

Proiect BD: Aplicatie web pentru gestionarea unui cabinet stomatologic

Descrierea temei

Proiectul presupune dezvoltarea unei aplicatii web si a unei baze de date pentru gestionarea unui cabinet stomatologic.

Folosind interfata web clientii se pot loga si isi pot face o programare, selectand un interval disponibil si operatia dorita.

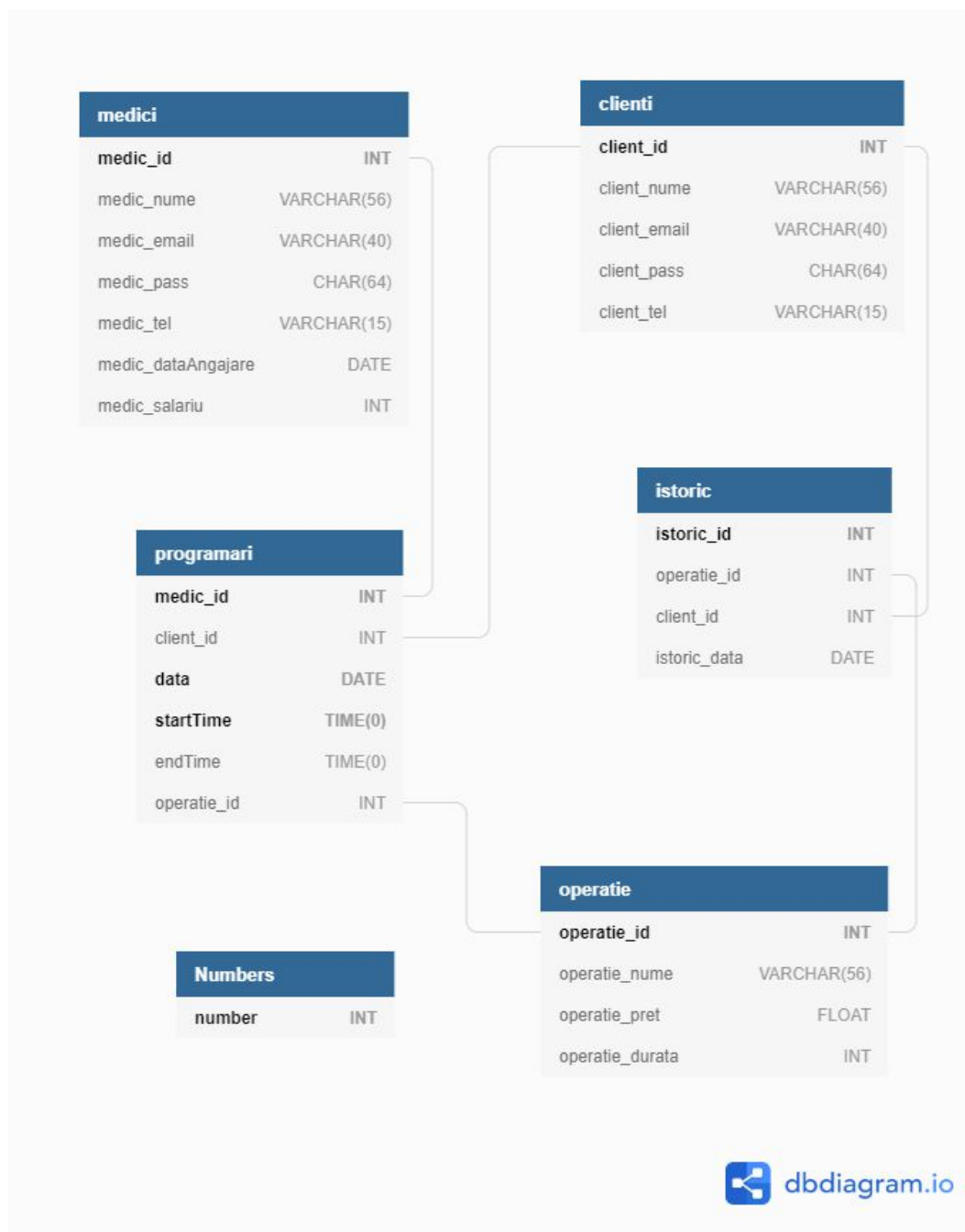
Medicii pot vizualiza programarile lor (un medic nu poate vedea programarile clientilor la alt medic din unitate) si pot cauta istoricul medical al clientilor.

Aplicatia a fost dezvoltata folosind limbajul **Python**. Interfata grafica este realizata cu ajutorul **HTML**, **CSS**, **javascript**/**jQuery**. Partea de backend a aplicatiei utilizeaza framework-ul **Flask**.

Baza de date este implementat folosind **MySQL**. Pentru a porni serverul web si serverul bazei de date este folosit **Docker**. Docker este folosit de asemenea pentru descarcarea/managementul tuturor dependintelor. Arhitectura abordata este bazata pe microservicii ce pot fi lansate/gestionate in mod independent cu ajutorul Docker.

- Descrierea bazei de date:

- Diagrama bazei de date



- Structura tabelor

Sunt create tabelele: **medici**, **clienti**, **programari**, **istoric**, **operatie**.

Tabela **operatie** contine descrierea operatiilor ce pot fi efectuate la cabinetul stomatologic.

Tabela **istoric** cuprinde istoricul medical al unui client.

■ Descrierea constrângerilor de integritate

Tabelele **medici**, **clienti**, **istoric**, **operatie** au ca si cheie primara un id de tipul INT ce se autoincrementeaza.

```
create table istoric(  
    istoric_id      INT NOT NULL AUTO_INCREMENT,  
    ...
```

Tabela **programari** are o *cheie primara compusa* formata din campurile:

```
CONSTRAINT PRIMARY KEY (medic_id, data, startTime),
```

Constrangeri suplimentare pentru tabela programari:

1. ora de start este divizibila cu 10 minute (09:00, 09:10, etc.)
2. ora de incheiere este divizibila cu 10 minute (09:30, 09:40, etc.)
3. programarile nu se pot face inainte de ora 09:00
4. programarile nu se pot termina dupa ora 17:00
5. ora de incheiere nu poate fi mai devreme decat ora de start

```
CONSTRAINT mustStartOnTenMinuteBoundary CHECK (  
    EXTRACT(MINUTE FROM startTime) % 10 = 0  
    AND EXTRACT(SECOND FROM startTime) = 0  
),  
CONSTRAINT mustEndOnTenMinuteBoundary CHECK (  
    EXTRACT(MINUTE FROM endTime) % 10 = 0  
    AND EXTRACT(SECOND FROM endTime) = 0  
),  
CONSTRAINT cannotStartBefore0900 CHECK (  
    EXTRACT(HOUR FROM startTime) >= 9  
),  
CONSTRAINT cannotEndAfter1700 CHECK (  
    EXTRACT(HOUR FROM (startTime - INTERVAL 1 SECOND)) < 17  
),  
CONSTRAINT mustEndAfterStart CHECK (  
    endTime > startTime  
)
```

■ Descrierea procedurilor și funcțiilor

Am creat o funcție ce determină dacă o programare se suprapune oricărei programări din tabela **programari**.

```
CREATE FUNCTION slotIsAvailable(  
    medic_id          INT,  
    slotStartDateTime DATETIME,  
    slotEndDateTime   DATETIME  
) RETURNS BOOLEAN NOT DETERMINISTIC  
BEGIN  
    RETURN CASE WHEN EXISTS (  
        -- This table will contain records iff the slot clashes with an existing appointment  
        SELECT TRUE  
        FROM programari AS p  
        WHERE  
            CONVERT(slotStartDateTime, TIME) < p.endTime -- These two conditions  
            AND CONVERT(slotEndDateTime, TIME) > p.startTime -- with the existing appointment  
            AND p.medic_id = medic_id  
            AND p.data = CONVERT(slotStartDateTime, DATE)  
        ) THEN FALSE ELSE TRUE  
    END;  
END; //
```

Pentru fiecare medic operația SELECT returnează TRUE dacă există programări în tabela ce se suprapun perioadei de timp [*slotStartDateTime:slotEndDateTime*].

Am creat un **trigger** ce verifică înainte de inserarea în tabela **programari**, cu ajutorul funcției **slotIsAvailable**, dacă o nouă programare se suprapune cu o programare existentă în tabela.

```
CREATE TRIGGER ensureNewAppointmentsDoNotClash  
    BEFORE INSERT ON programari  
    FOR EACH ROW  
BEGIN  
    IF NOT slotIsAvailable(  
        NEW.medic_id,  
        CAST( CONCAT(NEW.data, ' ', NEW.startTime) AS DATETIME ),  
        CAST( CONCAT(NEW.data, ' ', NEW.endTime) AS DATETIME )  
    ) THEN  
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Appointment clashes with an existing appointment';  
    END IF;  
END;  
END; //
```

Am creat un event ce se execută zilnic. Acesta verifică dacă o programare din tabela **programari** este înregistrată cu o dată din trecut. Dacă astfel de programări există, este inserată o nouă înregistrare în tabela **istoric** pentru clientul corespunzător, iar apoi este eliminată programarea din tabela **programari**.

```
CREATE EVENT removeOldAppointmentsEvent
ON SCHEDULE
  EVERY 1 DAY
ON COMPLETION PRESERVE
DO
  BEGIN
    INSERT INTO istoric (operatie_id, client_id, medic_id, data)
    SELECT operatie_id, client_id, medic_id, data
    FROM programari AS p
    WHERE p.data <= DATE(NOW());
    DELETE FROM programari p WHERE p.data <= DATE(NOW());
  END //
```

Astfel, tabela **programari** contine doar programari ce vor urma, programarile din trecut fiind *mutate* automat in tabela **istoric**.

Pentru a testa acest event am creat o procedura ce face aceleasi operatii si care poate fi apelata manual.

```
CREATE PROCEDURE testRemoveEvent()
BEGIN
  INSERT INTO istoric (operatie_id, client_id, istoric_data)
  SELECT operatie_id, client_id, data
  FROM programari AS p
  WHERE p.data <= DATE(NOW());
  DELETE FROM programari p WHERE p.data <= DATE(NOW());
END; //
```

Pentru a respecta politica de prelucrare a informatiilor, un client poate opta ca aplicatia sa stearga toate datele sale personale. Astfel, pentru constrangerea FOREIGN KEY din tabela **istoric** ce referentiaza intrarile tablei **clienti** este setata optiunea **ON DELETE CASCADE**.

Variabila **log_bin_trust_function_creators** este setata pentru a evita erorile din functia `slotIsAvailable`. Aceste erori apar deoarece aceasta functie este nedeterminista. Informatii suplimentare pentru functii

non-deterministe: <https://dev.mysql.com/doc/refman/8.0/en/stored-programs-logging.html>

```
-- this setting avoids ERROR 1418 (HY000) at slotIsAvailable function
SET GLOBAL log_bin_trust_function_creators = 1;
```


Variabila **event_scheduler** porneste thread-ul din daemonul mySql ce monitorizeaza evenimentele create. A fost setata pentru a asigura functionarea corecta a event-ului removeOldAppointmentsEvent descris mai sus.

```
-- START EVENT THREAD  
SET GLOBAL event_scheduler = ON;
```

Procesele MySql pot fi vizualizate cu *show processlist*;

```
mysql> show processlist;
```

| Id | User | Host | db | Command | Time | State | Info |
|-----|-----------------|------------------|------|---------|--------|------------------------|------------------|
| 4 | event_scheduler | localhost | NULL | Daemon | 102716 | Waiting on empty queue | NULL |
| 86 | root | 172.17.0.1:39664 | dent | Query | 0 | starting | show processlist |
| 123 | root | 172.17.0.1:39770 | dent | Sleep | 6219 | | NULL |
| 124 | root | 172.17.0.1:39772 | dent | Sleep | 6205 | | NULL |

4 rows in set (0.00 sec)

Clientii primesc o reucre de 20% pentru operatiile ce au un pret mai mare de 200 daca in ultimele 6 luni au avut macar 3 programari pentru operatii cu un cost mai mare de 200.

```
DELIMITER //
```

```
CREATE PROCEDURE getAllDiscountOperations(  
    client_email VARCHAR(40))  
BEGIN  
    SELECT o.operatie_id, o.operatie_nume, o.operatie_pret,  
        FORMAT(0.8 * o.operatie_pret, 2) AS operatie_discount,  
        o.operatie_durata  
    FROM operatie AS o,  
        clienti AS c,  
        istoric AS i  
    WHERE (c.client_email = client_email AND  
        i.client_id = c.client_id AND  
        o.operatie_pret > 200 AND  
        (SELECT COUNT(*) FROM istoric AS j  
            WHERE j.client_id = c.client_id AND  
                j.istoric_data > DATE_SUB(now(), INTERVAL 6 MONTH)) > 3)  
    GROUP BY o.operatie_id;  
END; //
```

```
DELIMITER ;
```

Medicii primesc, pe langa salariu, un bonus egal cu 10% din salariu plus 20 unitati monetare pentru fiecare programare din ultima luna, pentru care pretul programarii este mai mare de 200.

```
DELIMITER //
CREATE PROCEDURE getMedicBonus(medic_id INT)
BEGIN

SET @nr_programari = (SELECT COUNT(*)
                        FROM istoric AS i, operatie AS o
                        WHERE i.istoric_data > DATE_SUB(now(), INTERVAL 1 MONTH) AND
                              o.operatie_pret > 200 AND
                              i.operatie_id = o.operatie_id AND
                              i.medic_id = medic_id);

SELECT m.medic_nume, m.medic_salariu,
       ((0.1 * m.medic_salariu ) + (@nr_programari * 20)) AS medic_bonus
FROM medici AS m
WHERE m.medic_id = medic_id;

END; //
DELIMITER ;
```

■ Descrierea aplicației:

Arhitectura aplicatiei se bazeaza pe microservicii gestionate cu ajutorul **Docker**. Sursele se regasesc in doua directoare principale: **py** respectiv **bd**. Fisierul *docker-compose.yml* descrie cum vor fi lansate serviciile (porturi utilizate, versiuni biblioteci, spatiu stocare, etc.). Additional, directoarele py si bd contin fisiere Dockerfile responsabile cu managementul resurselor pentru fiecare serviciu.

```

1  version: '2'
2  services:
3    py:
4      build: ./py
5      links:
6        - db
7      ports:
8        - "5000:5000"
9
10   db:
11     image: mysql:5.7
12     ports:
13       - "32000:3306"
14     environment:
15       MYSQL_ROOT_PASSWORD: root
16     volumes:
17       - ./db:/docker-entrypoint-initdb.d/:ro

```

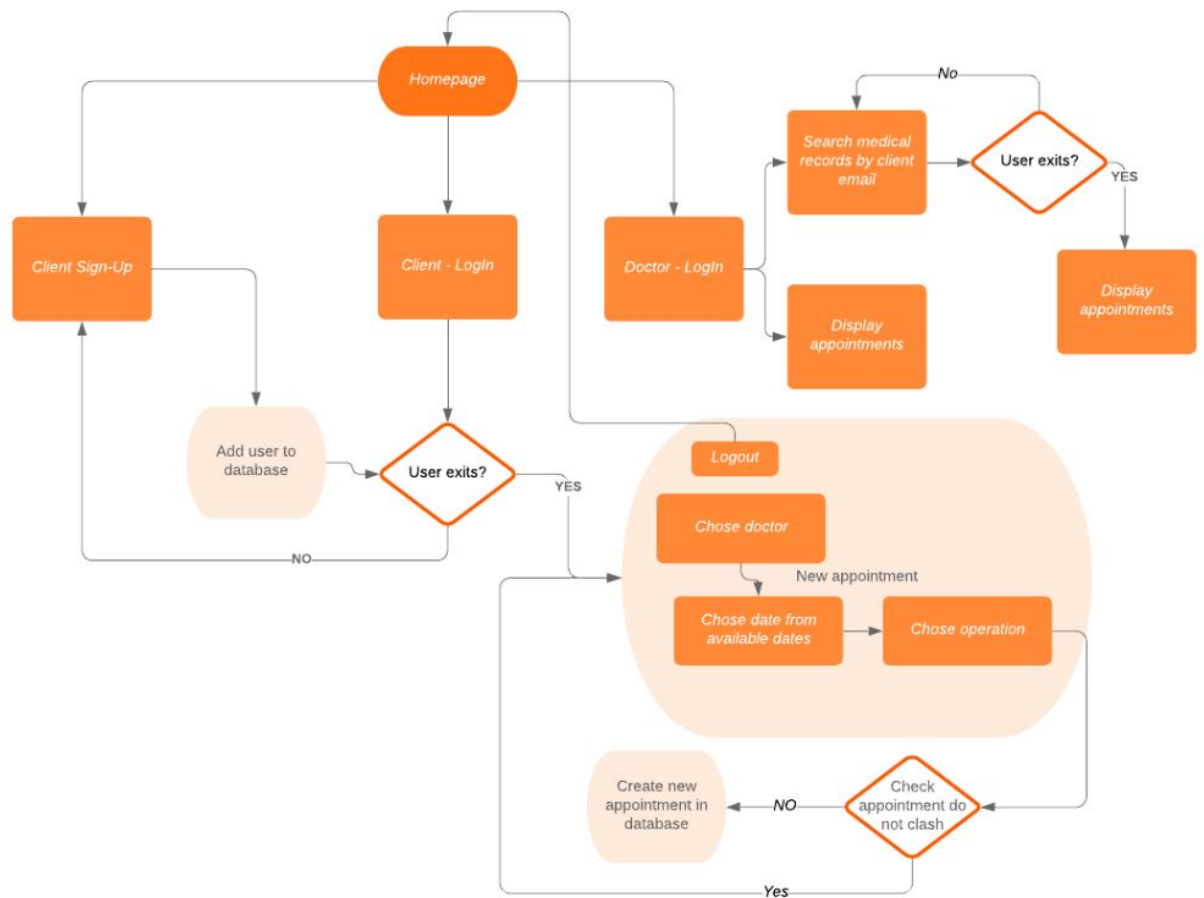
Serviciul bazei de date foloseste fisierul `init.sql` pentru a asigura consistenta schemei bazei de date. Intreg codul SQL folosit de aplicatie se afla in acest fisier. De exemplu, pentru un *query* simplu ce selecteaza date din tabela **clienti** este creata o procedura in fisierul *init.sql* ce mai apoi este apelata din codul python.

■ Structura claselor

Principalele fisiere sursa:

1. *app.py* - contine operatiile logice efectuate in urma interactiunii dintre utilizatori si interfatele grafice
2. *appointmentDetails.py* - incapsuleaza date pentru gestiunea programrilor
3. *operations.py* - incapsuleaza date pentru gestiunea/afisarea operatiilor
4. *records.py* - incapsuleaza date pentru gestiunea/afisarea istoricului medical pentru fiecare client
5. *credentials.py* - functii pentru autentificare, formatare string-uri, prelucrare input
6. *mysqlDb.py* - functii pentru interactiunea cu baza de date. Contine functii de tip wrapper ce apeleaza proceduri stocate in baza de date.

■ Diagrama de stări și fluxul de lucru (workflow) pentru aplicație



Un utilizator obisnuit (client) poate crea un cont nou (Sing-Up - utilizatorul este adaugat in tabela clienti) sau se poate loga daca are un cont existent. Odata logat, poate crea o programare alegand o data disponibila, un doctor si o operatie dintr-o lista de tip drop-down.

Un medic (adaugat in baza de date anterior) se poate loga pentru a vizualiza programarile sale (un medic nu poate vedea programarile altui medic) si poate cauta cu ajutorul e-mail-ului istoricul medical al unui client.

Cand o programare din tabela **programari** expira (*programare.data < data_curenta()*) aceasta este adaugata automat (cu ajutorul unui event) in tabela **istoric** si eliminata din tabela **programari**.

- Prezentarea modului în care se face conexiunea cu baza de date

Conexiunea la baza de date se realizeaza cu ajutorul bibliotecii *mysql.connector.py*.

```
mydb = mysql.connector.connect(  
    # host="db",  
    host="localhost",  
    user="root",  
    passwd="root",  
    # port="3306",  
    port="32000",  
    database="dent"  
)
```

- Capturi de ecran pentru interfețe și rapoarte

Pagina de start:

ANTODENT

[Home](#)

[Services](#)

[About Us](#)

[Doctors](#)

[Testimonials](#)

[Blog](#)

[Contact](#)

WELCOME TO ANTODENT

We Care For Your Smile

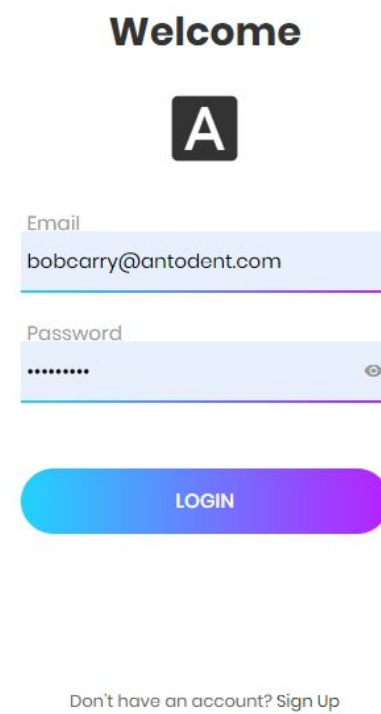
We strive to deliver the best dental medical services on the market. We want to make your life beautiful and simple. Need an apointment?

[Sign Up](#) or [Log In](#) ! Fast and easy!

[Contact Us](#)



Pagina de Log-In:



Welcome

A

Email

bobcarry@antodent.com

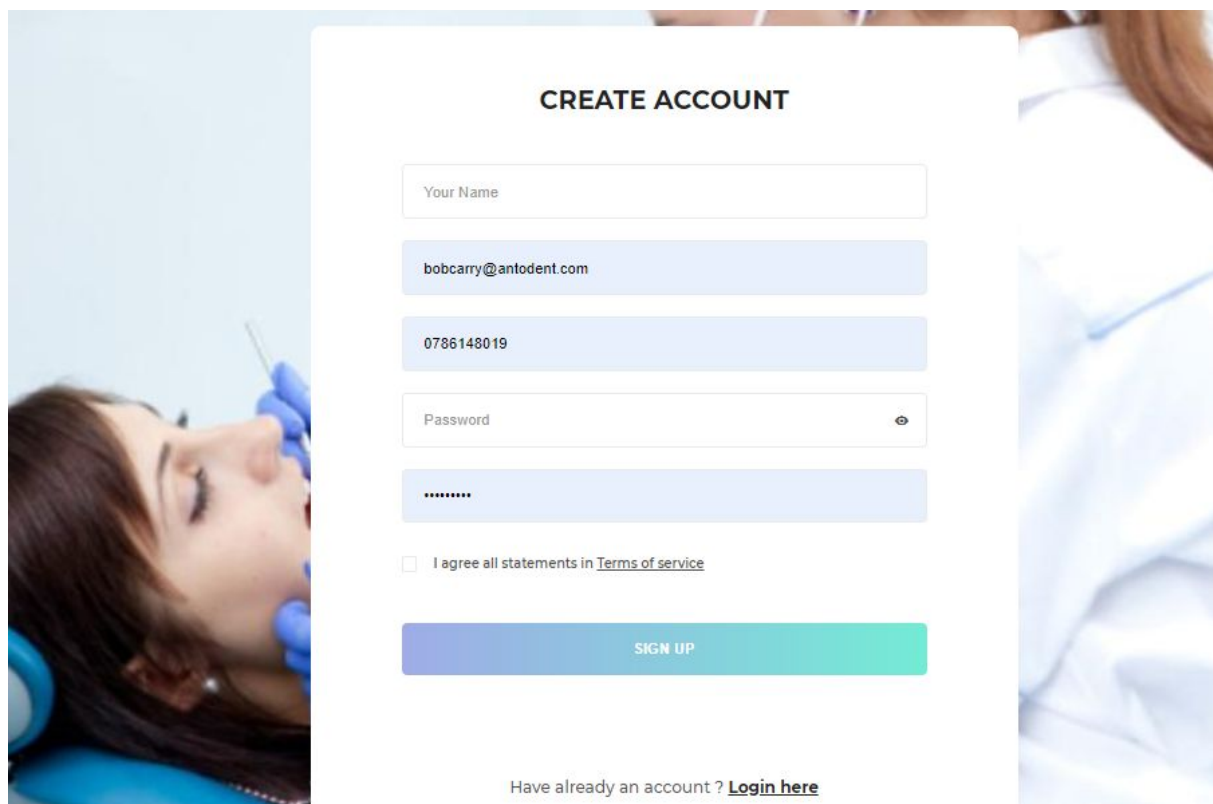
Password

••••••••

LOGIN

Don't have an account? [Sign Up](#)

Pagina Sing-Up:



CREATE ACCOUNT

Your Name

bobcarry@antodent.com

0786148019

Password

••••••••

☐ I agree all statements in [Terms of service](#)

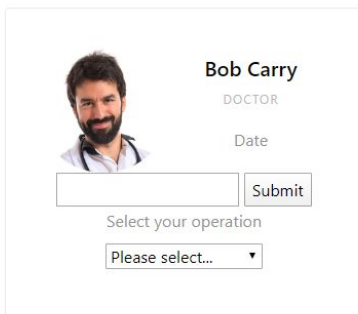
SIGN UP

Have already an account ? [Login here](#)

Pagina programari:

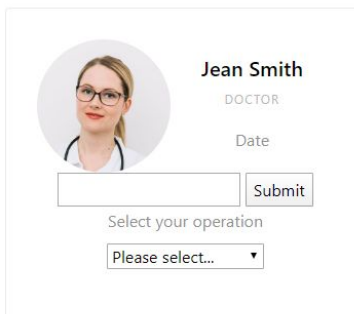
QUICK AND EASY!

Make your appointment right now!



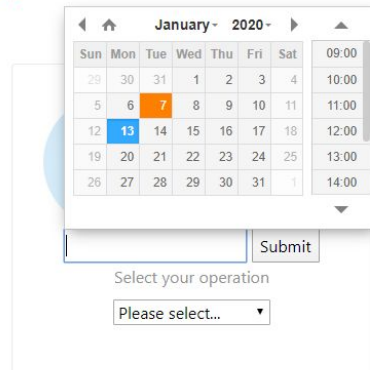
Bob Carry
DOCTOR
Date

Select your operation



Jean Smith
DOCTOR
Date

Select your operation




Calendar overlay: January - 2020 -
Sun Mon Tue Wed Thu Fri Sat
29 30 31 1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30 31
Hours: 09:00, 10:00, 11:00, 12:00, 13:00, 14:00

Select your operation

Vizualizare programari (pentru medici):

| Client name | Telephone | Date | Hour | Operation |
|-------------|------------|------------|----------|----------------|
| test client | 0722110234 | 2020-01-13 | 12:00:00 | Root Canals |
| test client | 0722110234 | 2020-02-18 | 9:00:00 | Gum Surgery |
| test client | 0722110234 | 2020-03-03 | 12:00:00 | Tooth Cleaning |

Cautare istoric medical (pentru medici):

AntoDent

Search medical records

Vizualizare istoric medical (pentru medici):

| Client name | | Telephone |
|-------------|--|------------|
| test client | | 0722110234 |

| Date | Operation |
|------------|------------------|
| 2017-04-20 | Tooth Extraction |
| 2017-04-20 | Tooth Extraction |
| 2020-04-20 | Tooth Extraction |
| 2020-04-20 | Dental Radiology |

■ Concluzii

Arhitectura bazata pe microservicii permite scalarea aplicatiei pe orizontala. Daca este nevoie de incarcarea rapida a paginilor web se pot lansa mai multe servere web ce servesc aceste pagini, servere ce pot comunica cu un singur serviciu ce interactioneaza cu baza de date. Similar, se pot porni cu usurinta mai multe servicii ce interactioneaza cu baza de date.

Pentru realizarea interfetelor grafice am utilizat resurse open-source (HTML, CSS, Javascript/ jQuery) mentionate in sectiunea *Bibliografie*.

Codul SQL este usor de parcurs, fiind stocat intr-un singur fisier (init.sql). Implementarea tuturor query-urilor cu ajutorul procedurilor stocate in baza de date face usor de utilizat intreaga functionalitate a bazei de date cu orice limbaj.

■ Bibliografie

Colorlib <https://colorlib.com/wp/templates/>

Datetimepicker <https://xdsoft.net/jqplugins/datetimepicker/>