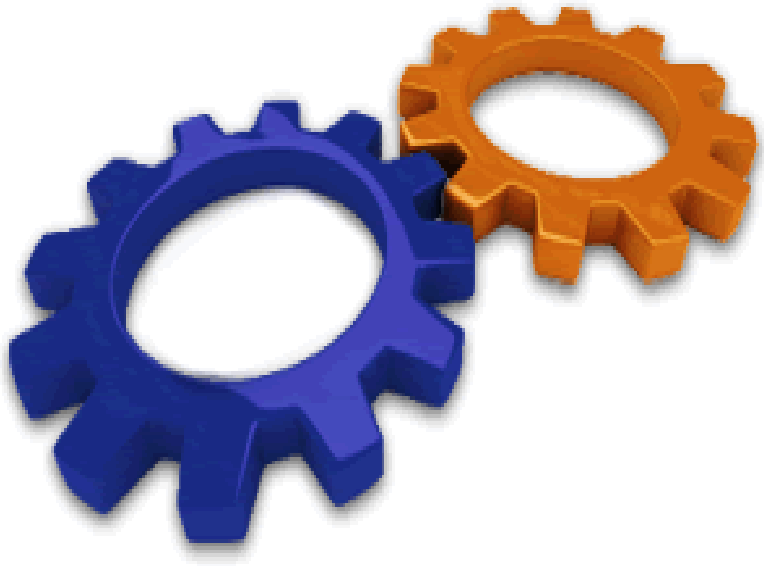
Cal-O-Web

eat to live, not live to eat

Arhitectura aplicației



Prof.coordonator: Coman Alexandru

Autori: Bârseti Andrei & Nedelcu Andreea (Anul II, grupa A6)

Cuprins:

1. Tehnologii folosite

Pentru a realiza front-end-ul am folosit HTML5 si CSS. Pentru a realiza partea de back-end, partea care face front-end-ul funcționabil folosind operații de tip CRUD în baza de date, am ales să folosim PHP.

2. Structura bazei de date

Vom avea o bază de date MySql pentru a salva date despre utilizator.

3. MVC

Vom folosi ca design pattern MVC, model view controller.

Model: această parte manipulează operațiunile logice și de utilizare de informație pentru a rezulta o formă ușor de înțeles de către utilizator.

View: acestui membru al familiei îi corespunde reprezentarea grafică, sau mai bine zis, exprimarea ultimei forme a datelor: interfața grafică ce interacționează cu utilizatorul final. Rolul său este de a evidenția informația obținută până ce ea ajunge la controller.

Controller: cu acest element putem controla accesul la aplicația noastră. Pot fi fișiere, scripturi sau programe, în general orice tip de informație permisă de interfață. În acest fel putem diversifica conținutul nostru de o formă dinamică și statică, în același timp.

Schema de funcționare a aplicației modelate după arhitectura MVC decurge, în linii mari, astfel:

Introducem username-ul si parola și apăsăm Login, controllerul primește aceste date de la view, acesta le trimite mai departe catre model care le verifică. Dacă acestea sunt unice modelul procură datele necesare pentru a trece la o pagină nouă, le trimite controllerului, iar acesta le dă mai departe view-ului pentru afișare. Însă dacă datele trimise nu sunt unice controllerul trimite un mesaj de eroare care este afișat in view.

1. Utilizatorul interacționează cu interfața. (exemplu: introducem username-ul si parola si apasăm butonul de Login)

2. Controller-ul primește acțiunea apăsării butonului și o converteste într-o acțiune pe intelesul model-ului.(controller-ul primește username-ul și parola introdusă).

3. Controller-ul notifică model-ul de acțiunea utilizatorului, urmând de obicei o schimbare a stării model-ului. (exemplu: model-ul reîmprospatează starea câmpului de adresă)

4. Un view interoghează model-ul pentru a genera o interfață corespunzatoare. (exemplu: view-ul afisează noua adresă alături de cea veche alături de un buton de confirmare)

5. Interfața așteaptă acțiuni suplimentare din partea utilizatorului, ciclul reluându-se.