***Descrierea arhitecturii aplicației PerF***

-Tehnologii Web-

#1 **Flow-ul aplicatiei (prin diagrame de use-case)**

a. Utilizatorul intra in aplicatie si se poate inregistra

b. Utilizatorul intra in aplicatie si se poate loga

1. Poate vizualiza o lista cu anunturi

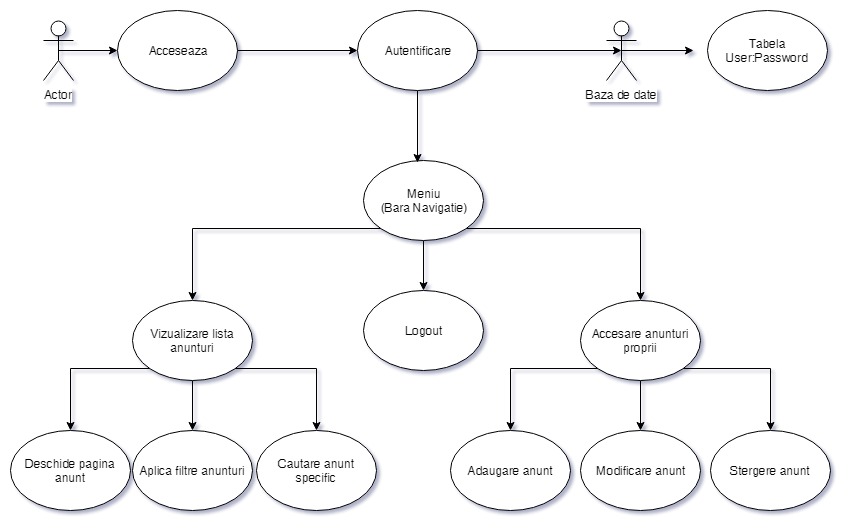
i. Poate deschide pagina unui anunt

ii. Poate modifica filtrele de afisare

iii. Poate cauta un anunt specific

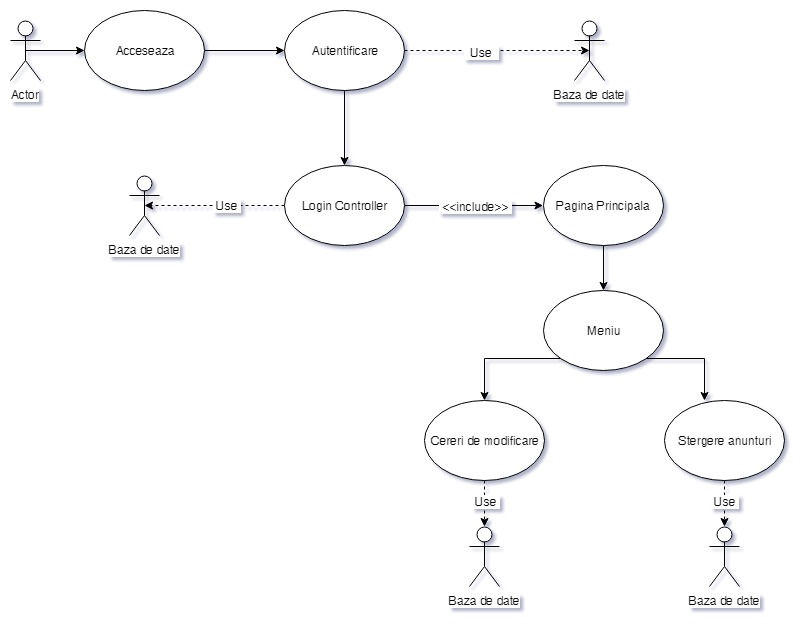
2. Poate adauga un anunt

3. Poate sa dea logout

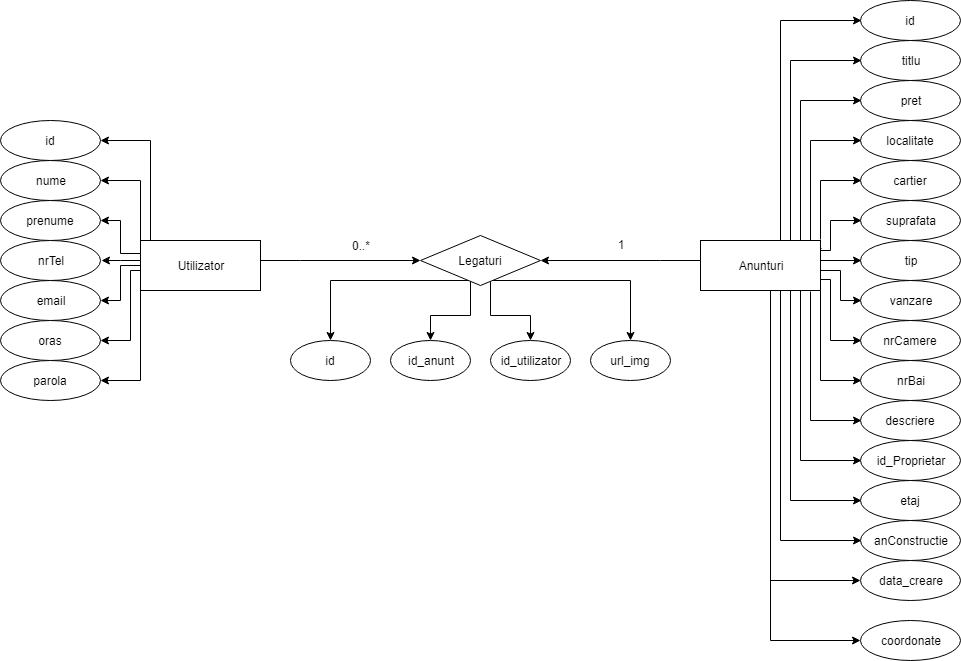


c. Administratorul poate sa vizualizeze cererile de adaugare anunt

d. Administratorul poate sa stearga/modifice un anunt



#2 **Baza de date**

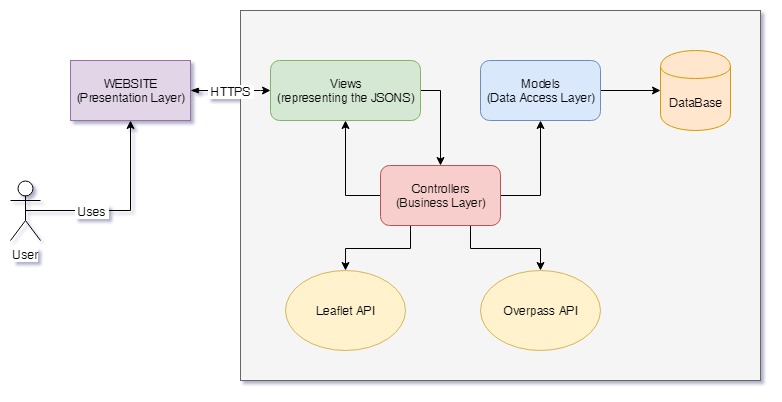


* Aplicatia va stoca și accesa datele cu ajutorul unei singure baze de date SQL;
* MySQL este una dintre cele mai populare baze de date destinate aplicațiilor web.
* Motivele pentru care am ales să folosim baza de date MySql sunt:
  + Familiaritatea echipei cu limbajul;
  + Oferă o mulțime de funcționalități chiar și pentru un motor de bază de date gratuit;
  + Există o varietate de interfețe de utilizator care pot fi implementate;
  + Acesta poate fi făcut pentru a lucra cu alte baze de date, inclusiv DB2 și Oracle;
* Informațiile aferente bazei de bate vor fi :
  + Cele legate de client, necesare înregistrării în aplicație (nume, prenume, email, parolă), dar și după înregistrare, pentru publicarea informațiilor suplimentare ale acestuia ( ex. date de contact);
  + Detaliile referitoare la un anunț pe care un client le poate introduce. Necesitate: clientul oferă amănunte în vederea unui anunț cât mai complex; aplicația folosește datele pentru a oferi clientilor posibilitatea de a sorta anunțurile căutate de aceștia.

#3 **Arhitectura aplicatiei**

Consideram ca pentru aplicatia pe care ne-am propus sa o realizam se preteaza patternul de MVC pentru a structura cat mai bine proiectul nostru. Astfel, pentru fiecare entitate din baza de date vom avea un model prin care sa interactionam cu datele fizice, cat si controllere, pentru a grupa functionalitatile comune si pentru a interactiona cu api-urile externe.

View-urile în cazul de fata sunt efectiv JSONurile care sunt trimise via HTTP ca raspunsuri la website-ul utilizat de un client uman.



Detaliem aici si controllerele pe care le vom utiliza in creearea logicii:

* LoginController (acceseaza modelul pentru Users)
* AnuntController (acceseaza modelul pentru Atunturi)
* RequestsController (acceceaza modelul pentru requests)

#4  **Autentificare**

* Pentru a persista utilizatorii logați, aplicația va folosi sesiuni PHP;
* O sesiune reprezintă o modalitate de a stoca informațiile unui singur client, pentru a fi folosite în paginile multiple ale unei aplicații;
* Variabilele session vor fi utilizabile, până când utilizatorul va închide browserul;
* Din punct de vedere al securității :
  + Sesiunea va expira după 30 de minute de inactivitate;
  + Funcția de logout explicit va expira, va distruge sesiunea, precum și cookie-urile sesiunii;
  + Se vor folosi exclusiv cookie-uri pentru a cripta ID-ul sesiunii;
  + Sesiunile care dureaza mai mult de o oră, vor expira și se va forța reautentificarea;

#5 **API-uri externe**

* Pentru implementarea hartii si a functionalitatilor acesteia am ales sa folosim 2 api-uri, leaflet si overpass.
* Leaflet este o librarie javascript, open-source, ce ofera posibilitatea de a creea harti interactive si personalizate.
* Am ales aceasta librarie datorita faptului ca este mobile-friendly, nu ocupa mult spatiu si este foarte flexibila, nedepinzand de o harta specifica atata timp cat aceasta respecta structura de tiles.
* Noi am ales sa folosim Open Street Map ca harta de baza pentru Leaflet.
* Am ales Overpass API deoarece aceasta se potriveste foarte bine cu nevoile aplicatiei. Aceasta e optimizata pentru query-uri rapide in baza de date OSM permitand accesul la o multitudine de date solicitate de useri.
* Se comporta ca o baza de date pe web, clientul trimite query-uri aplicatiei si aceasta acceseaza baza de date osm si trimite datele corespunzatoare inapoi.