszerkesztő környezetüket számos módon. Például megváltoztathatják a kinézetet (téma), átrendezhetik a szerkesztő gyorsbillentyű-kiosztását, és testreszabhatják az alapértelmezett beállításokat. Emellett a szoftver támogatja a kiegészítőket, amelyek lehetővé teszik további funkciók és testreszabási lehetőségek hozzáadását a kódszerkesztés folyamatához. Ezen funkciók és testreszabási lehetőségek révén a Visual Studio Code rendkívül alkalmazkodható és hatékony eszköz a fejlesztők számára, amely segíti őket a produktív munkavégzésben és a kódszerkesztési folyamat optimalizálásában.

## Laravel

Laravel egy elterjedt PHP keretrendszer, melyet az elegáns szintaxisa, erős funkciói és fejlesztőbarát környezete miatt ismernek. Széles körben használják webalkalmazások és API-k építésére a könnyű használhatósága, skálázhatósága és erőteljes eszközei miatt. Egyszerűsíti a backend fejlesztés folyamatát egy jól strukturált és fejlesztőbarát környezetet kínálva, ezért az egyik legnépszerűbb választás a megbízható webalkalmazások építéséhez. Segítik és gyorsítják a webalkalmazások fejlesztését, miközben fenntartják a kód tisztaságát és a biztonságot.

Eloquent ORM:

Eloquent egy olyan adatbázis-kezelő rendszer, ami az objektumok és az adatbázis táblák közötti kapcsolatot hozza létre. Ez lehetővé teszi, hogy az adatbázis műveleteket (lekérdezések, beszúrások, frissítések stb.) objektumokon keresztül végezzük, ami sokkal olvashatóbbá és könnyebben karbantarthatóvá teszi a kódot.

Routing:

A Laravelben könnyedén definiálhatunk útvonalakat a webes alkalmazásunk számára. Ez azt jelenti, hogy meghatározhatjuk, hogy egy adott URL vagy kérés mely vezérlőhöz és műveletekhez tartozik, így könnyen kezelhetjük az alkalmazás logikáját.

Middleware:

A Middleware egy olyan réteg a Laravel alkalmazásban, ami lehetővé teszi a kérés feldolgozását mielőtt az eljutna a végpontjához (controllerhez). Ez kiválóan alkalmas például az autentikációra, engedélyezésre, naplózásra vagy bármilyen egyéb előzetes feldolgozásra.

Blade Sablonmotor:

A Blade egy könnyen használható és kifejező sablonmotor a Laravelben, amely lehetővé teszi a PHP kód tiszta és strukturált beillesztését a nézetekbe. Ez segít elkerülni a felesleges ismétlődést és a könnyebb nézetek kezelését.

Artisan CLI:

Az Artisan egy parancssoros interfész, amely segít automatizálni a gyakran használt feladatokat, például adatbázis migrációk kezelését, szolgáltatások létrehozását vagy akár saját parancsok készítését az alkalmazáshoz.

Azonosítás és Jogosultságkezelés:

A Laravelben beépítve vannak egyszerűen használható eszközök az azonosításra és jogosultságkezelésre. Ez könnyűvé teszi az autentikációt és az alkalmazás belső jogosultságainak kezelését.

Tesztelés:

A Laravelben könnyű és integrált módon tesztelhetjük az alkalmazásunkat PHPUnit segítségével, így biztosíthatjuk az alkalmazás megbízhatóságát és stabilitását.

Biztonság: A Laravel törekszik a biztonságra, és számos beépített funkcióval rendelkezik a gyakori webes sebezhetőségek elleni védelemre, ezáltal segítve a fejlesztőket az alkalmazásuk biztonságának fenntartásában.

Közösség és Ökoszisztéma:

A Laravel hatalmas és aktív közösséggel rendelkezik, amely rengeteg kiegészítő csomagot, dokumentációt, és támogatást biztosít a fejlesztőknek, így könnyen bővíthető és frissíthető az alkalmazás.

## Github:

GitHub egy webes alapú platform, amelyet a fejlesztők használnak verziókezeléshez és kódmegosztáshoz. Alapvetően egy távoli tárhelyszolgáltatás, amely lehetővé teszi a fejlesztők számára, hogy tárolják és nyomon kövessék a projektjeik forráskódját, valamint együttműködjenek más fejlesztőkkel. Néhány alapvető jellemzője:

Verziókezelés (Version Control):

GitHub erőteljes verziókezelési rendszert biztosít, amely lehetővé teszi a fejlesztők számára, hogy nyomon kövessék a projektjük változásait, visszaállítsák előző verziókat, és könnyen együttműködjenek másokkal a kódon.

Tárolók (Repositories):

A projekt forráskódjának tárolására szolgáló helyek, amelyeket a fejlesztők létrehozhatnak és kezelhetnek. Minden tároló tartalmazza a projekt teljes történetét és összes változatát.

Ágak (Branches):

A fejlesztők ágakat hozhatnak létre a projektjükben, hogy elkülönítsék és kísérletezzenek új funkciókkal vagy változtatásokkal, mielőtt azokat a fő ágba integrálnák.

Problémák és Hibajegyek (Issues and Bug Tracking):

GitHub lehetővé teszi a felhasználók számára, hogy problémákat jelezzenek, és kövessék azok státuszát, illetve hibajegyeket nyissanak, majd kezeljék a projekt hibáit.

Pull kérések (Pull Requests):

Ez a mechanizmus a fejlesztők számára lehetővé teszi, hogy változtatásokat javasoljanak egy projektben, és kérjék azok beillesztését a fő ágba.

Collaboration (Együttműködés):

A GitHub lehetővé teszi a fejlesztők számára, hogy együttműködjenek a projekteken, kövessék egymás munkáját, és könnyen összehangolják a fejlesztési folyamatokat.

# Design: (UI/UX)

## Figma

Figma egy webes alapú tervezőeszköz, amelyet elsősorban felhasználói felület (UI) és felhasználói élmény (UX) tervezésre használnak. A platform lehetővé teszi, hogy tervezők és csapatok együttműködjenek a tervezési folyamat különböző szakaszaiban. Néhány jellemzője:

Kollaboráció:

Figma lehetővé teszi, hogy több felhasználó egyidejűleg dolgozhasson ugyanazon projektjén. A tervezők valós időben láthatják egymás munkáját, és könnyen megoszthatják gondolataikat.

Vektorgrafika:

Figma vektorgrafikus szerkesztőt kínál, amely lehetővé teszi a felhasználók számára, hogy létrehozzanak és módosítsanak vektorgrafikákat, például ikonokat vagy logókat.

Prototípusok:

A tervezők prototípusokat készíthetnek, hogy élethűen szimulálják a felhasználói interakciókat és az alkalmazás vagy weboldal működését.

Komponensek és Stílusok:

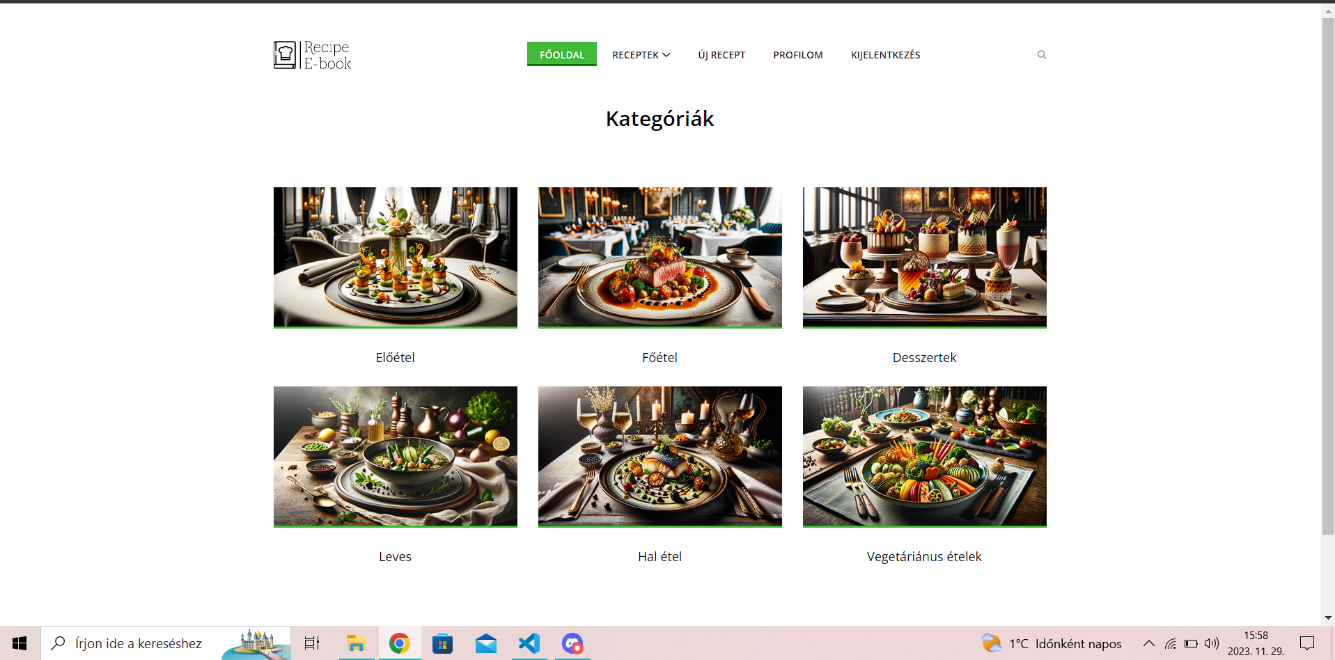
A Figma lehetővé teszi a tervezők számára, hogy komponenseket és stílusokat definiáljanak, így könnyedén alkalmazhatók az egész projektben. Ez segít az egységes dizájn fenntartásában.

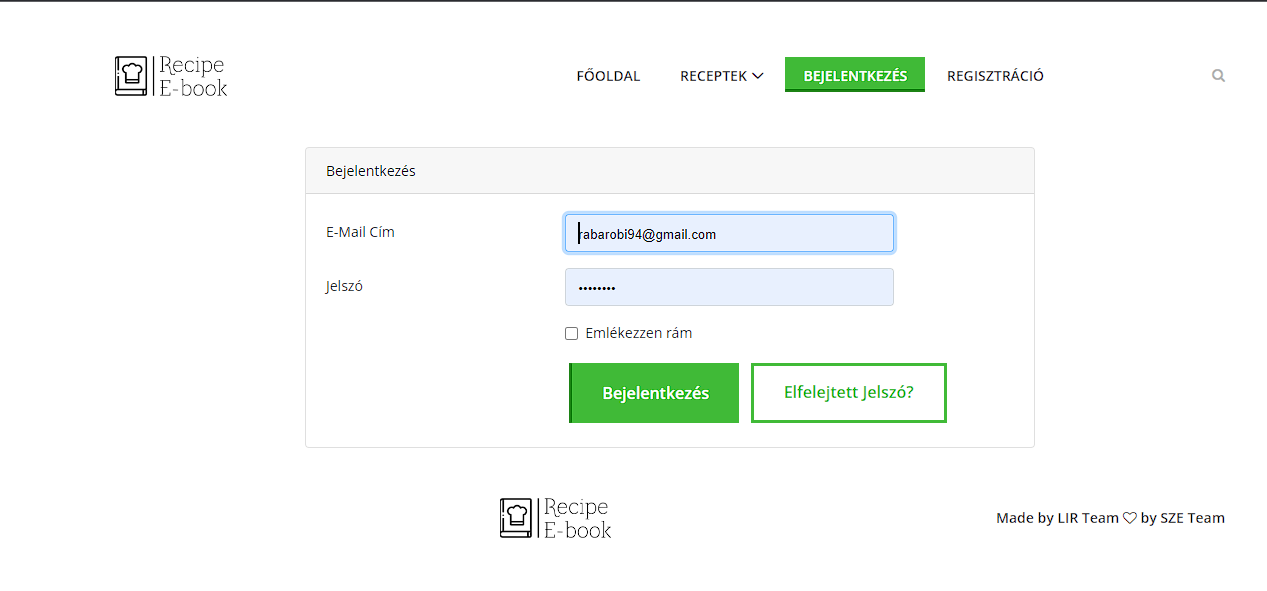
Interaktív Elemek:

A platform támogatja az interaktív elemek, animációk és átmenetek készítését, amelyek segítik a tervezőket a valósághűbb felhasználói élmény kialakításában.

Eszközintegrációk:

Figma könnyen integrálható más tervezési és fejlesztési eszközökkel, például a GitHubkal vagy a Slackel, ami további kollaborációs lehetőségeket kínál.

******

******

# Alapvető Design Irányelvek

Minimalizmus és Tisztaság:

A tervezés egyszerűsége és letisztultsága elsődleges szempont. Ennek célja, hogy a felhasználó könnyedén eligazodjon az alkalmazáson vagy weboldalon keresztül anélkül, hogy zavaró vagy felesleges elemekkel találkozna. A kevesebb információra való összpontosítás lehetővé teszi a lényeg kiemelését és a felhasználó számára valószínűsíti, hogy a keresett információt gyorsan megtalálja.

Színválasztás:

A színeknek fontos szerepük van a felhasználói élményben és a hangulat megteremtésében. Világos színek használata segít abban, hogy barátságos, hívogató légkört teremtsünk, ami könnyen befogadható a felhasználók számára. Ugyanakkor a kontrasztos színek hatékonyan kiemelik a fontosabb részeket, így segítve a felhasználókat azok azonosításában és az interakcióban velük.

Ikonok és Gombok:

Az áttekinthető és egyértelmű ikonok kulcsfontosságúak a felhasználók számára, hogy könnyedén megtalálják és használják az alkalmazás vagy weboldal funkcióit. Az érthető ikonok segítik a könnyű navigációt és a gyorsabb információelérhetőséget. Emellett a nagy, érintésre optimalizált gombok használata biztosítja, hogy a felhasználók kényelmesen és hatékonyan tudjanak interakcióba lépni az alkalmazással vagy weboldallal, függetlenül attól, hogy mobil vagy asztali eszközt használnak.

# Adatbázis-kezelés

## Docker

Docker egy olyan platform, ami lehetővé teszi számítógépes alkalmazások konténerekben történő futtatását. Ezek a konténerek olyan csomagok, amelyek tartalmazzák az alkalmazás összes szükséges dolgát: kódot, futási környezetet, függőségeket és konfigurációt. Könnyebben és hatékonyabban kezeljék az alkalmazásokat, csökkentsék a környezeti problémákat, gyorsan reagáljanak a változó igényekre, és biztonságosabb módon futtassák az alkalmazásaikat.

Könnyű hordozhatóság és környezetfüggetlenség:

A Docker konténerek teljesen elkülönülnek az alkalmazáson kívüli környezettől. Az alkalmazás teljes környezetét, beleértve a kódját, függőségeit, konfigurációját és futási környezetét is magában foglalják. Ez lehetővé teszi, hogy az alkalmazást ugyanolyan módon fussuk bármely környezetben, legyen az fejlesztői gép, tesztelési környezet vagy éles szerver.

Konzisztensség és környezetegységesítés:

A konténerek garantálják az alkalmazások konzisztens futtatását különböző környezetekben. Ez segít elkerülni a környezet-specifikus hibákat, amelyek gyakran előfordulhatnak, ha az alkalmazást különböző környezetekben futtatják.

Gyors telepítés és skálázhatóság:

A Docker konténerek gyorsan indíthatók és leállíthatók. Ez lehetővé teszi az alkalmazások gyors skálázhatóságát, hiszen könnyen lehet példányokat indítani a konténerek alapján. Ez különösen hasznos, amikor nagy terhelést kell kezelni, vagy amikor az alkalmazásnak rugalmasan kell reagálnia a változó igényekre.

Elszigetelés és biztonság:

A Docker konténerek elszigetelik az alkalmazásokat egymástól és a hoszt rendszertől is. Ez biztonságosabb futtatási környezetet teremt, mivel az egyes konténerek nem befolyásolják egymást, és nem érhetik el a hoszt rendszer folyamatait vagy erőforrásait.

Folytonos integráció és folytonos szállítás (CI/CD):

A Docker széles körben használják folytonos integrációhoz és folytonos szállításhoz. A konténerek lehetővé teszik az alkalmazások könnyű csomagolását, tesztelését és terjesztését, így lehetővé téve az agilis fejlesztési folyamatokat és a gyors kiadásokat.

## DBeaver

A DBeaver számos felhasználói réteg számára vonzóvá válik azáltal, hogy egyszerűen kezeli az adatbázisokat, támogatja a különböző adatbázis-rendszereket, és számos fejlett eszközt kínál az adatbázisok kezeléséhez és kezeléséhez. Rugalmassága és támogatottsága miatt sok fejlesztő és adatbázis-szakember számára az elsődleges választás adatbázis-kezelésre és SQL-lekérdezések végrehajtására.

Keresztplatformos alkalmazás:

A DBeaver elérhető Windowsra, macOS-re és Linuxra is, így lehetővé teszi a felhasználók számára az adatbázis-kezelés egyszerű és konzisztens módját különböző operációs rendszereken.

Kiterjedt adatbázis-támogatás:

Az alkalmazás több tucat adatbázis-rendszert támogat, többek között a relációs adatbázisokat (mint a MySQL, PostgreSQL, SQLite, MariaDB), NoSQL adatbázisokat (például MongoDB, Cassandra), valamint számos más adatbázis-technológiát (Oracle, Microsoft SQL Server, Firebird stb.). Ez lehetővé teszi az összes adatbázis típusának közös felhasználói felületen való kezelését.

Kapcsolatok és adatbázis-kezelés:

A DBeaver rugalmas eszközöket kínál az adatbázisokhoz való kapcsolódáshoz és kezeléshez. Képes távoli és helyi adatbázisokhoz is csatlakozni, lehetőséget biztosítva azok szerkezetének és tartalmának megtekintésére, valamint szerkesztésére.

SQL szerkesztő és végrehajtás:

A beépített SQL szerkesztő eszköz lehetővé teszi a felhasználók számára az SQL lekérdezések írását, szerkesztését és végrehajtását az adatbázisokon. Az eredményeket pedig könnyedén megtekinthetjük az alkalmazáson belül.

Adatbázis-objektumok kezelése:

Az alkalmazás lehetővé teszi az adatbázis-objektumok, mint például táblák, indexek, eljárások, nézetek stb. részletes megtekintését és kezelését. A felhasználók könnyen navigálhatnak az adatbázis struktúrájában, megtekinthetik azok tulajdonságait, és szükség esetén módosíthatják azokat.

Adatvizualizáció és exportálás:

A DBeaver adatok vizuális megjelenítését diagramokon keresztül teszi lehetővé, ami segít az adatok jobb megértésében és elemzésében. Emellett lehetőség van az adatok exportálására különböző formátumokba, így könnyen megoszthatók vagy felhasználhatók más eszközökben.

Fejlett eszközök és kiterjesztések:

Az alkalmazás további fejlett funkciókat és kiterjesztéseket kínál az adatbázis-kezeléshez. Ide tartoznak a szinkronizációs lehetőségek, a diagram szerkesztők, a folyamatmodellezési eszközök, valamint a beépített adatbázis-adminisztrációs funkciók.