Universitatea *Transilvania* din Braşov
Facultatea de Matematică şi Informatică
Departamentul de Matematică şi Informatică

Documentatia Proiectului Magazin de articole sportive

Căţoi Mihai-Alexandru
Enache Mihai Gabriel

Informatică Aplicată, Anul II
Grupa 10LF321

June 26, 2024

Documentatie

Conţinut

1 Prezentarea proiectului **1**

2 Tehnologiile folosite **1**

3 Baza de date **2**

4 API-Swagger **3**

5 Prezentare functionalitate aplicatie **4**

6 Concluzii si Contributii **4**

7 Link Git **4**

1 Prezentarea proiectului

Proiectul nostru este un API al unei aplicaţii de gestionat un magazin de articole sportive.

Proiectul îşi propune sa foloseasca endpoint-uri astfel încât să putem efectua operaţiile de CRUD asupra clienţilor, angajaţilor, comenzilor şi a produselor.

2 Tehnologiile folosite

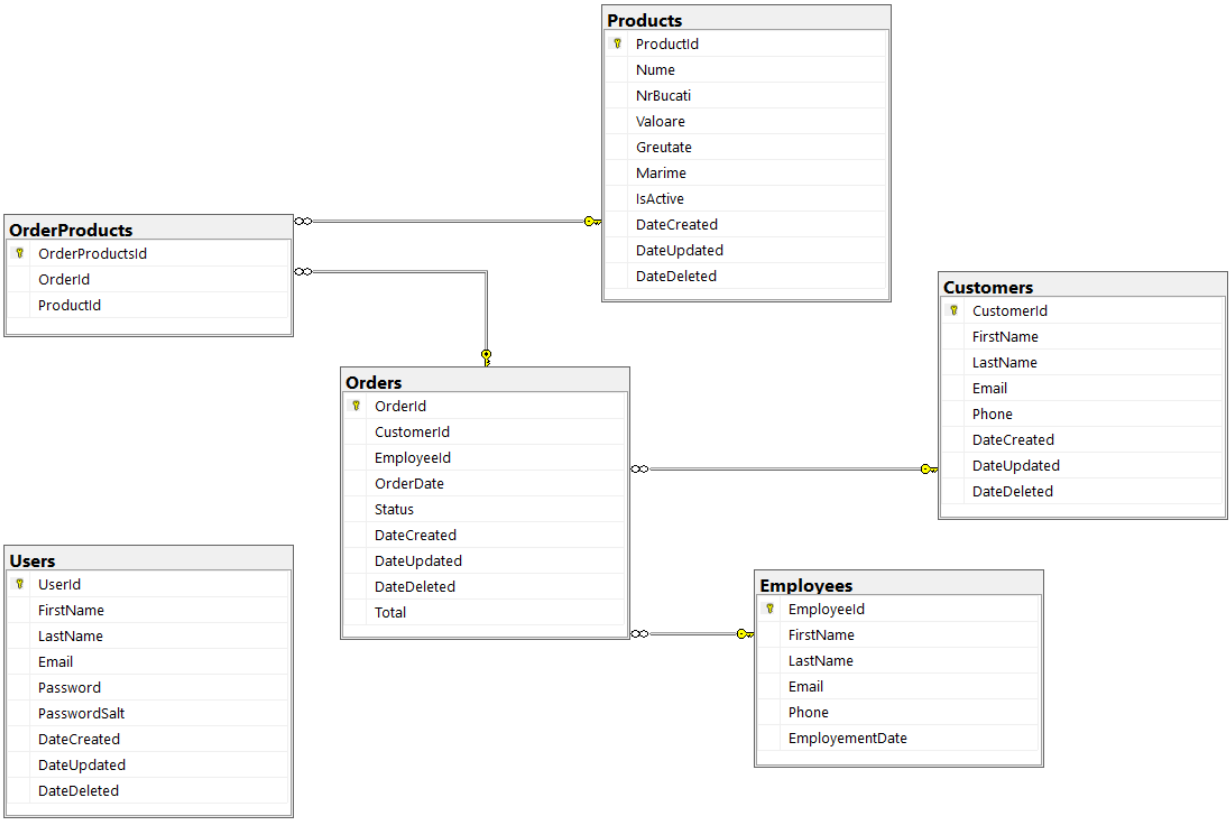
Am folosit ASP.NET Core Web API pentru baza API-ului alături de DLL-uri pentru implementarea claselor ce ţin de structura bazei de date şi de Serviciile folosite.

De asemenea am folosit Nugget-uri precum SwashBuckle pentru o interfaţă mai plăcută şi mai multe pachete din EntityFrameworkCore pentru a efectua migraţii şi a lucra mai usor cu baza de date.

Am mai folosit JWT pentru sistemul de register/login (autentificare/autorizare).

Pentru baza de date am folosit SSMS.

3 Baza de date

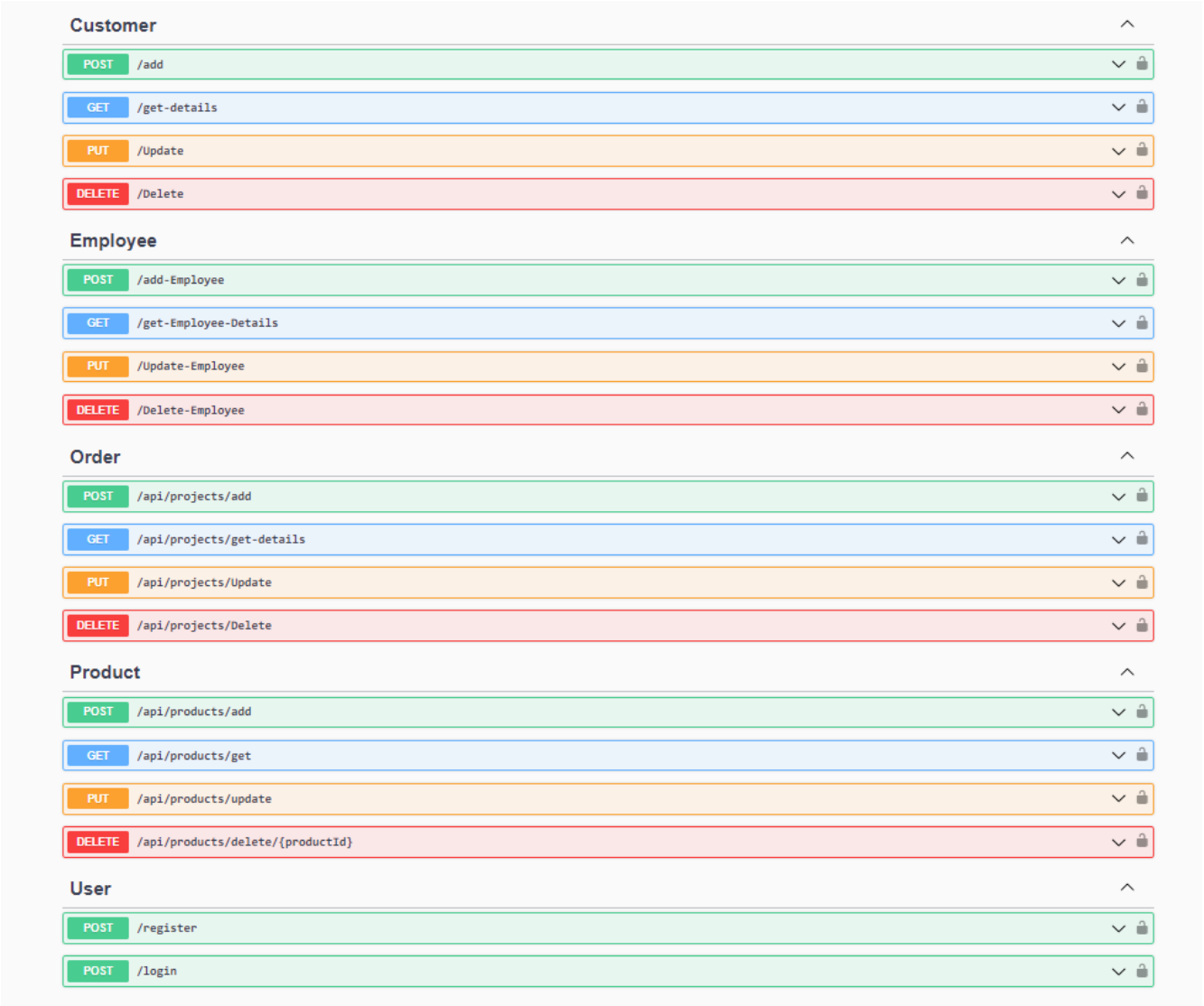


În baza de date avem tabelele Customer, Employee, Order şi Product. Deoarece relaţia Order-Product este Many-to-Many am inserat tabelul de legătură OrderProducts pentru a fi în forma normala 3.

Order are ca şi chei străine EmployeeId si CustomerId deoarece pe o comandă trebuie să apară clientul care a efectuat comanda şi angajatul care a procesat-o.

Tabela Users nu este legată de nici o entitate deoarece acela este doar contul de login.

4 API-Swagger



Fiecare controller are 4 endpoint-uri: Post(Create),Get(Read),Put(Update) si Delete.

- La endpoint-ul de **Post** se afişează în format json datele care trebuie completate şi la apăsarea butonului execute, acestea vor fi înregistrate în baza de date.
- La endpoint-ul **Get**, la apăsarea butonului execute se vor aduce datele din tabelul aferent din baza de date, structurate în format json, iar la final se va afişa count-ul (cate date au fost aduse).
- La endpoint-ul **Put**, se va afişa similar ca la endpoint-ul Post, iar datele completate vor înlocui datele existente la id-ul specificat.
- La endpoint-ul **Delete** se cere introducerea Id-ului din tabela aferentă, al item-ului care dorim sa fie şters.

La Ştergere sau Update, dacă se introduce un Id care nu există în baza de date, se va afişa un mesaj de succes deşi nu s-a întamplat nimic.

5 Prezentare functionalitate aplicatie

La autentificare există 2 roluri: Admin şi User.

- **Admin** - poate folosi toate endpoint-urile după cum doreşte.
- **User** - poate accesa doar endpoint-urile de Post,Put şi Get, deoarece updatarea unor date existente şi ştergerea acestora nu ar trebui să fie permise oricui.

6 Concluzii si Contributii

I Pentru început am discutat care să fie tematica proiectului şi care sa fie punctul de plecare.

II După ce am ales tema (magazin de articole sportive) am discutat despre schema bazei de date, până am ajuns la o formă finală.

III Ne-am împărţit sarcinile astfel încât amandoi să avem un numar egal de sarcini de făcut. Alex s-a ocupat de CRUD pentru Employee, Customer şi Order deoarece Order-ul îl avea deja făcut în Temă. Mihai s-a ocupat de CRUD pe Product,de tabelul de legătură, Autorizare şi filtrare la Get-uri.

IV În final am verificat amândoi să nu fi uitat nimic, şi am făcut mici retuşuri înainte de a fi siguri că este complet.

Din această experienţă am învăţat:

1. Să folosim JWT pentru autorizări/autentificări
2. Să folosim migraţii pentru a simplifica lucrul cu baze de date.
3. Să împărţim un proiect mare în suproiecte mai mici, fiecare cu propria funcţionalitate.
4. Să creăm un API de la 0 şi care sunt principiile pe care funcţionează.

7 Link Git

https://github.com/AlexCatoi/Proiect.Net