Visoka škola strukovnih studija za
informacione i komunikacione tehnologije

PerfumeShop API

https://github.com/LazicNenad/PerfumeShop

Smer: Internet tehnologije

Modul: Web programiranje

Predmet: Web Programiranje ASP

Nenad Lazić 9/19

Sadržaj

Contents

1. Uvod

1.1 Korišćeni programski jezici/tehnologije

Za potrebe sajta korišćene su sledeće tehnologije:

- ASP.NET Core (.NET 6 LTS)
- EntityFramework Core
- JWT Authorization
- BCrypt for password hashing
- FluentValidation for validation
- Newtonsoft.Json

1.2 Opis funkcionalnosti i korišćenje API-ja

Note*: Jednom migracijom se popunjava cela baza podataka sa svim vezama, inicijalno je bilo desetak migracija gde sam nailazio na neke greske kada sam dodavao tabele i zbog vremena predavanja projekta sam izbrisao snapshot i dodao novu migraciju koja je sadrzala kompletan sadrzaj prethodno dodat.

Prva stvar koja je potrebna da se uradi da bi API radio jeste da se u SSMS napravi baza podataka pod imenom PerfumeShop i da se kroz kod pokrenu migracije.

Nakon uspsno odradjene migracija potrebno je 'Seedovati' bazu podataka a to se vrsi na endpointu: /api/initialdata [Post Requestom].

S obzirom da je za vecinu slucajeva koriscenja (UseCases) potrebna autorizacija kao i da se na osnovo konkretnog korisnika kreiraju slucajevi koriscenje samo za njega, potrebno je

prvenstveno se registrovati na ruti (/api/users [Post requestom]) gde je definisano koja su polja obavezna i odradjena je validacija podataka na tom novou.
A nakon uspesne registracije potrebno je na ruti /api/token [Post Request] generisati token i sa tim tokenom se autorizovati na ruti /api/addUserUseCases kako bi se izvrsio insert u bazu podataka u tabelu UserUseCases gde ce se na osnovu ulogovanog korisnika omogućiti slučajevi korišćenja za njega (U ovom slučaju korisnik će dobiti sve privilegije našeg sistema).

Arhitektura projekta je takva da se trenutno sastoji od 5 nezavisnih celina odnosno 5 odvojenih projekata.
1. API Projekat
2. Application (Sloj projekta u kom se nalaze apstrakcije koje su podeljene na osnovu CQRS Patterna kojim se odvajaju komande(sve sto menja stanje sistema) i query)
3. DataAccess (Sloj u kom se nalazi konekcija sa bazom podataka kao i DbContext klasa koja predstavlja objekat baze podataka u celini kao i konfiguracije i migracije)
4. Domain (Najbitniji i najnezavisniji sloj u aplikaciji i u njemu se nalaze definicije entiteta koje nas sistem koristi odnosno u nasem konkretnom slucaju to su tabele u SQL Server bazi podataka)
5. Implementation (Sloj implmenetacije, konkretne implementacije apstraktnih stvari koje smo definisali u Application sloju)

Endpoints:

Brands: (Potrebna je autorizacija za svaki endpoint)
GET /api/brands - Dohvatanje svih brendova iz baze podataka, Omogucena paginacija i pretraga po keyword
GET /api/brands/{id} – Dohvatanje brenda po id-ju
POST /api/brands - Kreiranje novog brenda, Detaljna validacija je odradjena i konkretne poruke ce biti prikazane
PUT /api/brands/{id} – Update postojeceg brenda, Validacija je odradjena
DELETE /api/brands/{id} – Extension metod nad klasom PerfumeContext koja omogucava soft delete entiteta. Vrsi soft delete entiteta i takodje je primenjen globalni query filter kojim se ignorise entitet ukoliko je IsDeleted svojstvo postavljeno na true.

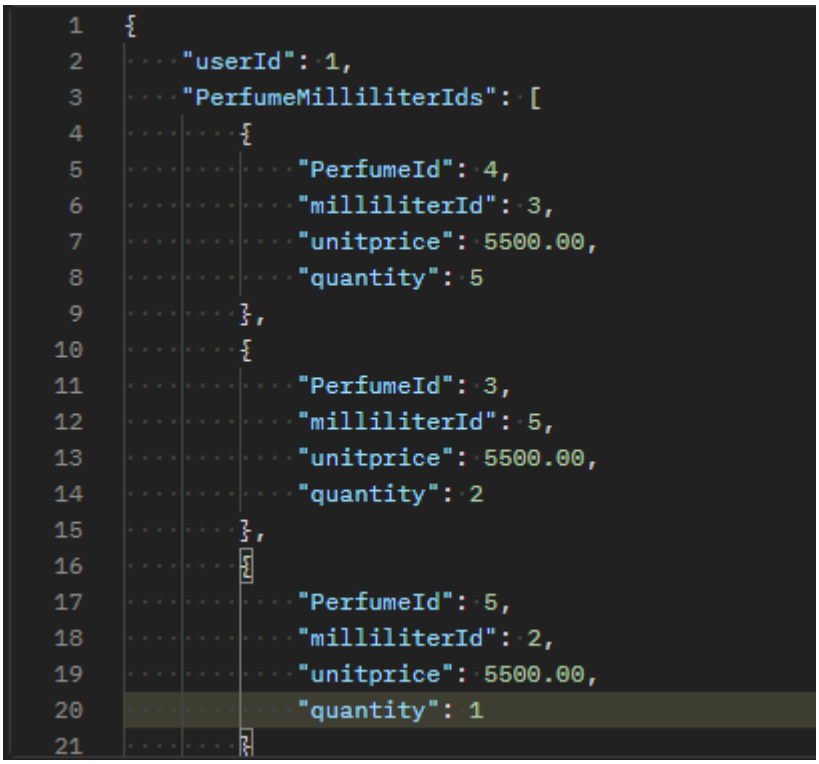
Milliliters (Slicna logika kao kod brands, jednostavan entitet sa jednim svojstvom)

Token:
POST /api/token – Generise JWT Token kojim je kasnije obavezna autorizacija.

Users:

GET /api/users –Dohvatanje svih korisnika sistema
POST /api/users – Registracija korisnika, uradjena detaljna validacija
Perfumes:
GET /api/perfumes – Dohvatanje svih parfema iz baze podataka, Omogucena pretraga po keyword, paginacija
GET /api/perfumes/{id} – dohvatanje parfema po idju
POST /api/perfumes – Unos novog parfema u bazu podataka, Omogucena detaljna validacija Name, Description, BrandId (Mora da postoji u bazi podataka, veci od 0), CategoryId(Mora da postoji u bazi i da je veci od 0), Niz ProductTypeIds (Da se ne ponavljaju id-jevi, da postoje svi u bazi podataka) kao i MilliliterIds.
PUT /api/perfumes/{id} – Update parfema po idju, detaljna validacija odradjena
DELETE /api/perfumes/{id} – Soft delete konkretan parfem i brisanje parfema sa tim idjem u svim vezivnim tabelama, u ovom konkretnom slucaju 2 vezivne tabele.

Orders je tabela koja koja sluzi da prihvati porudzbinu odredjenog usera i prepise kupljene stavke u tabelu OrderLine gde svaki red predstavlja jednu stavku porudzbine.



Ovako bi trebao da izgleda JSON za slanje i skladistenje stavki korpe korisnika.

Aplikacija je izradjena tako da koristi Globalni exception handler (Program.cs -> app.UseMiddleware<GlobalExceptionHandler>()) kao i da koristi UseCaseHandler koji ce

handlovati ili komandu ili query na nacin da ce se meriti vreme koje je trebalo da se izvrsi komanda, ko je izvrsio komandu, da li je autorizovan i ispisati to u konzoli.

Takodje su napravljene extension metode za ServiceCollection kako se ne bi pretrpao Program.cs odakle nam pocinje cela aplikacija.