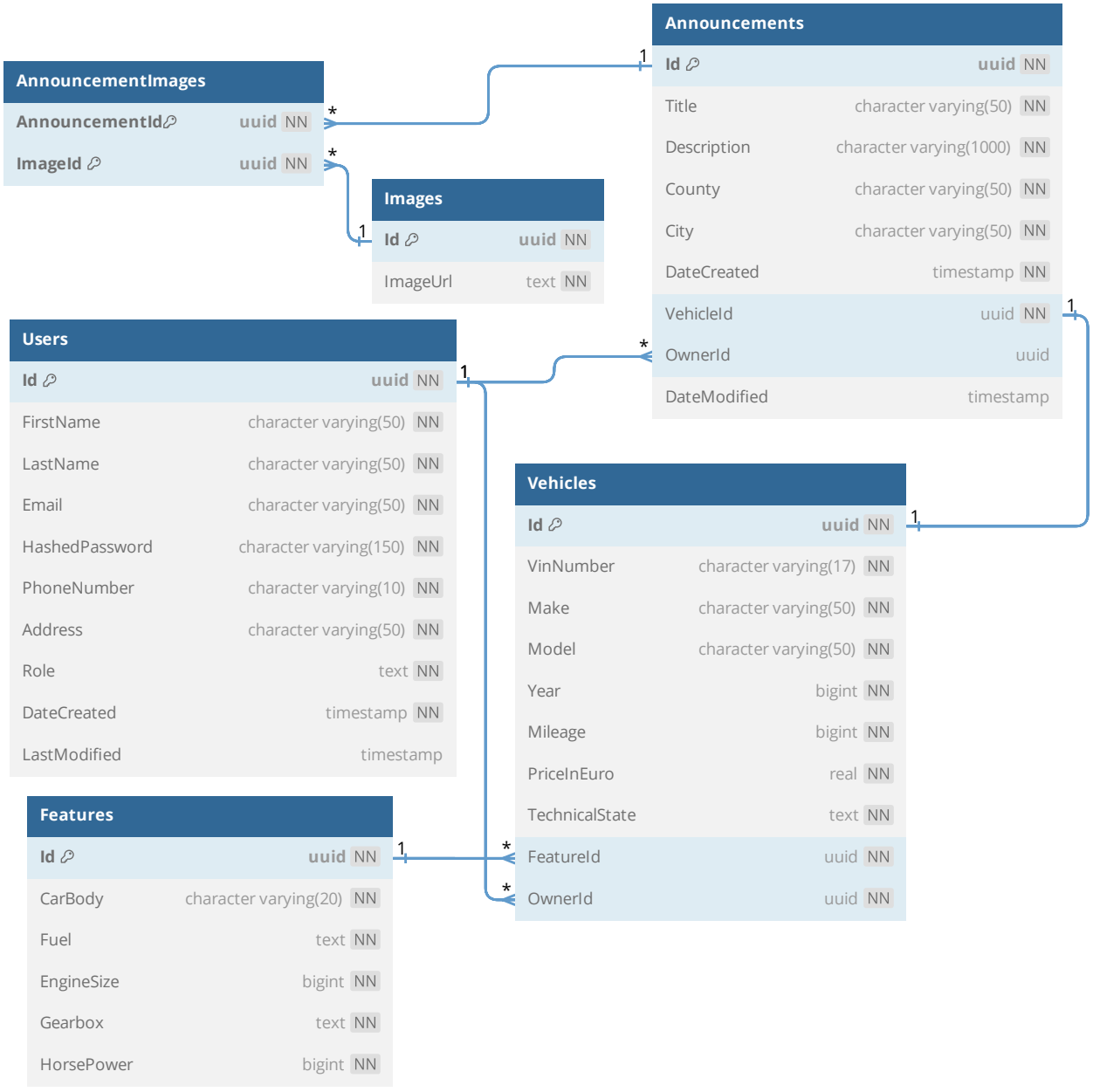
Enunt problema

Dezvoltarea unei aplicații web care să permită utilizatorilor să posteze și să caute anunțuri pentru mașini de vânzare. Aplicația ar trebui să ofere posibilitatea utilizatorilor de a crea conturi, de a introduce detalii despre mașinile pe care doresc să le vândă (inclusiv fotografii, specificații și prețuri), precum și de a căuta și filtra anunțurile existente în funcție de diferite criterii (cum ar fi marcă, model, an de fabricație, preț etc.). De asemenea, aplicația ar trebui să ofere o interfață prietenoasă și intuitivă, asigurând că utilizatorii pot interacționa ușor cu anunțurile și pot lua legătura unii cu alții în legătură cu potențialele tranzacții. În final, obiectivul este de a crea o platformă care facilitează procesul de cumpărare și vânzare a mașinilor și oferă o experiență plăcută utilizatorilor săi.

Diagrama bazei de date



Tehnologii folosite

Pentru Backend am folosit urmatoarele tehnologii:

- .NET 7 & ASP.NET CORE WEB API - pentru a permite creearea serviciilor web HTTP/RESTful. Acest framework facilitează construirea de API-uri (interfețe de programare a aplicațiilor) care pot fi accesate prin intermediul protocolului HTTP, oferind astfel comunicare între diverse aplicații sau platforme
- EntityFramework Core – un framework de tip ORM (Object-Relational Mapping) folosit pentru a facilita interacțiunea între bazele de date relaționale și aplicațiile software scrise în .NET. Scopul principal al EF Core este de a abstractiza și simplifica procesul de gestionare a datelor, permițând dezvoltatorilor să lucreze cu date ca și cum ar fi obiecte C#
- PostgreSQL - este un sistem de gestionare a bazelor de date relaționale (RDBMS) open-source și puternic cunoscut pentru performanța sa, extensibilitatea și caracteristicile avansate.
- JSON Web Token(JWT) Bearer - este un protocol de autentificare și autorizare utilizat în aplicațiile web și API-uri pentru a valida identitatea utilizatorilor și a asigura accesul securizat la resursele protejate. Acest protocol se bazează pe token-uri JSON semnate și/sau criptate
- Swagger - este o suită de instrumente open-source pentru proiectarea, documentarea și testarea API-urilor. Scopul principal al Swagger este de a simplifica procesul de dezvoltare și gestionare a API-urilor, permițând dezvoltatorilor să creeze documentație interactivă pentru API-urile lor și să testeze API-urile într-un mod convenabil.
- xUnit - framework de testare unitară open-source pentru limbajele de programare .NET. xUnit pune accent pe simplitate, modularitate și folosirea convențiilor pentru a ajuta dezvoltatorii să creeze și să ruleze teste unitare într-un mod eficient.
- FluentAssertions - bibliotecă open-source pentru testare în limbajul .NET care permite dezvoltatorilor să scrie assert-uri într-un stil fluent și expresiv. Scopul principal este de exprima clar comportamentul așteptat al codului testat.
- NSubstitute - o bibliotecă de simulare (mocking) pentru testare în limbajul .NET, care permite dezvoltatorilor să creeze obiecte simulate pentru a izola componentele testate și a verifica interacțiunile dintre acestea.

Pentru Frontend am folosit Angular este un framework open-source dezvoltat de Google pentru construirea aplicațiilor web moderne și dinamice. Acesta este scris în TypeScript și oferă un set complet de instrumente și funcționalități pentru dezvoltarea de aplicații web

Arhitectura

Pentru arhitectură am urmărit dezvoltarea unei aplicații web bazate pe o interfață de tip Single Page Application și un Web API RESTful orientat pe microservicii. Arhitectura orientată pe microservicii este un stil arhitectural în care aplicația este dezvoltată și organizată ca o colecție de servicii mici, independente și specializate, numite microservicii. Aceste microservicii sunt responsabile pentru funcționalități specifice și pot fi dezvoltate, implementate și scalate independent. Acest stil arhitectural oferă flexibilitate, scalabilitate și posibilitatea de a dezvolta și implementa fiecare componentă a aplicației independent. Am folosit urmatoarele servicii:

- AuthService – Acest microserviciu se ocupă de gestionarea autentificării și autorizării utilizatorilor. Furnizează funcționalități precum generare JWT, hashuirea parolelor pentru stocarea în interiorul

bazei de date, verificarea parolelor atunci când un utilizator încearcă să se conecteze, gestionarea rolurilor și permisiunilor.

- UserService - Scopul acestui microserviciu este de a gestiona utilizatorii din aplicație, inclusiv operațiuni precum înregistrarea, autentificarea, actualizarea și ștergerea utilizatorilor.
- VehicleService - se ocupă de gestionarea vehiculelor, inclusiv operații precum adăugarea, actualizarea și ștergerea vehiculelor, precum și de filtrarea detaliilor despre acestea.
- AnnouncementService - se ocupă de gestionarea anunțurilor legate de vehicule, inclusiv operații precum adăugarea, actualizarea și ștergerea anunțurilor, precum și filtrarea detaliilor despre acestea.

O aplicație SPA(Single Page Application) este un tip de aplicație web care rulează într-o singură pagină web și încarcă conținutul adițional pe aceeași pagină, fără a necesita încărcări complete ale paginii sau navigări între pagini separate. Într-o SPA, interacțiunea și navigarea utilizatorului sunt gestionate prin intermediul schimbărilor dinamic generare în cadrul aceleiași pagini,

Adnotări folosite:

- [ApiController] – folosit pentru a mapa o clasă drept controller în aplicație
- [Route("api/[controller]")] – folosit pentru redirecționa toate request-urile către controller
- [HttpGet] – folosit pentru a mapa un request de tip GET
- [HttpPost] - folosit pentru a mapa un request de tip POST
- [HttpPut] - folosit pentru a mapa un request de tip PUT
- [HttpDelete] - folosit pentru a mapa un request de tip DELETE
- [HttpPatch] – folosit pentru a mapa un request de tip PATCH
- [ProducesResponseType] – folosit pentru a documenta controller-ul în Swagger, informand utilizatorii despre ce response-uri ar putea returna acel endpoint
- [Authorize] – pentru a marca ca un endpoint poate fi folosit doar de catre utilizatorii cu anumite permisiuni
- [AllowAnonymous] – pentru a marca ca un endpoint poate fi folosit de catre toti utilizatorii aplicatiei
- [Required] – pentru a marca ca un camp este obligatoriu in interiorul unei clase
- [MaxLength] – pentru a arata ca un string poate avea un numar maxim de caractere
- [NotNull] – pentru a arata ca un camp al unei clase nu poate fi null
- [JsonIgnore] – pentru a nu serialize campul unei clase in documentul JSON