VeteriLand

Realizatori: Rotaru Alexandra-Beatrice, Zanceanu Ioan Robert

Grupa: A2, anul 2

Aplicatia pe care o vom implementa se va numi VeteriLand. Va reprezenta site-ul unui cabinet veterinar, in care detinatorii de animale vor putea face programari la diferiti doctori, intr-o anumita zi, la o anumita ora, dar numai dupa crearea unui cont de utilizator. Fiecare persoana, dupa ce isi va crea un cont, va fi introdusa in tabela Useri, indiferent daca va fi stapan sau doctor. Acestia vor fi diferentiati prin valorile coloanelor doctor si stapan din tabela Useri. Pentru a-si crea cont, o persoana poate sa aleaga dintre “Creare cont Stapan” si “Creare cont Doctor”. Cand se va crea un cont pentru stapan, se va pune valoarea 0 in campul doctor si 1 in campul stapan, iar atunci cand se va crea un cont pentru doctor, valorile celor doua coloane vor avea valorile inversate. Pentru logare, datele introduse ca username si password vor fi verificate in tabela Useri, pentru a securiza accesul la date. Fiecare consultatie va fi programata la diferenta de o ora, astfel incat sa nu existe intersectari pentru pacientii aceluiasi doctor. Dupa ce un pacient a fost consultat, acesta va fi introdus in tabela Tratament, care va contine o rubrica speciala, in care va fi trecut motivul vizitei la cabinet (consult periodic, o anumita problema de sanatate), cat si medicamentul ce i-au fost recomandate pentru administrare. Un doctor va avea acces la tabela Tratament, unde va introduce, dupa fiecare consult, informatii despre pacient.

Tabelele folosite:

1. Useri. Aceasta va contine id(cheie primara), email, password, doctor, stapan. Tabela va contine conturile care au fost create de catre utilizatori (atat stapani, cat si doctori). Pentru a avea acces la date va trebui, inainte, sa-ti faci un cont de utilizator. Informatiile care vor fi introduse pe pagina de logare vor fi verificate. Daca acestea sunt incluse in tabela Useri, adica daca contul exista, vei primi acces la date, altfel, nu te vei putea loga.
2. Doctori. Aceasta va contine id(cheie primara), nume, prenume, varsta, email, numar\_telefon. Tabela va contine informatii despre doctorii cabinetului veterinat.
3. DoctoriUseri. Aceasta va contine id(cheie primara), id\_user(cheie straina, reprezentand cheia primara din tabela Useri), id\_doctor(cheie straina, reprezentand cheia primara din tabela Doctori). Aceasta tabela reprezinta legatura dintre tabelele Useri si Doctori.
4. Tip\_Animal. Acesta va contine id(cheie primara), tip\_animal. Va contine toate tipurile de animale pe care le trateaza cabinetul veterinar.
5. Specializare. Aceasta va contine id(cheie primara), id\_doctor(cheie straina, reprezentand cheia primara din tabela Doctori), id\_animal(cheie straina, reprezentand cheia primara din tabela Tip\_Animal). Va reprezenta legatura dintre tabelele Doctori si Tip\_Animal. Astfel, fiecare doctor va fi specializat pe un anumit tip de animal (caine, pisica etc.). Tabela Doctor va contine id (cheie primara), numele, prenumele, varsta, email, numar\_telefon.
6. Stapani. Aceasta va contine id(cheie primara), nume, prenume, varsta, email, numar\_telefon. Tabela va contine informatii despre stapanii animalelor.
7. StapaniUseri. Aceasta va contine id(cheie primara), id\_user(cheie straina, reprezentand cheia primara din tabela Useri), id\_stapan(cheie straina, reprezentand cheia primara din tabela Stapani). Aceasta tabela reprezinta legatura dintre tabelele Useri si Stapani.
8. Pacienti. Aceasta va contine id(cheie primara), nume, varsta, tip\_animal. Tabela va contine informatii despre animalele stapanilor ce si-au creat cont.
9. Pacient\_Stapan. Aceasta va contine id(cheie primara), id\_pacient(cheie straina, reprezentand cheia primara din tabela Pacienti), id\_stapan(cheie straina, reprezentand cheia primara din tabela Stapani). Aceasta tabela reprezinta legatura dintre tabelele Pacienti si Stapani.
10. Tratamente. Aceasta va contine id(cheie primara), motiv(va reprezenta motivul pentru care a fost facuta programarea), medicatie(tratamentul ce i-a fost prescris pacientului dupa consult), tip\_animal. Aceasta tabela va reprezenta un istoric al programarilor, in care vor fi introduse datele in urma consultatiei, pentru a se tine evidenta pacientilor. Medicatia va fi data in functie de tipul animalului si de motivul pentru care a venit la cabinet(“Control periodic”, “Vaccin” etc.).
11. Pacient\_Tratament. Aceasta va contine id(cheie primara), id\_pacient(cheie straina, reprezentand cheia primara din tabela Pacienti), id\_tratament(cheie straina, reprezentand cheia primara din tabela Tratamente). Aceasta tabela reprezinta legatura dintre tabelele Pacienti si Tratamente.
12. Programari. Aceasta va contine id(cheie primara), id\_doctor(cheie straina, reprezentand cheia primara din tabela Doctori), id\_pacient(cheie straina, reprezentand cheia primara din tabela Pacienti), date(data la care este facuta programarea), ora(ora la care este facuta programarea), motiv(motivul programarii).

Indecsi :

1. Create index exista\_cont on useri(email, password); -> Ne va fi de folos pentru a verifica cu rapiditate in momentul logarii daca contul respective exista sau nu in baza de date. In cazul in care nu exista, utilizatorului nu ii este permisa logarea.

Views:

1. Create view find\_doctor(nume\_doctor, prenume\_doctor, specializare\_animal) as select nume, prenume, tip\_animal from doctori, tip\_animal, specializare where specializare.id\_doctor=doctori.id and specializare.id\_animal=tip\_animal.id; -> Folosit pentru a pune impreuna datele din tabelele Doctori si Tip\_Animal ce vor fi cautate de catre utilizatorii aplicatiei(doar cele dupa care dorim sa se faca cautarile: numele doctorului, prenumele doctorului, cat si tipul de animal de care se ocupa acesta).
2. Create view find\_programare\_for\_stapani(nume\_doctor, prenume\_doctor, data\_programare, ora\_programare) as select nume, prenume, data, ora from doctori, programari where doctori.id=programari.id\_doctor; -> Ne va ajuta sa verificam in momentul in care un utilizator doreste sa faca o programare, daca exista deja o programare la data si ora respectiva, la acel doctor. In acest caz, programarea nu va putea fi facuta.
3. Create view find\_programare\_for\_doctori(nume\_pacient, varsta\_pacient, data\_programare, ora\_programare, motiv\_programare) as select nume, varsta, data, ora, motiv from pacienti, programari where programari.id\_pacient=pacienti.id; -> Va contine toate programarile unui doctor. Astfel, doctoral va putea sa vada ce programari are la o anumita data (optional, poate verifica si in functie de ora).

Functii:

1. O functie pentru inserarea datelor in tabela Useri, Stapani/Doctori. Cand un utilizator al aplicatiei va completa formularul de inscriere, email-ul si parola care au fost alese de catre acestea vor fi trecute in tabela Useri, in coloanele cu email si password. Celelalte date vor fi introduse fie in tabela Stapan, fie in tabela Doctori, in functie de modul de creare ales de utilizatorul aplicatiei.
2. O functie care sa afiseze specializarea fiecarui doctor.
3. O functie care afiseze locurile disponibile pentru programari. Un utilizator va alege data la care doreste sa fie programata, iar in cazul in care data respectiva nu este disponibila, se vor afisa datile la diferenta de +/-2 zile de la data aleasa, care mai sunt disponibile.
4. O functie care sa asigneze unui doctor unul sau mai multi pacienti.
5. O functie care sa afiseze unui doctor programarile ce le are intr-o anumita zi.
6. O functie care sa-i afiseze fiecarui stapan animalele ce-i si au fost cel putin o data la cabinetul nostrum veterinar.
7. O functie care sa afiseze pentru fiecare pacient tratamentele ce i-au fost acordat.
8. O functie care sa afiseze doctorii dupa nume si prenume.
9. O functie care sa afiseze doctorii dupa nume.
10. O functie care sa afiseze doctorilor orele disponibile pentru programari intr-o anumita zi.
11. O functie care sa afiseze numarul de telefon al unui doctor sau stapan.
12. O functie care sa afiseze numele pacientului dupa id.
13. O functie care sa afiseze programarea unui pacient dupa numele acestuia.