ZÁRÓDOLGOZAT

Készítették:

Fazekas Réka

Dömötör Gyula

Kolyankovszky Gusztáv

Konzulens:

Farkas Zoltán

Miskolc

2023.

Miskolci SZC Kandó Kálmán Informatikai Technikum

Miskolci Szakképzési Centrum

SZOFTVERFEJLESZTŐ- ÉS TESZTELŐ SZAK

ZÁRÓDOLGOZAT

FADOKO

Étteremi program

Fazekas Réka

Dömötör Gyula

Kolyankovszky Gusztáv

2022-2023

Tartalomjegyzék

Köszönetnyilvánítás (Dömötör Gyula)

Ezúton szeretnénk megköszönni minden tanárunknak és minden olyan személynek, akik közvetlen, vagy közvetett módon is hozzájárultak ahhoz, hogy ez a vizsgaremek szakdolgozat létrejöhessen. Külön köszönet Kerényi Róbert Nándor, Kasza László Róbert, Németh Bence, Farkas Zoltán, Lázár Zsolt, Sándor Péter, Csontos Dénes tanár uraknak, valamint Sándorné Feke Réka és Domján Annamária tanárnőknek, akik egyengették, figyelemmel kísérték utunkat, javaslataikkal és segítő szándékú kritikáikkal előmozdították, hogy teljes értékű szoftverfejlesztő,- és tesztelőkké váljunk.

Miért ezt a programot választottuk?

Webalkalmazásunk megalkotásának elsődleges célja az volt, hogy a különböző szolgáltató,- és vendéglátóipari vállalkozások számára egy igényes felhasználói felülettel, hatékony adatbázissal és stabil backend szerverrel rendelkező olyan programot adjunk, mely lehetővé teszi a felhasználók részére:

- termékek (jelen esetben pizzák) megrendelését,
- a weboldalon való regisztrációt,
- új termékek felvételét, módosítását és törlését,
- a termékek tulajdonságainak adatbázisból való lekérdezését
- új felhasználó regisztrálását,
- módosítását és törlését,
- bejelentkezését, kijelentkezését.

A webalkalmazás adatbázisáról

A mindennapi életben szinte észre se vesszük, de folyamatosan találkozhatunk különböző adatbázisokkal. Már akkor is, ha egy fárasztó nap után kikapcsolódásképpen zenét hallgatunk, filmet nézünk, mivel a különböző alkalmazásokon keresztül érkező médiatartalmak lejátszólistái is adatbázisból származnak. Tovább gondolva, ha a könyvtár számítógépén utánanézünk egy-egy könyvnek, hogy éppen hány példány érhető el abban a pillanatban az adott könyvtárban. Vagy, ha raktárban dolgozunk egy multinacionális, vagy éppen kisebb ipari cégnél, a raktárnyilvántartás nagyon fontos részét az adatbázis képezi, ahol folyamatosan figyelemmel lehet kísérni alapanyagok, félkész, - és késztermékek mozgását, felhasználását. Illetve, hogy a jelen esethez is igazodjunk, vendéglátóipari egységek, éttermek ételrendelő alkalmazásainak adatbázisai.

Alapfogalmak

Adatbázis

olyan rendszer, amely lehetővé teszi különböző adatok hatékony tárolását, rendezését. Az adatok ebben lehetnek szöveges adatok, kép vagy videófájlok, valamint hangfájlok is, de sok másfajta adat is. Az adatok strukturált formában kerülnek tárolásra benne, adatmezők, oszlopok és rekordok alkotják. Segítségükkel nemcsak tárolhatjuk adatainkat, hanem biztosíthatjuk azok védelmét.

Adatbázis-kezelő rendszer

olyan szoftver, amely lehetővé teszi adatbázisok létrehozását, frissítését, az adatok módosítását és törlését. (DBMS - Database Management System). Több fajtája létezik:

- RDBMS (Relational Database Management System), relációs adatbázis kezelő rendszer

- OODBMS (Object Oriented Database Management System) objektum orientált adatbázis kezelő rendszer

Adatmodell

olyan absztrakt megjelenítése az adatoknak, amelyek az adatok összefüggéseit, kapcsolatait, viselkedését és jellemzőit írja le, diagramok, táblázatok vagy más grafikus formában. Ezekkel lehet megtervezni az adatbázisokat így téve hatékonyan használhatóvá őket. Több típusa is létezik ezeknek az adatmodelleknek, a legismertebb és legtöbbet használt az egyed-kapcsolat, vagy entitás-kapcsolat modell (ER modell ⇨ Entity-Relationship modell).

Kapcsolat típusok

- egy-egy kapcsolat ⇨ az egyik entitáshoz pontosan csak egy másik tartozhat és fordítva

- egy-több kapcsolat ⇨ egy entitáshoz több más entitás is tartozhat

- több-több kapcsolat ⇨ több entitás több entitáshoz is kapcsolódhat.

Entitás

Az adatbázis kezelő rendszerek az entitások révén kezelik és tárolják az adatokat. Lehet például ügyfél, megrendelés, termék, szállító, dolgozó stb.

Normálformák

A relációs adatbázisok tervezésének szabályai, melyek segítenek az adatok redundanciájának csökkentésében. Öt normálforma létezik, melyek:

- **első normálforma** (1NF) ⇨ minden adat egyértelműen definiált és oszlopokba van rendezve és ezek nem tartalmaznak ismétlődő adatokat

- **második normálforma** (2NF) ⇨ ugyanaz, mint az előbb, és minden nem elsődleges kulcsú oszlop teljesen függ a tábla elsődleges kulcsától

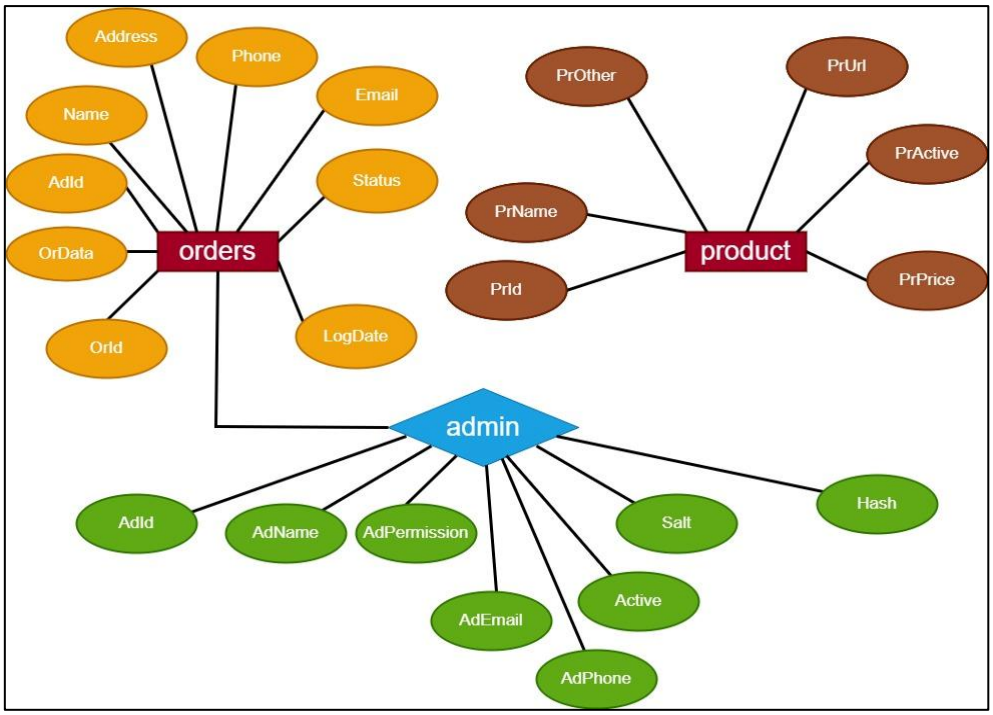
- **harmadik normálforma** (3NF) ⇨ a második normálforma tulajdonságait hordozza és minden nem elsődleges kulcsú oszlop nem függ más nem elsődleges kulcsú oszloptól

- **Boyce-Codd normálforma** (BCNF) ⇨ itt a harmadik normálformának felelnek meg az adatbázis táblái és minden funkcionális függőség az oszlopok között a tábla elsődleges kulcsára vonatkozik

- **negyedik normálforma** (4NF) ⇨ itt a Boyce-Codd normálformának felelnek meg az adatbázis táblái és nem tartalmaznak többértékű függőséget

Funkcionális függőség azt jelenti, hogy az egyik adat értéke meghatározható egy másik adat értékéből.

Az adatbázis megtervezésénél fontos lépés egy ER-diagram létrehozása. Az ER-diagram a különböző entitásokat és azoknak a kapcsolatait jeleníti meg. Használatuk számos előnnyel jár az adatbázis tervezési folyamatában. Általuk megérthetjük az üzleti folyamatokat, javítják az adatbázis integritását, lehetővé teszi az adatbázis tervező számára, hogy biztosítsa az adatok helyes tárolását és általuk könnyebben karbantartható és frissíthető az adatbázis.

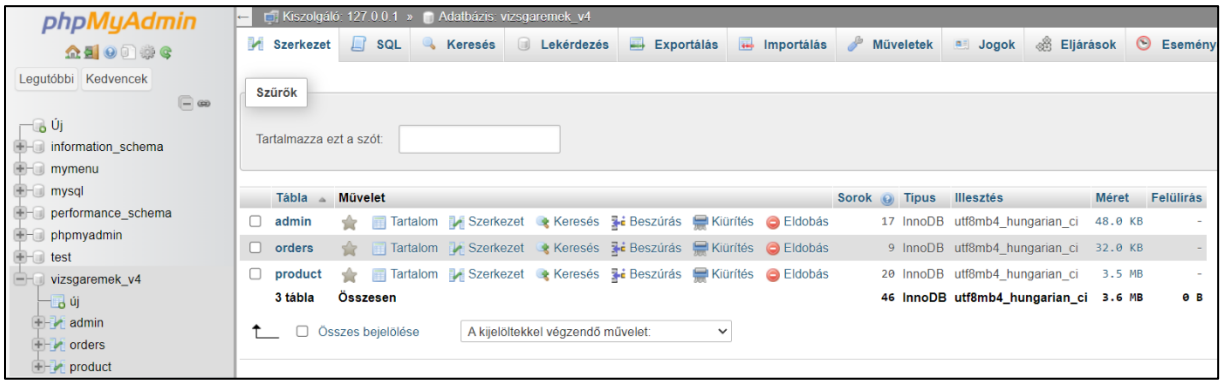


1.ábra: a vizsgaremek_v4 pizza adatbázis ER-diagramja

A pizza adatbázisom létrehozását is ez alapján terveztem meg. Az adatbázist és annak tábláit, adatait phpMyadmin felületen MySQL rendszerben valósítottam meg.

Az adatbázis-kezelő rendszer SQL (Structured Query Language Strukturált Lekérdező Nyelv) nyelven készült. Az adatbázis három táblából áll, admin, orders és product táblák. A következő oldalakon szemléltetem és írom le az egyes táblák tulajdonságait, valamint azt, hogy milyen adatokat tartalmaz.

Az adatbázis kezeléséhez a phpMyAdmin programot alkalmaztam.



2. ábra: Az ER-diagram alapján létrehozott vizsgaremek_v4 pizza adatbázis és táblái képernyőképpel

1. tábla - admin (személyes adatok)

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	**AdId**	int(11)			Nem	*Nincs*		AUTO_INCREMENT	Módosítás Eldobás Több
2	**AdName**	varchar(40)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
3	**AdPermission**	varchar(10)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
4	**AdEmail**	varchar(40)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
5	**AdPhone**	varchar(20)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
6	**Active**	int(2)			Nem	*Nincs*			Módosítás Eldobás Több
7	**Salt**	varchar(64)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
8	**Hash**	varchar(64)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több

3. ábra: az admin tábla képernyőképe

Az admin tábla szerkezete.

A táblában lévő adatok két felé választhatóak. Manuálisan bevitt/rögzített adatok és program által generált adatok. A generált adatok szolgálhatnak felhasználó részére kényelmi funkciót és biztonsági funkciót. Az azonosító AdId jelen esetben kényelmi, a Salt és a Hash biztonsági okokból lett generáltatva.

- **AdId** ⇨az adminisztrátor azonosító száma, amely az elsődleges kulcs (Primary key) a táblában, Auto Increment értékkel, mely az azonosító számokat automatikusan generálja, valamint egy integer egész szám

- **AdName**⇨az adminisztrátor neve, mely egy variálható karakter, azaz tartalmazhat számokat és betűket is azaz varchar, valamint az idegen kulcs (Foreign key) a táblában

- **AdPermission**⇨az admin felületen a beregisztrált felhasználók jogkör engedélyezési száma. Jelenleg 4-től 9-ig vannak jogosultságok.

- **AdEmail** ⇨ adminisztrátor, vagy felhasználó e-mail címe

- **AdPhone** ⇨ adminisztrátor, vagy felhasználó telefonszáma

- **Active** ⇨ két aktivitási fokozatot foglal magában: 0 = inaktív, 1 = aktív. Ha az adott dolgozó, vagy felhasználó, illetve pizza valamilyen okból adott pillanatban nem elérhető, akkor 0-ás jelzést kap, azaz inaktív, ha elérhető akkor 1-es státuszú, azaz elérhető, tehát aktív. Csak a megfelelő jogkörrel rendelkező adminisztrátor módosíthatja.

- **Salt** ⇨ ez a salt rendszer egy biztonságos jelszókezelő algoritmusra a bcryptre támaszkodik. Lényege, hogy a felhasználók és adminisztrátorok jelszavaihoz hozzáad egy véletlenszerű karakterláncot, amelyet salt-nak neveznek. A felhasználó által használni kívánt jelszó és ez a salt érték alkotja a hash értéket.

- **Hash** ⇨ a hash-érték különösen fontos a jelszókezelésben. Lényege, hogy nem tárolunk sehol jelszavakat, csak magát ezt a hash-értéket. Minden felhasználói regisztrációkor a jelszó beírásakor a slat-értékkel együtt generálódik. Ha megtörtént a regisztráció, az adatbázisban csak a salt és a hash-érték kerül tárolásra. Amikor a felhasználó bejelentkezik a saját jelszavával, a backend szerveren és az adatbázisban eltárolásra került hash-értéket keresi ki a rendszer az adott jelszóra és ha minden egyezik, megtörténik a felhasználó beléptetése. Fontos feladata tehát az adatvédelem és az illetéktelen hozzáférés megakadályozása.

Példa az admin tábla feltöltött állapotára.

	AdId	AdName	AdPermission	AdEmail	AdPhone	Active	Salt	Hash
Módosítás Másolás Törlés	1	Fazekas Réka	9	teszt@teszt.hu	+3630123457	1	26myfhtlKtlcw3JnmtQdusfpnZPhvUnjmiGdTc40uwaQdNbE9M...	4cd1d334928ebf22b247a8ffdfae3171d3d552f47ca281b183...
Módosítás Másolás Törlés	2	a	9	p@p.hu	+361222333	1	DO77MZpGD726inT4dhJcUpt7wCYb75UNfN8OhbZANZgJfSQVqI...	36cd6463bbda2722da147c757cacd5a652e8d0610f44bcb58b...
Módosítás Másolás Törlés	3	Kolyankovszky Gusztáv	9	kg@kg.hu	+3670	1	hyD7KTz06AdOpATFk7KWycl2NRYH0AQuTmoEfFdcsyiZki1xys...	7fdfcacc1bbc99a48778a18770ae72321a12e54417a05f1f82...
Módosítás Másolás Törlés	4	Domotor Gyula	9	dgy@dgy.hu	+3670	1	hyD7KTz06AdOpATFk7KWycl2NRYH0AQuTmoEfFdcsyiZki1xys...	7fdfcacc1bbc99a48778a18770ae72321a12e54417a05f1f82...
Módosítás Másolás Törlés	5	Kukor Ica	8	kukorica@freemail.hu	0649535150	1	ouMSJlf44lv5rstFpIfXzsiNJdmHcR487ZddgqOOL3zrxdMh8s...	3db705c8949e83b38acc1f8f70719f3e9407573e5317b59552...
Módosítás Másolás Törlés	6	Piz Zalán	7	pizzalan@gmail.com	+3630123456	1	CsyMa3fpjFAErzcBOhcN9DFBiik8qXSWiN3lKodg01OqHj1skE...	6f64b22508e43f9cde444264b8c4350c85b758bc420519e401...
Módosítás Másolás Törlés	7	Olivabogy Olivér	6	olivabogyo@gmail.com	+3620111222	1	En0EK85DxKag4gvAa19l980JuAT8Hs93UK5nXJJSAXaYcTfGyt...	a2674dd77b483963ce35e4d6bfc0e2796901c973c63a41272c...
Módosítás Másolás Törlés	8	Sonk Andrea	6	sonka@citromail.hu	+36301112226	1	ardSb2gFVlAnLQG5o02jP7pTwsQN0GZU8gK3WsNStAiLCaZdSq...	b166ce96c07cc339594fe5745c10117c6563ba1f2f1459f702...
Módosítás Másolás Törlés	9	Virs Lee	5	virsli@gmail.com	+36203336668	1	xNYk2DkTEcFfHTTQCnRUlDUCtj7eeGMs69mtnLhuuoPxTY4jG0...	f13957136a6d4ba221ae43c4fdf999ded63b2def2f40e41e81...
Módosítás Másolás Törlés	10	Fust Lee	5	fustli@citromail.hu	+36301234568	1	zuZUmPAyNbGZxJ5U4eeU40USrJn3ZfBD2iCJVh2aTT2poUd0HM...	2aabd0590dc2f028da18cb1343cfdacb3944896cabaeb0b658...
Módosítás Másolás Törlés	11	Kif Lee	4	kifli@gmail.com	+3670111222	1	AHaFeyH322iOOWFa67ASFgq4ZATgPAlKfWf5DZffzsLQceHqWm...	853dca1fed4f957638dbdc9a2aa4e5eb6c11ad629e0cf30ad9...
Módosítás Másolás Törlés	12	Bruce Lee	4	brucelee@citromail.hu	+36309998786	1	qQmwimloEIFwFCnNpbHzjJSyJ3PWhVmaTpMNFQkScWqABL97Bj...	fb05759325f3e3a986473d9edc350d17d60da4be3fc1173b24...
Módosítás Másolás Törlés	36	jog8	8	jog8@jog8.hu	+3670111222	1	lEpB7qoljUouMGiNRAyliYxAItSQyXfMiykdEilekxi6ky27r5...	f57432ca55b1832debda644b5e8ec16c3c0e2a170ade9f3459...
Módosítás Másolás Törlés	37	jog7	7	jog7@jog7.hu	+3612345678	1	3ZoZ9VVtNEfgBm9BNwZtMDVXLeVnpC5NLO8udC25gKCz9fyh6O...	2025a7ce662673fb3ff8978e7afb145086db20bdbc61719dce9...
Módosítás Másolás Törlés	38	jog6	6	jog6@teszt.hu	+3670111222	1	DpOccZptZS45FqL4hkMmM74zilmfa9cg2dyklzMglPiZzelOtY...	a603a5adc745ab68f488743a5d230250bb8e1ac67820958bb2...
Módosítás Másolás Törlés	39	jog5	5	jog5@teszt.hu	+3612345678	1	Sy0FR0dl9L3QLeALtKuwSwQdGCrcj3VvTIpC3gZHeDi44DOCzz...	25575b33aed3e8f2e5a6b9e53f309c1bfc95a9df46b668984f...
Módosítás Másolás Törlés	40	jog4	4	jog4@teszt.hu	+3670111222	1	sTEwId1DUUiytBH7nOk7rw76vbxULoHFYhUailhiFe5oqoimKB...	9d2d083762f86819ac18afb6add7a567702ff7189362760d17...

4. ábra: az admin tábla adattartalma

2. tábla - orders (rendelések)

	#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
☐	1	**OrId**	int(11)			Nem	*Nincs*		AUTO_INCREMENT	Módosítás Eldobás Több
☐	2	**OrData**	varchar(500)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
☐	3	**AdId**	int(11)			Igen	*NULL*			Módosítás Eldobás Több
☐	4	**Name**	varchar(100)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
☐	5	**Address**	varchar(100)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
☐	6	**Phone**	varchar(20)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
☐	7	**Email**	varchar(100)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
☐	8	**Status**	int(11)			Nem	*Nincs*			Módosítás Eldobás Több
☐	9	**LogDate**	datetime			Nem	*Nincs*			Módosítás Eldobás Több

5. ábra: az orders tábla képernyőképe

Az orders tábla szerkezete.

- **OrId** ⇨ a rendelés azonosító száma, amely egy integer egész szám, a tábla elsődleges kulcsa, a rendszer automatikusan generálja, ezért auto increment

- **OrData** ⇨ a megrendelt pizza azonosítószámát és méretét tartalmazza, melyet a product táblából kapunk meg

- **AdId** ⇨ annak az adminisztrátornak, vagy megfelelő jogkörrel felruházott felhasználónak az azonosítószáma, aki a pizzarendelést felvette. Ezt az adatot az admin táblából kapjuk

- **Name, Address, Phone, Email** ⇨ a pizzát megrendelő neve, lakcíme, telefonszáma és email címe, ahová ki kell szállítani a megrendelést

- **Status** ⇨ a megrendelés státuszát tartalmazza. Jelenleg öt ilyen státusz azonosító van, ami egy integer egész szám

- **LogDate** ⇨ a megrendelés dátuma, DateTime adattípus, év, hónap, nap, óra formátumban

	OrId	OrData	AdId	Name	Address	Phone	Email	Status	LogDate
☐ Módosítás Másolás Törlés	1	[{"prId":20,"prSize":"45"}]	11	Rend Elek	8500 Balatonrendek, József Attila utca 9865.	+36705555444	rendelek@gmail.com	4	2023-04-29 21:42:11
☐ Módosítás Másolás Törlés	2	[{"prId": 1,"prSize": "35"},{"prId": 2,"prSize": "...	11	Berend Elek	8565, Balatonrendes, Rendes utca 5647.	+36302222888	berendelek@freemail.hu	4	2023-04-29 21:42:23
☐ Módosítás Másolás Törlés	3	[{"prId":5,"prSize":"25"},{"prId":16,"prSize":"35"...	12	Megrend Elek	9855, Herend, Megren-dűlő 6587.	+36206666444	megrendelek@citromail.hu	3	2023-04-29 21:42:36
☐ Módosítás Másolás Törlés	4	[{"prId": 1,"prSize": "35"},{"prId": 2,"prSize": "...	9	Kirend Elek	7845, Ókörítőfülpös, Rendelő köz 456.	+36701111222	kirendelek@teszt.hu	2	2023-04-29 21:41:26
☐ Módosítás Másolás Törlés	5	[{"prId": 1,"prSize": "35"},{"prId": 2,"prSize": "...	9	Elrend Elek	6523, Elrendelőháza, Petőfi utca 4562.	+36209999444	elrendelek@citromail.hu	2	2023-04-29 21:41:16
☐ Módosítás Másolás Törlés	10	[{"prId": 1,"prSize": "35"}]	8	Elrend Elek	6523, Elrendelőháza, Petőfi utca 4562.	+36705555555	elrendelek@citromail.hu	1	2023-04-29 21:39:18
☐ Módosítás Másolás Törlés	11	[{"prId": 1,"prSize": "35"},{"prId": 2,"prSize": "...	7	Kirend Elek	teszt utca 7	+36705555666	teszt@teszt.hu	1	2023-04-29 21:39:04
☐ Módosítás Másolás Törlés	12	[{"prId":17,"prSize":"35"}]	7	Elrend Elek	8500 Balatonrendek, József Attila utca 9865.	+36705555666	rendelek@gmail.com	1	2023-04-30 13:23:40
☐ Módosítás Másolás Törlés	13	[{"prId":16,"prSize":"35"},{"prId":3,"prSize":"35"...	6	Megrend Elek	6523, Elrendelőháza, Petőfi utca 4562.	+36208888444	elrendelek@citromail.hu	0	2023-04-29 21:38:43

6. ábra: az orders (rendelések) tábla adattartalmának képernyőképe

3. tábla - product (termék)

#	Név	Típus	Illesztés	Tulajdonságok	Nulla	Alapértelmezett	Megjegyzések	Extra	Művelet
1	**PrId**	int(11)			Nem	*Nincs*		AUTO_INCREMENT	Módosítás Eldobás Több
2	**PrName**	varchar(40)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
3	**PrOther**	varchar(80)	utf8mb4_hungarian_ci		Nem	*Nincs*			Módosítás Eldobás Több
4	**PrUrl**	mediumblob			Nem	*Nincs*			Módosítás Eldobás Több
5	**PrActive**	int(2)			Nem	*Nincs*			Módosítás Eldobás Több
6	**PrPrice**	int(6)			Nem	*Nincs*			Módosítás Eldobás Több

7. ábra: a product (termék) tábla képernyőképe

A product tábla szerkezete

- **PrId** ⇨ a termék (pizza) azonosító száma, amely egy integer egész szám, a tábla elsődleges kulcsa, auto increment tulajdonsággal. Új pizza felvételénél automatikusan generálódik

- **PrName** ⇨ a termék (pizza) elnevezése, varchar adattípus

- **PrOther** ⇨ a termék (pizza) alkotóelemei, azaz a feltétek nevei, varchar adattípus

- **PrUrl** ⇨ a termékek (pizzák) képei vannak itt tárolva mediumblob adattípussal. A képek 400 pixel x 400 pixel nagyságúakra vannak átméretezve, hogy ne foglaljanak nagy helyet. Továbbá át vannak kódolva base64-es bináris formátumra és frontend oldalon lettek beillesztve és eltárolva az adatbázisban. A képek konvertálásához egy online felületen elérhető Base64 Image nevű programot használtam, amely megtekinthető a https://www.base64-image.de/ webcímen

- **PrActive** ⇨ kétfajta aktivitási státusz van. A 0-ás és az 1-es. Ha rendelhető a pizza, akkor 1-es ha nem, akkor 0-ás aktivitási státuszban van.

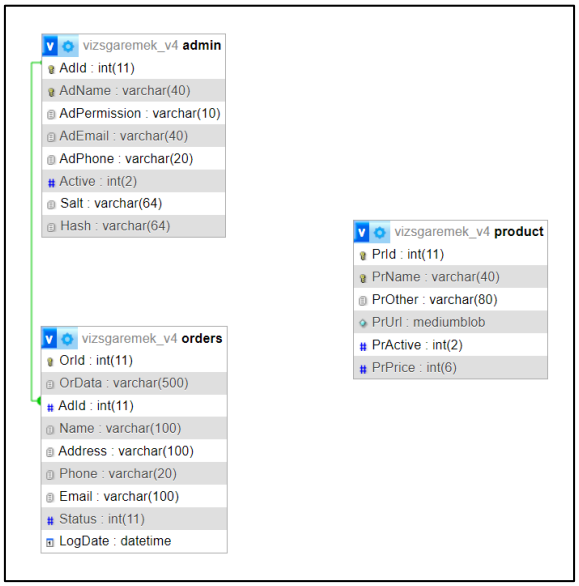
- **PrPrice** ⇨ a pizza ára, integer egész szám adattípussal

	PrId	PrName	PrOther	PrUrl	PrActive	PrPrice
Módosítás Másolás Törlés	1	Sonkás pizza	Sonka, sajt, oregánó	[BLOB - 128.4 KB]	1	1600
Módosítás Másolás Törlés	2	Szalámis pizza	szalámi, sajt	[BLOB - 122.1 KB]	1	1600
Módosítás Másolás Törlés	3	Tenger gyümölcsei	kagyló, rák, polip	[BLOB - 136.0 KB]	1	2000
Módosítás Másolás Törlés	4	Margherita	Paradicsomszósz, sajt	[BLOB - 128.7 KB]	1	2040
Módosítás Másolás Törlés	5	Prosciutto	Paradicsomszósz, sajt, sonka	[BLOB - 130.0 KB]	1	2400
Módosítás Másolás Törlés	6	Funghi	Paradicsomszósz, sajt, gomba	[BLOB - 134.6 KB]	1	2400
Módosítás Másolás Törlés	7	Tonno	Paradicsomszósz, sajt, tonhal, lilahagyma, olivabo...	[BLOB - 128.4 KB]	1	2540
Módosítás Másolás Törlés	8	Quattro Formaggi	Paradicsomszósz, négyféle sajt: trappista, füstölt...	[BLOB - 128.8 KB]	0	2540
Módosítás Másolás Törlés	9	Quattro Stangioni	Paradicsomszósz, sajt, bolognai ragu, gomba, sonka...	[BLOB - 132.5 KB]	1	2540
Módosítás Másolás Törlés	10	Hawaii	Paradicsomszósz, sajt, sonka, ananász	[BLOB - 125.5 KB]	1	2540
Módosítás Másolás Törlés	11	Giulia	Tejföl, sajt, tarja, szalámi, hagyma, bacon	[BLOB - 127.7 KB]	1	2540
Módosítás Másolás Törlés	12	Vegetariana	Paradicsomszósz, sajt, brokkoli, répa, gomba, para...	[BLOB - 135.2 KB]	1	2540
Módosítás Másolás Törlés	13	Grandiosa	Paradicsomszósz, sajt, szalámi, sonka, füstölt tar...	[BLOB - 142.4 KB]	0	2640
Módosítás Másolás Törlés	14	Bolognese	Bolognai ragu, sajt	[BLOB - 128.2 KB]	1	2540
Módosítás Másolás Törlés	15	Greek	Kapros tejföl, sajt, gyros hús, olívabogyó, paradi...	[BLOB - 127.3 KB]	1	2540
Módosítás Másolás Törlés	16	Ungherese	Paradicsomszósz, sajt, szalámi, füstölt tarja, pap...	[BLOB - 141.5 KB]	1	2540
Módosítás Másolás Törlés	17	Mais	Paradicsomszósz, sajt, sonka, szalámi, kukorica	[BLOB - 140.1 KB]	1	2540
Módosítás Másolás Törlés	18	Carbonara	Fokhagymás-tejszínes alap,sajt,füstölt sajt, bacon	[BLOB - 131.4 KB]	1	2540
Módosítás Másolás Törlés	19	Songoku	Paradicsomszósz, sajt, sonka, gomba, kukorica	[BLOB - 130.7 KB]	1	2640
Módosítás Másolás Törlés	20	Frutti di Mare	Paradicsomszósz, fokhagyma, sajt, tenger gyümölcse...	[BLOB - 136.0 KB]	0	2740

8. ábra: a product (termékek) tábla adattartalma képernyőképpel

A vizsgaremek_v4 nevű pizza adatbázis táblái közötti kapcsolat

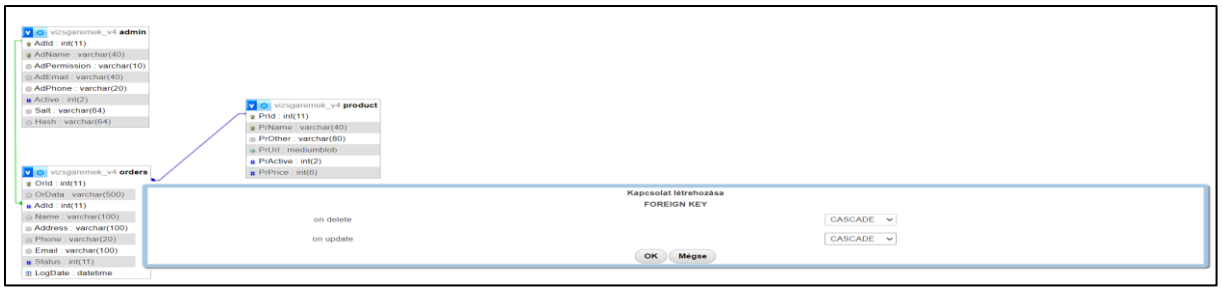
A phpMyadmin Tervező menüpontjában megnézhetjük az adatbázis táblái közötti kapcsolatot. Az ábra jól szemlélteti, hogy az admin tábla az AdId elnevezésű, elsődleges kulccsal ellátott mezője az orders tábla ugyanolyan nevű mezőjéhez kapcsolódik. Erre így azért volt szükség, mert egy megrendeléshez több termék is kapcsolódhat, ezért ez egy egy a többhöz típusú kapcsolat. Ugyanúgy, mint az admin és az orders táblában is, egy adminhoz több rendelés is kapcsolódhat.



9. ábra: a vizsgaremek_v4 elnevezésű adatbázis táblái közötti kapcsolat

A CASCADE művelet

A CASCDE művelet azt jelenti, hogy az eszközölt változtatások átterjednek minden olyan táblára, amelyekben az adott rekordot hivatkozó kulcsok vannak. Ha két tábla egymáshoz idegen kulcsok segítségével kapcsolódik és egy rekordot törlünk, beszúrunk, vagy frissítünk, akkor ez a művelet az összes kapcsolódó táblában is végbemegy. Általában ezt a műveletet akkor alkalmazzuk, amikor ezen kapcsolatok fontosak az adatintegritás és a hivatkozások egységessége szempontjából. Viszont, ha ez a CASCADE művelet rosszul van használva, az adatvesztéshez és nem kívánt összeomláshoz vezethet. Így ezért az javasolt, hogy használatát alaposan fontoljuk meg!



10. ábra: a CASCADE művelet képernyőképe

Backend bemutatása (Kolyankovszky Gusztáv)

Backend fejlesztéshez is számtalan eszköz áll rendelkezésre. Döntés előtt érdemes felmérni a használathoz szükséges környezetet, bár ezen a területen nagy az átjárhatóság.

Mire van szükség a backend fejlesztéshez? Fejlesztőeszköz, fejlesztő program, tesztelő program és mind három a feladatnak megfelelő ismerete.

Fejlesztő eszköz a Microsoft Visual Studio 2022

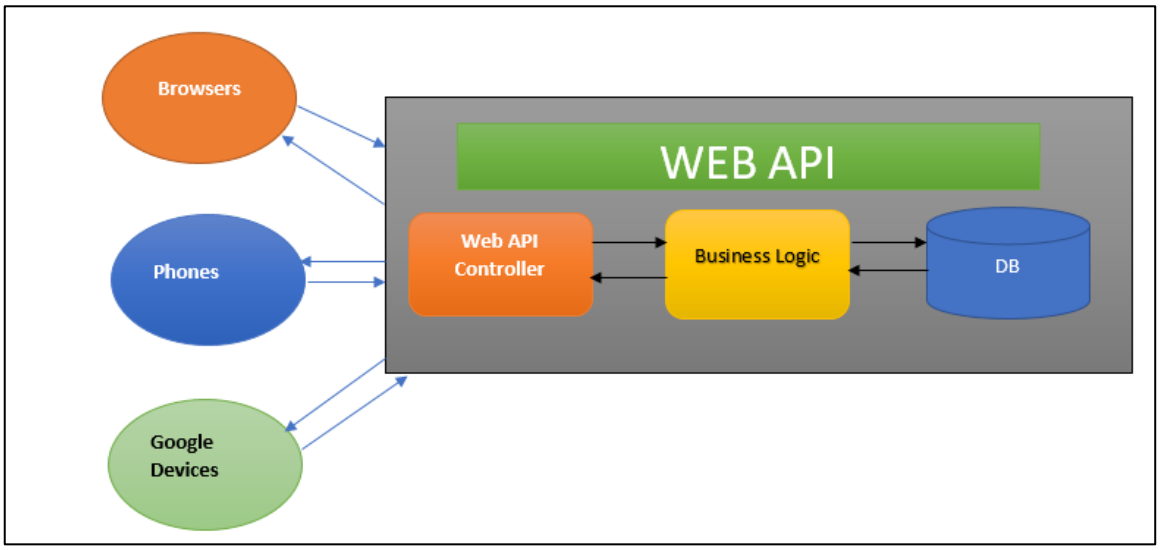
A Microsoft 1997-ben adta ki a több programozási nyelvet tartalmazó fejlesztő eszközt, a neve Microsoft Visual Studio 97. A mai napig 13 kiadás és 17 verzió készült el. Az évek során programozási nyelvekkel bővült. 2002-ben került fel a programozási nyelvek közé a C#. Szintén ekkor mutatták be a .NET keretrendszert. A C# a .NET keretrendszer részeként kifejlesztett objektum központú programozási nyelv, a fejlesztést a Microsoft végezte. A nyelv alapjait a C++ és a JAVA nyelvek adták. Fontos megjegyezni, hogy a .NET jelenleg több mint 40 programozási nyelvet támogat. A .NET keretrendszert platform független alkalmazásfejlesztésre szánták, az elképzelésnek célja volt a gyors fejlesztés lehetősége.

A program fejlesztéséhez használt alkalmazás

ASP.NET Core Web API

Tartalma: C#, Linux, macOS, Windows, Cloud Service Web, WebAPI

Az elnevezés több rövidítést és szóképet is tartalmaz. A rövid névben igyekeznek mindent felsorolni, amit tartalmazhat a program. A lecsót hasonló módon próbálnám leírni: PapParHagySzalKolOlSo, hmm lássuk be ezen a területen nem hétköznapi emberek dolgoznak.



Forrás: https://www.c-sharpcorner.com/article/asp-net-core-5-0-web-api/ időpont 2023.04.25.

Kibontva az ASP.NET Core (ASP = Active Server Pages, .NET = Network Enabled Technologies) egy klasszikus technológia az ASP.NET utódja web programozáshoz. Egy nyílt forráskódú, moduláris szerver oldali alkalmazás keretrendszere. Teljesíti a .NET elvárásokat, programozási nyelv független és .NET osztálykönyvtárral rendelkezik, több platformon működik. Alkalmas: Windows, Linux, macOS rendszerekhez.

Web API jelentése, Webes Application Programming Interface.

Feladata a felhasználó oldali webes applikációt összekötni például egy web serverrel az adatokhoz történő hozzáférés miatt. A Web API-t elsősorban adatbázis műveletekhez alkalmazzák. Adatbázis művelet az adat létrehozása CREATE, adat kiolvasása READ, adat szerkesztése UPDATE, adat törlése DELETE. A kezdőbetűket összeolvasva CRUD műveleteknek hívják őket. A kommunikációhoz http protokolt használ, például create = HTTP POST, read = HTTP GET, edit = HTTP PUT, delete = HTTP DELETE, a kérésekre a válasz JSON formátumban érkezik.

Forrás: https://www.c-sharpcorner.com/article/asp-net-core-5-0-web-api/ időpont 2023.04.25.

Az alkalmazást két keretrendszer alkotja a Microsoft AspNetCore.App és a Microsoft.NETCore.App.

Emellett feladattól függően szükség van kiegészítő csomagokra, amit a jelenleg feladathoz használtam, ezeket felsorolom, részletekbe nem megyek bele.

Microsoft.EntityFrameworkCore(5.0.17)

Microsoft.EntityFrameworkCore.Design(5.0.17)

Microsoft.EntityFrameworkCore.Tools(5.0.17)

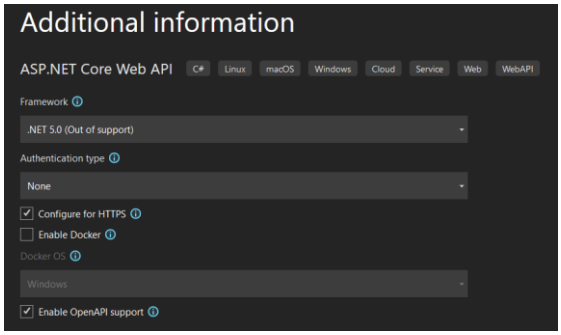
Microsoft.VisualStudio.Azure.Containers.Tools.Targets(1.17.0)

MySql.EntityFrameworkCore(5.0.17)

Swashbuckle.AspNetCore(5.6.3)

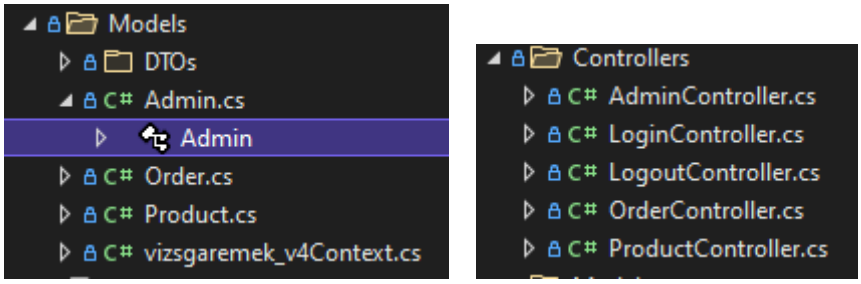
Kép a mellékletben.

A keretrendszerek verziói követik a haladás ütemét. Fejlesztés alkalmával érdemes a hosszú távon támogatott verziót választani, bár ez azt is jelenti, hogy az elején új dolgokat kell megismernünk és régi megszokásokat elfelejtenünk. Jelen esetben egy ismert verziót választottam, ami megfelel a feladat megvalósításához, Framework .NET 5.0.



Saját képernyőkép

A fejlesztő eszköz alap szerkezetet generál számunkra, amit célunknak megfelelően módosíthatunk, kibővíthetünk. Jelen esetben két plusz könyvtár Models, ide kerülnek az osztályok és a Controllers, itt kerülnek kialakításra a végpontok.



Saját képernyőkép

A végpontokat említettem. Korábban írtam CRUD műveletekről. A végpontok ezeket a műveleteket hajtják végre. Minden kontroller a szükséges műveletek halmazát tartalmazza. Nem biztos, hogy minden műveletre szükség van. Lehet olyan adatunk, amit nem akarunk törölni, de létrehozni, szerkeszteni igen? Ebben az esetben csak a CRU műveletekre van szükségünk. Adat, amit nem akarunk kiírni? Nehéz elképzelni, de lehet ilyen feladat. Mire van szükség, azt a tervezésénél határozzuk meg.

Az egyik fájlban történt változtatást kiemelem. A Program.cs fájlba kerül két biztonsági függvény, melyek a jelszavas azonosításhoz hasznosak.

GenerateSalt. A függvény meghatározott karakter készletből véletlenszerűen generál egy kriptográfiailag erős karakter sorozatot. A visszatérő érték 64 hosszú karakterlánc.

```
static int SaltLenght = 64;

public static Dictionary<string, Admin> LoggedInUsers = new Dictionary<string, Admin>();
1 reference
public static string GenerateSalt()
{
    Random rnd = new Random();
    string karakterek = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    string salt = "";
    for (int i = 0; i < SaltLenght; i++)
    {
        salt += karakterek[rnd.Next(karakterek.Length)];
    }
    return salt;
}
```

Saját képernyőkép

CreateSHA256. A függvény SHA-256 hash hexadecimális értéket ad vissza a karakterlánchoz.

```
public static string CreateSHA256(string input)
{
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] data = sha256.ComputeHash(Encoding.UTF8.GetBytes(input));
        var sBuilder = new StringBuilder();
        for (int i = 0; i < data.Length; i++)
        {
            sBuilder.Append(data[i].ToString("x2"));
        }
        return sBuilder.ToString();
    }
}
```

Saját képernyőkép

Tesztelő program

A backend fejlesztés folyamata, hogy az elkészített részek működését teszteljük. Mint tudjuk a backend nem a megjelenésért felel, a frontendről érkező kérések fogadása és továbbítása az adatbázis felé, ahonnan megkapja a kért adatokat és megfelelő formában vissza adja a frontend résznek.

A frontend résznek nincs szüksége emberi szemmel és értelemmel olvasható adatokra, hanem JSON formátumban vár válaszokat. Ez is olvasható gyakorlott szemmel, nézzünk erre egy példát a kiindulási alapra.

```
"/Admin": {
    "get": {
        "tags": [
            "Admin"
        ],
        "responses": {
            "200": {
                "description": "Success"
            }
        }
    },
```

Saját képernyőkép

Nézzük meg Swagger programmal, interaktív állapotban, kattintásra készen a mellékletben.

Swegger/OpenAPI program a kimenet megjelenítésére szolgál, ez egy nyílt forráskódú szoftver API fejlesztők részére. RESTful Web API tervezés, készítés, dokumentálás feladatokra.

Teszteléshez meg van szólítva a Startup.cs fájlban.

```
app.UseCors(options => options.AllowAnyOrigin().AllowAnyMethod().AllowAnyHeader());
if (env.IsDevelopment())
{
    app.UseDeveloperExceptionPage();
    app.UseSwagger();
    app.UseSwaggerUI(c => c.SwaggerEndpoint("/swagger/v1/swagger.json", "FadokoBackendV4 v1"));
}
```

Saját képernyőkép

Amikor elindítjuk a programot a fejlesztő eszközön, akkor a Swagger segítségével tudjuk emulálni a program működését. A három komponensből most elég az adatbázis és a backend, a frontend a megfelelő adatokat várja. Szeretném megtudni, hogy ki az első Admin az adatbázisban, feltételezem, az azonosító száma az 1-es.

Response body


```
{
  "adId": 1,
  "adName": "Fazekas Réka",
  "adPermission": "9",
  "adEmail": "f@f.hu",
  "adPhone": "+3636",
  "active": 1
}
```


Response headers

```
content-type: application/json; charset=utf-8
date: Sat29 Apr 2023 15:52:20 GMT
server: Kestrel
```

Saját képernyőkép

1. - A vizsgaremek - A frontend bemutatása (Fazekas Réka)

Az alkalmazás frontendje a Vue.js JavaScript könyvtár illetve a Bootstrap CSS keretrendszer segítségével készült, főleg asztali gépen való használatra, de reszponzív módon lett kialakítva, hogy bármilyen internetelléréssel rendelkező készüléken használható legyen (különböző méretű laptopok, tabletek illetve mobiltelefonok.

A frontend megtervezését, kialakítását és fejlesztői tesztelését Fazekas Réka végezte.

1.1 – Technológiai háttér

A frontend oldali alkalmazás kialakításához változatos technológiákat használtam fel, a szoftverfejlesztők aktuális eszköztárából válogatva.

1.1.1 – Szoftverarchitektúra

Alkalmazásunk a két jellemző szoftverfelépítést ötvözi, a „vékony szerver, vastag kliens” (thin server, thick client) illetve a „vastag szerver, vékony kliens” architektúra elemei is megtalálhatóak benne. A szerver felel az adatbázissal való kapcsolattartásért, az adatok lekérdezéséért és mentéséért, az adatok feldolgozását, ellenőrzését, megjelenítését és a felhasználó interakciókat pedig a kliens oldali alkalmazás végzi. Ez a felépítés gyakori a modern webes alkalmazások tervezésénél.

Ez a típusú szoftverfelépítés a többihez képest összetettebb tervezést és fejlesztést igényelt, főleg az adatok biztonságának garantálása miatt, de a modern szoftverfejlesztésnek megfelelően fontos szempont volt a felhasználói élmény (UX/UI) követelményeinek figyelembe vétele is.

1.1.2 – Fejlesztői környezet

A frontend kialakítása a Visual Studio Code (VS Code) nevű, nyílt forráskódú fejlesztői környezetben (IDE) történt. Ezt a programot a Microsoft fejlesztette ki, és jelenleg az egyik legelterjedtebb IDE a webfejlesztők körében, mivel ingyenesen elérhető, jó támogatottságú és könnyedén testreszabható a fejlesztői munkát nagyban segítő bővítményekkel. A frontend fejlesztése alatt én a Tabnine AI Autocomplete, a Vue Language Features (Volar), a Git Graph, GitLens, Pretty Formatter illetve a SonarLint ingyenesen elérhető bővítményeket használtam.

1.1.3 – Verziókezelés

A fejlesztés különböző állapotainak, változásainak lekövetésre git verziókezelő rendszert használtunk, a Visual Studio Code és a GitHub, valamint a GitHub és a GitLab felületek intgerációját kihasználva.

Mind a VS Code, mind a Gitlab felületére be tudunk jelentkezni a GitHub accountunkal, majd egy gyors hitelesítést követően máris rendelkezésünkre állnak a GitHubon elérhető tárolóink, valamint egy egyszerűen kezelhető CI/CD (Continous Integration/Continous Deployment) platform, amelyen többek közt könnyen megvalósítható az alkalmazás kódszintű automata tesztelése.

1.1.4 – Node.js és rá épülő tartalmak

A Vue.js használatához szükség volt a megfelelő környezet kialakítására, ami magába foglalta a Node.js, az NPM, és különböző NPM-en keresztül telepíthető package-ek telepítésére is.

Node.js

Nyílt forráskódú, JavaScript nyelven alapuló programozási környezet, szerveroldali megoldásokra. Aszinkron programozási modelleket használ ki, és a Google Chrome böngészőben támogatott V8 JavaScript motor működteti. Nagy előnye, hogy a fejlesztők készíthetnek és használhatnak saját modulokat, amelyek lehetővé teszik egyes kódok újrafelhasználását így a kódduplikáció elkerülését.

NPM

Az NPM (Node Package Manager) a Node.js által használt csomag- illetve modulkezelő rendszer. Ez teszi lehetővé, hogy a fejlesztők egyszerűen és könnyen telepítsék, frissítsék és kezeljék a Node.js-el kapcsolatos függőségeket (dependency-k).

Jelen vizsgamunkában a Bootstrap legfrissebb elérhető verziója, a Bootstrap ikongyűjteménye, az axios illetve az sha256 node package-ek lettek telepítve.

Bootstrap

A bootsrtap egy nyílt forráskódú frontend keretrendszer, az egyik legszélesebb körben webfejlesztésben használt CSS illetve JS framework. Előre elkészített, könnyedén testreszabható elemeket tartalmaz, amelynek segítségével egy kezdő fejlesztő is

kiemelkedően gyorsan és hatékonyan tud reszponzív weboldalakat és alkalmazásokat fejleszteni, amelyek így mobilra, tabletre, laptopra, asztali gépre egyaránt optimalizálva vannak.

Bár létezik a Vue.js-hez optimalizált verzió is, jelen projektben az NPM-en keresztül automatikusan a Bootstrap legfrissebb elérhető stabil verzióját vettük használatba, hogy minél tovább támogatott legyen az oldalhoz elkészített design.

Axios

Olyan JS könyvtár, ami könnyen kezelhetővé teszi a szerverrel való kommunikációt, azaz a HTTP kérések (get, post, put, delete) és válaszok kezelést, aszinkron módon is.

SHA-256

Alapvetően egy hash algoritmus, amelyet a Secure Hash Algorithm részeként fejlesztettek ki nemzetbiztonsági célokra. Ebben a projektben a felhasználók jelszavának titkosítására használtuk fel, backend és frontend oldalon is, kétszeres kódolással, ami gyakorlatilag felfejthetetlen titkosítást biztosít így jelentősen csökkentve a adatvédelmi kockázatot.

A frontend alapját adó Vue.js-t is az NPM segítségével telepítettem és konfiguráltam.

1.1.5 – Vue.js

A Vue.js egy JavaScripten alapuló, könnyen megtanulható és használható keretrendszer jellegű könyvtár, ami 2014-ben jelent meg először. Alapvetően egyszerű, de látványos, felhasználóbarát alkalmazások fejlesztésére szolgál.

A Vue.js egyik előnye a virtuális DOM-ot (Document Object Model) használ, azaz a változó frissülésével egyidőben az igazi DOM is módosul, ami sokkal gyorsabb teljesítményt jelent pl egy jQuery-t használó alkalmazással szemben.

Architektúrája az MVC-hez (Model-View-Controller) hasonló, az adatok, a metódusok és a html kód könnyen szétválasztható, áttekinthető. Az adatok még teljesebb szeparálására a Vuex állapotkezelő könyvtár ad módot, amelynek három fő része az állapot (State), a műveletek (Actions) illetve a mutációk (Mutations), amelyek mind különböző állapotokat meghatározó adatok, egy helyre gyűjtve.

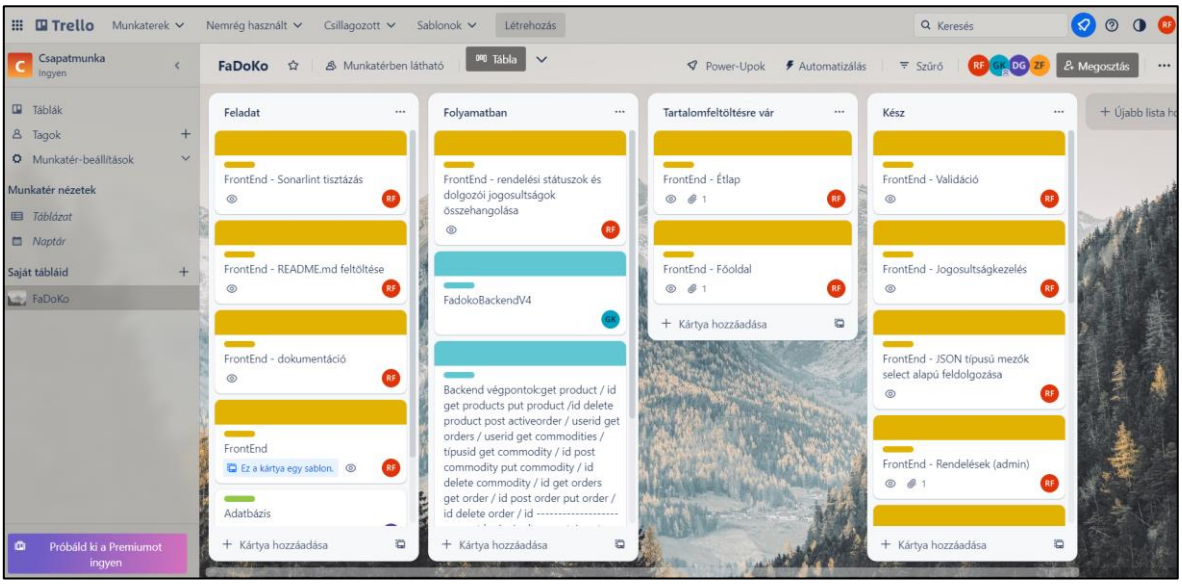
A Vue.js alapelemei a komponensek. Ezek olyan kódrészleteket tartalmaznak, amelyek az oldalon ismétlődnek, és a kiszervezésnek köszönhetően könnyedén újrafelhasználhatóak, így rövidítve és egyszerűsítve a kód egészét.

1.1.6 – Munkafolyamat lépéseinek vezetése

Mint a legtöbb modern fejlesztői feladat esetében, a feladatok és az elvégzett munka nyilvántartásához webes feladatkezelő rendszert használtunk. A választás a Trello nevű feladatkezelőre esett, amely ingyenesen is elérhető, és melyben az egy projekten dolgozó felhasználók megosztott táblákra helyezhetik el feladataikat, kártyák segítségével.

Az egyes kártyák címkézhetőek, kommentelhetőek, és fájlokat is tudunk csatolni hozzájuk, a frontend esetében például az egyes oldalakhoz illetve részletekhez kapcsolódó design vázlatokat csatoltam a kártyákhoz.

Módszertanilag az agilis megközelítéshez álltunk a legközelebb, amelyből a változásokra és az új igényekre történő rugalmas reagálást, a szinte állandó kommunikációt és feedbacket és a kockázatok korai azonosítását és kezelését is sikerült megvalósítani. A feladatkártyák megjelenítése a Kanban rendszert idézte.



1. ábra – A vizsgaremek projektjéhez tartozó Trello tábla, a fejlesztés egy adott szakaszában

1.2 – Használati útmutató

1.2.1 – Alapkoncepció

Az alkalmazás alapötlete a dolgozat korábbi szakaszában is tárgyaltan egy olyan felület megvalósítása, amelynek egy vendéglátóipari egység, konkrétan egy pizzéria vendégei, valamint dolgozói lesznek a felhasználói.

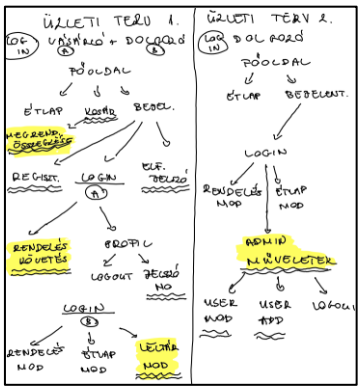
A weboldal megjelenését mobil, tablet, és laptop/asztali számítógép megjelenítésének megfelelően is optimalizáltam. Ennek érdekében a design tervek eleve úgy készültek, hogy Bootstrap 5.3-as keretrendszer segítségével könnyen megvalósíthatóak legyenek.

A frontend logikáját Vue.js JS könyvtár alapon alakítottam ki, így az felel az étlap tartalmának beolvasásáért, a bejelentkezésért, az oldalak közti váltásért és még sok minden másért is.

1.2.2 – Üzleti logika terve

Hosszasabb tervezést követően végül arra jutottunk, hogy az alkalmazást 2 alap felhasználási módra készítjük fel: egyrészről egy weboldal, ahol a vásárlók/vendégek tájékozódni tudnak az étterem alapinformációval kapcsolatban, mint az elhelyezkedés, az elérhetőségek, illetve az étlap.

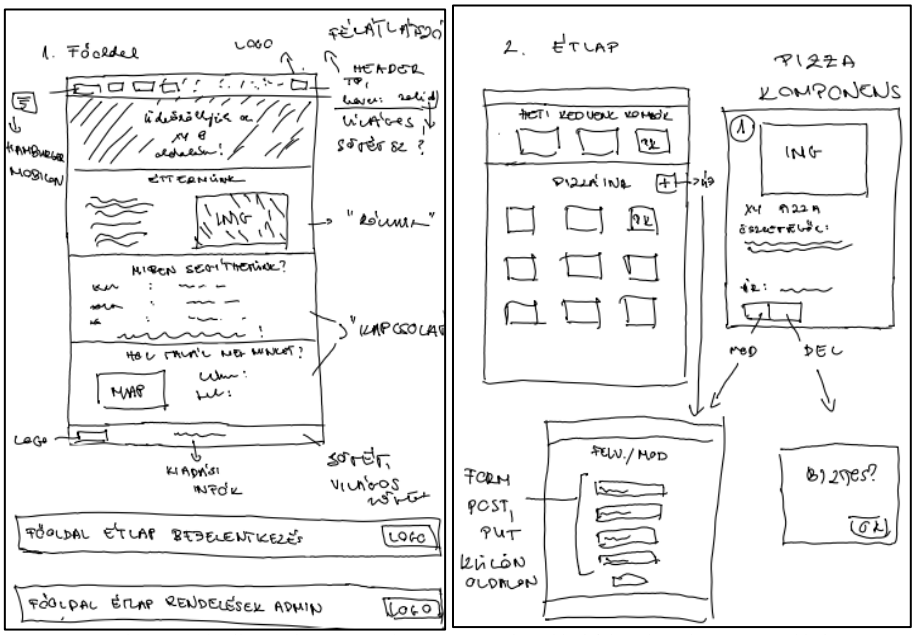
A másik felhasználási mód az étlap szerkesztését, a rendelések megtekintését és szerkesztését illetve a felhasználók megtekintését és szerkesztését teszi lehetővé jogosultságfüggően.



2. ábra – Üzleti terv kialakulása. A vizsgaremek végül a 2. üzleti tervet követve készült el

1.2.1 – Design-tervek

A tervezett alkalmazás összetettsége nem tette indokolttá grafikus tervezőprogramok használatát, így a szabadkezes vázlatok készültek, figyelembe véve hogy a terveket a Bootstrap 5.3-as verziójában található elemekkel könnyen fel lehessen építeni.



3. ábra – Szabadkezes design tervek, Bootstrap elemekből felépíthető komponensekkel

1.2.1 – Az alkalmazás leírása

BEJELENTKEZÉS NÉLKÜL ELÉRHETŐ OLDALAK

Az alkalmazás elsődleges elérési módja. Ennek megtekintéséhez nincs szükség bejelentkezésre, bárki szabadon elérheti és tájékozódhat róla az elérési út ismeretében. Elsődleges célja, hogy a felhasználók megtalálják az éttermet illetve ki tudják választani a megrendelni kívánt termékeket az oldalról. A főoldal és az étlap érhető el alapvetően ebben a formában, valamint az admin felület bejelentkezési ablaka.

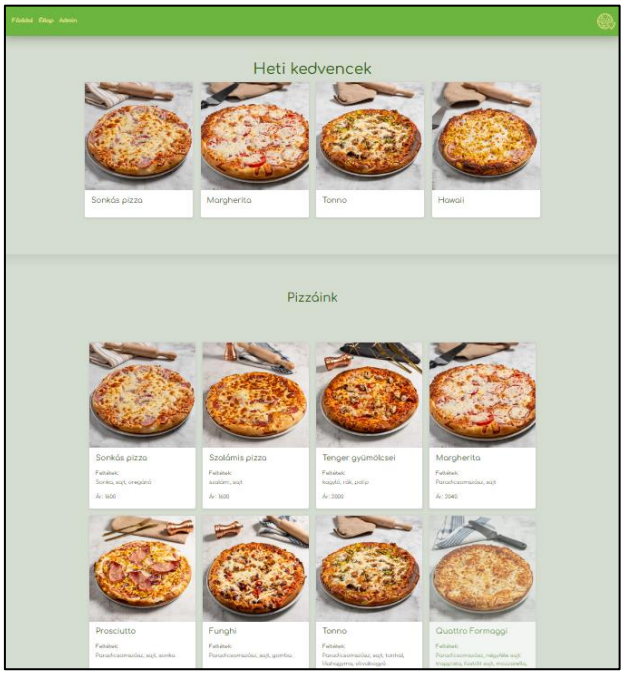
Főoldal

Erre az oldalra érkezik a felhasználó az alkalmazás indításakor illetve a kijelentkezést követően. Innen tud tájékozódni a hely jellegzetességeiről, elérhetőségeiről, elhelyezkedéséről. Az oldalt marketingszövegekkel töltöttem fel.

Az oldalt minden aloldalon ismétlődő fejléc és lábléc foglalja keretbe. Utóbbiban az oldal szerkesztőiről és az alkalmazás fejlesztésének idejéről helyeztem el információt, az előbbi pedig a mindenhonnan könnyen elérhető navigációt segíti. A felhasználó-interakciót egy egyszerű hover funkció könnyíti, a kurzor pozíciójának megfelelően a fejléc változtatja az áttetszőségét: lejjebb tekerve láttatni engedi a mögé kerülő tartalmat, amikor a kurzor a navigációs linkek közelébe kerül, akkor viszont elveszíti átlátszóságát, a menüpontok olvashatóságát növelve. Szerkezetileg a fejléc és a lábléc Bootsrtap elemek, az oldal blokkjai pedig Vue komponensek, ez lehetővé teszi a gyors variálást, ha szeretnénk egy blokkot kivenni vagy hozzátenni, vagy csak a sorrendjükön módosítani.

Étlap

Bejelentkezés nélkül az étlap funkciója a pizzák összetételéről, áráról illetve elérhetőségéről történő tájékoztatás.



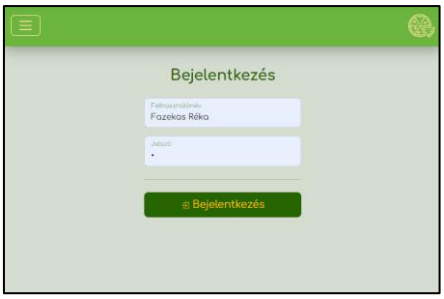
A nem elérhető pizzákat enyhe áttetszőséggel jelöltem, valamint szöveges visszajelzést is nyújtok a felhasználónak, hogy a tárgyban forgó termék nem elérhető aktuálisan.

Az Étlap oldal 2 fő blokkból áll, a heti kedvenceket bemutató blokkból, melyek a példában véletlenszerűen kerültek kiválasztásra, de tartalma könnyen testreszabható a frontend felületen a pizzák betöltésekor. A másik blokk a fő tartalom, a pizzák felsorolása.

Mind a kedvenc pizzák, mind az étlapot felépítő pizzák megjelenítéséért egy-egy Vue komponens elem felel, dinamikusan feltöltve adatokkal. Az egyes komponesekhez a Bootstrap card elemét használtam, a reszponzív oldalmegjelenítést pedig a grid rendszer tette lehetővé, amelyben oldalszélességtől függően jeleik meg 1, 2, 3 vagy 4 komponens egy sorban.

Admin (bejelentkezés)

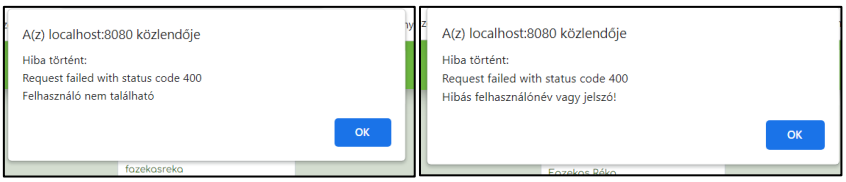
Ez az oldal már azoknak a felhasználóknak van fenntartva, akik rendelkeznek az oldalhoz belépési azonosítóval, ami esetünkben felhasználónév és jelszó.



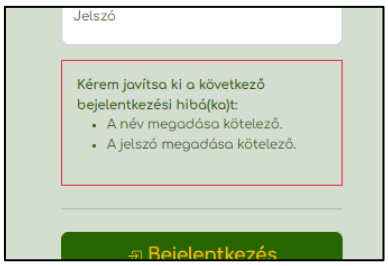
A bejelentkezés oldal tablet nézetben, színvilágban testreszabott Bootstrap hamburgermenüvel

Szerkezet szempontjából a bejelentkezés funkcióját, a kijelentkezést és az felhasználók kezelését ugyanaz a felület végzi, aminek a tartalmát dinamikusan töltöm be, jogosultságfüggően.

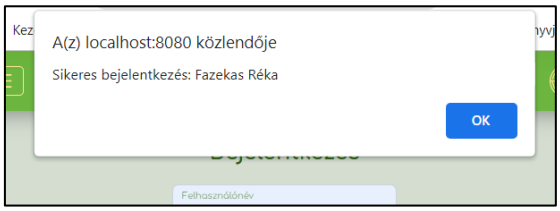
A bejelentkezéssel kapcsolatos hibaüzeneteket JS alert üzenetek formájában hozza a felhasználó tudomására az oldal, amely tartalmaz a hiba azonosítására alkalmas fejlesztői üzenetet és felhasználóbarát tájékoztatást is (nem létező felhasználó, hibás felhasználónév vagy jelszó).



A mezők validálása a Vue formvalidáló beépített funkciójával történik. A bejelentkezés oldalon csak azt ellenőrzi az alkalmazás, hogy a felhasználó mindkét mezőt kitöltötte-e. Ha nem, azt a beviteli mezők alatt, pirossal kijelölve, szöveges formában jelenítettem meg.



A helyes adatok megadását követően a sikeres bejelentkezésről is kap feedbacket a user, hogy milyen felhasználóval lépett be.



Az üzenet elfogadását követően átirányítom a felhasználót a főoldalra, ahonnan kiválaszthatja hogy melyik aloldalt szeretné megtekinteni illetve használatba venni a rendelkezésére állók közül.

A bejelentkezés folyamata a felhasználónév és az sha256-os kódolással titkosított jelszó ellenőrzésével történik, sikeres esetben beállításra kerül a felhasználónév, a felhasználóazonosító és a felhasználó jogosultsági szintje a Vuex változók használatával.

A jogosultságszintek a következőképpen változtatják az oldalak funkcióit illetve működését.

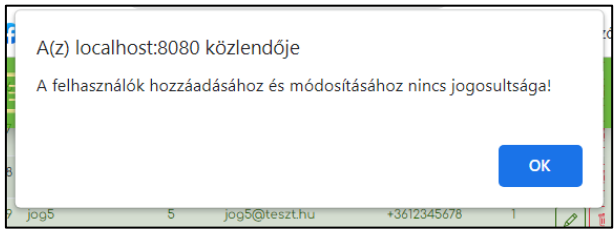
	4, 5, 6-os szintű jogosultság	7-es szintű jogosultság	8-as szintű jogosultság	9-es szintű jogosultság
Étlap	megtekintés	megtekintés	szerkesztés	szerkesztés
Rendelések	megtekintés	szerkesztés	szerkesztés	
Admin	saját adatok megtekintése + kijelentkezés	megtekintés + kijelentkezés	megtekintés + kijelentkezés	szerkesztés + kijelentkezés

1.Számú táblázat – a dolgozói jogosultságok és a velük összefüggő elérhető funkciók

BEJELENTKEZÉS UTÁN ELÉRHETŐ OLDALAK, FUNKCIÓK

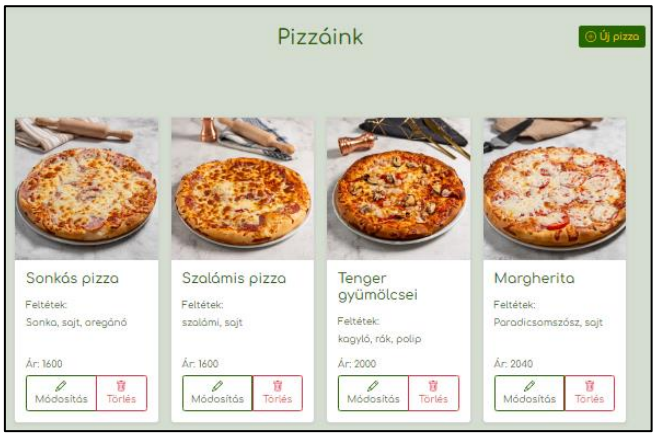
Az oldalakról általánosan elmondható, hogy mindegyik megjelenése függ a bejelentkezési állapottól és a jogosultsági szinttől. Amennyiben a bejelentkezési vizsgálat hibát jelez, a bejelentkezéssel elérhető oldalak (Rendelések, Pizzák hozzáadása, Pizzák módosítása) ha felhasználó épp rajtuk tartózkodik mikor véget ér a munkamenet, a következő üzenetet jelenítik meg egységesen.

A jogosultságoktól függően olyan oldalelemek jelennek meg, amik alacsonyabb szinttel nem elérhetőek, illetve ha olyan műveletet próbál végezni a felhasználó, akkor „JS ALERT” üzenetben kap visszajelzést a jogosultság hiányáról.

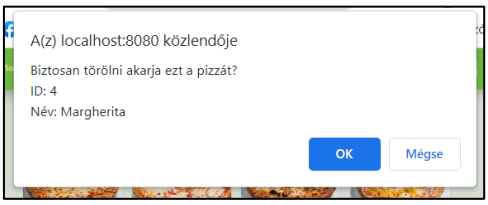


Étlap

Az Étlap oldal az 1.es táblázatnak megfelelően az alacsonyabb jogosultságú felhasználóknak nem mutat változást a bejelentkezés nélkül elérhető oldalképhez képest. 8-as jogosultsági szinttől lehetősége válik a felhasználónak a hozzáadási, módosítási és törlési funkciókra.



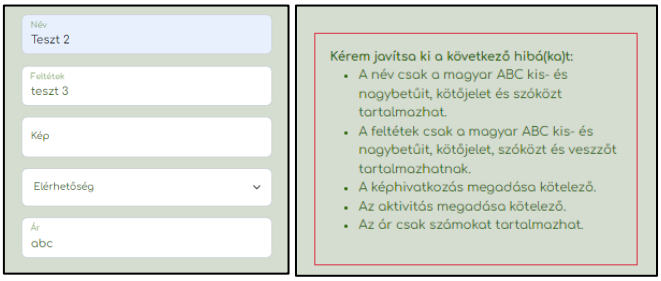
Ezek közül a törlés navigáció nélkül történik, egy confirm típusú JS megerősítéssel.



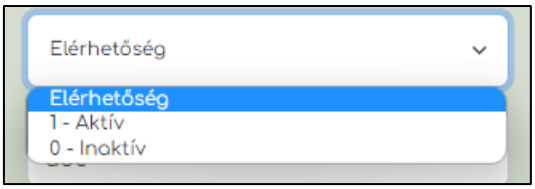
Pizzák hozzáadása és módosítása

Az „Új pizza” illetve a „Módosítás” gombok csak 8-as szint felett elérhető további oldalakra irányítanak át. A 2 oldal szerkezetét tekintve nagyon hasonló, fő különbségük az oldal címén kívül hogy a hozzáadás oldalon üres beviteli mezők szerepelnek oldalbetöltéskor, illetve felvitelkor ellenőrzésre kerül a bevitt név egyedisége. Módosítás oldalon ezzel szemben a kiválasztott pizza adataival előre fel vannak töltve a beviteli mezők, meggyorsítva a műveletet, és a módosítás mentésekor egy újabb megerősítés funkcióval ellenőrzöm, hogy a felhasználó biztosan végzett-e minden módosítással.

Az adatok ellenőrzése nagyobbrészt kétféle ellenőrzésen is átmegy, megvizsgálom az adat meglétét illetve formai helyességét is, amiről a hibának megfelelő visszajelzést kap a felhasználó.



Az elérhetőség, ami az adatbázisban számkóddal van tárolva, ezen a felületen kap szöveges címkét, egy select elemben.

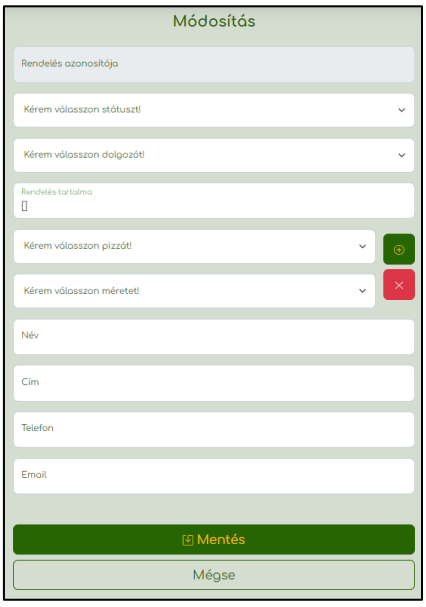


Ez a megoldás az olvashatóságon túl biztosítja, hogy csak az elvárt szerkezetnek megfelelő adat kerüljön felvitelre.

Rendelések

Az oldal minden bejelentkezett felhasználó számára megtekinthető, de csak 7-es jogosultságszinttől felfelé szerkeszthető a megtekintett adatok. Az oldal főbb részei a rendelések dinamikus táblázata, fordított sorrendbe rendezve, hogy a legfrissebb rendelés szerepeljen az oldal tetején.

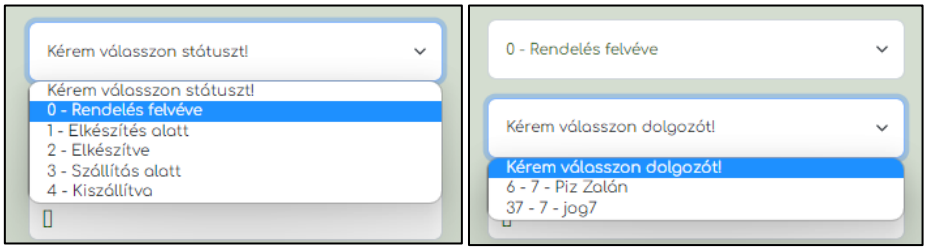
A másik blokk, ami az oldal fő logikáját tartalmazza. A rendelések felviteléért és állapotkezeléséért felelős „form”, amely az adatbázis számára értelmezhető módon, a frontenden kialakított logikával gyűjti össze a rendelések adatait.



A Rendeléskezelő form elemei

Rendelés azonosítója mező manuálisan nem módosítható, mivel automatikusan generálódik. azonban a rendelés azonosításakor a szemléletesebb megjelenítésért ez a mező is feltöltésre kerül a meglévő azonosítóval.

Státusz és dolgozóválasztás mezőnek a tartalma szorosan összefügg. A rendelések státusza a frontenden kap olvasható címkét.



Az adott rendelési státusz kiválasztásától függően, dinamikusan töltődik fel az adott státuszhoz kapcsolódó jogosultsági szintű illetve munkakörű dolgozók listája, ezzel biztosítom az adott rendeléssel való feladatot a megfelelő embernek lehessen dedikálni.

Rendelés tartalma

A rendelés tartalma JSON formátumú „stringben” tárolódik le, illetve kerül megjelenítésre a táblázatban. Mivel azonban ez átlag felhasználónak nem elég könnyen olvasható illetve írható formátum, hogy megtartsuk az adatok formai és tartalmi helyességét, egy olyan logikát dolgoztam ki, a pizzák dinamikusan betöltött listájából kiválasztott pizza és a pizza lehetséges méretének listájából kiválasztása után gombnyomással feldolgozza és a megfelelő formátumban menti az adatokat a rendelés tartalma mezőbe. A törlés gomb segítségével lehet alaphelyzetbe állítani az említett 2 listát.

A „select” mezőkkel a pizzák hozzáadása és módosítása oldalakhoz hasonlóan azt biztosítottam, hogy csak megfelelő adatok legyenek továbbítva a szerver felé mentésre. A kötetlen bevitelű mezők, a név, a telefon és az email „regex” kifejezésekkel lettek ellenőrizve, míg a címnél csak a mező kitöltöttségét ellenőriztem, számítva az adatok sokféleségére.

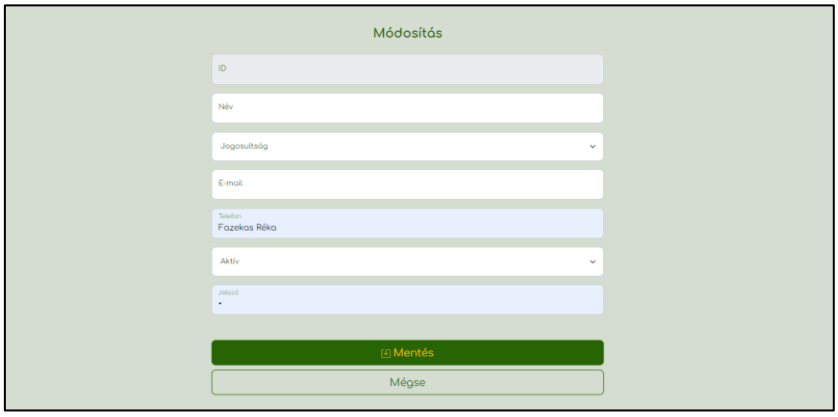
Admin (felhasználók kezelése)

Az admin oldal megjelenése és funkciói a leg sokszínűbbek. Ezt az oldalt minden jogosultsági szinten el lehet érni ugyan, de a tartalma a táblázatban is jelzett módokon a következőképp módosulhat.

6-os jogosultságszinttel bezárólag funkciója a bejelentkezett felhasználó adatainak és jogosultsági szintjének ellenőrzése, valamint az alkalmazásból való kijelentkezés.

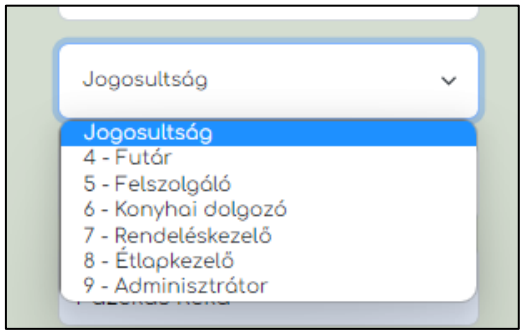
7-es szinttől az oldal kiegészül a regisztrált felhasználók listájával

Csak a 9-es jogosultsággal rendelkező felhasználók számára engedélyezett felhasználó módosításért felelő formmal.



A form logikája a Rendelések oldal módosításához hasonló kialakítást kapott illetve olyan logikát használ szerkezetileg, azaz szabad kezes és kötött választási lehetőségekkel feltöltött beviteli

mezőkből áll. Szerepel a már korábban felhasznált aktivitás mező, amely ebben az esetben a felhasználó érvényességét jelzi (az inaktív felhasználó nem tud bejelentkezni), illetve ezen az oldalon kapnak olvasható címkét a jogosultsági mezők, ezzel együtt az egyes dolgozók munkaköre is.



Mivel rendelések és a felhasználók adatainak megjelenítése esetében előre meghatározott számú adatot kellett megjeleníteni, ezért a táblázatok olyan css tulajdonságot kaptak, amivel oldalirányban görgethetővé váltak kisebb készülékekre optimalizált nézetben.

8	Sonk Andrea	6	sonka@citromail.hu	+36301112226	1
9	Virs Lee	5	virsli@gmail.com	+36203336668	1
10	Füst Lee	5	fustli@citromail.hu	+36301234568	1
11	Kif Lee	4	kifli@gmail.com	+3670111222	1
12	Bruce Lee	4	brucelee@citromail.hu	+36309998786	1
36	jog8	8	jog8@jog8.hu	+3670111222	1
37	jog7	7	jog7@jog7.hu	+3612345678	1
38	jog6	6	jog6@teszt.hu	+3670111222	1
39	jog5	5	jog5@teszt.hu	+3612345678	1
40	jog4	4	jog4@teszt.hu	+3670111222	1

Tesztelés

Az oldal tesztelése fejlesztés közben is folyamatosan zajlott a böngészők illetve a fejlesztői IDE-k beépített eszközeivel. A frontend elkészültével funkció- és részemről designteszteknek vetettük alá az oldalt, felmérvén az esetleges felhasználásból eredő hibákat. Ezen túl a kód átesett a Clean Code elvei szerinti vizsgálaton is, manuálisan illetve SonarLinttel. Technikai automata tesztelésnek a GitLap verziókezelő automata tesztelésre alkalmas CI/CD folyamatát használtuk, ami következő sikeres tesztekkel zárult.

Status	Job	Stage	Name	Duration	Coverage
passed	#4222651160 main 42076ee0	deploy	deploy-job	00:00:43 just now	
passed	#4222651159 main 42076ee0	test	lint-test-job	00:00:54 2 minutes ago	
passed	#4222651158 main 42076ee0	test	unit-test-job	00:01:40 1 minute ago	
passed	#4222651157 main 42076ee0	build	build-job	00:00:42 3 minutes ago	

Build teszt

```
Running with gitlab-runner 15.9.0~beta.115.g598a7c91 (598a7c91)
  on blue-1.shared.runners-manager.gitlab.com/default j1aLDqxS, system ID: s_b437a71a38f9
  feature flags: FF_USE_IMPROVED_URL_MASKING:true
Preparing the "docker+machine" executor                                  00:30
Using Docker executor with image ruby:3.1 ...
Pulling docker image ruby:3.1 ...
Using docker image sha256:31bdc991169bde4a2569bf48a57bc5b9e5c29e77ef7d936092d6764ab2197d6e for ruby:3.1 with digest ruby@sha256:a2
a403a04612ca96d27eb2c32f35a69f7a52bed2a135f7ccfdd0139ba5d6fd81 ...
Preparing environment                                                    00:03
Running on runner-j1aldqxs-project-45508266-concurrent-0 via runner-j1aldqxs-shared-1683152278-4c4883e3...
Getting source from Git repository                                       00:06
$ eval "$CI_PRE_CLONE_SCRIPT"
Fetching changes with git depth set to 20...
Initialized empty Git repository in /builds/FReka32/FaDoKo/.git/
Created fresh repository.
Checking out 42076ee0 as detached HEAD (ref is main)...
Skipping Git submodules setup
Executing "step_script" stage of the job script                          00:00
Using docker image sha256:31bdc991169bde4a2569bf48a57bc5b9e5c29e77ef7d936092d6764ab2197d6e for ruby:3.1 with digest ruby@sha256:a2
a403a04612ca96d27eb2c32f35a69f7a52bed2a135f7ccfdd0139ba5d6fd81 ...
$ echo "Compiling the code..."
Compiling the code...
$ echo "Compile complete."
Compile complete.
Cleaning up project directory and file based variables                   00:01
Job succeeded
```

További teszt eredmények a mellékletben.

Az alkalmazás fejlesztés szakaszát így sikeresen lezártuk, a program élesítésre, illetve általános felhasználásra késznek tekinthető.

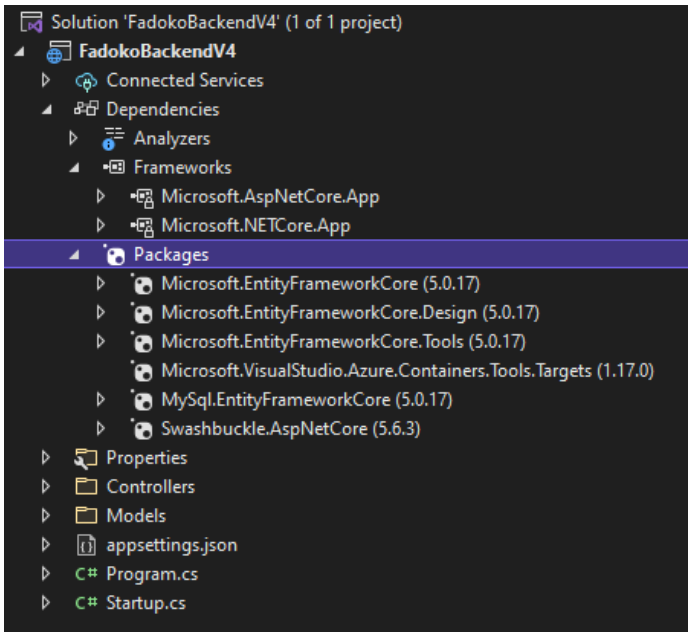
Irodalomjegyzék

1. Szelezsán János: Adatbázisok LSI 1992
2. Kupcsikné Fitus Ilona: Adatbázisok példatár LSI 1992
3. Luke Welling - Laura Thomson: PHP és MySQL webfejlesztőknek - Hogyan építsünk webáruházat? Perfact kiadó 2010
4. Steven Suehring - Janet Valade: PHP, MySQL, JavaScript & HTML5, Panem kiadó Tantusz könyvek 2014
5. Richard Blum: PHP, MySQL & JavaScript, Panem kiadó, Tantusz könyvek 2020
6. Szoftverarchitektúra https://simplicable.com/IT/thin-client-vs-thick-client
7. Fejlesztői környezet, Volar https://vuejs.org/
8. Fejlesztői környezet, Tabnine https://tabnine.com/
9. Fejlesztői környezet, GitLens https://gitkraken.com/
10. Vue.js https://ak-akademia.hu/mi-az-a-vue-js/
11. Bootstrap https://getbootstrap.com/docs/5.3/getting-started/introduction/

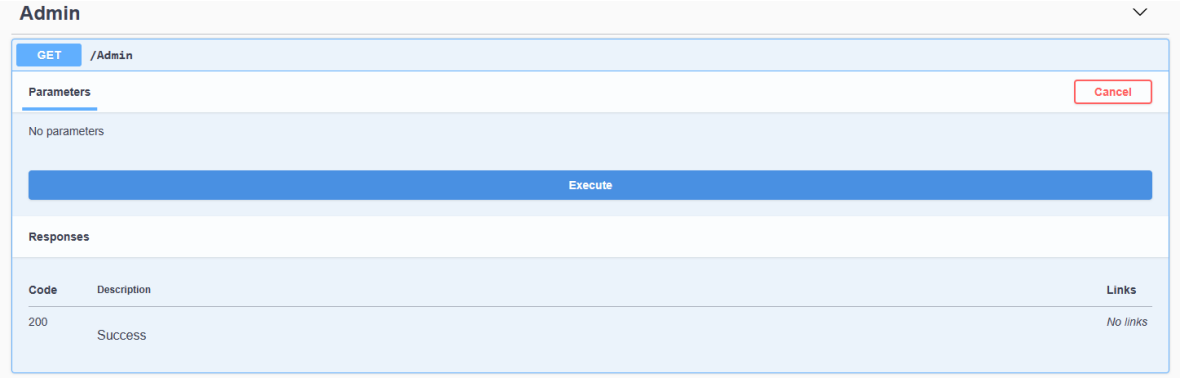
Mellékletek

Backend melléklet

Keretrendszerek és program csomagok

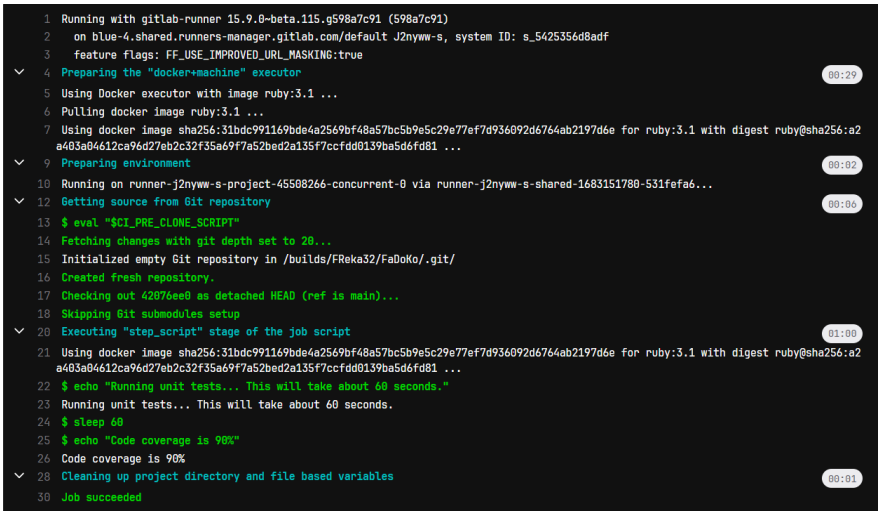


Swagger programmal, interaktív állapotban, kattintásra készen.

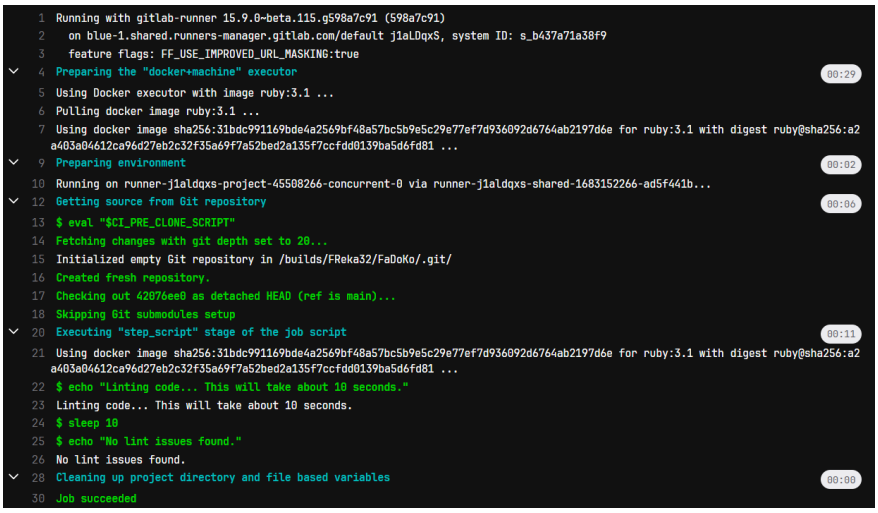


Frontend teszteredmények

Unit teszt



Lint teszt:



Deploy teszt: