1. **Introducere**
2. **Descrierea contextului aplicatiei**

* Scurt istoric al aplicatiilor web
* Arhitectura
* Componente generale ale unei arhitecturi pentru aplicatiile web
* Concluzii

1. **Dezvoltarea de aplicatii web pentru diferite motoare de cautare(3+4)**

* Limitarile care intervin folosind diferite frameworkuri
* Caracteristicile frameworkului Angular 5

1. **Diferite metode de a stoca informatiile**

* Scurt istoric al bazelor de date
* Mysql si conectarea la baza de date + oracle/SQL
* Concluzii

1. **Servere si diferite metode de prelucrarea a datelor**

* Scurt Istoric
* Diferite Frameworkuri
* Createa unui server
* Comunicarea dintre un server si partea de Front End
* Concluzii

1. **Recommendation System**

* Scurt Istoric
* Descriere + Python
* Collaborative Filtering
* Concluzii

1. **Descrierea Aplicatiei**

* Motive pentru alegerea temei
* Prezentarea conceptului aplicatiei
* Descrierea componentelor aplicatiei
* Tutorialul aplicatiei
* Utilizarea Aplicatiei

1. **Concluzie**
2. **Bibliografie**

**CAMBRIDGE SPARK.**

**Harvard**

**Medium**

* **The adaptive web (**Peter Brusilovsky
* Alfred Kobsa
* Wolfgang Nejdl

**)**

**Introducere**

Tema principala a acestei lucrari este un system de recomandare facut in python pentru un website care are o parte de front-end construita in AnuglarJs 5 si o baza de date in mysql si cu un backend facut in Nodejs .

Mai intai voi spune cateva cuvinde despre ce reprezinta un system recommendation si abordarea pe care acest proiect o va avea pentru aceasta tema.

In zilele de azi systemele de recomandare sunt create pentru a personaliza experienta ta pe internet respectiv site-urile pe care utilizatorii le aceseaza, spunadu-le acestora ce sa cumpere, ce sa manace, sau chiar cu cine ar trebui sa te imprietenesti. Oamenii tind sa placa lucruri care sunt similare cu lucrurile pe care le plac, si tind sa aibe aceleasi gusturi cu persoanele apropiate. Sistemele de recomandare tind ca inglobeze toate aceste modele ca sa prezica ce lucru iti place sau ce obiect ai vrea sa cumperi. E-commerce, social media, video si stirile de pe internet au abordat deja aceast lucru si au construit deja propriile lor sisteme de recomandare care au fost concentrate sa ajute clientii sa aleaga mai eficient produsele, care este o stratedigie in care castiga ambele parti.

Cele mai importante tipuri de sisteme de recomandare sunt :

* **Content-Based**
* **Collaborative Filtering**

**Collaborative filtering**

Collaborative filtering pentru produse este bazata pe cunosterea userilor si atitudinea acestora fata de acestea, care mai poarta si numele de intelepciunea multimii de a recomanda produse. In contrast Content-Based se bazeaza pe atributele produselor sau a obiectelor comercializate si similitudinile dintre ele.

In general, collaborative filtering sau CF este statia pentru sistemele de recomandare. Algoritmul are niste proprietati foarte interesante prin care este capabil sa invete de unul singur, ceea ce inseamna ca poate invata singur prorietatile sau atributele pe care sa le foloseasca.

Collaborative filtering este impartita in 2 mari mari abordari:

* **Memory-Based Collaborative Filtering**
* **Model-Based Collaborative filtering**

**Memory-Based Collaborative Filtering**

In acest caz se utilizeaza raitingul user-ului pentru ca algoritmul sa computeze similitudinile dintre user sau produs/item.

Aceasta abordare este folosita in general pentru partea de marketing. Ea a fost folosita la inceput de toate site-urile cu activitatea principala e-commerce. Este eficient si usor de implementat. Cele mai uzuale abordari sunt neighbourhood-based collaborative filtering si item-based/user-based top N recommendation, pe scurt se creaza o matice de M pe N care are pe coloana userii care folosesc acest site si pe linie produsele care se regasesc pe aceasta aplicatie.

Ca exemplu in abordarea user-based, raitingul pe care userul o da produsului i se calculeaza ca o agregare a unor asemanari dintre useri si raitingul lor pentru produse.

|  |  |  |  |
| --- | --- | --- | --- |
| Useri\Produse | P1 | P2 | P3 |
| U1 | 4 |  | 5 |
| U2 | 5 | 5 | 4 |
| U3 | 1 |  | 1 |

Din urmatorul tabel se observa clar ca user-ul u1 pare sa aibe gusturi similare cu user-ul u2 pe cand user-ul u3 pare sa aibe gusturi diferite de ceilalti 2. In concluzie user-ul u1 si user-ul u2 se numesc vecini sau neighbours, de aici si numele acestei abordari.

Pentru a calcula coeficientii si celelalte atributii ne stau la dispozitie formule si algoritmi specifici pe care ii voi prezenta in capitolele urmatoare.

**Model-Based Collaborative Filtering**

In aceasta abordare modelele sunt create sau construite utilizand diferite data-mining,algoritmi folositi in machine learning pentru a prezice raitingul utilizatorului pentru produsele care inca nu au fost evaluate.

Sunt foarte multi algoritmi model-based, amintesc cativa dintre ei si anume: Retea Bayesiana, Modele Clustering, modele latent semantic precum valori singurale de decompozitie.

Pentru aceasta abordare, metodele de reducere a dimensiunilor sunt folosite ca tehnici complementare pentru a imbunatati robustetea si acutitatea pentru abordarea model-based.

In acest sens, metode ca valori singulare de descompunere sau principiul de analiza a componentelor, cunoscute ca modele latent factor, se comprima intr-o matrice user-item intr-o dimensiune mica in termini de latent factor.

Un avantaj pentru folosirea acestei abordari este ca in schimb de o matrice de dimensiune foarte mare ce contine foarte multe valori lipsa, se va folosi o mult mai mica matrice intr-un spatiu dimensional mai mic.

**Content-Based**

Aceste sisteme de recomandare recomanda produse similare care au fost cumparate in trecut sau evaluate cu un scor mare pentru un anumit client sau user, neavand nicio legatura cu ce au evaluat sau cumparat alti useri.

Content-based recommendation systems analizarea atributele produselor pentru a identifica produse care sunt pe placul useri-lor. Pentru ca paritcularitatile sistemelor de recomandare difera bazate pe reprezentarea produselor sau obiectelor

In abordarea content-based, cuvintele cheie sunt pentru a descrie produsele si un profil de user este contruit pentru a indica ce tipuri de produse sunt placute/cumparate de user. In alte cuvinte, acesti algoritmi incearca sa recomande produse care sunt asemanatoare celor care au fost cumparate sau evaluate in trecut.

Pentru a abstractiza caracteristicile produselor in sistem, un algoritm de prezentare a produsului este aplicat. Un foarte utilizat algoritm este fregventa inversa a documentului mai numita si reprezentaria vectorial spatiala.

Pentru a crea un model de user sau un profil, sistemele se concentreaza pe doua tipuri de informatii:

* Un model care caracterizeaza preferintele utilizatorului
* Un model baza pe istoricul userului cu sistemul de recomandare

Practic aceste metode folosesc un profil de produse (un set discret de caracteristici si atribute) caracterizit acestea in sistem.

Sistemul creaza un profil de useri bazat pe niste vectori de ponderi pentru caracteristicile produselor. Aceste ponderi denota importanta fiecare dintre caracteristicile produselor pentru utilizator. Cele mai simple abordari folosesc mediar aritmetica a evaluarii produselor intr-un vector, pe cand alte metode mai sofisticate folosesc si tehnici de machine-learning precum Clasificatorul Bayesian, abore de decizie si asa mai departe.

O problema cheie a acestui sistem este daca acesta poate sa invete preferintele utilizatorului in materie de produse si actiunile acestuia cu privire la o singura sursa de continut si de a le utiliza in alte tipuri. Cand sistemul este limitat sa recomande acelasi tip de produs pe care userul l a placut, valoarea sistemului de recomandare scade vertiginos, in schimb daca userul primeste o recomandare care nu seamana neaparat cu produsele pe care acesta le a placut acest sistem de recoandare primeste o importanta aparte fiind mai complex decat cel prezentat anterior.

**Descrierea contextului aplicatiei**

Scurt istoric al aplicatiilor web

Prin web design se intelege crearea de site-uri web, de la momentul conceperii structurarii si interfetelor grafice pana la introducerea propriuzisa a datelor care alcatuiesc intr-un final continutul site-ului: imagini, elemente etc.

Primul designer web a fost Tim Berners Lee, inventatorul www-ului, care in anul 1991 a publicat primul site din istorie.

La inceput site-urile web aveau o complexitate mult mai scazuca ca acum, ele fiind realizate in mare parte in HTML, limbaj care nu este atat de puternic, permitand doar o serie limitata de formatari, precum inserarea de linkuri pentru a naviga de pe o pagina pe alta, iar designul web fiind mai mult un soi de programare web.

In present exista o preocupare ridicata pentru aspectul gafic al site-urilor, devenite puternice intrumente de comercializare, publicitate iar acest lucru justifica denumirea de web design.

Tehnologiile utilizate in prezent s-au diversificat foarte mult, in prezent fiind din ce in ce mai complexe, reamintim aici cateva limbaje: python cu frameworkul de web Django, php , javascript cu multitudinea de frameworkuri cele mai importante fiind (AngularJs, ReactJs , VueJs).

Companiile care au ca activitate principala productia de site-uri au angajati specializati pentru fiecare etapa din crearea si dezvoltarea unui site, de la stadiul de concept grafic la programare si editare de continut pentru motoarele de cautare adica SEO (search engine optimization).

Site-urile din prezent sunt din ce in ce mai mult axate pe dinamism si animatie interactiva, aceste necesitati fiind indeplinite de cunoscutul Adobe Flash, precum si alte programe aparute dupa acesta, care pot genera diferite fisiere animate in diferite formate care pot fi usor urcate pe site.

Site-urile sunt realizate si structurate astfel incat fiecare persoana fara cunostinta in domeniu sa poate sa le utilizeze cu usurinta , ele fiind afisate intr-un browser. Web designerii lucreaza in diferite limbaje din care apoi se genereaza intr-un limbaj specific al internetului HTML(HyperText Markup Language) sau direct in HTML.

Daca nu ar fi existat browserele specializate care sa stie sa interpreteze partea de script atunci site-urile ar fi niste blocuri de text fara imagini, fara prea multe puterea de interactiune din partea clientului.

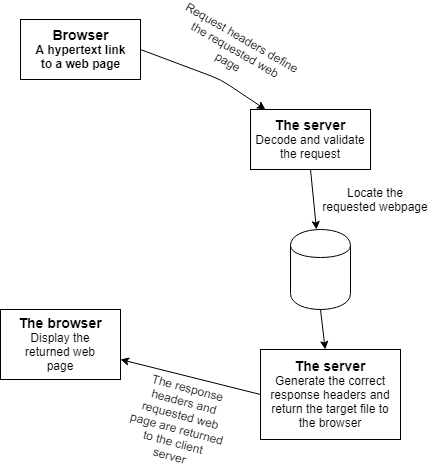
In principal limbajul HTML indica locul unde sa se afiseze eventualele elemente.

Componente generale ale unei arhitecturi pentru

aplicatii web

Arhitectura Web standard a fost proiectata astfel incat un calculator client poate interactiona cu un calculator server printr-o conectare printr-o retea TCP/IP.

Browser-ul si serverul comunica utilizand protocolul *Hypertext Transport Protocol* (HTTP). TCP/IP manevreaza detaliile de transport, asigurandu-se ca pachetele ajung la destinatie corect. HTTP structureaza comunicatia legata de fisierele ce se primesc si cum se interpreteaza ele. Aceasta permite software-ului Web sa fie focalizat asupra a ceea ce face mai bine: accesarea si afisarea continutului. Ca majoritatea proiectelor client/server, Web-ul utilizeaza o metoda cerere-raspuns, ceea ce inseamna ca serverul asteapta si asculta cererile clientului inainte de a raspunde la ele.



Browser-ul Web cere o anumita pagina web (URL – Uniform Resource Locator) de la server. Nu conteaza daca acel URL a fost introdus de utilizator, sau a fost asociat cu un hiperlink, un bookmark, sau cu un buton al browser-ului numit Home. Important este ca serverul de Web raspunde la cereri, de obicei trimise de un browser web sub controlul utilizatorului.

Software-ul serverului Web este instalat pe un calculator gazda potrivit. In functie de volumul cererilor pe care trebuie sa le manipuleze serverul se alege sistemul de operare si platforma harware pentru server. Odata instalate si configurate, majoritatea serverelor web doar asteapta si asculta daca exista cereri de la browser-ele web.

O data receptionata cererea, serverul intra in actiune. Serverul localizeaza URL-ul cerut si il copiaza pe conexiunea retea, unde TCP/IP il transporta la browser. Odata ce browser-ul primeste fisierul cerut (pagina web) de la server, browser-ul interpreteaza HTML-ul (si alte coduri incluse) si afiseaza fisierul (pagina web) corect (cu maximum capacitatilor sale) pe calculatorul client.

In general, web-ul a fost descris ca si cross-platforma dinamica, sistem informatic hipermedia distribuit ce ruleaza pe Internet. De fapt Web-ul este prima generatie a unei noi clase de sisteme distribuite ce continua sa evolueze rapid. Aceste sisteme au marea capacitate sa integreze sistemele informationale si bazele de date deja construite cu cele ce urmeaza sa fie construite.

Web-ul este usor de utilizat, iar HTML-ul este usor de scris. Totusi, Web-ul ca si platforma de dezvoltare sistem, este plin de provocari; aceasta intrucat dezvoltarea sistemelor presupune adesea sarcina dificila de integrare a oamenilor, a datelor, a proceselor, a retelelor cu tehnologii potrivite in vederea rezolvarii problemelor si crearii oportunitatilor. Dezvoltarea efectiva a aplicatiilor Web este creativa si implica o multime alti de factori pe linga cei legati de tehnologie.

Utilizarea Web-ului ca mediu de dezvoltare este relativ recenta ; datorita  mediului de dezvoltare, o aplicatie Web este in general formata din mai multe parti si sa foloseasca multe tehnologii. Cadrul de lucru pentru dezvoltarea Web constituie un exemplu de combinatie fericita intre o multime de tehnologii, fiecare limitata in domeniul sau de actiune. In ciuda acestor limitari, combinatia acestor tehnologii in aplicatii furnizeaza o modalitate extrem de eficienta pentru dezvoltarea de solutii pentru Internet si intranet.

Arhitectura

Arhitectura acestei aplicatii web este impartita in trei mari categorii si anume:

* Partea de server si api-urile aferente
* Baza de date
* Parte de Front cu

Serverul si api-urile aferente

Pentru aceasta aplicatie, am preferat sa utilizez NodeJs cu frameworkul express pentru a usura codul si pentru a avea o experienta mai placuta.

NodeJs este un limbaj de programare asyncron bazat pe limbajul de programare JavaScript. El este conceput pentru a constui aplicatii scalabile pentru partea de retele/servere.

Acest limbaj de programare este asemanator lui pyhton sau ruby, doar ca nodejs duce partea de event model mai departe si previne singur ciclarea evenimentelor ca parte de runtime .

Acest server de node mai are legaturi cu un api , pentru un chatbot care este atasat site-ului sau aplicatiei web. Numele acestui api este DialogFlow si este creat de google.

Acest api le permite user-ilor un nou fel de a interactiona cu produsele expuse prin intermediul aceste aplicatii. Acest produs este bazat pe interactiune pe cale orala sau prin text a clientului cu eventualul bot.

DialogFlow a fost conectat pe server prin cadrul unui webhook si primeste requesturi de la useri prin intermediul aplicatiei web, acesta la randul lui trimitand responsuri inapoi la aplicatie iar in final mesajul va fi afisat pe partea de front-end unde clientul va fi intampinat de un raspuns promt si precis.

Acest api este bazat pe intent-uri si are o masinarie de invatare pe server, fiecare fraza fiind analizata si interpretata. In functie de complexitatea botului creat, el poate sustine o discutie cu un eventual client creandui impresia de o adevarata conversatie.

!!!! Parte de siteme de recomandare api. Facut in python

Baza de date

Baza de date a fost creata in limbajul mysql care este un sistem de gesiune a bazalor de date si de relationare, produs de compania suedeza MySql. Este unul din cele mai populare SGBD-uri open-source la ora actuala.

Acest limbaj a fost creat in c++ si c. Parserul SQL a fost scris in yaac.

Are si parte de server si anume MySql server care contine librarii propri.

Aceasta are si o parte de vizualizare a bazei de date cu toate tabele aferente, unde proprietarul poate interactiona direct cu aceasta numita MySql workbench.

Aplicatia are la baza un numar de cinci tabela si anume:

* categorySubtype
* items
* itemsoferte
* raiting
* users

In subcapitolul de concluzii voi reveni cu o tabela uml in care voi explica relatiile dintre aceste tabele.

Partea de Front-end

Aceasta parte, fiind si cea mai generoasa, exercita posibilitatea de a vizualiza si interactiona cu toate produsele si atributele/optiunipe pe care aceasta aplicatie le ofera.

Pe scurt aplicatia este realizata in AngularJs 5 un framework popular de JavaScript, conceput de Google.

Acest framework este structurat pentru aplicatii web dinamice si permite utilizarea HTML-ului ca o parte din componenta creata. AngularJs creaza o legatura cu informatia si elimina injectia dependentelor usurand astfel metoda de codare si elimina cu aceasta ocazie si mult din textul codului.

AngularJs este bazat pe componete si le randeaza automat pe site fiind a single page aplication. Componentele sunt formate din patru fisiere putand fi generate automat cu ajutorul modulului angular-cli instalat prin node package module cu comanda „ng g c numecomponenta”.

Aceste folder care alcatuiesc corpul componente este formata din patru mari fisiere si care au fiecare in parte partea lui in conceptia designul aranjarea functionalitatea si testarea componetei:

* componenta.ts
* componenta.html
* componenta.css
* componenta.spec.ts

Componeta.ts are ca rol partea de functionalitate a componentei respective ea fiind gestionata prin intermediul limbajului typescript care este ulterior compilat si transformat in cod de javascrip. Limbajul typescript este dupa parerea mea mult mai puternic decat javascript avand avantaje esentiale : crearea de clase, interfete, multe module si o parte in care se pot predefinii tipul de date alea variabilelor dupa cum ii recomanda si numele.

Componenta.html are ca rol partea de pozitionare a elementelor cat si createa de legaturi cu componeta.ts. Index.html este componenta parinte care randeaza pe rand toate partile de html aferente componentelor in parte.

Componenta.css are ca rol stilizarea partii de html a acesteia, aceasta partea are un rol esential in vizualizarea componetelor ea fiind singura posibilitatea de silizare a elementelor. Aceasta componentea este legata strict de partea de html a acesteia neavand posibilitati de aplicare globala pentru toate componentele.

Pentru a putea stiliza global aveam o componenta globala si anume app.component.css unde putem accesa toate elementele din componentele aplicatiei.

Componenta.spec.ts are rolul de a testa acestea in modul unit testing pentru fisierul sursa. Acestea ruleaza in Jasmine javascript test framework prin intermediul lui Karma care ruleaza sarcinile cand testerul introduce in linia de comanda „ng test”.

Concluzii

In concluzii voi prezenta diagrama uml a aplicatiei elaborand si explicand pe larg legaturile dintre componentele esentiale alea aplicatiei.

Prima data voi face referinta la limbajele utilizate facand referinta la alternativele pe care un programator (web designer) le poate avea :

Pentru partea de backend/server-side, am ales nodejs pentru ca foloseste limbajul javascrip, fiind un limbaj usor de manevrat care a avut o ascensiune in ultimii ani fiind clasat pe pe trei intr-un clasament facut de github sub python si java.

Alegerea mea pentru partea de stocare de date si manipulare a fost mysql care are o interfata primitoare desi mongo are o mai usoara structura, nu foarte complexa bazate pe colectii care foloseste un limbaj noSql, am vrut sa raman la vechiul sql pentru ca foarte multe baza de date sunt gestionate si au nevoie in continuare de support pe acest limbaj in detrimentul lui mongoDb care este un produs recent scos pe piata.

Pentru partea de front-end, am ramas tot la promitatorul javascript, alegand frameworkul AngularJs.

Opinii pro si contra au existat tot timpul, multi din bransa web-designe-rilor au trecut rand pe rand la frameworkul Reactjs dezvoltat de Facebook, din cauza evolutiei si deprecierilor permanente pe care AngularJs le a avut. El avand actual 6 versiuni, si o schimbare radicala de arhitectura de la angularjs la angularjs 2.

In ultimult timp React a castigat teren si pe partea de joburi, oferind multe oportunitati proaspetilor candidati.

In schimb AngularJs ramane in top si promite in continuare, iar cei de la google pun din ce in ce mai mult accent pe aceasta parte de dezvoltare.

Cel mai proaspat framework VueJs vine cu o noua abordare, o arhitectura interesanta, fiind usor de manevrat, fiind recomandat pentru aplicatii mici deoarece arhitectura acestuia nu permite descompunerea in fisiere a celor 4 elemente esentiale, acestea fiind toate adunate in componeta acestula „numecomponeta.vue” avand o parte de template unde este randat html-ul o parte de script unde este introdusa partea de functionalitate si la sfarsit partea de stilizare in care este introdus codul de css.

In final voi prezenta arhitectura aplicatiei, baza de date, partea de web cat si apiurile care fac legatura cu acesta prin intermediul serverului.