A green and white logo

Description automatically generated

Department of Computer Science and Mathematics

CSC 443: Web Programming

Documentation of the

Blog Platform Website

(Final Project)

Dr. Shant Estepan

**A purple sign with white text

Description automatically generated**

Student Names:

Abdulaziz Al Sayyed (ID=202204622)

Abdullah Jrad (ID=202207733)

Lana Saab (ID=202201538)

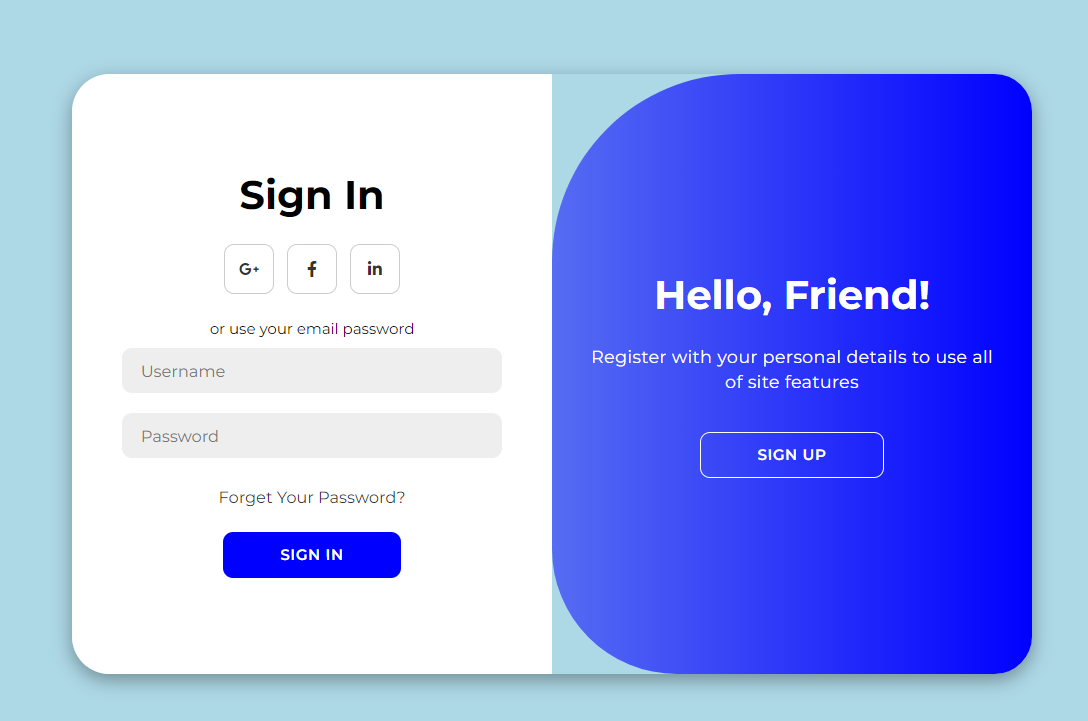
Nireez Al-Sweidan (ID=202205831)

**Submitted on May 1, 2024**

**Introduction:**

This project report details the development of a Blog Platform named “Creative Hub” and which is designed to facilitate interactive content creation and management. The platform features a robust user registration and authentication system, allowing users to securely create accounts and log in to access various functionalities. It supports comprehensive blog post management tools that enable users to create, edit, delete, and publish blog posts, complete with rich text formatting, images, and categorization options. Additionally, an administrative dashboard is implemented to provide administrators with the necessary tools to manage user accounts and oversee blog post activities efficiently. This report outlines the key functionalities, technical implementations, and user interactions within the platform.

**Registration Page:**

 A screenshot of a login form

Description automatically generated

You can sign up to create an account or sign in to your account.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.2/css/all.min.css">

    <title>Login Page</title>

    <style>

        @import url('https://fonts.googleapis.com/css2?family=Montserrat:wght@300;400;500;600;700&display=swap');

/\* /\* CSS Styles \*/

    </style>

</head>

<body>

    <div class="container" id="container" >

        <div class="form-container sign-up">

            <form id="signup-form" action="signupprocess.php" method="post">

                <h1>Create Account</h1>

                <div class="social-icons">

                    <a href="#" class="icon"><i class="fa-brands fa-google-plus-g"></i></a>

                    <a href="#" class="icon"><i class="fa-brands fa-facebook-f"></i></a>

                    <a href="#" class="icon"><i class="fa-brands fa-linkedin-in"></i></a>

                </div>

                <span>or use your email for registeration</span>

                <input type="text" id="username" name="username" placeholder="Username" required>

                <input type="email" id="email" name="email" placeholder="Email" required>

                <input type="password" id="password" name="password" placeholder="Password" required>

                <button type="submit">Sign Up</button>

            </form>

        </div>

        <div class="form-container sign-in">

            <form id="login-form" action="loginprocess.php" method="post">

              <h1>Sign In</h1>

              <div class="social-icons">

                <a href="#" class="icon"><i class="fa-brands fa-google-plus-g"></i></a>

                <a href="#" class="icon"><i class="fa-brands fa-facebook-f"></i></a>

                <a href="#" class="icon"><i class="fa-brands fa-linkedin-in"></i></a>

              </div>

              <span>or use your email password</span>

              <input type="text" id="username" name="username" placeholder="Username" required>

              <input type="password" id="password" name="password" placeholder="Password" required>

              <a href="#">Forget Your Password?</a>

              <button type="submit">Sign In</button>

            </form>

          </div>

        <div class="toggle-container">

            <div class="toggle">

                <div class="toggle-panel toggle-left">

                    <h1>Welcome Back!</h1>

                    <p>Enter your personal details to use all of site features</p>

                    <button class="hidden" id="login">Sign In</button>

                </div>

                <div class="toggle-panel toggle-right">

                    <h1>Hello, Friend!</h1>

                    <p>Register with your personal details to use all of site features</p>

                    <button class="hidden" id="register">Sign Up</button>

                </div>

            </div>

        </div>

    </div>

    <script>

        // Wait until the DOM is fully loaded

        document.addEventListener('DOMContentLoaded', function () {

            // Get the container element and both buttons

            const container = document.getElementById('container');

            const registerBtn = document.getElementById('register');

            const loginBtn = document.getElementById('login');

            // Add 'active' class when the register button is clicked

            registerBtn.addEventListener('click', () => {

                container.classList.add("active");

            });

            // Remove 'active' class when the login button is clicked

            loginBtn.addEventListener('click', () => {

                container.classList.remove("active");

            });

        });

    </script>

</body>

</html>

Signup process:

<?php

$servername= "localhost";

$usernamee = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $usernamee, $password, $database, "3308");

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$usernamee = "";

if ($\_SERVER["REQUEST\_METHOD"]=="POST") {

    $username=$\_POST['username'];

    $email=$\_POST['email'];

    $pass=$\_POST['password'];

    $sql = "INSERT INTO users(username, password, email, role) VALUES('$username','$pass','$email','user')";

    if ($conn->query($sql) === TRUE) {

        echo "New record created successfully";

    } else {

        echo "Error: " . $sql . "<br>" . $conn->error;

    }

    header("Location: registration.php");

}

else {

    echo "submission method not allowed";

}

$conn->close();

?>

Frontend: The page is structured using HTML5 elements like <div>, <form>, and <input>. Two main forms are present: one for signing up (<div class="form-container sign-up">) and one for signing in (<div class="form-container sign-in">). Social media icons and buttons are included for Google, Facebook, and LinkedIn authentication.

Backend: JavaScript: The script at the bottom of the HTML file waits for the DOM (Document Object Model) to be fully loaded before executing. It selects the container element and both buttons (register and login). Event listeners are added to both buttons: When the register button is clicked, the script adds the "active" class to the container, which likely triggers a CSS transition to reveal the sign-up form. When the login button is clicked, the script removes the "active" class from the container, presumably hiding the sign-up form and showing the sign-in form.

PHP and SQL: Create a user account by adding a new record in the “users” database table, creating a unique user id, and saving the added fields (username, email, password) in the record.

Users table:

A screenshot of a computer

Description automatically generated

Sign-in process:

<?php

$servername= "localhost";

$usernamee = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $usernamee, $password, $database, "3308");

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

else {

    echo "Connected successfully<br>";

}

$usernamee = "";

if ($\_SERVER["REQUEST\_METHOD"]=="POST") {

    if (isset($\_POST['username']) && isset($\_POST['password'])) {

        $user=$\_POST['username'];

        $pass=$\_POST['password'];

        $stmt = $conn->prepare("SELECT user\_id, username, password, role FROM users WHERE username= '$user' AND password= '$pass'");

        $stmt->bind\_param("ss", $user, $pass);

        $stmt->execute();

        $result = $stmt->get\_result();

        if ($result->num\_rows > 0) {

            session\_start();

            $row = $result->fetch\_assoc();

            $role = $row['role'];

            $user\_id =$row['user\_id'];

            $username = $row['username'];

            $\_SESSION['user\_id'] = $user\_id;

            $\_SESSION['username'] = $username;

            if ($role === 'admin') {

                header("Location: index2.php");

            }

            else if ($role === 'user') {

                header("Location: index.php?user\_id=" . $user\_id);

            }

            else {

                $error = "Unexpected user type";

            }

            exit();

        }

        else {

            echo "<script>alert('Invalid username or password');</script>";

        }

    }

    else {

        echo "<script>alert('login data not received');</script>";

    }

}

else {

    echo "<script>alert('submission method not allowed');</script>";

}

$conn->close();

?>

<script>

  window.location.href = "registration.php";

</script>

Identify user type (Admin/user) and display his respective dashboard.

A screenshot of a login form

Description automatically generatedJane Smith, a blogger who is already registered on our website, is recognized as a "user" type, and is taken to her designated dashboard upon login. Her user\_id is 3.

Start the session and get the id and the username of the user.

<?php

session\_start();

$user\_id = $\_SESSION['user\_id'];

$username = $\_SESSION['username'];

$servername = "localhost";

$usernamee = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $usernamee, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

$sql2 = "SELECT \* FROM posts WHERE user\_id=?";

$stmt = $conn->prepare($sql2);

$stmt->bind\_param("i", $user\_id);

$stmt->execute();

$result2 = $stmt->get\_result();

$posts = [];

while ($row = $result2->fetch\_assoc()) {

    $posts[] = $row;

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

  $title = $\_POST['title'];

  $content = $\_POST['content'];

  if (isset($\_FILES['image']) && $\_FILES['image']['error'] == 0) {

      $image = file\_get\_contents($\_FILES['image']['tmp\_name']);

      $stmt = $conn->prepare("INSERT INTO posts (user\_id, title, content, image\_url) VALUES (?, ?, ?, ?)");

      $stmt->bind\_param("isss", $user\_id, $title, $content, $image);

      $stmt->execute();

      if ($stmt->error) { die('Execute failed: ' . $stmt->error); }

      $stmt->close();

  } else {

    header("Location: index2.php"); ;

  }

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>User Dashboard</title>

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

  <style>

  </style>

</head>

<body>

  <header>

  <img src="log.jpg" alt="BlogPlatform logo">

    <h1><?php echo htmlspecialchars($username);?> Posts!</h1>

    <nav>

        <ul>

            <li><a href="index.php">Main Page</a></li>

            <li><a href="user.php">My Posts</a></li>

            <li><a href="activity.php">User Activity</a></li>

        </ul>

    </nav>

  </header>

  <?php if (isset($posts) && !empty($posts)): ?>

  <div class="main" id="main">

    <div class="posts">

        <?php foreach ($posts as $post):?>

          <div class="post-card">

            <img src="image.php?post\_id=<?= $post['post\_id'] ?>" alt="Post Image" class="post-image">

            <div class="post-info">

            <img src="w2.jpg" alt="User Avatar" class="user-avatar">

                <div class="post-meta">

                    <span class="user-name">Blogger ID: <?= htmlspecialchars($post['user\_id']) ?></span>

                    <span class="post-date">Created at: <?= htmlspecialchars($post['created\_at']) ?></span>

                </div>

            </div>

            <h2 class="post-title">Post Title: <?= htmlspecialchars($post['title']) ?></h2>

            <p class="post-excerpt"><?= htmlspecialchars($post['content']) ?></p>

            <div class="post-interactions">

                <span class="likes"><i class="fa fa-heart"></i> <?= htmlspecialchars($post['likes\_count']) ?></span>

            </div>

            <div class="post-actions">

            <button class="remove-btn"><a href="delete.php?post\_id=<?php echo $post['post\_id']; ?>">Delete</a></button>

            <button class="update-btn"><a href="updatepost.php?post\_id=<?php echo $post['post\_id']; ?>">Update</a></button>

            </div>

        </div>

        <?php endforeach; ?>

    </div>

   </div>

   <?php else: ?>

    <p>You don't have any posts yet.</p>

    <?php endif; ?>

    <div class="container">

      <h2>Create New Post</h2>

      <form id="postForm" action="user.php" method="post" enctype="multipart/form-data">

        <label for="postTitle">Title:</label>

        <input type="text" id="title" name="title">

        <label for="postContent">Content:</label>

        <textarea id="content" name="content"></textarea>

        <label for="postImage">Image:</label>

        <input type="file" id="image" name="image">

        <button type="submit">Create Post</button>

      </form>

    </div>

<footer>

    <p>©All rights reserved.</p>

</footer>

</body>

</html>

In the posts div, all posts with user\_id = 3 (jane\_smith id) will appear.

Frontend:

**Header Section:** Contains the platform's logo on the left side. Displays the user's username followed by "Posts!" indicating that it's their dashboard. Includes a navigation menu on the right side with links to the main page, user's posts, and user activity.

**Main Content:** If the user has any posts ($posts), they are displayed within the main content area. Each post is represented by a post card, which includes: An image representing the post content. User avatar and post metadata (user ID and creation date). Post title and excerpt. Post interactions like the number of likes. Post actions, including buttons to update or remove the post. Create New Post Section: Provides a form for users to create new posts. Includes fields for title, content, and image upload.

**Footer Section:** Contains copyright information indicating that all rights are reserved.

Jane Smith Dashboard:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

She has 3 created posts till now, each one is presented in one postcard.

She can create a post by adding the title, and content, and uploading an image, then clicking on Create Post.

<?php

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $username, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

session\_start();

$user\_id = $\_SESSION['user\_id'];

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

  $title = $\_POST['title'];

  $content = $\_POST['content'];

  $image\_url = $\_POST['image\_url'];

  $sql = "INSERT INTO posts (user\_id, title, content, image\_url) VALUES ('$user\_id', '$title', '$content', '$image\_url')";

  $conn->query($sql);

}

$conn->close();

?>

<?php header("Location: user.php"); ?>

Her Dashboard after adding the new post:

A screenshot of a phone

Description automatically generated

She can also update or delete a post.

Update the title of post2 to “Walking Through Fall”, and the content to “Crimson leaves carpet the path, a hallmark of autumn's arrival.”, Delete post7.

**Update post:**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $username, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

$post\_id = $\_GET['post\_id'];

if ($post\_id > 0) {

  $sql = "SELECT \* FROM posts WHERE post\_id = ?";

  $stmt = $conn->prepare($sql);

  $stmt->bind\_param("i", $post\_id);

  $stmt->execute();

  $result = $stmt->get\_result();

  if ($result->num\_rows == 1) {

    $post = $result->fetch\_assoc();

  } else {

    echo "<p style='color: red;'>Error: Post not found.</p>";

  }

  $stmt->close();

} else {

  echo "<p style='color: red;'>Error: Invalid post ID.</p>";

}

if ($\_SERVER["REQUEST\_METHOD"] == "POST" && $post\_id > 0) {

    $title = $\_POST['title'];

    $content = $\_POST['content'];

    $image\_url = $\_POST['image\_url'];

    $sql = "UPDATE posts SET title = ?, content = ?, image\_url = ? WHERE post\_id = ?";

    $stmt = $conn->prepare($sql);

    $stmt->bind\_param("ssss", $title, $content, $image\_url, $post\_id);

    $stmt->execute();

    header("Location: user.php");

}

$conn->close();

?>

<?php if (isset($post)): ?>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Update Post</title>

</head>

<body>

  <h2>Update Post</h2>

  <form action="updatepost.php?post\_id=<?php echo $post['post\_id']; ?>" method="post">

    <input type="hidden" name="post\_id" value="<?php echo $post['post\_id']; ?>">

    <label for="title">Title:</label>

    <input type="text" name="title" id="title" value="<?php echo $post['title']; ?>">

    <br>

    <label for="content">Content:</label>

    <input type="text" name="content" id="content" value="<?php echo $post['content']; ?>">

    <br>

    <label for="image\_url">Image:</label>

    <input type="file" name="image" id="image">

    <br>

    <button type="submit">Update Post</button>

  </form>

</body>

</html>

<?php endif; ?>

Frontend: Metadata in the <head> section specifying the document's character encoding, viewport settings, and title.

Visible content in the <body> section, which includes: A heading <h2> indicating the purpose of the form.

A <form> element with the action set to "updatepost.php" to handle form submission. Hidden input field to pass the post ID to the server.

Input fields for the post title, content, and image. A submit <button> to update the post.

A screenshot of a computer screen

Description automatically generated A screenshot of a computer

Description automatically generated

**Delete post:**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $username, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

$post\_id = $\_GET['post\_id'];

if ($post\_id > 0) {

  $sql = "DELETE FROM posts WHERE post\_id = '$post\_id'";

  $conn->query($sql);

}

else {

    echo "<p style='color: red;'>Error: Invalid book ID.</p>";

  }

$conn->close();

?>

Her Dashboard after modifications:

A screenshot of a phone

Description automatically generated

She can view other posts published by other bloggers on the main page:

<?php

session\_start();

$user\_id = $\_SESSION['user\_id'];

$username = $\_SESSION['username'];

$servername = "localhost";

$usernamee = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $usernamee, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

$sql2 = "SELECT \* FROM posts";

$stmt = $conn->prepare($sql2);

$stmt->execute();

$result2 = $stmt->get\_result();

$posts = [];

while ($row = $result2->fetch\_assoc()) {

    $posts[] = $row;

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>User Dashboard</title>

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

  <style>

  </style>

</head>

<body>

  <header>

  <img src="log.jpg" alt="BlogPlatform logo">

    <h1>All Posts!</h1>

    <nav>

        <ul>

            <li><a href="index.php">Main Page</a></li>

            <li><a href="user.php">My Posts</a></li>

            <li><a href="activity.php">User Activity</a></li>

        </ul>

    </nav>

  </header>

  <?php if (isset($posts) && !empty($posts)): ?>

  <div class="main" id="main">

    <div class="posts">

        <?php foreach ($posts as $post): ?>

            <div class="post-card">

            <img src="image.php?post\_id=<?= $post['post\_id'] ?>" alt="Post Image" class="post-image">

            <div class="post-info">

            <img src="w2.jpg" alt="User Avatar" class="user-avatar">

                <div class="post-meta">

                    <span class="user-name">Blogger ID: <?= htmlspecialchars($post['user\_id']) ?></span>

                    <span class="post-date">Created at: <?= htmlspecialchars($post['created\_at']) ?></span>

                </div>

            </div>

            <h2 class="post-title">Post Title: <?= htmlspecialchars($post['title']) ?></h2>

            <p class="post-excerpt"><?= htmlspecialchars($post['content']) ?></p>

            <div class="post-interactions">

                <span class="likes"><i class="fa fa-heart"></i> <?= htmlspecialchars($post['likes\_count']) ?></span>

            </div>

        </div>

        <?php endforeach; ?>

    </div>

   </div>

   <?php else: ?>

    <p>You don't have any posts yet.</p>

    <?php endif; ?>

<footer>

<p>©All rights reserved.</p>

</footer>

</body>

</html>

Frontend:

**Header Section:** Contains the platform's logo. Displays the title "All Posts!" indicating the purpose of the dashboard. Includes a navigation menu with links to the pages of this dashboard including the main page, user's posts, and user activity.

**Main Content:** Checks if there are any posts available using PHP ($posts array). If posts exist, they are displayed. Each post is represented by a post card containing: An image representing the post content, user ID, creation date, post title, excerpt, post interactions like the number of likes. If the user doesn't have any posts, a message is displayed indicating so.

**Footer Section:** Contains copyright information indicating that all rights are reserved.

A screenshot of a phone

Description automatically generated

She can also view the user activity page:

Activity page:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>User Dashboard - Blog</title>

  <link href="https://fonts.googleapis.com/css2?family=Nunito:wght@400;700&display=swap" rel="stylesheet">

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

  <style>

  </style>

</head>

<body>

  <header>

  <img src="log.jpg" alt="BlogPlatform logo">

    <h1>User Dashboard - Blog</h1>

    <nav>

      <ul>

        <li><a href="index.php">Main Page</a></li>

        <li><a href="user.php">My Posts</a></li>

        <li><a href="activity.php">User Activity</a></li>

      </ul>

    </nav>

  </header>

  <div class="chart-container">

    <canvas id="postsChart"></canvas>

  </div>

  <footer>

    <p>©All rights reserved.</p>

  </footer>

  <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

  <script>

    document.addEventListener('DOMContentLoaded', function() {

      var ctx = document.getElementById('postsChart').getContext('2d');

      var postsChart = new Chart(ctx, {

        type: 'bar',

        data: {

          labels: ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December'],

          datasets: [{

            label: 'Number of Posts',

            data: [10, 12, 8, 15, 9, 7, 14, 11, 6, 5, 8, 10],

            backgroundColor: 'rgba(54, 162, 235, 0.2)',

            borderColor: 'rgba(54, 162, 235, 1)',

            borderWidth: 1

          }]

        },

        options: {

          scales: {

            y: {

              beginAtZero: true

            }

          }

        }

      });

    });

  </script>

</body>

</html>

A screenshot of a graph

Description automatically generated

Frontend:

**Header Section:** Displays the platform's logo and the title "User Dashboard - Blog." Includes a navigation menu with links to the pages of this dashboard including the main page, user's posts, and user activity.

**Main Content:** Contains a chart container with a canvas element to render the chart. The chart is generated using Chart.js, a JavaScript library for creating charts. The chart represents the number of posts over the months of the year.

Footer Section: Contains copyright information indicating that all rights are reserved.

**Admin Dashboard:**

**A screenshot of a login page

Description automatically generated**

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Admin Dashboard</title>

  <style>

</style>

</head>

<body>

  <header>

  <img src="log.jpg" alt="BlogPlatform logo">

    <h1>Admin Dashboard</h1>

    <nav>

      <ul>

        <li><a href="index2.php">Manage Accounts</a></li>

        <li><a href="posts.php">Manage Posts</a></li>

      </ul>

    </nav>

  </header>

  <main>

    <div id="userContainer">

    </div>

    <div id="add\_form">

    <form id="addUserForm" action="add\_user.php" method="post">

    <input type="hidden" name="action" value="add">

    <h2>Add Account</h2>

    <label for="username">Username:</label>

    <input type="text" id="username" name="username" required>

    <label for="password">Password:</label>

    <input type="password" id="password" name="password" required>

    <label for="email">Email:</label>

    <input type="email" id="email" name="email" required>

    <button type="submit">Add</button>

</form>

    </div>

  </main>

  <footer>

  <p>©All rights reserved.</p>

  </footer>

  <script src="admin.js"></script>

</body>

</html>

Frontend:

**Header Section:** Displays the platform's logo and the title "Admin Dashboard." Includes a navigation menu with links to manage accounts and posts page.

**Main Content:** Contains a main section with two div elements: one for displaying user accounts and another for adding a new account. The "userContainer" div is likely intended to display existing user accounts. The "add\_form" div includes a form ("addUserForm") for adding a new user account. It consists of input fields for username, password, and email, along with a submit button.

**Footer Section:** Contains copyright information indicating that all rights are reserved. Script: Includes a JavaScript file "admin.js" to handle dynamic functionality on the admin dashboard, such as adding new user accounts.

document.addEventListener('DOMContentLoaded', function() {

    fetchUsers();

});

function fetchUsers() {

    fetch('fetch\_users.php')

    .then(response => response.json())

    .then(users => {

        const container = document.getElementById('userContainer');

        container.innerHTML = '';

        users.forEach(user => {

            container.appendChild(createUserCard(user));

        });

    })

    .catch(error => console.error('Error loading users:', error));

}

function createUserCard(user) {

    const div = document.createElement('div');

    div.className = 'user-card';

    div.id = 'user-' + user.user\_id;

    div.innerHTML = `<h3>${user.username}</h3>

        <div class="user-info">

            <span>Username:</span> ${user.username}<br>

            <span>Email:</span> ${user.email}<br>

            <span>Post Type:</span> ${user.role === 'admin' ? 'Admin' : 'User'}

        </div>

        <div class="user-actions">

            <button onclick="editUser(${user.user\_id})">Edit</button>

            <button onclick="deleteUser(${user.user\_id})">Delete</button>

        </div>`;

    return div;

}

function editUser(userId) {

    console.log('Edit User:', userId);

    window.location.href = `edit\_user.php?user\_id=${userId}`;

}

function deleteUser(userId) {

    if (confirm('Are you sure you want to delete this user?')) {

        const formData = new URLSearchParams();

        formData.append('user\_id', userId);

        fetch('delete\_user.php', {

            method: 'POST',

            headers: {

                'Content-Type': 'application/x-www-form-urlencoded',

            },

            body: formData

        })

        .then(response => response.json())

        .then(data => {

            console.log('Delete response:', data);

            if (data.success) {

                alert('User deleted successfully');

                fetchUsers();

            } else {

                alert('Error deleting user: ' + data.error);

            }

        })

        .catch(error => console.error('Error deleting user:', error));

    }

}

document.getElementById('addUserForm').addEventListener('submit', function(event) {

    event.preventDefault();

    const formData = new URLSearchParams(new FormData(this));

    fetch('add\_user.php', {

        method: 'POST',

        headers: {

            'Content-Type': 'application/x-www-form-urlencoded',

        },

        body: formData

    })

    .then(response => response.json())

    .then(data => {

        console.log('Add response:', data);

        if (data.success) {

            alert('User added successfully');

            fetchUsers();

        } else {

            alert('Error adding user: ' + data.error);

        }

    })

    .catch(error => console.error('Error adding user:', error));

});

<?php

session\_start();

header('Content-Type: application/json');

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$port = "3308";

$conn = new mysqli($servername, $username, $password, $database, $port);

if ($conn->connect\_error) {

    echo json\_encode(['error' => 'Database connection failed: ' . $conn->connect\_error]);

    exit;

}

$result = $conn->query("SELECT user\_id, username, email, role FROM users");

$users = [];

if ($result) {

    while ($user = $result->fetch\_assoc()) {

        $users[] = $user;

    }

    echo json\_encode($users);

} else {

    echo json\_encode(['error' => 'No users found']);

}

?>

**A screenshot of a computer

Description automatically generated**

The admin has all the user cards, he can edit their information, and delete them.

Edit the email of joh\_doe to [john@gmail.com](mailto:john@gmail.com)

**Edit user:**

<?php

session\_start();

$userId = isset($\_GET['user\_id']) ? intval($\_GET['user\_id']) : 0;

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$port = "3308";

$conn = new mysqli($servername, $username, $password, $database, $port);

if ($conn->connect\_error) {

    die("Connection failed: " . $conn->connect\_error);

}

$user = null;

if ($userId) {

    $stmt = $conn->prepare("SELECT user\_id, username, email FROM users WHERE user\_id = ?");

    $stmt->bind\_param("i", $userId);

    $stmt->execute();

    $result = $stmt->get\_result();

    if ($result->num\_rows == 1) {

        $user = $result->fetch\_assoc();

    } else {

        die("User not found");

    }

}

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

    $username = $\_POST['username'];

    $email = $\_POST['email'];

    $stmt = $conn->prepare("UPDATE users SET username = ?, email = ? WHERE user\_id = ?");

    $stmt->bind\_param("ssi", $username, $email, $userId);

    if ($stmt->execute()) {

        echo "User updated successfully.";

        header("Location: index2.php");

        exit();

    } else {

        echo "Error updating user.";

    }

}

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Edit User</title>

</head>

<body>

    <h1>Edit User</h1>

    <?php if ($user): ?>

        <form method="post">

            <label for="username">Username:</label>

            <input type="text" id="username" name="username" value="<?= htmlspecialchars($user['username']) ?>" required>

            <label for="email">Email:</label>

            <input type="email" id="email" name="email" value="<?= htmlspecialchars($user['email']) ?>" required>

            <label for="role">Role:</label>

            <button type="submit">Update</button>

        </form>

    <?php else: ?>

        <p>User not found.</p>

    <?php endif; ?>

</body>

</html>

**Title:** Specifies the title of the page as "Edit User". Main Content: Heading: Displays "Edit User" as the main heading.

**PHP Condition:** Checks if the $user variable is set. If $user is set, it displays a form for editing user information.

**Input Fields:** Includes fields for editing the username and email address. The values of these fields are pre-filled with the user's current username and email, respectively.

**Role:** The role field is missing, possibly intended to be added later.

**Submit Button:** Allows the user to submit the updated information. If $user is not set, it displays a message indicating that the user was not found.

The page structure is simple, focusing primarily on editing user information and providing feedback if the user is not found.

A close-up of a email

Description automatically generated

**A close-up of a email

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

Also he can add users by filling in information in the add account section.

<?php

session\_start();

header('Content-Type: application/json');

$servername = "localhost";

$username = "root";

$password = "";

$database = "blog";

$port = "3308";

$conn = new mysqli($servername, $username, $password, $database, $port);

if ($conn->connect\_error) {

    echo json\_encode(['error' => 'Database connection failed: ' . $conn->connect\_error]);

    exit;

}

$username = $\_POST['username'] ?? '';

$password = $\_POST['password'] ?? '';

$email = $\_POST['email'] ?? '';

$role = 'user';

if (!$username || !$password || !$email) {

    echo json\_encode(['error' => 'All fields are required.']);

    exit;

}

$passwordHash = password\_hash($password, PASSWORD\_DEFAULT);

$stmt = $conn->prepare("INSERT INTO users (username, password, email, role) VALUES (?, ?, ?, ?)");

$stmt->bind\_param("ssss", $username, $passwordHash, $email, $role);

if ($stmt->execute()) {

    echo json\_encode(['success' => 'User added successfully']);

} else {

    echo json\_encode(['error' => 'Error adding user: ' . $stmt->error]);

}

?>

A screenshot of a login form

Description automatically generated

A screenshot of a computer

Description automatically generated

He can also delete the posts of the bloggers by navigating to Manage Posts and deleting the desired post.

<?php

session\_start();

$user\_id = $\_SESSION['user\_id'];

$username = $\_SESSION['username'];

$servername = "localhost";

$usernamee = "root";

$password = "";

$database = "blog";

$conn = new mysqli($servername, $usernamee, $password, $database, "3308");

if ($conn->connect\_error) {

  die("Connection failed: " . $conn->connect\_error);

}

$sql2 = "SELECT \* FROM posts";

$stmt = $conn->prepare($sql2);

$stmt->execute();

$result2 = $stmt->get\_result();

$posts = [];

while ($row = $result2->fetch\_assoc()) {

    $posts[] = $row;

}

$conn->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Admin Dashboard</title>

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.0.0-beta3/css/all.min.css">

  <style>

  </style>

</head>

<body>

  <header>

  <img src="log.jpg" alt="BlogPlatform logo">

    <h1>Admin Dashboard</h1>

    <nav>

        <ul>

        <li><a href="index2.php">Manage Accounts</a></li>

        <li><a href="posts.php">Manage Posts</a></li>

        </ul>

    </nav>

  </header>

  <?php if (isset($posts) && !empty($posts)): ?>

  <div class="main" id="main">

    <div class="posts">

        <?php foreach ($posts as $post): ?>

            <div class="post-card">

            <img src="image.php?post\_id=<?= $post['post\_id'] ?>" alt="Post Image" class="post-image">

            <div class="post-info">

            <img src="w2.jpg" alt="User Avatar" class="user-avatar">

                <div class="post-meta">

                    <span class="user-name">Blogger ID: <?= htmlspecialchars($post['user\_id']) ?></span>

                    <span class="post-date">Created at: <?= htmlspecialchars($post['created\_at']) ?></span>

                </div>

            </div>

            <h2 class="post-title">Post Title: <?= htmlspecialchars($post['title']) ?></h2>

            <p class="post-excerpt"><?= htmlspecialchars($post['content']) ?></p>

            <div class="post-interactions">

                <span class="likes"><i class="fa fa-heart"></i> <?= htmlspecialchars($post['likes\_count']) ?></span>

            </div>

            <div class="post-actions">

            <button class="remove-btn"><a href="delete.php?post\_id=<?php echo $post['post\_id']; ?>">Delete</a></button>

            </div>

        </div>

        <?php endforeach; ?>

    </div>

   </div>

   <?php else: ?>

    <p>You don't have any posts yet.</p>

    <?php endif; ?>

<footer>

<p>©All rights reserved.</p>

</footer>

</body>

</html>

Frontend:

**Header Section:** Displays the platform's logo and the title "Admin Dashboard." Includes a navigation menu with links to manage accounts and posts pages.

**Main Content:** Checks if there are any posts available using PHP ($posts array). If posts exist, they are displayed. Each post is represented by a post card containing: An image representing the post content, user ID, creation date, post title, excerpt, and interactions (likes count). Provides a button to delete the post.

**Footer Section:** Contains copyright information indicating that all rights are reserved.

**Delete post:**

<?php

session\_start();

header('Content-Type: application/json');

$userId = isset($\_POST['user\_id']) ? intval($\_POST['user\_id']) : 0;

$conn = new mysqli("localhost", "root", "", "blog", "3308");

if ($conn->connect\_error) {

    echo json\_encode(['error' => 'Database connection failed: ' . $conn->connect\_error]);

    exit;

}

if ($userId > 0) {

    $stmt = $conn->prepare("DELETE FROM users WHERE user\_id = ?");

    $stmt->bind\_param("i", $userId);

    if ($stmt->execute()) {

        echo json\_encode(['success' => 'User deleted successfully']);

    } else {

        echo json\_encode(['error' => 'Error deleting user: ' . $stmt->error]);

    }

} else {

    echo json\_encode(['error' => 'Invalid user ID']);

}

?>

A screenshot of a computer

Description automatically generated

**Summary:**

All the provided HTML templates follow user-friendly and responsive CSS styles, ensuring a seamless experience across different devices and screen sizes. The designs prioritize readability, clarity, and ease of navigation, making them accessible to users of all levels. Each template incorporates responsive design principles, allowing content to adapt and display optimally on various devices, from large desktop screens to smaller mobile devices. Additionally, consistent branding elements such as logos and color schemes contribute to a cohesive and professional appearance throughout the interfaces. Overall, the combination of user-friendly HTML structures and responsive CSS styles ensures an enjoyable and accessible experience for users interacting with the respective dashboards and interfaces.

Also, PHP for server-side processing, SQL for database interactions, and JavaScript for client-side interactivity, ensure smooth functionality and dynamic content delivery.