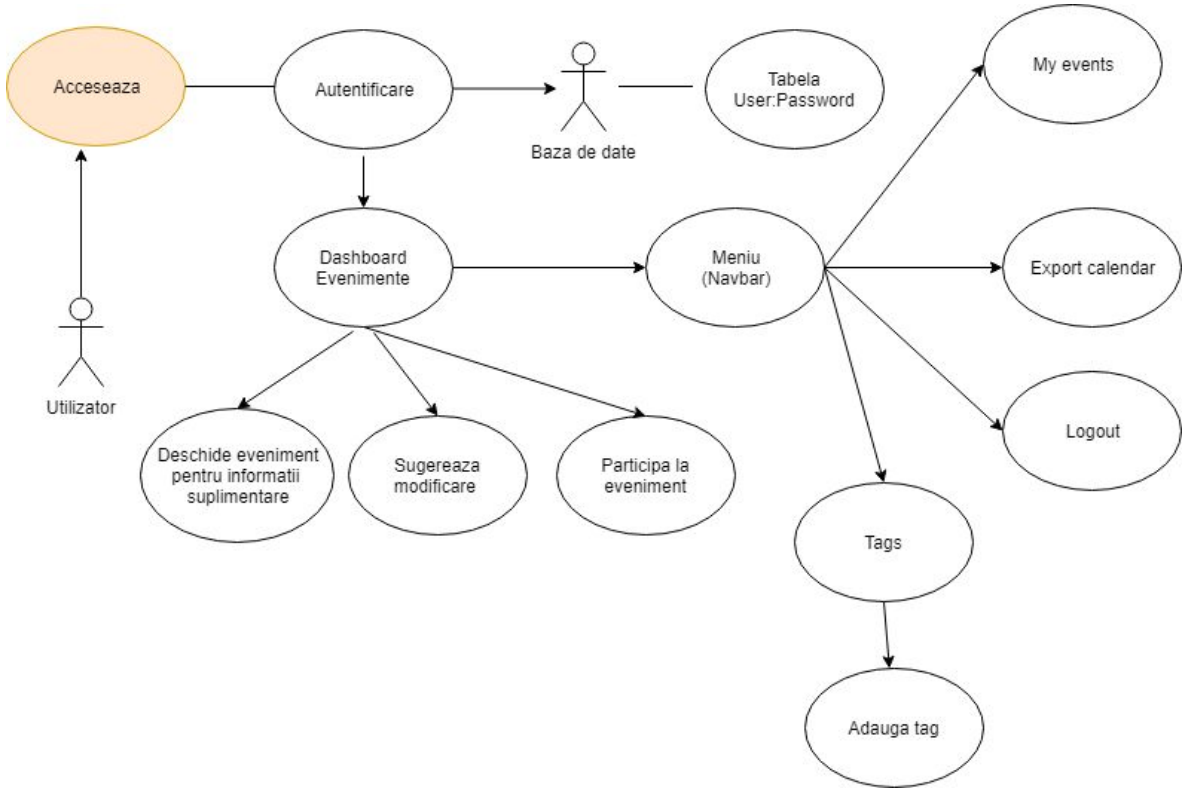
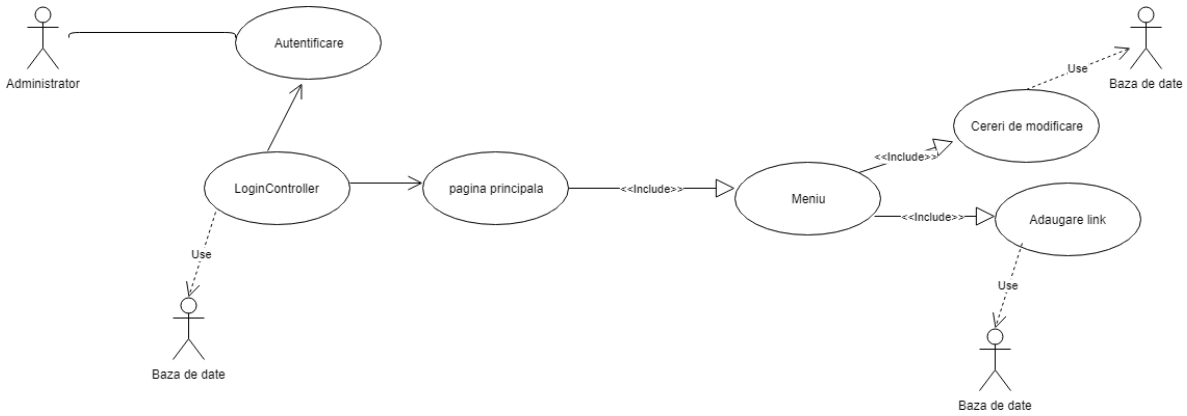
Autonomus 2.0

1. Flow-ul aplicatiei (prin diagrame de use-case)
    a. Utilizatorul intra în aplicatie si se poate inregistra
    b. Utilizatorul intra în aplicatie si se poate loga
        i. Poate urmari un feed cu evenimente
            1. Poate deschide pagina unui eveniment pentru a vedea mai multe informatii
            2. Poate deschide pagina si edita informatiile
        Ii. Poate selecta ce tipuri de evenimente sa urmareasca
        Iii. Poate vizualiza la ce evenimente a dat “going”
        Iii. Poate exporta evenimentele la care a dat “going” în ICalendar
        Iii. Poate sa dea logout

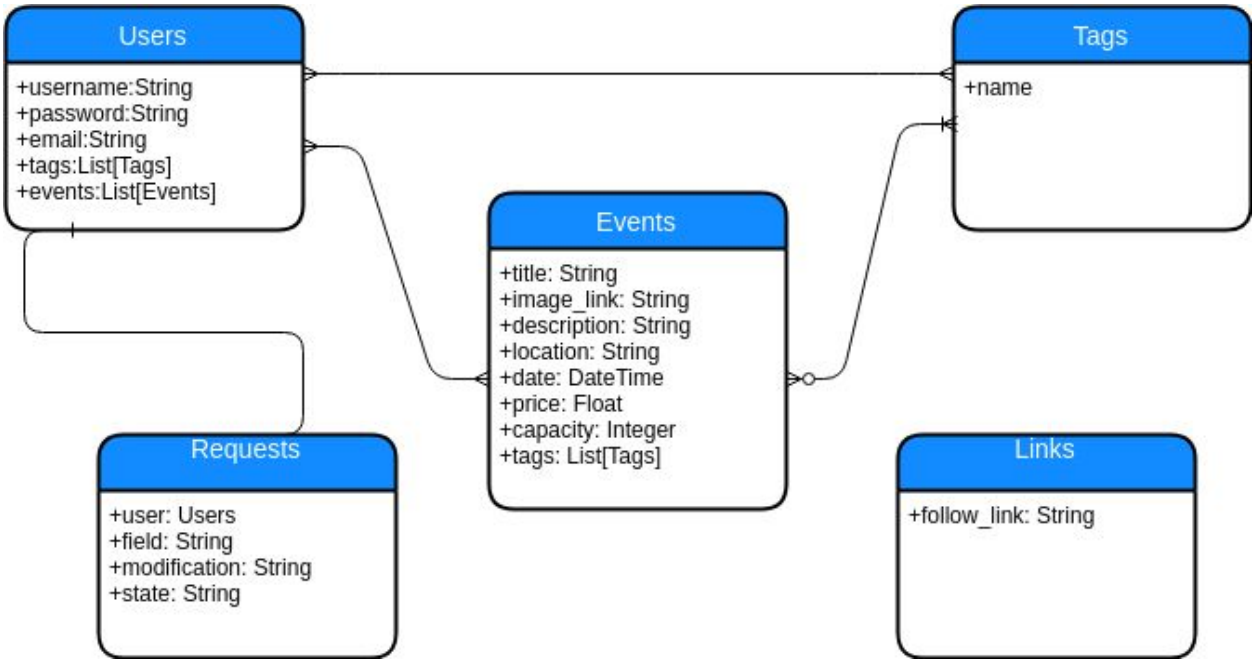


c. Adminul poate adauga link la o pagina de facebook sau grup de meetup a caror evenimente vor fi “urmarite”

d. Poate vizualiza cererile de modificare a evenimentelor si le poate aproba/respinge



2. Baza de date



Am considerat de la inceput ca utilizarea unei solutii de baze de date locale iese din discutie avand în vedere problemele de “sharuire” a acesteia si riscul crescut de a pierde baza de date prin lipsa replicarii.

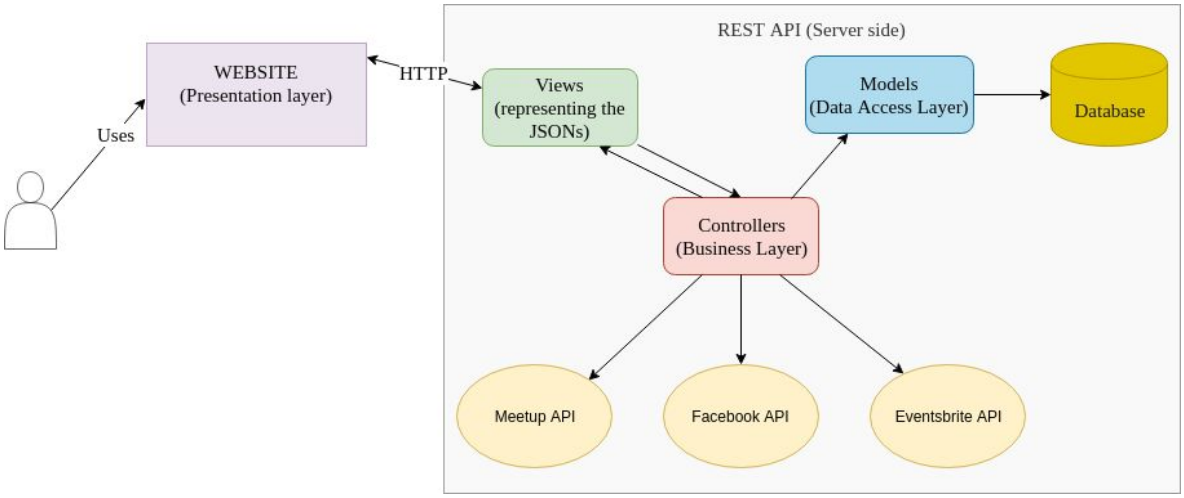
Am optat pentru o varianta în cloud a bazei de date - Google Cloud - care a fost prima optiune datorita usurintei de utilizare si lipsa costurilor initiale (comparat cu AWS). Aici am considerat doua alternative, baza de date relationala si nerelationala: Google Cloud Datastore si Google Cloud SQL.

Cea nerelationala ne ofera foarte multa flexibilitate deoarece nu trebuie sa definim schema si crea tabelele apriori operatiilor, permitandu-ne sa definim de fapt schema ad hoc, aspect destul de util avand în vedere contextul aplicatiei si partea de crawling unde ne putem trezi cu ‘surprize’ (campuri noi, lipsa unor campuri pe care ni le-am propus,

formate diferite). Un alt avantaj major al datastore e ca se ocupa singur de partea de scalabilitate, în timp ce varianta SQL are la baza tot mySql si devine responsabilitatea ta scalarea. De asemenea, Datastore suporta orice ne-am dori din partea SQL - interogarii în format SQL, tranzactii ACID, chei straine, indecsi si poate fi utilizat via un RESTful API, decupland partea de logica de model. Ulterior, datorita crawlerii a multor surse de evenimente putem ajunge în punctul în care am dori sa paralelizam procesarea evenimentelor si datastore-ul suporta integrarea cu Google Cloud DataFlow ce ne permite scrierea unor map-reducere pentru procesarea paralela a unor batch-uri de date.

3. Arhitectura aplicatiei

Consideram ca pentru aplicatia pe care ne-am propus sa o realizam se preteaza patternul de MVC pentru realizarea REST api-ului. 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. Pentru intreaga aplicatie web pe care noi o realizam, tinem cont si de separarea pe layere a scopului pt o mai buna decuplare la nivel de aplicatie (N-layer arhitecture), separand managementul datelor de logica aplicatiei si de view-ul final pe care il utilizeaza clientul.



Detaliem aici si controllerele pe care le vom utiliza în creearea logicii:
LoginController (acceseaza modelul pt Users)
EventsController (acceseaza modelul pt Events)
TagsController (acceseaza modelul pt Tags)
LinksController (acceseaza modelul pt Links)
RequestsController (acceseaza modelul pt requests)
De asemenea, pentru a realiza exportul de ICalendaro vom utiliza icalendar din Python, cu controllerul aferent CalendarController.

Pentru a obtine informatie actualizata cu evenimente pentru utilizatorii aplicatiei, vom folosi Google Scheduler si vom programa niste cronjoburi ce vor actualiza datele din baza de date.

4. Autentificare token

În vederea autentificarii, vom utiliza JSON Web Toolkit pentru a asigura nivelul de incredere necesar efectuarii anumitor operatii intre client si API.

5. API Extern

Cu scopul de a obtine evenimentele din diferite platforme pentru a le servi utilizatorilor nostri, vom apela la trei API-uri externe: Eventsbrite API, Meetup API si Facebook API.

Daca în cazul primelor doua lucrurile sunt destul de clare (endpoint Eventsbrite, endpoint Meetup), Facebook a închis dupa scandalul Analytica endpointurile pentru events, facand imposibil utilizarea API-ului în acest scop, decat de persoane autorizate (faci o cerere cu aplicatia ta si este revizuita intr-o durata considerabila de timp). Vom incerca sa obtinem un token de access insa alternativa ramane sa utilizam varianta de mobile a websiteului care nu foloseste AJAX si poate fi crawlata. Deoarece Facebook e foarte bun la a detecta boti, vom dezactiva JavaScriptul pentru a nu trimite activitatea noastra ca utilizator si Selenium în incercarea de a mima activitatea unui utilizator Facebook care cauta evenimente. Pentru crawling putem folosi Beautiful Soup si Scrapy.