**Documentatia Proiectului**

-animeList-

***Prezentare proiect***

Ati avut vreodata nevoie de informatii despre un *anime* si nu ati gasit tot ce doreati dupa primul *search*? Sau probabil ati gasit informatii care nu portretizau destul de bine lumea unui *anime* pe care l-ati vizionat? Credeti ca anumite *anime*-uri merita sa ajunga mult mai populare decat sunt, dar nu aveti idee cum sa ajungeti la acest rezultat? Proiectul prezentat in aceasta documentatie este raspunsul la aceste probleme si multe altele din acelasi spectru.

Comunitatea este o parte importanta din viata animalelor de pe acest pamant. Totusi, spre deosebire de alte animale, omul a ajuns sa faca parte dintr-o comunitate mult mai complexa. Am ajuns sa avem comunitati micute, de exemplu cartierul in care ne-am nascut, dar si comunitati la o scara mai mare, cum ar fi comunitatea oamenilor care se uita la *anime*-uri, *target*-ul acestul proiect. Aceasta ultima comunitate se intinde pe suprafete foarte mari, doi membrii putand sa apartina diferitor tari si chiar continente. Ce a facilitat aceasta dezvoltare de comunitati la scara larga? Dezvoltarea tehnologiei. Oamenii au ajuns sa poata comunica intre ei din oricare colt al lumii, se pot ajuta unii pe altii desi sunt la o distanta de mii de kilometrii departare. Pe aceasta idee a fost construit proiectul ce urmeaza sa fie prezentat in paginile de mai jos.

AnimeList propune utilizatorului o aplicatie cu continut accesibil pentru cei ce isi fac un cont*.* *User*-ul va putea sa acceseze informatii legate de *anime*-uri si chiar sa editeze partile care sunt incorect scrise de utilizatorul precedent. Dupa crearea unui cont, utilizatorul va avea acces complet la continutul platformei. Adaugarea de *anime*-uri este la doar un buton distanta. Utilizatorul ofera datele necesare, specificate pe site, iar noi de ocupam de restul (crearea paginii *anime*-ului, asezarea in pagina a elementelor etc). Exista pagina principala unde se pot vedea toate *anime*-urile disponibile pe site, cu mai multe optiuni de sortari (dupa rank, alfabetic etc). Pe aceasta pagina exista si un *search bar* care permite cautarea dupa nume a *anime*-ului si accesarea paginii acestuia. In coltul din dreapta sus se afla butonul care duce la pagina de ‘’*About’’* a site-ului. Aceasta ofera informatii legate de autorii site-ului, dar si moduri de contact, in cazul plangerilor.

O parte foarte importanta a *site*-ului este controlul complet pe care il are utilizatorul, acesta avand acces si la conturile celorlalti utiliaztori. S-a ales aceasta abordare pentru a permite independeta completa a programului de programatorii care l-au creat si pentru a observa evolutia acestuia, tot proiectul reprezentand un experiment social foarte interesant. Vor reusi utilizatorii sa ajunga la un compromis legat de datele prezente pe *site*? Bineinteles, initial multe conturi vor fi sterse din pura invidie sau chair ura, dar sigur se v-a ajunge la un acord comun in cele din urma.

***Tehnologii folosite***

Stilizarea proiectului s-a facut cu ajutorul *Bulma*. Acesta este un *open source CSS framework* gratuit care ofera componente *frontend* prestabilite care sunt usor integrabile direct in interfata web dezvoltata de programator.

De ce Bulma?

Unul din motivele evidente pentru care am ales sa folosim Bulma este accesibilitatea acesteia. Descarcarea si instalarea *framework*-ului este intuitiva, avand multe tutoriale pe *site*-ul lui. Bulma se instaleaza printr-o simpla comanda in consola si nu necesita fisiere sau librarii aditionale.

Un alt motiv pentru alegerea acestui framework a fost complexitatea si estetica placuta a diferitelor *template*-uri disponibile. De la simple variabile de stilizare pana la elemente de UI, *search* *bar*-uri, *checkbox*-uri, *input box*-uri, cotainere de imagini, *Bulma* ofera un *framework* usor de folosit, cu numeroase optiuni pentru construirea paginilor web. Un exemplu din proiectul prezentat este meniul de sortare prezent pe pagina principala a *site-*ului. In implementarea lui s-a folosit un *drop down menu* pentru o organizare a paginii similara cu majoritatea *site*-urilor deja existente, in vederea facilitarii intelegerii *site*-ului de catre utilizator.

Pentru cei interesati de Bulma, care doresc sa aprofundeze documentatia oficiala folosita in crearea proiectului, link-ul cautat este: <https://bulma.io/documentation/>

***“Backend”***

Deoarece am dorit sa facem un site care sa contina o multitudine de informatii despre anime-uri, am avut nevoie de o baza de date unde sa stocam toate aceste informatii. Toate fake API-urile pe care le-am gasit fie trebuiau platite fie nu mai erau active, asa ca am decis sa implementam un API folosind un fisier de tip .JSON si sa il hostam local. Acesta este alcatuit din 2 sectiuni: *Anime*, care pastreaza toate *anime*-urile si datele acestora si *Accounts*, care tine evidenta conturilor utilizatorilor. Pentru fiecare au fost implementate operatiile CRUD necesare.

La deschiderea *site*-ului, un *user* este prezentat cu 2 optiuni, logare sau creare de cont. La partea de cont nou au fost implementati validatori pentru a verifica *input*-urile utilizatorului. Daca toate *input*-urile trec de validatori, un nou cont este creat si adaugat in API. Partea de logare implica cautarea *username*-ului si parolei introduse de *user*, in API. In cazul in care acestea sunt gasite, *user*-ul este trimis la pagina principala a *site*-ului.

In pagina principala *user*-ul are optiunea de a edita sau sterge anumite informatii legate de *anime*-uri, dar acesta poate si adauga un *anime* complet nou. Exact aceleasi optiuni sunt prezentate si cand vine vorba de conturi. Utilizatorul isi poate modifica datele contului sau chiar sterge un cont, cu conditia sa fie introdus *username*-ul acestuia. Pentru implementarea acestora s-au utilizat *request*-uirle: GET, pentru partea de *read*, POST, pentru partea de *create*, PUT, pentru partea de *update* si DELETE, pentru *delete*.

Tot pe pagina principala este prezent un tabel cu toate *anime*-urile *site*-ului. In implementarea acestuia s-a folosit *mat-table*, componenta furnizata de *site*-ul *Material Angular*. Tabelul foloseste un serviciu care ii permite sa ia datele din API.

***Arhitectura proiectului***

Structura poriectului este una simpla si intuitiva, aceasta fiind compusa din trei module principale: *Home-mod, Search-mod* si *Manage-mod*. Pe langa aceste trei module s-a folosit o componenta *header,* una *footer* si o componenta *support*. Cele din urma sunt folosite ca si compoenente deoarece *header*-ul si *footer*-ul apar pe fiecare pagina web, iar pagina de *support* este una simpla, care nu necesita mult timp de incarcare si care este accesibila de pe orice pagina a *site*-ului.

Motivul utilizarii acestei structure pe 3 module a fost evitarea incarcarii nefolositoare a paginilor pentru utilizator, astfel realizand un *user-experice* placut si lipsit de timpi de incarcare colosali.

Modulul *Home-mod* cuprinde componentele *log in, register* si validatorii acestora. Ele au fost puse intr-un modul deoarece sunt similare ca structura, iar demersul unui utilizator pe site sa arate asa: utilizatorul intra pe *site*, acceseaza pagina de *register*, isi creeaza un cont, iar apoi este trimis pe pagina de *log in* pentru a-si accesa contul. Faptul ca aceste doua componente apartin aceluiasi modul face ca timpi de incarcare dintre componenta de *register* si cea de *log in* sa fie inexistenti.

Modulul *Search-mod* contine lista de *anime*-uri si compoenta pentru afisarea detaliilor fiecarui *anime*. In componenta pentru lista de *anime*-uri au fost incluse si functii pentru sortarea listei in functie de nume, *rating*, popularitate si studio. Acestea doua au fost grupate intr-un singur modul deoarce compoenta *Anime* depinde de coponenta pentru lista de *anime*-uri numita *Ranking*-*list,* in sensul in care un utilizator va apasa pe numele unui *anime* si va fi redirectionat spre o pagina care cuprinde o poza reprezentativa pentru *anime*-ul respectiv, o descriere a actiunii, a personajelor si alte informatii folositoare despre *anime*-ul selectat. Astfel, componenta *Anime*, se foloseste exclusiv de informatiile furnizate de catre utilizator prin selectia unui *anime* din lista, iar faptul ca au fost grupate in acelasi modul face transmiterea de date de la o componenta la cealalta mai simpla.

Modulul de *Manage* cuprinde componentele de *manage* *account* si de *manage* *anime*, care sunt foarte similare in design si in functionalitate. Acestea cuprind partea de *update*si *delete*din operatiile CRUD-urile pentru ‘’baza de date’’ furnizata de API. A fost considerat faptul ca performanta nu va fi afectata daca odata cu partea de *manage* *account* se initializa si *manage* *anime*, si s-a preferat gruparea acestor doua componente in module, ca sa nu fi incarcate pentru fiecare pagina din aplicatie, fapt ce ajuta macar putin la performanta aplicatiei.

Despre baza de date folosita, aceasta este (cum a fost explicat si in sectiunea despre *request*-urile folosite), aceasta este formata dintr-un API hostat local format din 2 sectii: *Anime* si *Accounts.*

Sectia de *Accounts* cuprinde: un id care este unic, numele complet al *user*-ului, *username*-ul, *email*-ul, parola si raspunsul la intrebarea de Securitate.

Sectia de Anime cuprinde: un id unic, numele *anime*-ului, *rating*-ul, numarul de episode, studioul care a facut *anime*-ul, genul, un link catre imaginea afisata pe pagina *anime*-ului si descrierea acestuia.

Experienta pe *site* pentru un utilizator ar arata cam asa: acesta acceseaza site-ul si ajunge pe pagina de *home* unde are doua butoane*: Log In* si *Register*. Apasa pe butonul de *Register*, isi creeaza un cont dupa care este trimis la pagina de *Log In* unde, dupa ce introduce corect datele contului, este trimis pe pagina principala unde ii va fi aratata o lista cu toate *anime*-urile de pe site. Utilizatorul va putea alege un *anime* si sa se duca pe pagina acestuia, unde ii vor fi afisate mai multe detalii despre *anime*-ul respectiv. De asemenea, pe pagina cu lista de *anime*-uri, utilizatorul poate sa acceseze pagina de manage account unde poate sa isi schimbe datele contului. In *header* se afla butonul de *About* pe care utilizatorul poate sa il acceseze de pe orice pagina.

***Utilizarea aplicatiei***

Aplicatia dispune de un *header* si un *footer*, indiferent de pagina care este accesata. *Footer*-ul are numele autorilor, *header*-ul fiind mai complex. Acesta are logo-ul si numele site-ului, apasarea numelui ducand la intoarcerea la pagina *login/register*. Tot in header exista si butonul de *About*, care duce utilizatorul la o pagina cu detaliile celor care au realizat *site*-ul.

La deschiderea aplicatiei apar optiunile de *log* *in* si *register*. Daca se alege optiunea de register, utilizatorul completeaza datele in functie de criteriile fiecarui camp, iar la apasarea butonului register este redirectionat la pagina de *log* *in*.

Dupa o logare corecta, se ajunge la pagina principala a site-ului. Aceasta pagina are un tabel cu *anime*-urile inscrise pe site pana la momentul respectiv ordonate *by* *default* in functie de popularitate. Utilizatorul dispune de un *drop down menu* in care exista mai multe posibilitati de sortare a *anime*-urilor: alfabetic dupa nume sau studio, dupa *rating* si revenirea la optiunea de sortare *default*, cea dupa popularitate. Tabelul contine informatiile de baza ale *anime*-ului: locul in tabelul popularitatii, numele, *rating*-ul, numarul de episoade, studioul si genurile din care face parte. Numele anime-ului in tabel este *clickable*, accesarea acestuia redirectionand utilizatorul la pagina *anime*-ului respectiv.

Pagina principala contine si un *search* *bar*, in care utilizatorul poate cauta un *anime*. Daca este introdus un nume de *anime* valid, apasarea butonului de *search* duce la afisarea paginii *anime*-ului respectiv. Altfel, pe pagina apare un *message* *box* cu ‘*’Anime not found’’* . Langa *search bar* apare butonul de *Manage Anime*. Apasarea acestuia duce utilizatorul la o pagina in care poate adauga un anime, sau modifica unul existent. Asemanator cu acest buton, exista si butonul *Manage* *Account*, care ii permite utilizatorului modificarea si stergerea unui cont deja existent, cu conditia de a introduce *username*-ului corespunzator.

***Concluzii***

*Task*-ul de a crea un site a fost privit de noi ca o provocare. Posibilitatile erau destul de numeroase, dar partea cea mai importanta pentru noi a fost sa gasim ‘*’hook’’*-ul, motivul pentru care cineva ar alege produsul nostru in defavoarea altora: un *site* cu control nelimitat, cu scopul de a testa abilitatea oamenilor de a ajunge la un consens, a parut o idee destul de indrazneata pentru a fi puna in aplicare.

Proiectul a fost realizat cu gandul la un *site* deja existent, Wikipedia (un *site* modificat de utilizatori). Noi am luat conceptul de “poate contribui oricine” si l-am dus in extrem, in incercarea de a testa reactiile publicului, intr-un experiment social improvizat.

Fiind primul nostru proiect realizat in Javascript, folosind *framework*-ul Angular, a fost pe cat de frustrant, pe atat de interesant sa descoperim probleme si rezolvarile lor. Experienta capatata ne-a ajutat sa ne imbunatatim abilitatile de coordonare si lucru in echipa; generalitatea task-ului ne-a oferit oportunitatea de a ne dezvolta creativitatea, si faptul ca proiectul depindea de domeniul de *frontend*, in care nu mai lucrasem, am invatat foarte multe lucruri noi, in special legate de acest domeniu.