Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

DOCUMENTAȚIA PROIECTULUI

PropertyRental

STUDENȚI
Stan Ioana
Sandu Tudor

Universitatea Transilvania din Brașov
Facultatea de Matematică și Informatică

Cuprins

1. Prezentarea proiectului

PropertyRental este o aplicație web API pentru închirieri pe termen scurt (short-term stays), similară cu Airbnb. Aplicația permite utilizatorilor să își înregistreze proprietățile pentru închiriere și să efectueze rezervări pentru proprietățile altor utilizatori.

Probleme abordate:

- Facilitează conectarea între proprietarii de imobile și chiriaşii pe termen scurt
- Oferă un sistem de management al rezervărilor cu validări complexe
- Implementează un sistem de review-uri pentru calitatea serviciilor
- Automatizează calculul prețurilor și validarea disponibilității

2. Tehnologiile folosite

- **Backend Framework**: ASP.NET Core 9.0 (C#)
- **Baza de date**: SQLite cu Entity Framework Core 9.0
- **Autentificare**: ASP.NET Core Identity cu JWT Bearer tokens
- **Documentație API**: Swagger/OpenAPI (Swashbuckle.AspNetCore)
- **Middleware**: Middleware personalizat pentru gestionarea globală a excepțiilor
- **Securitate**: JWT (System.IdentityModel.Tokens.Jwt), bcrypt pentru criptarea parolelor

3. Baza de date

Diagrama bazei de date și relațiile:

Tabele principale:

- **ApplicationUser** (extinde IdentityUser)
- **Property** (proprietăți pentru închiriere)
- **Booking** (rezervări)
- **Review** (recenzii)

Relațiile:

- Un **ApplicationUser** poate avea multiple **Properties** (1:N)
- Un **ApplicationUser** poate avea multiple **Bookings** ca tenant (1:N)
- O **Property** poate avea multiple **Bookings** (1:N)
- Un **Booking** poate avea un singur **Review** (1:1)
- Un **ApplicationUser** poate primi multiple **Reviews** ca host (1:N)

Descrierea tabelelor:

ApplicationUser:

- Extinde ASP.NET Identity cu FirstName, LastName, Address, DateRegistered
- Relații: Properties (ca owner), Bookings (ca tenant), ReceivedReviews (ca host)

Property:

- Conține detalii complete despre proprietate: titlu; descriere; adresă; preț pe noapte
- Amenități: WiFi, parcare, bucătărie, mașină de spălat, aer condiționat
- Restricții: numărul minim/maxim de nopți, numărul maxim de oaspeți
- Indexuri pentru optimizarea căutărilor după oraș, preț, disponibilitate

Booking:

- Status-uri: Pending, Approved, Rejected
- Validări: date de început/sfârșit, conflicte cu alte rezervări
- Calculare automată a prețului total
- Restricții de anulare pentru rezervările aprobate

Review:

- Rating de la 1 la 5 stele
- Restricție: un singur review pentru fiecare rezervare
- Poate fi creat doar după finalizarea rezervării

4. Prezentarea API-ului

API-ul oferă următoarele endpoint-uri organizate în 4 controllere principale:

AuthController (/api/auth)

- **POST /register** – Înregistrarea utilizatorilor noi
- **POST /login** – Autentificarea utilizatorilor existenți

PropertiesController (/api/properties)

- **GET /** – Listarea proprietăților cu filtrare și paginare
- **GET /{id}** – Detalii proprietate specifică
- **POST /** – Crearea unei proprietăți noi (autentificare necesară)
- **PUT /{id}** – Actualizarea proprietății (doar owner-ul)
- **DELETE /{id}** – Ștergerea proprietății (doar owner-ul)

BookingsController (/api/bookings)

- **POST /** – Crearea unei rezervări noi
- **GET /{id}** – Detalii rezervare specifică
- **GET /my-bookings** – Rezervările utilizatorului curent
- **GET /property/{propertyId}** – Rezervările pentru o proprietate (doar owner)
- **PUT /{id}/status** – Aprobare/respingere rezervare (doar owner-ul proprietății)
- **DELETE /{id}** – Anularea rezervării (doar tenant-ul)

ReviewsController (/api/reviews)

- **POST /** – Crearea unui review nou
- **GET /{id}** – Detalii review specific
- **GET /host/{hostId}** – Toate review-urile unui host cu rating mediu
- **GET /booking/{bookingId}** – Review-ul unei rezervări specifice

Caracteristici importante ale API-ului:

- Autentificare JWT pentru endpoint-urile protejate
- Validări complexe pentru rezervări (conflicte de date, restricții proprietate)
- Filtrare avansată pentru proprietăți (oraș, preț, număr de oaspeți etc.)
- Paginare pentru listări
- Middleware pentru gestionarea globală a excepțiilor
- Swagger UI integrat pentru testare și documentație

5. Utilizarea aplicației

Tipuri de utilizatori:

1. Utilizatori neautentificați:

- Pot naviga și căuta proprietăți
- Pot vizualiza detaliile proprietăților
- Nu pot face rezervări sau adăuga proprietăți

2. Utilizatori autentificați (Guests/Tenants):

- Toate drepturile utilizatorilor neautentificați
- Pot face rezervări pentru proprietăți
- Pot vedea și gestiona rezervările proprii
- Pot anula rezervările (în anumite condiții)
- Pot scrie reviews după finalizarea rezervărilor

3. Proprietari (Hosts/Owners):

- Toate drepturile utilizatorilor autentificați
- Pot adăuga, edita și șterge proprietățile proprii
- Pot aproba sau respinge rezervările pentru proprietățile lor
- Pot vizualiza toate rezervările pentru proprietățile lor
- Primesc reviews de la oaspeți

Workflow-ul aplicației:

1. **Înregistrare/Autentificare** – utilizatorii se înregistrează și primesc JWT token
2. **Adăugare proprietăți** – proprietarii își adaugă imobilele cu detalii complete
3. **Căutare și rezervare** – oaspeții caută și rezervă proprietăți
4. **Aprobare rezervări** – proprietarii aprobă sau resping rezervările
5. **Review-uri** – după finalizarea rezervărilor, oaspeții pot lăsa recenzii

Securitate și validări:

- Parolele sunt criptate cu Identity
- JWT tokens pentru autentificare
- Validări pentru conflicte de rezervări
- Restricții de ownership pentru operațiuni CRUD
- Validări pentru review-uri (doar după rezervări finalizate)

6. Concluzii și contribuții

Împărțirea task-urilor în echipă

Ioana:

- Configurarea proiectului ASP.NET Core și structura de bază
- Implementarea sistemului de autentificare cu ASP.NET Core Identity și JWT
- Dezvoltarea AuthController-ului (register/login)
- Crearea modelelor de date (ApplicationUser, Property, Booking, Review)
- Configurarea ApplicationDbContext și relațiilor Entity Framework
- Implementarea PropertiesController-ului cu toate operațiunile CRUD
- Dezvoltarea sistemului de filtrare avansată și paginare pentru proprietăți
- Implementarea middleware-ului pentru gestionarea globală a excepțiilor
- Configurarea Swagger pentru documentația API
- Optimizarea performanței cu indexuri pentru căutări

Tudor:

- Implementarea BookingsController-ului cu logica complexă de rezervări
- Dezvoltarea sistemului de validare pentru conflicte de rezervări
- Implementarea calculării automate a prețurilor totale
- Crearea ReviewsController-ului cu toate restricțiile business
- Dezvoltarea logicii pentru aprobare/respingere rezervări
- Implementarea sistemului de review-uri cu validări (doar după rezervări completate)
- Crearea și rularea migrațiilor pentru baza de date
- Dezvoltarea DataSeeder-ului pentru date de test
- Testing și debugging pentru toate endpoint-urile
- Optimizarea query-urilor pentru performanță

Realizări ale proiectului

Acest proiect demonstrează implementarea unei aplicații complexe de rezervări cu:

- Arhitectură API REST well-structured
- Sistem complet de autentificare și autorizare
- Bază de date relațională cu relații complexe
- Validări business complexe
- Gestionare centralizată a erorilor
- Documentație API integrată

Tehnologii și concepte învățate:

- ASP.NET Core Web API development
- Entity Framework Core cu Code-First approach
- ASP.NET Core Identity pentru user management
- JWT authentication și authorization
- Complex business logic validation
- RESTful API design principles
- Database design cu relații one-to-many și one-to-one
- Middleware development pentru cross-cutting concerns
- API documentation cu Swagger/OpenAPI

Provocări tehnice rezolvate:

- Gestionarea conflictelor de rezervări cu validări complexe pe intervale de timp
- Calcularea automată a prețurilor cu validarea restricțiilor proprietății
- Implementarea sistemului de review-uri cu reguli stricte de business
- Optimizarea query-urilor cu indexuri strategice
- Middleware pentru gestionarea globală a excepțiilor
- Securizarea API-ului cu JWT și autorizare bazată pe roluri
- Design eficient al bazei de date pentru interogări complexe

Colaborare și metodologie:

- Utilizarea Git pentru version control și colaborare
- Code review mutual pentru calitatea codului
- Împărțirea responsabilităților pe domenii funcționale
- Testing colaborativ și debugging în echipă
- Documentarea comună a API-ului și logicii de business

7. Link Git către codul proiectului

https://github.com/eexxio/PropertyRental