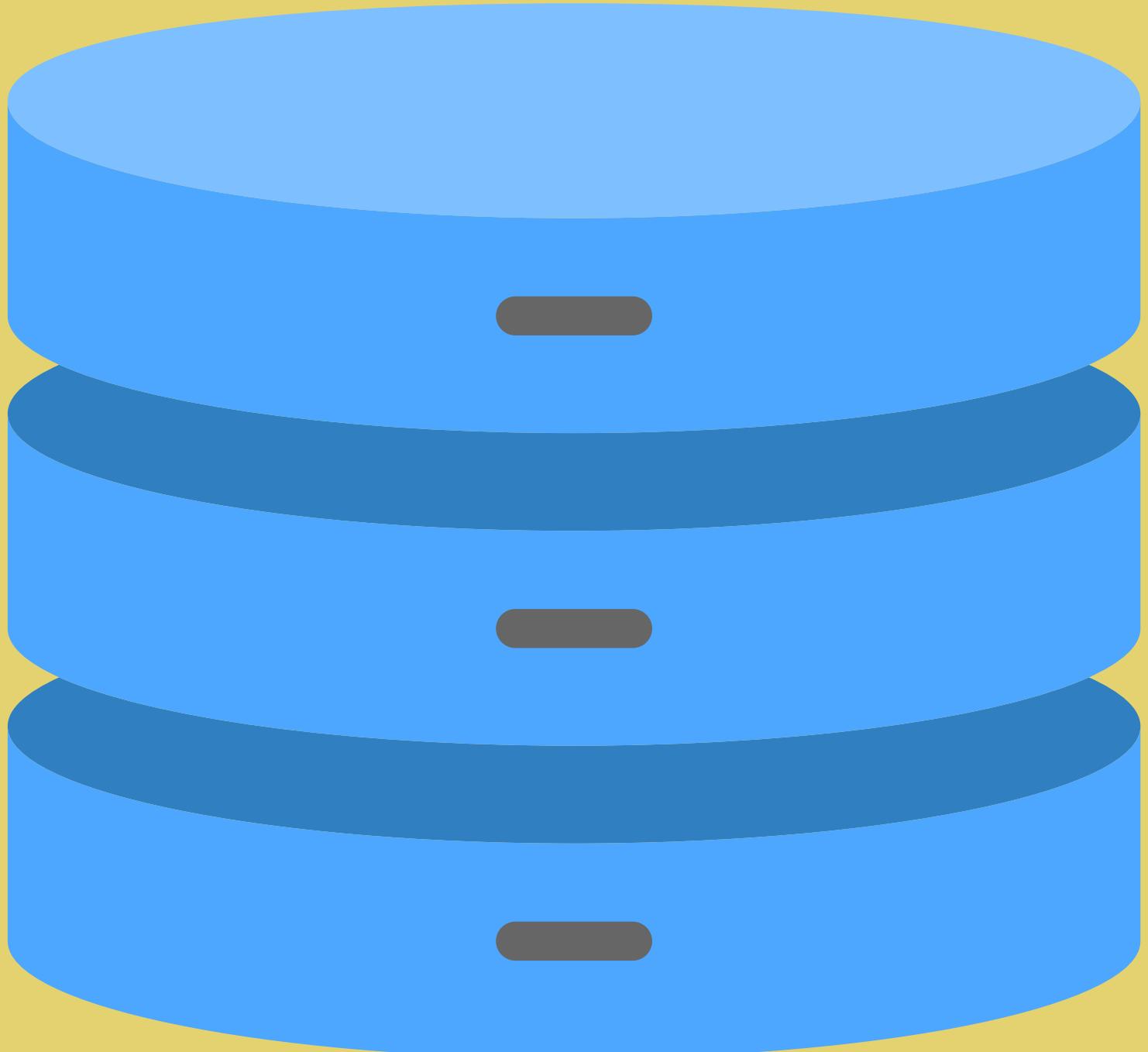


Laboratorul 6

Tehnologii Web

Baze de date in PHP



Obiectivele laboratorului



**XAMPP and
phpMyAdmin**



MySQLi



PDO



CRUD in DB

XAMPP and phpMyAdmin



Control Panel

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]

XAMPP Control Panel v3.2.2

Service	Module	PID(s)	Port(s)	Actions
	Apache	6384 21924	80, 443	Stop Admin Config Logs
	MySQL	14168	3306	Stop Admin Config Logs
	FileZilla			Start Admin Config Logs
	Mercury			Start Admin Config Logs
	Tomcat			Start Admin Config Logs

[Config](#) [Netstat](#) [Shell](#) [Explorer](#) [Services](#) [Help](#) [Quit](#)

```
1:21:15 PM [main] Initializing Control Panel
1:21:15 PM [main] Windows Version: Enterprise 64-bit
1:21:15 PM [main] XAMPP Version: 7.2.9
1:21:15 PM [main] Control Panel Version: 3.2.2 [ Compiled: Nov 12th 2015 ]
1:21:15 PM [main] You are not running with administrator rights! This will work for
most application stuff but whenever you do something with services
there will be a security dialogue or things will break! So think
about running this application with administrator rights!
1:21:15 PM [main] XAMPP Installation Directory: "s:\xampp\" 
1:21:15 PM [main] Checking for prerequisites
1:21:16 PM [main] All prerequisites found
1:21:16 PM [main] Initializing Modules
1:21:16 PM [main] Starting Check-Timer
1:21:16 PM [main] Control Panel Ready
1:21:18 PM [Apache] Attempting to start Apache app...
1:21:18 PM [Apache] Status change detected: running
1:21:20 PM [mysql] Attempting to start MySQL app...
1:21:21 PM [mysql] Status change detected: running
```

phpMyAdmin

phpMyAdmin demo - MySQL

Databases SQL Status Users Export Import Settings Replication VariablesCharsets Engines

You are using the demo server. You can do anything here, but please do not change root, debian-sys-maint and pma users. More information is available at demo.phpmyadmin.net.

General Settings

Change password

Server connection collation: utf8mb4_general_ci

Appearance Settings

Language: English

Theme: pmahomme

Font size: 82%

More settings

Web server

- nginx/1.2.1
- Database client version: libmysql - 5.5.40
- PHP extension: mysqli
- Show PHP information

phpMyAdmin

- Version information: 4.3.0 (up to date)
- Git revision: from branch, committed on Jan 01, 1970 at 01:00 AM by
- Documentation
- Wiki
- Official Homepage
- Contribute
- Get support
- List of changes

Console

Press Ctrl+Enter to execute query

Bookmarks Options History Clear

Recent Favorites

Filter databases by name or regex X

New

- #
- airlineflights
- Alus
- aqlegendary
- aseguradora
- BD2
- br
- christmasevents
- CSV_DB
- Databaseregister
- dbname
- DBO
- demo
- dragonb
- ducon
- elostudios
- Ganges
- hamza
- hamza.kuri
- information_schema
- ITEC3020
- kekec
- Koledj_BNV

SQL scripts

```
-- Create database
CREATE DATABASE IF NOT EXISTS lab7;

-- Create user with privileges for lab7 database
CREATE USER 'php'@'localhost' IDENTIFIED BY 'php';
GRANT ALL PRIVILEGES ON lab7.* TO 'php'@'localhost';
FLUSH PRIVILEGES;

-- Switch to lab7 database
USE lab7;

-- Create users table
CREATE TABLE IF NOT EXISTS users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL
);
```

MySQLi

```
// Database connection parameters
$servername = "localhost";
$username = "lab7";
$password = "php";
$database = "lab7";

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
```

MySQLi

```
// Database connection parameters
$servername = "localhost";
$username = "lab7";
$password = "php";
$database = "lab7";

// Create connection
$conn = new mysqli($servername, $username, $password, $database);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// Close connection
$conn->close();
```

SQL Injection

```
// Sanitize and retrieve username from GET parameter
$username = $_GET['name'];

//SQL Injection
//name=' OR '1'='1
// Construct SQL query
$sql = "SELECT * FROM users WHERE name = '$username';

// Execute the query
$result = $conn->query($sql);
```

PDO

```
// Establish database connection using PDO
$dsn = "mysql:host=".DB_SERVER.";dbname=".DB_NAME;
$pdo = new PDO($dsn, DB_USERNAME, DB_PASSWORD);

// Set PDO error mode to exception
$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

// Construct SQL query
$sql = "SELECT * FROM users WHERE name = :username";
```

PDO

```
// Prepare the SQL statement
$stmt = $pdo->prepare($sql);

// Bind parameter
$stmt->bindParam(':username', $username, PDO::PARAM_STR);

// Execute the prepared statement
$stmt->execute();
```

PDO

```
// Fetch all rows from the result set
$rows = $stmt->fetchAll(PDO::FETCH_ASSOC);

// Check if user exists
if ($rows) {
    // Loop through the rows
    foreach ($rows as $row) {
        // Output user details
        echo "User ID: " . $row['id'] . "<br>";
        echo "Name: " . $row['name'] . "<br>";
        echo "Email: " . $row['email'] . "<br>";
        echo "<br>";
    }
}
```

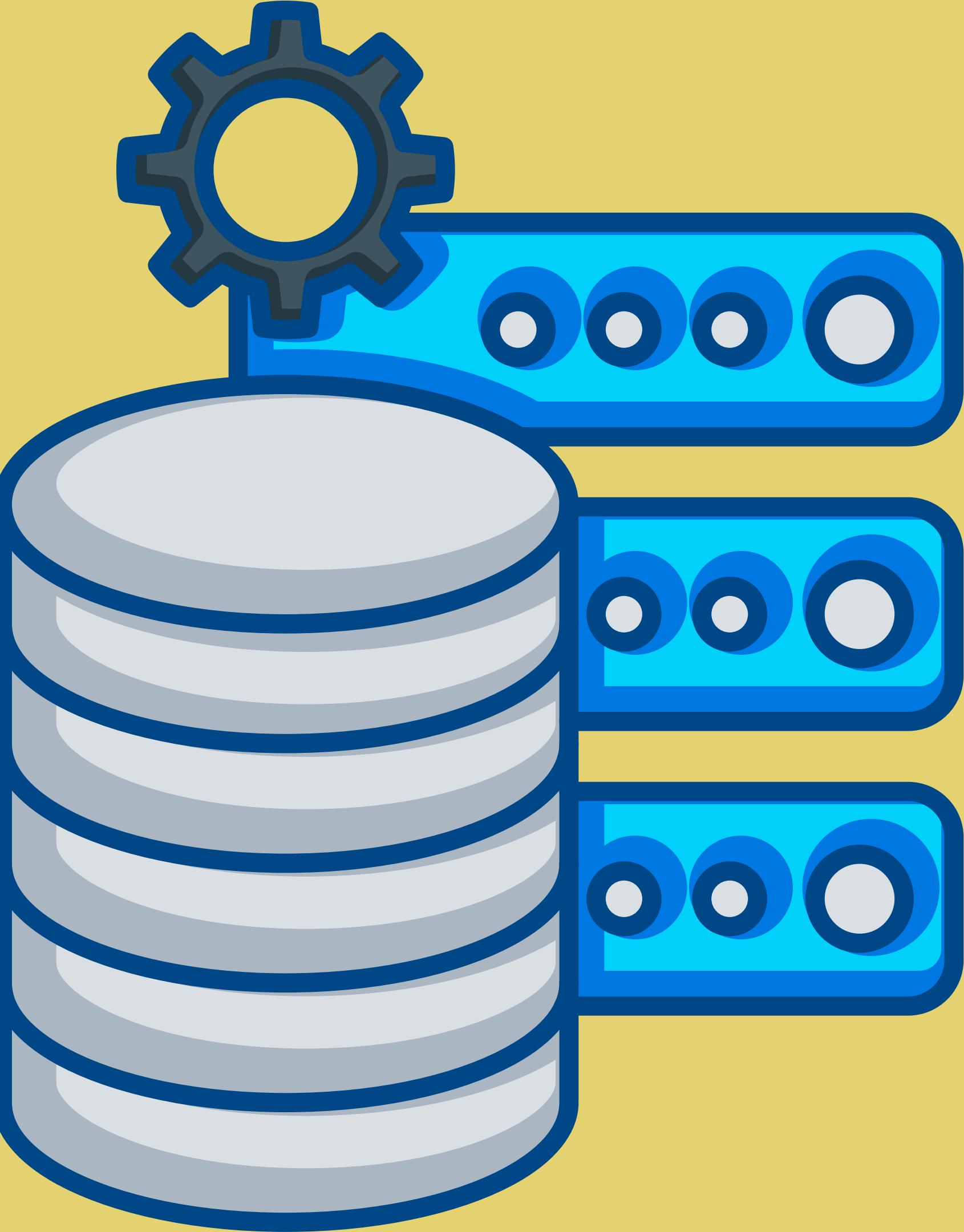
DB config file

```
<?php  
// Database configuration constants  
define('DB_SERVER', 'localhost');  
define('DB_USERNAME', 'lab7');  
define('DB_PASSWORD', 'php');  
define('DB_NAME', 'lab7');  
?>  
  
// Include database configuration  
require_once 'config.php';
```

DB config file

```
<?php  
// Database configuration constants  
define('DB_SERVER', 'localhost');  
define('DB_USERNAME', 'lab7');  
define('DB_PASSWORD', 'php');  
define('DB_NAME', 'lab7');  
?>  
  
// Include database configuration  
require_once 'config.php';
```

CRUD operations



CRUD

```
public function getUserId($id) {
    try {
        $sql = "SELECT * FROM users WHERE id = :id";
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(':id', $id, PDO::PARAM_INT);
        $stmt->execute();
        $user = $stmt->fetch(PDO::FETCH_ASSOC);
        return $user;
    } catch (PDOException $e) {
        echo "Error fetching user: " . $e->getMessage();
        return null;
    }
}

public function addUser($name, $email) {
    try {
        $sql = "INSERT INTO users (name, email) VALUES (:name, :email)";
        $stmt = $this->pdo->prepare($sql);
        $stmt->bindParam(':name', $name, PDO::PARAM_STR);
        $stmt->bindParam(':email', $email, PDO::PARAM_STR);
        $stmt->execute();
        return true;
    } catch (PDOException $e) {
        echo "Error adding user: " . $e->getMessage();
        return false;
    }
}
```

CRUD

```
<h2>User Details</h2>
<?php
// Check if user ID is provided in the URL
if (isset($_GET['id'])) {
    // Get user ID from URL
    $userId = $_GET['id'];

    // Instantiate User class
    $user = new User();

    // Get user details by ID
    $userDetails = $user->getUserById($userId);

    if ($userDetails) {
        echo "<p><strong>ID:</strong> {$userDetails['id']}</p>";
        echo "<p><strong>Name:</strong> {$userDetails['name']}</p>";
        echo "<p><strong>Email:</strong> {$userDetails['email']}</p>";
        echo "<br><a href='users.php'>&laquo; Back to Users</a>";
    } else {
        echo "User not found.";
    }
} else {
    echo "User ID not provided.";
}
```

Exerciții

**“LEARNING NEVER
EXHAUSTS THE MIND.”**

— Leonardo da Vinci