**H.M.H.B. - “Healthy mind in a healthy body”**

-secțiunea web-

Site creat de: **Borcan Aniela** – **Eleva cls a XI-a A**

**Colegiul Național B. P. Hasdeu Buzău**

Profesor coordonator:

**Nicolae Olăroiu**

Cuprins

[**H.M.H.B. - “ Healthy mind in a healthy body ”** 3](#_Toc4307608)

[**Avantajele utilizării site-ului H.M.H.B.** 5](#_Toc4307609)

[**Prezentarea site-ului** 6](#_Toc4307610)

[**Realizarea site-ului** 8](#_Toc4307611)

[**Bibliografie**](#_Toc4307613) 19

# **H.M.H.B. - “ Healthy mind in a healthy body ”**

Internetul reprezintă, in zilele noastre, unul dintre cele mai importante mijloace de informare, comunicare, și distracție. Informarea se realizează și în domeniul sanatății, astfel, primul pas spre însănătoșire și vindecare este căutarea unei instituții autorizate specializată în domeniul necesar. Mai mult, vizita la clinică începe în mediul online.

**" H.M.H.B. - Healthy mind in a healthy body "**  este un site foarte ușor de utilizat, ce ajută orice utilizator să se informeze în legătură cu toate clinicile medicale din județul Buzău (sau orice alt județ). Informațiile de foarte bună calitate și mereu actualizate vă vor ajuta să vă alegeți clinica perfectă pentru nevoile voastre. Pagina web urmărește nu numai să ajute oamenii ci și să le ofere încredere și susținere în deciziile pe care le iau. Prin intermediul unui design accesibil oricărui utilizator, vă puteți documenta despre serviciile oferite de clinicile medicale. **H.M.H.B.** oferă în prezent informații despre specialitați medicale precum: medicina internă, cardiologie, boli metabolice, nutriție și diabet, endocrinologie, gastroenterologie, pneumologie, oftalmologie, ORL, psihiatrie, ginecologie, chirurgie, ortopedie, neurologie, dermatologie, stomatologie, psihologie, oncologie, urologie.

Site-ul a fost realizat în scopul folosirii acestuia drept un material de informare corect și amănunțit accesibil la orice oră din zi si noapte.

Utilizarea acestui site este foarte folositoare în economisirea timpului necesar documentării și alegerii doctorului pe specializarea ce ne interesează.

Lovindu-ne de greutatea alegerii de către bolnavi a unei clinici cât mai apropiate de locuință și a unor doctori cât mai specializați, am vrut să ofer viitorilor pacienți o sarcină mai ușoară. Pentru mine, fiecare pacient este important și nu uit că înainte de toate este om și are nevoie să fie tratat cât mai repede. De aceea acest site a fost făcut din respect pentru oameni și pentru sănătatea lor.

# **Avantajele utilizarii site-ului H.M.H.B.**

1. Documentarea și selectarea clinicii și a doctorului pe domeniul căutat se face extrem de rapid ajutând astfel la vindecarea pacientului.
2. Poate fi accesat de pe orice device care are conexiune stabila la internet.
3. Site-ul este securizat împotriva SQL Injection.

# **Prezentarea site-ului**

Site-ul este foarte ușor de accesat și utilizat.

Interfața programului este concepută astfel ca utilizatorul să găsească informațiile necesare ușor și rapid. Prezintă un meniu sugestiv cu mai multe secțiuni: HOME, DESPRE NOI, SERVICII, ECHIPA și CONTACT.

Secțiunea "HOME" este atractivă prin culori și design și oferă posibilitatea utilizatorului să aleaga zona care îl interesează printr-un simplu "click" pe regiunea corporală a imaginii din aceasta pagina web. Astfel se va deschide o nouă pagină cu toate domeniile aferente zonei respective. Alegând domeniul, și dând “click”, accesăm pagina clinicilor existente în județ (pe specialitatea respectiva).Intrând pe clinică se va afișa logo-ul firmei, numele, adresa, telefonul, email-ul, medicii și specializările lor.

Secțiunea "DESPRE NOI" oferă câteva informații despre site și de asemenea o serie de 3 articole. Acestea pot fi accesate printr-un "click". Astfel se va deschide o nouă pagină ce conține știrea respectivă. Totodată, acestea pot fi modificate și încărcate de administrator (sau de cel care deține site-ul) cu ajutorul unei alte pagini web. Administratorul poate adauga în baza de date titlul știrei alături de conținut, care va fi scris sub forma de cod HTML, dar și de o poza semnificativă. Baza de date a fost creată astfel încât să se poată încărca orice știre.

Secțiunea "SERVICII" afișează toate domeniile prezente în baza de date necesare pentru a acoperi o gamă largă de specialități. Informațiile sunt aranjate într-un mod elegant și atractiv. Prin alegerea unui domeniu (dând "click" pe poza respectivă) se vor afișa clinicile corespunzătoare. Se ajunge, astfel, din nou la clinici unde vom găsi toate datele despre acestea, inclusiv localizarea pe harta.

Pentru a fi cât mai atractiv pentru utilizator am facut secțiunea "ECHIPA" în care sunt prezentați câțiva din doctorii specializați.

Secțiunea "CONTACT" oferă posibilitatea utilizatorilor să facă sugestii administratorului în legatură cu îmbunătățirea site-ului prin trimiterea unui email.

# **Realizarea site-ului**

Site-ul este unul dinamic și responsive. Este realizat în php, utilizând arhitectura MVC – Model – View – Controller. Pentru implementarea acesteia, am utilizat framework-ul php CodeIgniter folosind programarea orientată pe obiecte (POO) în php.

Baza de date a fost creată în MySql, folosind phpMyadmin. Pentru realizarea interfeței, am utilizat HTML 5, CSS 3.0 , jQuery , Bootstrap. Pentru trimiterea email-urilor am folosit PhpMailer, iar pentru localizarea clinicilor pe hartă Google Maps. De asemenea, site-ul este securizat împotriva SQL Injection. Logo-ul acestui site a fost creat integral in Logo Maker & Adobe Photoshop CS6 2018.

**Prezentarea arhitecturii MVC:**

Cea mai ușoara metodă de a realiza o aplicatie care realizează aceste operații este de a pune laolaltă operațiile și de a le trata ca pe un tot. Aceasta metodă este buna în sensul că este ușor de implementat. Ulterior apar însă probleme când se dorește schimbarea uneia din componentele fluxului de date,spre exemplu atunci când se dorește schimbarea interfeței. O alta problemă ține de logica de business ce trebuie încorporată, logica, care și ea este supusa schimbărilor și care merge dincolo de simpla interschimbare de informație.

Apare așadar nevoia de a modulariza aplicația, de a delimita în mod clar părțile componente spre a putea fi ușor modificate și ca după modificare să fie încă compatibile cu celelalte module ce formează aplicația.

O soluție la această problemă este arhitectura Model-View-Controller(MVC) care separă partea de stocare a datelor de cea de prezentare și de prelucrare. Avem așadar 3 clase distincte:

---Model-ul se ocupă de comportarea și datele aplicației; răspunde la cereri despre starea sistemului, la cereri de schimbare de stare și notifica utilizatorul atunci când aceste schimbări au avut loc pentru ca acesta să poată reacționa.

---View-ul transpune model-ul într-o formă care permite o interacționare ușoară, în mod tipic o interfața vizuală. Pot exista multiple view-uri pentru un singur model pentru scopuri diferite.

---Controller-ul primește input de la utilizator și inițiază un răspuns în urma cererilor către obiectele model. Controller-ul este cel care controlează celelalte două clase de obiecte, view si model, instructându-le să execute operații pe baza input-ului primit de la utilizator.

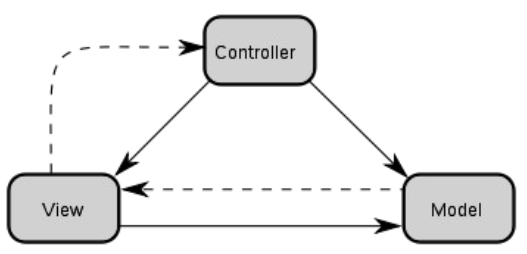


Diagrama arhitecturii MVC prezintă liniile solide ca asocieri directe iar liniile punctate ca asocieri indirecte.

**Funcționare:**

Schema de funcționare a unei aplicații modelate după arhitectura MVC decurge, în linii mari, în felul următor:

1. User-ul interacționează cu interfața. (exemplu: apasă un buton la tastatură)

2. Controller-ul primește acțiunea apăsării butonului și o convertește într-o acțiune pe întelesul model-ului.

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

4. Un view interoghează model-ul pentru a genera o interfața corespunzătoare.

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

**Folosirea unui framework în php permite:**

* Abstractizarea bazei de date
* Abstractizarea cache-ului
* Management-ul form-urilor
* Autentificarea
* Depanare uşoară
* Internaţionalizarea

**Avantajele folosirii unui framework: portabilitate**(Abstractizarea bazei de date şi a cache-ului fac ca aplicaţiile construite cu un framework să poată rula pe servere, având diferite configuraţii fără a fi necesară rescrierea vreunui rând de cod**.**), **timp de dezvoltare mai scurt** (Asta se datorează în primul rând faptului că nu mai trebuie să scriem codul pentru form-uri, înregistrare, log in şi log-out etc.), **securitate** (Framework-ul se ocupă de securitatea pentru modulele buid-in: autentificare, formulare, query-uri SQL), **suportul comunităţii**, **plugin-uri şi module**, **standard de codare**, **documentaţie**.

**CodeIgniter PHP framework**

CodeIgniter, pe de altă parte, se axează pe simplitate, atât în instalare cât și în dezvoltare și mentenanță. Dezvoltat deja de 10 ani, ajuns la versiunea 3.0.6, CodeIgniter poate fi folosit pentru orice tip de proiect, de la blog sau site de prezentare la portal, REST Server sau magazin e-commerce.

Comparativ cu celelalte framework-uri folosite la ora actuală, CodeIgniter stă extraordinar de bine raportat la ușurința de utilizare. Astfel, este în top 5, atât la popularitate cât și la memorie consumată sau viteză.

CodeIgniter merita incercat pentru ca:

* Este gratuit.
* Prezinta ușurință în instalare (rulează pe orice tip de hosting).
* Versiunea PHP minimă este PHP 5.4.
* Memorie consumată este foarte puțină.
* Este foarte rapid.
* Are structură Model-View-Controller.
* Are documentație completa.
* Prezinta dezvoltare continuă și suport pentru Bug-uri/Feature-uri pe Github.



**Routing** - Configurarea routing-ului, foarte simplă, din application/config/routing.php. Este permisă scrierea întregului URL sau se pot folosi expresii RegExp. Metoda de lucru este clasică, /controller/method/. Primul segment din URL este controller-ul ce va fi apelat, al doilea segment, metoda apelată.

**Security** - CodeIgniter are incluse default caracteristici clasice de securitate: securizarea adresei url (URI), securizarea register\_globals ($\_GET, $\_POST, $\_REQUEST, $\_COOKIES), XSS filtering, CSRF protection, Form Validation.

**Application Controller** - Controler-ul se extinde prin CI\_Controller. (locație: application/controllers/)

**Views** - View-ul este o pagină sau un fragment html (exemplu: header, footer). View-ul nu este apelat direct de URI, ci prin intermediul controller-ului, cu sintaxa: $this->load->view('name'); Locație: application/views. Nu este necesar să fie folosit vreun "Template Engine", care ar îngreuna atât programarea cât și mentenanța. Codul PHP standard este suficient.

**Caching** - Un element important în dezvoltarea de proiecte este și sistemul de caching. CodeIgniter permite caching atât pe pagini (view) cât și pe rezultate din baze de date.

**Models** - Modelele sunt clase folosite pentru manipularea datelor (și a bazelor de date). Locație: application/models. Modelul extinde un model de bază (CI\_Model) și poate fi apelat din Controller după ce este inițializat, fie din fișierul de configurare config/autoload.php, fie cu sintaxa: $this->load->model('model\_name');

**Baze de date** - CodeIgniter suportă cele mai folosite baze de date: cubrid, ibase, mssql, mysql (mysqli), oci8, odbc, pdo, postrgre, sqlite, sqlite3. Configurarea este foarte simplă în application/config/database.php . Se pot folosi mai multe baze de date simultan.

**Libraries** - Pentru orice proiect avem nevoie și de cod propriu. Application/libraries este locul librăriilor proprii. Aceste sunt clase încărcate în Controller cu sintaxa: $this->load->library('someclass').

**Languages** - Traducerea conținutului este un alt aspect important al oricărui proiect. Încărcarea fișierelor de traducere: $this->lang->load('filename', 'language') , afișarea unei linii de traduceri: $this->lang->line('language\_key').

**Helpers** - Pentru funcții proprii, se pot folosi fișierele "helper", unde se pot declara toate funcțiile necesare.

CodeIgniter vine la pachet cu librării proprii necesare în aproape orice proiect, în număr de 30. Printre ele: Benchmark, Email, Caching, Encrypt, File Upload, Form Validation, Image Manipulation, etc.

**PHP OOP**

OOP (Object Oriented Programming - Programare Orientată pe obiecte) este un concept (sau tehnică) de programare care tratează datele și funcțiile ca obiecte. Important la acest concept este să fie înțeleasa bine diferența dintre Clasa (Class) si Obiect (Object).

• Clasa este o structură de date compactă, care conține variabile (numite proprietăți) și funcții (numite metode) prin care se pot crea diferite instrucțiuni legate între ele în acea structură.

• Obiectele sunt elementele din script create cu o clasă, și prin care se pot folosi proprietățile și metodele din acea clasa.

* Creare clasă

Definirea unei clase începe cu un cuvântul cheie "class", urmat de numele clasei și corpul ei (cuprins între acolade). În corpul clasei se definesc proprietățile și metodele ei.

- Proprietățile sunt variabile definite în interiorul clasei.

- Metodele sunt funcții create în interiorul clasei.

Structura general de alcătuire a unei clase este următoarea:

class nume\_clasa {

$proprietate;

function metoda() {

// Codul functiei

}

}

* Utilizare clasă

După ce o clasa e creată, pentru a putea fi utilizată trebuie creată o instanță de obiect a clasei. Aceasta instanță se declara în scriptul PHP, cu formula: $nume\_obiect = new Nume\_Clasa(); (- "nume\_obiect" este numele obiectului prin care pot fi utilizate proprietățile și metodele acelei clase)

* Metoda constructor

Metoda constructor e un tip special de funcție din interiorul Clasei. Se declară cu numele \_\_construct (doua caractere "\_" în față). Această metodă este mereu "public" chiar dacă nu se menționează special atributul "public". Diferența față de celelalte funcții este aceea că Metoda Constructor e apelată automat când se creaza o instanța la clasa respectivă, codul din ea se execută în momentul când e creat obiectul (cu new).

* Metoda Destructor

Metoda Destructor se crează cu numele \_\_destruct (doua caractere '\_' la început). Această metodă este opusul lui \_\_construct. Daca metoda constructor e apelata automat când se crează o instanța de obiect la clasa, metoda Destructor se apelează automat când e ștearsă, cu unset(), instanța la acea clasă.

* Pe lângă proprietăți (variabile) și metode (funcții), în corpul clasei pot fi definite și constante, prin cuvantul const.
* Pe lângă atributele: "public", "private" și "protected"; prin care se definește domeniul de acces al proprietăților și metodelor; acestea pot fi create și ca statice, astfel, vor aparține exclusiv clasei, și nu pot fi apelate prin instanța de obiect a clasei, ci direct prin ea, cu operatorul două-puncte (::).

Principii de bază:

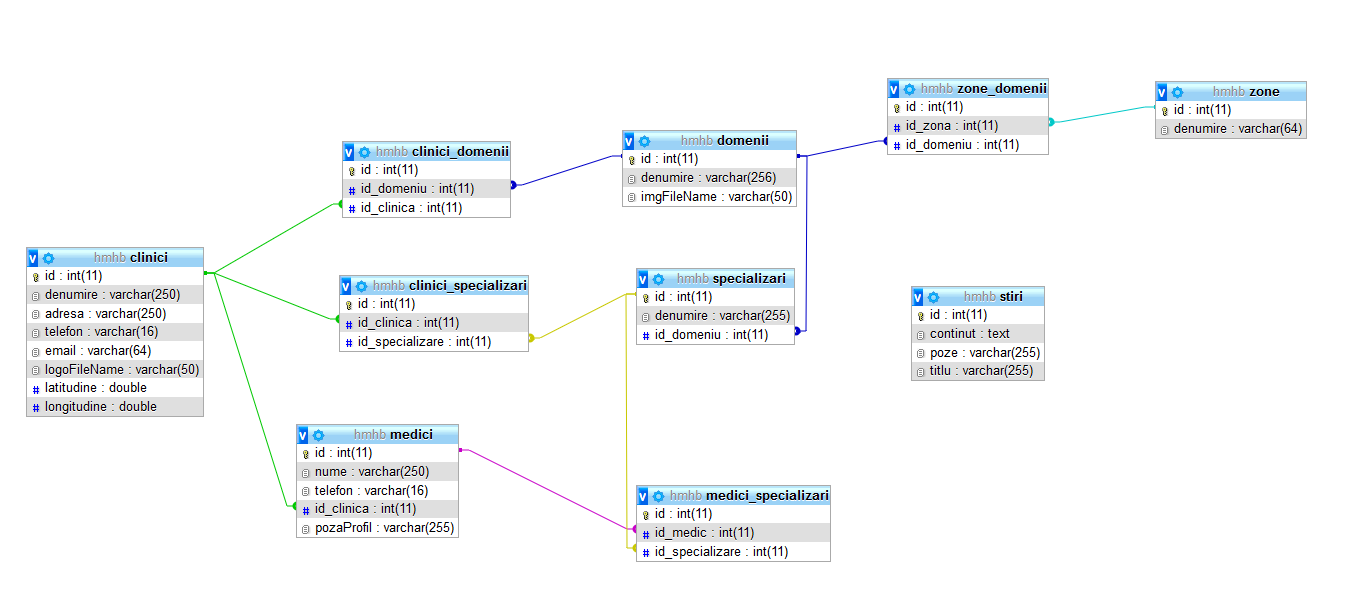
* Abstractizarea
* Încapsularea
* Polimorfismul
* Moștenirea

1. **Prezentarea componentei „Model” a site-ului**

În cadrul site-ului meu H.M.H.B., în folder-ul „models” se află fișierele php „AboutModel.php”, „ClinicsModel.php”, „ServicesModel.php”, “ClinicsPageModel”, “ShowNewsModel”, „TeamModel”, „UploadClinicsModel”, „UploadDoctorsModel”, „UploadNewsModel”, „UploadSpecModel” care conțin interogările SQL. Clasele din CodeIgniter ce conțin interogările sunt:

* AboutModel – Această clasă extrage din baza de date ultimele 3 articole pentru a fi afișate în pagina About.php.
* ClinicsModel – Această clasă extrage din baza de date clinicile aferente id-ului domeniului pe care s-a dat „click” în pagina sevices.php.
* ServicesModel – Această clasă scoate din baza de date toate domeniile afișate în ordine alfabetică.
* ClinicsPageModel – Această clasă extrage informațiile corespunzătoare fiecărei clinici.
* ShowNewsModel – Această clasă scoate din baza de date titlul și conținutul unei știri selectate.
* TeamModel – Această clasă scoate din baza de date informațiile despre medici(nume, specializare).
* UploadClinicsModel – Prin intermediul acestei clase administratorul poate introduce informații în baza de date. De asemenea aceasta clasa extrage id-ul clinicii corespunzatoare informațiilor introduse.
* UploadDoctorsModel – Prin intermediul acestei clase administratorul poate introduce informații în baza de date despre doctorii specializați, dar și extrage id-ul clinicii corespunzatoare informațiilor introduse.
* UploadNewsModel – Administratorul poate pune în baza de date articole interesante și actualizate.
* UploadSpecModel – Administratorul poate pune in baza de date specializările necesare medicilor.

Schema cu structura bazei de date și a relațiilor între tabele:



1. **Prezentarea componentei „View”** În folder-ul „views” sunt prezente fișierele php about.php, clinics.php, clinicsAdmin.php, clinicsPage.php, clinicsPageScript.php, contact.php, doctorsAdmin.php, index.php, indexAdmin.php, newsAdmin.php, sevices.php, sowNews.php, specsAdmin.php, team.php și în folder-ul templates header.php, head.php, footer.php, headerAdmin.php și script.php.

**Website-ul este responsive** întrucât se poate adapta la diferite rezoluţii şi dispozitive pe care este afişat (PC-uri, laptopuri, tablete, smartphones etc), fără a suferi degradări majore. Site-ul conține imagini flexibile care iși pot modifica lăţimea şi înălţimea în funcţie de rezoluţia divice-ului.

HTML5 este un limbaj pentru structurarea și prezentarea conținutului pentru World Wide Web, o tehnologie nucleu pentru Internet propusă inițial pentru software-ul Opera.

CSS (Cascading Style Sheets) este un standard pentru formatarea elementelor unui document HTML. Stilurile se pot atașa elementelor HTML prin intermediul unor fișiere externe sau în cadrul documentului, prin elementul <style> și/sau atributul style. CSS se poate utiliza și pentru formatarea elementelor XHTML, XML și SVGL.

Bootstrap este un set de instrumente open source pentru dezvoltarea cu HTML, CSS și JS.

jQuery este o platformă de dezvoltare JavaScript, concepută pentru a ușura și îmbunătăți procese precum traversarea arborelui DOM în HTML, managementul inter-browser al evenimentelor, animații și cereri tip AJAX. jQuery a fost gândit să fie cât mai mic posibil, disponibil în toate versiunile de browsere importante existente, și să respecte filosofia "Unobtrusive JavaScript.

**3.Prezentarea componentei „Controller”**

In folder-ul „controllers” se află urmatoarele fișiere PHP: About.php, AjaxCalls.php, Clinics.php, ClinicsAdmin.php, ClinicsPage.php, Contact.php, DoctorsAdmin.php, Email.php, Index.php, IndexAdmin.php, NewsAdmin.php, Services.php, ShowNews.php, SpecsAdmin.php, Team.php, UploadClinics.php, UploadDoctors.php, UploadNews.php, UploadSpec.php. În fiecare dintre aceste fișiere există o clasă (ex. class About extends CI\_Controller{..} ) care prin intermediul constructorului încarcă model-ul corespunzător ($this->load->model('AboutModel');) și fișierul url\_helper. Prin intermediul funcției index (public function index()) se apelează fișierul care este în concordanța cu acesta(ex. about.php) din views și fișierele din ..views/templates (header, headerAdmin, head, footer si script).

**PHPMailer** este o bibliotecă de coduri care permite trimiterea (transportul) de e-mailuri în siguranță și cu ușurință prin intermediul codului PHP de la un server web ( MUA la serverul MSA ).

Trimiterea directă a e-mailurilor prin cod PHP necesită familiarizarea la nivel înalt a protocoalelor standard SMTP și a problemelor legate de acestea spam . Din 2001, PHPMailer este una dintre soluțiile populare pentru aceste probleme pe PHP .

# **Bibliografie**

* <https://unsplash.com/>
* <https://en.wikipedia.org/wiki/PHPMailer>
* <https://ro.wikipedia.org/wiki/HTML5>
* <https://ro.wikipedia.org/wiki/Cascading_Style_Sheets>
* <https://ro.wikipedia.org/wiki/JQuery>
* <https://ro.wikipedia.org/wiki/Programare_orientat%C4%83_pe_obiecte>
* <https://www.todaysoftmag.ro/article/1938/codeigniter-php-framework-o-unealta-simpla-si-utila>
* <https://ctrl-d.ro/editorial/de-ce-un-framework/>
* <http://stst.elia.pub.ro/news/IS/IS_Teme1101/mvc.pdf>