DOCUMENTAȚIA PROIECTULUI

Gestionarea cheltuielilor personale

Studenti: - Enache Nicolae

-Ignat Dan

1.Prezentarea proiectului

Acest proiect isi propune crearea unui API in ASP.NET pentru gestionarea cheltuielilor personale. Proiectul isi propune sa puna la dispozitia utilizatorilor o platforma prin care isi pot crea un cont personal compus din numele lor, adresa email si o parola. Utilizatorii pot folosi aceasta aplicatie pentru a tine cont de proprile lor cheltuieli sau a calcula suma totala a cheltuielilor facute intr-o anumita perioada de timp.

2.Tehnologiile folosite

In aplicatia noastra sunt implicate mai multe tehnologii si concepte ASP.NET. Cele mai semnificative tehnologii folosite sunt urmatoarele:

2.1. ASP.NET Core

Este un framework-ul open-source dezvoltat de Microsoft pentru a construi aplicatii web rapide si scalabile. Prin utilizarea lui, ne putem folosi de atribute speciale precum [ApiController], [HttpGet], [HttpPost], [HttpPut] si [HttpDelete] pentru definirea rutelor si metodelor HTTP.

2.2. Entity Framework Core

Framework folosit pentru interactiunea cu baza de date prin DbContext, DbSet, metode asincrone pentru accesul la baza de date etc.

2.3. JWT Authentication + Authorization

Folosita pentru securizarea endpointurilor API. Prin atributul [Authorize(Roles = "Admin")], endpointul poate fi accesat doar de utilizatori cu rolul de admin.

2.4. Swagger UI

Reprezinta UI-ul folosit pentru documentarea și testarea API-ului în browser. Prin acesta se pot testa metodele GET, POST, PUT si DELETE.

3.Prezentarea tabelelor

Proiectul este compus din 3 tabele principale. Clasa de entitate a fiecarui tabel este derivata dintr-o clasa abstract BaseEntity, care contine un camp de tip intreg pentru ID si doua campuri de tip DateTime pentru a salva data la care a fost creat sau modificat un obiect. Cele 3 tabele principale sunt urmatoarele:

3.1. Tabelul User

Reprezinta tabelul in care se salveaza datele utilizatorului. Campurile clasei User sunt:
-string Name
-string Password
-string Email
-string Role (un user poate sa fie user normal sau admin)
-bool IsActive
-Lista de obiecte de tip Expense care pastreaza cheltuielile utilizatorului.

3.2. Tabelul Expense

Reprezinta tabelul folosit pentru fiecare cheltuiala. Clasa este compusa din:
-double Amount
-DateTime PaymentDate
-string Receiver (persoana sau organizatia la care a fost facuta cheltuiala)
-User User (utilizatorul la care apartine cheltuiala)
-int UserId
-Category Category (categoria din care face parte)
-int CategoryId

3.3. Tabelul Category

Reprezinta tabelul in care se salveaza categoriile pentru cheltuieli. In clasa tabelului se afla:
-string Name
-string Description
-List de cheltuieli din care sunt de tipul categoriei respective

3.4. Relatiile dintre tabele

In proiect exista 2 relatii de tip one-to-many intre User si Expense (un utilizator poate avea mai multe cheltuieli) si intre Category is Expense (o categorie poate sa apartina la mai multe cheltuieli).

4. Prezentarea API-ului

Proiectul poate fi categorizat in 3 parti diferite bazate pe subproiectele aplicatiei:

4.1. ExpensesManager.Api

In acest subproiect se afla clasele **Controller** care preiau cererile HTTP (GET, POST, PUT, DELETE) si le redirectioneaza catre Service

4.2. ExpensesManager.Core

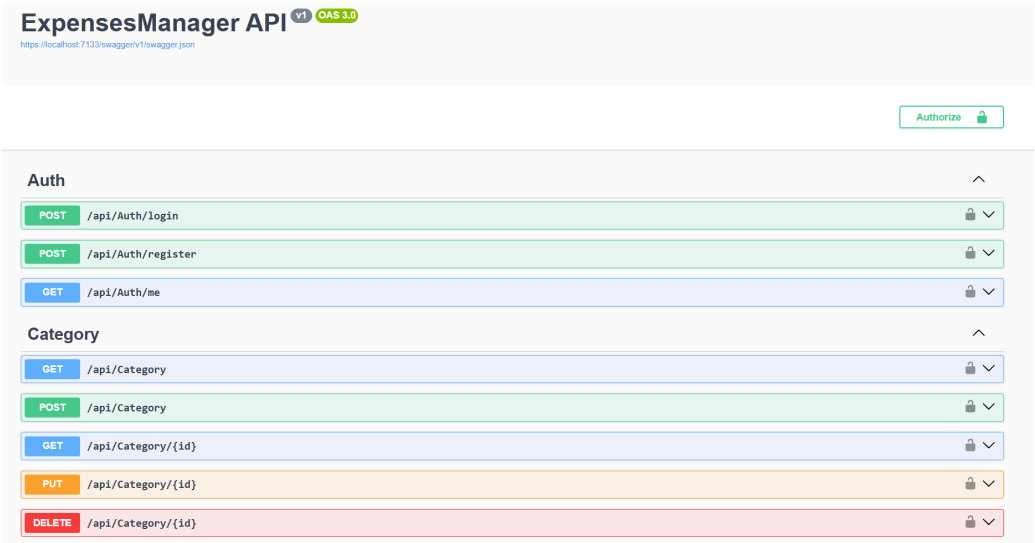
Contine clasele de entitate si clasele **DTO** care sunt folosite pentru a transfera date intre client si server fara sa expuna direct entitatile care pot contine relatii complexe sau date sensibile. Clasa **Service** face conversia intre entitati si dto-uri, apeleaza metode din Repository este responsabila de business logic.

4.3. ExpensesManager.Database

Contine clasele **Repository** care se ocupa de operatiile cu baza de date (CRUD: Create, Read, Update, Delete).

4.4. Operatiile CRUD

Proiectul contine toate cele 4 operatii CRUD folosite pentru manipularea datelor. Acestea sunt:
-Create/POST (crearea unui obiect nou)
-Read/GET (citirea unui obiect existent)
-Update/PUT (actualizarea unui obiect existent)
-Delete/DELETE (stergerea unui obiect existent)



5. Cum poate fi utilizata aplicatia?

Aplicatia poate sa fie utila pentru oamenii care doresc sa isi salveze cheltuielile facute si sa vada cat de multi bani cheltuiesc intr-o anumita perioada de timp. Utilizatori de tip user normal pot sa isi vada sau sa isi modifice proprile date, iar cei cu rolul admin pot sa vada datele pentru fiecare utilizator.

6. Ce am invatat?

Prin acest proiect am reusit sa vedem ce reprezinta programarea de tip backend si am putut folosi concepte de baze de date pe care le-am invatat in semestre trecute intr-un proiect mai complex.

7. Impartirea task-urilor

Enache Nicolae - Crearea claselor entitate si DTO, crearea claselor Controller, Service si Repository pentru tabele Expense, Category si (partial) User precum si operatiile CRUD din acestea.

Ignat Dan - Autentificarea si autorizarea de user prin JWT, paginarea, crearea claselor Controller, Service si Repository pentru tabelul User precum si operatiile CRUD din acesta.

8. Link GitHub

https://github.com/RedMateria-hub/ExpensesManager