

Universitatea Tehnică a Moldovei
Facultatea *Calculatoare, Informatică și Microelectronică*
Specialitatea *Tehnologii Informaționale*



Raport

la lucrarea de laborator nr. 4

Tema: “*Utilizarea limbajului PHP pentru generarea conținutului dinamic pentru aplicația Web. Conectare la baza de date MySQL. Lucrul cu fișiere*”

Disciplina: “Tehnologii web”

A efectuat:

Student grupa TI-231 FR

Apareci Aurica

A verificat:

Asistent universitar

Rusu Viorel

Chișinău 2025

Cuprins

1. Cadru teoretic.....	3
2. Repere teoretice	4
3. Sarcini practice.....	5
5. Concluzii.....	8
6. Webografie.....	9

1. Cadru teoretic

Tema lucrării: Utilizarea limbajului PHP pentru generarea conținutului dinamic pentru aplicația Web.
Conectare la baza de date MySQL. Lucrul cu fișiere

Sarcina practică:

- a) se continua tema din lab.1, unele pagini fiind transformate astfel incit sa fie create dinamic
- b) trebuie sa contina functionalitati ca conectare la BD, gestionarea informatiei din citeva tebele (citire, scriere, modificare, stergere)
- d) exersati citirea/scrierea in fisiere prin implementarea unor exemple, chiar daca in logica site-ului nu e nevoie.

2. Repere teoretice

PHP (Hypertext Preprocessor) este un limbaj de programare server-side, special conceput pentru dezvoltarea aplicațiilor web dinamice. Acesta permite inserarea logicii de programare direct în paginile HTML, oferind posibilitatea de a genera conținut personalizat în funcție de utilizator, date sau acțiuni. Prin utilizarea PHP, paginile web nu mai sunt statice, ci pot răspunde în timp real la interacțiunea utilizatorului, afișând informații preluate dintr-o bază de date, rezultatele unui formular sau conținut actualizat automat.

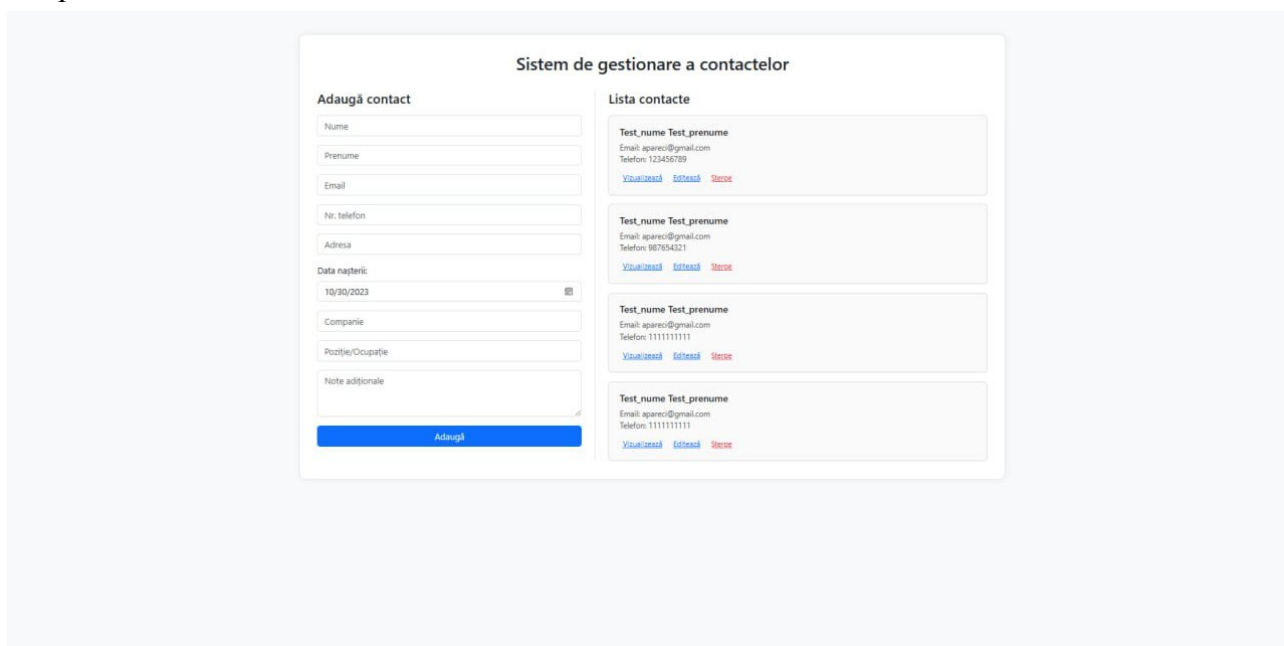
MySQL este un sistem de gestiune a bazelor de date relaționale (RDBMS) utilizat frecvent în aplicațiile web. PHP oferă suport nativ pentru conectarea și interacțiunea cu baze de date MySQL, prin extensii precum mysqli sau PDO. Conectarea presupune stabilirea unei legături între aplicația PHP și serverul MySQL, folosind un set de credențiale (nume de utilizator, parolă, nume bază de date). Ulterior, aplicația poate executa interogări SQL pentru a adăuga, modifica, șterge sau prelua date în mod dinamic, în funcție de logica aplicației.

PHP permite manipularea fișierelor de pe server printr-o serie de funcții dedicate. Aceste funcții oferă posibilitatea de a crea, citi, scrie, modifica sau șterge fișiere, facilitând stocarea de date în afara bazei de date, logarea activităților sau generarea de fișiere temporare. Prin utilizarea funcțiilor precum `fopen()`, `fread()`, `fwrite()` sau `fclose()`, dezvoltatorul poate gestiona eficient fluxul de informații stocate în fișiere. De asemenea, PHP permite verificarea existenței unui fișier (`file_exists()`) sau citirea conținutului său complet (`file_get_contents()`).

Aplicațiile web dinamice au avantajul de a oferi o experiență personalizată și interactivă utilizatorilor. Prin integrarea dintre PHP și MySQL, datele sunt gestionate eficient, iar interfețele sunt adaptate nevoilor fiecărui utilizator. De exemplu, un sistem de autentificare, un panou de administrare sau un catalog de produse sunt exemple comune de componente dinamice ce necesită generarea de conținut în funcție de datele din baza de date și acțiunile utilizatorului.

3. Sarcini practice

În cadrul acestei lucrări de laborator, am realizat o aplicație web simplă pentru gestionarea unei liste de contacte, utilizând limbajul PHP pentru generarea conținutului dinamic. Aplicația permite utilizatorului să adauge, vizualizeze, editeze și să șteargă contacte, oferind o interfață intuitivă și funcțională. Datele introduse de utilizator sunt salvate atât într-o bază de date MySQL, pentru o gestionare eficientă și structurată a informațiilor, cât și într-un fișier text, ca metodă alternativă de stocare și backup. Prin această abordare, s-a demonstrat utilizarea combinată a tehnologiilor web pentru dezvoltarea unei aplicații dinamice, cu funcționalități de manipulare a datelor și persistență multiplă.



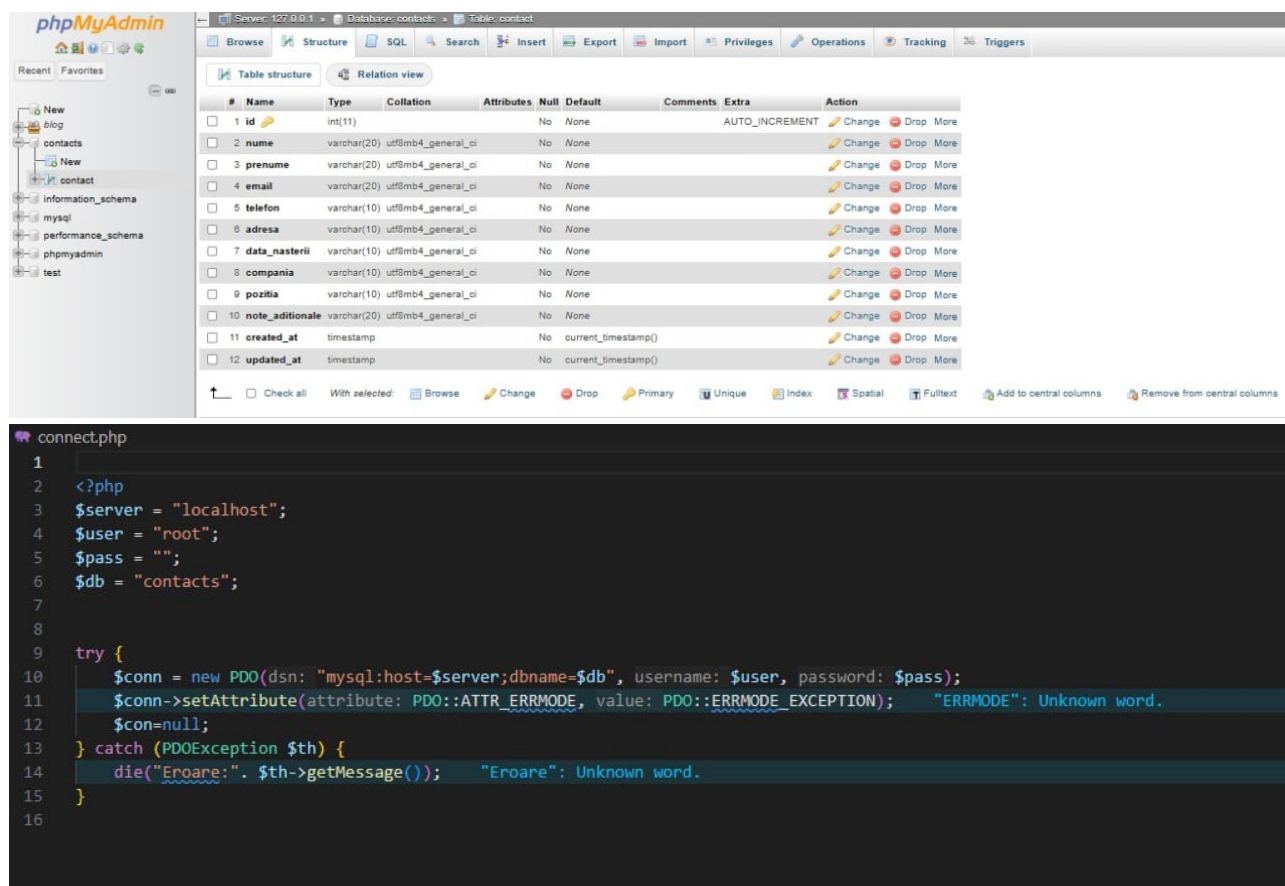
Adaugă contact	
Nume	<input type="text"/>
Prenume	<input type="text"/>
Email	<input type="text"/>
Nr. telefon	<input type="text"/>
Adresa	<input type="text"/>
Data nașterii	<input type="text" value="10/20/2023"/>
Companie	<input type="text"/>
Poziție/Ocupație	<input type="text"/>
Note adiționale	<input type="text"/>
<input type="button" value="Adaugă"/>	

Lista contacte	
Test_nume Test_prenume Email: apareci@gmail.com Telefon: 123456789	Vizualizează Editează Șterge
Test_nume Test_prenume Email: apareci@gmail.com Telefon: 987654321	Vizualizează Editează Șterge
Test_nume Test_prenume Email: apareci@gmail.com Telefon: 111111111	Vizualizează Editează Șterge
Test_nume Test_prenume Email: apareci@gmail.com Telefon: 111111111	Vizualizează Editează Șterge

Pentru implementarea aplicației de gestionare a contactelor, a fost creată o bază de date numită *contacts*, în care se află tabela *contact*. Această tabelă conține câmpuri esențiale pentru stocarea informațiilor despre fiecare contact: id (cheie primară, auto-incrementabilă), nume, prenume, email, telefon, adresa, data_nasterii, compania, pozitia, note_aditionale, precum și două câmpuri de tip timestamp, *created_at* și *updated_at*, care înregistrează automat momentul creării și modificării fiecărei înregistrări. Fiecare câmp este de tip VARCHAR, adecvat pentru stocarea datelor textuale, cu dimensiuni alese corespunzător pentru tipul de informație. Structura bazei de date a fost realizată cu ajutorul interfeței phpMyAdmin, asigurând o organizare clară și eficientă a datelor.

Pentru realizarea conexiunii între aplicația PHP și baza de date MySQL, a fost creat un fișier denumit *connect.php*, în care s-a utilizat extensia PDO (PHP Data Objects) pentru o conexiune sigură și flexibilă. În acest fișier, au fost definite variabilele necesare (server, utilizator, parolă și nume bază de date), iar ulterior a fost instanțiat un obiect PDO folosind aceste informații. În plus, s-a setat modul de raportare a erorilor la *ERRMODE_EXCEPTION*, permițând tratarea acestora

prin blocuri try-catch, ceea ce facilitează identificarea și gestionarea eventualelor probleme de conectare. Astfel, s-a asigurat o interacțiune corectă și controlată cu baza de date.



The image shows a screenshot of the phpMyAdmin interface. The top part displays the 'Table structure' view for the 'contact' table. The table has 12 columns: id, nume, prenume, email, telefon, adresa, data_nasterii, compania, pozitia, note_aditionale, created_at, and updated_at. The bottom part shows a PHP code snippet in 'connect.php' that establishes a database connection using PDO and handles exceptions.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	nume	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
3	prenume	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
4	email	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
5	telefon	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
6	adresa	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
7	data_nasterii	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
8	compania	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
9	pozitia	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More
10	note_aditionale	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
11	created_at	timestamp			No	current_timestamp()			Change Drop More
12	updated_at	timestamp			No	current_timestamp()			Change Drop More

```
1
2
3 <?php
4 $server = "localhost";
5 $user = "root";
6 $pass = "";
7 $db = "contacts";
8
9
10 try {
11     $conn = new PDO(dsn: "mysql:host=$server;dbname=$db", username: $user, password: $pass);
12     $conn->setAttribute(attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION); "ERRMODE": Unknown word.
13     $con=null;
14 } catch (PDOException $th) {
15     die("Eroare:". $th->getMessage()); "Eroare": Unknown word.
16 }
```

Fișierul view.php are rolul de a gestiona atât afișarea detaliilor unui contact existent în baza de date, cât și preluarea datelor introduse într-un formular pentru actualizare. Codul este structurat astfel încât să trateze separat cererile de tip GET și POST, în funcție de acțiunea utilizatorului. În cazul unei cereri GET, este extras parametrul id din URL, apoi este pregătită și executată o interogare SQL care selectează toate informațiile asociate acelui contact din tabela contact. Rezultatul interogării este stocat într-un array asociativ, iar dacă nu este găsit niciun rezultat, utilizatorul este redirecționat către pagina index.php.

În cazul unei cereri POST, scriptul preia datele completate de utilizator în formularul HTML, corespunzătoare câmpurilor nume, prenume, email, telefon, adresa, data_nasterii, compania, pozitia și note_aditionale. Aceste date sunt salvate în variabile PHP, urmând a fi utilizate (cel mai probabil în continuare, într-un bloc de actualizare) pentru modificarea înregistrării existente în baza de date. La final, utilizatorul este redirecționat înapoi la pagina principală. Acest fișier ilustrează un flux tipic de preluare și procesare a datelor într-o aplicație web PHP, în care interacțiunea cu baza de date este realizată prin PDO și este adaptată în funcție de metoda de cerere HTTP utilizată.

```

1 <?php
2 include 'connect.php';
3 if($_SERVER["REQUEST_METHOD"]=="GET"){
4     $id=$_GET["id"];
5
6     $sql="SELECT * FROM contact WHERE id=:id";
7     $stmt=$conn->prepare(query: $sql);
8     $stmt->bindParam(param: ":id",var: &$id);
9     $stmt->execute();
10    $user=$stmt->fetch(mode: PDO::FETCH_ASSOC);
11
12    if(!$user){
13        header(header: "Location:index.php");
14    }
15 }
16 if($_SERVER["REQUEST_METHOD"]=="POST"){
17     $id=$_POST["id"];
18     $nume=$_POST["nume"];    "nume": Unknown word.
19     $prenume=$_POST["prenume"];    "prenume": Unknown word.
20     $email=$_POST["email"];
21     $telefon=$_POST["telefon"];    "telefon": Unknown word.
22     $adresa=$_POST["adresa"];    "adresa": Unknown word.
23     $data_nasterii=$_POST["data_nasterii"];    "nasterii": Unknown word.
24     $compania=$_POST["compania"];    "compania": Unknown word.
25     $pozitie=$_POST["pozitie"];    "pozitie": Unknown word.
26     $note_aditionale=$_POST["note_aditionale"];    "aditionale": Unknown word.
27
28     header(header: "Location: index.php");
29 }
30 ?>

```

```

<body>
<div class="container my-5">
<div class="main-container">
<div class="row">
<div class="col-md-7 ps-4">
<h4 class="mb-3">Lista contacte</h4>    "Lista": Unknown word.
<div class="contact-list">
<?php
include 'connect.php';
$sql = "SELECT * FROM contact";
$stmt = $conn->query(query: $sql);
$results = $stmt->fetchAll(mode: PDO::FETCH_ASSOC);
$conn = null;
?>
<?php foreach ($results as $row): ?>
<div class="contact-card">
<div class="contact-header"><?> $row['nume'] . ' ' . $row['prenume'] ?></div>    "nume": Unknown word.
<div class="contact-details">
Email: <?> $row['email'] ?><br>
Telefon: <?> $row['telefon'] ?>    "Telefon": Unknown word.
</div>
<div class="contact-actions">
<a href="view.php?id=?> $row['id'] ?>" class="contact-btn">Vizualizează</a>    "Vizualizează": Unknown word.
<a href="edit.php?id=?> $row['id'] ?>" class="contact-btn">Editează</a>    "Editează": Unknown word.
<a href="delete.php?id=?> $row['id'] ?>" class="contact-btn text-danger">Șterge</a>    "Șterge": Unknown word.
</div>
</div>
<?php endforeach; ?>
</div>
</div>

```

Pentru a asigura o metodă alternativă de stocare și urmărire a datelor, aplicația implementează și scrierea informațiilor într-un fișier text denumit *contacte_log.txt*. La fiecare adăugare sau modificare a unui contact, datele relevante, precum numele, prenumele, emailul, telefonul, adresa și alte câmpuri, sunt concatenate sub formă de text și înregistrate în acest fișier, separate prin delimitatori pentru lizibilitate. Acest proces se realizează utilizând funcțiile PHP de lucru cu fișiere, cum ar fi `fopen()`, `fwrite()` și `fclose()`, permițând astfel păstrarea unui jurnal simplu cu toate modificările efectuate asupra contactelor, util atât în scopuri de backup, cât și pentru trasabilitate.

```

FOLDERS: LABORATOR_4
edit.php    contacte_log.txt x
add.php
connect.php
contacte_log.txt
delete.php
edit.php
index.php
view.php

```

```

1 Nume: Test_nume | Prenume: Test_prenume | Email: apareci@gmail.com | Telefon: 987654321 | Adresa: test_adresa | Da
2

```

5. Concluzii

În concluzie, această lucrare de laborator a demonstrat modul în care limbajul PHP poate fi utilizat eficient pentru dezvoltarea unei aplicații web dinamice, integrând atât interacțiunea cu o bază de date MySQL, cât și manipularea fișierelor. Prin implementarea aplicației pentru gestionarea contactelor, s-a evidențiat capacitatea PHP de a prelucra cereri de tip GET și POST, de a gestiona conținutul în mod dinamic și de a asigura persistența datelor într-un mod sigur și structurat. Utilizarea extensiei PDO a oferit un nivel ridicat de securitate și flexibilitate în lucrul cu baza de date, în timp ce salvarea informațiilor într-un fișier a permis realizarea unei soluții de backup simplă. Această aplicație a contribuit la înțelegerea practică a modului în care se construiesc aplicații web care răspund cerințelor reale de stocare și gestionare a datelor utilizatorilor.

6. Webografie

1. <https://else.fcim.utm.md/course/view.php?id=2318>
2. <https://www.php.net/>
3. <https://www.w3schools.com/js/DEFAULT.asp>
4. <https://chatgpt.com/>