# TaskManager

Aplicația web *TaskManager* este destinată managementului sarcinilor. Utilizatorii aplicației pot crea, modifica diverse sarcini care pot fi atribuite altor angajați sau nu.

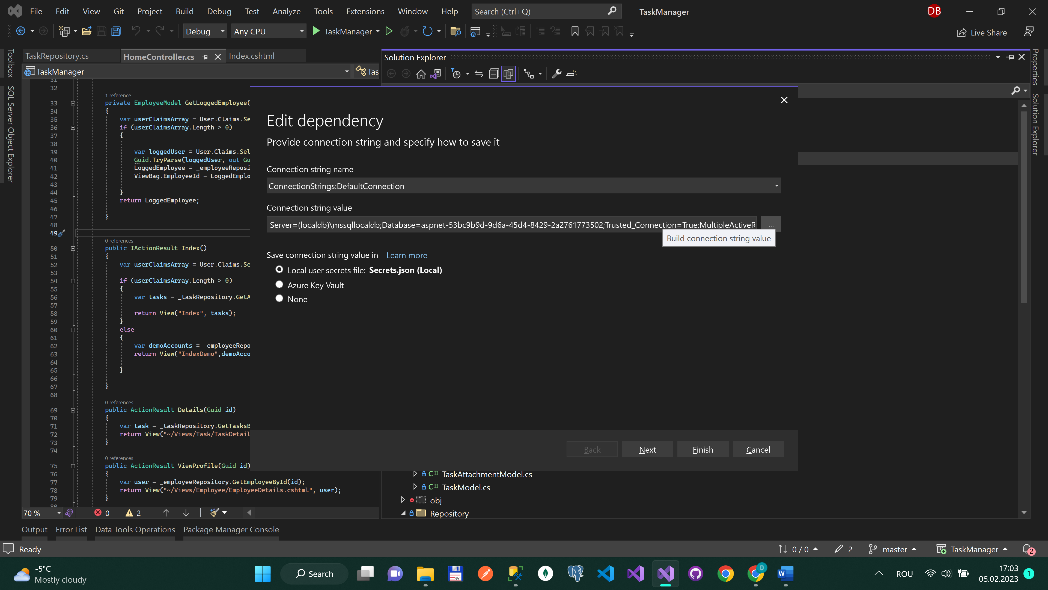
Pentru realizarea aplicației a fost realizată o bază de date ce cuprinde următoarele tabele:

* Departments;
* Employees;
* JobTitles;
* TaskAttachment;
* Tasks.

Diagrama bazei de date este următoarea:

Poza diagrama

A screenshot of a computer

Description automatically generatedConectarea la baza de date a fost realizată prin secțiunea *Connected Services*, dând click pe *Edit*.

A screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface, application

Description automatically generatedA screenshot of a computer

Description automatically generated with medium confidenceGraphical user interface

Description automatically generatedCrearea tabelelor a fost realizată utilizând secțiunea *SQL Server Object Explorer*. Tabelele create sunt prezentate în pozele următoare:

Graphical user interface

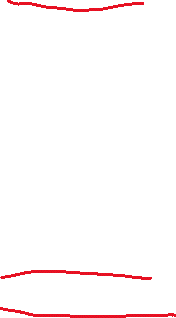
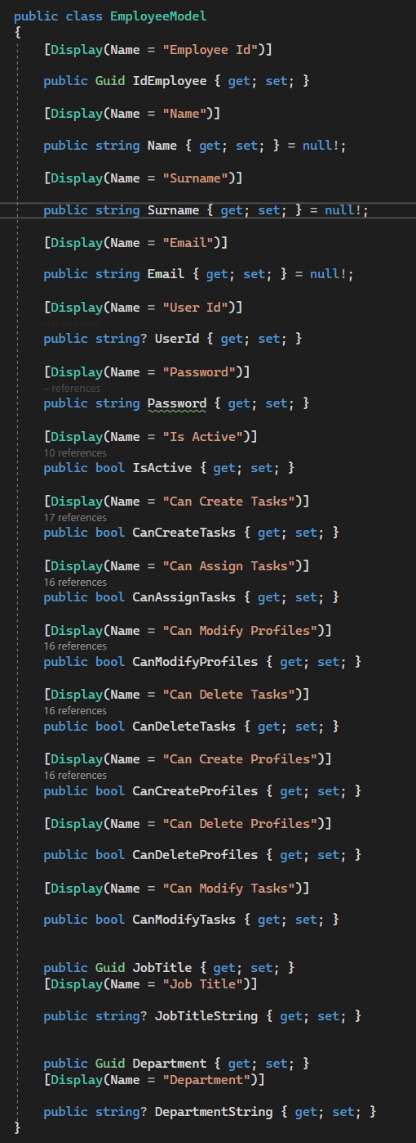
Description automatically generated

După realizarea conectarea la baza de date creată și generarea tabelelor, au fost create modelele într-un nou folder din *Models*, denumit *DBObjects*, prin rularea în *Package Manager Console* a comenzii următoare:

*Scaffold-DbContext Name=DefaultConnection Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models\DBObjects -ContextDir Data -Context ApplicationDbContext -f*

Text

Description automatically generatedDupă au fost create manual modelele utilizate în aplicație, acestea fiind asemănătoare celor create automat. Totuși există anumite diferențe necesare pentru interpretarea datelor.



În cazul prezentat mai sus, adăugarea definițiilor pentru *JobTitleString* și *DepartmentString* au fost necesare pentru afișarea acestora în tabele.

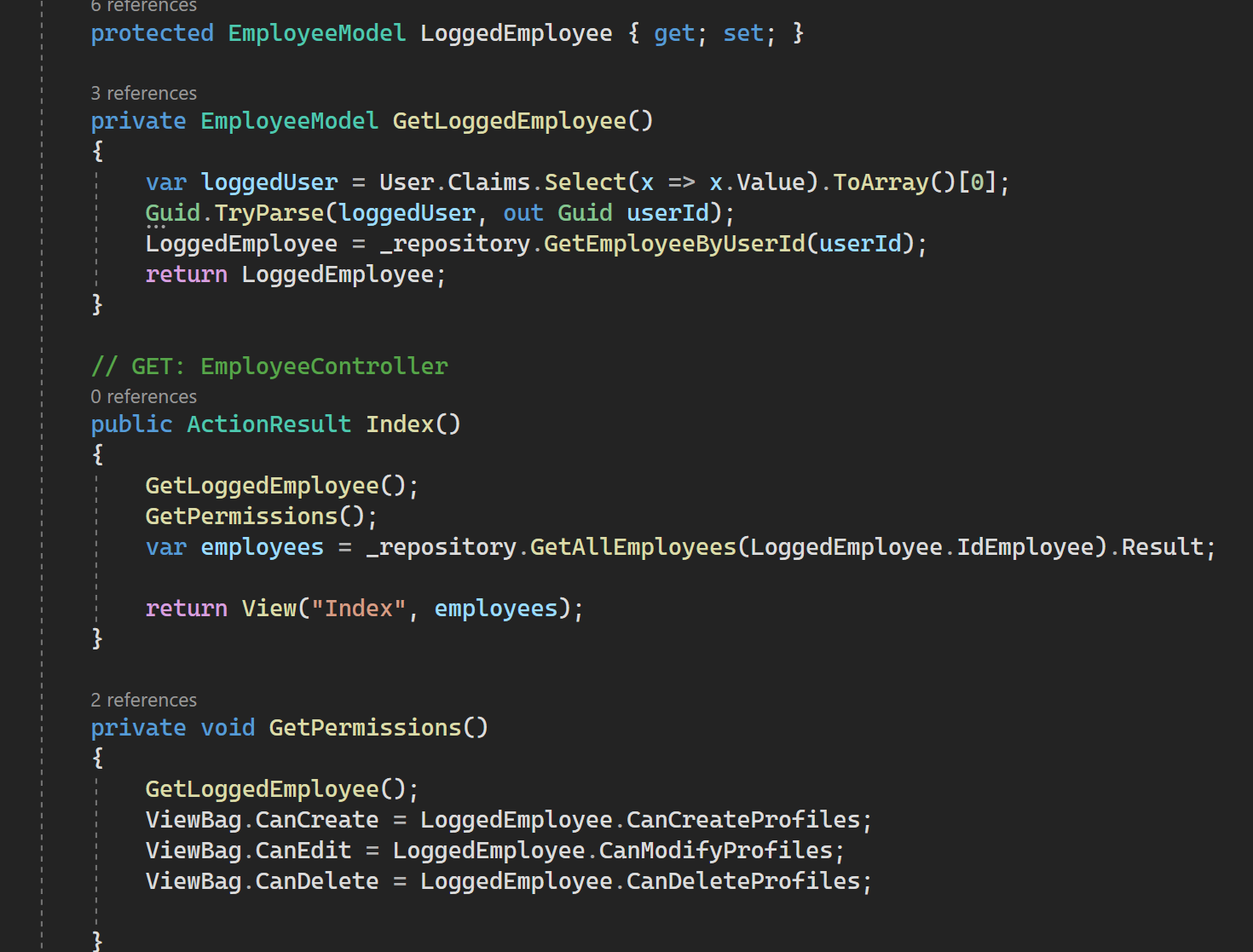
După crearea modelelor, au fost create clasele repository pentru fiecare tabel al bazei de date, acestea cuprinzând metodele necesare pentru manipularea datelor.

După crearea claselor repository, au fost create controller-ilor pentru fiecare repository, iar pentru fiecare controller – o serie de view-uri.

## EmployeeController

Pentru EmployeeController au fost create următoarele view-uri:

* *EmployeeCreate*
* *EmployeeDelete*
* *EmployeeDetails*
* *EmployeeEdit*
* *Index*

View-ul *Index* cuprinde lista angajaților, pagina fiind populată cu ajutorul metodei *Index*, care arată în felul următor:

Metoda *GetLoggedEmployee* este utilizată pentru determinarea angajatului logat, astfel determinând și permisiunile acestuia. Lista de angajați afișată în pagina *Index* nu cuprinde angajatul logat.

User-ul logat este eliminat la selectarea tuturor user-ilor, în metoda *GetAllEmpoyees.*

Text

Description automatically generatedPagina arată în felul următor:

Graphical user interface, text, application, email

Description automatically generatedÎn funcție de drepturile utilizatorului, acesta poate crea, edita, șterge datele unui angajat. Pentru fiecare dintre acțiunile respective există câte un view.

Pentru editare utilizatorul trebuie să dea click pe *Edit*, după care va fi redirecționat spre un formular ce cuprinde datele existente ale angajatului.

A screenshot of a computer

Description automatically generated with medium confidenceMetoda *SelectCategory* are ca scop selectarea valorilor pentru dropdown-urile *Department* și *Job Title.*

Graphical user interface, text, application, email

Description automatically generatedText

Description automatically generatedFormularul de editare arată în felul următor:

Salvarea datelor se face prin click pe *Save*:

Text

Description automatically generated

Pentru salvare, este invocată metoda *UpdateEmployee* care arată în felul următor:

Text

Description automatically generated