Arhitectura Aplicatiei

-Lost Pets-

**1.Detalii aplicatie-Flows**

Aplicatia pune la dispozitia clientilor un serviciu de gasire a animalutului pierdut.

Utilizatorii pot adauga anunturi cu animale pierdute/gasite, isi pot crea un cont pentru a primi mesaje, notificari si de asemenea, au acces la diferite servicii cum ar fi crearea de pliante pentru animale disparute, alerte, locatia unde a fost ultima data vazut un animal, precum si descarcarea de statistici privind numarul de cazuri de animale disparute intr-o anumita perioada de timp.

Pentru a putea adauga un anunt, pentru a putea primi notificari si a fi la curent cu toate noutatile, utilizatorul va trebui sa-si creeze un cont in aplicatie.

La momentul inregistrarii, aplicatia va interoga baza de date pentru a vedea daca nu mai exista un utilizator cu acelasi email.Utilizatorul va completa un formular de inregistrare, acesta va trebui sa furnizeze informatii precum numele si prenumele, o parola (care va fi hash-uita) si toate aceste informatii vor fi salvate in database.

Fiecare user va fi notificat printr-un email atunci cand cineva adauga un nou anunt cu un animal ce se afla in aceeasi zona cu userul respectiv.

De asemenea, un user va avea acces zilnic la articole de specialitate .

**2.Database**

Pentru detaliile userilor vom folosi o baza de date SQL, numita SeekDB. Aceasta va contine un tabel cu username-ul fiecarui user, adresa de email, parola, un id unic pentru fiecare user si adresa unde acesta locuieste pentru a putea fi notificat usor.

Pentru detaliile legate de anunturi vom avea doua tabele, tabelul pentru animalele pierdute, si tabelul pentru animalele gasite.

**3.Autentificare**

Dupa inregistrare, user ul se poate loga in cont folosind email-ul ales si parola, si are acces la formularele de gasit/pierdut animal. Parola va fi hash-uita si server-ul va face un query in baza de date cu email-ul introdus. Daca mail-ul este gasit si hash-urile fac match, atunci user-ul va fi logat cu success.

**4.Persistenta utilizatorului logat**

Ne-am gandit la doua metode pentru a mentine utilizatorul logat o anumta perioada de timp, fara ca acesta sa fie nevoit sa se logheze pe site.

* Server-ul tine evidenta IP-urilor logate recent, daca IP-ul utilizatorului este in acea evidenta, utilizatorul va fi logat automat. IP-urile din aceasta evidenta vor avea asociate un timestamp al ultimei accesari a user-ului. IP-urile sunt sterse din lista cand “perioada” lor expira, utilizatorul fiind nevoit sa se relogheze cu credentialele sale.
* Site-ul creeaza un fisier mic pe masina utilizatorului ce joaca rolul unui fel de cookie (poate fi folosit si pentru alte functionalitati nu doar persistenta, ramane de vazut la implementare).Numele fisierului va fi generat in functie de email-ul utilizatorului(nu e batut in cuie/subject to change). Similar primei metode, “cookie”-ul va tine inauntru un timestamp. Cand utilizatorul se logheaza, site-ul va verifica existenta fisierului respectiv, daca exista, va interoga informatiile din el. Daca fisierul “expira”, site-ul va sterge fisierul si utilizatorul va fi nevoit sa se logheze, facand astfel un fisier nou.
* JWT

Primele 2 metode au avantaje cat si dezavantaje.

Prima metoda nu presupune crearea unui fisier pe disk-ul utilizatorului, fisier care poate fi interceptat (inca nu am gasit o solutie pentru problema asta de securitate dar sigur sunt) dar nu este scalabila pentru multi utilizatori, e costisitoare din punct de vedere al performantei pentru foarte multi utilizatori.

A doua metoda nu afecteaza performanta si/sau experienta utilizatorului dar poate fi un punct vulnerabil din punct de vedere al securitatii.

Metodele prezentate nu sunt finale, poate ne gandim la alta sau reusim sa facem o combinatie dintre primele doua prezentate pentru a obtine un rezultat care pastreaza avantajele metodelor, sau JWT.

**5.API-uri**

Ca api extern, vom folosi api-ul de la OpenStreetMap pentru a oferi informatii cat mai precise utilizatorului in ceea ce priveste zona in care a fost pierdut animalul de companie.

**6.MVC**

Am ales sa folosim design pattern-ul MVC pentru site-ul nostru datorita popularitatii sale in cadrul aplicatiilor web.

*1.Views*

Vom avea views pentru paginile deja existente (login,register,homepage,etc) cat si pentru viitoare pagini. Aceste views vor primi date direct din controllers

*2.Controllers*

Vom avea un controller care face update view-ului cu lost and found pets, acesta va folosi modelul care ia din baza de date informatiile aferente si le va trimite view-ului

Vom avea un controller care va interactiona cu utilizatorul, acesta afiseaza diferite mesaje (logare esuata/reusita, etc) folosind atat baza de date cu users cat si view-urile aferente.

*3.Models*

Vom avea cate un model pentru fiecare entitate din SeekDB care va interactiona cu baza de date corespunzatoare si va lua/adauga informatiile necesare atunci cand sunt cerute.