Tehničko veleučilište u Zagrebu

MC2 natjecanje - tim MK2

Aplikacija za ugostiteljske objekte

Autori:

Maja Dabčević

Kristina Aničić

Kristijan Kerhin

Ožujak 2025.

POSLOVNI PLAN

Opis

Ideja za aplikaciju osmišljena je vlastitim iskustvom, situacijama i suočavanjem s nedostatcima ugostiteljskih objekata, što dovodi do negativnih mišljenja na sami objekt. Ugostiteljstvo se suočava s nedostatkom radne snage, što uzrokuje pad pružanja kvalitetne usluge gostima.

Situacije kao što su:

1. gost dolazi u kafić i sjedne na vanjsku terasu, većeg kapaciteta, koja je 90% popunjenosti. Vrlo vjerojatno, konobar neće i ne može odmah primijetiti da je netko sjeo za stol. Pogotovo, ako je u tom trenutku konobar bio unutar kafića(šank). Gost ne želi duže vrijeme(5-10+min) čekati za narudžbu.
2. Slična situacija kao u prvoj, kada je gost već za stolom i želi platit račun, često čeka priliku u kojoj može napraviti interakciju/kontakt/komunikaciju(verbalna/neverbalna) s konobarom kako bi mu dao do znanja da želi platiti račun.
3. Kao u drugoj situaciji, ali gost želi novu narudžbu.

Problemi

1. Konobar ne može primijetiti svakog novog gosta koji dođe i sjedne za neki stol.
2. Konobar mora prvo doći do gosta kako bi naručio piće, te slijedećim dolaskom mu donese piće.
3. Kada gost pozove konobara, konobar dolazi do gosta kako bi samo saznao zašto ga je gost pozvao.
4. U opisanim problemima(1-3) konobar ima „prazan hod“ što znatno utječe na brzinu usluge svakom slijedećem gostu. Također, ako su stepenice sastavni dio konobarovog radnog prostora, nije poželjno ni zabavno imati prazan hod tijekom višesatne smjene.

Rješenje

Aplikacija koja nudi mogućnost interakcije gosta i djelatnika(konobar/šank) kako bi olakšala i ubrzala rad ugostiteljskim djelatnicima i pružila kvalitetniju uslugu gostima.

MK2 PROJEKTNA DOKUMENTACIJA

1. Arhitektura rješenja

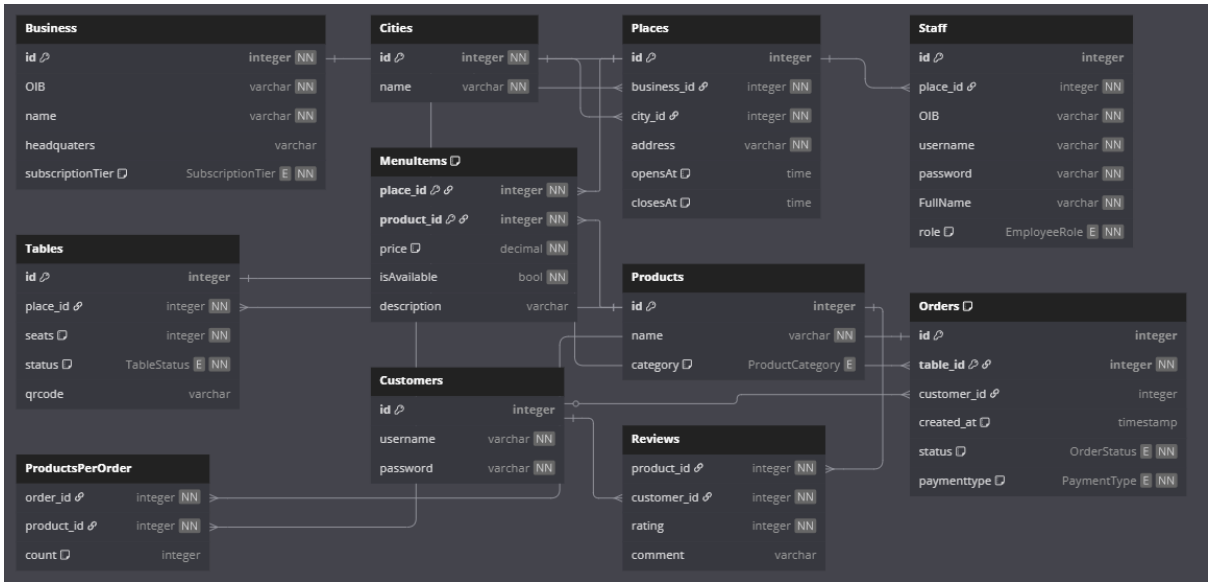
Web-aplikacija dizajnirana na temelju RESTful komunikacije između .NET pozadinskog sustava i React korisničkog sučelja. U budućnosti je plan unaprijediti komunikaciju korištenjem WebSocket-a gdje ima potrebe za time.

1.1. Baza podataka

Za potrebe razvoja koristimo najnoviju verziju PostgreSQL-a postavljenu unutar Docker kontejnera s inicijalizacijskim skriptama koje sadrže strukturu baze (Slika 1) i početne podatke potrebne za testiranje tijeka rada rješenja. Svaki put kada se skripta „spoji“ s glavnom granom, lokalna baza podataka treba se ponovno inicijalizirati kako bi se održalo konzistentno stanje baze na svakom računalu.

Odlučili smo se za korištenje višeorganizacijske (multi-tenant) baze podataka, dok će se ograničenja vlasništva među entitetima rješavati s poslužiteljske strane (backendu).

Prilagođeni „tipovi“ definirani su kao PostgreSQL enumeracije (enumi) kako bi se smanjio broj pomoćnih tablica (statusi narudžbi, načini plaćanja, uloge zaposlenika, razine pretplate i sl.).



Slika 1: Struktura baze podataka

1.2. Poslužiteljska aplikacija (backend)

Za pozadinski dio koristimo .NET 9, strukturiran kao više odvojenih projekata: Data, Domain, Backend, Tests, itd., kako bi se postigla bolja iskoristivost koda i lakše održavanje.

1.2.1. Data

Biblioteka klasa (class library) koja sadrži modele, enum vrijednosti i DbContext potreban za komunikaciju s bazom podataka putem Entity Frameworka.

1.2.2. Domain

Biblioteka klasa namijenjena za implementaciju poslovne logike.

Trenutno uključuje:

- Generički repozitorij
- Servise
- Pomoćne klase (utilities)
- Sučelja (interfaces)
- DTO-ove

Odlučeno je da svaki servis ima pristup samo potrebnim repozitorijima, umjesto čitavom kontekstu baze podataka, posebno za operacije stvaranja, ažuriranja i brisanja. Bilo kakve prilagođene upite moguće je provoditi pomoću izložene IQueryable metode unutar repozitorija, no takve implementacije treba pažljivo izvesti.

Servisi dohvaćaju podatke pomoću repozitorija (preko modela), a zatim ih transformiraju u odgovarajuće DTO-ove ovisno o zahtjevu kontrolera. Transformacija se obavlja pomoću AutoMapper alata, koristeći unaprijed definirane mape za svaki par DTO <-> entitet.

Autorizacija i međuentitetska ograničenja rješavaju se pomoću JWT tokena iz zahtjeva. To uključuje provjeru uloge korisnika prije izvođenja zahtjeva, te izdvajanje placeid parametra koji se koristi za filtriranje entiteta ovisno o objektu u kojem je korisnik zaposlen.

1.2.3. „Backend“

ASP.NET Core projekt koji sadrži kontrolere, Dependency Injection i potrebne konfiguracije. Pri pokretanju u razvojnom okruženju, pokreće Scalar UI sučelje temeljeno na OpenAPI dokumentaciji – koristi se za ručno testiranje API endpointa.

1.2.4. „Tests“

Test projekt koji trenutno sadrži samo jedinicne testove servisa temeljene na NUnit i NSubstitute.

1.3. Korisničko sučelje (Frontend)

Frontend koristi najnoviju verziju Node.js i React, trenutno konfiguriran s Tailwind CSS-om bez korištenja vanjskih knjižnica za gotove komponente.

Odlučeno je koristiti TypeScript i .tsx datoteke.

2. Postavljanje razvojnog okruženja (Development setup)

Potrebno je slijediti navedene korake za potpuni setup:

1. Instalacija Docker Desktop alata: https://www.docker.com/products/docker-desktop/
2. Kloniranje repozitorija (potrebno zatražiti pristup): https://github.com/mdabcevic/mk2
3. Instalacija NET 9 SDK: https://dotnet.microsoft.com/en-us/download/dotnet/9.0
4. Ažuriranje Visual Studio u skladu s novim SDK.
5. Instalacija Node.js okruženja: https://nodejs.org/en/download
6. Otvoriti direktorij BartenderBackend u konzoli – tamo se nalazi docker-compose.yml datoteka
7. Pokrenuti naredbu: docker compose up
8. Provjera na Docker Desktop sučelju je li Postgres kontejner pokrenut. Dodatno, pogledati logove kontejnera za provjeru konkretnih koraka: stvaranje, popunjavanje i operativni status baze
9. Otvoriti direktorij /frontend/mk2 u konzoli.
10. Pokrenuti naredbu: npm install

Nakon izvršavanja prethodnih koraka, uspješno je postavljeno okruženje te je spremno za pokretanje poslužitelsjke aplikacije i / ili korisničkog sučelja pomoću navedenih naredbi:

- Otvoriti Backend.sln (Visual Studio) unutar backend direktorija; pokretanjem se otvara Scalar UI putem google Chrome koji omogućava pregled i testiranje kontrolera tokom razvoja
- U direktoriju frontend/mk2 pokrenuti naredbu: npm run dev

3. Ključne značajke

Kako su poslovni objekti naši glavni korisnici, fokusirani smo na implementaciju sljedećih značajki:

- Generiranje statičkog QR koda za svaki stol koji se mora isprintati i postaviti
- QR kod povezan je s API kontrolerom i omogućava upravljanje narudžbama i stolovima
    - Potvrđivanje / odbacivanje narudžbu
    - Označavanje stola kao slobodnog / rezerviranog
- Obavijesti o promjeni statusa
- Upravljanje objektom (za menadžere)
    - Registracija zaposlenika
    - Dodavanje stolova
    - Uređivanje jelovnika, ponuda, događaja i popusta
- Autentifikacija i autorizacija za zaposlenike i menadžere
- Pregled objekta s praćenjem statusa stolova i narudžbi po stolu
- Analitička nadzorna ploča na temelju stolova, narudžbi i proizvoda

Sve navedene značajke bit će raspoređene po razinama pretplate.

Dodatne implementacije dostupne za javnost:

- API za dohvaćanje i filtriranje dostupnih objekata, jelovnika, ponuda / događaja i trenutne zauzetosti
- Ograničeno upravljanje stolovima / narudžbama nakon skeniranja
- Notifikacije

4. Stanje razvoja

Evo popisa koraka koji su trenutno u raspravi, dovršeni ili u tijeku:

- Postavljanje projekta – vidi poglavlja 1 i 2
- Djelomično popunjavanje baze:
    - Poslovni korisnici i njihove pretplate
    - Popis objekata po poslovnom korisniku
    - Zaposlenici u navedenim objektima
    - Proizvodi i kategorije (u izradi)
    - Jelovnik za svaki objekt
- Osnovna komunikacija Backend <> Baza
- Prikaz objekata na frontend-u (zasad pomoću placeholder podataka – backend još u izradi)
- Lokalizacija korisničkog sučelja (hrvatski / engleski)
- JWT pristupna kontrola
- CRUD operacije za poslovne korisnike, zaposlenike, proizvode i jelovnik (ovisno o korisniku)
- Navigacija i admin prikazi na frontend-u
- Web skener za QR kodove

Više informacija dostupno je na GitHubu, gdje se prati stanje zadataka, grananja i zahtjeva te raspravlja o implementacijama.