Aplicatie pentru firma de service de dispositive

1. **Definirea cerințelor pentru aplicația aleasă**
    a. **Lista funcționalităților**
        - **Funcționalități pentru Clienți**
            - **Date personale**: Vizualizarea și actualizarea datelor personale.
            - **Adăugare dispozitiv:** Permite clientului să adauge un dispozitiv nou pentru reparații
            - **Vizualizare dispozitive:** Vizualizare dispozitivelor asociate contului clientului.
            - **Istoric reparații:** Vizualizarea istoricului reparațiilor pentru dispozitivele aduse.
            - **Vizualizare facturi:** Accesarea facturilor emise pentru client.
            - **Servicii disponibile:** Consultarea listei de servicii oferite de firmă și estimarea costurilor.
        - **Funcționalități pentru Administratori**
            - **Gestionare utilizatori:** Vizualizarea, crearea conturilor de utilizatori.
            - **Asignare tehnicieni:** Asocierea tehnicienilor la reparații cu statusul „în așteptare”.
            - **Reparații finalizate:** Vizualizarea reparațiilor finalizate.
            - **Top clienți după dispozitive:** Lista clienților ordonați descrescător după numărul de dispozitive aduse pentru reparații.
            - **Clienți fără dispozitive:** Identificarea clienților care nu au adus dispozitive.
            - **Clienți cu facturi peste media:** Lista clienților care au o sumă totală facturată mai mare decât media tuturor facturilor.
            - **Finalizare reparații:** Schimbarea statusului unei reparații la „finalizată”.

- **Ștergere reparații:** Eliminarea unei reparații din system doar dacă aceasta se află “În așteptare”.
- **Ștergere tehnicieni:** Eliminarea unui tehnician din baza de date.
- **Reparații, servicii aferente, cost:** Vizualizarea reparațiilor și serviciilor asociate, împreună cu costurile totale
- **Reparații cu mai multe servicii:** Identificarea reparațiilor care au asociate mai multe servicii.
- **Statistici servicii:** Generarea statisticilor pentru serviciile oferite (de exemplu, câți clienți au utilizat **fiecare serviciu).**

b. Identificarea tabelelor, stabilirea câmpurilor și a tipurior de date

- Clienți
  - id_client - INT (PRIMARY KEY, AUTO_INCREMENT)
  - nume - VARCHAR(50) (NOT NULL)
  - prenume - VARCHAR(50) (NOT NULL)
  - adresa - VARCHAR(100)
  - telefon - VARCHAR(15)
  - email - VARCHAR(50)
  - CNP - VARCHAR(13) (UNIQUE, NOT NULL)
- Administratori
  - id_administrator - INT (PRIMARY KEY, AUTO_INCREMENT)
  - nume_complet - VARCHAR(100) (NOT NULL)
  - adresa - VARCHAR(100)
  - telefon - VARCHAR(15)
  - email - VARCHAR(50) (NOT NULL)
  - CNP - VARCHAR(13) (UNIQUE, NOT NULL)
- Utilizatori
  - id_utilizator - INT (PRIMARY KEY, AUTO_INCREMENT)
  - nume_utilizator - VARCHAR(50) (NOT NULL)
  - parola - VARCHAR(50) (NOT NULL)
  - rol - ENUM('client', 'administrator') (NOT NULL)

  - id_client - INT (FOREIGN KEY → Clienti(id_client))
  - id_administrator - INT (FOREIGN KEY → Administratori(id_administrator))
- Dispozitive
  - id_dispozitiv - INT (PRIMARY KEY, AUTO_INCREMENT)
  - tip - VARCHAR(50) (NOT NULL)
  - marca - VARCHAR(50) (NOT NULL)
  - model - VARCHAR(50) (NOT NULL)
  - id_client - INT (FOREIGN KEY → Clienti(id_client))
- Reparații
  - id_reparatie - INT (PRIMARY KEY, AUTO_INCREMENT)
  - data - DATE (NOT NULL)
  - status - VARCHAR(20) (NOT NULL)
  - id_dispozitiv - INT (FOREIGN KEY → Dispozitive(id_dispozitiv))
- Servicii
  - id_serviciu - INT (PRIMARY KEY, AUTO_INCREMENT)
  - denumire - VARCHAR(50) (NOT NULL)
  - pret - DECIMAL(10, 2) (NOT NULL)
  - durata - INT (NOT NULL, în minute)
- Tehnicieni
  - _id_tehnician - INT (PRIMARY KEY, AUTO_INCREMENT)
  - nume_tehnician - VARCHAR(50) (NOT NULL)
  - prenume_tehnician - VARCHAR(50) (NOT NULL)
  - specializare - VARCHAR(50)
  - numar_telefon - VARCHAR(15)
- Facturi
  - id_factura - INT (PRIMARY KEY, AUTO_INCREMENT)
  - data_facturare - DATE (NOT NULL)
  - suma - DECIMAL(10, 2) (NOT NULL)
  - id_client - INT (FOREIGN KEY → Clienti(id_client))
- Reparatii_Servicii (Tabel Intermediar)

- id_reparatie - INT (FOREIGN KEY → Reparatii(id_reparatie))
- id_serviciu - INT (FOREIGN KEY → Servicii(id_serviciu))
- **Cheie Primară:** (id_reparatie, id_serviciu)

- Reparatii_Tehnicieni (Tabel Intermediar)
  - id_reparatie - INT (FOREIGN KEY → Reparatii(id_reparatie))
  - id_tehnician - INT (FOREIGN KEY → Tehnicieni(id_tehnician))
  - **Cheie Primară:** (id_reparatie, id_tehnician)

c. Constrângeri de integritate impuse

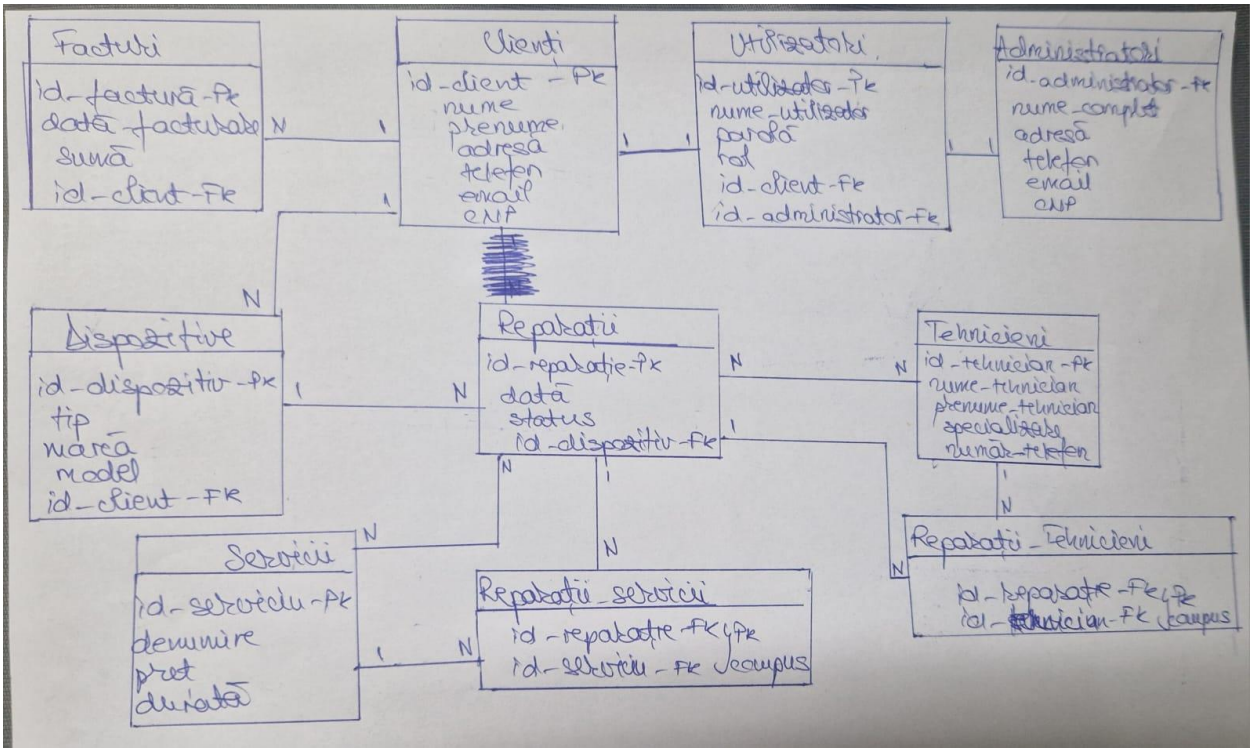
- Chei primare (PRIMARY KEY): id_client, id_utilizator, id_administrator, id_dispozitiv, id_reparatie, id_serviciu, id_tehnician, id_factura; tabelele intermediare au chei primare compuse.
- Chei străine (FOREIGN KEY):
  - id_client în Dispozitive → Clienti(id_client).
  - id_dispozitiv în Reparatii → Dispozitive(id_dispozitiv).
  - id_reparatie în Reparatii_Servicii → Reparatii(id_reparatie).
  - id_serviciu în Reparatii_Servicii → Servicii(id_serviciu).
  - id_reparatie în Reparatii_Tehnicieni → Reparatii(id_reparatie).
  - id_tehnician în Reparatii_Tehnicieni → Tehnicieni(id_tehnician).
- Unicitate (UNIQUE): CNP este unic în tabelele Clienti și Administratori.
- Not Null (NOT NULL): nume, prenume în Clienti, nume_utilizator, parola, rol în Utilizatori, tip, marca, model în Dispozitive, data și status în Reparatii.
- Enumerație (ENUM): rol în Utilizatori permite doar valorile client și administrator.

d. Relații între tabele

1) **Utilizatori - Clienți (1-1):** Fiecare utilizator asociat unui client reprezintă contul acelui client în sistem și fiecare

utilizator asociat unui client reprezintă contul acelui client în sistem.

2) **Utilizatori - Administratori (1-1):** Fiecare utilizator asociat unui administrator reprezintă contul acelui administrator în sistem, fiecare administrator din tabelul Administratori poate avea un singur cont în tabelul Utilizatori.
3) **Clienți - Dispozitive (1-N):** Un client poate deține mai multe dispozitive, un dispozitiv aparține unui singur client.
4) **Clienți - Facturi (1-N):** Fiecare client poate avea mai multe facturi associate, o factură aparține unui singur client.
5) **Dispozitive - Reparații (1-N):** Fiecare dispozitiv poate avea mai multe reparații asociate.
6) **Reparații - Tehnicieni (N-N):** O reparație poate implica mai mulți tehnicieni, iar un tehnician poate lucra la mai multe reparații.
7) **Reparații - Servicii (N-N):** O reparație poate include mai multe servicii, iar un serviciu poate fi utilizat în mai multe reparații.



2. Funcționarea aplicației

Aplicația este o platformă web care permite gestionarea clienților, dispozitivelor, reparațiilor și utilizatorilor în funcție de rolul acestora (client sau administrator).

1. Conectarea bazei de date

- Aplicația folosește **MySQL** pentru stocarea informațiilor, incluzând tabele precum:
  - Clienți
  - Utilizatori
  - Dispozitive
  - Reparații
  - Servicii
  - Tehnicieni.
- Conexiunea la baza de date este realizată cu **Node.js**, folosind biblioteca mysql2.
- Detaliile de conectare sunt configurate într-un fișier .env pentru securitate și flexibilitate.

2. Backend Node.js

- **Serverul Node.js** gestionează operațiile CRUD (Create, Read, Update, Delete) prin rute API:
  - **GET**: pentru afișarea datelor (ex. lista reparațiilor sau utilizatorilor).
  - **POST**: pentru adăugarea datelor (ex. adăugarea unui client sau tehnician).
  - **PUT**: pentru actualizarea datelor (ex. modificarea datelor personale ale unui client).
  - **DELETE**: pentru ștergerea datelor (ex. eliminarea reparațiilor sau tehnicienilor inactivi).

- Comunicarea cu frontend-ul se face prin format **JSON**.

3. Frontend web

- **Interfața web** este realizată cu **HTML**, **CSS** și **JavaScript**, având următoarele componente:
  - **Dashboard pentru clienți**:
    - Vizualizarea datelor personale și a dispozitivelor.
    - Istoricul reparațiilor, inclusiv serviciile și costurile asociate.
    - Formular pentru actualizarea informațiilor personale.
  - **Dashboard pentru administratori**:
    - Gestionarea utilizatorilor: vizualizare, adăugare, modificare.
    - Vizualizarea reparațiilor și raportarea serviciilor populare sau clienților activi.
    - Asignarea tehnicienilor pentru reparații.
- Aplicația oferă un **design responsive** și utilizatorii beneficiază de feedback prin alerte și mesaje în timp real.

4. Gestionarea dispozitivelor și reparațiilor

- **Clienți**:
  - Pot adăuga dispozitive pentru reparații.
  - Pot vizualiza istoricul reparațiilor și starea lor (ex. Finalizată, În așteptare).
- **Administratori**:
  - Pot asigna tehnicieni la reparații.
  - Pot vedea detalii despre reparații, inclusiv serviciile asociate și costurile totale.

5. Comunicarea între backend și frontend

- Aplicația folosește **fetch API** pentru a interacționa cu serverul prin rute REST:
    - **GET**: pentru afișarea datelor din tabele (ex. dispozitive, reparații, utilizatori).
    - **POST** și **PUT**: pentru adăugarea și actualizarea informațiilor.
    - **DELETE**: pentru ștergerea informațiilor redundante sau inexacte.

6. Securitate și validare

- **Autentificarea utilizatorilor**:
    - Se verifică dacă ID-ul clientului sau administratorului este stocat în sessionStorage sau localStorage.
    - Dacă utilizatorul nu este autentificat, este redirecționat către pagina de login.
- **Validarea datelor**:
    - Este realizată atât la nivel de client (formulare interactive), cât și la nivel de server (API).
- **Protecția datelor**:
    - Detaliile critice sunt gestionate cu fișiere .env pentru a evita expunerea informațiilor sensibile.

7. Rapoarte și statistici

- Aplicația permite administratorilor să genereze rapoarte precum:
    - Lista reparațiilor complexe (cu mai multe servicii).
    - Clienții cei mai activi (după numărul dispozitivelor reparate).
    - Serviciile populare, în funcție de numărul de utilizări.

Interogări

- Simple **cu JOIN-URI**
    1) **Istoricul reparațiilor unui client -** Această interogare returnează istoricul complet al reparațiilor efectuate pentru dispozitivele unui client specific, incluzând detalii despre dispozitive și statusul fiecărei reparații.

       SELECT r.id_reparatie, r.data, r.status, d.tip AS tip_dispozitiv, d.marca, d.model
       FROM reparatii r JOIN dispozitive d ON r.id_dispozitiv = d.id_dispozitiv
       WHERE d.id_client = ? ORDER BY r.data DESC;

    2) **Endpoint pentru obținerea tuturor utilizatorilor -** Acest endpoint returnează lista tuturor utilizatorilor din baza de date, incluzând detalii suplimentare despre clienții sau administratorii asociați fiecărui utilizator.

       SELECT u.id_utilizator, u.nume_utilizator, u.rol, c.nume AS nume_client, c.prenume AS prenume_client, a.nume_complet AS nume_administrator
       FROM utilizatori u LEFT JOIN clienti c ON u.id_client = c.id_client
       LEFT JOIN administratori a ON
       u.id_administrator = a.id_administrator;

    3) **Reparațiile cu serviciile associate (pentru tabelul din asignare tehnicieni) -** Această interogare folosește LEFT JOIN pentru a lega tabelul reparatii cu reparatii_servicii și servicii, astfel încât să preia toate reparațiile, inclusiv cele fără servicii asociate (datorită naturii LEFT JOIN). Detaliile dispozitivelor sunt obținute printr-un JOIN între reparatii și dispozitive, iar rezultatele sunt filtrate pentru reparațiile cu status = 'In asteptare'. Prin GROUP_CONCAT, serviciile asociate fiecărei reparații sunt concatenate într-o listă separată prin virgulă, iar GROUP BY grupează datele pe baza fiecărei reparații și dispozitivului său, astfel încât rezultatul să fie unificat într-un singur rând pentru fiecare reparație. Scopul interogării este să returneze reparațiile în așteptare, împreună cu detalii despre dispozitive și serviciile necesare pentru fiecare reparație.

       SELECT r.id_reparatie, d.tip AS tip_dispozitiv, d.marca AS marca_dispozitiv, d.model AS model_dispozitiv, r.status,

```
GROUP_CONCAT(s.denumire SEPARATOR ', ') AS
servicii_necesare
FROM reparatii r JOIN dispozitive d ON r.id_dispozitiv =
d.id_dispozitiv
                    LEFT JOIN reparatii_servicii rs ON r.id_reparatie =
rs.id_reparatie
                    LEFT JOIN servicii s ON rs.id_serviciu =
s.id_serviciu
WHERE r.status = 'In asteptare'
GROUP BY r.id_reparatie, d.tip, d.marca, d.model, r.status;
```

4) **Servicii și numărul total de clienți care le-au ales -** Această interogare returnează lista serviciilor oferite de firmă, împreună cu numărul total de clienți distincti care au ales fiecare serviciu. Scopul este de a analiza popularitatea fiecărui serviciu în rândul clienților.

```
SELECT s.denumire AS Serviciu, COUNT(DISTINCT c.id_client)
AS NrClienti
FROM servicii s JOIN reparatii_servicii rs ON s.id_serviciu =
rs.id_serviciu
                    JOIN reparatii r ON rs.id_reparatie = r.id_reparatie
                    JOIN dispozitive d ON r.id_dispozitiv =
                    d.id_dispozitiv
                    JOIN clienti c ON d.id_client = c.id_client
                    GROUP BY s.id_serviciu, s.denumire

                     ORDER BY NrClienti DESC;
```

- **Complexe cu subcereri**
  1. **Vizualizarea reparațiilor finalizate - Interogare complexă cu subcerere în clauza SELECT** - Această interogare **vizualizează lista reparațiilor finalizate** din sistem, oferind detalii despre dispozitivele reparate și calculând **costul total al fiecărei reparații**. Costul total este obținut folosind o **subinterogare** care calculează suma prețurilor serviciilor asociate fiecărei reparații.

```
SELECT
    r.id_reparatie,
    d.tip AS tip_dispozitiv,
    d.marca AS marca_dispozitiv,
    d.model AS model_dispozitiv,
```

```
    r.data,
    r.status,
    (
        SELECT SUM(s.pret)
        FROM reparatii_servicii rs
        JOIN servicii s ON rs.id_serviciu = s.id_serviciu
        WHERE rs.id_reparatie = r.id_reparatie
    ) AS cost_total
FROM reparatii r
JOIN dispozitive d ON r.id_dispozitiv = d.id_dispozitiv
WHERE r.status = 'Finalizata';
```

2. **Vizualizarea clienților ordonați după numărul de dispozitive aduse la reparat - Interogare complexă cu subinterogare în clauza ORDER BY -** Această interogare returnează lista clienților, ordonată descrescător după numărul de dispozitive aduse la reparat. Folosește o subinterogare în clauza ORDER BY pentru a calcula numărul de dispozitive asociate fiecărui client.

```
SELECT
    c.nume AS Nume,
    c.prenume AS Prenume
FROM clienti c
ORDER BY (
    SELECT COUNT(d.id_dispozitiv)
FROM dispozitive d
    WHERE d.id_client = c.id_client
) DESC;
```

3. **Obținerea listei de clienți care nu au adus dispozitive pentru reparații - Interogare complexă cu subinterogare în clauza WHERE -** Această interogare returnează lista clienților care nu au adus niciun dispozitiv pentru reparații. Utilizează o subinterogare în clauza WHERE pentru a exclude clienții care au dispozitive înregistrate în tabelul dispozitive.

```
SELECT c.nume, c.prenume
FROM clienti c
WHERE c.id_client NOT IN (
    SELECT d.id_client
    FROM dispozitive d
);
```

4. **Clienții cu suma totală facturată mai mare decât media facturilor - Interogare complexă cu subinterogare în clauza HAVING -** Această interogare selectează clienții a căror sumă totală facturată este mai mare decât media tuturor facturilor.

Utilizează o subinterogare în clauza HAVING pentru a compara suma totală a facturilor fiecărui client cu media generală.

SELECT c.nume, c.prenume, SUM(f.suma) AS suma_totala FROM clienti c JOIN facturi f ON c.id_client = f.id_client GROUP BY c.id_client, c.nume, c.prenume HAVING SUM(f.suma) > (SELECT AVG(suma) FROM facturi);

5. **Preluarea tehnicienilor inactivi (folosită în ștergerea de tehnicieni) - Interogare complexă cu subinterogare în clauza WHERE -** Această interogare identifică tehnicienii inactivi, adică cei care nu sunt asociați niciunei reparații. Utilizează o subinterogare în clauza WHERE pentru a exclude tehnicienii care au lucrat la cel puțin o reparație.

   SELECT t.id_tehnician, t.nume_tehnician, t.prenume_tehnician, t.specializare FROM tehnicieni t WHERE t.id_tehnician NOT IN (SELECT DISTINCT id_tehnician FROM reparatii_tehnicieni);
6. **Asignarea unui tehnician la o reparație -** Această interogare permite asocierea unui tehnician la o reparație specifică, doar dacă: Reparația are statusul In asteptare.
   SELECT r.id_reparatie, r.data, r.status, d.tip AS tip_dispozitiv, d.marca, d.model
   FROM reparatii r
   JOIN dispozitive d ON r.id_dispozitiv = d.id_dispozitiv
   WHERE r.status = 'In asteptare';

   Tehnicianul există în baza de date.
   SELECT id_tehnician, nume_tehnician, prenume_tehnician, specializare
   FROM tehnicieni;

   INSERT INTO reparatii_tehnicieni (id_reparatie, id_tehnician) SELECT ?, ? WHERE EXISTS (SELECT 1 FROM reparatii WHERE id_reparatie = ? AND status = 'In asteptare') AND EXISTS (SELECT 1 FROM tehnicieni WHERE id_tehnician = ?);

- **Update**
  1) **Actualizarea datelor personale ale unui client** - permite actualizarea datelor personale ale unui client identificat prin id_client.

     UPDATE clienti SET nume = ?, prenume = ?, adresa = ?, telefon = ?, email = ? WHERE id_client = ? ;
  2) **Actualizarea statusului unei reparații la Finalizata**

```
UPDATE reparatii
SET status = 'Finalizata'
WHERE id_reparatie = ?
AND status = 'In lucru';
```

- **Insert**
  - **Adăugarea unui dispozitiv, reparațiilor aferente și crearea facturii**

    ```
    INSERT INTO dispozitive (tip, marca, model, id_client) VALUES (?, ?, ?, ?);
    INSERT INTO reparatii (data, status, id_dispozitiv)
                VALUES (NOW(), 'In asteptare', ?);
    INSERT INTO reparatii_servicii (id_reparatie, id_serviciu)
                  VALUES (?, ?);
    INSERT INTO facturi (data_facturare, suma, id_client)
           VALUES (NOW(), ?, ?);
    ```
  - **Crearea unui nou cont de utilizator (SIGNUP)**

    ```
    INSERT INTO clienti (nume, prenume, adresa, telefon, email, CNP)
    VALUES (?, ?, ?, ?, ?, ?);
    ```
  - I**nserare utilizatori noi de către admini –** exact ca la signup, dar si pentru administrator
- **Delete**
  - **Ștergerea unei reparații -** Acest endpoint permite ștergerea unei reparații și a serviciilor asociate acesteia doar dacă reparația are statusul In asteptare. Este un proces în doi pași:
    Se șterg mai întâi serviciile asociate din tabelul intermediar reparatii_servicii.

    ```
    DELETE FROM reparatii_servicii
       WHERE id_reparatie = ?;
    ```

    Se șterge reparația propriu-zisă din tabelul reparatii.

    ```
    DELETE FROM reparatii
         WHERE id_reparatie = ? AND status = 'In asteptare';
    ```
  - **Ștergerea unui technician inactive -** Acest endpoint permite ștergerea unui tehnician din baza de date doar dacă tehnicianul nu mai are asociate reparații în tabela Reparatii_Tehnicieni.

    ```
    SELECT t.id_tehnician, t.nume_tehnician, t.prenume_tehnician,
    t.specializare
       FROM tehnicieni t
       WHERE t.id_tehnician NOT IN (
         SELECT DISTINCT id_tehnician FROM reparatii_tehnicieni
       );
    DELETE FROM Reparatii_Tehnicieni
    ```

```
   WHERE id_tehnician = ?;
DELETE FROM Tehnicieni
     WHERE id_tehnician = ? AND id_tehnician NOT IN (
       SELECT DISTINCT id_tehnician FROM Reparatii_Tehnicieni
     );
```