Sistem de management al unor clinici medicale

1. Script creare tabele

-- clinica\_medicala.Clinica definition

CREATE TABLE `Clinica` (

`IdClinica` int(11) NOT NULL AUTO\_INCREMENT,

`NumeClinica` varchar(100) NOT NULL,

`TelefonClinica` varchar(11) NOT NULL,

`AdresaClinica` varchar(250) NOT NULL,

PRIMARY KEY (`IdClinica`),

UNIQUE KEY `Clinica\_NumeClinica\_UNIQUE` (`NumeClinica`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Specializare definition

CREATE TABLE `Specializare` (

`IdSpecializare` int(11) NOT NULL AUTO\_INCREMENT,

`NumeSpecializare` varchar(100) NOT NULL,

PRIMARY KEY (`IdSpecializare`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Utilizator definition

CREATE TABLE `Utilizator` (

`IdUtilizator` int(11) NOT NULL AUTO\_INCREMENT,

`Email` varchar(100) NOT NULL,

`Parola` varchar(100) NOT NULL,

`Rol` enum('pacient','admin','medic','asistent') NOT NULL DEFAULT 'pacient',

`DataCrearii` datetime NOT NULL DEFAULT current\_timestamp(),

`DataModificarii` datetime DEFAULT NULL,

PRIMARY KEY (`IdUtilizator`),

UNIQUE KEY `User\_Email\_UNIQUE` (`Email`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Asistent definition

CREATE TABLE `Asistent` (

`IdAsistent` int(11) NOT NULL AUTO\_INCREMENT,

`IdUtilizator` int(11) NOT NULL,

`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

`Email` varchar(100) NOT NULL,

`Telefon` varchar(11) NOT NULL,

`IdClinica` int(11) NOT NULL,

`DataCrearii` datetime NOT NULL DEFAULT current\_timestamp(),

PRIMARY KEY (`IdAsistent`),

UNIQUE KEY `Asistent\_Email\_UNIQUE` (`Email`),

UNIQUE KEY `Asistent\_Telefon\_UNIQUE` (`Telefon`),

KEY `Asistent\_Clinica\_FK` (`IdClinica`),

KEY `Asistent\_Utilizator\_FK` (`IdUtilizator`),

CONSTRAINT `Asistent\_Clinica\_FK` FOREIGN KEY (`IdClinica`) REFERENCES `Clinica` (`IdClinica`),

CONSTRAINT `Asistent\_Utilizator\_FK` FOREIGN KEY (`IdUtilizator`) REFERENCES `Utilizator` (`IdUtilizator`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Medic definition

CREATE TABLE `Medic` (

`IdMedic` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

`Email` varchar(100) NOT NULL,

`Telefon` varchar(11) NOT NULL,

`IdSpecializare` int(11) NOT NULL,

`IdClinica` int(11) NOT NULL,

`IdUtilizator` int(11) NOT NULL,

`DataCrearii` datetime NOT NULL DEFAULT current\_timestamp(),

PRIMARY KEY (`IdMedic`),

UNIQUE KEY `Medic\_Email\_UNIQUE` (`Email`),

UNIQUE KEY `Medic\_Telefon\_UNIQUE` (`Telefon`),

KEY `Medic\_Clinica\_FK` (`IdClinica`),

KEY `Medic\_Specializare\_FK` (`IdSpecializare`),

KEY `Medic\_Utilizator\_FK` (`IdUtilizator`),

CONSTRAINT `Medic\_Clinica\_FK` FOREIGN KEY (`IdClinica`) REFERENCES `Clinica` (`IdClinica`),

CONSTRAINT `Medic\_Specializare\_FK` FOREIGN KEY (`IdSpecializare`) REFERENCES `Specializare` (`IdSpecializare`),

CONSTRAINT `Medic\_Utilizator\_FK` FOREIGN KEY (`IdUtilizator`) REFERENCES `Utilizator` (`IdUtilizator`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Pacient definition

CREATE TABLE `Pacient` (

`IdPacient` int(11) NOT NULL AUTO\_INCREMENT,

`Nume` varchar(100) NOT NULL,

`Prenume` varchar(100) NOT NULL,

`Email` varchar(100) NOT NULL,

`Telefon` varchar(11) NOT NULL,

`CNP` varchar(13) NOT NULL,

`DataNasterii` date NOT NULL,

`DataCrearii` datetime NOT NULL DEFAULT current\_timestamp(),

`DataModificarii` datetime DEFAULT NULL,

`IdUtilizator` int(11) DEFAULT NULL,

PRIMARY KEY (`IdPacient`),

UNIQUE KEY `Pacient\_Email\_UNIQUE` (`Email`),

UNIQUE KEY `Pacient\_Telefon\_UNIQUE` (`Telefon`),

UNIQUE KEY `Pacient\_CNP\_UNIQUE` (`CNP`),

KEY `Pacient\_Utilizator\_FK` (`IdUtilizator`),

CONSTRAINT `Pacient\_Utilizator\_FK` FOREIGN KEY (`IdUtilizator`) REFERENCES `Utilizator` (`IdUtilizator`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.PacientMedic definition

CREATE TABLE `PacientMedic` (

`IdPacient` int(11) NOT NULL,

`IdMedic` int(11) NOT NULL,

PRIMARY KEY (`IdPacient`,`IdMedic`),

KEY `IdMedic` (`IdMedic`),

CONSTRAINT `PacientMedic\_ibfk\_1` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`) ON DELETE CASCADE,

CONSTRAINT `PacientMedic\_ibfk\_2` FOREIGN KEY (`IdMedic`) REFERENCES `Medic` (`IdMedic`) ON DELETE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.ProgramDoctor definition

CREATE TABLE `ProgramDoctor` (

`IdProgramMedic` int(11) NOT NULL AUTO\_INCREMENT,

`IdMedic` int(11) NOT NULL,

`ZiuaSaptamanii` enum('luni','marti','miercuri','joi','vineri','sambata','duminica') NOT NULL DEFAULT 'luni',

`OraInceput` time NOT NULL,

`OraSfarsit` time NOT NULL,

PRIMARY KEY (`IdProgramMedic`),

KEY `ProgramDoctor\_Medic\_FK` (`IdMedic`),

CONSTRAINT `ProgramDoctor\_Medic\_FK` FOREIGN KEY (`IdMedic`) REFERENCES `Medic` (`IdMedic`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.ServiciuMedical definition

CREATE TABLE `ServiciuMedical` (

`IdServiciuMedical` int(11) NOT NULL AUTO\_INCREMENT,

`IdSpecializare` int(11) NOT NULL,

`Nume` varchar(100) NOT NULL,

`Pret` decimal(10,2) NOT NULL,

PRIMARY KEY (`IdServiciuMedical`),

KEY `ServiciuMedical\_Specializare\_FK` (`IdSpecializare`),

CONSTRAINT `ServiciuMedical\_Specializare\_FK` FOREIGN KEY (`IdSpecializare`) REFERENCES `Specializare` (`IdSpecializare`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Analize definition

CREATE TABLE `Analize` (

`IdAnaliza` int(11) NOT NULL AUTO\_INCREMENT,

`IdPacient` int(11) NOT NULL,

`IdSpecializare` int(11) NOT NULL,

`CodAcces` varchar(20) NOT NULL,

`DataAnaliza` datetime NOT NULL DEFAULT current\_timestamp(),

`DataPrimireAnaliza` datetime DEFAULT NULL,

`Status` enum('in curs','finalizata') NOT NULL DEFAULT 'in curs',

PRIMARY KEY (`IdAnaliza`),

UNIQUE KEY `Analize\_CodAcces\_UNIQUE` (`CodAcces`),

KEY `Analize\_Pacient\_FK` (`IdPacient`),

KEY `Analize\_Specializare\_FK` (`IdSpecializare`),

CONSTRAINT `Analize\_Pacient\_FK` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`),

CONSTRAINT `Analize\_Specializare\_FK` FOREIGN KEY (`IdSpecializare`) REFERENCES `Specializare` (`IdSpecializare`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.DetaliiAnaliza definition

CREATE TABLE `DetaliiAnaliza` (

`IdDetaliiAnaliza` int(11) NOT NULL AUTO\_INCREMENT,

`IdAnaliza` int(11) NOT NULL,

`NumeTest` varchar(100) NOT NULL,

`ValoareObtinuta` decimal(10,2) NOT NULL,

`IntervalReferinta` decimal(10,2) NOT NULL,

`UnitateMasura` varchar(10) NOT NULL,

PRIMARY KEY (`IdDetaliiAnaliza`),

KEY `DetaliiAnaliza\_Analize\_FK` (`IdAnaliza`),

CONSTRAINT `DetaliiAnaliza\_Analize\_FK` FOREIGN KEY (`IdAnaliza`) REFERENCES `Analize` (`IdAnaliza`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.ProgramareAnaliza definition

CREATE TABLE `ProgramareAnaliza` (

`IdProgramareAnaliza` int(11) NOT NULL AUTO\_INCREMENT,

`IdPacient` int(11) NOT NULL,

`IdAnaliza` int(11) NOT NULL,

`DataProgramarii` date NOT NULL,

`OraProgramarii` time NOT NULL,

`StatusProgramare` enum('in asteptare','confirmata','anulata') DEFAULT 'in asteptare',

PRIMARY KEY (`IdProgramareAnaliza`),

KEY `ProgramareAnaliza\_Analize\_FK` (`IdAnaliza`),

KEY `ProgramareAnaliza\_Pacient\_FK` (`IdPacient`),

CONSTRAINT `ProgramareAnaliza\_Analize\_FK` FOREIGN KEY (`IdAnaliza`) REFERENCES `Analize` (`IdAnaliza`),

CONSTRAINT `ProgramareAnaliza\_Pacient\_FK` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.ProgramareConsultatie definition

CREATE TABLE `ProgramareConsultatie` (

`IdProgramareConsultatie` int(11) NOT NULL AUTO\_INCREMENT,

`IdPacient` int(11) NOT NULL,

`IdMedic` int(11) NOT NULL,

`IdServiciuMedical` int(11) NOT NULL,

`DataProgramarii` date NOT NULL,

`OraProgramarii` time NOT NULL,

`StatusProgramare` enum('in asteptare','confirmata','anulata') NOT NULL DEFAULT 'in asteptare',

PRIMARY KEY (`IdProgramareConsultatie`),

KEY `Programare\_Pacient\_FK` (`IdPacient`),

KEY `Programare\_Medic\_FK` (`IdMedic`),

KEY `Programare\_ServiciuMedical\_FK` (`IdServiciuMedical`),

CONSTRAINT `Programare\_Medic\_FK` FOREIGN KEY (`IdMedic`) REFERENCES `Medic` (`IdMedic`),

CONSTRAINT `Programare\_Pacient\_FK` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`),

CONSTRAINT `Programare\_ServiciuMedical\_FK` FOREIGN KEY (`IdServiciuMedical`) REFERENCES `ServiciuMedical` (`IdServiciuMedical`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.ScrisoareMedicala definition

CREATE TABLE `ScrisoareMedicala` (

`IdScrisoareMedicala` int(11) NOT NULL AUTO\_INCREMENT,

`IdPacient` int(11) DEFAULT NULL,

`IdMedic` int(11) NOT NULL,

`IdProgramareConsultatie` int(11) NOT NULL,

`IdServiciuMedical` int(11) NOT NULL,

`IdSpecializare` int(11) NOT NULL,

`Diagnostic` varchar(200) NOT NULL,

`Continut` varchar(500) NOT NULL,

`DataCrearii` datetime NOT NULL DEFAULT current\_timestamp(),

`DataModificarii` datetime DEFAULT NULL,

PRIMARY KEY (`IdScrisoareMedicala`),

KEY `ScrisoareMedicala\_Pacient\_FK` (`IdPacient`),

KEY `ScrisoareMedicala\_Medic\_FK` (`IdMedic`),

KEY `ScrisoareMedicala\_ProgramareConsultatie\_FK` (`IdProgramareConsultatie`),

KEY `ScrisoareMedicala\_Specializare\_FK` (`IdSpecializare`),

KEY `ScrisoareMedicala\_ServiciuMedical\_FK` (`IdServiciuMedical`),

CONSTRAINT `ScrisoareMedicala\_Medic\_FK` FOREIGN KEY (`IdMedic`) REFERENCES `Medic` (`IdMedic`),

CONSTRAINT `ScrisoareMedicala\_Pacient\_FK` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`),

CONSTRAINT `ScrisoareMedicala\_ProgramareConsultatie\_FK` FOREIGN KEY (`IdProgramareConsultatie`) REFERENCES `ProgramareConsultatie` (`IdProgramareConsultatie`),

CONSTRAINT `ScrisoareMedicala\_ServiciuMedical\_FK` FOREIGN KEY (`IdServiciuMedical`) REFERENCES `ServiciuMedical` (`IdServiciuMedical`),

CONSTRAINT `ScrisoareMedicala\_Specializare\_FK` FOREIGN KEY (`IdSpecializare`) REFERENCES `Specializare` (`IdSpecializare`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

-- clinica\_medicala.Notificare definition

CREATE TABLE `Notificare` (

`IdNotificare` int(11) NOT NULL AUTO\_INCREMENT,

`IdProgramareConsultatie` int(11) DEFAULT NULL,

`IdProgramareAnaliza` int(11) DEFAULT NULL,

`IdPacient` int(11) NOT NULL,

`DataNotificare` datetime NOT NULL,

`TipNotificare` enum('reamintire','confirmare','anulata') NOT NULL DEFAULT 'reamintire',

PRIMARY KEY (`IdNotificare`),

KEY `Notificare\_ProgramareAnaliza\_FK` (`IdProgramareAnaliza`),

KEY `Notificare\_ProgramareConsultatie\_FK` (`IdProgramareConsultatie`),

KEY `Notificare\_Pacient\_FK` (`IdPacient`),

CONSTRAINT `Notificare\_Pacient\_FK` FOREIGN KEY (`IdPacient`) REFERENCES `Pacient` (`IdPacient`),

CONSTRAINT `Notificare\_ProgramareAnaliza\_FK` FOREIGN KEY (`IdProgramareAnaliza`) REFERENCES `ProgramareAnaliza` (`IdProgramareAnaliza`),

CONSTRAINT `Notificare\_ProgramareConsultatie\_FK` FOREIGN KEY (`IdProgramareConsultatie`) REFERENCES `ProgramareConsultatie` (`IdProgramareConsultatie`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_uca1400\_ai\_ci;

1. Relatii intre tabele
2. **Clinica - Medic** - Relatie **One-to-Many**

* O clinica poate avea mai multi medici, dar fiecare medic este asociat unei singure clinici.

1. **Clinica - Asistent** - Relatie **One-to-Many**

* O clinica poate avea mai multi asistenti, dar fiecare asistent este asociat unei singure clinici.

1. **Specializare - Medic** - Relatie **One-to-Many**

* O specializare poate avea mai multi medici, dar fiecare medic are o specializare.

1. **Utilizator - Medic** - Relatie **One-to-One**

* Fiecare medic este asociat unui cont de utilizator si nu poate avea mai multe conturi.

1. **Utilizator - Asistent** - Relatie **One-to-One**

* Fiecare asistent este asociat unui cont de utilizator si nu poate avea mai multe conturi.

1. **Utilizator - Pacient** - Relatie **One-to-One**

* Fiecare pacient este asociat unui cont de utilizator si nu poate avea mai multe conturi.

1. **Pacient - Analize** - Relatie **One-to-Many**

* Un pacient poate avea mai multe analize, dar fiecare analiza este asociata unui singur pacient.

1. **Specializare - Analize** - Relatie **One-to-Many**

* Fiecare analiza apartine unei singure specializari, dar o specializare poate fi folosita de mai multe ori pentru diferite analize.

1. **Analize - DetaliiAnaliza** - Relatie **One-to-Many**

* O analiza poate contine mai multe detalii (rezultate ale mai multor teste), dar fiecare detaliu este asociat unei singure analize.

1. **Medic - ProgramDoctor** - Relatie **One-to-Many**

* Un medic poate avea program definit pentru mai multe zile, dar fiecare inregistrare de program este asociata unui singur medic.

1. **Specializare - ServiciuMedical** - Relatie **One-to-Many**

* O specializare poate include mai multe servicii medicale (mai multe tipuri de consultatii sau proceduri medicale).

1. **Pacient - ProgramareAnaliza** - Relatie **One-to-Many**

* Un pacient poate avea mai multe programari pentru analize, dar fiecare programare este asociata unui singur pacient.

1. **Pacient - ProgramareConsultatie** - Relatie **One-to-Many**

* Un pacient poate avea mai multe programari pentru consultatii, dar fiecare programare este asociata unui singur pacient.

1. **Medic - ProgramareConsultatie** - Relatie **One-to-Many**

* Un medic poate avea mai multe programari de consultatii, dar fiecare programare este asociata unui singur medic.

1. **ServiciuMedical - ProgramareConsultatie** - Relatie **One-to-Many**

* Fiecare programare este pentru un anumit tip de serviciu medical (consultatie, o procedura specifica).

1. **Pacient - ScrisoareMedicala** - Relatie **One-to-Many**

* Un pacient poate avea mai multe scrisori medicale, dar fiecare scrisoare este asociata unui singur pacient.

1. **Medic - ScrisoareMedicala** - Relatie **One-to-Many**

* Un medic poate emite mai multe scrisori medicale, dar fiecare scrisoare este emisa de un singur medic.

1. **ProgramareConsultatie - ScrisoareMedicala** - Relatie **One-to-One**

* O scrisoare medicala este asociata unei singure programari de consultatie si invers.

1. **Notificare - ProgramareConsultatie și ProgramareAnaliza** - Relatie **One-to-Many**

* O notificare poate fi legata fie de o programare de consultatie, fie de o programare de analiza, dar programarile pot avea mai multe notificari.

1. **Pacient - Medic** - Relatie **Many-to-Many**

* Un pacient poate merge la mai multi doctori, iar un medic poate trata mai multi pacienti.

1. Gestionare aplicatie

### Log in:

* Se va face autentificarea cu email si parola.
* Daca utilizatorul a uitat parola, i se va trimite un mail pentru a reseta parola.
* Daca nu are cont, pacientul isi va crea un cont, in care i se vor cere:
  + nume
  + prenume
  + email
  + numar de telefon
  + data nasterii
  + cnp
* Utilizatorii pentru medici si asistenti vor fi creati de catre admin.

### Interfata principala:

* Adminul:
  + poate modifica / adauga clinici
  + poate modifica / adauga medici
  + poate modifica / adauga asistenti
  + poate modifica / adauga specializarile medicilor
  + poate modifica / adauga serviciile medicale pe care le ofera clinica
  + poate modifica / adauga programul doctorilor
* Pacientul:
  + poate vedea toate clinicile disponibile
  + poate vedea toate serviciile medicale oferite de fiecare clinica in parte
  + poate vedea toate specializarile medicilor din fiecare clinica
  + poate vedea programul doctorilor
  + isi poate face programari la consulatii sau analize medicale
  + isi poate vedea scrisorile medicale date de medicii la care a fost
  + isi poate vedea analizele medicale pe care le-a facut
    - acestea pot fi vazute chiar daca pacientul nu are cont creat, la clinica i se poate da un cod de acces
* Medicul:
  + poate vedea toate programarile pe care le-a avut si care urmeaza sa le aiba
  + poate accesa scrisorile medicale scrise de el pentru fiecare pacient pe care l-a avut
  + poate accesa fiecare analiza medicala a pacientului acestuia, chiar daca nu apartine de specializarea lui
  + poate crea cate o scrisoare medicala pentru fiecare programare
  + poate modifica o scrisoare medicala, pentru a mai adauga observatii
* Asistentul
  + poate vedea programarile pentru fiecare doctor in parte
  + poate vedea programul fiecarui doctor
  + poate vedea scrisorile medicale ale fiecarui pacient ce a fost in orice clinica
  + poate vedea analizele medicale ale fiecarui pacient ce a fost in orice clinica
  + poate face programari pentru pacienti atat la consultatii pentru orice medic din orice clinica, cat si la analize medicale, chiar daca pacientul are cont creat sau nu

### Notificari:

* Notificarile vor fi trimise pacientilor in momentul in care s-a facut o programare, o anulare a programarii, sau cu o zi inainte, pentru a reaminti pacientului ca are o programare.

