DOCUMENTAȚIA PROIECTULUI

**Supermarket API**

STUDENȚI

Neagu Radu Mihai

Sârbu Alin Nicușor

**Cuprins**

1. Prezentarea proiectului, ce isi propune, ce probleme rezolva

2. Tehnologiile folosite

3. Baza de date: diagrama bazei de date + scurta prezentare a tabelelor si a relatiilor dintre ele

4. Prezentarea API-ului: screenshot la Swagger pentru a putea vedea toate endpoint-ruile + scurta descriere a CRUD-urilor

5. Prezentare despre cum poate fi utilizata aplicatia: tipuri de utilizatori, ce vede fiecare, autentificare etc

6. Concluzii si contributii (cum v-ati impartit task-urile, ce ati invatat in urma acestui proiect etc)

7. Link GIT catre codul proiectului

1. **Prezentarea proiectului ce își propune ce probleme rezolvă**

Proiectul **SupermarketAPI** își propune să ofere o soluție modernă și eficientă pentru gestionarea operațiunilor zilnice ale unui supermarket. API-ul nostru contribuie la gestionarea produselor, procesarea comenzilor și administrarea utilizatorilor în funcție de tip: utilizator regulat sau admin. Prin automatizarea și simplificarea acestor procese, API-ul contribuie la reducerea erorilor umane, îmbunătățirea eficienței operaționale și oferirea unei experiențe mai bune pentru clienți.

**Problemele specifice pe care le rezolvă acest proiect:**

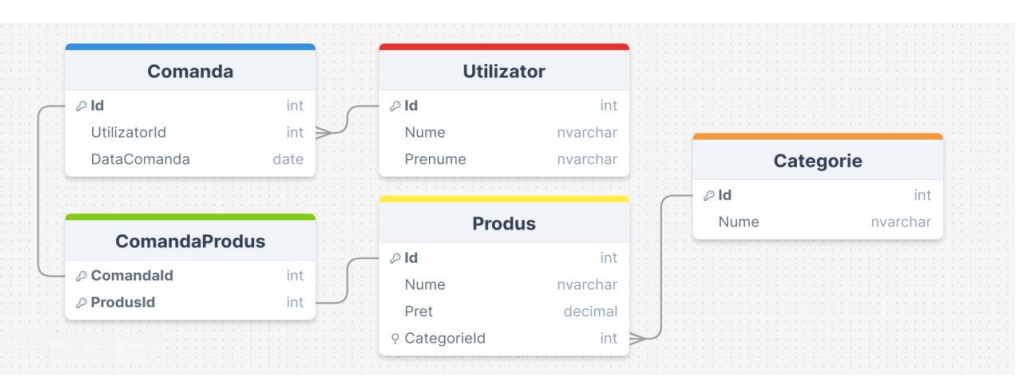
* Gestionarea eficientă a comenzilor
* Procesarea și evitarea erorilor cu privire la comenzi
* Raspuns specific pentru fiecare request în parte

1. **Tehnologiile folosite**

Pentru dezvoltarea proiectului SupermarketAPI am folosit următoarele tehnologii:

* **Backend**: ASP.NET Core cu C# pentru server și managementul API-ului.
* **Baza de date**: Entity Framework Core pentru ORM (Object-Relational Mapping) și SQL Server pentru stocarea și gestionarea datelor.
* **Autentificare și autorizare**: ASP.NET Core Identity și JSON Web Tokens (JWT) pentru autentificare și autorizare.

**3. Baza de date**



**Diagrama bazei de date**

**Tabele și relații**

**Comanda (Command):**

* Id (int) - Cheia primară, identificator unic al comenzii.
* UtilizatorId (int) - Cheie străină, face referire la Id-ul din tabelul Utilizator.
* DataComanda (date) - Data la care a fost plasată comanda.

**Utilizator (Users):**

* Id (int) - Cheia primară, identificator unic al utilizatorului.
* Nume (nvarchar) - Numele utilizatorului.
* Prenume (nvarchar) - Prenumele utilizatorului.

**Categorie (Categories):**

* Id (int) - Cheia primară, identificator unic al categoriei.
* Nume (nvarchar) - Numele categoriei.

**ComandaProdus (CommandProduct):**

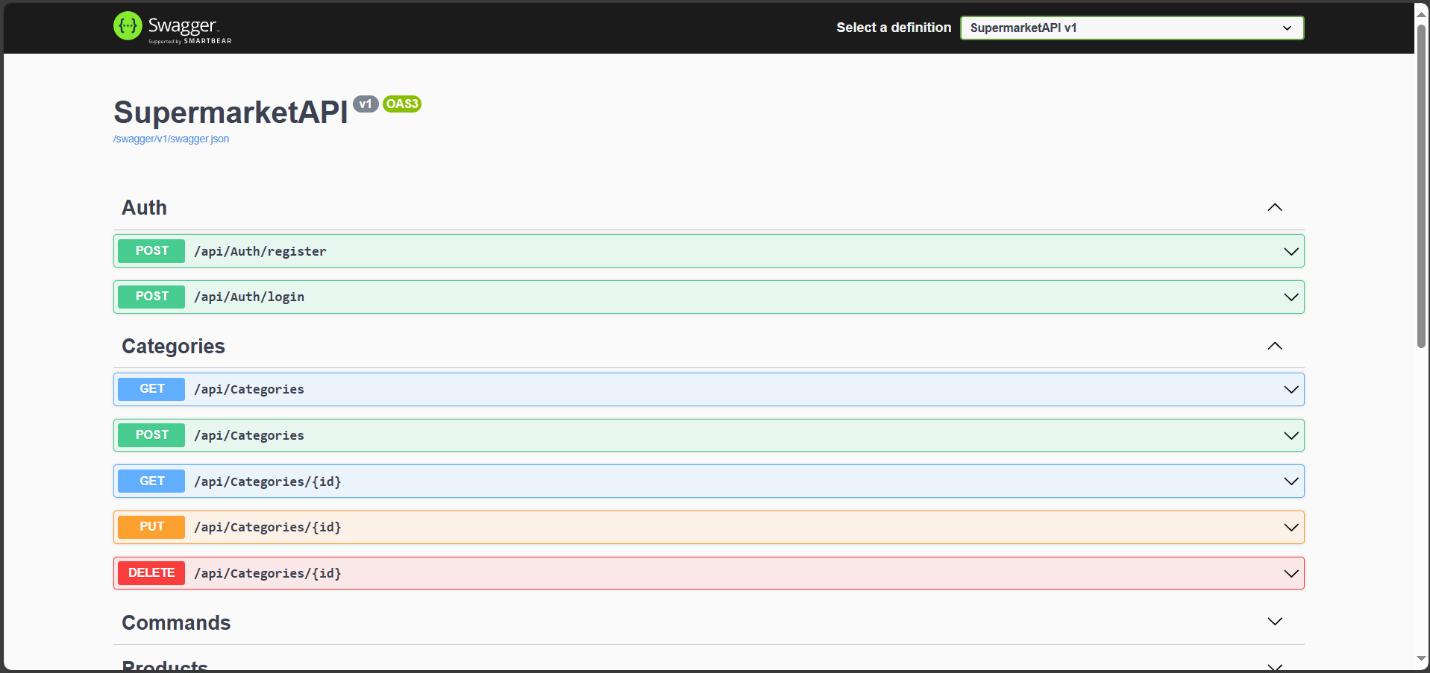
* ComandaId (int) - Cheie străină, face referire la Id din tabelul Comanda.
* ProdusId (int) - Cheie străină, face referire la Id din tabelul Produs.

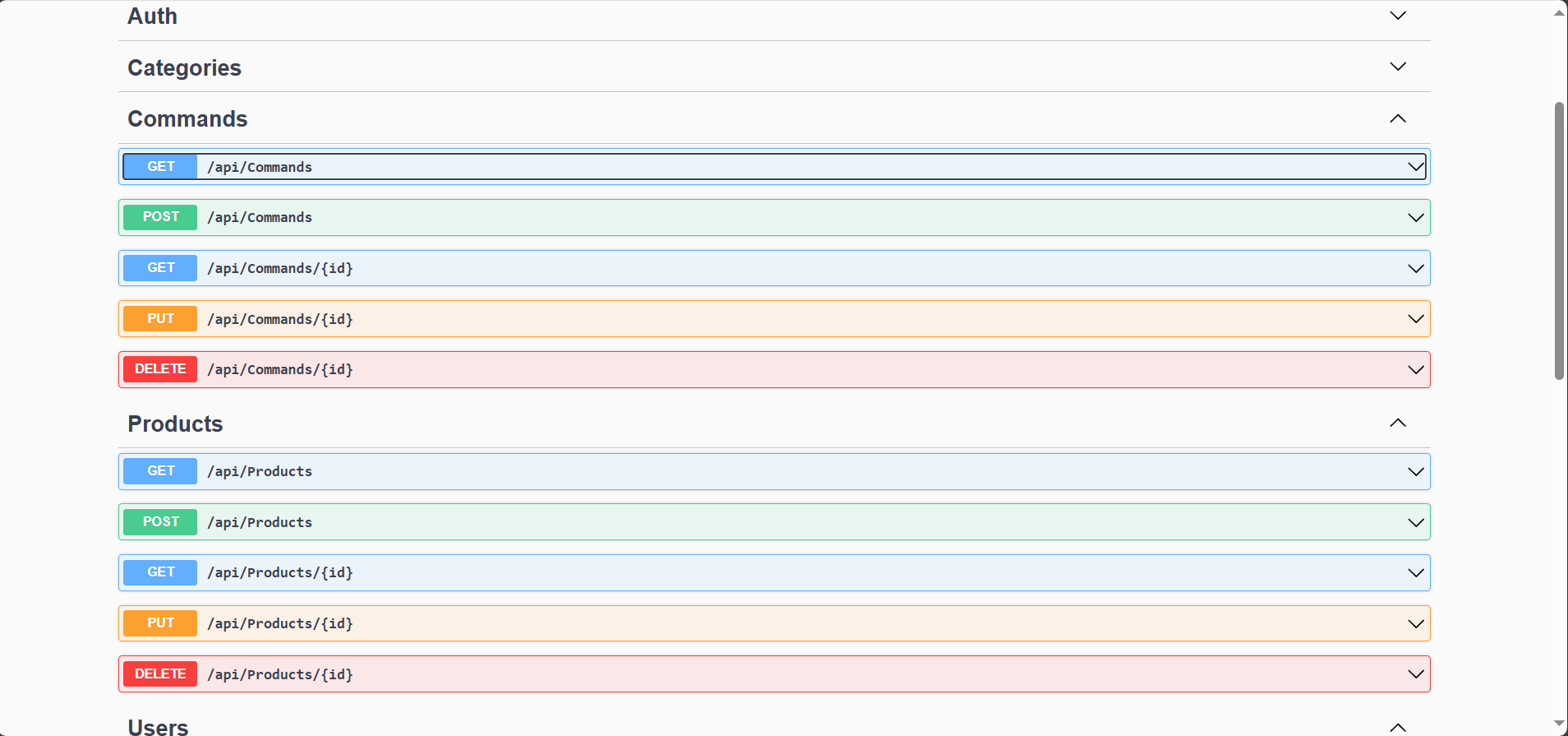
**Produs (Products):**

* Id (int) - Cheia primară, identificator unic al produsului.
* Nume (nvarchar) - Numele produsului.
* Pret (decimal) - Prețul produsului.
* CategorieId (int) - Cheie străină, face referire la Id din tabelul Categorie.

**Relații între tabele**

* **Comanda și Utilizator:** Un utilizator poate avea mai multe comenzi. UtilizatorId din tabelul Comanda face referire la Id din tabelul Utilizator **(Relație de tip one-to-many)**
* **Categorie și Produs:** **Relație de tip one-to-many** (o categorie poate avea mai multe produse, dar un produs poate avea o singură categorie). CategorieId din tabelul Produs face referire la Id din tabelul Categorie.
* **Comanda și Produs:** Relație de tip many-to-many, gestionată prin tabelul intermediar ComandaProdus. ComandaId din tabelul ComandaProdus face referire la Id din tabelul Comanda, iar ProdusId din tabelul ComandaProdus face referire la Id din tabelul Produs
  1. **Prezentarea API-ului**

****

****

**Scurtă descriere a CRUD-urilor:**

* **Create:**
  + POST /api/products - Adaugă un nou produs.
  + POST /api/categories - Adaugă o nouă categorie.
  + POST /api/orders - Creează o nouă comandă.
  + POST /api/users/register - Înregistrează un nou utilizator.
* **Read:**
  + GET /api/products - Returnează lista produselor.
  + GET /api/products/{id} - Returnează detaliile unui produs specific.
  + GET /api/categories - Returnează lista categoriilor.
  + GET /api/categories/{id} - Returnează detaliile unei categorii specifice.
  + GET /api/orders - Returnează lista comenzilor.
  + GET /api/orders/{id} - Returnează detaliile unei comenzi specifice.
  + GET /api/users - Returnează lista utilizatorilor.
  + GET /api/users/{id} - Returnează detaliile unui utilizator specific.
* **Update:**
  + PUT /api/products/{id} - Actualizează informațiile unui produs.
  + PUT /api/categories/{id} - Actualizează informațiile unei categorii.
  + PUT /api/orders/{id} - Actualizează statusul unei comenzi.
  + PUT /api/users/{id} - Actualizează informațiile unui utilizator.
* **Delete:**
  + DELETE /api/products/{id} - Șterge un produs.
  + DELETE /api/categories/{id} - Șterge o categorie.
  + DELETE /api/orders/{id} - Șterge o comandă.
  + DELETE /api/users/{id} - Șterge un utilizator.
  1. **Prezentare despre cum poate fi utilizată aplicația**

**Tipuri de utilizatori**

* **Admin:**
  + **Ce poate face:** Gestionarea completă a produselor, categoriilor, comenzilor și utilizatorilor. Beneficiază de acces complet la toate funcționalitățile aplicației.
  + **Ce vede:** Dashboard cu opțiuni pentru gestionarea produselor, categoriilor, comenzilor și utilizatorilor, statistici și rapoarte.
* **Angajat:**
  + **Ce poate face:** Gestionarea stocurilor de produse, actualizarea informațiilor produselor și gestionarea comenzilor (actualizare status).
  + **Ce vede:** Dashboard cu lista produselor și comenzilor, opțiuni pentru actualizarea produselor și statusului comenzilor.

**Autentificare și autorizare**

* **Admin:**
  + După autentificare, adminul vede un dashboard cu toate opțiunile de administrare. Accesul este restricționat pe baza rolului de admin.
* **Angajat:**
  + După autentificare, angajatul are acces la funcțiile de gestionare a stocurilor și comenzilor. Accesul este restricționat pe baza rolului de angajat.
  1. **Concluzii și contribuții**

**Concluzii**

Proiectul SupermarketAPI ne-a oferit oportunitatea de a colabora eficient și de a dezvolta o aplicație complexă care să răspundă cerințelor unui supermarket modern. Am reușit să implementăm funcționalități esențiale pentru gestionarea produselor, categoriilor, comenzilor și utilizatorilor, asigurând în același timp securitatea și integritatea datelor.

**Contribuții**

* **Neagu Radu Mihai:**
  + **Task-uri:** Dezvoltarea backend-ului cu ASP.NET Core, inițializarea bazei de date și implementarea endpoint-urilor pentru gestionarea comenzilor și utilizatorilor.
  + **Ce a învățat:** Optimizarea performanței aplicației și gestionarea eficientă a bazelor de date.
* **Sârbu Alin-Nicușor:**
  + **Task-uri:** Realizarea de modele, integrarea cu backend-ul, configuarea bazei de date și implementarea interfeței pentru gestionarea produselor și categoriilor.
  + **Ce a învățat:** Crearea interfețelor, utilizarea eficientă a ASP.NET pentru dezvoltarea aplicațiilor web și aprofundarea noțiunilor cu privire la baze de date

**7. Link GIT către codul proiectului**

Pentru a accesa codul proiectului, vizitați repository-ul nostru pe GitHub:

[alinsarbu23/DAWM-Project (github.com)](https://github.com/alinsarbu23/DAWM-Project)