

Grupa 5, Semigrupa 2, An 2021-2022

Proiect IS

Rezervare pentru Sala Fitness

Pop Alexandra

Tarta Manuel

Orsan Tudor

CUPRINS:

1) Descrierea proiectului: In ce limbaj am lucrat, In ce framework, temele proiectului.

2) Diagrama use-case / Prezentare a functionalitatilor proiectului.

3) Implementarea proiectului, clase, metode, etc...

Impartirea proiectului in fisiere HTML, folosirea modelelor pentru a reprezenta anumite tabele, folosirea link-urilor URL pentru pagini, clasele ce reprezinta logica din views, design pattern-urile folosite.

4) Diagramele folosite / Activity Diagram, Sequence Diagram, State Diagram, ce ilustreaza fiecare.

5) Manual de utilizare a proiectului.

6) Bibliografie.

1) DESCRIEREA PROIECTULUI:

Acest proiect reprezinta un site unde te poti inregistra pentru a merge la o sala, clasa de fitness (Gym Class). Am lucrat in limbajul Python, in framework-ul Django, si am implementat functionalitati ce ajuta clienti sa se inroleze la o anumita clasa de fitness, in fiecare zi.

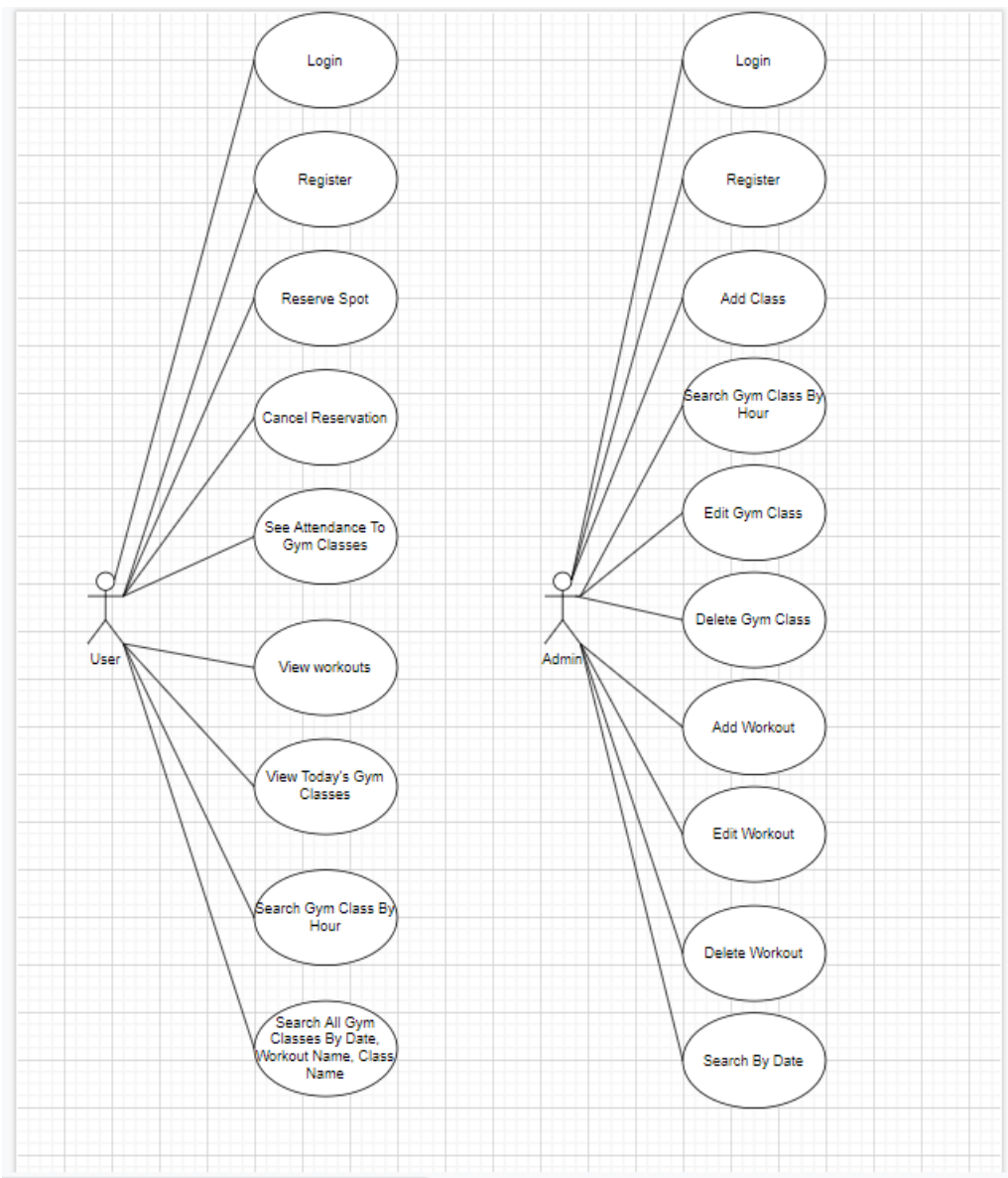
Ce teme sunt abordate in acest proiect: Avem functionalitate de login, ce lasa noi useri sa se inroleze la acest site, si sa beneficieze de serviciile sale. De asemenea, avem un Administrator, ce poate manipula datele pe care le observa un Client (De exemplu clasele actual disponibile, workout-urile specifice unei anumite zile, si toate datele legate de ce se afiseaza pentru un User). Dupa inregistrarea user-ului nou, si logarea sa, el are acces la un numar anume de beneficii. Pe prima pagina, se pot observa clasele ce sunt disponibile in ziua respectiva. Aceste clase sunt marcate dupa ora la care se intampla, workout-ul acelei clase (avem un singur workout disponibil pe zi), si cati useri sunt deja inscrisi la clasa respectiva. La vizualizarea workout-ului pentru clasa respectiva, dupa ce consideri descrierea acelui workout, poti alege sa te inscrii sau nu la clasa respectiva. Dupa, daca decizi, poti anula acea inscriere, astfel nu mai esti programat pentru clasa respectiva. Atunci cand te programezi la o clasa, nu mai poti sa te programezi la alta clasa, si este marcata clasa respectiva pentru acel user, pentru acea zi. De asemenea, poti vizualiza

toate clasele la care ai participat pana in acel moment. Toate aceste date, si anume clasele, pot fi cautate dupa anumite criterii, precum ora la care incep, data la care se afla, numele clasei, workout-ului, etc...;

Folosind framework-ul Django, am implementat toate aceste functionalitati, avand acel admin pentru controlul claselor si a workout-urilor, iar userii avand posibilitatea de a alege datele ce le convin in fiecare zi in care aceste clase au loc.

2) DIAGRAMA USE-CASE:

a) Diagrama Use-Case propriu-zisa:



b) Explicatiile pentru aceasta diagrama:

User-ul are urmatoarele actiuni posibile:

1. Login: introduce username si parola in field-urile destinate acestora, iar daca acestea sunt corecte este redirectionat catre pagina principala a client-ului, altfel primeste un mesaj de esec.
2. Register: trebuie sa introduca un username care nu a mai fost utilizat de alt user si o parola care sa respecte cateva constrangeri. In caz de success, i se inregistreaza contul si este redirectionat catre pagina principala, altfel, in caz de esec, i se afiseaza mesaje de atentionare si de indrumare.
3. Reserve Spot: alege clasa la care doreste sa isi faca rezervare si apasa pe butonul de "Reserve". Daca acesta nu mai are alta rezervare in acea zi si clasa nu este plina (maxim 5 persoane la o clasa), atunci rezervarea se termina cu succes si client-ul este adaugat in lista de participanti, altfel lista nu sufera nicio modificare.
4. Cancel Reservation: alege o clasa si apasa butonul de "Cancel". Daca a avut o rezervare la acea clasa, numele ii este scos de pe lista de participanti, altfel lista nu sufera modificari.
5. See Attendance To Gym Classes: dupa apasarea butonului operatiunii, user-ul este redirectionat catre o alta pagina unde ii sunt listate toate rezervarile facute pana in acel moment, in fiecare zi, luna si an. Daca nu are nicio rezervare facuta pana atunci, i se afiseaza un mesaj de atentionare.

6. View workouts: user-ul apasa pe butonul de “View workout” din dreptul unei clase, si dupa este redirectionat catre o alta pagina unde poate vedea toate detaliile legate de workout-ul acelei clase: nume, ora, descriere.
7. View Today’s Gym Classes: client-ul va vedea pe pagina principala doar clasele de fitness din ziua in care acesta se logheaza in aplicatie.
8. Search Gym Class By Hour: in pagina principala, user-ul poate sa dea search la clasele din acea zi dupa ora.
9. Search All Gym Classes By Date, Workout Name, Class Name: din fereastra in care isi poate vedea rezervarile facute pana in acel moment, poate sa caute anumite rezervari dupa data in care au avut loc, dupa numele workout-ului sau numele clasei. Daca ce a cautat nu exista, va primi un mesaj de atentionare.

Admin-ul are urmatoarele actiuni posibile:

1. Login: introduce username si parola in field-urile destinate acestora, iar daca acestea sunt corecte este redirectionat catre pagina principala a admin-ului, altfel primeste un mesaj de esec.
2. Register: trebuie sa introduca un username care nu a mai fost utilizat de alt user si o parola care sa respecte cateva constrangeri. In caz de success, i se inregistreaza contul si este redirectionat catre pagina principala, altfel, in caz de esec, i se afiseaza mesaje de atentionare si de indrumare.
3. Add Class: admin-ul poate sa adauge clase de fitness pentru oricare zi din an, adaugand informatii obligatorii legate de

workout-ul de la acea clasa, ora la care va avea loc si numele acesteia. Daca ora introdusa de acesta nu este corecta, va fi attentionat printr-un mesaj.

4. Search Gym Class By Hour: in pagina principala, admin-ul poate sa dea search la toate clasele dupa ora.
5. Edit Gym Class: in cazul unor schimbari in program, poate sa modifice detalii legate de workout, nume, ora, la clasele existente.
6. Delete Gym Class: poate sa stearga orice clasa existenta din baza de date, apasand butonul de "Delete" specific acelei clase.
7. Add Workout: admin-ul poate sa adauge workout-uri corespunzatoare oricarei zile din an, oferind informatii obligatorii despre nume, data la care are loc si descrierea exercitiilor.
8. Edit Workout: la fel ca la clasele de fitness, poate sa modifice unele detalii la orice workout.
9. Delete Workout: admin-ul poate sa stearga orice workout care nu mai este necesar, iar odata cu stergerea acestuia, vor fi sterse si toate clasele atribuite lui.
10. Search By Date: poate sa caute workout-urile in functie de data in care acestea sunt setate sa aiba loc, pentru o administrare mai usoara a acestora.

3) IMPLEMENTAREA PROIECTULUI:

Codul a fost implementat astfel:

Implementarea are patru componente: crearea modelelor pe care lucram (tabelele din baza de date), a view-urilor care sunt ca un controller, componentele care realizeaza logica care leaga modelele de interfetele grafice, fisierele html care sunt interfata grafica propriu-zisa in care plasam butoane, text field-uri, etc si urls in care zicem fiecare html ce view foloseste.

Ca modele avem doua: Workout si ClasaPrincipala.

Modelele le-am realizat impreuna pentru ca am dorit sa avem o baza pe care sa o stim toti si sa putem extinde mai usor proiectul.

Workout are ca si campuri un titlu, o data in care se foloseste acel workout la toate clasele din acea zi si o descriptie in care se precizeaza ce presupune acel workout din punct de vedere al exercitiilor. Toate aceste campuri pot fi editate de catre administrator. Am facut ca ordonarea intre workout-uri sa se faca in functie de data, cele mai vechi workout-uri fiind afisate primele.

ClasaPrincipala este tabelul care contine toate clasele de la acest gym. Ca si campuri, contine titlul, workout-ul care se va

tine la aceasta clasa, ora de inceput, un boolean complete care ne indica daca numarul de useri maximi pentru acea clasa a fost atins, o lista de participanti si numarul de participanti actual. Lista si numarul de participanti nu sunt editabile nici de catre administrator, dar sunt vizibile pentru un user care vrea sa isi rezerve un loc la acea clasa. Am decis ca numarul maxim de useri care se pot inscrie la o clasa sa fie 5 si ca un user sa se poate inscrie la o singura clasa din aceeasi zi.

Ca si view-uri, avem 14 si o sa vorbesc despre fiecare in parte. O sa vorbesc in acelasi timp si de fisierul html, cele doua fiind strans legate.

Avem un view, CustomLoginView care reprezinta fereastra de logare. Aceasta clasa mosteneste de la LoginView din django, deci logica de adaugare a userului in baza de date este facuta in spate. Am setat ca atunci cand un user este autentificat deja sa nu mai poata sa acceseze aceasta pagina ca sa nu se poata loga din nou. Metoda get_succes_url indica pagina la care sa fie redirectionat utilizatorul dupa ce a reusit sa se logheze cu succes. Noi avem doua tipuri de utilizatori: un administrator, care are username-ul Administrator si mai multi useri. Avem un if in care daca username-ul este Administrator, acesta este redirectionat la pagina principala pentru admin, altfel este user si este redirectionat spre pagina principala a userilor.

In html, avem un field pentru username, unul pentru parola, un buton de login si in caz ca utilizatorul nu are un cont inca, un link care il duce la pagina de register. La html avem un fisier principal main.html pe care il extind toate celelalte fisiere.

Acesta are cod general pentru dimensiunea paginilor, aranjarea in pagina, culori.

Alt view este CustomRegisterView care reprezinta fereastra pentru crearea unui cont. Si aceasta clasa mosteneste de la FormView din django deci are un template prestabilit. Are un field pentru username cu explicatii despre cum trebuie sa fie usernameul, un field pentru parola din nou cu explicatii despre ce trebuie sa contina parola si un field in care trebuie sa reintroduci parola pentru verificare. Pe langa asta mai este un buton care realizeaza inregistrarea si un link spre pagina de login in cazul in care utilizatorul are deja un cont. Si aici avem logica ca daca utilizatorul este logat deja este redirectionat pe pagina principala ca sa nu se poata inregistra din nou si dupa o inregistrare reusita, utilizatorul este si logat automat.

Login si Register le-am facut impreuna pentru ca au facut parte din tutorialul de obisnuire cu Django.

Dupa, avem pagina principala pentru administrator care mosteneste de la ListView. Asta inseamna ca pe aceasta pagina avem o lista, in acest caz o lista de clase pe care administratorul le poate edita sau le poate sterge. Obiectul pe care lucram, adica clase aici este ales prin setarea modelului in view (model = ClasaPrincipala). Aceasta lista poate fi filtrata in functie de ora prin apasarea unui buton. Pentru asta avem o metoda

get_context_data in care avem o lista cu toate clasele din tabela. Pe aceasta aplicam filter in functie de ora pe care o dam noi ca input. Aceasta lista filtrata o trimitem dupa in html cu context_object_name, o parcurgem cu un for si o afisam intr-un tabel. Tabelul contine titlul clasei, titlul workout-ului, ora de inceput, lista de participanti si pentru fiecare astfel de clasa din tabel avem optiunea sa dam edit sau delete. Cele doua presupun apasarea pe un link care te duce la o alta pagina pe care se va realiza operatia dorita.

Avem un parametru in plus la PaginaPrincipalaView, LoginRequireMixin care nu lasa un utilizator sa ajunga pe aceasta pagina daca nu este logat deja. Astfel ca, un utilizator nelogat va fi redirectionat automat spre pagina de logare.

La html avem un header care zice Hello, (username), un buton care este fie logout, un buton de Add Gym Class prin care administratorul este redirectionat spre pagina de pe care poate crea o clasa noua, tabela cu toate clasele si butoanele de edit si delete pentru fiecare clasa.

Clasa PaginaPrincipalaView impreuna cu html-ul paginaPrincipala.html a fost implementata de Tarta Manuel.

Pagina de creare a unei noi clase este implementata de clasa CustomCreateView1 care mosteneste de la CreateView din Django, deci si aceasta porneste de la un template existent. Si acest view are ca model tot ClasaPrincipala. Are ca field-uri

atributele unei clase, deci titlu, workout, ora de inceput si daca este complete sau nu. Metoda `form_valid` ne ajuta sa nu mai trebuiasca sa aleaga userul la crearea clasei, asa ca acesta este setat automat prin `form.instance.user = self.request.user`. Metoda `get` este pentru cazul in care un user care nu ar trebui sa ajunga pe aceasta pagina, totusi o acceseaza scriind direct adresa url a acestei pagini. In acest caz, orice user care nu are username Administrator este redirectionat spre pagina principala a userilor.

Fisierul html folosit pentru acest view este `create_update.html` care contine un header, un buton de Go Back care ne duce inapoi pe pagina principala si formul, adica toate acele campuri pe care le-am descris mai sus si care trebuie completate pentru a se putea crea noua clasa. Nu se poate lasa niciun field necompletat.

Clasa `CustomCreateView1` si `create_update.html` au fost implementate de Orsan Tudor.

Pagina de editare a unei clase existente este implementata de `CustomUpdateView1` si este foarte asemanatoare cu cea de creare a unei clase noi. Foloseste clasa `UpdateView` care contine si ea campurile clasei care pot fi editate, aceleasi ca si la creare si butonul de Submit care realizeaza operatia de editare. Pe langa acele campuri mai este butonul de Go Back pentru ca foloseste acelasi html ca si clasa pentru creare.

Pagina de delete este implementata de CustomDeleteView1. Fiind delete pentru o clasa, foloseste ca model ClasaPrincipala. Campul succes_url indica pagina spre care sa se faca redirectionarea in cazul in care stergerea a fost realizata cu succes. Metoda get este folosita in cazul in care un user care nu este administratorul incearca sa acceseze aceasta pagina. In acel caz, acel user va fi redirectionat la pagina principala pentru useri.

Fisierul delete.html este cel care realizeaza interfata pentru aceasta operatie. Contine un buton de Go Back care il duce pe utilizator inapoi pe pagina principala, un mesaj in care utilizatorul este intrebat daca este sigur ca vrea sa stearga aceasta clasa si butonul care realizeaza stergerea in spate. Si clasa CustomDeleteView1 implementeaza DeleteView din Django deci modificarile in baza de date la apasarea butonul sunt facute in spate, fara sa fi implementat noi. Am folosit si aici LoginRequiredMixin ca sa nu lasam utilizatori care nu sunt logati sa acceseze aceasta pagina si sa fie redirectionati la pagina de logare.

Paginile pentru editare si stergere a unei clase au fost implementate de Pop Alexandra

Inafara de cele zise pana acum, pe pagina principala a administratorului se mai afla si un buton care il duce spre pagina

pentru workout-uri. Aceasta pagina merge exact pe aceeasi idee cu pagina principala cu lista de clase, doar ca aici este o lista de workout-uri. Aici ordinea se face in functie de data stabilita pentru workout. Si aici, administratorul are optiunea sa filtreze lista astfel incat sa vada doar workout-urile dintr-o anumita data. Implementarea este la fel ca si la filtrarea claselor. View-ul este PaginaWorkoutView, are modelul Workout si metoda get care nu lasa utilizatorii care nu sunt logati pe aceasta pagina. De pe aceasta pagina administratorul mai are posibilitatea sa adauge un workout, operatie identica cu cea de la clasa, operatiile de edit si delete pentru fiecare workout si un buton de Go Back pentru a te intoarce la pagina principala.

Fisierul html este workoutPage.html care contine un buton, un tabel si cele doua linkuri spre paginile de edit si delete.

Pagina de workout a fost implementata de Tudor Orsan.

Paginile de add workout, edit si delete nu au nimic diferit fata de cele de la clase, doar folosesc un model diferit, deci au field-uri diferite. La workout, nu putem crea 2 workout-uri in aceeasi data.

Pagina de Add Workout si Edit Workout a fost implementata de Pop Alexandra, iar pagina de Delete Workout de Tarta Manuel.

Pagina principala pentru useri este implementata cu clasa `PaginaUserView`. Foloseste ca model `ClasaPrincipala` pentru ca pe aceasta pagina un utilizator poate vedea clasele disponibile din ziua curenta. Pentru o clasa, acesta poate vedea, numele clasei, numele workout-ului, ora la care se tine clasa, lista de participanti la acea clasa si numarul de locuri ocupate la acea clasa. Pe langa lista de clase, aceasta pagina mai contine un buton de search pentru a filtra clasele in functie de ora, un buton pentru a afla mai multe detalii despre workout-ul de la oricare dintre clase (clasele din aceeasi zi au acelasi workout, diferenta este ora la care se tin) si un buton care ne duce la o alta pagina pe care putem vedea istoricul claselor la care am luat parte.

Fisierul html pentru acest view este `paginaPrincipalaUser.html` care seamana cu cel de la administrator. Are un header cu Hello, (user), un buton de logout, un alt header care zice Vizualizare clase disponibile si data curenta, butonul de search, tabelul cu clasele si pentru fiecare clasa e butonul de View Workout si la final butonul de View Class Attendance.

Pagina principala pentru useri a fost implementata de Pop Alexandra.

Pagina de View Class Attendance seamana cu paginile principala. Este implementata prin clasa `CustomAttendanceListView` si contine filtrarea listelor prin butoane de search, trei in acest caz, in functie de data, de numele

workout-ului si de numele clasei. Pe langa asta mai contine lista claselor care reprezinta toate clasele la care a participat userul logat.

Pagina propriu-zisa este implementata cu fisierul `viewClassAttendanceList.html` care contine un titlu, un buton de Go Back, cele 3 butoane de search tabela pentru lista de clase care contine numele clasei, numele workoutului, data, ora si descrierea.

Pagina de View Class Attendance a fost implementata de Orsan Tudor.

Pagina pentru View Workout este implementata cu clasa `CustomWorkoutView` si de fisierul `html workoutPage.html`. `CustomWorkoutView` contine doua metode importante: `reserveSpot` si `cancelReservation`. Metoda `reserveSpot` primeste ca parametru un id care este primary key-ul clasei la care acel user vrea sa isi rezerve un loc. Inainte sa se poata face asta, este verificat daca acel user nu are un loc rezervat si in alta clasa din aceeasi zi. Daca mai are o rezervare, este redirectionat spre pagina principala si rezervarea nu se mai face. Daca nu mai are o alta rezervare se verifica daca clasa respectiva mai are locuri disponibile. Daca nu mai are, userul este redirectionat la pagina principala, daca mai are locuri rezervarea este facuta, userul este introdus in lista de participanti si numarul participantilor creste.

La metoda `cancelReservation` avem din nou un id, primary key-ul clasei respective. Verificam daca userul este in lista de participanti ai acelei clase. Daca este, il scoatem din lista si scadem numarul de participanti. Avem niste conditii pentru a merge bine legaturile cu alte metode. De exemplu daca lista este goala dam return direct, daca acel user era singurul participant, punem pe acele campuri `None`.

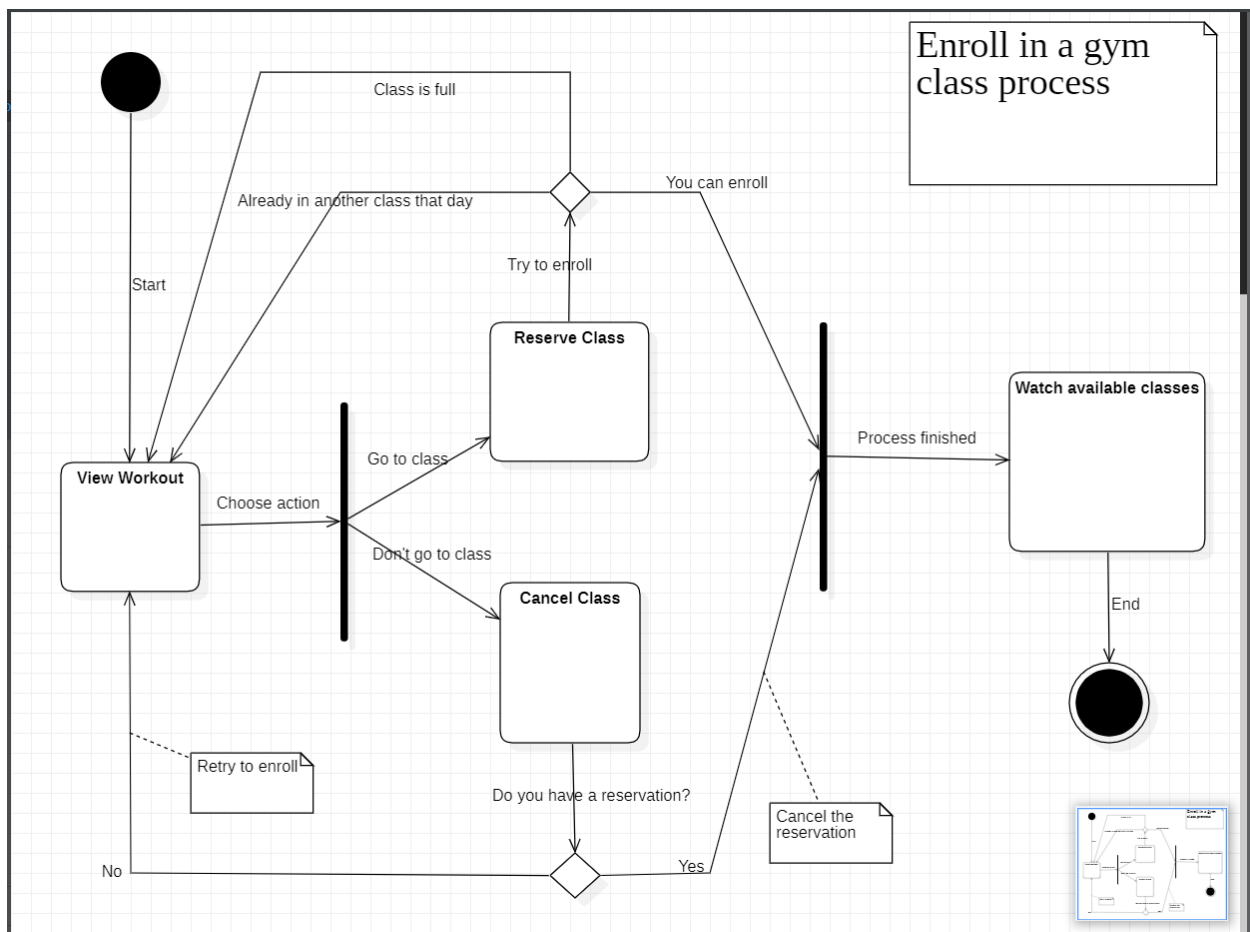
In html avem doar datele despre workout, adica titlul, data si descrierea si cele 2 butoane.

Functionalitatile aici au fost impartite la 2. Tarta Manuel a implementat `reserveSpot` si Pop Alexandra a implementat `cancelReservation`.

4) DIAGrameLE FOLOSITE:

Am impartit astfel diagramele:

a) Activity Diagram: (Orsan Tudor)



Logica acestei diagrame merge astfel:

Vrem sa ajungem la o concluzie cu starea unui user in functie de clasa in care se afla. Deci, pentru start, avem modul in care se afla la inceput, inrolat sau nu, si la final, avem modul in care se afla dupa ce a luat anumite actiuni. Intai, mergem pe pagina de ViewWorkout, pentru a alege ce dorim in continuare. Daca alegem sa dam cancel la class, avem urmatoarele situatii:

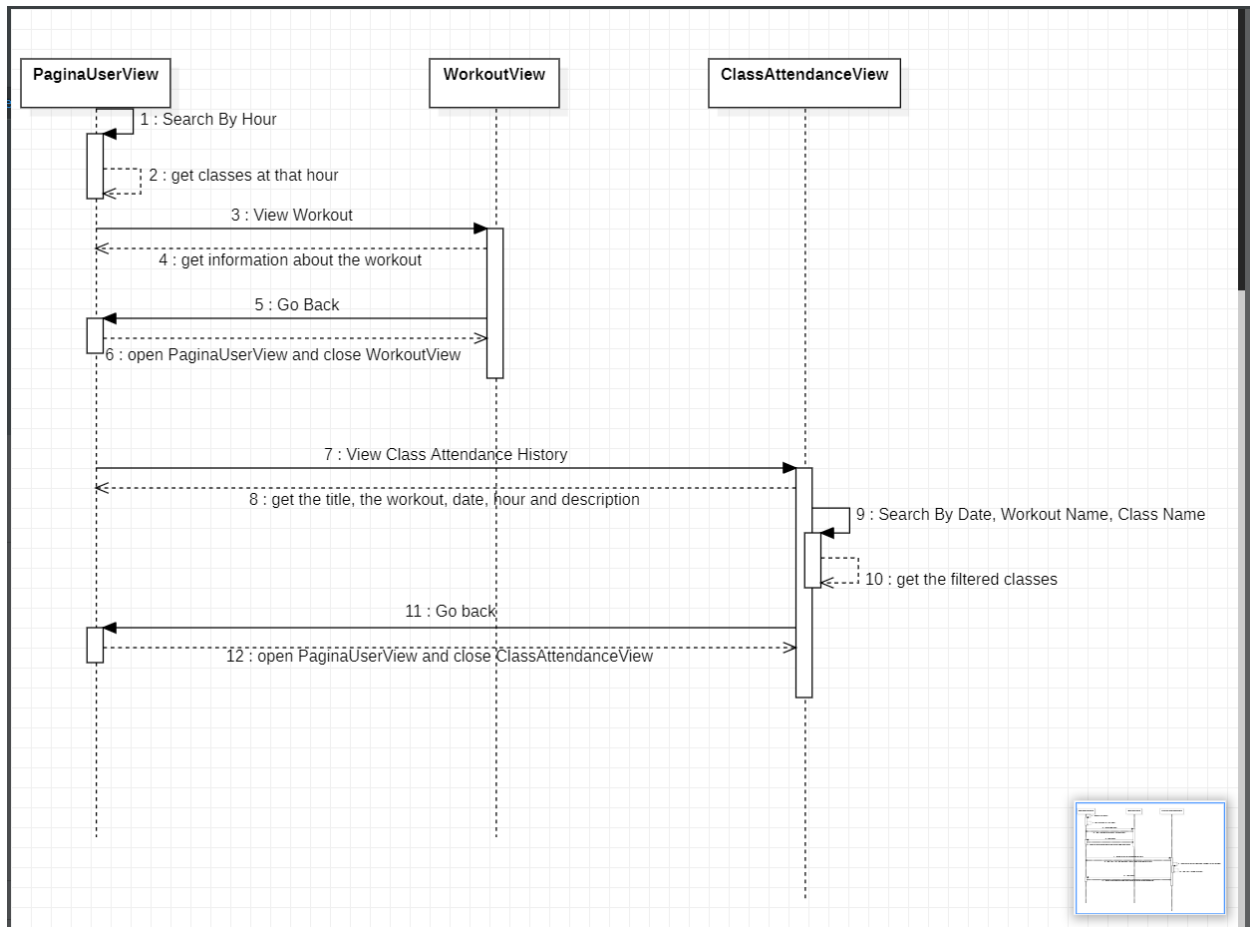
-Daca nu avem rezervare facuta, ne ducem inapoi la ViewWorkout. Daca avem mergem inapoi pe pagina de vazut clasele disponibile in ziua respectiva, acum fara sa mai avem inrolarea activa, terminand acest proces.

Pentru cand vrem sa ne inrolam la o clasa, incercam sa o facem, si avem unele situatii:

-Daca avem o clasa plina, adica toti 5 posibili utilizatori sunt inrolati deja, atunci mergem inapoi la pagina de ViewWorkout, fara sa ne fi inrolat, putem incepe de la capat. La fel si daca suntem deja inrolati la alta clasa, nu ne putem inrola la cea actuala, deci mergem inapoi la pagina de ViewWorkout. In schimb, daca nu suntem in aceste situatii, inseamna ca putem sa ne inrolam (mai este si cazul in care deja suntem inrolati, nu se intampla nimic in acel caz), si vom ajunge tot la pagina de vizualizare clase disponibile, dar acum fiind inrolati.

Acestea sunt cazurile de functionare abordate in aceasta diagrama. Daca ar fi sa adaug ceva la functionare, ar fi unele mesaje in cod pentru a intelege user-ul mai bine ce se intampla, in schimb, user-ul nu poate executa actiuni ilegale, deci nu este prea mare problema oricum.

b) Sequence Diagram: (Tarta Manuel)



Logica acestei diagrame merge astfel:

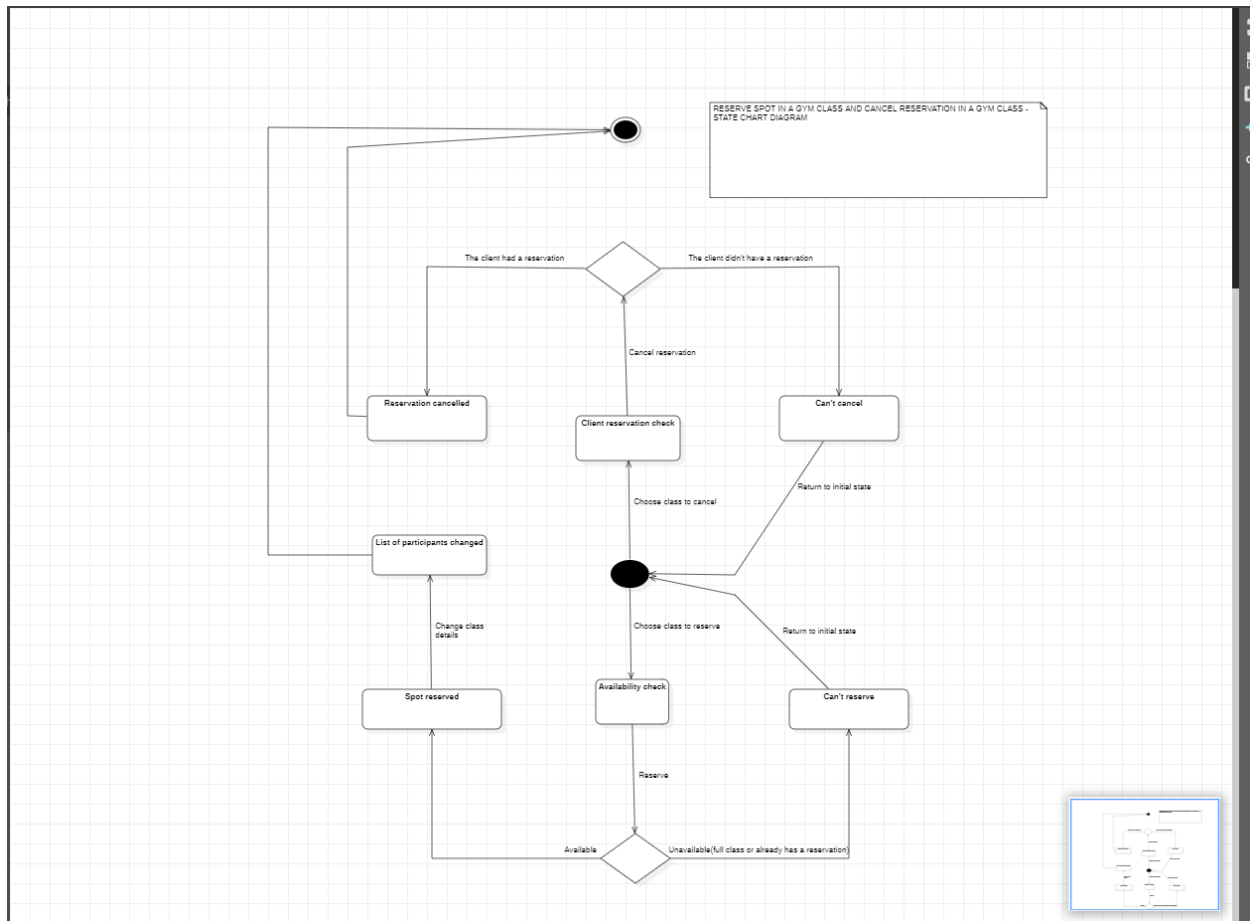
Diagrama de secventa descrie functionalitatea paginei principale a userului in relatia cu pagina de view workout si pagina de view class attendance history. Prima actiune care se poate realiza este cea de search care primeste ca raspuns o lista filtrate in functie de input. Dupa se poate apasa pe View Workout care deschide o alta pagina cu informatii despre

workout-ul clasei respective. De aici, urmatoarea posibilitate este sa apasam Go Back ca sa ne intoarcem pe pagina principala.

Optiunea pe care o avem dupa este sa apasam pe butonul View Class Attendance History care ne da ca raspuns o lista cu toate clasele la care acel user a participat. De aici avem posibilitatea sa facem search in functie de date, workout name si class name si primim o lista filtrate.

Ultima posibilitate este sa dam Go Back si sa ne intoarcem pe pagina principala.

c) State Diagram: (Pop Alexandra)



Logica acestei diagrame merge astfel:

Diagrama prezinta doua actiuni realizate de user: cea de rezervare a unui loc si de anulare a unei rezervari la o clasa de fitness.

In cazul unei rezervari, dupa alegerea clasei la care se doreste rezervarea, si dupa apasarea butonului de "Reserve", se face o verificare a starii in care se afla acea clasa.

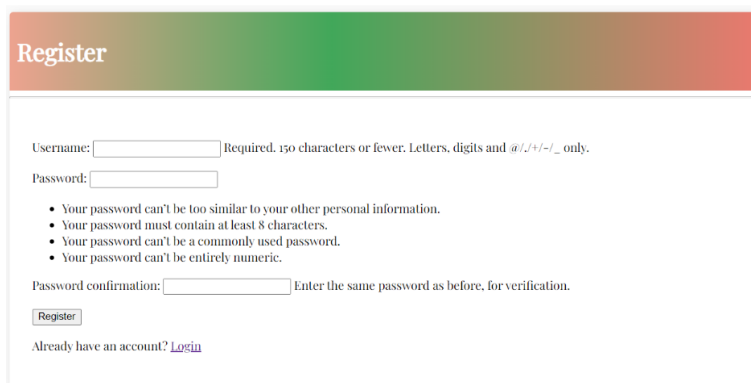
Daca user-ul nu este inrolat la o alta clasa din acea zi si daca acea clasa pe care a ales-o nu este plina, atunci rezervarea se poate realiza, iar user-ul este adaugat in lista de participanti a clasei, si actiunea este finalizata cu succes. Daca aceste conditii nu sunt satisfacute, rezervarea nu se poate realiza, iar clasa se intoarce in starea initiala din care a pornit, iar user-ul poate in continuare sa realizeze actiuni.

In cazul anularii unei rezervari, se alege clasa la care se doreste anularea rezervarii, si apoi se verifica conditia ca user-ul sa fie inrolat la acea clasa. Daca aceasta conditie este indeplinita, se realizeaza anularea, user-ul este scos de pe lista de participanti de la acea clasa, si se ajunge in starea finala cu success. In cazul in care nu se indeplineste conditia, anularea nu se realizeaza si are loc o intoarcere in starea initiala.

5) MANUALUL DE UTILIZARE FOLOSIT:

Urmatoarele explicatii vor ajuta cu modul de folosinta al acestui proiect:

a) Pentru inceput, avem register:



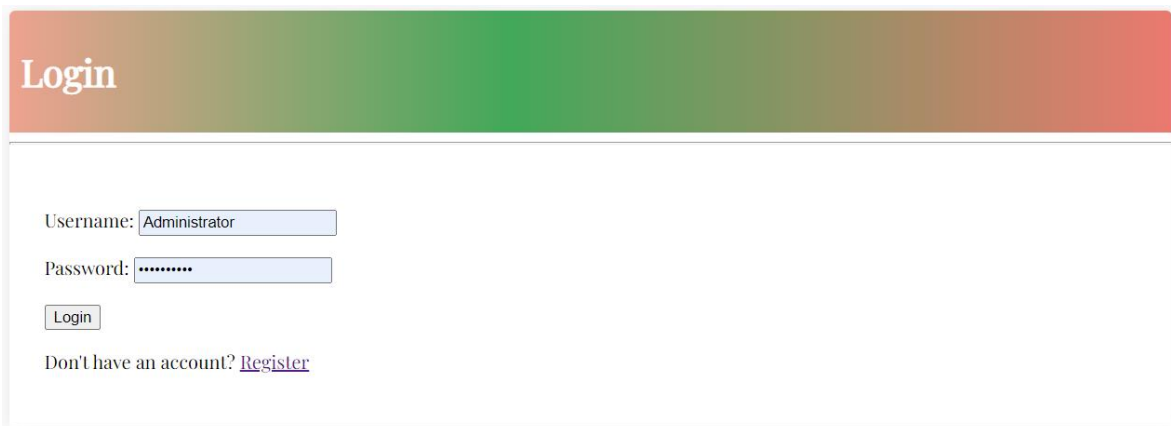
The screenshot shows a web registration form with a header bar containing the word "Register". The form fields and text are as follows:

- Username:** [input field] Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.
- Password:** [input field]
- Your password can't be too similar to your other personal information.
 - Your password must contain at least 8 characters.
 - Your password can't be a commonly used password.
 - Your password can't be entirely numeric.
- Password confirmation:** [input field] Enter the same password as before, for verification.
-
- Already have an account? [Login](#)

-Poti sa te inregistrezi ca un nou user, dand un nume si o parola dupa anumite specificatii explicate pe acea pagina. Daca datele introduse sunt corect puse, atunci pot da register. Daca doresti sa mergi inapoi, pentru ca ai deja un cont, poti da pe butonul de

Login. Numele este unic, asa ca am inregistrat un Administrator cu numele Administrator si parola AdminAdmin pentru a se ocupa de treburile administrative. In rest, toti ceilalti useri se pot inregistra cu orice nume si orice parola. Userii inregistrati de noi sunt dupa formula User1 si parola User1User1, pana la 10. (Pentru simplitatea verificarilor)

b) Avem functionalitatea pentru pagina de login:



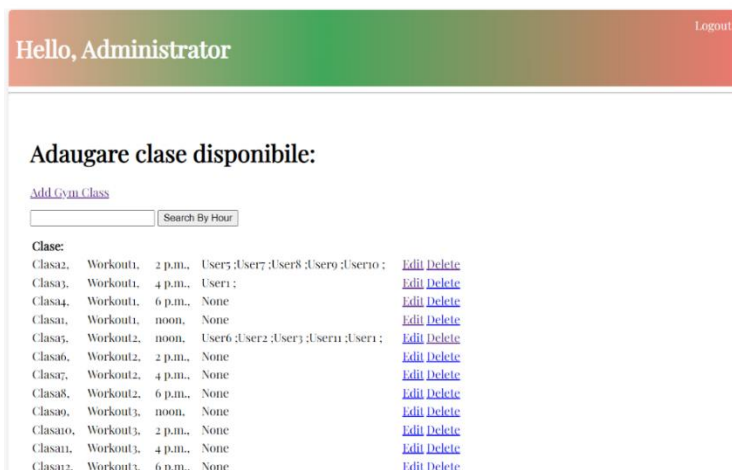
The screenshot shows a login page with a green header bar containing the word "Login" in white. Below the header is a white form area. Inside the form, there are two input fields: "Username:" with the text "Administrator" and "Password:" with masked characters "*****". Below these fields is a "Login" button. At the bottom of the form, there is a link that says "Don't have an account? [Register](#)".

-Dupa inregistrare, doar numele si parolele inregistrate functioneaza. Trebuie introduse ambele, altfel nu poti sa te loghezi. Exista, cum ziceam, doar un nume per user, deci nu se pot repeta.

-Ce am facut pentru a avea siguranta pe acest site este ca am restrictionat paginile ce nu ar trebui accesate de anumiti useri. Adica doar user-ul numit Administrator are acces la paginile de

editare si creare, iar ceilalti useri sunt redirectionati la paginile lor. Nici prin modificarea link-ului nu se poate accesa paginile, deoarece este redirectionare din nou, deci accesul este bine impartit.

c) Pentru prima pagina a administratorului:



Header: Hello, Administrator Logout

Adaugare clase disponibile:

[Add Gym Class](#)

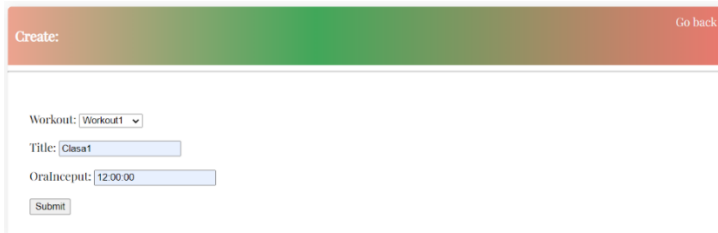
Search By Hour

Clase:					
Clasa2.	Workout1.	2 p.m.	User5 ;User7 ;User8 ;User9 ;User10 ;	Edit	Delete
Clasa3.	Workout1.	4 p.m.	User1 ;	Edit	Delete
Clasa4.	Workout1.	6 p.m.	None	Edit	Delete
Clasa.	Workout1.	noon.	None	Edit	Delete
Clasa5.	Workout2.	noon.	User6 ;User2 ;User3 ;User11 ;User1 ;	Edit	Delete
Clasa6.	Workout2.	2 p.m.	None	Edit	Delete
Clasa7.	Workout2.	4 p.m.	None	Edit	Delete
Clasa8.	Workout2.	6 p.m.	None	Edit	Delete
Clasa9.	Workout3.	noon.	None	Edit	Delete
Clasa10.	Workout3.	2 p.m.	None	Edit	Delete
Clasa11.	Workout3.	4 p.m.	None	Edit	Delete
Clasa12.	Workout3.	6 p.m.	None	Edit	Delete

-De aici putem accesa aproximativ toate functionalitatile administratorului. Putem sa cautam clase dupa ora la care se afla (4 cauta de exemplu 14, ne uitam daca ce cautam contine 4), si dupa o cautare ramane scris la ce am dat search inainte, putand continua cautarea, putem da edit si delete la clasele existente, putem da add la noi clase, si putem sa ne uitam la pagina de workouts; De asemenea, de acum majoritatea paginilor au ori un buton de Logout, care iti permite logarea cu alt user, eliminand

user-ul actual din folosire, ori un buton de back, ce duce catre pagina unde erai inainte. Aceste functionalitati merg la fel peste tot.

d) Pentru pagina de creare a noi clase:



The screenshot shows a web form titled 'Create:' in a red header bar. In the top right corner of the header is a 'Go back' link. The form contains three input fields: a dropdown menu for 'Workout' with 'Workout1' selected, a text input for 'Title' with 'Clasa1' entered, and a text input for 'Orainceput' with '12:00:00' entered. A 'Submit' button is located at the bottom left of the form.

-Aici cream dand un workout deja existent, nu putem sa nu avem un workout in acea zi, de asemenea, este necesara si o ora de inceput, si un titlu al clasei anume. Ora de inceput foloseste field de date, pentru a nu putea introduce alte date, nefolositoare.

e) Cand facem update la o clasa: (Sau la workout)

Create Workout: [Go back](#)

Title:

Date:

5 push ups

Description:

-Putem sa schimbam aceste date introduse mai devreme, si anume sa alegem alte campuri, alt workout, o data corect pusa, pana si titlul poate fi schimbat.

f) Daca vrem sa stergem o clasa:

[Go back](#)

Are you sure you want to delete this class: Clasa2?

-Cand stergem o clasa, inseamna ca nu va mai aparea in baza de date nicaieri, deci nici user-ul nici admin-ul nu mai au acces la clasa stearsa.

g) Dupa create, edit, delete pentru clase, avem pagina de workout:

Workout Page: [Go back](#)

[Add Workout](#)

Workouts:

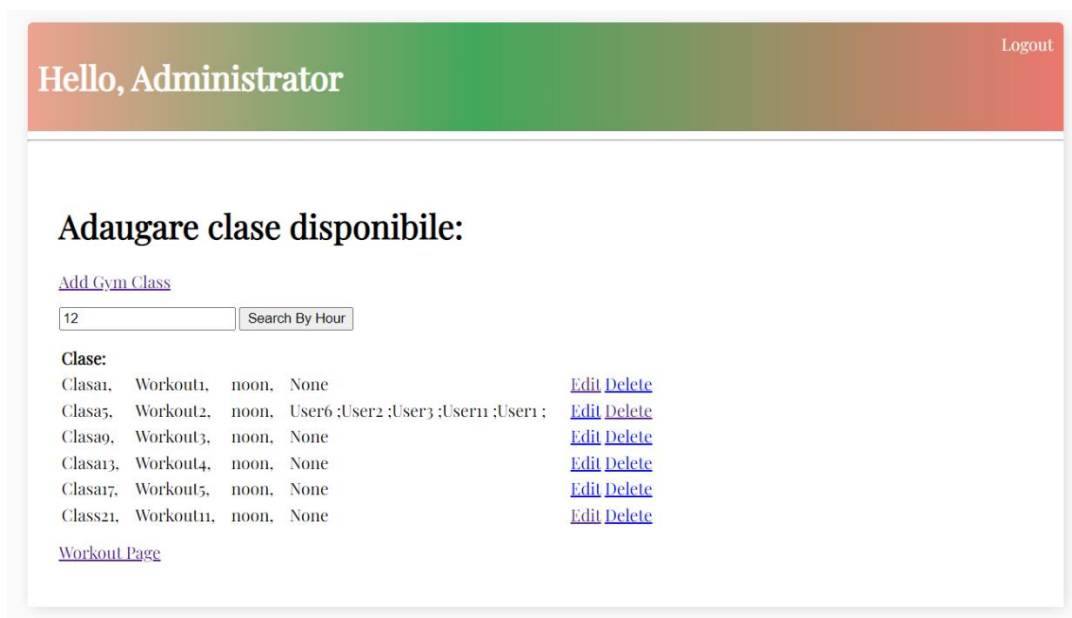
Workout1,	Jan. 1, 2022	Edit Delete
Workout2,	Jan. 2, 2022	Edit Delete
Workout3,	Jan. 3, 2022	Edit Delete
Workout4,	Jan. 4, 2022	Edit Delete
Workout5,	Jan. 5, 2022	Edit Delete
Workout6,	Jan. 6, 2022	Edit Delete
Workout7,	Jan. 7, 2022	Edit Delete
Workout8,	Jan. 8, 2022	Edit Delete
Workout9,	Jan. 9, 2022	Edit Delete
Workout10,	Jan. 10, 2022	Edit Delete
Workout11,	Jan. 11, 2022	Edit Delete
Workout12,	Jan. 12, 2022	Edit Delete
Workout13,	Jan. 13, 2022	Edit Delete
Workout14,	Jan. 14, 2022	Edit Delete
Workout15,	Jan. 15, 2022	Edit Delete

-Pentru aceasta pagina, putem face aproximativ aceleasi operatii ca pentru clasa. Afisam unele campuri, precum data la care este acel workout, titlul sau, si putem din nou cauta, de data aceasta dupa data la care se afla. 2022 este de exemplu un an pe care il poti cauta, si dupa vei avea doar workout-urile din 2022.

Operatiile mai sunt Create (Add), Edit si Delete, exact ca pentru clasa. Inafara de faptul ca avem alte campuri, si anume titlu, data si description, nu avem nimic diferit aici. Este de remarcat faptul

ca atunci cand se sterge un Workout, se sterge si clasa aferenta lui, deoarece nu mai exista clasa daca nu exista un workout pentru acea clasa.

h) Pentru functionalitatea de search a administratorului:



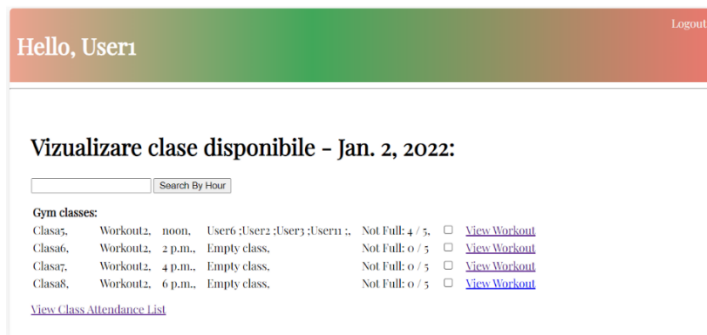
The screenshot shows the administrator interface. At the top, there is a green header with the text "Hello, Administrator" and a "Logout" link. Below the header, there is a section titled "Adaugare clase disponibile:". Under this title, there is a link "Add Gym Class". Below the link, there is a search bar with the number "12" entered and a button labeled "Search By Hour". Below the search bar, there is a table with the following data:

Clase:				
Clasa1,	Workout1,	noon,	None	Edit Delete
Clasa5,	Workout2,	noon,	User6 ;User2 ;User3 ;User11 ;User1 :	Edit Delete
Clasa9,	Workout3,	noon,	None	Edit Delete
Clasa13,	Workout4,	noon,	None	Edit Delete
Clasa17,	Workout5,	noon,	None	Edit Delete
Class21,	Workout11,	noon,	None	Edit Delete

Below the table, there is a link "Workout Page".

-Atunci cand cauti de exemplu 14, se vor afisa doar clasele relevante, si anume cele cu 2 p.m.; O functionalitate simplista, explicata si mai sus;

i) Am trecut acum la pagina user-ului:



-La aceasta pagina au acces, si sunt redirectionati de fiecare data la inceput, toti userii noi, prima pagina dupa logare.

Functionalitatile sunt urmatoarele: Putem sa cautam dupa hour clasele, la fel ca pentru administrator. Din prima, toate clasele vizibile sunt doar cele din ziua actuala, scrisa mai sus, in acest caz 2 ianuarie 2022. De aici poti viziona workout-ul actual, al clasei respective (toate clasele avand acelasi workout, dar la ore diferite), si sa vezi toate clasele la care ai participat pana in acea zi.

j) User-ul poate sa se uite la pagina workout-ului:

Workout:

Go back

Title: Workout2

Date: Jan. 2, 2022

Description: 5 pull ups

Reserve Class

Cancel Reservation

-Pe aceasta pagina, pe langa datele de vizionat, adica titlul, data si descriptia workout-ului selectat (la anumita clasa), putem sa rezervam anumita clasa, sau sa dam cancel la rezervare, in cazul in care avem deja, Atunci cand dam pe butonul de rezervare, suntem directionati inapoi la pagina principala a user-ului, si aici exista mai multe cazuri: Daca este clasa full, daca suntem deja inregistrati la aceasta clasa sau la altele din aceasta zi, atunci nu ne inregistreaza la clasa respectiva, in schimb, daca nu sunt indeplinite aceste conditii, ne inregistreaza. Pentru cancel, daca nu avem rezervare la acea clasa, nu se intampla nimic, dar daca avem, ne va scoate din lista celor rezervati la acea clasa (putand sa ne rezervam locul la alta clasa, tot din acea zi).

k) Ce se intampla cand faci rezervarea:



-Acum ca am facut rezervarea aparem pe pagina principala, la clasa anume, si avem o bifa in dreptul clasei selectate. Nu putem bifa manual, este bifat automat atunci cand te inregistrezi la o clasa anume. Sunt afisate locurile disponibile, daca se ajunge la 5/5, inseamna ca avem o clasa Full, si nu ne mai putem inregistra, nu mai este disponibila pentru inregistrare. In schimb, esti inregistrat, si apare faptul ca trebuie sa te prezinti la acea clasa in mod particular.

1) Pentru istoricul claselor unui user:

Vizualizare a istoricului claselor pentru User1

Go back

Search by date:

Search

Search by workout name:

Search

Search by class name:

Search

Lista claselor precedente:

Clasa1, Workout1, Jan. 1, 2022, 4 p.m., 5 push ups ;

Clasa5, Workout2, Jan. 2, 2022, noon, 5 pull ups ;

-Ce putem vedea pe aceasta pagina: Putem vedea toate clasele la care am participat, cu anumite campuri atat din clase, cat si din workout-ul acelei clase (precum titlurile, data, ora, etc...). Aceste clase sunt salvate in baza de date, in fiecare zi cand se alege o clasa noua (si raman acolo si dupa terminarea zilei anume). Ca si inainte, avem mai multe posibilitati pentru cautare, dupa data, workout name si class name, toate fiind cautari separate (Nu se intampla in acelasi timp, deci poti cauta doar dupa un criteriu anume odata).

m) Ce se intampla cand nu se gaseste data cautata:

Vizualizare a istoricului claselor pentru User1Go back

Search by date:

Search

Search by workout name:

Workout2

Search

Search by class name:

Search

Lista claselor precedente:

Nu ati participat la clase in trecut!

-Atunci cand nu se gaseste acea data cautata, de exemplu WorkoutJ, atunci cand noi am participat doar la WorkoutI, inseamna ca lista va fi goala: Exista un mesaj specific pentru acel caz, daca nu am participat la nici o clasa cu acele specificatii.

Acestea au fost majoritatea cazurilor de functionare posibile, dupa acest manual se poate intelege majoritatea functionalitatilor posibile, atat din punctul de vedere al administratorului, cat si a user-ului de rand.

6) BIBLIOGRAFIE:

Resursele folosite pentru acest proiect:

- [1] <https://docs.djangoproject.com/en/4.0/intro/tutorial01/>
- [2] <https://docs.djangoproject.com/en/4.0/ref/utils/>
- [3] https://www.tutorialspoint.com/django/django_basics.htm
- [4] <https://www.w3schools.com/>
- [5] <https://docs.djangoproject.com/en/4.0/topics/db/queries/>
- [6] <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django> .