



UNIVERSITATEA DIN BUCUREȘTI

**FACULTATEA
DE
MATEMATICĂ ȘI INFORMATICĂ**



SPECIALIZAREA INFORMATICA

Lucrare de licență

Calorie

Coordonator științific: Radu Boriga

Absolvent: Radu Cosmin-Andrei

București,

Iunie 2023

Rezumat

Într-o lume în care ne confruntăm tot mai des cu problemele ce țin de sănătatea alimentară, ar trebui să devină o preocupare importantă abordarea unui stil de viață mai benefic corpului. Multe persoane, din cauza lipsei de informare sau din cauza programului încărcat, nu au cum să gestioneze raportul caloric, astfel încât să își satisfacă nevoile individuale.

În ziua de astăzi, putem să ne bucurăm de apogeul erei din punct de vedere digital, astfel, este foarte ușor să ne consultăm alimentația folosind o aplicație web.

Prin urmare, în această lucrare urmează să prezint procesul dezvoltării unei aplicații web care vine în ajutorul oamenilor, cu scopul calculării necesarului caloric zilnic și recomandării de rețete mulate pe nevoile fiecăruia. Aplicația mea își propune să ofere soluții personalizate în materie de alimentație sănătoasă, ținând cont atât de obiectivele fiecărui individ, cât și de preferințele sale.

Consider că am elaborat o aplicație destinată fiecărei categorii de oameni, unicitatea ei fiind conturată de adaptabilitatea planurilor alimentare, în funcție de preferințele fiecărui utilizator, existând posibilitatea de personalizare în funcție de nevoile fiecărui om, precum diferite cerințe dietetice sau crearea de rețete pe baza produselor preferate.

Această aplicație constituie un instrument util și practic din toate punctele de vedere, aducând o contribuție semnificativă domeniului alimentației sănătoase. Are scopul de a reliefa importanța unui stil de viață controlat din punct de vedere alimentar, oferind un acces rapid la strategia nutrițională recomandată.

Summary

In a world where we increasingly face the challenge of balancing our diet in a healthy manner, it should become a significant concern to adopt a lifestyle that benefits our bodies. Many individuals, due to a lack of information or busy schedules, struggle to manage their calorie intake in order to meet their individual needs.

Today, we can take advantage of the digital era's peak, making it very easy to monitor our diet using a web application. Therefore, this paper aims to present the development process of a web application that assists people in calculating their daily calorie requirements and provides recipe recommendations tailored to their needs. My application aims to offer personalized solutions in the realm of healthy eating, taking into account each individual's goals and preferences.

I believe I have created an application that caters to every category of people, with its uniqueness shaped by the adaptability of recipes based on each user's preferences, including the ability to customize requirements such as different dietary needs or selecting preferred products.

This application serves as a practical and valuable tool in every aspect, making a significant contribution to the field of healthy eating. Its purpose is to highlight the importance of a controlled dietary lifestyle, providing quick access to the recommended nutritional strategy.

Cuprins

1. Introducere	6
1.1 Motivație	6
1.2 Istoric.....	9
2. Tehnologiile folosite.....	11
2.1 Introducere in tehnologii	11
2.2 Tehnologii relevante.....	11
2.2.1 MySQL.....	11
2.2.2 React.....	11
2.2.3 Node.js.....	11
2.3 Aplicații relevante	12
3. Preliminarii.....	13
3.1 Necesarul caloric	13
3.1.1 Descriere.....	13
3.1.2 Formula de calcul	13
3.2 Implementare.....	14
4. Arhitectura aplicației	16
4.1 Arhitectura generală	16
4.1.1 Descrierea arhitecturii generale.....	16
4.1.2 Componente principale	16
4.2 Împărțirea pe rute	16
4.2.1 Rutele	17
4.2.2 Navigarea intre rute.....	20
5. Baza de date	21
5.1 Structura bazei de date	21

5.1.1	Tabele	21
5.1.2	Constrângeri	23
5.1.3	Relații	24
5.2	Interacțiunea cu baza de date.....	25
5.2.1	Conexiune.....	25
5.2.2	CRUD.....	26
6.	Funcționalități.....	30
6.1	Introducere in funcționalități.....	30
6.1.1	Funcționalitate cheie	30
6.1.2	Descriere detaliată si exemple.....	30
6.2	Funcționalități auxiliare.....	33
7.	Idei de dezvoltare	39
8.	Concluzie	40
	Bibliografie.....	41

1. Introducere

1.1 Motivație

În decursul ultimilor ani, predominant fiind timpul afectat de către pandemie, mulți oameni fie și-au neglijat dieta, fie au abordat un stil de viață mai sănătos. Indiferent de caz, un mare procentaj, din cauză că se confruntă cu dificultăți de gestionare corectă și de atingere a obiectivelor nutriționale, pentru a menține un echilibru zilnic al caloriilor ar putea găsi această aplicație ca fiind o soluție eficientă și, mai ales, ușor accesibilă.

Motivația mea pentru dezvoltarea acestei aplicații are la bază atât dorința de a aduce o schimbare pozitivă asupra vieții oamenilor, dar și experiența personală care m-a făcut să conștientizez beneficiile aduse prin intermediul unei alimentații sănătoase și echilibrate.

În plus, am convingerea că, ținând cont de contextul actual, sustragerea informațiilor combinată cu utilizarea tehnologiei reprezintă un proces facil, accelerant pentru trecerea la un stil de viață sănătos. Astfel, se pune accentul pe eficiența adaptării unui regim cu ajutorul telefonului, deoarece acest dispozitiv este prezent în viețile noastre, utilizatorul scutește mult timp, programul încărcat devenind un obstacol diminuat.

Doresc să ofer prin intermediul acestei aplicații o experiență benefică utilizatorului, punându-i la dispoziție instrumentele și cunoștințele necesare, astfel încât să descopere o soluție, personalizată și adaptabilă nevoilor, pentru dorințele lor cu referire la slăbit, îngrășat sau menținut într-un stil sănătos.

O alimentație nesănătoasă și lipsită de un program bine structurat va avea repercusiuni pe viitor. Situația actuală a oamenilor în ceea ce privește dieta este una complexă. Majoritatea persoanelor au un ritm alert al vieții, ajungând să se confrunte cu lipsa de timp, însă există și cazul persoanelor care doresc să adopte diete speciale, cum ar fi veganismul sau vegetarianismul, dorind să aibă acces mai ușor la mai multe idei de rețete.

De exemplu, există mai multe studii care m-au motivat să abordez tema mâncatului haotic sau nesănătos, astfel, doresc să menționez două dintre acestea pentru a sublinia mai în profunzime motivul îngrijorării mele în legătură cu expunerea oamenilor unui stil alimentar nesănătos.

Un studiu realizat de Pew Research Center[1] a constatat faptul că aproximativ 59% dintre americanii care au ajutat la construirea acestui studiu spun că mănâncă mai mult decât ar trebui de fapt să o facă. Deși majoritatea a declarat faptul că depășesc limita superioară a cantității de mâncare consumată, un alt raport descris de el ilustrează scăderea plăcerii oamenilor de a mânca.

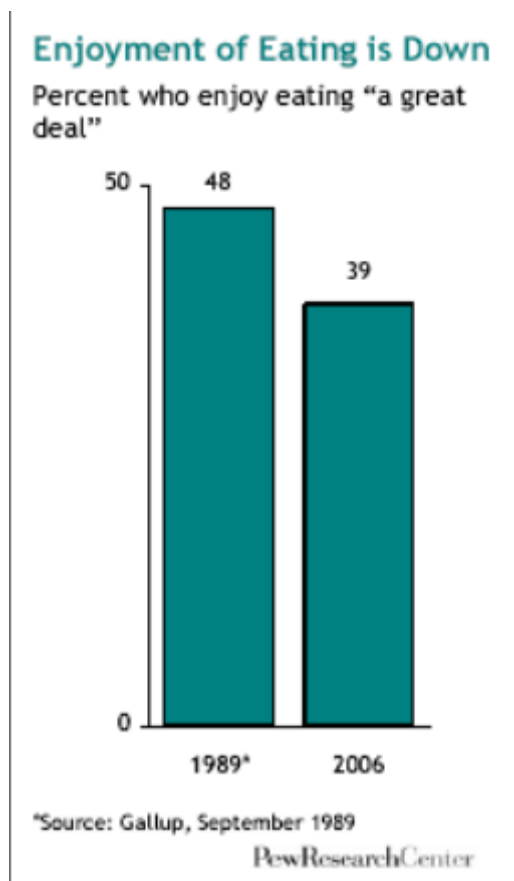


Fig 1.1.1 Placerea de a manca

Această diagramă conturează și mai mult importanța unei astfel de aplicații, întrucât lipsa plăcerii de a mânca poate proveni de la o dietă repetitivă, fiecare persoană simțind nevoia de diversificare, lucrarea mea satisfăcând genul ăsta de dorințe prin numărul mare de rețete regăsite și descrise.

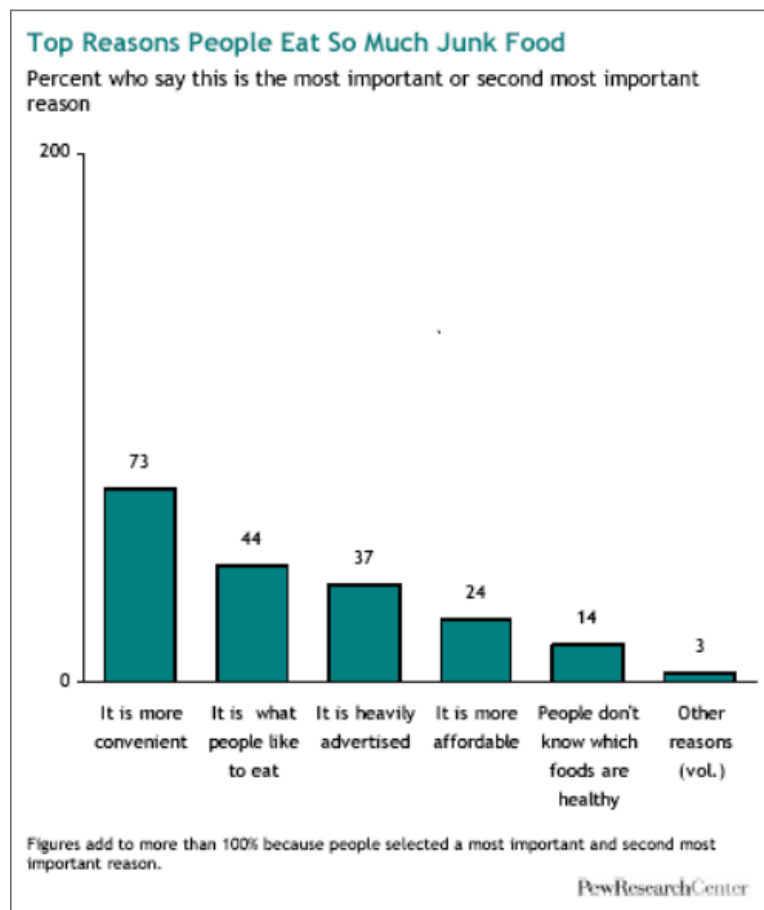


Fig 1.1.2 Motivele consumului de mancare nesanatoasa

Pe lângă cele menționate mai sus, diagrama anterior prezentată susține necesitatea aplicației, două dintre cele mai întâlnite motive (unul dintre ele fiind și motivul principal) pentru care lumea mănâncă Fast Food sunt ușurința de a o face și lipsa de informație în legătură cu ce este sănătos și ce nu. O astfel de aplicație ar atenua aceste practici nesănătoase.

Un alt studiu semnificativ care m-a motivat să creez această aplicație, este dezvoltat de The Hope Center[2] și se rezumă la cantitatea redusă de mâncare pe care o consumă studenții. Un factor foarte important de anunțat este stilul haotic de viață al studenților, nu doar că nu mănâncă suficient, dar nici nu o fac atunci când ar trebui.

Pe lângă stresul indus de viața de zi cu zi, acest sentiment poate fi amplificat de un mod neregulat de alimentare.

Astfel, acest studiu a descoperit că 45% dintre studenții care au participat la acest interviu mănâncă mai puțin decât ar trebui, fapt ce ar trebui reglat pentru a atinge capacitate și performanțe maxime.

Sunt încrezător că această aplicație web va inspira oamenii să facă alegeri cât mai benefice lor. Prin intermediul acestei motivații puternice, doresc să contribui la promovarea unei culturi a sănătății și bunăstării, iar aplicația dezvoltată și descrisă în această lucrare este primul meu instrument prin care pot aduce această schimbare pozitivă.

1.2 Istoric

Evoluția tehnologică este reflectată în profunzime de aplicațiile destinate persoanelor ce doresc să își calculeze kaloriile zilnice. În decursul a mai multor decenii, aceste aplicații au jucat un rol important în educarea oamenilor despre importanța unei alimentații echilibrate și în facilitarea monitorizării kaloriilor consumate zilnic.

Astfel, odată cu dezvoltarea tehnologiei, au apărut progrese mari și asupra aplicațiilor de acest gen, astăzi existând metode mult mai precise și mai eficiente. De exemplu, în primă fază, acestea erau disponibile doar pe calculatoarele personale și necesitau introducerea manuală a datelor despre alimente și porțiile consumate. Însă, din cauza timpului necesar în cantități mari, utilizatorii predominanți ai acestor aplicații erau specialiștii în nutriție și cercetătorii, întrucât le ofereau un bun mod de aprofundare a domeniului.

Ulterior, datorită utilizării cât mai frecvente a tehnologiei mobile, dezvoltatorii au făcut posibilă accesarea acestor aplicații de calculat kaloriile și de pe telefoanele mobile, publicul țintă devenind mult mai larg.

Aceste aplicații au început să introducă baze de date extinse cu alimente și nutrienți, facilitând căutarea și înregistrarea alimentelor consumate. Un foarte mare progress înregistrându-se în momentul implementării scanării codurilor de bare, cu scopul de a simplifica procesul de înregistrare a alimentelor ambalate.

În prezent, aplicațiile de calculat kaloriile au devenit un instrument indispensabil pentru cei care doresc să își îmbunătățească sănătatea și să își atingă obiectivele nutriționale.

Astfel, în urma unui studiu, au fost colectate date de la 1357 de adulți, în cadrul căruia 964 de oameni (71%) au folosit o aplicație de urmărit kaloriile într-un anumit moment al vieții, iar 531 de oameni (39%) dețin la momentul actual cel puțin o aplicație de acest gen.[3]

Prin urmare, în prezent există mai multe aplicații folositoare pentru evidențierea numărului de calorii consumate:

1. MyFitnessPal: este cea mai bine văzută, din punct de vedere al funcționalităților. Oferă posibilitatea de a înregistra ușor alimentele prin scanarea codurilor de bare, pune la dispoziție o vastă bază de date alimentară, înregistrarea meselor la restaurante, importarea de rețete și calcularea automată a kaloriilor, Aplicația oferă grafice de greutate și posibilitatea de a încarcă fotografii, cu scopul de a ține cont de progres. Mai mult de atât, dispune de un process de conectare și comunicare cu prietenii, permițând susținerea în procesul lor de dezvoltare nutritională. Prin urmare, este o unealtă cuprinzătoare pentru atingerea obiectivelor de sanatate și fitness.[4]
2. LifeSum: este foarte sus catalogată datorită accentului pe care îl pune pe mâncarea sănătoasă, calitatea alimentelor fiind valorificată prin intermediul acestei aplicații ușor de folosit. Sunt exemplificate și explicate mesele necesare pe zi, valoarea nutritivă a diferitelor alimente și importanța motivației în atingerea obiectivelor zilnice. [4]
3. MyPlate Calorie Counter: spre deosebire de alte aplicații destinate aceluiași obiectiv, această aplicație oferă un plan alimentar întins pe 8 săptămâni, cu mese și gustări balansate nutritional, rețetele fiind sugerate în funcție de câte calorii arzi în ziua respectivă. Tactica abordată de această aplicație are ca scop schimbarea obiceiurilor alimentare. În plus, oferă posibilitatea de conectare cu o comunitate de utilizatori și pune la dispoziție videoclipuri motivaționale cu povestiri și experiențe de success. În cadrul acesteia există și o bază de date cu exerciții fizice pentru a urmări kaloriile arse.[4]

2. Tehnologiile folosite

2.1 Introducere in tehnologii

Pentru dezvoltarea aplicației web, am utilizat trei tehnologii esențiale cu scopul de a oferi o aplicație completă și eficientă. În acest capitol voi prezenta principalele tehnologii folosite în cadrul proiectului. Fiecare dintre acestea are un rol unic și important în arhitectura aplicației.

Componenta server, realizată cu ajutorul Node.js, Componenta client, realizată cu ajutorul React, și componenta bază de date, realizată cu ajutorul MySQL.

2.2 Tehnologii relevante

2.2.1 MySQL

MySQL este un sistem de gestiune a bazelor de date relaționale, foarte des utilizat în dezvoltarea de aplicații web ce necesită stocarea structurată și eficientă a datelor.

Acesta utilizează limbajul SQL pentru a face interogări și a manipula datele, dar și permite lucrul cu tabele, precum crearea și ștergerea acestora, inserarea și actualizarea entităților introduse, și efectuarea de operații complexe de agregare a datelor.

2.2.2 React

React este o librărie open-source a limbajului JavaScript, ce are ca principal scop, dezvoltarea interfețelor utilizator ale aplicațiilor web.

Printre multiplele caracteristici și avantaje ale React, se numără: posibilitatea de a folosi componente reutilizabile, și Virtual DOM, ce permite React să eficientizeze actualizările interfeței prin actualizarea separată a elementelor modificate. [5]

2.2.3 Node.js

Node.js este un mediu open-source de execuție, ce permite rularea codului JavaScript nu doar în browser, ci și pe componenta server a unei aplicații web.

Printre multiplele caracteristici și avantaje, se numără: faptul că Node.js utilizează limbajul destinat pentru componenta client, JavaScript, și pe partea de backend, lucru ce facilitează dezvoltarea de aplicații web într-un singur limbaj și permite distribuirea de cod între server și client.[6]

2.3 Aplicații relevante

Pentru scrierea și centralizarea codului am folosit aplicația Visual Studio Code, alături de o multitudine de extensii ce facilitează clarificarea liniilor de cod. Visual Studio Code este un editor de cod dezvoltat de Microsoft ce oferă o gama largă de funcționalități.

Pentru testarea cererilor din backend am folosit Insomnia, o aplicație folosită pentru testarea și depanarea aplicațiilor web și a serviciilor API. Aceasta permite gestionarea și organizarea cererilor, dar și vizualizarea răspunsurilor primite de la server.

Pentru gestionarea elementelor din baza de date, am folosit MySQL Workbench, o aplicație ce oferă un mediu de dezvoltare pentru administrarea bazelor de date, fiind o unealtă cu o interfață simplificată.

3. Preliminarii

3.1 Necesarul caloric

3.1.1 Descriere

Necesarul caloric zilnic reprezintă numărul de calorii pe care un individ ce dorește să își mențină greutatea constantă trebuie să îl consume în fiecare zi. În funcție de obiectivele personale, acesta poate fi personalizat atât pentru slăbire, prin scăderea necesarului cu aproximativ 10%, dar și pentru creștere în masă prin adăugarea unui procent de 10% la necesarul caloric zilnic pentru menținere.

Acesta este unic pentru fiecare persoană în parte și depinde de o mulțime de factori precum: tipul metabolismului, vârsta, înălțimea, greutatea, sexul, și frecvența activităților fizice.

Un aspect foarte important este acela că necesarul caloric zilnic al unui individ se modifică în mod constant, în funcție de creșterea sau scăderea în greutate și de frecvența activității fizice practicate de acesta. Astfel trebuie calculat periodic pentru a menține parcursul spre rezultatul dorit.

3.1.2 Formula de calcul

Conform paginii web a ministerului sănătății Emiratelor Arabe Unite, calculul numărului de calorii necesare unui individ zilnic, depinde de două variabile: rata metabolică bazală și factorul de activitate fizică. [7]

În primul rând, rata metabolică bazală (BMR) este o variabilă ce depinde în mare parte de sex-ul individului, și pleacă de la +5 pentru bărbați, iar pentru femei de la -161, la care se adună de zece ori greutatea, exprimată în kilograme, apoi, de 6,25 ori înălțimea, exprimată în centimetri, iar apoi se scade de 5 ori vârsta persoanei. În final crește cu un procent de 5% pentru persoanele ectomorfe, scade cu un procent de 5% pentru persoanele endomorfe, și rămâne constantă pentru persoanele mezomorfe.

În al doilea rând factorul de activitate fizică este reprezentat de un număr real între 1,2 și 1,9, și este direct proporțional cu frecvența practicării de activități fizice. Astfel, pentru o persoană sedentară acesta va fi minim, iar pentru o persoană care, fie practică sporturi de performanță, fie are un stil de viață foarte activ, acesta va fi maxim. Această variabilă nu depinde numai de activitățile fizice de tip sport, ci se referă pe larg la stilul de viață activă al unei persoane.

În final, se înmultesc cele două variabile prezentate anterior, iar rezultatul este exact necesarul caloric zilnic al unei persoane.

3.2 Implementare

Pentru a calcula necesarul caloric am folosit datele introduse de utilizator in formularul de creare a profilului. Varsta, greutatea si inaltimea sunt campuri cu alegere liber. In schimb celelalte campuri fie au variante de raspuns predefinite, fie restrictii, de exemplu:

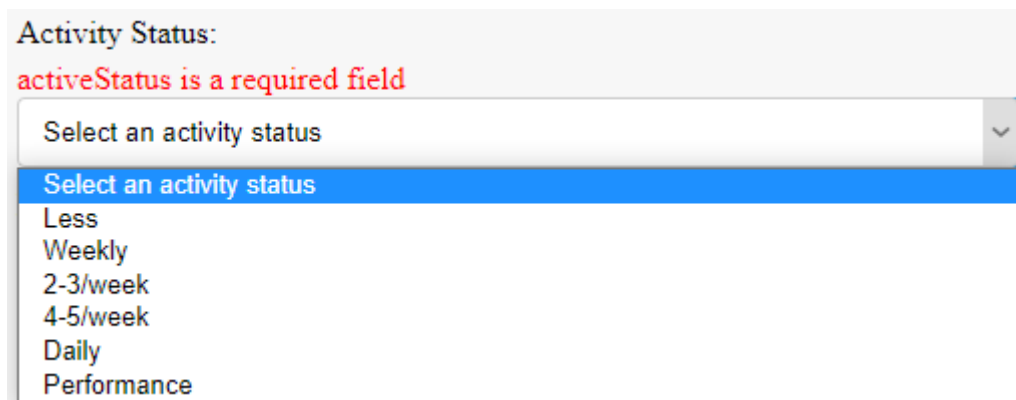


Fig 3.2.1 Variante predefinite de completare

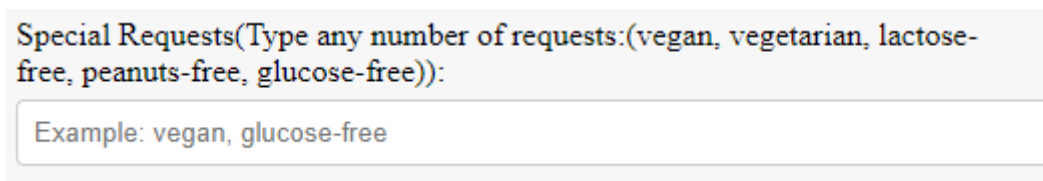


Fig 3.2.2 Variante sugerate de completare

Aceste date sunt transmise de la componenta client catre server, in ruta “Profiles”, operatia post, ce permite crearea unui profil si asocierea acestuia unui utilizator. Codul pentru calculul necesarului caloric pe baza datelor utilizatorului urmeaza strict formula de calcul prezentata.

```
let BMR = 0;

if (profile.gender === "male") {
    BMR += 10* profile.weight + 6.25 * profile.height - 5* profile.age + 5;
} else {
    BMR += 10* profile.weight + 6.25 * profile.height - 5* profile.age - 161;}

switch (profile.metabolismType) {
```

```
    case "endomorph":
        BMR *= 0.95;
        break;
    case "mezomorph":
        BMR *= 1;
        break;
    case "ectomorph":
        BMR *= 1.05;
        break;}

let activityScore = 0;
switch (profile.activeStatus) {
    case "less":
        activityScore = 1.2;
        break;
    case "weekly":
        activityScore = 1.27;
        break;
    case "2-3/week":
        activityScore = 1.375;
        break;
    case "4-5/week":
        activityScore = 1.55;
        break;
    case "daily":
        activityScore = 1.725;
        break;
    case "performance":
        activityScore = 1.9;
        break;
}
```

4. Arhitectura si design

4.1 Arhitectura generala

4.1.1 Descrierea arhitecturii generale

Aplicația a fost dezvoltată utilizând o arhitectură bazată pe tehnologiile antementionate în capitolul 2. Această arhitectură presupune interacțiunea componentelor de frontend și backend, precum și gestionarea datelor și a informațiilor cu ajutorul unei baze de date.

4.1.2 Componente principale

Arhitectura aplicației este construită în jurul următoarelor componente principale:

Componenta client, sau componenta frontend, realizată folosind biblioteca React. Aceasta reprezintă interfața pe care utilizatorul o are în fața și cu ajutorul căreia acesta interacționează cu aplicația.

Componenta server, sau componenta backend, realizată cu ajutorul framework-ului Node.js. Aceasta procesează solicitările primite de la componenta client și întoarce ca răspuns datele corespunzătoare.

Componenta baza de date, realizată cu ajutorul MySQL, utilizată pentru gestionarea și stocarea datelor, date ce pot fi prelucrate prin intermediul componentei server.

Componenta API, realizată cu ajutorul bibliotecii Axios, utilizată pentru a realiza query-uri între componenta client și componenta server. Aceasta facilitează comunicarea în mod asincron și transferul de informații între componente. [8]

4.2 Impartirea pe rute

Pentru a avea o arhitectura mult mai organizata, am ales sa impart aplicatia in multiple rute, astfel fiecare functionalitate sa aiba cel putin o ruta unica pentru a nu interfera cu celelalte.

4.2.1 Rutele

În continuare voi prezenta fiecare rută a aplicației cu exemple relevante.

```
JS Addingredient.js
JS AddProduct.js
JS AddRecipe.js
JS CreateProfile.js
JS GenerateMealPlan.js
JS Home.js
JS Ingredient.js
JS Ingredients.js
```

Fig 4.2.1.1 Rute 1

```
JS Login.js
JS MealPlan.js
JS MealPlans.js
JS Product.js
JS Products.js
JS Recipe.js
JS Recipes.js
JS Register.js
```

Fig 4.2.1.2 Rute 2

Ruta Home: în care are loc principala funcționalitate a aplicației, mai exact generarea de planuri alimentare personalizate. Aceasta referențiază alte 2 rute asociate planurilor alimentare.

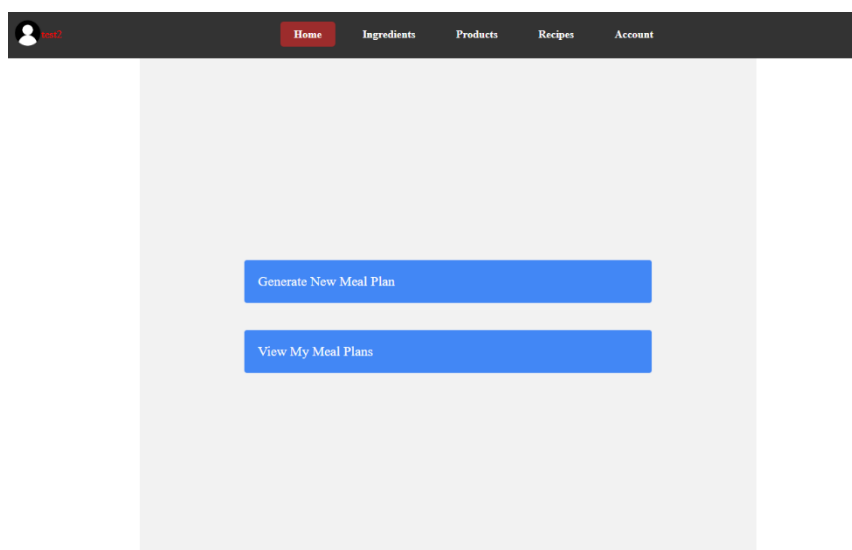


Fig 4.2.1.3 Arhitectura si designul rutei Home

Ruta Ingredients, în care utilizatorul poate vedea informații despre ingrediente. Aceasta referențiază o altă rută ce permite adăugarea de ingrediente

Add Other Ingredient

Chicken Breast

Unit of measurement: 100g
Calories: 165

MEAT

Fig 4.2.1.4 Arhitectura si designul rutei Ingredients

Rutele Products și Recipes, construite în același mod ca ruta Ingredients, în care utilizatorul poate vedea informații despre produse și rețete, având fiecare referințe către rute de adăugare de produse, respectiv rețete noi.

Ruta Login, în care utilizatorul se poate autentifica pentru a își vedea sau genera planurile alimentare personalizate. Aceasta oferă posibilitatea de deconectare, sau navigare către ruta de înregistrare.

Email:

Password:

Login

Not Signed in yet? Create Account

Log Out

Fig 4.2.1.5 Arhitectura si designul rutei Account

Ruta Register, în care un utilizator își poate crea un cont nou. Aceasta oferă posibilitatea navigării către ruta Login în contextul în care utilizatorul este deja înregistrat.

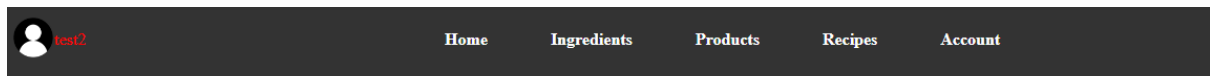
A light grey rectangular form containing two input fields. The first is labeled 'E-mail:' with the placeholder text 'Example: mail@mail.com'. The second is labeled 'Password:' with the placeholder text 'Example: pass1234'. Below these fields are two blue buttons. The top button is labeled 'Register'. The bottom button is labeled 'Already have an account? Sign In'.

Fig 4.2.1.6 Arhitectura si designul rutei Register

Ruta de creare a profilului, ruta în care suntem redirecționați după autentificare, doar în contextul în care profilul nu este deja setat. Aceasta conține un formular, care în urma completării acestuia asociază utilizatorului un profil.

A light grey rectangular form with a dark grey navigation bar at the top. The navigation bar contains a user profile icon and the text 'test2', followed by menu items: 'Home', 'Ingredients', 'Products', 'Recipes', and 'Account'. The form itself contains several input fields and dropdown menus. The fields are labeled: 'Your Name:' (placeholder: 'Example: John'), 'Age:' (placeholder: 'Example: 15'), 'Gender:' (dropdown: 'Select your gender'), 'Height(cm):' (placeholder: 'Example: 170'), 'Weight(kg):' (placeholder: 'Example: 73'), 'Activity Status:' (dropdown: 'Select an activity status'), 'Metabolism Type:' (dropdown: 'Select a metabolism type'), and 'Special Requests(Type any number of requests:(vegan, vegetarian, lactose-free, peanuts-free, glucose-free)):' (placeholder: 'Example: vegan, glucose-free'). At the bottom of the form is a green button labeled 'Create Profile'.

Fig 4.2.1.7 Arhitectura si designul rutei CreateProfile

În exact aceeași manieră, sunt create și rutele de adăugare a ingredientelor, produselor și rețetelor, prin formulare ale căror informații sunt prelucrate și trimise către componenta server pentru a fi adăugate la baza de date.

4.2.2 Navigarea între rute

Pentru a face posibilă navigarea între rutele aplicației am adăugat componenta NavBar, care este o bară de navigare cu 5 butoane, ce facilitează utilizarea eficientă și rapidă a aplicației.

Pe lângă navigare, mai prezintă două caracteristici importante. În primul rând utilizatorul va ști în permanență numele paginii sau al rutei în care se află, butonul corespunzător acestora fiind marcat.

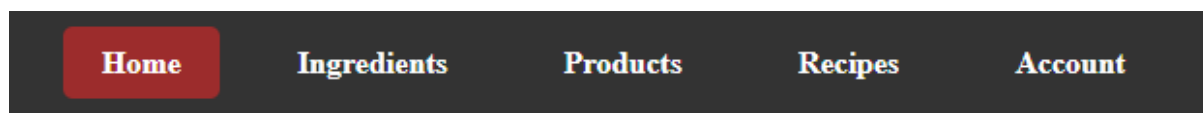


Fig 4.2.2.1 Rutele din bara de navigatie

În al doilea rând, afișarea adresei de e-mail a utilizatorului, doar în contextul în care acesta este autentificat



Fig 4.2.2.2 Afișarea utilizatorului autentificat in bara de navigatie

5. Baza de date

5.1 Structura bazei de date

5.1.1 Tabele

Pentru a obține o logică mult mai clară a schemei generale a aplicației, am folosit următoarele tabele: Ingredients, Products, Recipes, MealPlans, Profiles, Users. În continuare voi descrie, pe rând, tabelele și relațiile dintre acestea.

Tabela Ingredients, pe lângă cheia primară, care este un întreg de tipul index, stochează detalii specifice ingredientelor alimentare. Conține câmpurile: name, category, caloriesPerPortion și portion.

În coloana name se află un șir de caractere reprezentând numele ingredientului descris.

În coloana category se află un șir de caractere, reprezentând un cuvânt din următoarele: meat, dairy, vegetables, carbohydrates, spices, descriind exact tipul ingredientului.

În coloana portion se află un șir de caractere, reprezentând un cuvânt din următoarele: 100g, tablespoon, teaspoon, piece.

În coloana caloriesPerPortion se află un număr întreg, reprezentând numărul de calorii al unei porții din acel ingredient.

Tabela Products, pe lângă cheia primară, care este un întreg de tipul index, stochează detalii specifice produselor alimentare. Conține câmpurile: name, category, quantity, ingredients și totalCalories.

În coloana name se află un șir de caractere reprezentând numele produsului descris.

În coloana category se află un șir de caractere, reprezentând un cuvânt din următoarele: meat, dairy, vegetables, carbohydrates, spices, descriind exact tipul produsului.

În coloana quantity se află un număr real, reprezentând exact cantitatea produsului descris, în kilograme.

În coloana ingredients se află un dicționar, în care cheile sunt reprezentate de șiruri de caractere descriind numele ingredientelor din tabelul Ingredients, iar valorile sunt reprezentate de numere reale, descriind cantitatea din fiecare ingredient pe care produsul îl conține în componența sa.

În coloana totalCalories se află un număr întreg, reprezentând totalul caloric pe care produsul îl are.

Tabela Recipes, pe lângă cheia primară, care este un întreg de tipul index, stochează detalii specifice rețetelor alimentare. Conține câmpurile: name, text, category, quantity, ingredients, products și totalCalories.

În coloana name se află un șir de caractere reprezentând numele rețetei descrise.

În coloana text se află un text reprezentând toți pașii necesari pentru realizarea rețetei descrise.

În coloana category se află un șir de caractere, reprezentând un cuvânt din următoarele: vegan, vegetarian, lactose-free, peanuts-free, glucose-free, dar și cuvântul vid, descriind exact tipul rețetei.

În coloana quantity se află un număr real, reprezentând exact cantitatea rețetei descrise, în kilograme.

În coloana ingredients se află un dicționar, în care cheile sunt reprezentate de șiruri de caractere descriind numele ingredientelor din tabelul Ingredients, iar valorile sunt reprezentate de numere reale, descriind cantitatea din fiecare ingredient pe care rețeta îl conține în componența sa.

În coloana products se află un dicționar, în care cheile sunt reprezentate de șiruri de caractere descriind numele produselor din tabelul Products, iar valorile sunt reprezentate de numere reale, descriind cantitatea din fiecare produs pe care rețeta îl conține în componența sa.

În coloana totalCalories se află un număr întreg, reprezentând totalul caloric pe care rețeta îl are.

Tabela MealPlans, pe lângă cheia primară, care este un întreg de tipul index, stochează detalii specifice planurilor alimentare. Conține câmpurile: name, type, recipes.

În coloana name se află un șir de caractere reprezentând numele planului alimentar descris.

În coloana type se află un șir de caractere, reprezentând un cuvânt din următoarele: cut, maintain, bulk, descriind scopul pe care planul alimentar îl are.

În coloana recipes se află un array, în care elementele sunt reprezentate de obiecte din tabelul Recipes, descriind rețetele ce intră în componența planului alimentar.

Tabela Users, pe lângă cheia primară, care este un întreg de tipul index, stochează detalii specifice despre contul unui utilizator. Conține câmpurile email și passwordÎn coloana email se

află un șir de caractere, reprezentând adresa de email pe care utilizatorul a folosit-o la crearea contului.

În coloana password se află un șir de caractere, reprezentând parola pe care utilizatorul a folosit-o la crearea contului.

Tabela Profiles, pe lângă cheia primară, care este un întreg de tipul index, stochează, pentru un utilizator, datele necesare aplicării formulei de calcul al necesarului caloric zilnic. Conține câmpurile name, age, gender, height, weight, activeStatus, metabolismType, constraints, score.

În coloana name se află un șir de caractere reprezentând numele utilizatorului.

În coloana age se află un număr întreg, reprezentând vârsta utilizatorului exprimată în ani.

În coloana gender se află un șir de caractere, reprezentând un cuvânt din următoarele: male, female, descriind genul unui utilizator.

În coloana height se află un număr întreg, reprezentând înălțimea, exprimată în centimetri, a utilizatorului.

În coloana weight, se află un număr întreg, reprezentând greutatea, exprimată în kilograme, a utilizatorului.

În coloana activeStatus, se află un șir de caractere, reprezentând o structură de cifre și cuvinte din următoarele: performance, daily, 2-3/week, 4-5/day, weekly, less, descriind frecvența activității fizice practicate de utilizator.

În coloana metabolismType se află un șir de caractere, reprezentând un cuvânt din următoarele: endomorph, mesomorph, ectomorph, descriind tipul de metabolism al utilizatorului.

În coloana constraints se află un șir de caractere, reprezentând unul sau mai multe cuvinte, descriind restricții alimentare. De exemplu: lactose-free.

În coloana score se află un număr real, ce descrie numărul de calorii pe care utilizatorul trebuie să îl consume zilnic pentru a își menține greutatea.

5.1.2 Constrangeri

În acest capitol voi prezenta o parte din constrângerile utilizate pentru câmpurile tabelor din modelul prezentat anterior, exemple și explicații ale folosirii acestora, și repartizarea pe coloane și tabele.

Pentru a mă asigura de unicitatea și imposibilitatea de a rămâne necompletat a unui câmp, am folosit proprietățile “nenul” și “unic”, reprezentate în cod astfel:

```
allowNull: false, unique: true,
```

Pentru a trata cazul în care un câmp este gol, am adăugat validarea “completat”, reprezentată în cod astfel:

```
validate: {notEmpty: { msg: 'Calories per portion is required' },},
```

Pentru a valida apartenența informației reținute de un câmp la o listă de posibilități acceptate am utilizat validarea “apartine”, reprezentată în cod astfel:

```
validate: {isIn: {args: [['100g', 'tablespoon', 'teaspoon', 'piece']],  
msg: 'Invalid portion',},},
```

Anunțarea încălcării validărilor este anunțată prin mesaje de eroare.

La nivelul tabelii Ingredients, există următoarele constrângeri: name-nenul, unic, completat, category-nenul, completat, aparține, caloriesPerPortion- nenul, completat, portion-nenul, completat, aparține.

La nivelul tabelii Products, există următoarele constrângeri: name-nenul, unic, completat, category- aparține, quantity- nenul, completat, ingredients- nenul, totalCalories-nenul, completat.

La nivelul tabelii Recipes, există următoarele constrângeri: name-nenul, unic, completat, text- nenul, category-nenul, aparține, quantity- nenul, completat, ingredients- nenul, totalCalories- nenul, completat.

La nivelul tabelii MealPlans, există următoarele constrângeri: name-nenul, completat, type- nenul, completat, aparține,.

La nivelul tabelii Users, există următoarele constrângeri: email- nenul, unic, completat, password- nenul, completat.

La nivelul tabelii Profiles, există următoarele constrângeri: name- nenul,completat, age-nenul, completat, gender- nenul, completat, aparține, height- nenul, completat, weight-nenul, completat, activeStatus-nenul, completat, aparține, metabolismType: nenul, completat, aparține.

5.1.3 Relații

Pentru ca aplicația să funcționeze optim, pentru a reduce redundanța datelor, și pentru a simplifica interogările complexe, sunt necesare relațiile între tabele. Acestea permit, de asemenea, o organizare structurată.

Am practicat două tipuri de abordări pentru crearea relațiilor. În primul rând am folosit asocierile puse la dispoziție de către ORM-ul Sequelizee. [9]

Relația One-to-One se realizează prin definirea unei asocieri între două tabele, folosind metoda “hasOne”. De exemplu, relația care permite asocierea unui singur utilizator la un singur profil, este reprezentată în cod în următorul mod:

```
Users.associate = (models) => {  
  Users.hasOne(models.Profiles, {  
    onDelete: "cascade",  
  });  
};
```

Relația One-to-Many, realizată similar relației anterioară, folosind metoda “hasMany”. De exemplu, relația ce permite asocierea mai multor planuri alimentare unui singur utilizator:

```
Users.associate = (models) => {  
  Users.hasMany(models.MealPlans, {  
    onDelete: "cascade",  
  });  
}
```

Constrângerea “onDelete: “cascade” presupune ștergerea unui element atunci când elementul asociat acestuia este șters. Ca exemplu, dacă ștergem un utilizator din baza de date, profilul și planurile alimentare asociate acestuia vor fi șterse și ele.

În al 2-lea rând, am simulat relațiile folosind structuri de date, mai exact dicționare și array-uri încastate în tipul de date JSON. Astfel, o relație One-to-Many presupune adăugarea unei coloane într-un tabel care reține elemente de tipul JSON, fie obiecte mapate că șiruri de caractere, fie obiecte în formă JSON.

Exemplu de câmp care reține tipul de date antementionat:

```
ingredients: {type: DataTypes.JSON, allowNull: false,},
```

Acest câmp este folosit pentru a reține în tabelul Recipes un dicționar cu elemente de forma “cheie”: “valoare”, unde cheia este numele unui element din tabelul Ingredients, iar valoarea este un număr real reprezentant cantitatea.

5.2 Interacțiunea cu baza de date

5.2.1 Conexiune

Conexiunea la baza de date MySQL folosind un Object-Relational Mapping destinat pentru Node.js. Sequelize facilitează interacțiunea simplificată cu baza de date.

Pentru a crea conexiunea, avem nevoie de modulele Sequelize, Express și Cors. Aceste trei module, oferă, un mediu eficient pentru dezvoltarea de aplicații web care necesită interacțiune cu o bază de date.

Modulul Sequelize este folosit pentru a putea crea o conexiune cu baza de date și pentru manipularea datelor din aceasta. Modulul Express este folosit pentru crearea serverului web și pentru a gestiona rutele. Nu în ultimul rând, modulul Cors este folosit pentru configurarea politicilor de securitate cross-origin.

Dupa importarea modulelor, se creează o instanță, adăugând și middleware-ul necesar, pentru a configura serverul Express. Middleware-ul “express.json()” permite interpretarea datelor sub formă de json-uri trimise de către client, iar “cors()” permite comunicarea între diverse origini.

Conectarea efectivă la baza de date, are loc folosind modulul Sequelize. În primă instanță, se atribuie datele de conectare, numele bazei de date, numele utilizatorului și parola. Dupa care, se creează o instanță Sequelize și folosind datele de conectare se realizează conexiunea cu baza de date

Pentru a avea o structură a codului cât mai organizată, pentru fiecare tabelă creez o rută, pentru manipularea datelor din aceasta. În fiecare rută am folosit o funcție de controller, care, utilizând metodele CRUD din Sequelize, permite efectuarea de operații asupra bazei de date.

Finalizarea conexiunii se realizează o dată cu pornirea serverului Express pe un anumit port setat, iar conexiunea serverului cu baza de date este sincronizată, fapt obținut din apelul funcției “sequelize.sync()”.

Urmând toți pașii enumerați anterior, am putut crea o aplicație web funcțională, care permite următorul flux: Cererile trimise de către componenta client sau frontend primesc răspunsurile aferente, prin interacțiunea serverului cu baza de date și prelucrarea informațiilor din aceasta.

5.2.2 CRUD

În primul rând, ce este o operație CRUD? O operație CRUD (Create, Read, Update, Delete) reprezintă o simplă acțiune ce poate fi efectuată asupra informațiilor din baza de date. Această permite manipularea datelor și asigură funcționalitatea aplicației. Urmează să prezint succint fiecare tip de operație CRUD, adăugând și exemple din cod.

Operația Create. Această operație permite crearea unei noi entități și adăugarea de elemente noi în baza de date. În componenta server, datele primite de la client sunt prelucrate, validate, și apoi încastrate într-un obiect folosind ORM-ul Sequelize.

Ca exemplu, am ales operația de creare a unei noi rețete. Datele trimise de componenta client sunt prelucrate astfel încât să se calculeze suma caloriilor, dată de numărul de porții din fiecare ingredient sau produs.

```
router.post("/", async (req, res) => {
  const recipe = req.body;
  const listOfIngredients = await Ingredients.findAll();
  const listOfProducts = await Products.findAll();
  let sumOfCalories = 0;
  listOfIngredients.forEach((ingredient) => {
    const name = String(ingredient.name);
    if (name in recipe.ingredients) {
      const quantity = parseFloat(recipe.ingredients[name]);
      const caloriesPerPortion = ingredient.caloriesPerPortion;
      sumOfCalories += quantity * caloriesPerPortion; });
  listOfProducts.forEach((product) => {
    const name = String(product.name);
    if (name in recipe.products) {
      const quantity = parseFloat(recipe.products[name]);
      const caloriesPerPortion = product.totalCalories;
      sumOfCalories += quantity * caloriesPerPortion; });
  recipe.totalCalories = sumOfCalories
  await Recipes.create(recipe);
  res.json(recipe.totalCalories);
});
```

Operația Read permite obținerea datelor din baza de date, considerând anumite filtre trimise de către componenta client. Pentru a extrage informațiile din baza de date, se utilizează interogări de tipul “get”, urmând ca datele extrase să fie returnate către front-end pentru a fi utilizate și afișate în interfață utilizator.

Ca exemplu am ales returnarea ingredientelor folosite într-o rețetă, pe baza unui identificator trimis de către client. Am aplicat 2 filtre de tipul “where”, unul pentru a obține datele despre o rețetă pe baza id-ului, iar al doilea pentru a extrage doar ingredientele folosite în rețetă din lista cu toate ingredientele.

```
router.get('/getIngredientsFromRecipe/:id', async (req, res) => {
  const recipeId = req.params.id;
  const recipe = await Recipes.findOne({
    where: { id : recipeId } });
  const usedIngredients = Object.keys(recipe.ingredients);
  const listOfIngredients = await Ingredients.findAll({
```

```
where: { name: { [Op.in]: usedIngredients } }));  
res.json(listOfIngredients));
```

Operația Update, presupune actualizarea datelor unei entități aflate în baza de date. Serverul primește de la client un identificator și un set de date pe care se dorește să le actualizăm. Se extrage entitatea referențiată de identificator, se schimbă datele cu cele noi, iar apoi și câmpurile care depind de acestea, după care se salvează obiectul.

Un exemplu îl constituie actualizarea datelor din profilul utilizatorului. În această aplicație se pot schimbă: vârstă, greutatea și frecvența activităților fizice, pentru a genera planuri nutriționale cât mai personalizate. Câmpul score este calculat automat în funcție de datele primite de la client pentru actualizare.

```
router.put('/:userId', async (req,res) => {  
  const userId = req.params.userId;  
  const {age, weight, activeStatus} = req.body;  
  const profile = await Profiles.findOne({ where: { UserId : userId } });  
  profile.age = age;  
  profile.weight = weight;  
  profile.activeStatus = activeStatus;  
  let BMR = 0;  
  if (profile.gender === "male") {  
    BMR += 10* profile.weight + 6.25 * profile.height - 5* profile.age + 5;  
  } else {  
    BMR += 10* profile.weight + 6.25 * profile.height - 5* profile.age - 161;  
  }  
  switch (profile.metabolismType) {  
    case "endomorph":  
      BMR *= 0.95;  
      break;  
    case "mezomorph":  
      BMR *= 1;  
      break;  
    case "ectomorph":  
      BMR *= 1.05;  
      break;}  
  let activityScore = 0;  
  switch (profile.activeStatus) {  
    case "less":  
      activityScore = 1.2;  
      break;  
    case "weekly":  
      activityScore = 1.27;  
      break;  
    case "2-3/week":  
      activityScore = 1.375;  
      break;
```

```

    case "4-5/week":
      activityScore = 1.55;
      break;
    case "daily":
      activityScore = 1.725;
      break;
    case "performance":
      activityScore = 1.9;
      break; }
profile.score = parseFloat(BMR * activityScore) ;
await profile.save();
res.json(profile);});

```

Operația Delete permite ștergerea unei entități din baza de date. Clientul trimite un identificator al entității pe care o dorește eliminată, apoi după identificarea entității, aceasta va fi distrusă.

De exemplu, operația de ștergere a unui produs din baza de date, referențiat prin numele trimis de către client. Întâi se aplică o filtrare pentru a găsi entitatea, iar apoi este ștearsă din tabelă.

```

router.delete("/delete/:name", async (req, res) => {
  const productName = req.params.name;

  const product = await Products.findOne({
    where: { name : productName }
  });
  await product.destroy();
  res.json("Delete product: "+ productName+ " Worked");
});

```

6. Funcționalități

6.1 Introducere in funcționalități

Acest subcapitol oferă o vedere de ansamblu a aplicației web și a obiectivelor funcționale pe care acestea trebuie să le îndeplinească. Aplicația urmărește atingerea unor scopuri cheie când vine vorba de sănătatea unei persoane, cum ar fi echilibrul alimentar și traseul către forma fizică dorită de către utilizator.

6.1.1 Funcționalitate cheie

Aplicația își propune să ajute utilizatorii să creeze planuri alimentare, fără a necesita o implicare activă a acestora. Astfel, cum am prezentat anterior, aplicația oferă rețete detaliate, care conțin atât produse și ingrediente, cât și detalii despre cantitatea în care acestea sunt adăugate, și numărul de calorii al fiecăruia.

Planurile alimentare sunt concepute considerând nevoile fiecărui utilizator.

Tot ce trebuie să facă un utilizator de la deschiderea aplicației până la utilizarea planurilor alimentare generate de acesta se află la o distanță de doar câteva click-uri.

Fiecare plan alimentar conține trei rețete. Suma kaloriilor fiecărei porții dintr-o rețetă se încadrează exact în necesarul caloric pe care utilizatorul trebuie să îl consume zilnic pentru a își menține greutatea corporală actuală.

6.1.2 Descriere detaliată si exemple

Următoarea imagine prezintă o vedere de ansamblu asupra funcționalității cheie a aplicației web.

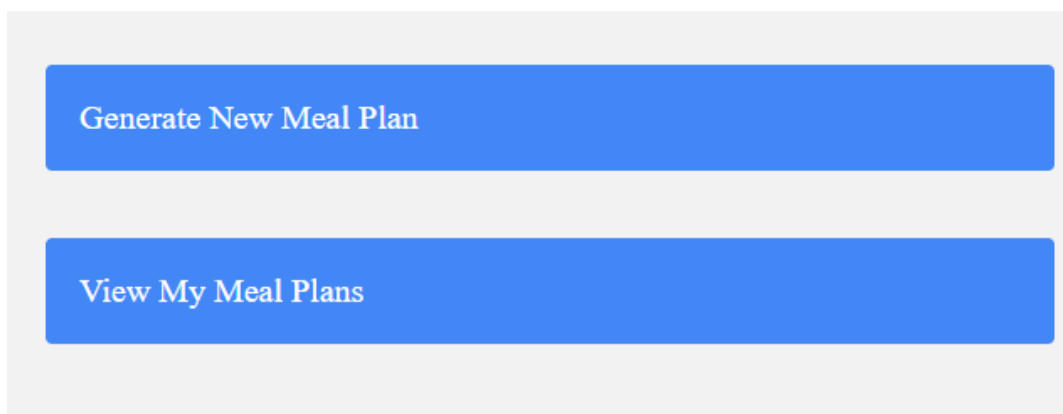
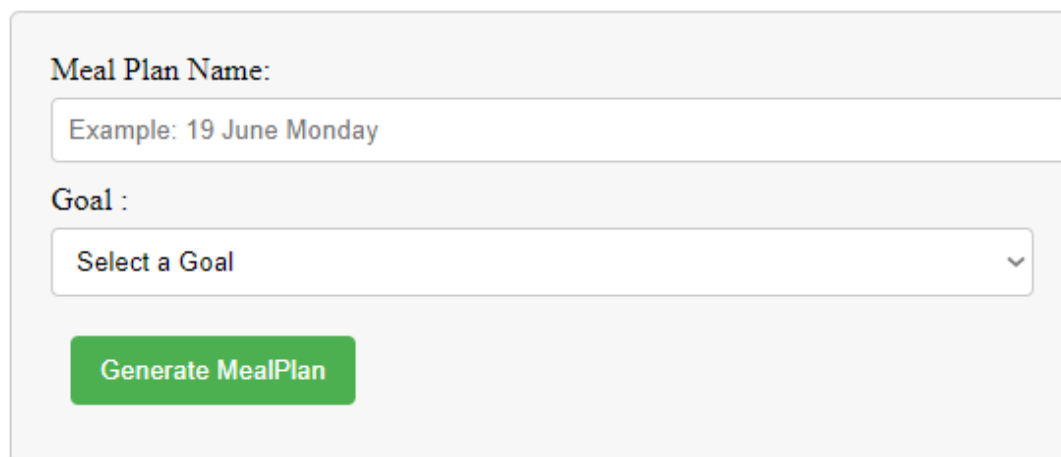


Fig 6.1.2.1 Funcționalitatea principală

Funcționalitatea cheie implică două subfuncționalități strâns legate de această. Prima, generarea unui nou plan alimentar, poate fi accesată prin apăsarea butonului “Generate New Meal Plan”

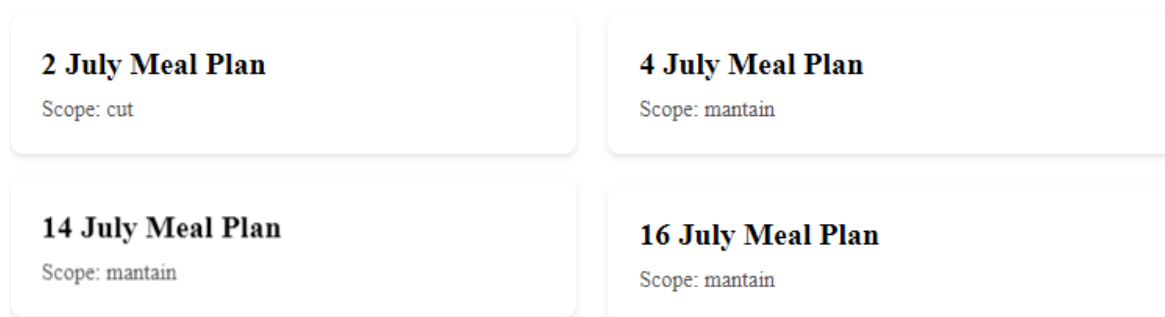
La apăsarea butonului suntem redirecționați către o altă ruta, ce conține un mic formular, ce așteaptă informații despre numele planului alimentar, și goal-ul acestuia, care poate fi: Loose Weight – pierdere în greutate, Maintain – menținere a greutății, Gain Weight – creștere în greutate.



The image shows a web form for generating meal plans. It has a light gray background. At the top, it says "Meal Plan Name:" followed by a text input field containing "Example: 19 June Monday". Below that, it says "Goal :" followed by a dropdown menu with "Select a Goal" and a downward arrow. At the bottom, there is a green button with the text "Generate MealPlan".

Fig 6.1.2.2 Formularul pentru generarea planurilor alimentare

După completarea formularului și apăsarea butonului de generare, utilizatorul este redirecționat către cea de a 2-a subfuncționalitate, ce poate fi accesată și prin apăsarea butonului “View My Meal Plans”. În această pagină sunt afișate toate planurile alimentare generate de utilizator, sub forma unei liste.



The image shows a list of meal plans for a user. There are four cards arranged in a 2x2 grid. Each card has a title and a subtitle. The first card is titled "2 July Meal Plan" with subtitle "Scope: cut". The second card is titled "4 July Meal Plan" with subtitle "Scope: mantain". The third card is titled "14 July Meal Plan" with subtitle "Scope: mantain". The fourth card is titled "16 July Meal Plan" with subtitle "Scope: mantain".

Fig 6.1.2.3 Afișarea planurilor alimentare ale unui utilizator

Pentru a putea vedea detaliile despre un plan alimentar, mai exact rețetele generate de acesta, utilizatorul trebuie să selecteze prin click planul pe care dorește să îl expandeze. Practic va fi redirecționat către o rută nouă ce conține detalii despre selecția făcută.

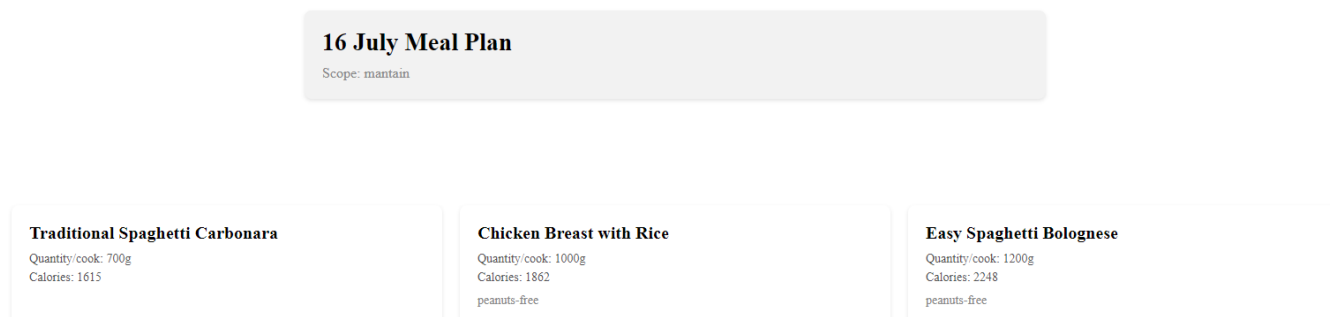


Fig 6.1.2.4 Detaliile despre un plan alimentar

Întrucât în această pagină sunt prezentate doar detalii succinte despre fiecare rețetă, cum ar fi numele, cantitatea, numărul de calorii și restricțiile, putem selecta o rețetă și să aflăm toate detaliile despre aceasta:

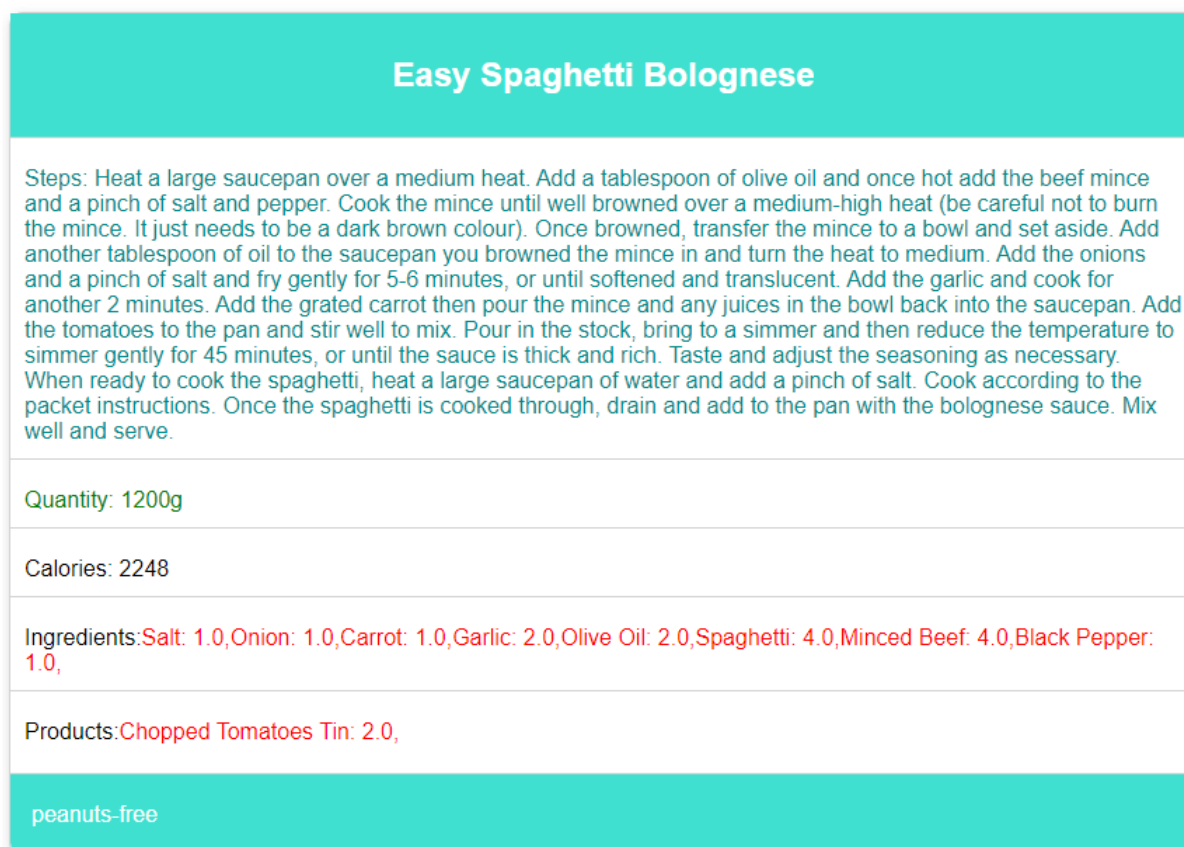


Fig 6.1.2.5 Afisarea unei retete in aplicatie

Precum și ingredientele și produsele utilizate în componența sa, cu detalii succinte despre acestea, de exemplu: cantitatea, unitatea de măsură și numărul de calorii al fiecăruia.

The figure displays four distinct cards for ingredients and products. Each card has a red header with the item name, a light gray body with details, and a red footer with a category label.

- Black Pepper:** Header 'Black Pepper', body 'Unit of measurement: teaspoon' and 'Calories: 3', footer 'SPICES'.
- Chopped Tomatoes Tin:** Header 'Chopped Tomatoes Tin', body 'Package quantity: 0.4' and 'Calories: 80', footer 'VEGETABLES'.
- Carrot:** Header 'Carrot', body 'Unit of measurement: 100g' and 'Calories: 41', footer 'VEGETABLES'.
- Category Card:** A red card with the label 'VEGETABLES'.

Fig 6.1.2.6 Ingredientele și produsele dintr-o rețetă

6.2 Funcționalități auxiliare

Pe lângă funcționalitatea principală a aplicației, se enumeră o mulțime de obiective funcționale, și anume: crearea unui cont de utilizator, persistența datelor prin asocierea unui utilizator cu un singur profil, vizualizarea de ingrediente, produse și rețete din baza de date, adăugarea de ingrediente și produse noi, crearea de rețete folosind produsele și ingredientele preferate, calculul necesarului caloric zilnic prin intermediul datelor introduse de utilizator, și, nu în ultimul rând generarea aleatorie a rețetelor dintr-un plan alimentar.

Pentru ca un utilizator să își creeze un cont în aplicație este necesar să introducă o adresă de e-mail ce nu a mai fost folosită în aplicație, și o parolă de minim opt caractere.

The login form consists of two input fields: 'Email:' and 'Password:'. Below the 'Password:' field are three buttons: 'Login', 'Not Signed in yet? Create Account', and 'Log Out'.

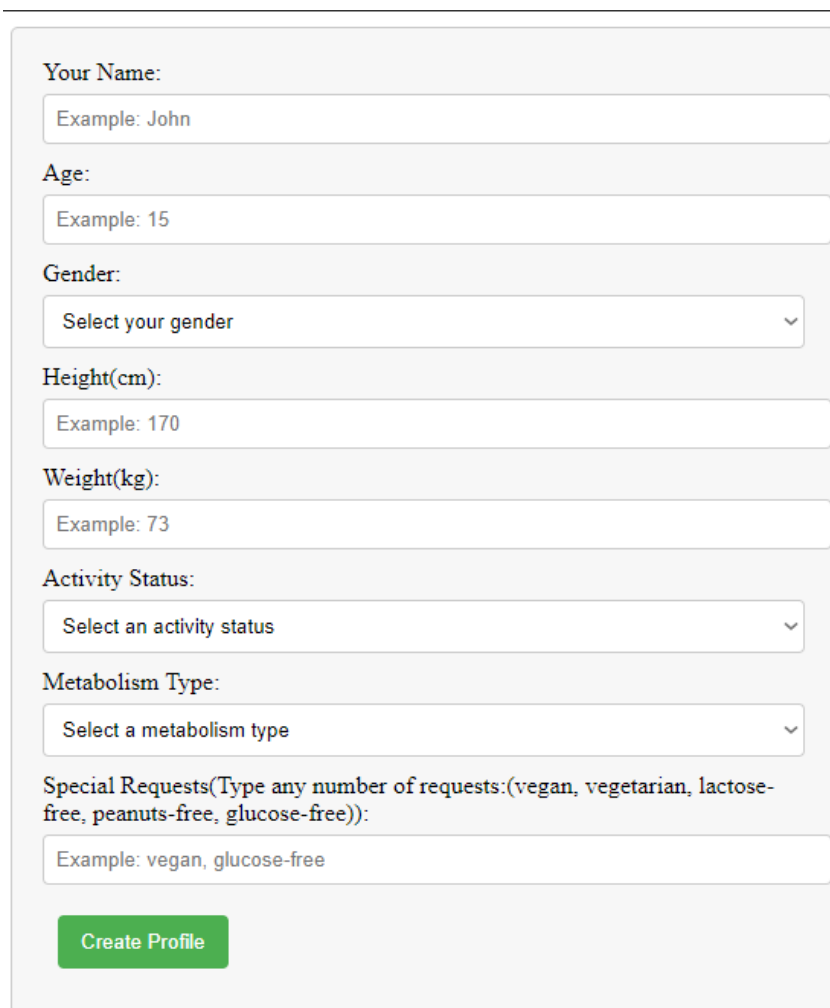
Fig 6.2.1 Formularul de autentificare

The registration form features two input fields: 'E-mail:' and 'Password:'. Below the 'Password:' field are two buttons: 'Register' and 'Already have an account? Sign In'.

Fig 6.2.2 Formularul de înregistrare

Dacă există în baza de date un utilizator cu adresa de e-mail respectivă, înseamnă că utilizatorul are deja un cont creat, și poate sări la pasul de autentificare. Pas în care I se cer aceleași două date despre cont.

Considerând faptul că un utilizator și-a creat contul, dar încă nu are un profil setat, acesta va fi redirecționat către pagina de creare a profilului. Pagină ce conține un form cu date personale despre corpul, metabolismul și frecvența cu care acesta practică activități fizice.



The image shows a web form for creating a profile. It is titled 'Your Name:' and contains several input fields and dropdown menus. The fields are: 'Your Name:' with an example 'John', 'Age:' with an example '15', 'Gender:' with a dropdown 'Select your gender', 'Height(cm):' with an example '170', 'Weight(kg):' with an example '73', 'Activity Status:' with a dropdown 'Select an activity status', 'Metabolism Type:' with a dropdown 'Select a metabolism type', and 'Special Requests(Type any number of requests:(vegan, vegetarian, lactose-free, peanuts-free, glucose-free)):' with an example 'vegan, glucose-free'. At the bottom is a green button labeled 'Create Profile'.

Fig 6.2.3 Formularul pentru setarea profilului

O altă funcționalitate ce se află în strânsă relație cu setarea profilului, este calcularea necesarului caloric al utilizatorului în funcție de variabilele introduse de acesta. Pentru acest calcul se utilizează o formulă științifică

După setarea profilului, la fiecare autentificare utilizatorul va fi redirecționat către ruta în care se regăsește funcționalitatea principală.

Utilizatorul poate vedea oricând, indiferent dacă este autentificat sau nu, ingredientele, produsele și rețetele existente în baza de date. Paginile Ingredients și Products afișează și butoane pentru adăugarea de ingrediente.



Fig 6.2.4 Afișarea și adăugarea produselor

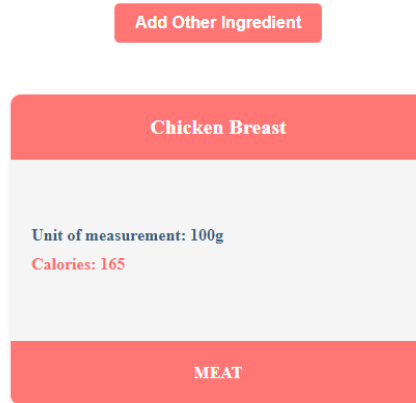


Fig 6.2.5 Afișarea și adăugarea ingredientelor

Pentru fiecare produs din listă, se pot afișa detalii și ingredientele folosite în componența acestuia. La o singură selectare a produsului dorit utilizatorul este redirecționat către pagina produsului.

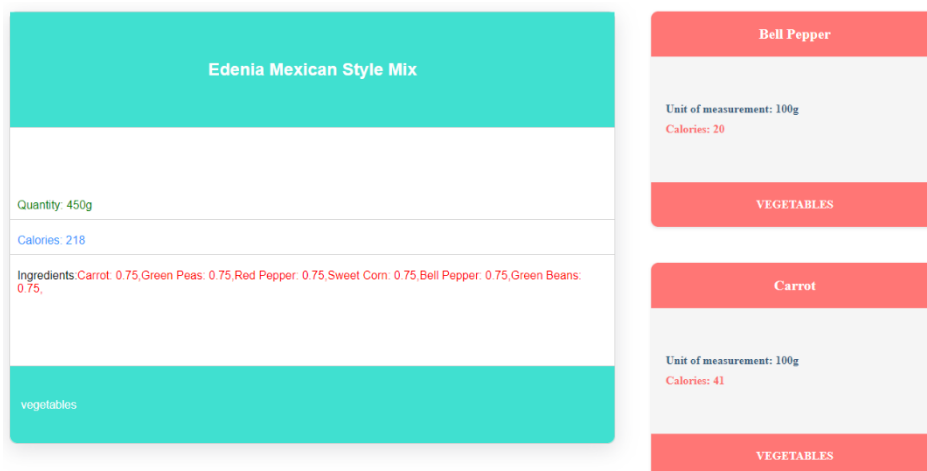


Fig 6.2.6 Afișarea unui produs cu ingredientele din componența sa

În același mod sunt afișate și rețetele, existând și posibilitatea de a crea o rețetă nouă cu ajutorul ingredientelor și produselor deja existente.

Pentru adăugarea în baza de date a unui obiect de tip ingredient, produs sau rețetă am realizat formulare, în care utilizatorul introduce datele despre fiecare element pe care dorește să îl adauge sau creeze.

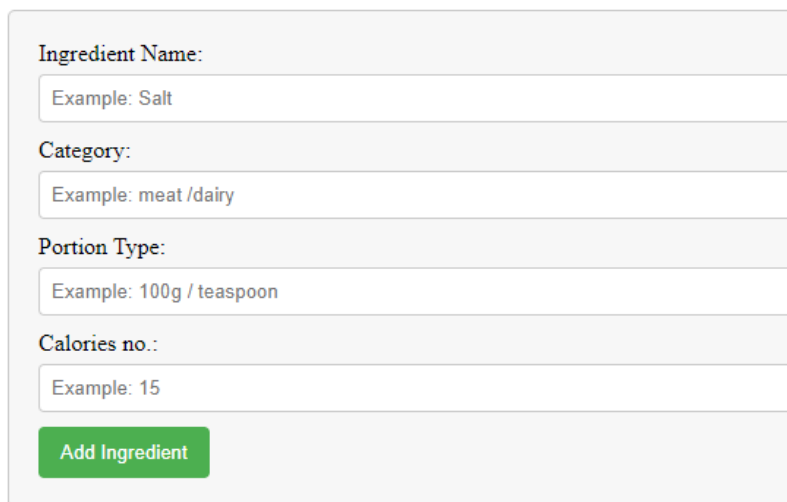
A screenshot of a web form for adding an ingredient. It contains four text input fields with placeholder text: 'Ingredient Name:' (Example: Salt), 'Category:' (Example: meat /dairy), 'Portion Type:' (Example: 100g / teaspoon), and 'Calories no.:' (Example: 15). Below the fields is a green button labeled 'Add Ingredient'.

Fig 6.2.7 Formularul de adăugare a unui ingredient

Pentru adăugarea în baza de date a unui obiect de tipul produs, în formularul asociat acestei acțiuni, apare conceptul de adăugare de ingrediente deja existente în baza de date, sau de ștergere a unui ingredient adăugat greșit.

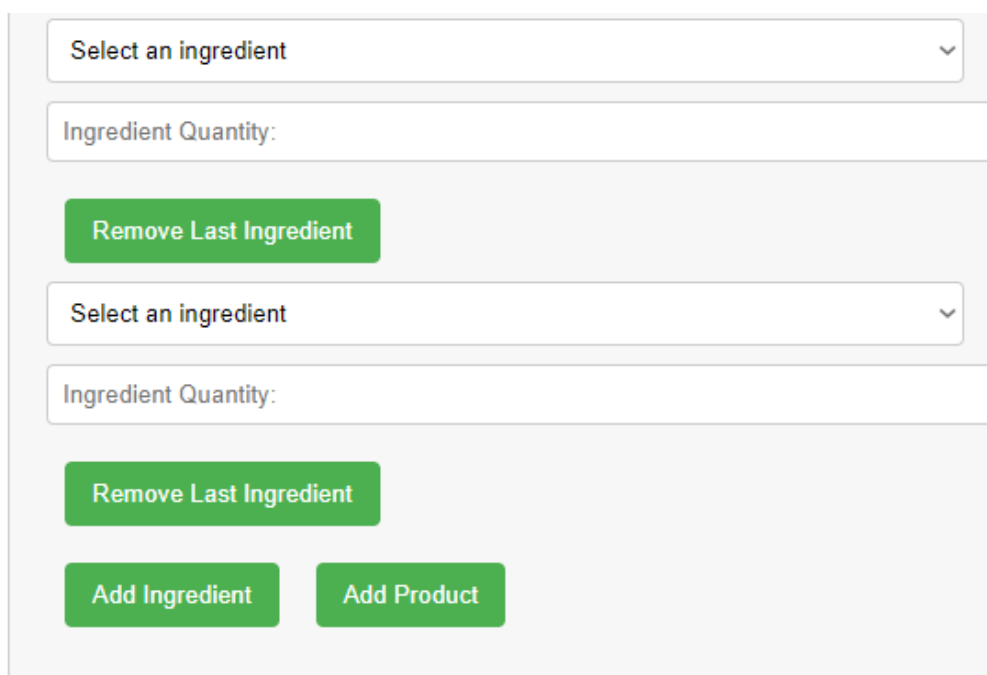
A screenshot of a web form for adding ingredients to a product. It features two identical sections. Each section starts with a dropdown menu labeled 'Select an ingredient' and a text input field labeled 'Ingredient Quantity:'. Below each input field is a green button labeled 'Remove Last Ingredient'. At the bottom of the form are two green buttons: 'Add Ingredient' and 'Add Product'.

Fig 6.2.8 Adăugarea ingredientelor la crearea produselor

Pentru crearea de rețete noi, se folosește conceptul anterior atât pentru gestionarea ingredientelor, cât și pentru gestionarea produselor.

The form is divided into two main sections: 'Ingredients' and 'Products'. The 'Ingredients' section includes a dropdown menu labeled 'Select an ingredient', a text input field labeled 'Ingredient Quantity:', a green button 'Remove Last Ingredient', and another green button 'Add Ingredient'. The 'Products' section includes a dropdown menu labeled 'Select a Product', a text input field labeled 'Product Quantity:', another dropdown menu labeled 'Select a Product', and a text input field labeled 'Product Quantity:'. At the bottom of the form, there are three green buttons: 'Remove Last Product', 'Add Product', and 'Add Recipe'.

Fig 6.2.9 Adaugarea produselor si a ingredientelor la crearea retetelor

O ultimă funcționalitate auxiliară o reprezintă factorul de aleatoritate al alegerii rețetelor dintr-un plan alimentar și respectarea constrângerilor generate de condițiile fiecărui utilizator, astfel, de exemplu unei persoane intoleranță la lactoză nu i se vor atribui rețete cu lactate în planul alimentar.

În funcție de scopul ales în formularul de generare a planului alimentar, necesarul caloric va scădea cu 300 în cazul în care se alege “Loose Weight”, va crește cu 300 în cazul în care se alege “Gain Weight”, sau rămâne constant în cazul alegerii “Maintain”, după cum se observă în codul următor:

```
switch(data.type) {
  case 'cut':
    leftScore = score - 300;
    break;
  case 'mantain':
    leftScore = score;
    break;
  case 'bulk':
    leftScore = score + 300;
    break;
  default:
    break;
```

Filtrarea rețetelor în funcție de constrângeri presupune eliminarea din colecția ce conține toate rețetele, pe cele ce nu satisfac preferințele și nevoile utilizatorului.

```
constraints = response.data.constraints;
if (constraints) {
  const constrainedRecipes = listOfRecipes.filter((recipe) =>
    constraints.includes(recipe.category)
  );
  setlistOfConstrainedRecipes(constrainedRecipes);
} else {
  setlistOfConstrainedRecipes(listOfRecipes);
}
```

Alegerea rețetelor este una parțial aleatorie, practic rețetele sunt alese aleatoriu din colecția de rețete filtrate, dar sunt alese astfel încât suma caloriilor să se încadreze în scorul modificat.

```
while (selectedRecipeIds.length < 3) {
  const randomArrayIndex = Math.floor(Math.random() *
listOfConstrainedRecipes.length);
  const randomRecipe =
listOfConstrainedRecipes[randomArrayIndex];

  if (randomRecipe.totalCalories <= leftScore ) {
    selectedRecipeIds.push(randomRecipe.id);
    leftScore -= randomRecipe.totalCalories;
    listOfConstrainedRecipes.splice(randomArrayIndex, 1);
  }
  else {
    listOfConstrainedRecipes.splice(randomArrayIndex, 1);
  }
}
```

O mică explicație a algoritmului ar fi: Cât timp nu au fost alese 3 rețete, generăm un număr aleator, ce reprezintă indicele unei rețete stocate, în baza de date, și verificăm dacă numărul de calorii al unei porții se încadrează în necesarul caloric rămas, iar dacă o face adăugăm rețeta la planul nutrițional, o ștergem eliminăm din lista de rețete pentru a nu fi adăugată din nou, și actualizăm necesarul caloric.

7. Idei de dezvoltare

Modul în care aplicația Calorie este formată permite adăugarea de funcționalități și concepte noi.

Versiunile următoare ale aplicației vor conține:

1. Sistem de preferințe, utilizatorul putând alege o listă cu elemente favorite din ingrediente, produse și rețete, ce va fi luată în calcul la generarea planului alimentar;
2. Sistem de tip checklist pentru a putea verifica în timp real ingredientele la care utilizatorul are acces;
3. Sistem de înlocuire a ingredientelor cu variante similare;
4. Funcție de reminder, notificări setate la anumite ore pentru amintirea utilizatorului că este timpul să ia masa;
5. Posibilitatea de a introduce gustări între mese, care să se încadreze în necesarul caloric;
6. Posibilitatea de a genera planuri nutriționale pe perioade mai lungi de timp, alături de actualizarea în timp real a profilului cu date estimate;

8. Concluzie

În concluzie, în componența acestei lucrări, am descris procesul de dezvoltare a aplicației web ce generează planuri alimentare personalizate, adaptate la nevoile, condițiile și preferințele fiecărui utilizator.

Aplicația este construită în jurul unei arhitecturi modulare bine definite, împărțind funcționalitățile în diverse componente și rute specifice, combinând tehnologii precum Node.js, React, MySQL, Sequelize, Axios, etc. pentru a realiza interacțiunea între componente și receptivitatea interfeței utilizator.

Prin tema aleasă, consider că fiecare utilizator își poate schimba stilul de viață, folosind aplicația doar câteva minute în fiecare zi. Aceasta oferă o soluție convenabilă pentru persoanele cu timp liber limitat care doresc să facă o schimbare benefică sănătății.

În ansamblu, aplicația Calorie, pune la dispoziția utilizatorilor, o unealtă foarte ușor de utilizat, ce oferă o experiență interactivă și constructivă pentru gestionarea alimentației indivizilor, ținând cont și de cele mai fine nevoi și preferințe ale acestora.

Bibliografie

- [1] Pew Research Center, *Eating More; Enjoying Less*, <https://www.pewresearch.org/social-trends/2006/04/19/eating-more-enjoying-less/>, accesat 10.06.2023
- [2] Jay Garg, *Learning on an Empty Stomach*, <https://harvardpolitics.com/learning-on-an-empty-stomach/>, publicat: 27.02.2020, accesat 10.06.2023
- [3] Bethany Norton, Jake Linardon, Mariel Messer, Melanie Smart, Zoe McClure, *Using an app to count calories: Motives, perceptions, and connections to thinness- and muscularity-oriented disordered eating*, <https://pubmed.ncbi.nlm.nih.gov/34543856/>, publicat: 15.09.2021, accesat 12.06.2023
- [4] Amanda Capritto, Natalie A Rahhal, *Best Calorie Counter Apps*, <https://www.verywellfit.com/best-calorie-counter-apps-4777302#:~:text=MyFitnessPal%20earns%20the%20top%20spot,connect%20with%20friends%20for%20support.&text=Health%20and%20fitness%20advocates%20and,living%20apps%20on%20the%20market>, publicat: 12.05.2023, accesat: 14.06.2023
- [5] Documentatie oficiala React, <https://legacy.reactjs.org/docs/getting-started.html>, accesat: 03.06.2023
- [6] Documentatie oficiala Node.js, <https://nodejs.org/en/docs>, accesat: 04.06.2023
- [7] Pagina oficiala a Ministerului Sanatatii Emiratelor Arabe Unite, *Calories Calculation*, [https://mohap.gov.ae/en/more/awareness-center/calories-calculation#:~:text=If%20you%20are%20sedentary%20\(little,Calorie-Calculation%20%3D%20BMR%20x%201.55](https://mohap.gov.ae/en/more/awareness-center/calories-calculation#:~:text=If%20you%20are%20sedentary%20(little,Calorie-Calculation%20%3D%20BMR%20x%201.55), accesat: 08.06.2023
- [8] Documentatie oficiala Axios, <https://axios-http.com/docs/intro>, accesat: 04.06.2023
- [9] Documentatie oficiala Sequelize, <https://sequelize.org/docs/v6/getting-started/>, accesat: 03.06.2023