



# Proiectarea aplicatiilor web

## Proiect Individual

Dental Clinic Website

**Nedelcu Robert-Ionut CR3.2B**  
**04/06/2023**

# 1. Descrieti rolul si obiectivele aplicatiei. Specificati in detaliu cerintele aplicatiei.

## Rolul aplicatiei

Aplicația pentru site-ul cabinetului stomatologic “Smile Dent” are rolul de a oferi o experiență eficientă și ușor de utilizat atât pacienților care vor să se programeze, să vadă serviciile și tratamentele cât și membrii echipei și alte detalii oferite despre cabinetul stomatologic, cât și personalului medical care pot avea o evidență a programărilor.

## Obiective

### Programare online:

Aplicația trebuie să permită pacienților să se programeze rapid și ușor prin intermediul site-ului, acesta fiind o funcționalitate foarte importantă deoarece din ce am observat multe site-uri aflate în folosință nu dispun de aceasta. Pacienții ar trebui să aibă posibilitatea de a alege data și ora disponibilă și de a primi confirmarea programării printr-un mesaj.

### Informații despre servicii:

Aplicația trebuie să furnizeze informații detaliate despre serviciile oferite de cabinetul stomatologic. Aceasta ar trebui să includă descrieri ale procedurilor stomatologice, costuri estimative, fotografii înainte și după și eventuale oferte și tarifele corespunzătoare.

### Contact și suport:

Aplicația trebuie să ofere opțiuni de contact și suport pentru pacienți. Acestea ar trebui să includă număr de telefon, e-mail și un formular de contact prin care pacienții pot trimite întrebări sau solicitări de asistență.

### Informații despre personalul medical:

Aplicația web ar trebui să ofere informații detaliate despre medicii și personalul stomatologic din cabinet. Aceasta poate include fotografii, experiență, specializări și recenzii ale pacienților.

Securitate și confidențialitate:

Site-ul trebuie să asigure o securitate solidă a datelor pacienților și să respecte standardele de confidențialitate privind informațiile medicale. Aceasta implică utilizarea unor măsuri de securitate, cum ar fi criptarea datelor și protejarea împotriva accesului neautorizat.

Design intuitiv și responsiv:

Trebuie să fie ușor de utilizat și să ofere o experiență intuitivă pentru utilizatori. Interfața grafică ar trebui să fie atractivă, să fie ușor de navigat și să se adapteze la diferite dispozitive, cum ar fi computerele, telefoanele mobile și tabletele.

## **2. Modelati aplicatia folosind diagrame UML.**

## Diagrama de cazuri de utilizare

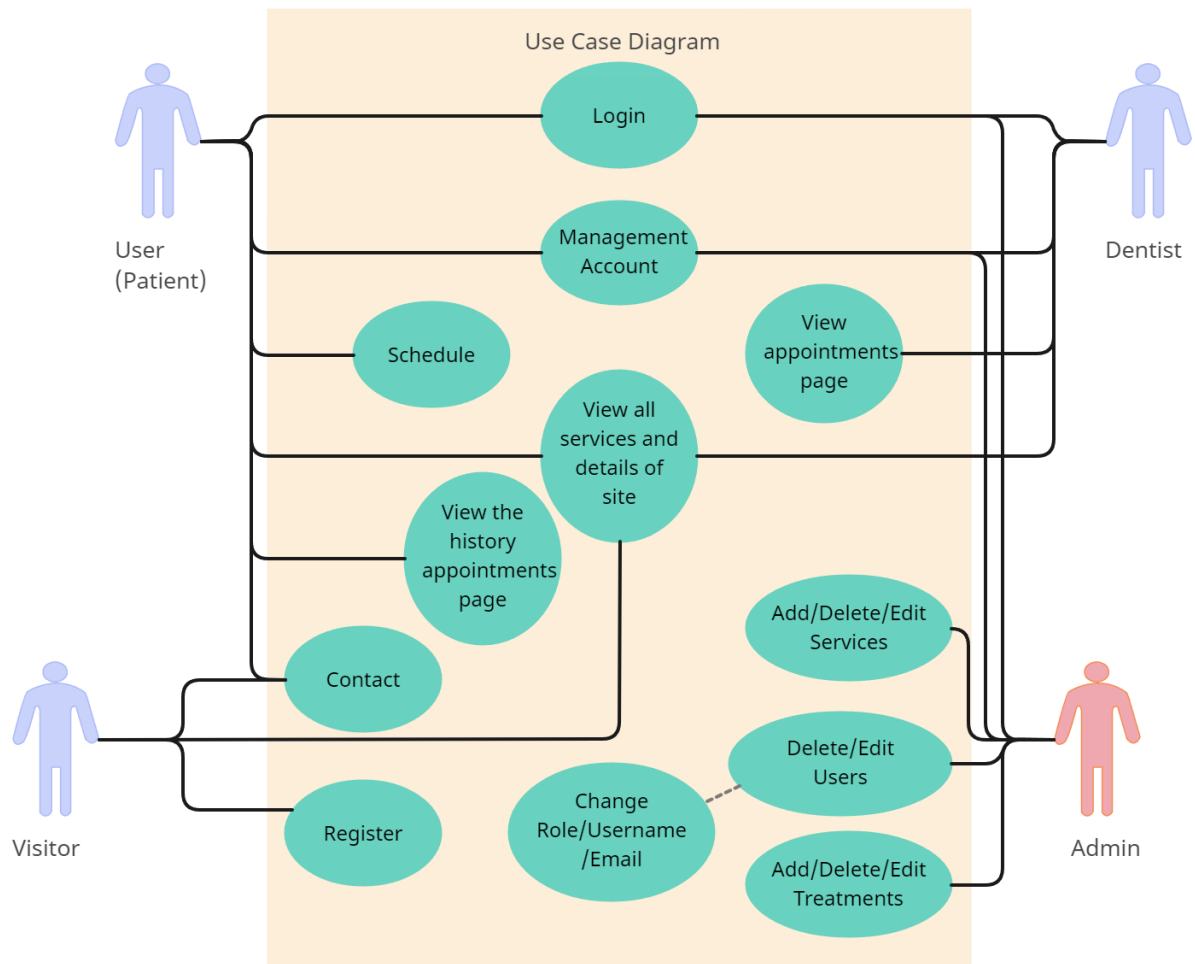
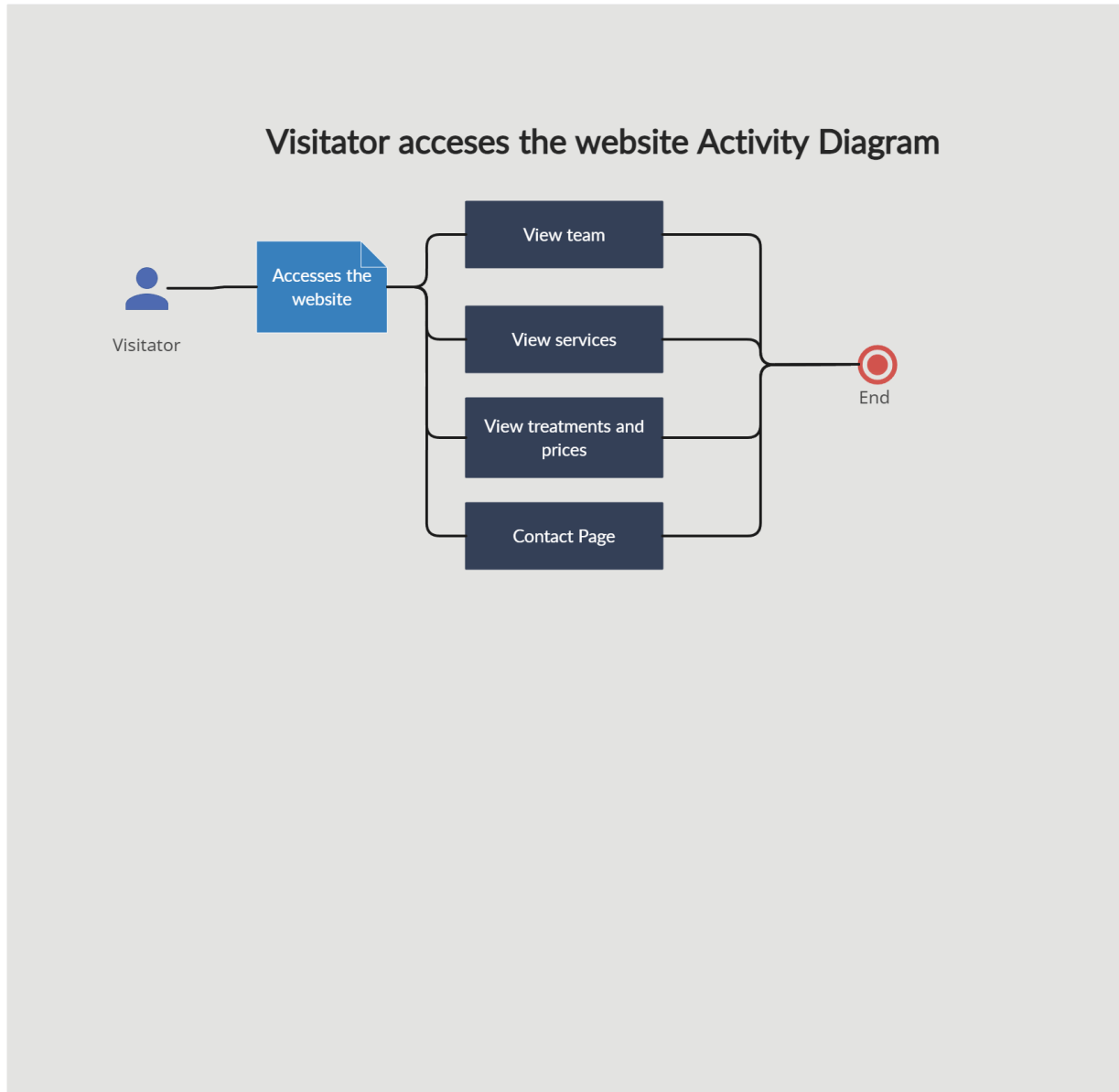
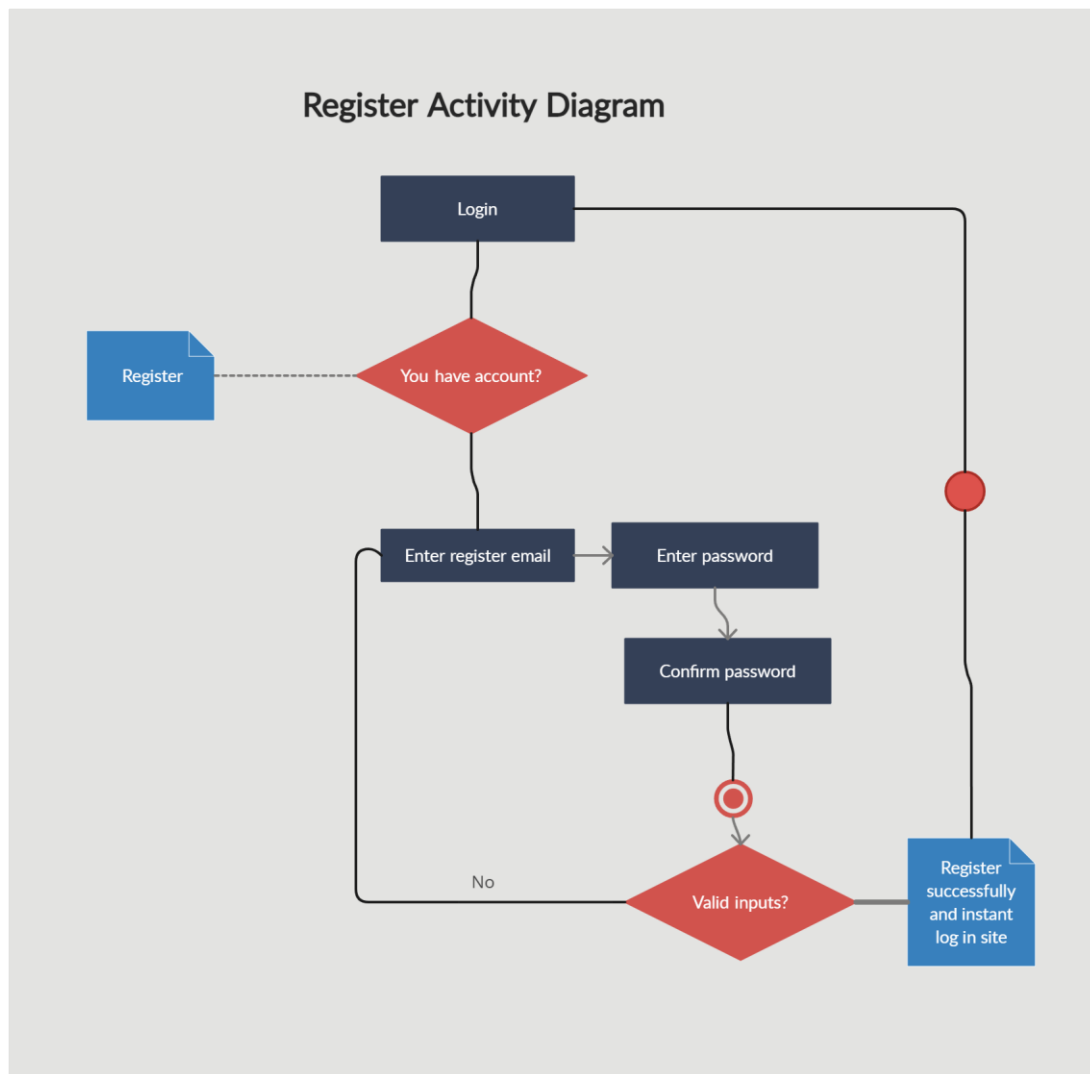


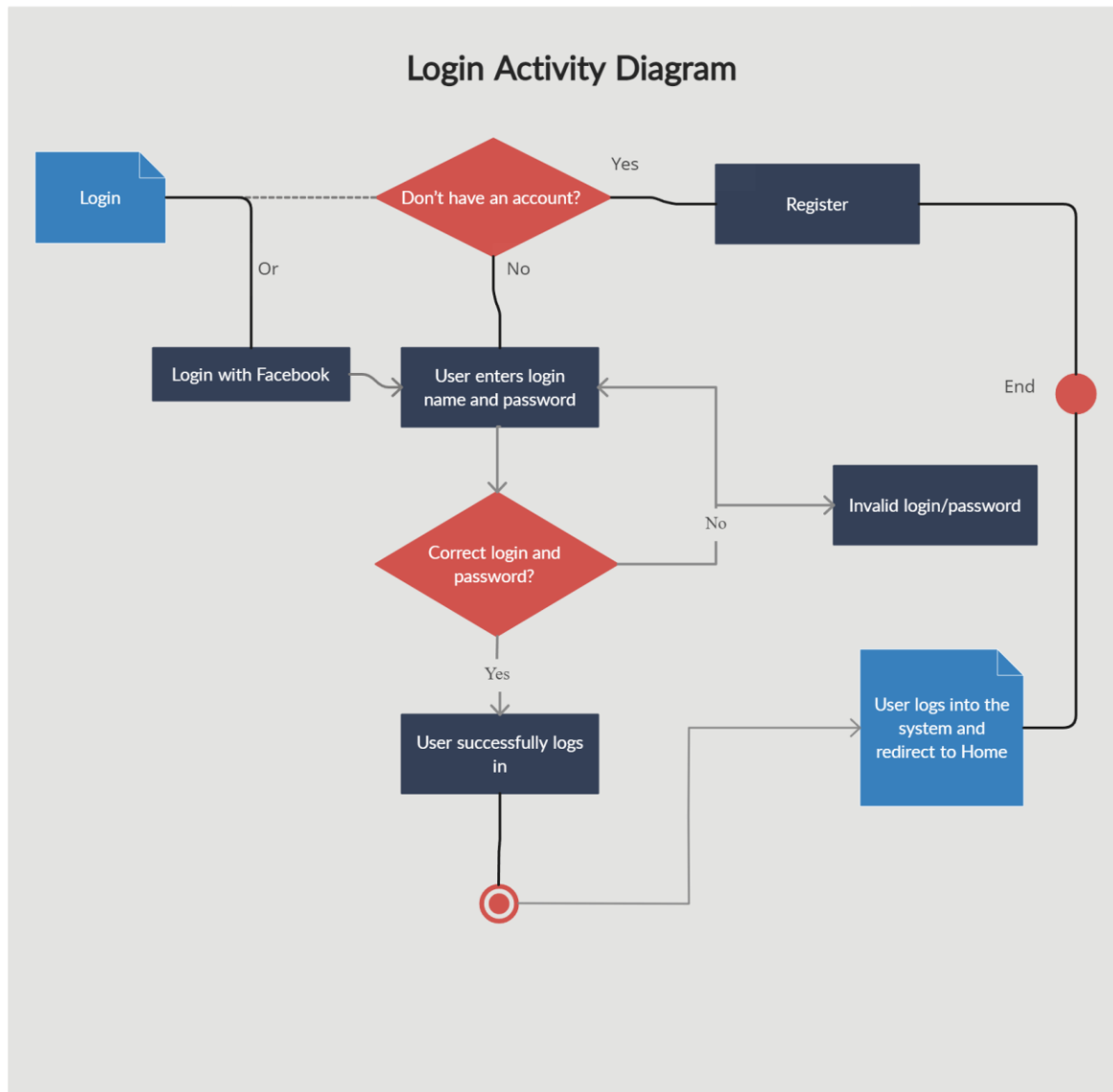
Diagrama de activitati pentru accesarea site-ului de catre un vizitator



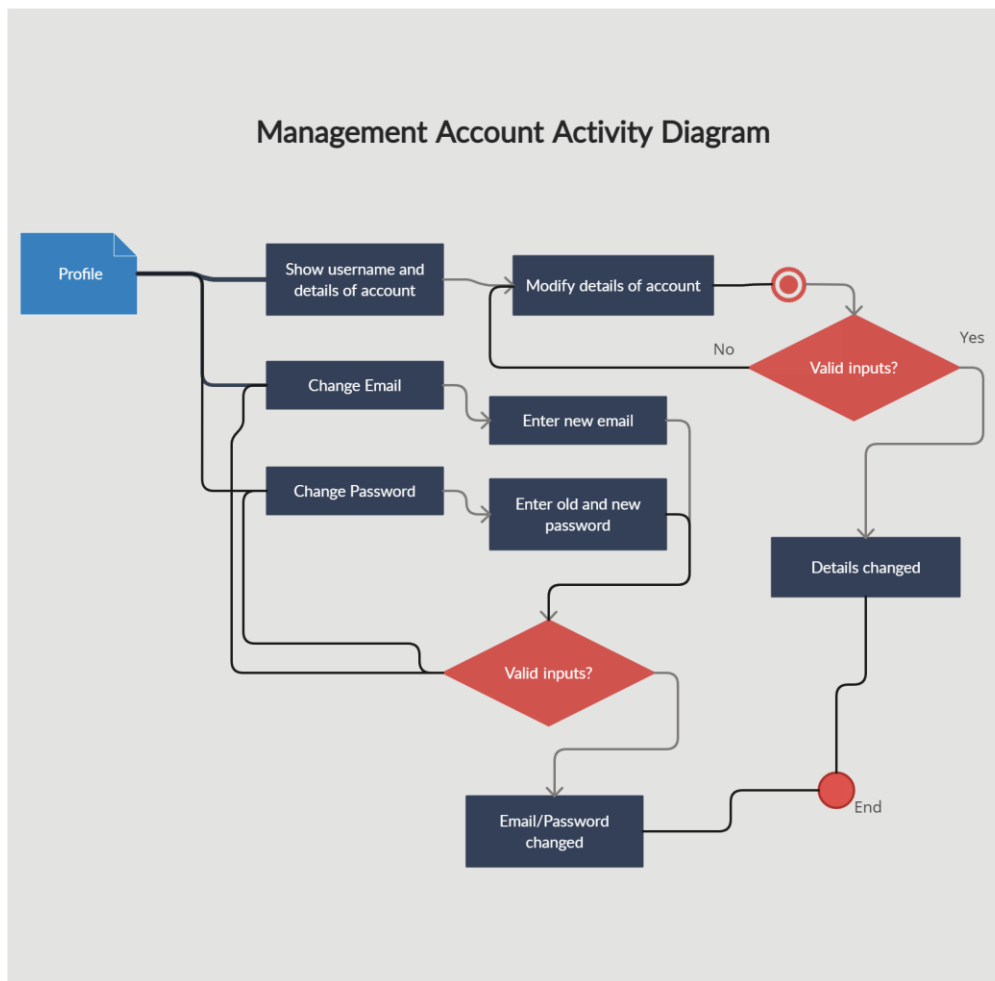
## Diagrama de activitati pentru inregistrare



## Diagrama de activitati pentru logare

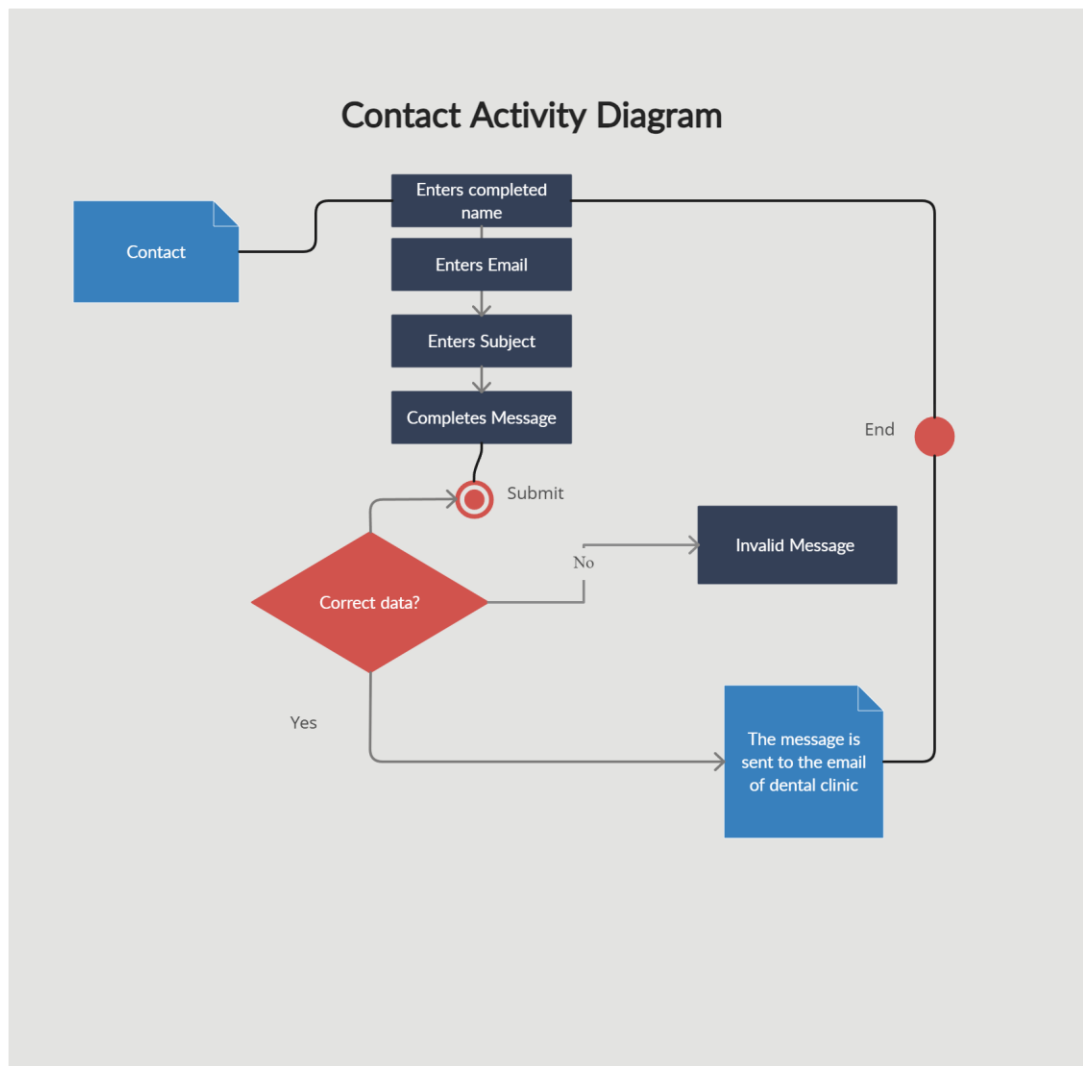


## Diagrama de activitati pentru editarea profilului de utilizator

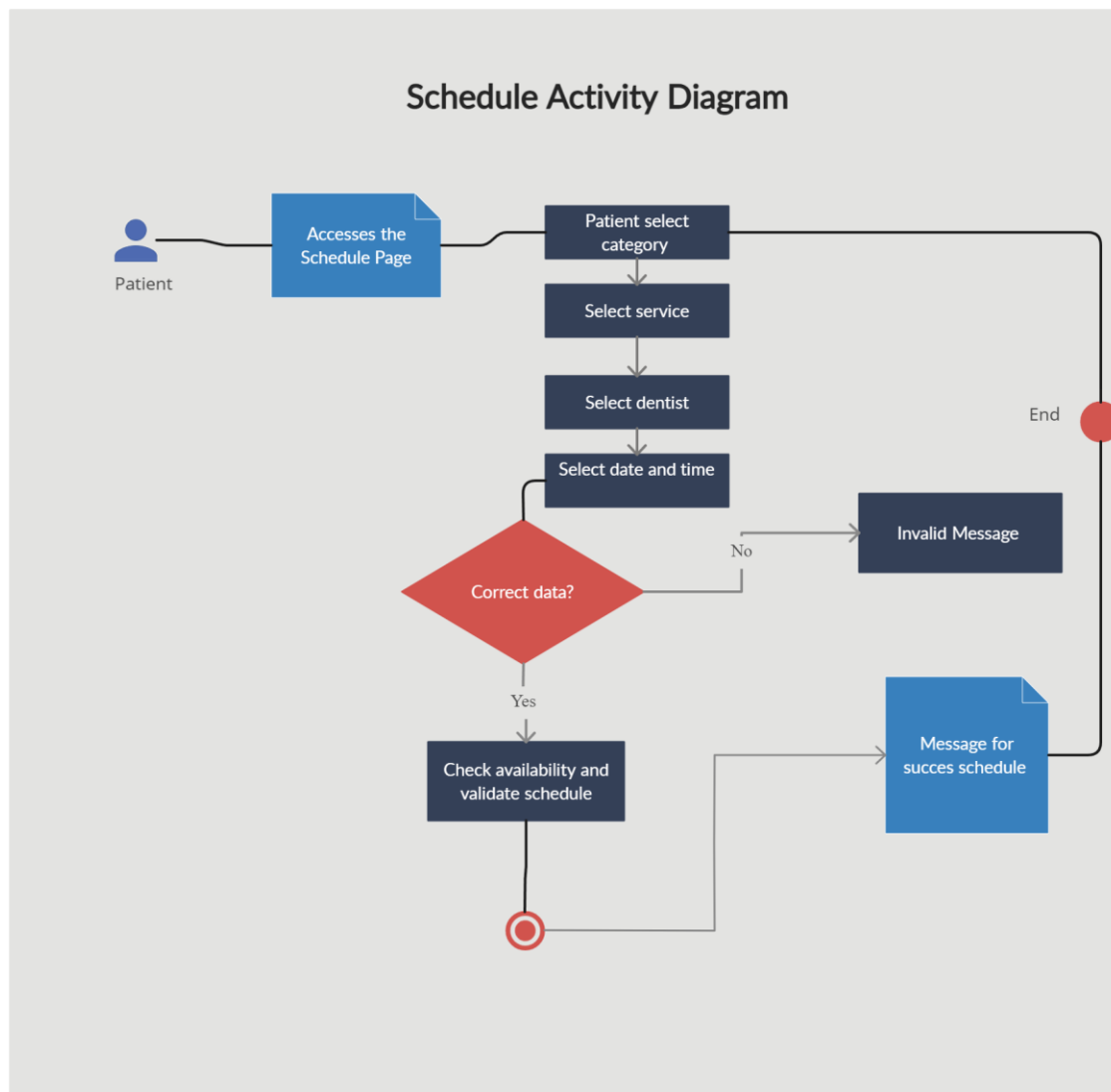




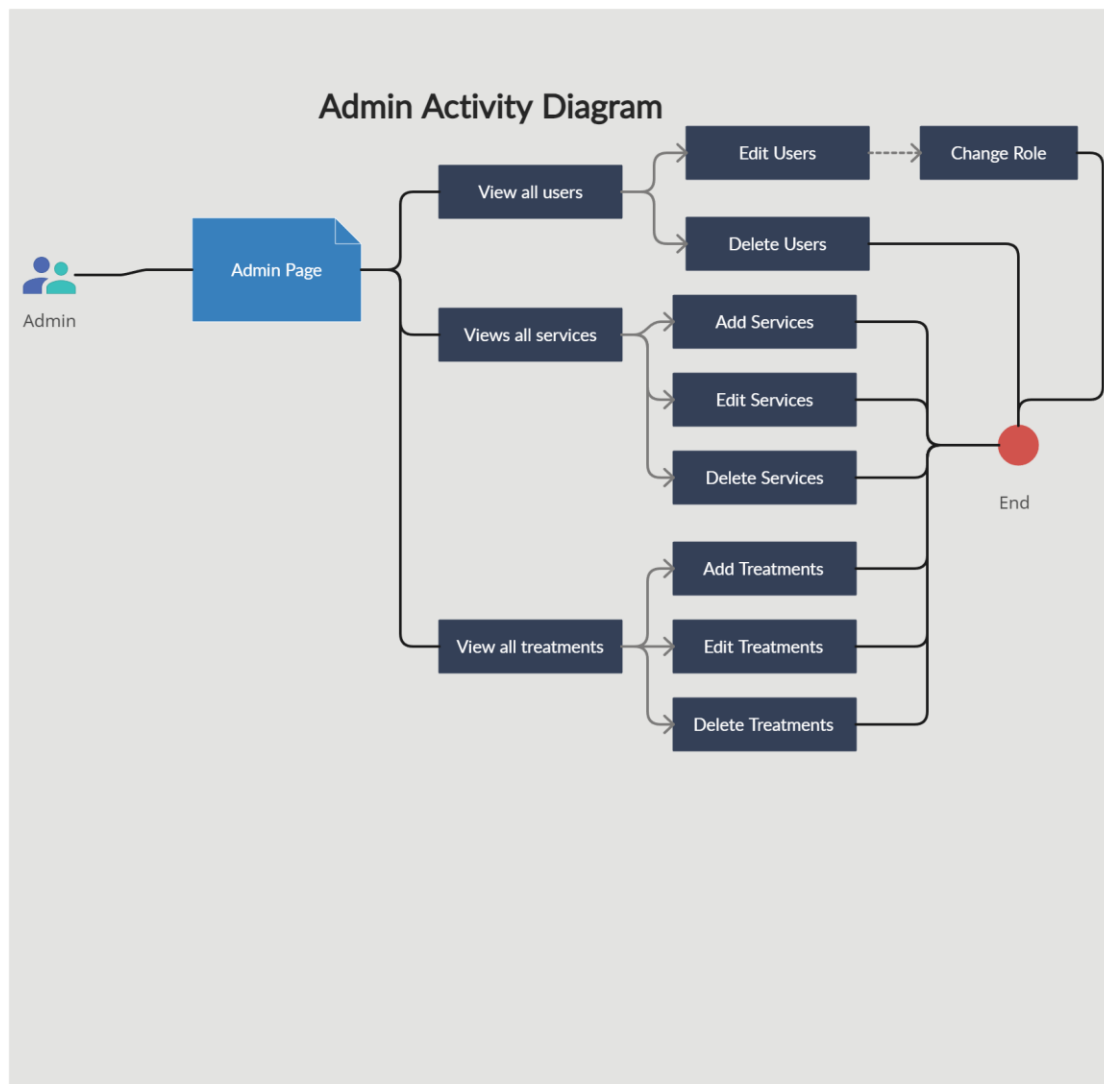
## Diagrama de activitati pentru contact



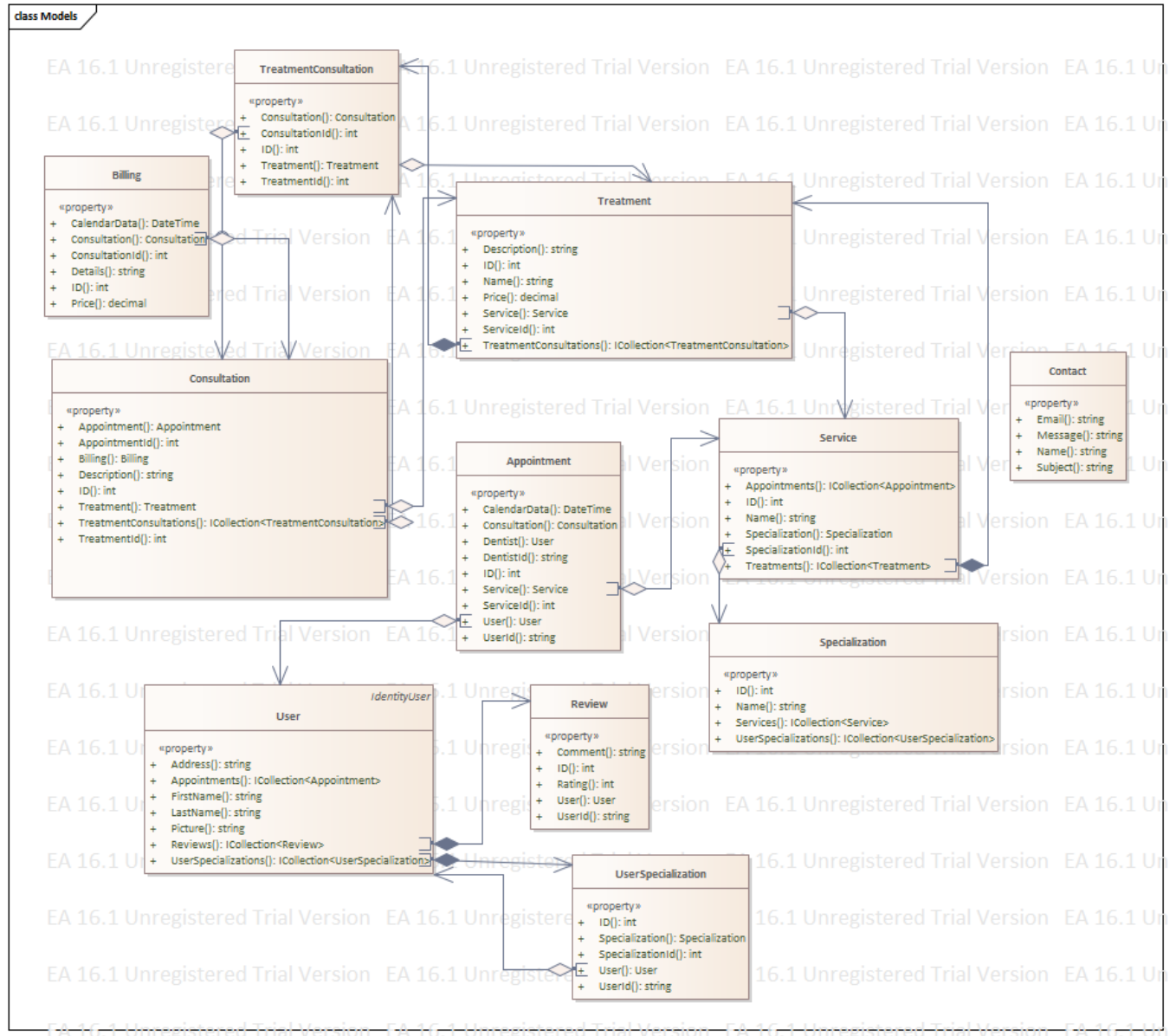
## Diagrama de activitati pentru programare



## Diagrama de activitati pentru pagina de admin

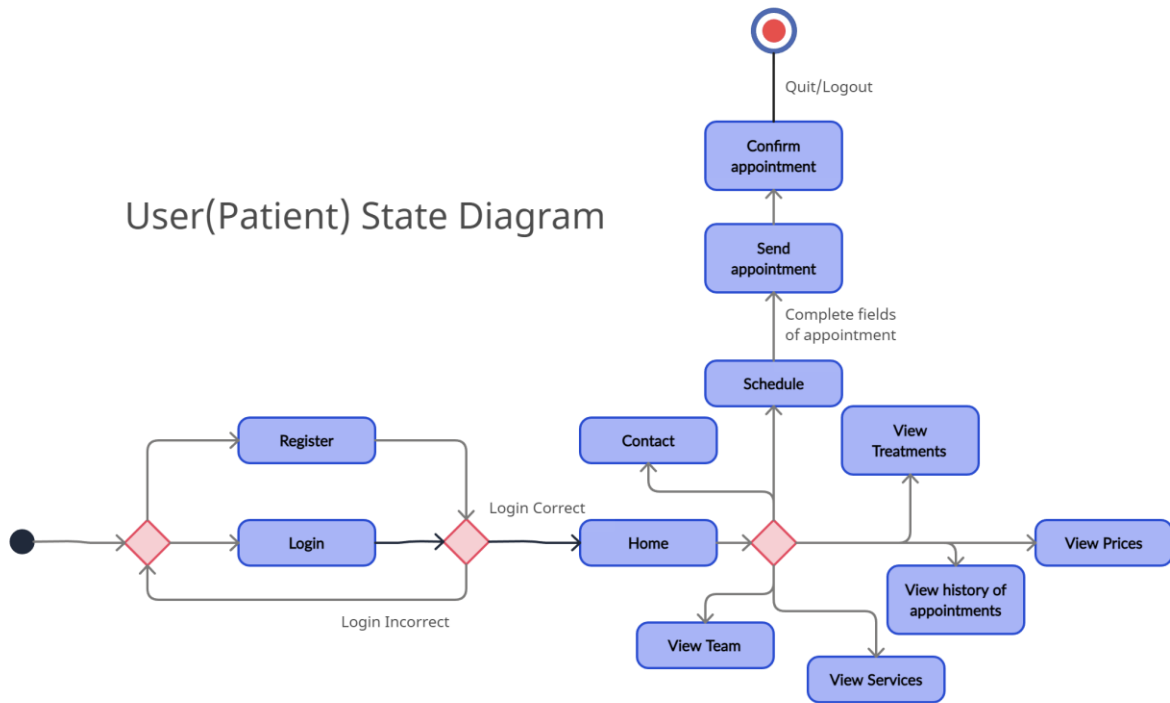


## Diagrama de clase

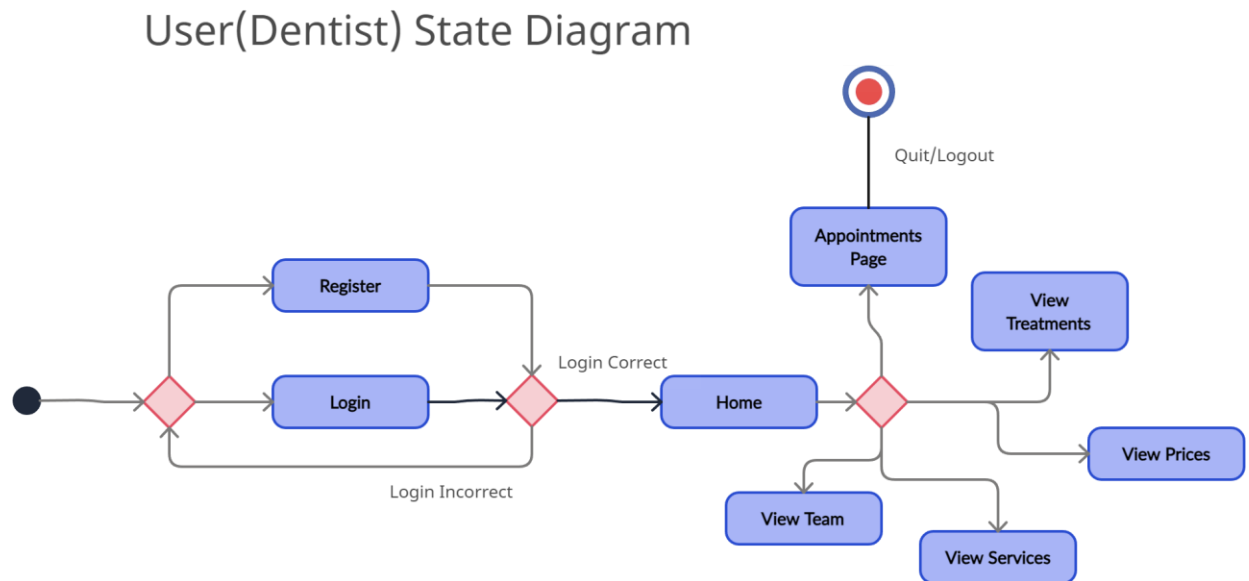


## Diagrama de stare

Pentru utilizatorul cu rol de User (Patient)

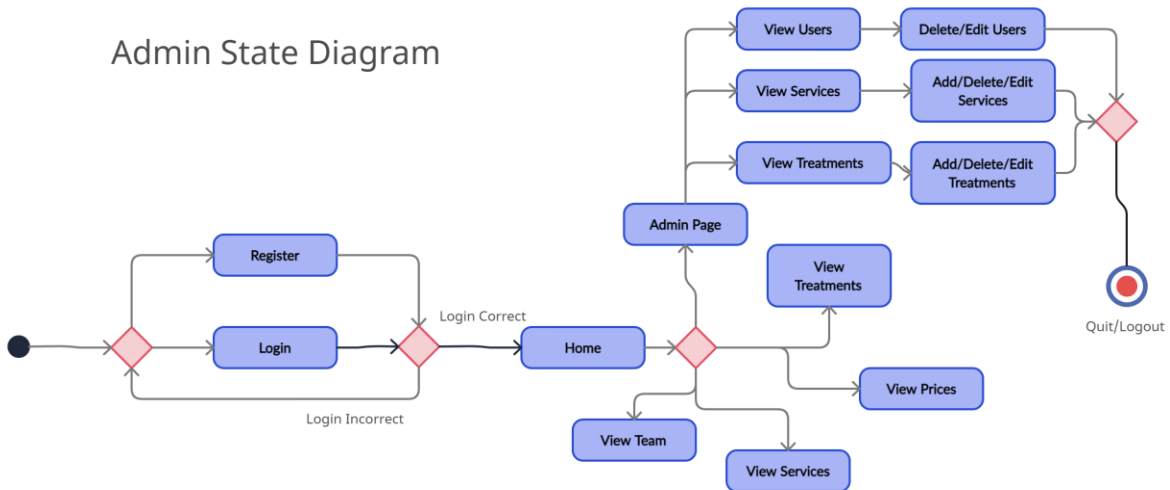


Pentru utilizatorul cu rol de Dentist

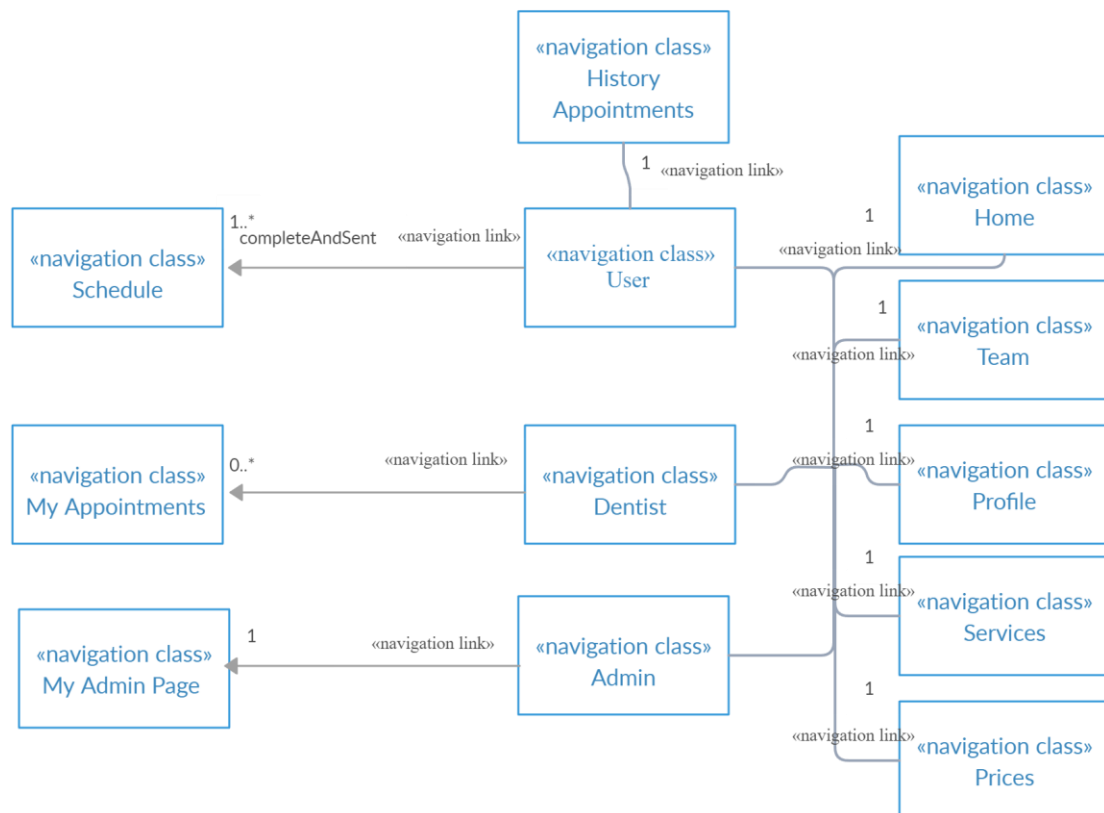


Pentru utilizatorul cu rol de Admin

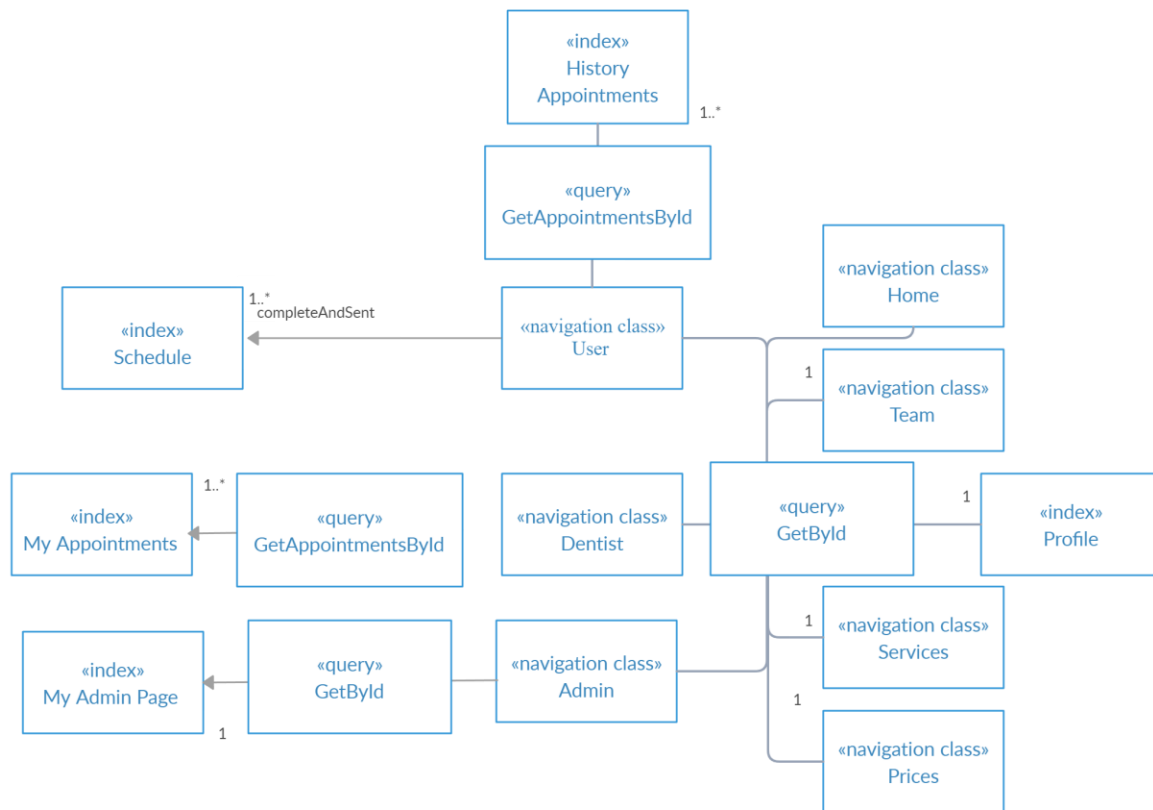
Admin State Diagram



Model de structura hipertext



# Model de acces



# Pagina de prezentare

Logo

HomeEchipaProgrameaza-teServiciiContactTarifeInregistrareAutentificare

Creaza Cont

Email

example.email@gmail.com

Password

Enter at least 8+ characters

Confirm Password

Enter at least 8+ characters

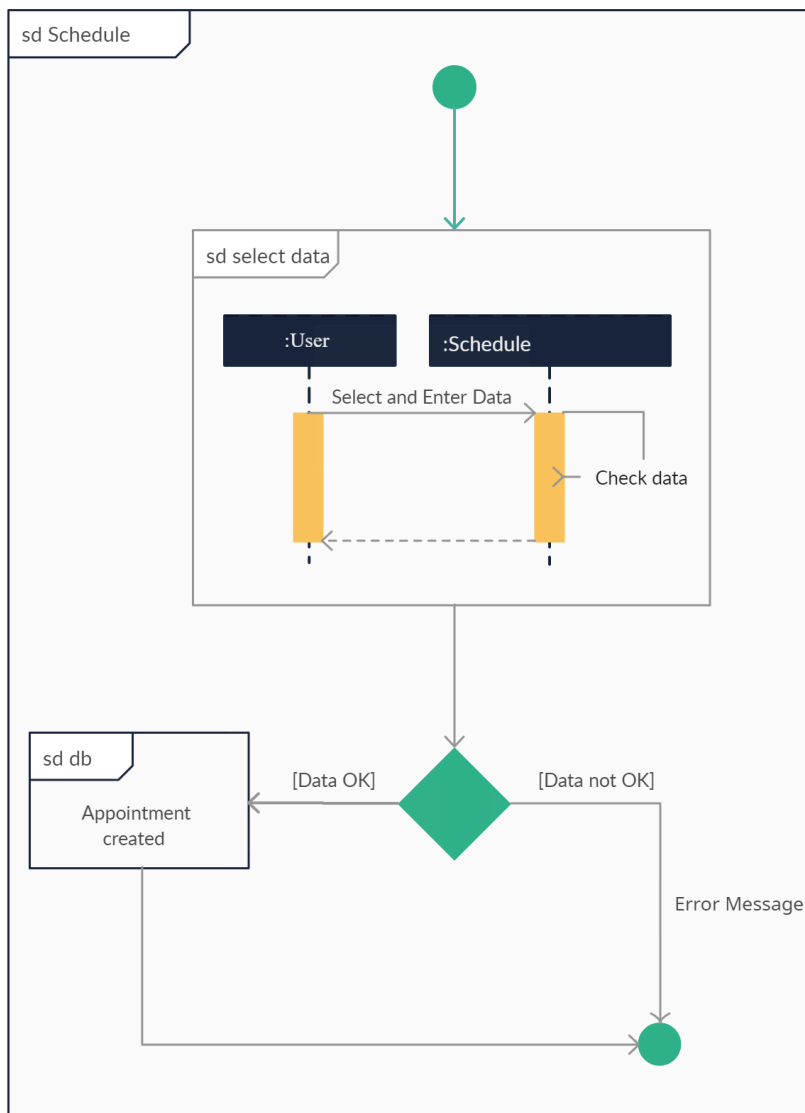
Inregistrare

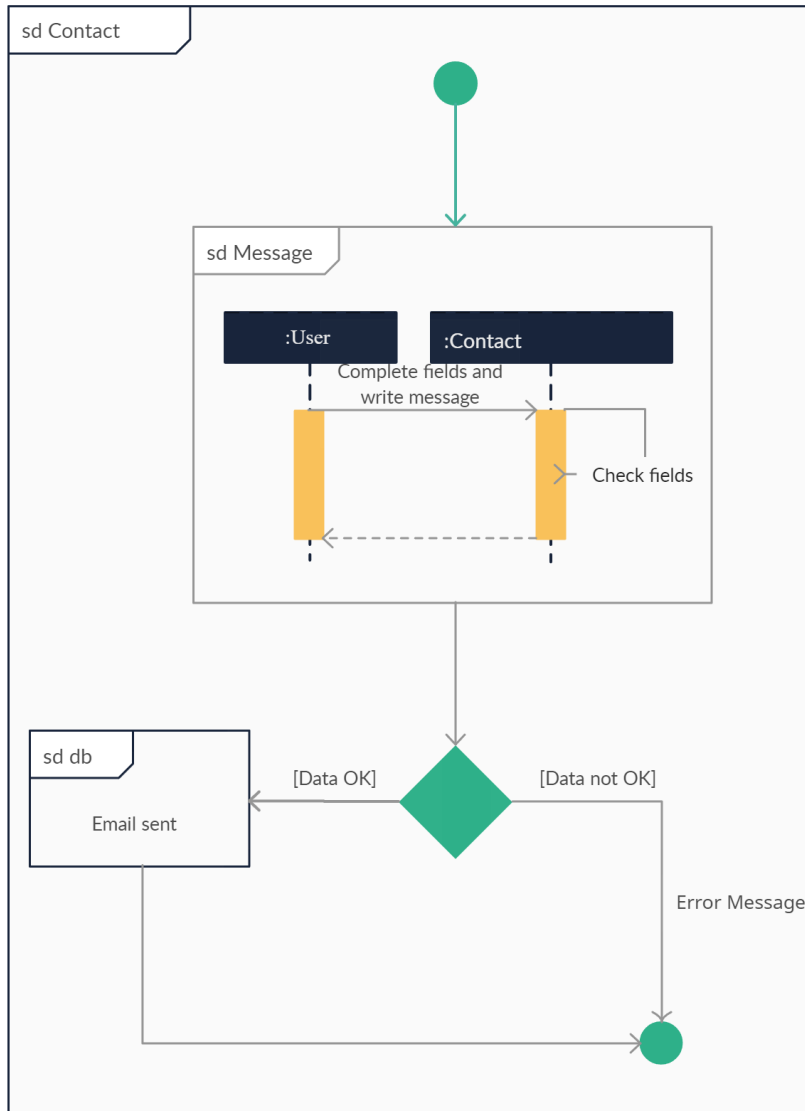
Ai deja un cont?

Autentificare



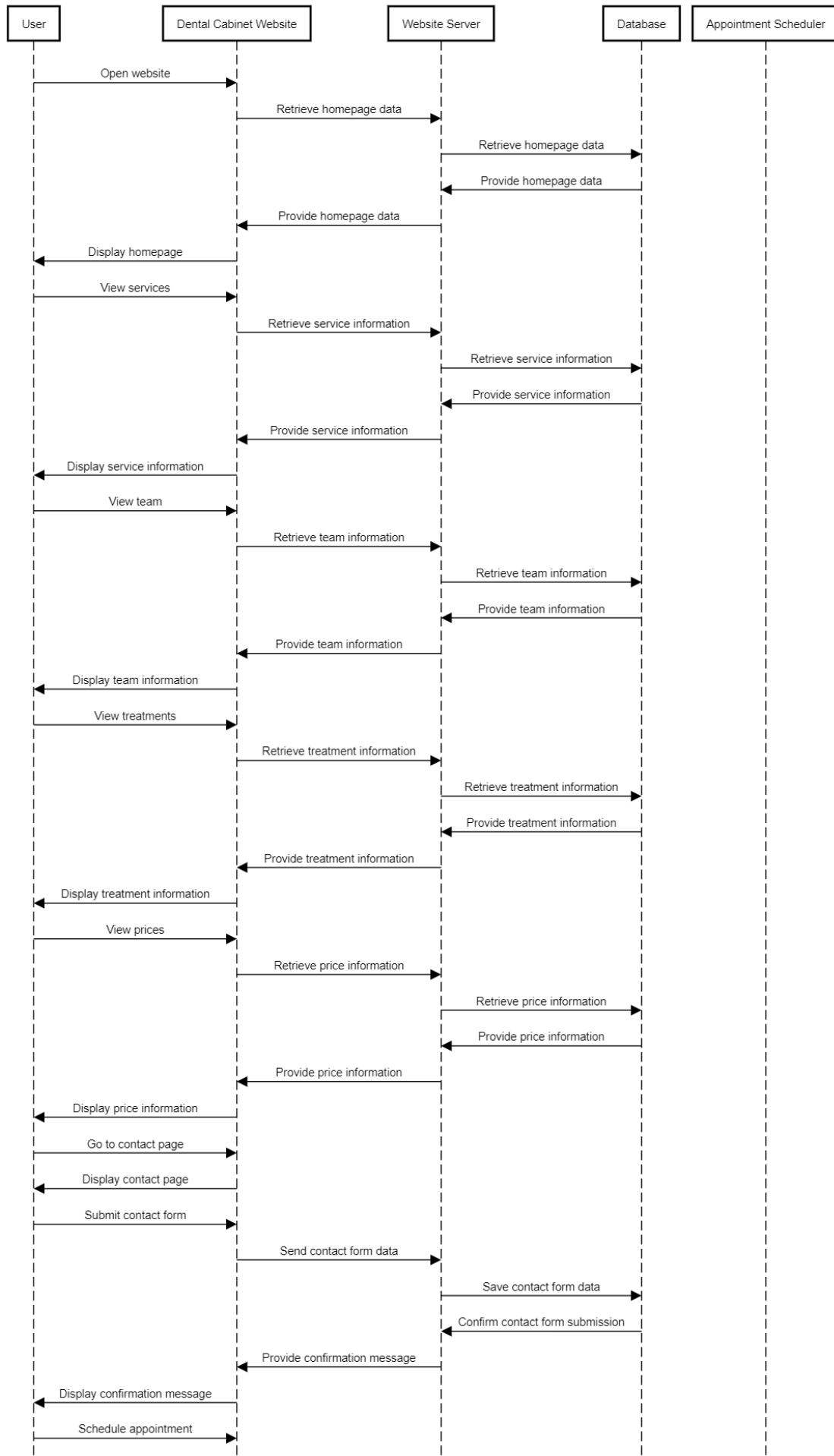
# Diagrama de interactiune



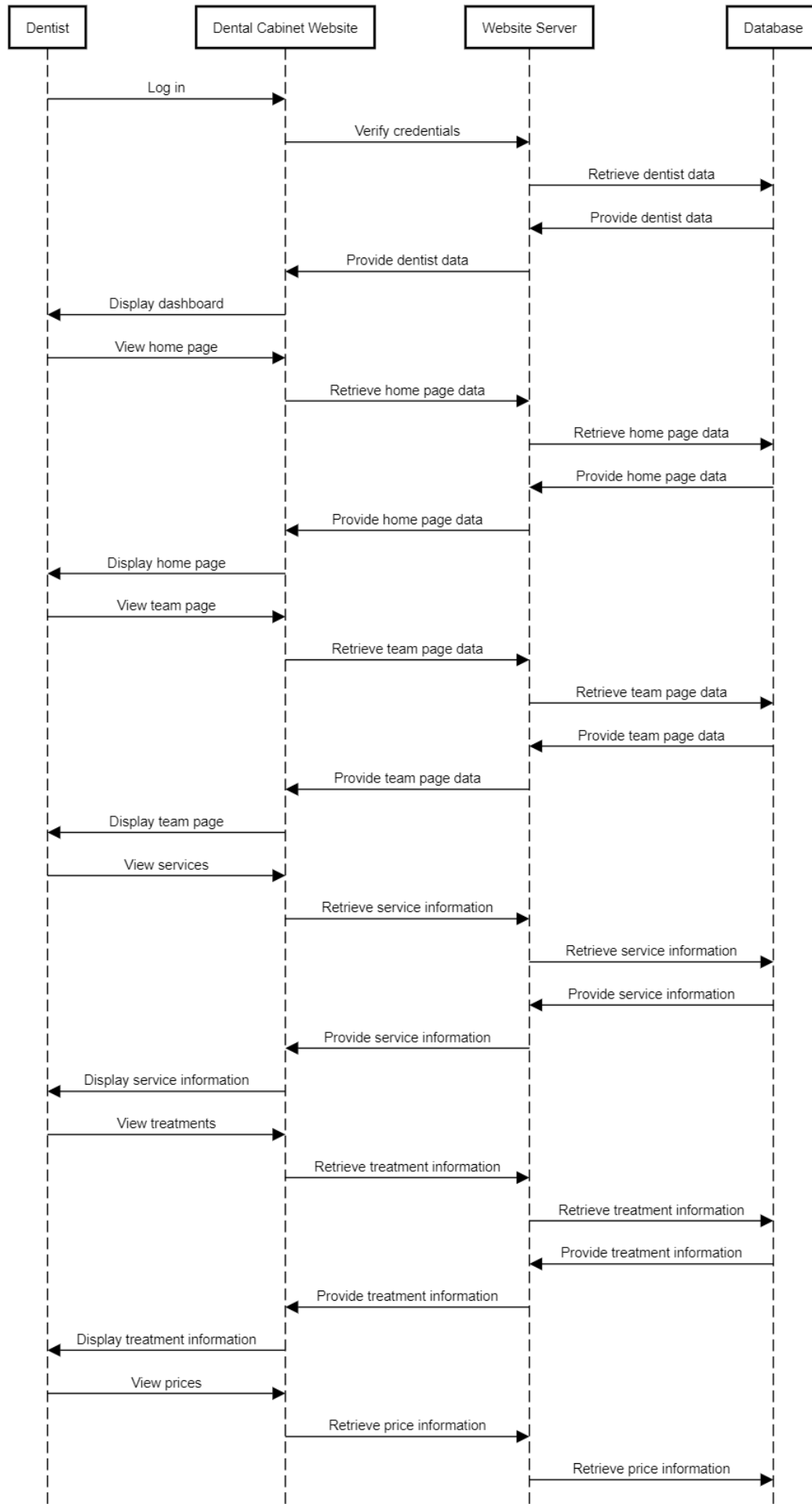


## Diagrama de secventa

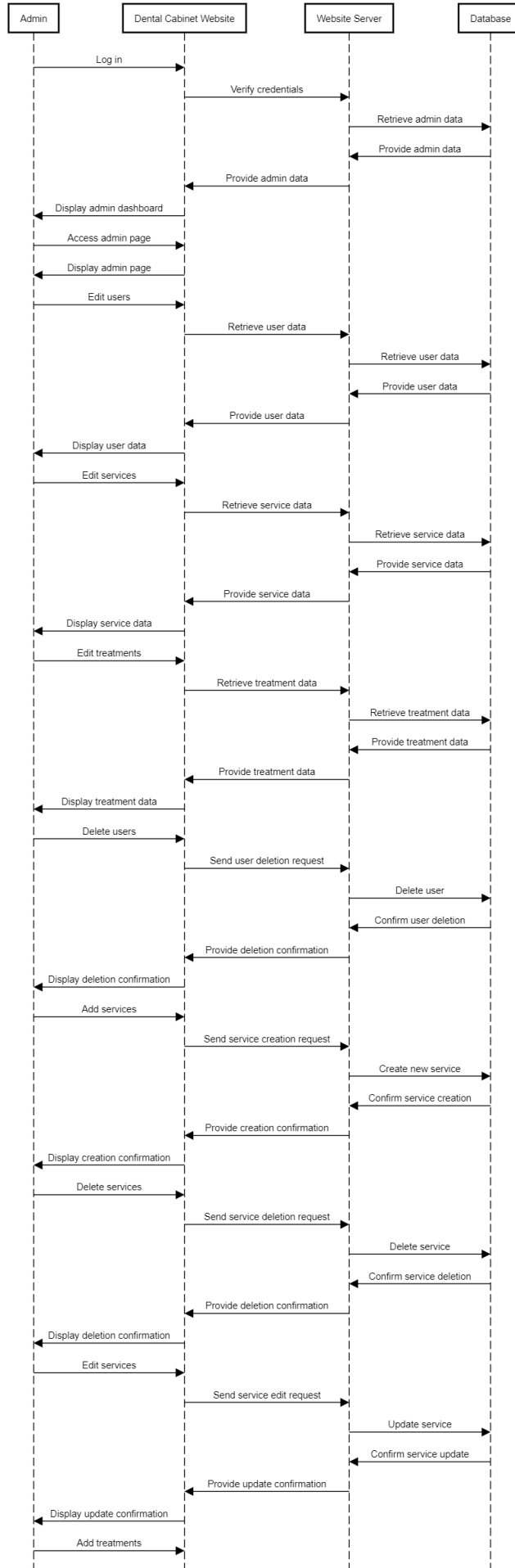
Pentru utilizatorul cu rol de User (Patient)



Pentru utilizatorul cu rol de dentist



Pentru utilizatorul cu rol admin



# 3. Descrieti arhitectura sistemului si structura bazei de date (inclusiv diagrame).

## Arhitectura sistemului

Sistemul Cabinet stomatologic este dezvoltat utilizând tehnologiile C#, ASP.NET și Entity Framework Core.

Arhitectura aplicației se bazează pe modelul arhitectural MVC (Model-View Controller), care permite separarea logică a componentelor sistemului.

Componentele principale ale arhitecturii sunt:

- Model: Reprezintă structura datelor și logica de afaceri a aplicației. În cadrul sistemului Cabinet stomatologic, sunt definite următoarele entități:
  - User: Această entitate reprezintă un utilizator înregistrat în sistem. Are attribute precum FirstName (prenumele), LastName (numele), Address (adresa), Picture (imaginea profilului), PhoneNumber, Email, Password. De asemenea, este asociată cu alte entități prin intermediul relațiilor.
  - UserSpecialization: Această entitate definește specializările utilizatorilor fiind tabelă de legătură many-to-many. Este o entitate de asociere care leagă utilizatorii de specializări specifice. Are attribute precum ID (identificatorul), SpecializationId (identificatorul specializării) și UserId (identificatorul utilizatorului).
  - TreatmentConsultation: Această entitate reprezintă aplicarea unui tratament într-o anumită consultație. Este o entitate de asociere many-to-many între o consultație și un tratament specific. Attributele includ ID (identificatorul), ConsultationId (identificatorul consultației) și TreatmentId (identificatorul tratamentului).



- **Treatment:** Această entitate descrie tratamentele disponibile în cabinetul stomatologic. Atributele includ ID (identificatorul), ServiceId (identificatorul serviciului asociat), Name (numele tratamentului), Price (prețul tratamentului) și Description (descrierea tratamentului).
- **Specialization:** Această entitate reprezintă specializările medicale disponibile în cadrul cabinetului stomatologic. Are atribute precum ID (identificatorul) și Name (numele specializării).
- **Service:** Această entitate reprezintă serviciile oferite în cabinetul stomatologic. Atributele includ ID (identificatorul), SpecializationId (identificatorul specializării asociate), Name (numele serviciului) și relații către alte entități.
- **Review:** Această entitate reprezintă evaluările și comentariile utilizatorilor despre cabinetul stomatologic. Atributele includ ID (identificatorul), UserId (identificatorul utilizatorului care a scris evaluarea), Rating (nota acordată) și Comment (comentariul).
- **Contact:** Reprezintă formularul de contact prin care utilizatorii pot trimite mesaje către cabinetul stomatologic. Atributele includ Name (numele utilizatorului care trimite mesajul), Email (adresa de email), Subject (subiectul mesajului) și Message (mesajul).
- **Consultation:** Această entitate reprezintă o consultație în cadrul cabinetului stomatologic. Are atribute precum ID (identificatorul), AppointmentId (identificatorul programării asociate), TreatmentId (identificatorul tratamentului asociat), Description (descrierea consultației) și relații către alte entități.
- **Billing:** Această entitate reprezintă detaliile facturării pentru o consultație. Atributele includ ID (identificatorul), ConsultationId (identificatorul consultației asociate), Price (prețul consultației), CalendarData (data facturării) și Details (detaliile facturării).
- **Appointment:** Reprezintă o programare în cadrul cabinetului stomatologic. Are atribute precum ID (identificatorul), UserId (identificatorul utilizatorului), DentistId (identificatorul dentistului), ServiceId (identificatorul serviciului asociat), CalendarData (data și ora programării) și relații către alte entități.

- View: Componenta View este responsabilă de afișarea informațiilor utilizatorului și de interacțiunea cu acesta. Sunt create pagini web pentru fiecare rol de utilizator, care permit afișarea detaliilor despre utilizator și permit efectuarea acțiunilor specifice fiecărui rol, de exemplu pagina pentru funcționalitatea de programare pentru utilizatorii care au rol de User(Pacient), pagina de admin pentru admin, pagina cu istoricul de programări atât pentru pacient cât și pentru utilizatorul cu rol Dentist.
- Controller: Controllerul acționează ca intermediar între Model și View. Gestionează fluxul de date și logica aplicației, preluând solicitările utilizatorului de la View și le procesează corespunzător. Permite controlul între interfața site-ului și a paginilor funcționale "Programeaza-te", "Contact" și alte pagini care extrag informații cu baza de date a cabinetului stomatologic.

## Structura bazei de date

Baza de date a sistemului Cabinet stomatologic "Smile Dent" este proiectată pentru a stoca informațiile despre utilizatori, specializări, servicii, tratamente, programări, consultații, review-uri și facturi. Utilizăm Entity Framework Core pentru a gestiona interacțiunea cu baza de date.

Structura bazei de date este definită prin următoarele entități:

- Tabelul "User": Acest tabel stochează informațiile despre utilizatori, cum ar fi FirstName (prenumele), LastName (numele), Address (adresa), Picture (imaginea de profil)

```

99+ references
public class User : IdentityUser
{
    9 references | 1/1 passing
    public string FirstName { get; set; }
    9 references | 1/1 passing
    public string LastName { get; set; }
    3 references
    public string Address { get; set; }
    4 references
    public string Picture { get; set; }
    1 reference
    public virtual ICollection<Review> Reviews { get; set; }
    1 reference
    public virtual ICollection<Appointment> Appointments { get; set; }
    3 references
    public virtual ICollection<UserSpecialization> UserSpecializations { get; set; }
}

```

- Tabelul "UserSpecializations": Acest tabel este o entitate de asociere care leagă utilizatorii de specializări specifice. Are coloane precum ID (identificatorul), SpecializationId (identificatorul specializării) și UserId (identificatorul utilizatorului).

```

15 references
public class UserSpecialization
{
    0 references
    public int ID { get; set; }
    5 references
    public int SpecializationId { get; set; }
    5 references
    public string UserId { get; set; }

    2 references
    public virtual Specialization Specialization { get; set; }
    1 reference
    public virtual User User { get; set; }
}

```

- 
- Tabelul "TreatmentConsultations": Acest tabel este o entitate de asociere între consultații și tratamente specifice. Atributele includ ID (identificatorul), ConsultationId (identificatorul consultației) și TreatmentId (identificatorul tratamentului).

```

public class TreatmentConsultation
{
    1 reference
    public int ID { get; set; }
    5 references
    public int ConsultationId { get; set; }
    5 references
    public int TreatmentId { get; set; }

    1 reference
    public virtual Consultation Consultation { get; set; }
    1 reference
    public virtual Treatment Treatment { get; set; }
}

```

- 
- Tabelul "Treatment": Acest tabel stochează informații despre tratamente, cum ar fi ServiceId (identificatorul serviciului asociat), Name (numele tratamentului), Price (prețul tratamentului) și Description (descrierea tratamentului).

```

public class Treatment
{
    2 references
    public int ID { get; set; }
    2 references
    public int ServiceId { get; set; }
    5 references
    public string Name { get; set; }
    5 references
    public decimal Price { get; set; }
    5 references
    public string Description { get; set; }

    2 references
    public virtual Service Service { get; set; }
    1 reference
    public virtual ICollection<TreatmentConsultation> TreatmentConsultations { get; set; }
}

```

-

- Tabelul "Specialization": Acest tabel conține informații despre specializări, cum ar fi ID (identificatorul) și Name (numele specializării).

```

31 references
public class Specialization
{
    4 references | 1/1 passing
    public int ID { get; set; }
    4 references | 1/1 passing
    public string Name { get; set; }
    3 references
    public virtual ICollection<Service> Services { get; set; }
    1 reference
    public virtual ICollection<UserSpecialization> UserSpecializations { get; set; }
}

```

- Tabelul "Service": Acest tabel stochează informații despre servicii, cum ar fi SpecializationId (identificatorul specializării asociate) și Name (numele serviciului).

```

public class Service
{
    11 references | 6/6 passing
    public int ID { get; set; }
    3 references
    public int SpecializationId { get; set; }
    14 references | 6/6 passing
    public string Name { get; set; }
    2 references
    public virtual Specialization Specialization { get; set; }
    3 references
    public virtual ICollection<Treatment> Treatments { get; set; }
    1 reference
    public virtual ICollection<Appointment> Appointments { get; set; }
}

```

- Tabelul "Review": Acest tabel stochează evaluările și comentariile utilizatorilor despre cabinetul stomatologic, cum ar fi ID (identificatorul), UserId (identificatorul utilizatorului care a scris evaluarea), Rating (nota acordată) și Comment (comentariul).

```

20 references
public class Review
{
    1 reference
    public int ID { get; set; }
    1 reference
    public string UserId { get; set; }
    4 references
    public int Rating { get; set; }
    0 references
    public string Comment { get; set; }
    3 references
    public virtual User User { get; set; }
}

```

- Tabelul "Contact": Acest tabel stochează informațiile despre mesajele de contact trimise de utilizatori, cum ar fi Name (numele utilizatorului

care trimite mesajul), Email (adresa de email), Subject (subiectul mesajului) și Message (mesajul).

```
1 reference
public class Contact
{
    [Required(ErrorMessage = "Vă rugăm să introduceți numele dumneavoastră")]
    1 reference
    public string Name { get; set; }

    [Required(ErrorMessage = "Vă rugăm să introduceți adresa dumneavoastră de email")]
    [EmailAddress(ErrorMessage = "Vă rugăm să introduceți o adresă de email validă")]
    1 reference
    public string Email { get; set; }

    [Required(ErrorMessage = "Vă rugăm să introduceți subiectul mesajului")]
    1 reference
    public string Subject { get; set; }

    [Required(ErrorMessage = "Vă rugăm să introduceți un mesaj")]
    1 reference
    public string Message { get; set; }
}
```

- Tabelul "Consultation": Acest tabel stochează informații despre consultații, cum ar fi AppointmentId (identificatorul programării asociate), TreatmentId (identificatorul tratamentului asociat), Description (descrierea consultației) și relații către alte tabele.

```
public class Consultation
{
    public int ID { get; set; }

    0 references
    public string Description { get; set; }
    1 reference
    public virtual Billing Billing { get; set; }
    1 reference
    public virtual ICollection<TreatmentConsultation> TreatmentConsultations { get; set; }
}
```

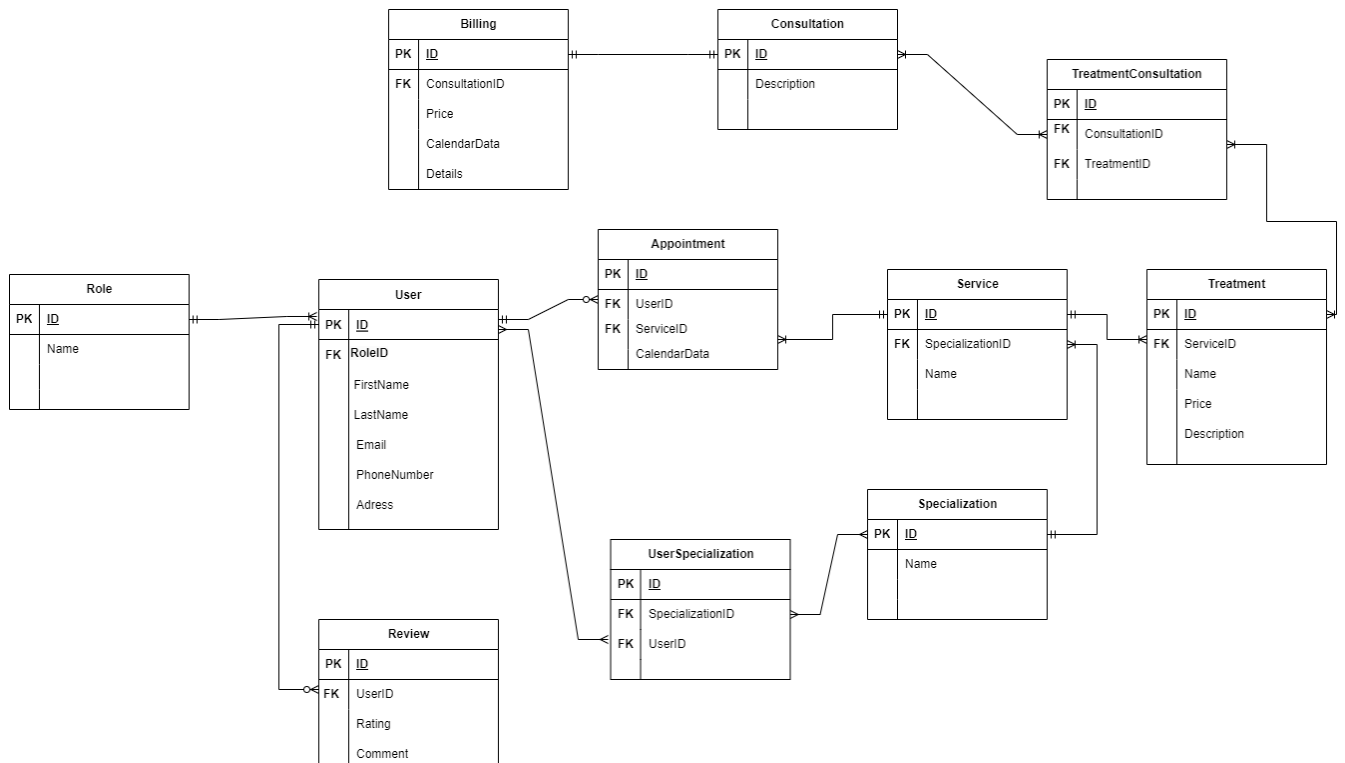
- Tabelul "Billing": Acest tabel stochează informații despre facturări, cum ar fi ConsultationId (identificatorul consultației asociate), Price (prețul consultației), CalendarData (data facturării) și Details (detaliile facturării).

```
1 reference
public class Billing
{
    1 reference
    public int ID { get; set; }
    1 reference
    public int ConsultationId { get; set; }
    0 references
    public decimal Price { get; set; }
    0 references
    public DateTime CalendarData { get; set; }
    0 references
    public string Details { get; set; }
    1 reference
    public virtual Consultation Consultation { get; set; }
}
```

- Tabelul "Appointment": Acest tabel stochează informații despre programări, cum ar fi UserId (identificatorul utilizatorului), DentistId (identificatorul dentistului), ServiceId (identificatorul serviciului asociat), CalendarData (data și ora programării) și relații către alte tabele.

```
public class Appointment
{
    5 references
    public int ID { get; set; }
    7 references
    public string UserId { get; set; }
    6 references
    public string DentistId { get; set; }
    5 references
    public int ServiceId { get; set; }
    10 references
    public DateTime CalendarData { get; set; }
    5 references
    public virtual User User { get; set; }
    5 references
    public virtual User Dentist { get; set; }
    4 references
    public virtual Service Service { get; set; }
}
```

## Modelul ER al bazei de date



## 4. Descrieti functionalitatile pe care le-ati implementat si ilustrati-le cu capturi de ecran (screenshots).

Funcționalitatea pentru Vizitator:

Vizualizarea detaliilor despre site:

- Vizitatorul poate accesa pagina principală a site-ului și poate vedea informații generale despre cabinetul stomatologic care sunt afișate pe pagina de home.



Formular de contact:

- Vizitatorul poate accesa pagina de contact și poate completa un formular de contact pentru a trimite un mesaj către cabinetul stomatologic. Formularul de contact solicita informații precum numele, adresa de email, subiectul mesajului și mesajul în sine. După completarea formularului, vizitatorul poate trimite mesajul către cabinetul stomatologic.

**SmileDent**

Home Echipa Programeaza-te Servicii Contact Tarife Inregistrare Autentificare

## Contact

Pentru a ne contacta, vă rugăm să folosiți informațiile de mai jos:

Strada Calea Bucuresti nr. 1, Craiova  
0777 123 456  
contact@smiledent.ro

### Trimiteți-ne un mesaj

Nume complet

Adresa de email

Subiect

Mesaj

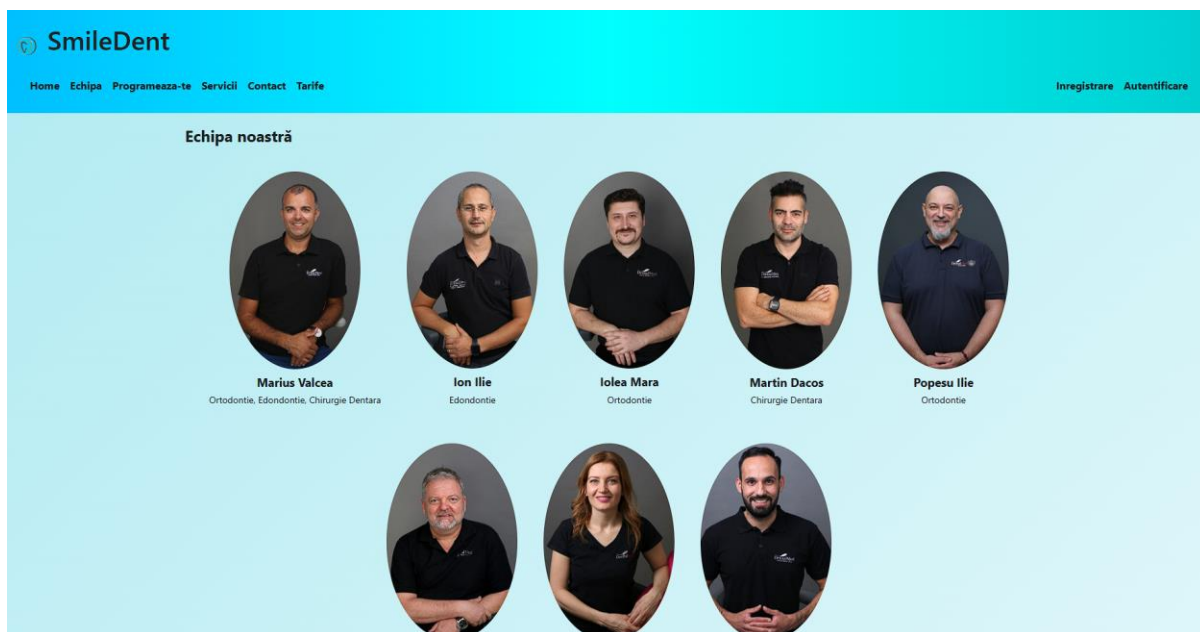
Trimiteți

Navigare în site și acces la informații generale:

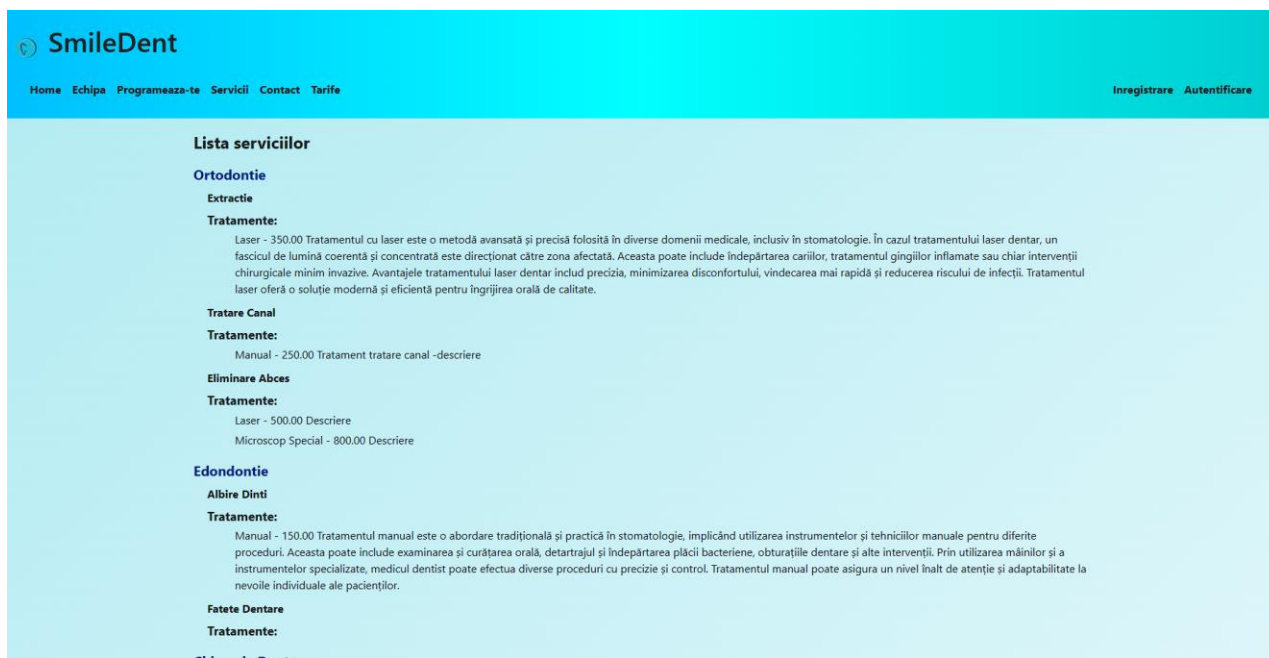
- Vizitatorul poate naviga în diverse secțiuni ale site-ului, cum ar fi pagina principală de home, serviciile, echipa, tratamentele și tarifele, pagina de contact și altele, pentru a obține informații generale despre cabinetul stomatologic.

De exemplu poate să acceseze pagina de "Echipa" unde poate vizualiza toți membrii echipei dentare, inclusiv specializările lor.





Poate să acceseze pagina de "Servicii" unde poate vizualiza toate serviciile oferite, tratamentele cât și o descriere a lor.



Poate să acceseze si pagina de "Tarife" unde va putea vizualiza tarifele pentru serviciile oferite.


SmileDent

[Home](#)
[Echipa](#)
[Programeaza-te](#)
[Servicii](#)
[Contact](#)
[Tarife](#)

[Inregistrare](#)
[Autentificare](#)

### Tarife


Fundamentul activității SmileDent este sănătatea dentară a pacientului pe termen lung. Și pentru că zămbetul reprezintă o investiție pentru toată viața este important ca alegerea făcută să fie cea mai potrivită pentru tine.

### Tratamente disponibile la SmileDent

Numele Tratatamentului	Descriere	Pret
Laser	Tratatamentul cu laser este o metodă avansată și precisă folosită în diverse domenii medicale, inclusiv în stomatologie. În cazul tratamentului laser dentar, un fascicul de lumină coerentă și concentrată este direcționat către zona afectată. Aceasta poate include îndepărtarea cariilor, tratamentul gingivilor inflamate sau chiar intervenții chirurgicale minim invazive. Avantajele tratamentului laser dentar includ precizia, minimizarea disconfortului, vindecarea mai rapidă și reducerea riscului de infecții. Tratatamentul laser oferă o soluție modernă și eficientă pentru îngrijirea orală de calitate.	350.00 RON
Manual	Tratatamentul manual este o abordare tradițională și practică în stomatologie, implicând utilizarea instrumentelor și tehnicilor manuale pentru diferite proceduri. Aceasta poate include examinarea și curățarea orală, detartrajul și îndepărtarea plăcii bacteriene, obturațiile dentare și alte intervenții. Prin utilizarea mâinilor și a instrumentelor specializate, medicul dentist poate efectua diverse proceduri cu precizie și control. Tratatamentul manual poate asigura un nivel înalt de atenție și adaptabilitate la nevoile individuale ale pacienților.	150.00 RON

Tratatamentul cu laser este garantat a fi cel mai eficient și sigur în

Dacă vizitatorul încearcă sa acceseze pagina "Programeaza-te", el va fi redirecționat către pagina de "Inregistrare"


SmileDent

[Home](#)
[Echipa](#)
[Programeaza-te](#)
[Servicii](#)
[Contact](#)
[Tarife](#)

[Inregistrare](#)
[Autentificare](#)

### CREAZA CONT

Email

Parola

Repetă parola

Inregistrare

Ai deja un cont?

Autentificare

## Funcționalitatea pentru utilizatorul cu rol de User(Pacient):

Utilizatorul se loghează pe site prin pagina de "Autentificare":

SmileDent

Home Echipa Programeaza-te Servicii Contact Tarife

Inregistrare Autentificare

### INTRA IN CONT

Email

Parola

☐ Remember me?

[U ai uitat parola?](#)

**Autentificare**

[Vrei sa iti creezi cont?](#)

**Inregistrare**


Autentificare cu

facebook


El se poate loga în site prin contul cu care s-a înregistrat sau prin autentificarea cu un factor extern și anume prin autentificare cu Facebook.

- Vizualizarea detaliilor despre site: Utilizatorul poate accesa toate paginile pe care le poate accesa si un vizitator, și în schimb datorita faptului ca este logat având un cont , acesta își poate vedea profilul contului , schimba email-ul și parola cât si detaliile proprii ale contului său .

Modificare informații proprii cont utilizator:

 SmileDent

[Home](#) [Echipa](#) [Programeaza-te](#) [Servicii](#) [Contact](#) [Tarife](#) [Programarile Mele](#)



## Manage your account

### Change your account settings

Profile

Email

Password

External logins

Two-factor authentication

Personal data

Profile

Username

dentist3@gmail.com

First Name

Robert

Last Name

Nedelcu

Address

Strada Calea Bucuresti Nr 107 B

Phone number


0763984394

Profile picture


https://www.dental-med.ro/wp-content/uploads/2020/02/Pope

Save

Schimbare email cont utilizator:

 SmileDent

[Home](#) [Echipa](#) [Programeaza-te](#) [Servicii](#) [Contact](#) [Tarife](#) [Programarile Mele](#)



## Manage your account

### Change your account settings

Profile

Email

Password

External logins

Two-factor authentication

Personal data

Manage Email

Email

dentist3@gmail.com

✓

New email

dentist3@gmail.com

Change email

Schimbare parola cont utilizator:

SmileDent

Home Echipa Programeaza-te Servicii Contact Tarife Programarile Mele

## Manage your account

### Change your account settings

Profile

Email

**Password**

External logins

Two-factor authentication

Personal data

#### Change password

Current password

New password

Confirm new password

Update password

- Programarea: Utilizatorul poate accesa pagina de programare, unde poate selecta categoria apoi un serviciu și un dentist disponibil și poate selecta o dată și o oră pentru programare. După finalizarea programării, utilizatorul primește o confirmare a programării.

SmileDent

Home Echipa Programeaza-te Servicii Contact Tarife Programarile Mele

Selectează specializarea:

Ortodontie

Selectează serviciul:

Extracție

Selectează medicul:

Marius Valcea

Data:

06/21/2023

Ora:

02:45 PM

Programează-te

Programarea a fost realizată cu succes.

- Vizualizarea istoricului de programări: Utilizatorul poate accesa pagina cu istoricul de programări, unde poate vedea toate programările anterioare făcute de el, inclusiv detalii precum data, serviciul și dentistul selectat.

SmileDent			
<a href="#">Home</a> <a href="#">Echipa</a> <a href="#">Programazea-te</a> <a href="#">Servicii</a> <a href="#">Contact</a> <a href="#">Tarife</a> <a href="#">Programarile Mele</a>			
Programarile Mele			
Serviciu	Data	Ora	Nume Dentist
Extractie	5/14/2023	5/14/2023 7:00:00 AM	Nedelcu Robert
Extractie	5/1/2023	5/1/2023 6:01:00 AM	Nedelcu Robert
Extractie	5/8/2023	5/8/2023 9:32:00 AM	Dragomir Ilie
Tratare Canal	6/13/2023	6/13/2023 6:00:00 PM	Marius Valcea
Extractie	6/22/2023	6/22/2023 11:02:00 PM	Marius Valcea
Aparat Dentar	6/7/2023	6/7/2023 11:05:00 AM	Martin Dacos

## Funcționalitatea pentru utilizatorul cu rol de Dentist:


- Vizualizarea detaliilor despre site: Dentistul poate accesa toate paginile pe care le pot accesa atât vizitatorul cât si userul(pacientul).
- Vizualizarea programărilor: Dentistul poate accesa pagina dedicată programărilor și poate vedea lista programărilor care i-au fost atribuite. Pentru fiecare programare, el poate vedea detaliile precum data, serviciul și pacientul asociat.




Programarile Mele			
Serviciu	Data	Ora	Nume Pacient
Aparat Dentar	6/7/2023	6/7/2023 11:05:00 AM	Andrei Popescu
Aparat Dentar	6/12/2023	6/12/2023 11:23:00 AM	Ionut Ilea

## Funcționalitatea pentru utilizatorul cu rol de Admin:

- Vizualizarea detaliilor despre site: Administratorul poate accesa toate paginile pe care le pot accesa si ceilalati utilizatori,însă nu are pagina "Programarile Mele" el ocupându-se doar cu administrarea cabinetului stomatologic.
- Administrarea utilizatorilor: Administratorul are acces la o pagină specială de administrare, unde poate vizualiza lista utilizatorilor înregistrați, le poate edita informațiile , le poate schimba rolul sau le poate șterge conturile.



[Home](#)
[Echipa](#)
[Programeaza-te](#)
[Servicii](#)
[Contact](#)
[Tarife](#)
[Admin](#)



### Admin

#### Users

Username	Email	Role	Actions
marcus@gmail.com	marcus@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
dentist4@gmail.com	dentist4@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
admin@gmail.com	admin@gmail.com	Admin	<a href="#">Save</a> <a href="#">Delete</a>
nedelcurobert62@gmail.com	nedelcurobert62@gmail.com	User	<a href="#">Save</a> <a href="#">Delete</a>
nedelcurobert652@gmail.com	nedelcurobert652@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
dentist3@gmail.com	dentist3@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
deianu@gmail.com	deianu@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
dentist5@gmail.com	dentist5@gmail.com	Dentist	<a href="#">Save</a> <a href="#">Delete</a>
usernaw@gmail.com	usernaw@gmail.com	Admin	<a href="#">Save</a> <a href="#">Delete</a>
ion@gmail.com	ion@gmail.com	Admin	<a href="#">Save</a> <a href="#">Delete</a>

- Administrarea serviciilor și tratamentelor: Administratorul poate accesa aceeași pagină dedicată administrării userilor, serviciilor și tratamentelor. El poate adăuga, edita sau șterge servicii și tratamente disponibile în cadrul cabinetului stomatologic.

Name	Actions
Extracție	<a href="#">Save</a> <a href="#">Delete</a>
Albire Dinti	<a href="#">Save</a> <a href="#">Delete</a>
Aparat Dentar	<a href="#">Save</a> <a href="#">Delete</a>
Tratare Canal	<a href="#">Save</a> <a href="#">Delete</a>
Eliminare Abces	<a href="#">Save</a> <a href="#">Delete</a>
Fatete Dentare	<a href="#">Save</a> <a href="#">Delete</a>

#### Treatments

Name	Description	Price	Actions
Laser	Tratamentul cu laser este o metodă avansată	350.00	<a href="#">Save</a> <a href="#">Delete</a>
Manual	Tratamentul manual este o abordare tradițională	150.00	<a href="#">Save</a> <a href="#">Delete</a>
Microscop	Tratamentul cu microscop reprezintă o tehnică	50.00	<a href="#">Save</a> <a href="#">Delete</a>
Manual	Tratament tratare canal - descriere	250.00	<a href="#">Save</a> <a href="#">Delete</a>
Laser	Descriere	500.00	<a href="#">Save</a> <a href="#">Delete</a>
Microscop Special	Descriere	800.00	<a href="#">Save</a> <a href="#">Delete</a>



## 5. Ce metode de testare ati folosit pentru aplicatia voastra? Descrieti procesul de testare, precum si o problema identificata si solutia aplicata pentru rezolvarea sa.

### Testarea unitară:

Am creat și rulat teste unitare folosind framework-ul de testare NUnit. Aceste teste au verificat individual funcționalitățile și comportamentul metodelor pentru functionalitatea site-ului pentru a ne asigura că totul este corect și produc rezultatele așteptate.

Am creat teste NUnit pentru funcționalitate de programare în site, fiind cea mai importanta si pentru operațiile pe serviciile oferite în cabinetul stomatologic:

Teste NUnit pentru serviciul de programare

```

public void GetServices_ReturnsServicesFromRepository()
{
    int specializationId = 1;
    var expectedServices = new List<Service>
    {
        new Service { ID = 1, Name = "Service 1" },
        new Service { ID = 2, Name = "Service 2" }
    };
    _appointmentRepositoryMock.Setup(r => r.GetServices(specializationId)).Returns(expectedServices);

    var result = _appointmentService.GetServices(specializationId);

    Assert.AreEqual(expectedServices, result);
}

[Test]
| 0 references
public void GetDentists_ReturnsDentistsFromRepository()
{
    int specializationId = 1;
    var expectedDentists = new List<User>
    {
        new User { Id = "1", FirstName = "John", LastName = "Doe" },
        new User { Id = "2", FirstName = "Jane", LastName = "Smith" }
    };
    _appointmentRepositoryMock.Setup(r => r.GetDentists(specializationId)).Returns(expectedDentists);

    var result = _appointmentService.GetDentists(specializationId);

    Assert.AreEqual(expectedDentists, result);
}

```

```

[Test]
| 0 references
public async Task GetAvailableAppointmentsAsync_ReturnsAvailableAppointmentsFromRepository()
{
    int serviceId = 1;
    string dentistId = "1";
    var expectedAppointments = new List<Appointment>
    {
        new Appointment { ID = 1, CalendarData = new DateTime(2023, 5, 23).Add(new TimeSpan(9, 0, 0)) },
        new Appointment { ID = 2, CalendarData = new DateTime(2023, 5, 24).Add(new TimeSpan(14, 30, 0)) }
    };
    _appointmentRepositoryMock.Setup(r => r.GetAvailableAppointmentsAsync(serviceId, dentistId)).ReturnsAsync(expectedAppointments);

    var result = await _appointmentService.GetAvailableAppointmentsAsync(serviceId, dentistId);

    Assert.AreEqual(expectedAppointments, result);
}

```

```

[Test]
0 references
public async Task CreateAppointmentAsync_CreatesAppointmentInRepository()
{
    var appointmentDTO = new AppointmentDTO
    {
        UserId = "1",
        DentistId = "2",
        ServiceId = 1,
        Date = new DateTime(2023, 5, 25),
        Time = new TimeSpan(10, 0, 0)
    };
    var expectedAppointment = new Appointment
    {
        UserId = appointmentDTO.UserId,
        DentistId = appointmentDTO.DentistId,
        ServiceId = appointmentDTO.ServiceId,
        CalendarData = appointmentDTO.Date.Date.Add(appointmentDTO.Time)
    };

    _appointmentRepositoryMock.Setup(r => r.CreateAppointmentAsync(It.IsAny<Appointment>()))
        .Returns(Task.CompletedTask);

    await _appointmentService.CreateAppointmentAsync(appointmentDTO);

    _appointmentRepositoryMock.Verify(r => r.CreateAppointmentAsync(It.Is<Appointment>(a =>
        a.UserId == expectedAppointment.UserId &&
        a.DentistId == expectedAppointment.DentistId &&
        a.ServiceId == expectedAppointment.ServiceId &&
        a.CalendarData == expectedAppointment.CalendarData)), Times.Once);
}

```

```

[Test]
0 references
public void GetSpecializations_ReturnsSpecializationsFromRepository()
{
    var expectedSpecializations = new List<Specialization>
    {
        new Specialization { ID = 1, Name = "Specialization 1" },
        new Specialization { ID = 2, Name = "Specialization 2" }
    };
    _appointmentRepositoryMock.Setup(r => r.GetSpecializations())
        .Returns(expectedSpecializations);

    var result = _appointmentService.GetSpecializations();

    Assert.AreEqual(expectedSpecializations, result);
}

[Test]
0 references
public void GetAppointmentsById_ReturnsAppointmentsFromRepository()
{
    string userId = "1";
    var expectedAppointments = new List<Appointment>
    {
        new Appointment { ID = 1, CalendarData = new DateTime(2023, 5, 23).Add(new TimeSpan(9, 0, 0)) },
        new Appointment { ID = 2, CalendarData = new DateTime(2023, 5, 24).Add(new TimeSpan(14, 30, 0)) }
    };

    _appointmentRepositoryMock.Setup(r => r.GetAppointmentsById(userId))
        .Returns(expectedAppointments);

    var result = _appointmentService.GetAppointmentsById(userId);

    Assert.AreEqual(expectedAppointments, result);
}

```

## Teste NUnit pentru funcționalitățile serviciilor cabinetului stomatologic

```
[Test]
0 references
public void GetAllServices_ReturnsAllServicesFromRepository()
{
    var expectedServices = new List<Service>
    {
        new Service { ID = 1, Name = "Service 1" },
        new Service { ID = 2, Name = "Service 2" }
    };
    _serviceRepositoryMock.Setup(r => r.GetAllServices()).Returns(expectedServices);

    var result = _serviceService.GetAllServices();

    Assert.AreEqual(expectedServices, result);
}

[Test]
0 references
public void GetServicesBySpecialization_ReturnsServicesBySpecializationFromRepository()
{
    int specializationId = 1;
    var expectedServices = new List<Service>
    {
        new Service { ID = 1, Name = "Service 1" },
        new Service { ID = 2, Name = "Service 2" }
    };
    _serviceRepositoryMock.Setup(r => r.GetServicesBySpecialization(specializationId)).Returns(expectedServices);

    var result = _serviceService.GetServicesBySpecialization(specializationId);

    Assert.AreEqual(expectedServices, result);
}
```

```
[Test]
0 references
public void GetServiceById_ReturnsServiceByIdFromRepository()
{
    int serviceId = 1;
    var expectedService = new Service { ID = serviceId, Name = "Service 1" };
    _serviceRepositoryMock.Setup(r => r.GetServiceById(serviceId)).Returns(expectedService);

    var result = _serviceService.GetServiceById(serviceId);

    Assert.AreEqual(expectedService, result);
}

[Test]
0 references
public async Task AddServiceAsync_AddsServiceToRepository()
{
    var service = new Service { ID = 1, Name = "Service 1" };

    _serviceRepositoryMock.Setup(r => r.AddServiceAsync(It.IsAny<Service>())).Returns(Task.CompletedTask);

    await _serviceService.AddServiceAsync(service);

    _serviceRepositoryMock.Verify(r => r.AddServiceAsync(service), Times.Once);
}

[Test]
0 references
public async Task UpdateServiceAsync_UpdatesServiceInRepository()
{
    var service = new Service { ID = 1, Name = "Updated Service" };

    _serviceRepositoryMock.Setup(r => r.UpdateServiceAsync(It.IsAny<Service>())).Returns(Task.CompletedTask);

    await _serviceService.UpdateServiceAsync(service);

    _serviceRepositoryMock.Verify(r => r.UpdateServiceAsync(service), Times.Once);
}
```

```

[Test]
0 references
public async Task DeleteServiceAsync_DeletesServiceFromRepository()
{
    int serviceId = 1;

    _serviceRepositoryMock.Setup(r => r.DeleteServiceAsync(serviceId)).Returns(Task.CompletedTask);

    await _serviceService.DeleteServiceAsync(serviceId);

    _serviceRepositoryMock.Verify(r => r.DeleteServiceAsync(serviceId), Times.Once);
}

```

Observăm ca toate testele au întors rezultatul dorit.

Test	Duration	Traits	Error Message
NUnitTests (12)	113 ms		
DentalClinicWebsite.Tests.Services (12)	113 ms		
AppointmentServiceTests (6)	107 ms		
CreateAppointmentAsync_Crea...	95 ms		
GetAppointmentsById_Returns...	5 ms		
GetAvailableAppointmentsAsyn...	4 ms		
GetDentists_ReturnsDentistsFro...	2 ms		
GetServices_ReturnsServicesFro...	1 ms		
GetSpecializations_ReturnsSpec...	< 1 ms		
ServiceServiceTests (6)	6 ms		
AddServiceAsync_AddsServiceT...	3 ms		
DeleteServiceAsync_DeletesSer...	1 ms		
GetAllServices_ReturnsAllServ...	< 1 ms		
GetServiceById_ReturnsServiceB...	1 ms		
GetServicesBySpecialization_Ret...	< 1 ms		
UpdateServiceAsync_UpdatesS...	1 ms		

**Group Summary**  
AppointmentServiceTests  
Tests in group: 6  
Total Duration: 107 ms  
**Outcomes**  
6 Passed

## Testarea manulă:

În această abordare, am explorat manual aplicația pentru a verifica interacțiunile dintre componente.

Identificarea funcționalităților cheie: Am început prin a identifica toate funcționalitățile importante ale site-ului, precum înregistrarea utilizatorilor, autentificarea, gestionarea profilului, efectuarea programărilor, vizualizarea informațiilor.

Elaborarea scenariilor de testare: Am creat scenarii de testare detaliate pentru fiecare funcționalitate. Aceste scenarii au inclus pașii specifici pe care utilizatorul trebuie să îi urmeze și rezultatele așteptate.

Testarea interfeței: Am verificat interfețele site-ului pentru a ne asigura că elementele grafice, butoanele, formularele și link-urile funcționează corect și sunt afișate corespunzător pe browser.

Testarea fluxului de navigare: Am parcurs diferite secțiuni ale site-ului și am verificat fluxul de navigare între pagini, asigurându-ne că utilizatorul poate accesa și utiliza toate funcționalitățile într-un mod coerent și intuitiv.

Testarea introducerii datelor: Am efectuat teste pentru a verifica corectitudinea introducerii și validării datelor în formularele site-ului , pagina de înregistrare , autentificare ,contact,programare. Am introdus valori valide și invalide pentru a ne asigura că erorile și mesajele de avertizare sunt afișate corespunzător.

Testarea funcționalităților specifice: Am concentrat testarea pe funcționalitățile cheie ale site-ului, cum ar fi înregistrarea utilizatorilor, autentificarea, gestionarea profilului, efectuarea programărilor ,vizualizarea paginii de programari, încărcarea informațiilor din baza de date pe pagini. Am verificat dacă acestea funcționează conform specificațiilor și au fost rezultatele așteptate.

Testarea erorilor : Am intenționat să introducem erori în diferite etape ale utilizării site-ului pentru a vedea cum acesta gestionează și afișează mesajele de eroare și pentru a ne asigura că utilizatorul poate reveni la starea corectă în cazul unei erori.De exemplu am încercat sa lăsăm spațiile goale atunci când vrem să ne programăm , iar aplicatia va răspunde cu un mesaj de eroare.

Testarea performanței: Am verificat timpul de încărcare al paginilor și viteza de răspuns a site-ului în diferite scenarii și sub sarcini de lucru diferite pentru a ne asigura că site-ul poate gestiona cererile utilizatorilor în mod eficient și fără întârzieri semnificative.

În timpul procesului de testare, am identificat doua probleme, una fiind gestionarea corectă a autorizării utilizatorilor în aplicație.

Am observat că un utilizator fără rolul adecvat putea accesa pagina de admin care este destinata doar adminului.

Pentru a rezolva această problemă, am revizuit logica de autorizare și am implementat verificări suplimentare pentru a asigura că doar utilizatorii cu rolurile corespunzătoare pot accesa funcționalitățile respective, astfel dacă un utilizator cu rol "Dentist" sau "User" se conectează în aplicație, pagina de admin nu va mai fi vizibilă și nici nu va mai putea fi accesata.

Cont de Dentist:

Cont de Admin:

Altă problemă identificată a fost cea în care utilizatorul lăsa câmpurile libere in pagina de "Programaaza-te", acesta nu selecta nimic , iar aplicatia nu îi oferea un mesaj de eroare returnând in schimb o exceptie.

**An unhandled exception occurred while processing the request.**

ArgumentException: Value cannot be null or empty. (Parameter 'viewPath')

Microsoft.AspNetCore.Mvc.ViewEngines.CompositeViewEngine.GetView(string executingFilePath, string viewPath, bool isMainPage)

[Stack](#) [Query](#) [Cookies](#) [Headers](#) [Routing](#)

**ArgumentException: Value cannot be null or empty. (Parameter 'viewPath')**

```
Microsoft.AspNetCore.Mvc.ViewEngines.CompositeViewEngine.GetView(string executingFilePath, string viewPath, bool isMainPage)
Microsoft.AspNetCore.Mvc.ViewFeatures.ViewResultExecutor.FindView(ActionContext actionContext, ViewResult viewResult)
Microsoft.AspNetCore.Mvc.ViewFeatures.ViewResultExecutor.ExecuteAsync(ActionContext context, ViewResult result)
Microsoft.AspNetCore.Mvc.ViewResult.ExecuteResultAsync(ActionContext context)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeNextResultFilterAsync>g__Awaited|30_0<TFilter, TFilterAsync>(ResourceInvoker invoker, Task lastTask, State next, Scope scope, object state, bool isCompleted)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.Rethrow(ResultExecutedContextSealed context)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.ResultNext<TFilter, TFilterAsync>(ref State next, ref Scope scope, ref object state, ref bool isCompleted)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.InvokeResultFilters()
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeNextResourceFilter>g__Awaited|25_0(ResourceInvoker invoker, Task lastTask, State next, Scope scope, object state, bool isCompleted)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.Rethrow(ResourceExecutedContextSealed context)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.Next(ref State next, ref Scope scope, ref object state, ref bool isCompleted)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.InvokeFilterPipelineAsync()
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeAsync>g__Awaited|17_0(ResourceInvoker invoker, Task task, IDisposable scope)
Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeAsync>g__Awaited|17_0(ResourceInvoker invoker, Task task, IDisposable scope)
Microsoft.AspNetCore.Routing.EndpointMiddleware.<Invoke>g__AwaitRequestTask|6_0(Endpoint endpoint, Task requestTask, ILogger logger)
Microsoft.AspNetCore.Authorization.AuthorizationMiddleware.Invoke(HttpContext context)
Microsoft.AspNetCore.Diagnostics.DeveloperExceptionPageMiddlewareImpl.Invoke(HttpContext context)
```

[Show raw exception details](#)

Pentru rezolvarea acestei probleme, am schimbat disponibilitatea butonului "Programaaza-te", astfel încat daca unul sau mai multe câmpuri nu sunt completate sau selectate butonul va fi dezactivat , iar dacă toate câmpurile sunt completate și selectate acesta se va activa.

Buton dezactivat

The screenshot shows the SmileDent website's appointment form. The header is blue with the SmileDent logo and navigation links: Home, Echipa, Programeaza-te, Servicii, Contact, Tarife, Admin. The form has a light blue background and contains the following fields: 'Selectează specializarea:' with 'Ortodontie' selected; 'Selectează serviciul:' with 'Extracție' selected; 'Selectează medicul:' with 'Selectează medicul' as a placeholder; 'Data:' with '06/07/2023' and a calendar icon; 'Ora:' with '07:03 PM' and a clock icon. At the bottom is a blue button labeled 'Programează-te', which is currently disabled.

Buton activat

This screenshot shows the same SmileDent appointment form, but the 'Programează-te' button is now active (highlighted in a darker blue). The form fields are filled with: 'Ortodontie' for specialization, 'Extracție' for service, 'Iolea Mara' for the doctor, '06/16/2023' for the date, and '07:03 PM' for the time.

Am retestat apoi scenariile de autorizare și am verificat că problema a fost rezolvată cu succes.

## 6. Elaborati un plan de promovare al aplicatiei voastre.

Aș începe prin identificarea unui cabinet stomatologic și contactarea acestuia.



## Colaborarea cu un cabinet stomatologic local:

- Prezentarea aplicației și beneficiile pe care le oferă pentru pacienți și medici prin istoricul de programări online și prin funcționalitatea de programare online, în multe site-uri aflate în folosință în ziua de azi nu există posibilitatea de programare online ci doar telefonic sau prin email deci ar fi un avantaj major.
- Propunerea unei colaborări prin intermediul căreia putem promova cabinetul lor stomatologic în aplicația noastră, oferindu-le o prezență mai mare și accesibilitate atât pentru pacienți cât și pentru echipa lor de medici.

## Personalizarea site-ului pentru cabinetele stomatologice partenere:

- Colaborarea strânsă cu fiecare cabinet stomatologic pentru a personaliza și configura platforma în funcție de nevoile și specificațiile lor.
- Asigurarea unei experiențe consistente și adaptate pentru fiecare cabinet stomatologic în parte.

## Publicitate locală:

- Plasarea anunțurilor în publicații locale, cum ar fi ziarele locale sau revistele de sănătate.
- Realizarea de materiale de promovare, cum ar fi flyere sau pliante, și distribuirea acestora la cabinete stomatologice care încă nu au o aplicație în folosință.

## Marketing digital local:

- Utilizarea strategiilor de marketing online, cum ar fi publicitatea pe rețelele de socializare cum ar fi "Facebook", "Google", pentru a ajunge la potențiali cumpărători pentru cabinete stomatologice din zona respectivă.
- Crearea și promovarea de conținut relevant și util despre sănătatea orală și serviciile stomatologice în zona respectivă, pentru a atrage

atenția și a obține vizibilitatea aplicației noastre și a cabinetelor stomatologice partenere.

### Recomandari și testimoniale:

- Solicitarea de testimoniale și feedback de la pacienții care utilizează aplicația noastră și beneficiază de serviciile cabinetelor stomatologice partenere.
- Evidențierea acestor testimoniale pozitive și recomandări în materialele noastre de promovare și pe website-ul aplicației.
- Încurajarea pacienților mulțumiți să recomande aplicația noastră și cabinetele stomatologice partenere către prieteni, familie și pe rețelele sociale.

### Monitorizarea performanțelor și optimizare:

- Monitorizarea performanțelor campaniilor de promovare și a utilizării site-urilor de către pacienți.
- Colectarea feedback-ului și sugestiilor de la cabinetele stomatologice partenere și pacienți pentru a optimiza continuu site-urile și strategiile de promovare.

**7. Descrieti aspectele de utilizabilitate si accesibilitate relevante pentru aplicatia voastra; cum ati putea sa le integrati / implementati?**

# Utilizabilitate

## Interfață intuitivă:

Vom crea de fiecare dată pentru orice cabinet stomatologic o interfață intuitivă și ușor de utilizat, astfel încât pacienții și dentiștii să poată naviga și accesa funcționalitățile aplicației fără dificultate și să își poată vedea lejer un istoric al programărilor. Vom utiliza principii de design intuitiv, cum ar fi plasarea strategică a elementelor de navigare și organizarea clară a conținutului, de exemplu am ales culoarea temei site-ului ca fiind una albastru deschis astfel încât să se adapteze pentru un cabinet stomatologic.

## Funcționalități simplificate:

Am creat cât mai simplu și mai ușor de utilizat pagina de "Programează-te" și paginile de informații ale aplicației pentru a facilita pacienților programarea, vizualizarea informațiilor despre servicii și medici, precum și gestionarea datelor personale.

## Ghidare:

Am introdus indicații pentru a ghida pacienții în utilizarea aplicației. Acest lucru poate include mesaje de eroare clare și ușor de înțeles, sugestii de completare automată a câmpurilor și afișarea progresului în timpul unui proces complex, cum ar fi completarea unui formular de programare sau contact.

De exemplu dacă un utilizator uită să completeze un câmp de text atunci când se programează sau activează o cerere de contact, acestuia îi va apărea un mesaj de atenționare să completeze acel câmp de text.

**SmileDent**

Home Echipa Programeaza-te Servicii Contact Tarife

Inregistrare Autentificare

### Contact

Pentru a ne contacta, vă rugăm să folosiți informațiile de mai jos:

Strada Calea Bucuresti nr. 1, Craiova  
0777 123 456  
contact@smiledent.ro

#### Trimiteți-ne un mesaj

Nume complet

Adresa de email

Subiect

Mesaj

Trimiteți

## Accesibilitate

Design responsiv:

Am dezvoltat aplicația noastră astfel încât să fie compatibilă cu diferite dispozitive și dimensiuni de ecran, asigurându-ne că utilizatorii pot accesa și utiliza aplicația de pe desktop, laptop, tabletă sau smartphone fără dificultate.

Contrast și vizibilitate:

Am utilizat scheme de culori și combinații de contrast care ajută citirea și vizualizarea conținutului pentru toți utilizatorii. Am ales culoarea albastru deschis pentru a nu deranjeze vizibil și în același timp fiind în tema cu aplicația în sine pentru un cabinet stomatologic și un font vizibil și citibil pentru toți utilizatorii.

Am asigurat că dimensiunea fontului și spațiul dintre elementele de text sunt optimizate pentru o citire ușoară.

Navigare simplă:

Am implementat o navigare clară și ușoară în cadrul aplicației, utilizând structuri de meniu și linkuri bine organizate. Am asigurat etichete descriptive pentru butoane și linkuri, astfel încât utilizatorii să poată naviga cu ușurință prin aplicație și să identifice rapid funcționalitățile dorite.

Testare și feedback:

Am efectuat teste periodice de accesibilitate și vom solicita feedback de la utilizatori pentru a identifica și corecta eventualele probleme sau buguri legate de accesibilitate. Vom fi deschiși la sugestii și vom îmbunătăți continuu aplicația pentru a asigura o experiență accesibilă pentru toți utilizatorii.

Integrarea și implementarea acestor aspecte de utilizabilitate și accesibilitate vor fi prioritare în procesul nostru de dezvoltare a aplicației pentru a asigura o experiență optimă și inclusivă atât pentru pacienți, dentisti și cumpărătorii noștri.

## 8. Descrieti aspectele de securitate relevante pentru aplicatia voastra; cum ati putea sa le integrati /implementati?

Aplicația este implementată folosind ASP.NET Entity Framework Core, astfel încât ASP.NET folosește Identity pentru criptarea datelor și autorizarea prin autentificare, iar ASP.NET și Entity Framework Core sunt perfect compatibile cu utilizarea protocolului HTTPS pentru a securiza comunicațiile între utilizatori și server, ceea ce înseamnă că aplicația noastră respectă standardele de securitate a datelor atât pacienților cât și dentiștilor.

Criptarea datelor:

Pentru a asigura confidențialitatea datelor utilizatorilor, am implementat criptarea parolei utilizatorilor, în baza de date. Folosind Identity Entity Framework Core, vom asigura că parolele sunt stocate într-un format criptat, ceea ce reduce riscul de compromitere a acestora în cazul unui atac asupra bazei de date.

Utilizarea Identity User din Entity Framework Core:

Am integrat Identity User, o componentă puternică de autentificare și autorizare din Entity Framework Core MVC. Acesta ne permite să gestionăm autentificarea utilizatorilor, să controlăm accesul la resurse și să aplicăm politici de securitate personalizate.

Am utilizat Identity User pentru a autentifica utilizatorii în funcție de rolurile lor, de exemplu utilizatorii cu rol de Dentist și User nu au acces la pagina de administrare care este vizibilă și accesibilă doar de către admin. Am făcut asta verificând rolul utilizatorului conectat în fișierele cshtml și am pus autorizare pe controlere.

```
@if (User.Identity.IsAuthenticated && User.IsInRole("Admin"))
{
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="" asp-controller="Admin" asp-action="Admin">Admin</a>
    </li>
}
@if (User.Identity.IsAuthenticated && User.IsInRole("Dentist") || User.IsInRole("User"))
{
    <li class="nav-item">
        <a class="nav-link text-dark" asp-area="" asp-controller="MySchedules" asp-action="MySchedules">Programările Mele</a>
    </li>
}

namespace DentalClinicWebsite.Controllers
{
    [Authorize(Roles = "Admin")]
    1 reference
    public class AdminController : Controller
    {
        private readonly IUserService _userService;
        private readonly IServiceService _serviceService;
        private readonly ITreatmentService _treatmentService;

        public readonly AdminDTO _adminDTO;

        0 references
        public AdminController(IUserService userService, IServiceService serviceService, ITreatmentService treatmentService)
        {
            _userService = userService;
            _serviceService = serviceService;
            _treatmentService = treatmentService;
            _adminDTO = new AdminDTO()
            {
                Users = _userService.GetAllUsersAsync(),
                Services = _serviceService.GetAllServices(),
                Treatments = _treatmentService.GetAllTreatments()
            };
        }
    }
}
```

Validarea și complexitatea parolelor:

Am implementat măsuri de securitate pentru a ne asigura că utilizatorii aleg parole puternice și să le reducem vulnerabilitatea la ghicire a parolei. Am impus validări stricte de complexitate a parolilor, acestea fiind lungimea minimă, utilizarea de caractere speciale, litere mari și mici și cifre.

```
builder.Services.Configure<IdentityOptions>(options =>
{
    // Password settings
    options.Password.RequireDigit = true;
    options.Password.RequiredLength = 8;
    options.Password.RequireNonAlphanumeric = false;
    options.Password.RequireUppercase = true;
    options.Password.RequireLowercase = false;
    options.Password.RequiredUniqueChars = 6;

    // Lockout settings
    options.Lockout.DefaultLockoutTimeSpan = TimeSpan.FromMinutes(30);
    options.Lockout.MaxFailedAccessAttempts = 10;
    options.Lockout.AllowedForNewUsers = true;

    // User Settings
    options.User.RequireUniqueEmail = true;
    options.SignIn.RequireConfirmedEmail = false;
});
```

Actualizarea și gestionarea securității:

Se va monitoriza și actualiza în mod regulat componentele software și bibliotecile utilizate în cadrul aplicației noastre pentru cabinet stomatologic pentru a se asigura că sunt la zi cu cele mai recente patch-uri de securitate. Voi urmări recomandările și alertele de securitate emise de furnizorii de software și voi implementa prompt măsurile corective necesare pentru a remedia orice vulnerabilități potențiale.

## 9. Enumerati limitările si punctele slabe ale aplicatiei voastre, precum si

# imbunatatirile / extinderile care s-ar putea realiza ulterior.

## Limitări și puncte slabe

### Scalabilitate limitată:

Aplicația noastră este implementată pentru un singur cabinet stomatologic, iar dacă vrem să vindem aplicația la mai multe cabinete stomatologice trebuie implementată și stilizată pe gusturile cumpărătorilor.

În prezent, nu dispune de capacități de extindere pentru a gestiona mai multe cabinete. Pentru a permite utilizarea aplicației în mai multe cabinete stomatologice, ar trebui să dezvoltăm pe viitor funcționalități suplimentare pentru a permite să avem mai multe cabinete și să izolăm datele între ele.

### Lipsa funcționalităților avansate:

Aplicația noastră nu dispune de funcționalități de generare a rapoartelor pentru consultațiile cu facturile pentru tratamentele efectuate pe o programare care s-a realizat.

### Design și aspect vizual limitat:

Site-ul web este stilizat ușor și nu are efecte sau culori și animații scoase în evidență. Deși interfața utilizator are o structură clară și intuitivă, stilizarea și designul grafic sunt foarte simple și mai puțin atractive în comparație cu alte aplicații deja existente în folosință. Acest lucru poate avea un impact asupra percepției vizuale și atractivității generale a aplicației noastre de către pacienți sau cumpărători.

## Imbunătățiri și extinderi



Am putea începe prin a implementa funcționalitățile avansate descrise la capitolul de mai sus, pentru ușurința cabinetelor stomatologice de a ține o evidență a consultațiilor și facturilor generate pe programari.

Al doilea pas ar fi cel de extindere a aplicației astfel încât să avem o platformă unde fiecare cabinet stomatologic ar putea să își înscrie propria afacere în aplicația noastră și nu am mai fi ca de fiecare dată când vindem aplicația la diferiți cumpărători care dețin un cabinet stomatologic să facem ajustări majore de design și funcționalitate care să se plieze pe gusturile lor.

Un alt pas ar fi acela de a crea o pagină de Blog unde dentiștii ar putea să scrie sau să încarce materiale sau informații despre un anumit serviciu aflat la modă sau despre o reclamă a cabinetului pusă în presă sau la televizor, sau să interacționeze între ei.

Am putea adăuga și interacțiunea dintre pacient și dentist atunci când se realizează o consultație pe baza unei programări astfel încât pacientul după efectuarea tratamentului să aibă posibilitatea să adauge un review cu o notă și un mesaj în cadrul medicului la care s-a programat, ar fi o funcționalitate chiar draguța și interesantă.

În continuare am putea adăuga o pagină de noutăți care să conțină hyperlink-uri către tutoriale de utilizare a aparatelor dentare sau periutelelor de dinți electrice sau altor aparaturi la modă, cât și videoclipuri despre sedințe cu pacienții a unor tratamente inovative cu aparate de ultimă tehnologie.

Iar un ultim pas ar fi cel de stilizare a interfeței cu utilizatorul astfel încât să fie cât mai user-friendly și cât mai atractivă având animații aflate în temă cu tot ce ține de un cabinet stomatologic și de serviciile oferite de acesta, în prezent fiind cât mai simplă și statică.

## 10. Raport propriu de evaluare

Nota pe care considerați că o meritați la laborator:

Chiar daca am luat nota 10 la laborator , mai este ceva de muncă la aplicație pentru a putea fi folosită în realitate , iar nota pe care o consider ca o merit este 10 , deoarece am inclus fiecare cerință de laborator care a fost divulgată, în aplicație.

**Nota pe care considerati ca o meritati la examen (documentul curent):**

Am încercat sa ating cât de mult pot fiecare subpunct în parte, dar mereu orice lucru poate fi îmbunătățit și niciodata nu exista perfecțiune , asa că eu consider ca pentru acest document aş merita nota 9.

**Număr de ore alocat pentru elaborarea acestui document:**

Numărul de ore il aproximez ca fiind aproape de 32 ore.

**Resurse folosite pentru elaborarea acestui document:**

<https://learn.microsoft.com/enus/troubleshoot/developer/webapps/aspnet/development/security-overview>

[https://www.tutorialspoint.com/asp.net/asp.net\\_security.htm](https://www.tutorialspoint.com/asp.net/asp.net_security.htm)

<https://learn.microsoft.com/en-us/dotnet/core/testing/unit-testing-with-nunit>

<https://chat.openai.com/>

Cursurile aflate pe Classroom

**Preferati acest tip e evaluare sau un examen clasic? Care e mai util/relevant? Care necesită mai mult timp de pregătire?**

În ciuda faptului ca un proiect pentru o aplicație web necesită destul timp , posibil mai mult timp decât un examen clasic propriu-zis , consider ca implementarea unui astfel de proiect este mult mai util si mai relevant decat acel examen propriu zis , deoarece la examen de obicei se învata pentru a promova, în schimb practica este mama învățaturii și pot spune ca dupa realizarea acestei aplicații chiar am învățat destule lucruri frumoase.

## Impresii generale despre laborator & curs si sugestii de îmbunătățire:

Pentru mine cel puțin a fost totul în regula cum s-au desfășurat, totul a decurs destul de bine ,iar prezentările de laborator chiar daca a fost mult de muncă si de informat și învățat au fost adevărate provocari care s-au terminat cu un feedback pozitiv.

Ca și sugestii ce as putea include ar fi modul de lucru și diferite tool-uri pentru a stiliza cat mai avansat un site, cel puțin pentru mine a fost destul de greu să îl fac sa arate cât mai frumos.