Cal-o-Web

Cerinta:

Sa se dezvolte o aplicatie Web care sa permita utilizatorilor autentificati sa-si monitorizeze caloriile consumate pe parcursul unei unitati de timp (zi, saptamina, luna, an) in functie de ce au consumat in acea unitate de timp. De asemenea, se vor putea propune schimbari in dieta fiecarui utilizator pe baza preferintelor personale (e.g., daca pentru Tuxy se constata ca numarul de calorii lunare a scazut, atunci i se poate propune ca duminica sa consume 3 linguri de dulceata sau o banana). Aplicatia va oferi statistici, inclusiv vizualizari de interes, pe baza caloriilor acumulate si (evolutiei) greutatii – introduse manual de utilizator. Rapoartele generate vor fi exportate in format XML.

Arhitectura Aplicatie

1.Tipuri de utilizatori

Utilizatorul neautentificat: poate doar vizualiza: pagina de start, pagina 'About' a sitului, se poate autentifica sau inregistra.

Utilizatorul autentificat: cu beneficii in plus fata de cel neautentificat, isi va putea vizualiza datele personale, va putea introduce ce a mancat intr-o anumita zi, va putea vizualiza atat mesele introduse anterior cat si interpretarile acestor date sub forma unor statistici si recomandari. De asemenea, acesta va putea introduce in baza de date un fel de mancare daca acesta nu exista deja.

2.Tehnologii Folosite

Partea de front-end a fost implementata folosind **HTML5, CSS** si putin **JavaScript** pentru dinamica paginii. Pentru partea de back-end vom folosi serverul web **Apache**, deoarece este puternic si flexibil, permitand modificarea web serverului dupa bunul plac si limbajul de programare **PHP**, fiind usor de integrat intr-o pagina web HTML. Aceasta componenta (back-end) implementeaza functionalitatile front-endului, executa operatii CRUD in baza de date. Vom utiliza o baza de date **MySql** pentru a retine datele utilizatorilor, alimentele si relatia dintre acestea.

3.Dessign Patternul folosit

Aplicația este structurată pe 3 niveluri: *view, controller, model*.

Nivelul *view* este reprezentat de interfața aplicației. Când utilizatorii accesează aplicația aceștia își pot crea un cont de utilizator sau se pot autentifica. La crearea

unui nou cont de utilizator, aplicația valideaza datele introduse de ei. Dupa ce utilizatorul s-a conectat folosind datele contului acesta poate adauga alimentele consummate in ziua respectiva sau poate vizualiza un istoric al meselor.

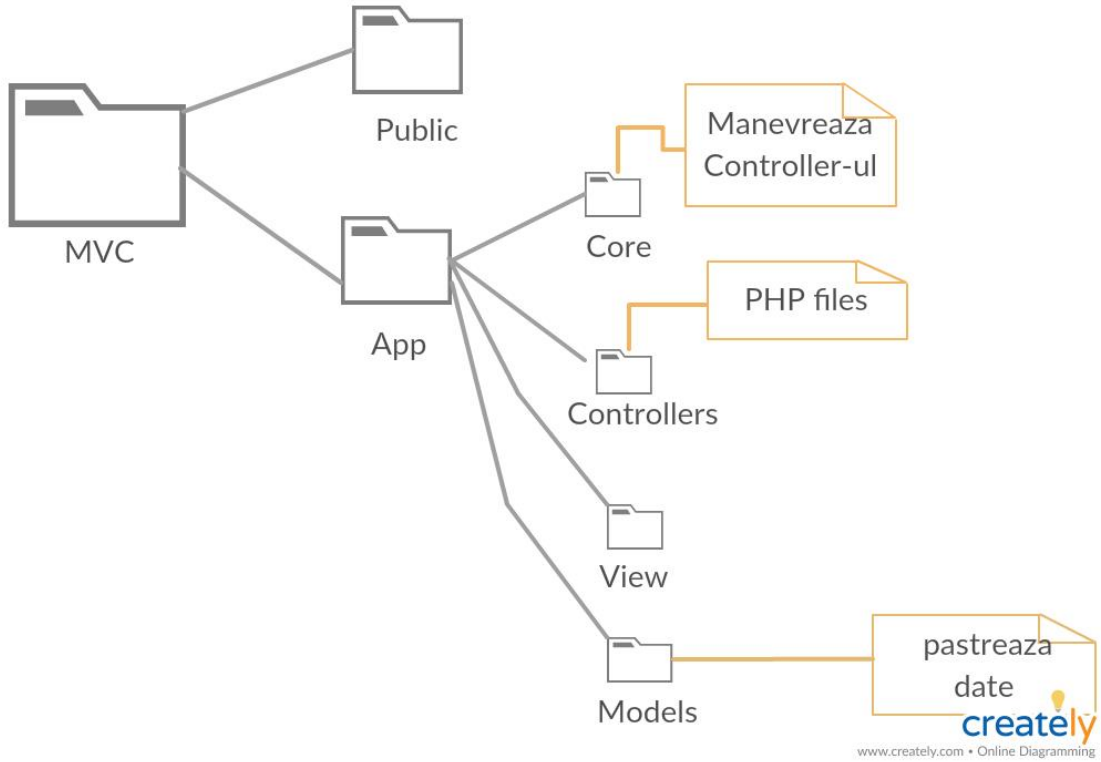
Nivelul *controller* al aplicației care preia inputul de la user. Controllerul va prelua date de la nivelul model pe crae le va prelucra pentru ca apoi sa fie afisate utilizatorului.

Nivelul *model* este responsabil cu preluarea si prelucrarea datelor care vor fi retinute intr-o baza de date.

Un utilizator face o cerere prin URL-ul, iar serverul Web trateaza cererea prin executarea fisierului index.php. Fisierul index.php creaza o instanta application si o ruleaza.

```php
<?php
        require_once '../app/init.php';
         $app = new App;
```

Aplicatia obtine informatii detaliate despre cererea utilizatorului, acesta fiind locul unde se executa procesarea cererilor client. Rolul principal este analizarea cererii client si transmiterea ei la controller-ul corespunzator pentru a fi procesata in continuare. Atunci cand ruleaza, un controller executa un action cerut de client. De obicei, action apeleaza modelele necesare si va genera un view corespunzator (rezultatul vazut de client).

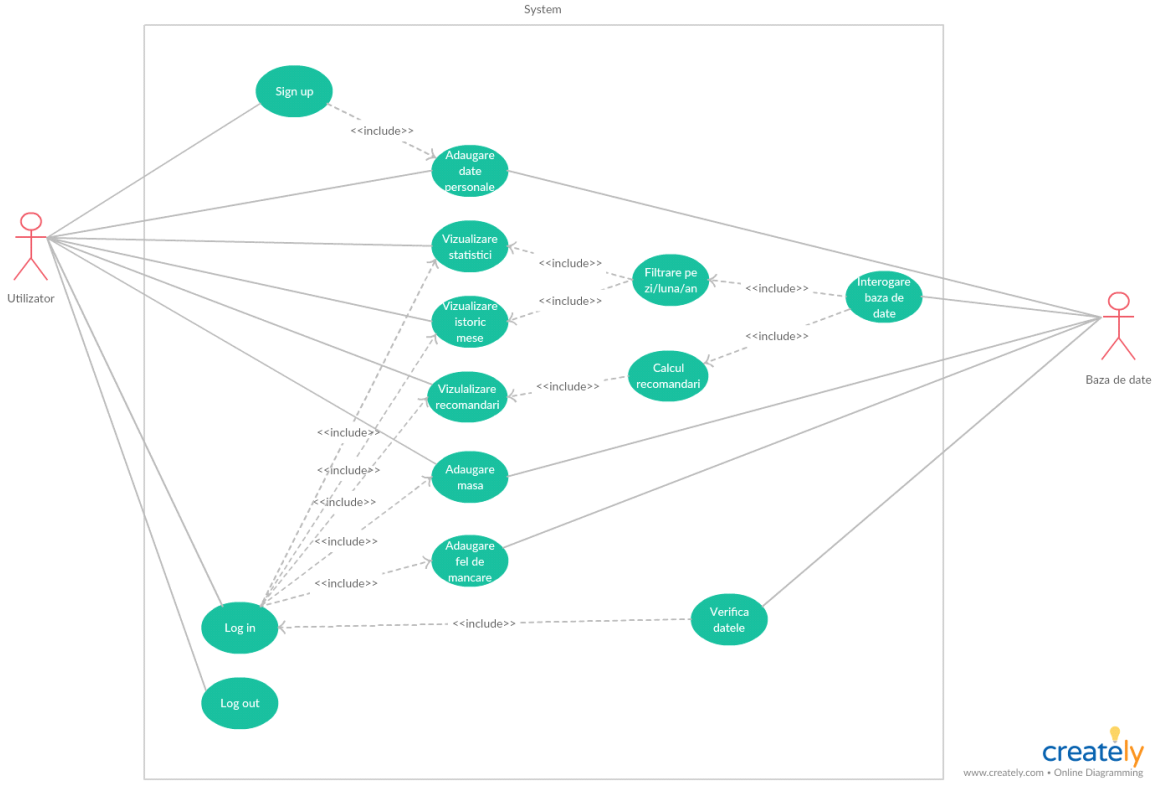


Exemplu de comunicare intre view, controller si model in timpul interactionarii cu aplicatia: Ne aflam in pagina principala si dorim sa accesam contul nostru de utilizatori(LogIn), apasam linkul de LogIn sau register care ne va redirectiona catre pagina corespunzatoare. Introducem datele in casutele username si password si apasam butonul LogIn, in acest moment componenta controller primeste de la view aceste date, username-ul si parola pe care le-am introdus, controllerul paseaza aceste date modelului iar modelul verifica in baza de date, mai exact in tabela user daca aceste date introduse sunt unice, daca da, atunci modelul ia din tabela datele necesare afisarii urmatoarei pagini, le paseaza controllerului, iar acesta le da mai departe view-ului pentru ca acesta sa afiseze pagina corespunzatoare(History/AddFood), daca nu, atunci controllerul da mai departe un mesaj de eroare care este mai apoi afisat in view, in pagina Log in.

4. Stil arhitectural

Ca stil arhitectural, vom folosi paradigma REST. Aceasta paradigm presupune separarea serverului de client la nivel de implementare, aceasta modularizare ajutand la o modificare eficienta a codului si gestionarea facila a aplicatiei web. Asadar, toate datele fiind pastrate in baza de date la care este conectat serverul.

5. Diagrama Use Case



6. Scenarii de utilizare

6.1.Utilizatorul doreste sa se logheze

- Acceseaza site-ul
- Acceseaza linkul care il va redirectiona catre pagina de LogIn sau Register
- Selecteaza LogIn
- Introduce Username-ul si Parola
- Apasa butonul de submit; daca datele sunt valide atunci va putea intra in cont si efectua alte operatiuni, altfel va primi un mesaj de eroare.

6.2.Utilizatorul doreste sa se delogheze

• Apasa butonul de Log Out care va distruge sesiune curenta si il va redirectiona catre pagina principala a aplicatiei web

6.3.Utilizatorul doreste sa se inregistreze

- Acceseaza site-ul
- Acceseaza linkul care il va redirectiona catre pagina de LogIn sau Register
- Selecteaza Register
- Introduce Username-ul pe care si l-a ales si Parola de doua ori
- Apasa butonul de submit
- Daca datele sunt valide (nu exista un alt utilizator cu acelasi username), utilizatorul va fi inregistrat in baza de date fiind redirectionat catre pagina de inregistare a datelor personale. In caz contrar i se va semnala o eroare si i se va cere reintroducerea datelor

6.4.Utilizatorul isi adauga datele personale

- completeaza datele personale in casutele corespunzatoare
- daca nu completeaza datele obligatorii i se va semnala acest lucru printr-un mesaj
- va apasa butonul se Submit care va trimite datele care se vor salva in baza de date
- aceste date vor putea fi vizualizate si modificate ulterior intr-o pagina de tip 'Contul meu'

6.5.Utilizatorul inregistrat doreste sa adauge alimentele consumate

- Acceseaza linkul Add Food
- Utilizatorul completeaza in casuta corespunzatoare din formular alimentele consumate
- Prin apasarea butonului submit datele vor fi trimise la baza de date unde vor fi inregistrate in tabelul 'Meals'. In acelasi timp cu ajutorul metodei POST datele introduse anterior vor putea fi vizualizate in aceeasi pagina, intr-un div corespunzator tipului de masa inregistrat (Breakfast, Lunch, Dinner, Snack).

6.6.Utilizatorul inregistrat doreste sa adauge un aliment nou in baza de date

Deoarece exista posibilitatea ca nu toate alimentele sa se gaseasca in baza de date, la adaugarea alimentelor consumate exista si butonul de 'Add food' care va redirectiona utilizatorul catre o noua pagina unde:

- Utilizatorul completeaza casutele corespunzatoare
- Prin apasarea butonului submit datele vor fi trimise la baza de date unde vor fi inregistrate in tabelul 'Aliments'.

6.7.Utilizatorul inregistrat doreste sa vizualizeze statisticile personale

- Acceseaza 'Contul meu' unde la sectiunea statistici
- Alege un criteriu de filtrare a datelor: dupa zi, luna sau an
- Va putea vedea detalii despre evolutia greutatii si a numarului de calorii consumate (cate proteine sau grasimi a consumat), alimente preferate

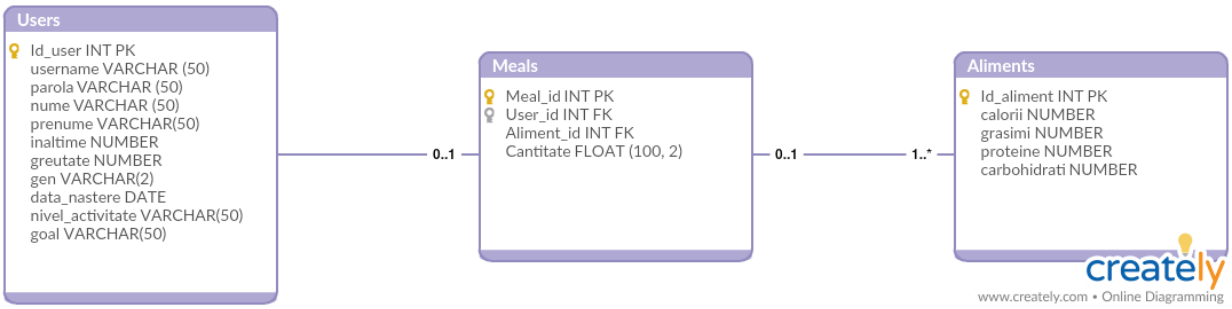
6.8.Utilizatorul inregistrat doreste sa vizualizeze istoricul meselor

- Acceseaza 'Historic'
- Alege un criteriu de filtrare a datelor: dupa zi, luna sau an
- Va putea vedea concret ce alimente a consumat intr-o anumita perioada de timp

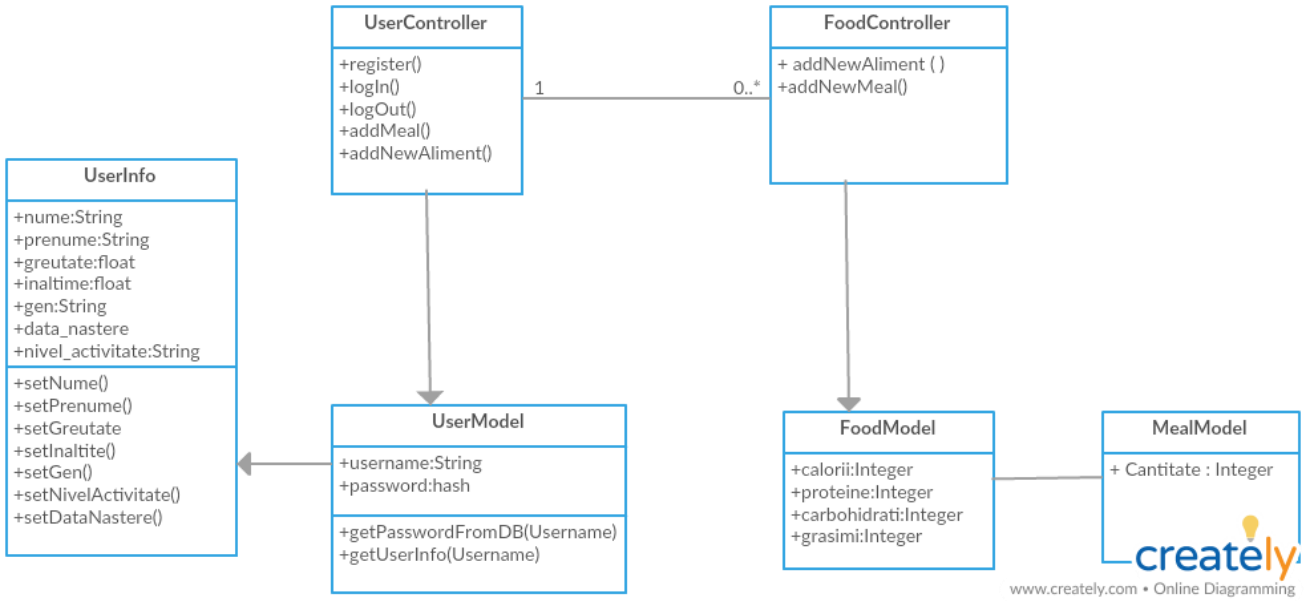
6.9.Utilizatorul inregistrat primeste recomandari

- In pagina 'Historic' sau in sectiunea cu statistici vor exista containere speciale in care utilizatorii vor primii recomandari:
- daca in medie, pe parcursul unei luni, a depasit cantitatea zilnica de calorii recomandate i se va sugera sa scada consumul alimentelor care apar de ce le mai multe ori in baza de date (au fost consumate de cel putin 2 ori in acea luna) si care contin cele mai multe grasimi
- daca, dimportiva, nu acumuleaza numarul de calorii recomandate i se va sugera sa consume alte alimente, fie pe baza unei categorii de 'alimente sanatoase' existente in baza de date, fie pe baza alimentelor sale preferate (cele care apar de cele mai multe ori in tabelul 'Meals')

7. Structura bazei de date



8.Diagrama UML



Bibliografie:

https://www.yiiframework.com/doc/guide/1.1/ro/basics.mvc

https://www.youtube.com/watch?v=OsCTzGASImQ&list=PLfdtiltiRHWGXVHXX09fxXDi-DqInchFD

https://www.codeofaninja.com/2017/02/create-simple-rest-api-in-php.html