



PHP WEB DEVELOPMENT

**Aplicatie web pentru prezentare-
editare Portofoliu Personal**

Profesorul cursului:
Adrian Adiaconitei

Cursantul:
Cosmin Nicolae Gherghina
Data predarii
27.09.2021

Cuprins

Descriere Aplicatie Web.....	2
Tehnologiile utilizate in realizarea aplicatiei	2
Prezentare Aplicatie	3
Utilizarea Aplicatiei	4
Structura bazei de date si a aplicatiei	6
Principalele functionalizati ale Aplicatiei:	8

Aplicatie web pentru prezentare-editare Portofoliu Personal

1. Descriere Aplicatie Web

Aplicatia este realizata pentru a prezenta/updata portofoliu propriu, permite vizualizarea portofoliului personal pentru utilizatorii care sunt inregistrati, pentru adaugare/editare de rubrici noi din portofoliu prin operatii CRUD precum si a trimite un mesaj de contact.

Proiectul este creat cu ajutorul unui framework MVC, realizat de la inceput.

2. Tehnologiile utilizate in realizarea aplicatiei

In elaborarea proiectului am folosit urmatoarele tehnologii:

- HTML – pentru realizarea structurii site-ului
- CSS – pentru pozitionarea elementelor in pagina si aspectul general
- PHP OOP – pentru interactiunea cu baza de date
- MySQL PDO – pentru realizarea bazei de date
- MVC – pentru structura de a creea aplicatia

Testarea, rularea, si baza de date a aplicatiei s-a facut pe server Apache, instalat prin pachete WAMP

<http://localhost/cosminmvcla/>

Versionarea codului a fost realizata cu github

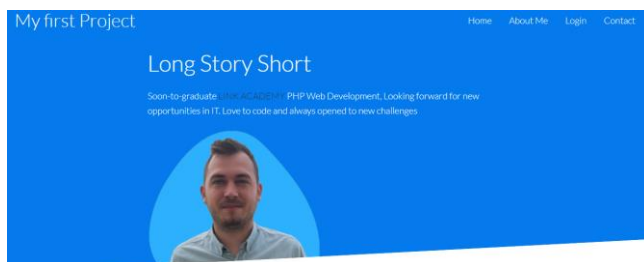
<https://github.com/cosming/cosminmvcla>

3. Prezentare aplicatie

Site-ul este realizat pentru a prezenta si updata portofoliul propriu; portofoliul poate fi vizualizat numai de utilizatorii care sunt logati, iar modificarile pot fi facute numai de contul de admin.

Vizitatorii pot avea acces la pagina de Home, About Me, Login, Register si Contact.

Vizitator



Utilizator logat



I. Formularul de Register

Campuri de completat:

- Username
- Email
- Password
- Confirm Password
- Un buton de redirectionare pentru formularul de **Login**
- Butonul de Register

Mesaje de avertizare sunt realizate in PHP pentru a ajuta utilizator in a completa corect formularul:

- Please enter name
- Please enter email
- Email is already taken
- Please enter password
- Password must be at least 6 characters
- Please confirm password
- Password do not match

Dupa inregistrarea utilizatorului acesta este redirectionat automat la pagina de Login

II. Formularul de Login

Campuri de completat:

- Email
- Password
- Un buton de redirectionare pentru formularul de **Register**
- Butonul de login

Mesaje de avertizare sunt realizate in PHP pentru a ajuta utilizator in a completa corect formularul:

- Please enter email
- Please enter password
- No user found
- Password incorrect

Dupa logarea utilizatorului este redirectionat automat la pagina de Home

Vizitatorii au acces la pagina de Home, About Me, My Profile, Log out si Contact

III. Posts

In pagina de Posts se regasesc cele 4 operatiuni pentru baza de date, CRUD, unde accesul il are admin-ul; postarile din Post pot fi vizualizate in My Profile.

Pagina de Posts contine un buton Add Post care duce catre un formular de adaugare a postarilor in baza de date; campuri de completat:

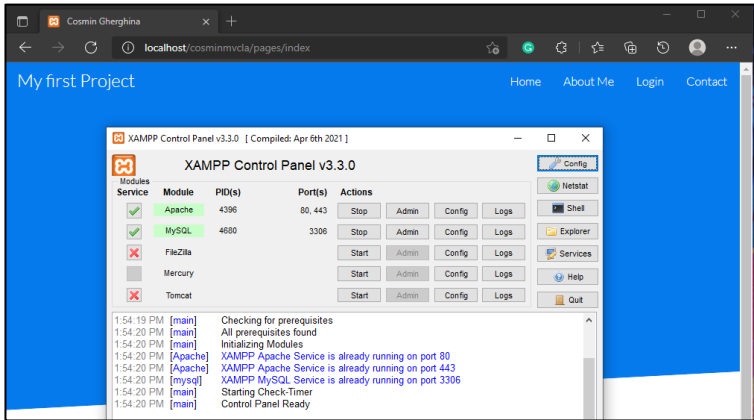
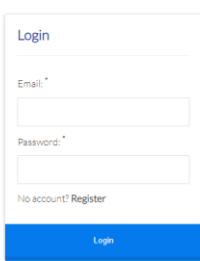
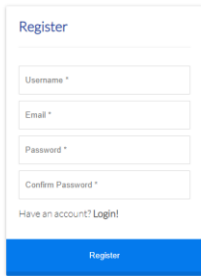
- Title
- Body
- Butonul de Submit

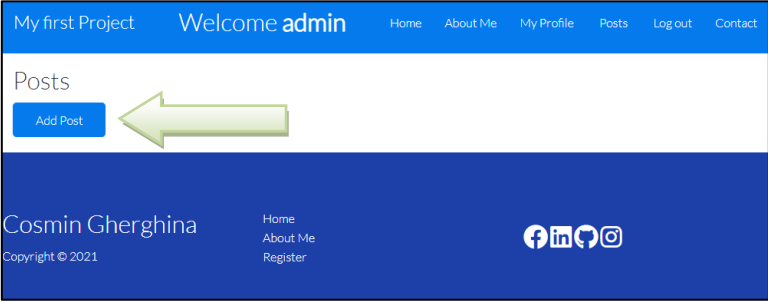
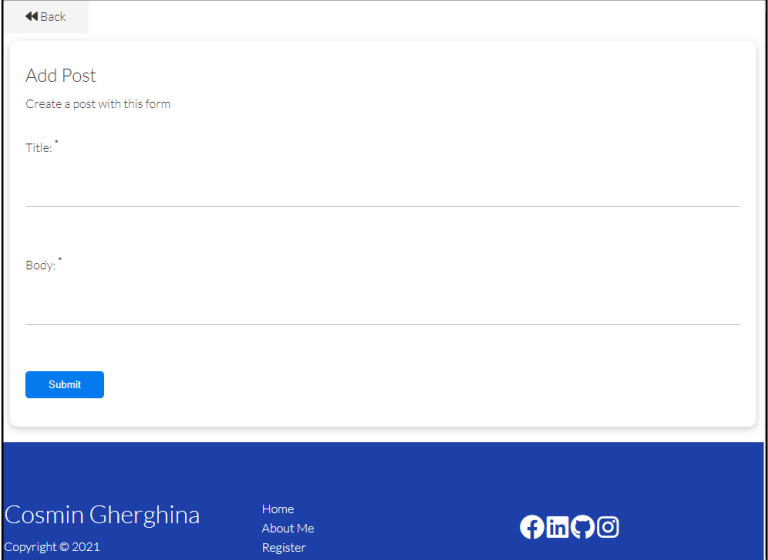
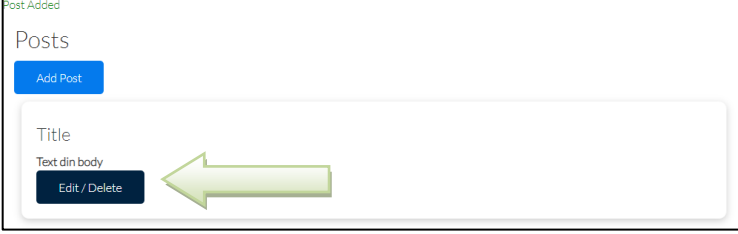

Mesaje de avertizare sunt realizate in PHP pentru a ajuta utilizator in a completa corect formularul:


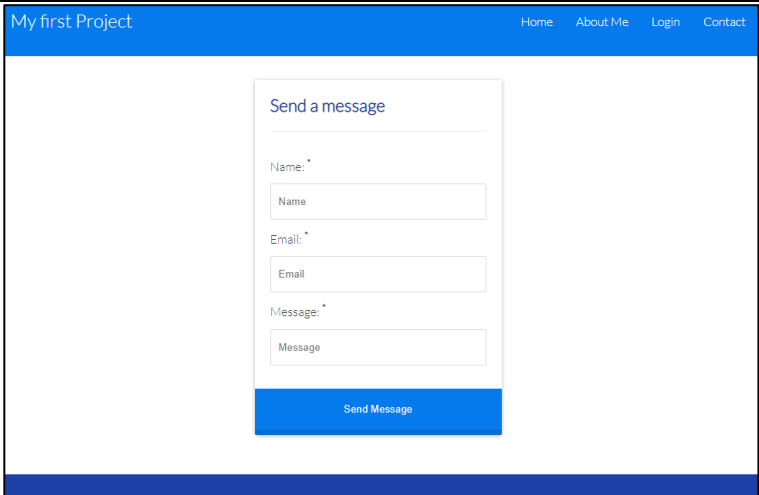
- Please enter title
- Please enter body text

Butonul de Edit/Delete care permite editarea sau stergerea postarii. Dupa terminarea operatiilor CRUD textul modificat este afisat atat in Posts cat si in pagina My Profile.

4. Utilizarea aplicatiei:

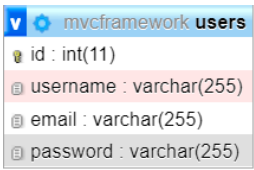
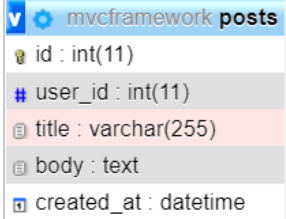
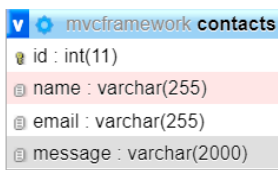
<i>Instructiuni</i>	<i>Imagine Aplicatie</i>
<p>Accesarea aplicatiei:</p> <p>Accesarea aplicatiei se face local dupa pornirea aplicatiei XAMPP si pornirea serviciului de Apache si MySQL http://localhost/cosminmvcla</p>	
<p>Logare/Inregistrare</p> <p>Se acceseaza pagina de Login din bara de navigatie si se completeaza campurile cu datele solicitate, daca nu exista cont se acceseaza butonul de Register care trimite utilizatorul catre pagina de Register pentru a-si crea cont</p>	<div data-bbox="721 1215 1118 1682">  </div> <div data-bbox="1127 1215 1533 1682">  </div>

<p>Adaugarea Postarilor</p> <p>Pagina de Posts poate fi accesata numai de utilizatorii care au drepturi de admin.</p>	 
<p>Editarea Postarilor</p> <p>Dupa adaugarea postului acesta va aparea in Posts cu un buton de Edit/Delete care permite modificarea si stergerea postarilor</p>	 

<p>Afisarea Postarilor</p> <p>Postarile sunt afisate in Posts si in pagina My Profile</p>	 <p>The screenshot shows a user profile for 'KnowBe4'. It includes an education section for 'University Politehnica Of Bucharest' with two entries: a Master's degree from Oct 2013 to Jul 2015 in 'Faculty of Material Science and Engineering' focusing on 'Obtaining, processing and characterizing metallic nanomaterials', and a Bachelor's degree from Oct 2009 to Jul 2013 in the same faculty, 'Department Medical Engineering'. Below this is a 'Work Experience' section with a 'Title' field and a 'Text din body' field, with a green arrow pointing to the latter.</p>
<p>Pagina de Contact</p> <p>Pagina de Contact contine un formular pentru a trimite un mesaj, aceasta poate fi accesata de toti utilizatorii aplicatiei</p>	 <p>The screenshot shows a contact page titled 'My first Project' with a navigation bar containing 'Home', 'About Me', 'Login', and 'Contact'. The main content area features a 'Send a message' form with fields for 'Name', 'Email', and 'Message', each with a '*' indicating it is required. A blue 'Send Message' button is at the bottom of the form.</p>

5. Structura bazei de date si a fisierelor:

a. Structura Bazei de date – Aplicatia foloseste 3 tabele

Users <i>Inregistrare si Logare</i>	Posts <i>Postari</i>	Contacts <i>Mesajele de contact</i>
 <p>The 'users' table schema shows: 'id' as an integer(11) with a primary key icon, 'username' as a varchar(255), 'email' as a varchar(255), and 'password' as a varchar(255).</p>	 <p>The 'posts' table schema shows: 'id' as an integer(11) with a primary key icon, 'user_id' as an integer(11) with a foreign key icon pointing to 'users', 'title' as a varchar(255), 'body' as a text field, and 'created_at' as a datetime field.</p>	 <p>The 'contacts' table schema shows: 'id' as an integer(11) with a primary key icon, 'name' as a varchar(255), 'email' as a varchar(255), and 'message' as a varchar(2000).</p>

b. Structura fisierelor

- ◆ cosminmvcla
 - app
 - config
 - config.php *parametrii pentru **DB+APPROOT+URLROOT+SITENAME***
 - controller
 - Contacts.php *logica pentru formularul de Contact*
 - Pages.php *logica pentru pagini si incarcarea view*
 - Posts.php *logica pentru postari*
 - Users.php *logica pentru utilizatori(inregistrare/logare/delogare)*
 - helpers
 - Sesion_helper.php *afisarea mesaje, si sesiunea daca utilizatorul este logat*
 - url_helper.php *functie de redirect pentru pagini*
 - libraries
 - Controller.php *logica pentru incarcare models si views*
 - Core.php *logica pentru creare si incarcare core controller*
 - Database.php *logica pentru interogari in baza de date*
 - Models
 - Contact.php *interogari la baza de date pentru contacts*
 - Post.php *interogari la baza de date pentru posts*
 - User.php *interogari la baza de date pentru users*
 - Views
 - contacts
 - ◆ Cont.php *pagina de contact*
 - inc
 - ◆ footer.php *footerul din pagini*
 - ◆ header.php *headerul din pagini*
 - ◆ navbar.php *bara de navigatie in aplicatie*
 - pages
 - ◆ about.php *pagina about*
 - ◆ index.php *pagina index/home page*
 - ◆ profile.php *pagina de my profile*
 - posts
 - ◆ add.php *pagina de adaugare post*
 - ◆ edit.php *pagina de editare post*
 - ◆ index.php *pagina de afisa a post-urilor*
 - ◆ show.php *pagina de editare/stergere a posturilor*
 - users
 - ◆ login.php *pagina de logare*
 - ◆ register.php *pagina de inregistrare*
 - bootstrap.php *autoloader*
 - public
 - css *fișierele css*
 - img *imaginile*
 - js *fișierele javascript*

6. Principalele functionalizati ale Aplicatiei:

1. Functionalitatea de Register:

```

public function register(){
    // Verifiare dupa metoda POST
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        // Curatarea datelor venite prin POST
        $_POST = filter_input_array(INPUT_POST, FILTER_SANITIZE_STRING);
        $data = [
            'username' => trim($_POST['username']),
            'email' => trim($_POST['email']),
            'password' => trim($_POST['password']),
            'confirmPassword' => trim($_POST['confirmPassword']),
            'usernameError' => "",
            'emailError' => "",
            'passwordError' => "",
            'confirmPasswordError' => "";
        ];
        // Validare Email
        if (empty($data['email'])) {
            $data['emailError'] = 'Pleae enter email';
        } else {
            // Check email
            if ($this->userModel->findUserByEmail($data['email'])) {
                $data['emailError'] = 'Email is already taken';
            }
        }
        // Validare username
        if (empty($data['username'])) {
            $data['usernameError'] = 'Pleae enter name';
        }
        // Validare parola
        if (empty($data['password'])) {
            $data['passwordError'] = 'Please enter password';
        } elseif (strlen($data['password']) < 6) {
            $data['passwordError'] = 'Password must be at least 6 characters';
        }
        // Validare confirmare parola
        if (empty($data['confirmPassword'])) {
            $data['confirmPasswordError'] = 'Please confirm password';
        } else {
            if ($data['password'] != $data['confirmPassword']) {
                $data['confirmPasswordError'] = 'Passwords do not match';
            }
        }
        // Ne asiguram ca erorile sunt goale
        if (empty($data['emailError']) && empty($data['usernameError']) && empty($data['passwordError']) &&
        empty($data['confirmPasswordError'])) {
            // Hash parola
            $data['password'] = password_hash($data['password'], PASSWORD_DEFAULT);
            // Register utilizator
            if ($this->userModel->register($data)) {
                flash('register_success', 'You are registered and can log in');
                redirect('users/login');
            } else {
                die('Something went wrong');
            }
        } else {
            // Incarcare Views cu erori
            $this->view('users/register', $data);
        }
    } else {
        $this->view('users/register', $data);
    }
}

```

2. Functionalitatea de Login

```
public function login()
{
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {

        $_POST = filter_input_array(INPUT_POST, FILTER_SANITIZE_STRING);

        $data = [
            'email' => trim($_POST['email']),
            'password' => trim($_POST['password']),
            'emailError' => "",
            'passwordError' => "",
        ];

        // Validare Email
        if (empty($data['email'])) {
            $data['emailError'] = 'Please enter email';
        }

        // Validare parola
        if (empty($data['password'])) {
            $data['passwordError'] = 'Please enter password';
        }

        // Verificare dupa email
        if ($this->userModel->findUserByEmail($data['email'])) {

        } else {
            // Daca userul nu este gasit aruncam urmatoarea eroare
            $data['emailError'] = 'No user found';
        }

        // Nu sunt erori
        if (empty($data['emailError']) && empty($data['passwordError'])) {

            // Logare utilizator
            $loggedInUser = $this->userModel->login($data['email'], $data['password']);

            if ($loggedInUser) {
                // Creare Sesiune de logat
                $this->createUserSession($loggedInUser);
            } else {
                $data['passwordError'] = 'Password incorrect';

                $this->view('users/login', $data);
            }
        } else {
            $this->view('users/login', $data);
        }
    } else {
        $data = [
            'email' => "",
            'password' => "",
            'emailError' => "",
            'passwordError' => "",
        ];
        $this->view('users/login', $data);
    }
}
```

3. Functionalitatea de creare sesiune

```
public function createUserSession($user)
{
    $_SESSION['user_id'] = $user->id;
    $_SESSION['user_email'] = $user->email;
    $_SESSION['user_name'] = $user->username;
    redirect('pages');
}
```

4. Functionalitatea de Logout

```
public function logout()
{
    unset($_SESSION['user_id']);
    unset($_SESSION['user_email']);
    unset($_SESSION['user_name']);
    session_destroy();
    redirect('users/login');
}
```

5. Functionalitatea de Adaugare Post

```
public function add()
{
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        // Curatare ce vine din posts
        $_POST = filter_input_array(INPUT_POST, FILTER_SANITIZE_STRING);
        $data = [
            'title' => trim($_POST['title']),
            'body' => trim($_POST['body']),
            'user_id' => $_SESSION['user_id'],
            'title_err' => '',
            'body_err' => ''
        ];
        // Validare date
        if (empty($data['title'])) {
            $data['title_err'] = 'Please enter title';
        }
        if (empty($data['body'])) {
            $data['body_err'] = 'Please enter body text';
        }
        // Nu sunt erori
        if (empty($data['title_err']) && empty($data['body_err'])) {
            if ($this->postModel->addPost($data)) {
                flash('post_message', 'Post Added');
                redirect('posts');
            } else {
                die('Something went wrong');
            }
        } else {
            $this->view('posts/add', $data);
        }
    } else {
        $data = [
            'title' => '',
            'body' => ''
        ];
        $this->view('posts/add', $data);
    }
}
```

6. Functionalitatea de Editare post

```
public function edit($id)
{
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        $_POST = filter_input_array(INPUT_POST, FILTER_SANITIZE_STRING);

        $data = [
            'id' => $id,
            'title' => trim($_POST['title']),
            'body' => trim($_POST['body']),
            'user_id' => $_SESSION['user_id'],
            'title_err' => '',
            'body_err' => ''
        ];

        if (empty($data['title'])) {
            $data['title_err'] = 'Please enter title';
        }
        if (empty($data['body'])) {
            $data['body_err'] = 'Please enter body text';
        }

        if (empty($data['title_err']) && empty($data['body_err'])) {
            // Validated
            if ($this->postModel->updatePost($data)) {
                flash('post_message', 'Post Updated');
                redirect('posts');
            } else {
                die('Something went wrong');
            }
        } else {
            $this->view('posts/edit', $data);
        }
    } else {
        // Preluare posts din Model
        $post = $this->postModel->getPostById($id);
        // Verificare sesiune User
        if ($post->user_id != $_SESSION['user_id']) {
            redirect('posts');
        }
        $data = [
            'id' => $id,
            'title' => $post->title,
            'body' => $post->body
        ];
        $this->view('posts/edit', $data);
    }
}
```

7. Functionalitatea de Stergere Post

```
public function delete($id) {
    if ($_SERVER['REQUEST_METHOD'] == 'POST') {
        $post = $this->postModel->getPostById($id);
        if ($post->user_id != $_SESSION['user_id']) {
            redirect('posts');
        }
        if ($this->postModel->deletePost($id)) {
            flash('post_message', 'Post Removed');
            redirect('posts');
        } else {
            die('Something went wrong');
        }
    } else {
        redirect('posts');
    }
}
```