HealthTech

Cuprins

1. Introducere

Scopul documentației: Descrierea scopului și audienței documentației.

Prezentarea generală a aplicației: O descriere scurtă a aplicației, inclusiv funcționalitatea și scopul acesteia.

2. Configurarea mediului de dezvoltare

Cerințe de sistem: Specificațiile hardware și software necesare.

Instalarea dependențelor.

3. Arhitectura aplicației

Diagrama arhitecturii: O diagramă care prezintă structura aplicației și relațiile dintre module.

Descrierea componentelor: Detalii despre module, clase, metode și funcționalități specifice.

4. Ghid de programare

Standarde de codificare: Convenții de numire, stiluri de codificare și practici recomandate.

Exemple de cod: Snippet-uri de cod care ilustrează utilizarea API-urilor și a funcțiilor cheie.

5. Managementul bazelor de date

Schema bazei de date: Diagrama și descrierea structurii bazei de date.

Instrucțiuni pentru migrații: Proceduri pentru actualizarea sau modificarea schemei bazei de date.

6. Testare și Depanare

Strategii de testare: Metodologii și tipuri de teste utilizate (unit testing, integration testing etc.).

Instrucțiuni de depanare: Sfaturi și tehnici pentru identificarea și rezolvarea problemelor.

1. Introducere

Scopul Documentației

Documentația are scopul de a oferi un set comprehensiv de informații și ghiduri pentru a facilita înțelegerea, dezvoltarea și întreținerea continuă a proiectului.

Furnizarea unui instrument de referință pentru orice dezvoltator sau membru al echipei care interacționează cu aplicația.

Descrierea Proiectului: Sistem de Gestiune a Programărilor într-o Clinică Medicală

Proiectul constă în dezvoltarea unei aplicații web avansate pentru gestionarea eficientă a programărilor în cadrul unei clinici medicale. Utilizatorii au posibilitatea de a-și crea un cont personal și de a se autentifica pentru a accesa funcționalitățile oferite de aplicație.

Funcționalități Cheie:

1.Autentificare și Creare Cont:

Utilizatorii pot să-și creeze un cont personal și să se autentifice pentru a beneficia de funcționalități personalizate.

2.Programare la Doctor:

Utilizatorii pot programa întâlniri la un doctor specific din cadrul unei anumite specializări. Interfață intuitivă pentru vizualizarea intervalului orar și a zilei de disponibilitate a fiecărui doctor.

3.Gestionarea Programărilor:

Utilizatorii pot vizualiza detaliile programărilor curente și anterioare din contul lor personal.

4.Puncte și Beneficii:

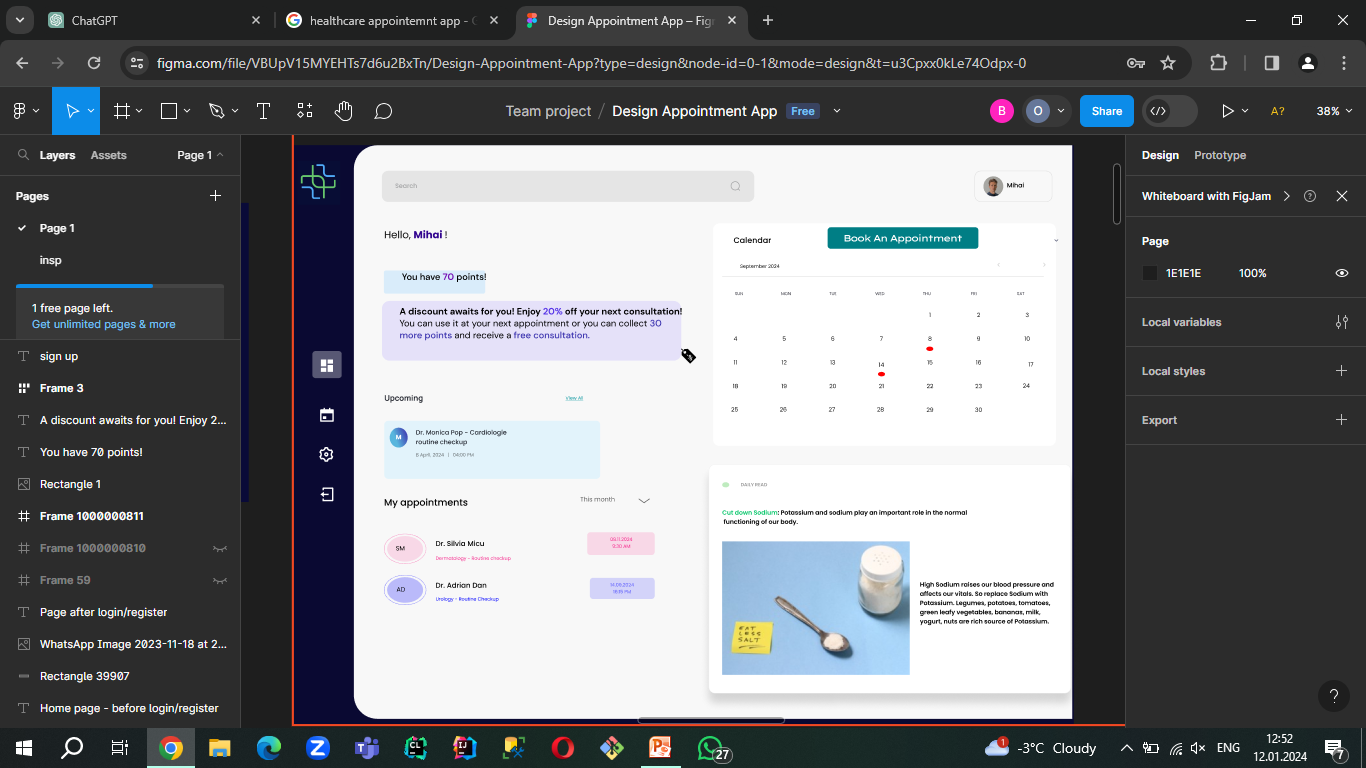
Pentru fiecare programare realizată, utilizatorilor li se alocă puncte.

Punctele acumulate generează beneficii sub forma unor discounturi, vizualizabile în contul personal al utilizatorului.

Detalii Despre Personalul Clinic :

Informații despre Medici: Detaliate pentru fiecare medic din cadrul clinicii .

Servicii Medicale Oferite: Descrieri ale serviciilor medicale furnizate de clinică.



A screenshot of a computer

Description automatically generated

Obiectivul Principal:

Scopul aplicației este de a oferi o soluție eficientă și accesibilă pentru gestionarea programărilor în cadrul unei clinici medicale. Echipa de dezvoltare a luat în considerare soluțiile existente pe piață, implementând funcționalități inovatoare precum sistemul de puncte pentru a încuraja utilizatorii să beneficieze de serviciile oferite de clinică.

Public Țintă al Documentației:

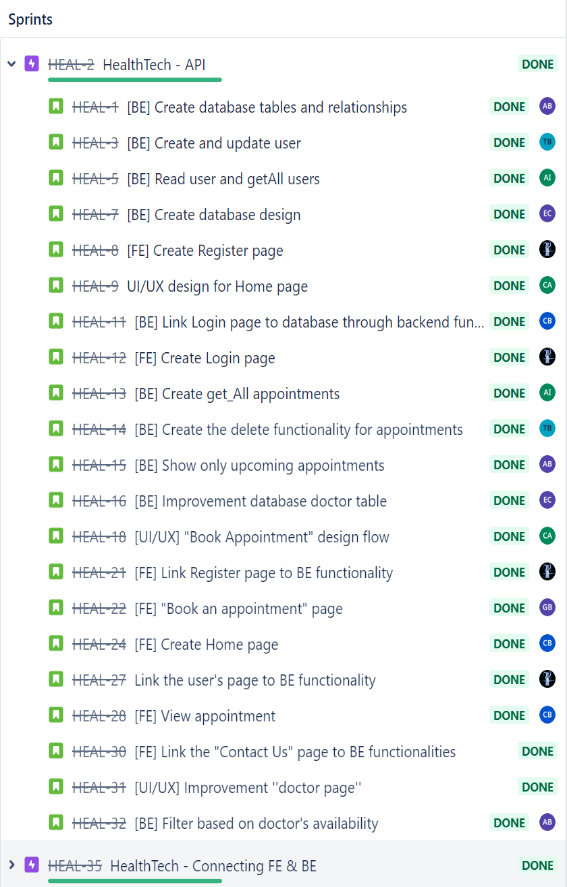
Documentația este destinată unei viitoare echipe de dezvoltare, in eventualitatea preluarii proiectul. Ea oferă o perspectivă cuprinzătoare asupra arhitecturii, structurii și comportamentului aplicației, asigurând o tranziție fără probleme și continuarea dezvoltării cu succes.



2. Configurarea mediului de dezvoltare

Cerințe de sistem: Specificațiile hardware și software necesare.

Instalarea dependențelor.

* Workload management: *Jira, GitHub*

**Sprint 1:**

* app’s database
* CRUD for user
* Login page (design + implementation)
* Register page (design + implementation)

**Sprint 2:**

* Book appointment and view appointments functionalities
* Filters and controllers
* Home page, User’s page, Contact Us page
* (design + implementation)

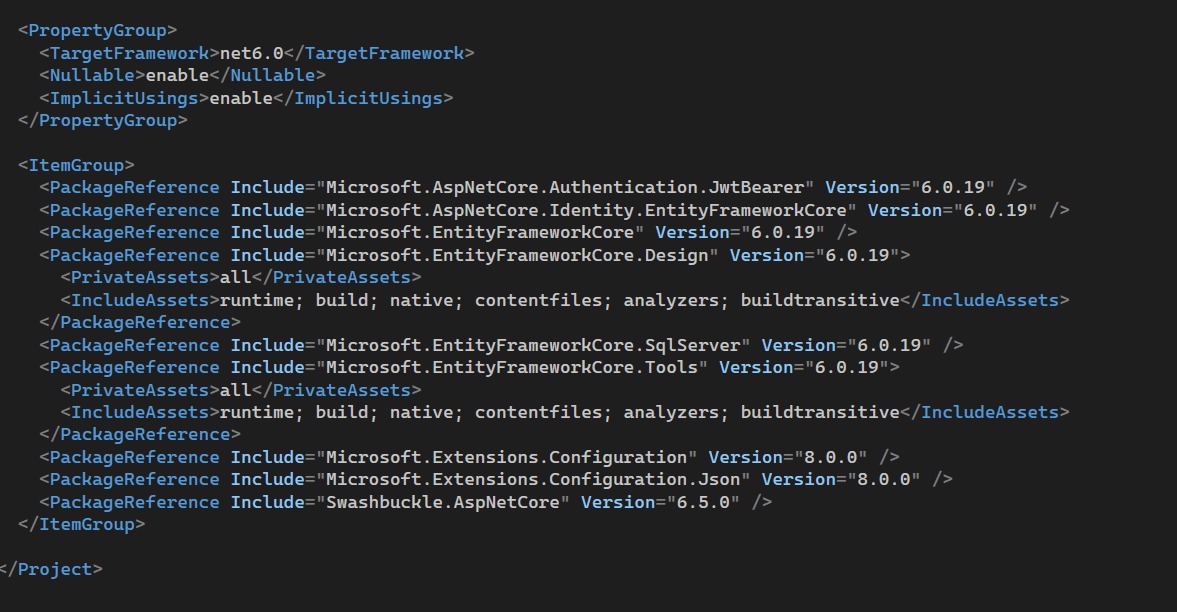
A screenshot of a computer

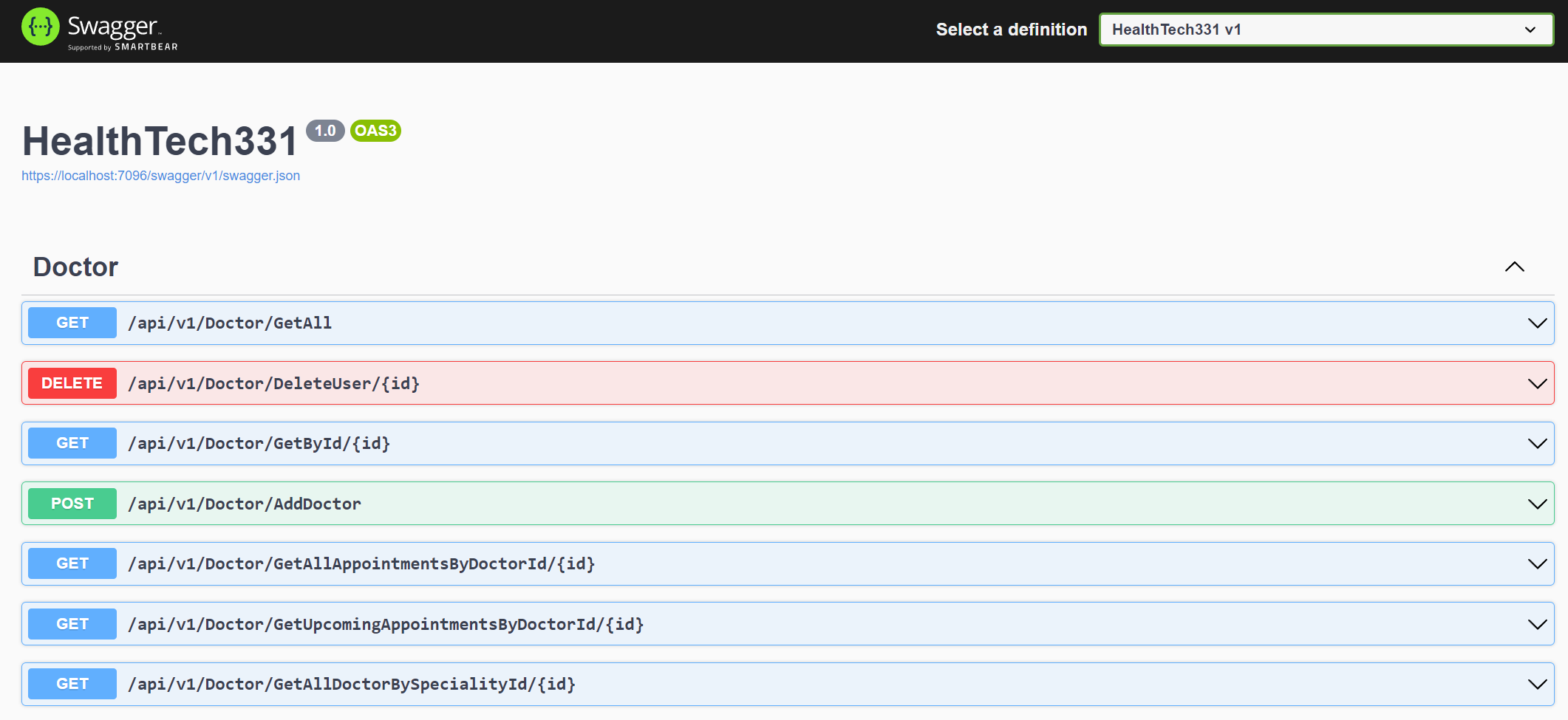
Description automatically generated

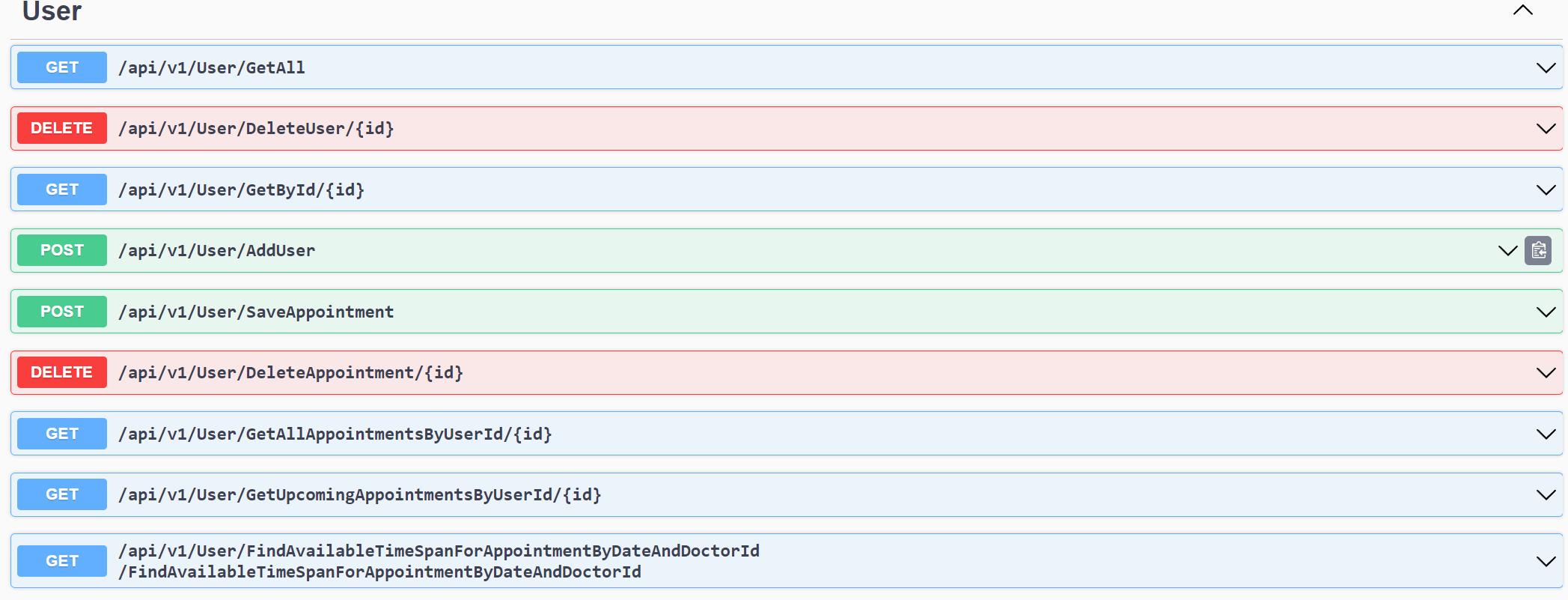
**Sprint 3:**

* Bonus points functionality
* Points page (design + implementation)
* App documentation

* UI/UX design: *Figma*
* Front-end: *VSC, CSS, Vanilla Javascript, React*
* Back-end: *.Net, C#, SQL Server Management Studio, Swagger*







3. Arhitectura aplicației

1. Interfața Utilizator (UI):
   * Interacțiunea cu utilizatorii pentru programări, vizualizarea informațiilor medicale și gestionarea conturilor.
2. Backend (Server-Side):
   * Servicii Web/API-uri:
     + Gestionarea cererilor de la interfața utilizatorului.
     + Procesarea și validarea datelor primite de la utilizatori.
     + Comunicarea cu baza de date pentru a salva și recupera date.
   * Business Logic:
     + Gestionarea logicii de afaceri pentru funcționalități precum programările, gestionarea utilizatorilor și doctorilor.
     + Aplicarea regulilor de afaceri, cum ar fi restricțiile de puncte pentru utilizatori.
   * Bază de Date:
     + Stocarea persistentă a datelor despre utilizatori, programări, doctori și specialități.
     + Menținerea integrității referențiale între entitățile diferite.
3. Autentificare:
   * Modul de autentificare pentru a se asigura că doar utilizatorii autorizați au acces la datele lor.

Descrierea Componentelor:

1. Modulul de Autentificare și Gestionare a Utilizatorilor

* Clase:
  + UserController: Furnizează funcționalități de autentificare și gestionare a sesiunilor.
  + UserRepository: Se ocupă de interacțiunea cu baza de date pentru operațiuni legate de utilizatori.
* Funcționalități:
  + Înregistrarea și autentificarea utilizatorilor (pacienți).
  + Gestionarea și actualizarea profilurilor utilizatorilor.
  + Adăugarea, modificarea și ștergerea pacienților din sistem.
  + Programarea consultațiilor și a investigațiilor.

2. Modulul de Gestionare a Personalului Medical

* Clase:
  + DoctorController: Furnizează funcționalități pentru gestionarea activităților personalului medical.
  + SpecialityRepository: Clasa pentru definirea specialităților medicale ale personalului medical.
* Funcționalități:
  + Gestionarea listei de medici și a specialităților acestora.
  + Programarea activităților medicale și monitorizarea resurselor alocate fiecărui medic.

3. Modulul de Gestionare a Programărilor

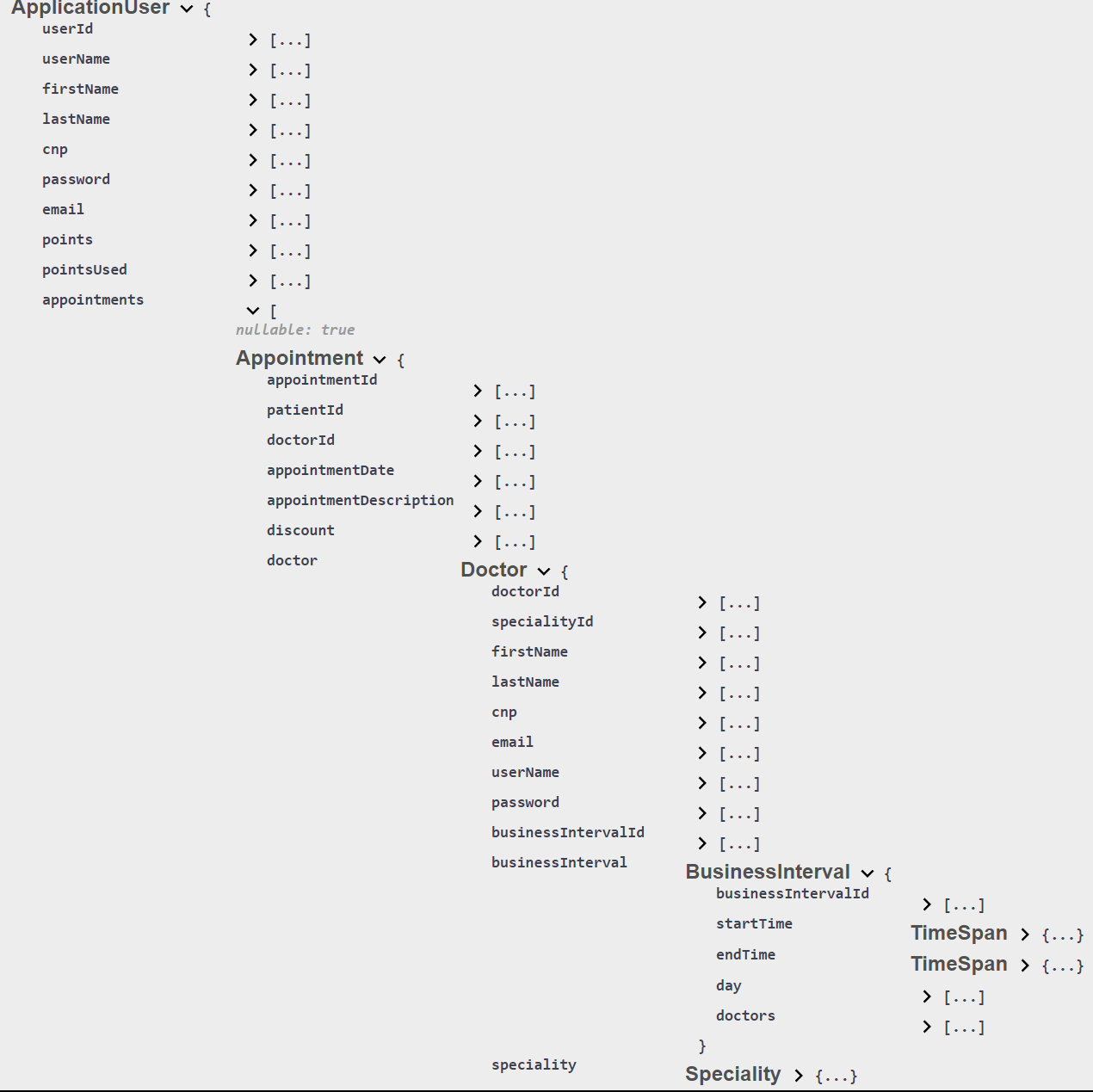
* Clase:
  + AppointmentRepository: Furnizează funcționalități pentru gestionarea programărilor pacienților.
* Funcționalități:
  + Crearea și ștergerea programărilor pentru pacienți.
  + Generarea de notificări pentru amintiri legate de programări.

A screenshot of a computer

Description automatically generated

Considerații Arhitecturale:

* Arhitectură de Nivel Înalt: Aplicația este concepută folosind o arhitectură stratificată, unde nivelurile de prezentare, logică de afaceri și acces la date sunt bine separate.
* Comunicare între Module: Modulele comunică între ele utilizând interfețe bine definite, reducând astfel cuplajul dintre ele.
* Securitate: Accesul la datele sensibile este protejat prin mecanisme de autentificare și autorizare. Datele sensibile sunt stocate și transferate în mod securizat.
* Extensibilitate: Arhitectura este concepută pentru a permite extensibilitatea ușoară, astfel încât noi funcționalități să poată fi adăugate cu minim de efort.





5. Managementul bazelor de date

1. Diagrama

A screenshot of a computer

Description automatically generated

2. Descrierea Structurii

Baza de date a aplicației pentru clinica privată este proiectată să gestioneze informații despre utilizatori, pacienți, medici, programări și alte entități relevante. Structura bazei de date include următoarele tabele principale:

1. Tabelul ApplicationUser:
   * UserId: Cheie primară, identificator unic pentru utilizator.
   * UserName, FirstName, LastName, CNP, Password, Email: Informații despre utilizator.
2. Tabelul Speciality:
   * SpecialityId: Cheie primară, identificator unic pentru specialitate.
   * SpecialityName, SpecialityDescription: Informații despre specialitate.
3. Tabelul Doctor:
   * DoctorId: Cheie primară, identificator unic pentru doctor.
   * SpecialityId: Cheie externă către Speciality.
   * FirstName, LastName, CNP, Email, Password, UserName: Informații despre doctor.
   * BusinessIntervalId: Cheie externă către BusinessInterval.
4. Tabelul BusinessInterval:
   * BusinessIntervalId: Cheie primară, identificator unic pentru intervalul de lucru.
   * StartTime, EndTime, Day: Informații despre intervalul de lucru.
5. Tabelul Appointment:
   * AppointmentId: Cheie primară, identificator unic pentru programare.
   * PatientId, DoctorId: Chei externe către ApplicationUser și Doctor.
   * AppointmentDate, AppointmentDescription: Informații despre programare.
   * Discount

Relațiile:

* Relația între ApplicationUser și Appointment:
  + Tip de relație: One-to-Many
  + Sensul relației: Un utilizator poate avea mai multe programări, dar o programare este asociată cu un singur utilizator (pacient).
* Relația între Doctor și Appointment:
  + Tip de relație: One-to-Many
  + Sensul relației: Un doctor poate avea mai multe programări, dar o programare este asociată cu un singur doctor.
* Relația între Speciality și Doctor:
  + Tip de relație: One-to-Many
  + Sensul relației: O specialitate poate fi asociată cu mai mulți doctori, dar un doctor are o singură specialitate.
* Relația între BusinessInterval și Doctor:
  + Tip de relație: One-to-Many
  + Sensul relației: Fiecare doctor are un interval orar, iar fiecare interval orar este asociat unuia sau mai multor doctori.